

Dracula Reference

**Product Version IC23.1
September 2023**

© 2023 Cadence Design Systems, Inc. All rights reserved.
Printed in the United States of America.

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 800.862.4522.

All other trademarks are the property of their respective holders.

Restricted Permission: This publication is protected by copyright law and international treaties and contains trade secrets and proprietary information owned by Cadence. Unauthorized reproduction or distribution of this publication, or any portion of it, may result in civil and criminal penalties. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. Unless otherwise agreed to by Cadence in writing, this statement grants Cadence customers permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used only in accordance with a written agreement between Cadence and its customer.
2. The publication may not be modified in any way.
3. Any authorized copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement.
4. The information contained in this document cannot be used in the development of like products or software, whether for internal or external use, and shall not be used for the benefit of any other party, whether or not for consideration.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information. Cadence is committed to use respectful language in our code and communications. We are also active in the removal and/or replacement of inappropriate language from existing content. This product documentation may however contain material that is no longer considered appropriate but still reflects long-standing industry terminology. Such content will be addressed at a time when the related software can be updated without end-user impact.

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor

Contents

<u>Preface</u>	25
<u>Related Documents</u>	25
<u>Typographic and Syntax Conventions</u>	26
<u>1</u>	
<u>Overview</u>	29
<u>Introducing Dracula</u>	29
<u>Flat and Hierarchical Databases</u>	30
<u>Dracula Applications</u>	31
<u>Dracula Interfaces</u>	33
<u>Introducing Hierarchical Dracula</u>	34
<u>Hierarchical Structure</u>	35
<u>Functions Not Available in Hierarchical Rules Files</u>	37
<u>Hierarchical Dracula Modes</u>	37
<u>2</u>	
<u>Using Dracula</u>	39
<u>Verifying a Design with Dracula</u>	39
<u>Creating a Rules File</u>	40
<u>Rules File Structure</u>	40
<u>Syntax</u>	43
<u>Rules File Example</u>	44
<u>Compiling the Rules File</u>	45
<u>Running PDRACULA</u>	47
<u>Using PDRACULA Commands</u>	47
<u>/ABORT</u>	48
<u>/CLEAR</u>	49
<u>/DCL</u>	50
<u>/FINISH</u>	51
<u>/GET</u>	53

Dracula Reference

<u>/HELP</u>	55
<u>/LIST</u>	57
<u>/NEXT</u>	58
<u>/PRINT</u>	59
<u>/RESTART</u>	60
<u>/STATUS</u>	61
<u>/STORE</u>	62
<u>Running Dracula</u>	62
<u>Interpreting Dracula Output</u>	63
<u>Dracula Distributed Processing</u>	65
<u>How Dracula Distributed Works</u>	66
<u>Setting Up for Dracula Distributed Processing</u>	66
<u>Creating a Node Configuration File</u>	68
<u>Running Dracula Distributed Processing</u>	70
<u>Distributed Dracula Files</u>	71
<u>Dracula Distributed Processing and the Dracula Graphical User Interface Tool</u>	75
<u>Dracula Distributed Processing in the Dracula 4.51 Release</u>	75
<u>Dracula Distributed Processing in the Dracula 4.6 release</u>	75
<u>Dracula Distributed Processing in the Dracula 4.7 release</u>	76
<u>Dracula Distributed Processing in the Dracula 4.8 release</u>	76
<u>Running Dracula in 32-bit or 64-bit Modes</u>	77

3

<u>Checking Design Rules (DRC)</u>	79
<u>Design Rules Checker</u>	79
<u>DRC Commands</u>	79
<u>Conjunctive Rules</u>	81
<u>DRC Example</u>	81
<u>Description Block</u>	82
<u>Input-Layer Block</u>	82
<u>Operation Block</u>	83
<u>Executing DRC from a Design Rule Library</u>	91
<u>Developing and Executing a Design Rule</u>	95
<u>Viewing the Results</u>	96
<u>Viewing the Violations</u>	98

Dracula Reference

<u>Database Comparison</u>	109
<u>Hierarchical and Multilevel Hierarchical Design Rules Checker</u>	110
<u>Prerequisites</u>	110
<u>Selecting HDRC Hcells</u>	111
<u>Creating Hcells, Hcell Environment, and Composite Plane</u>	112
<u>HDRC Modes</u>	114
<u>HDRC Error Reports</u>	118
<u>Sample HDRC Rules File</u>	127

4

<u>Checking Electrical Rules (ERC)</u>	129
<u>Electrical Rules Checker</u>	129
<u>ERC Commands</u>	130
<u>ERC Output</u>	133
<u>ERC Examples</u>	135
<u>Description Block</u>	136
<u>Input Layer Block</u>	136
<u>Operation Block</u>	137
<u>NMOS ERC Rules File</u>	138
<u>CMOS ERC Rules File</u>	141
<u>Hierarchical ERC</u>	143
<u>HERC Check Modes</u>	144
<u>HERC Limitations</u>	144
<u>HERC Commands</u>	145

5

<u>Compiling Network Descriptions (LOGLVS)</u>	147
<u>Overview</u>	147
<u>Initial Correspondence Nodes</u>	149
<u>Node Names</u>	150
<u>Preparing Netlists</u>	151
<u>LOGLVS Command Sequence</u>	153
<u>Control Commands</u>	153
<u>Parse Commands</u>	155
<u>Expand or Flatten Commands</u>	155

Dracula Reference

<u>LOGLVS Examples</u>	155
<u>CDL Control Commands</u>	156
*.BIPOLAR	157
*.BUSDELIMITER	158
*.CAPA	159
*.CAPAREA	161
*.CAPVAL	162
*.DEFAULT	163
*.DIOAREA	164
*.DIOPERI	165
*.DIODE	166
*.EQUATION	167
*.EQUIV	168
*.GNONSWAP	172
*.LDD	173
*.MEGA	174
*.NONSWAP	175
*.NOPIN	177
*.NOSUB M	178
*.PIN	179
*.PININFO	180
*.RESI	182
*.RESSIZE	184
*.RESVAL	185
*.REVERSE	186
*.SCALE	187
*.SPICE	189
*.UNSPEC	190
*.UNSPEC-MOS	192
<u>SPICE and HSPICE Commands</u>	193
<u>.ENDS or .EOM</u>	194
<u>.GLOBAL</u>	195
<u>.INCLUDE</u>	196
<u>.PARAM</u>	197
<u>.SUBCKT or .MACRO</u>	200
<u>.SWAP and Special Elements</u>	202

Dracula Reference

<u>Subcircuit Calls</u>	204
<u>CDL Element Definition</u>	206
<u>Capacitor Syntax</u>	206
<u>Diode Syntax</u>	207
<u>Resistor Syntax</u>	208
<u>Transistor (BJT) Syntax</u>	209
<u>Transistor (JFET) Syntax</u>	211
<u>Transistor (MOSFET) Syntax</u>	212
<u>CDL Macro Expansion</u>	213
<u>CDL File Example</u>	215
<u>Preparing Data Files</u>	216
<u>Preparing SPICE Files</u>	216
<u>Preparing Verilog Files</u>	219
<u>Running LOGLVS</u>	221
<u>CASE</u>	223
<u>CELL</u>	224
<u>CIRCUIT</u>	226
<u>CONVERT</u>	227
<u>DATAFORMAT</u>	228
<u>DELCKT</u>	229
<u>DELECEL</u>	230
<u>DEVMAP</u>	231
<u>DXF</u>	233
<u>ENV</u>	234
<u>EQUIV</u>	235
<u>EXIT</u>	236
<u>FDELIMITER</u>	237
<u>FPIN</u>	238
<u>GENPAD</u>	239
<u>GET</u>	240
<u>GLOBAL</u>	241
<u>HELP</u>	242
<u>HTV/DRE</u>	243
<u>KCELL</u>	244
<u>KFPIN</u>	245
<u>LINK</u>	246

Dracula Reference

<u>MOS EXTENDED REP</u>	247
<u>NO WARNING</u>	248
<u>NVER</u>	249
<u>PRECISION</u>	250
<u>PRINT</u>	251
<u>RESET</u>	252
<u>RESISTOR</u>	253
<u>SAVE</u>	254
<u>SET FANIN</u>	255
<u>SET FANOUT</u>	256
<u>STOP</u>	257
<u>STORE</u>	258
<u>SUMMARY</u>	259
<u>TNAME</u>	260
<u>TRANSISTOR</u>	261
<u>TYPE</u>	262
<u>UNLIMIT</u>	263
<u>Shell Environment Variables Used By LOGLVS</u>	264
<u>LOGLVS Example</u>	266
<u>Using Hierarchical LOGLVS</u>	267
<u>CDL and SPICE Formats</u>	268
<u>Hcell Definition Examples</u>	270
<u>LOGLVS Examples</u>	271
<u>SPICE</u>	274

6

<u>Comparing Layouts and Schematics (LVS)</u>	275
<u>LVS Overview</u>	275
<u>Layout Versus Layout</u>	276
<u>Schematic Versus Schematic</u>	276
<u>Hierarchical Layout Versus Schematic</u>	277
<u>Initial Correspondence Node Pairs</u>	277
<u>Trace and Discrepancy Points</u>	277
<u>Using LVS</u>	278
<u>LVL Overview</u>	281

Dracula Reference

<u>Preparing for LVL</u>	282
<u>Rules Files</u>	283
<u>Running LVL</u>	285
<u>SVS Overview</u>	285
<u>Running SVS in Flat Mode</u>	286
<u>Running SVS in Cell or Composite Mode</u>	286
<u>Running SVS in Hierarchical Mode</u>	286
<u>LVS Report Format</u>	287
<u>Sample LVS Report</u>	289
<u>Discrepancy Report Conventions</u>	293
<u>LVS Error Types</u>	295
<u>LVS Error Type 1</u>	297
<u>LVS Error Type 2</u>	298
<u>LVS Error Type 3</u>	298
<u>LVS Error Type 4</u>	299
<u>LVS Error Type 5</u>	299
<u>LVS Error Type 6</u>	300
<u>LVS Error Type 7</u>	300
<u>LVS Error Type 8</u>	300
<u>LVS Error Type 9</u>	301
<u>LVS Error Type 10</u>	301
<u>LVS Error Type 11</u>	303
<u>LVS Error Type 12</u>	303
<u>LVS Error Type 13</u>	304
<u>LVS Error Type 14</u>	304
<u>LVS Error Type 15</u>	306
<u>Device Recognition in LVS</u>	306
<u>NMOS Inverter</u>	306
<u>NMOS NOR Gate</u>	307
<u>NMOS NAND Gate</u>	308
<u>CMOS Inverter</u>	309
<u>Simple Parallel Pull-Up (PUP) Structure</u>	309
<u>Simple Series Pull-Up (SUP) Structure</u>	310
<u>Simple Parallel Pull-Down (PDW) Structure</u>	310
<u>Simple Series Pull-Down (SDW) Structure</u>	311
<u>Complex Gate Including PUPI, SUPI, PDWI, and SDWI</u>	312

Dracula Reference

<u>PUPI and SDWI</u>	313
<u>SUPI and PDWI</u>	315
<u>PUPI and SDWI</u>	316
<u>PUPI and SDWI</u>	318
<u>SUPI and SDWI</u>	319
<u>CMOS NAND</u>	321
<u>CMOS NOR</u>	321
<u>CMOS AOI</u>	322
<u>CMOS AOIs</u>	323
<u>Simple Parallel Middle (PMID) Structure</u>	323
<u>Simple Series Middle (SMID) Structure</u>	324
<u>Undefined Elements</u>	324
<u>Isolating and Debugging Errors</u>	324
<u>Interpreting an LVS Discrepancy</u>	325
<u>Logical Equivalence in MOS Digital Circuits</u>	327
<u>Calculations on Parallel MOS Devices for LVS</u>	328
<u>W/L Calculation</u>	328
<u>W_{eff} MULDELW Calculation</u>	329
<u>MOS Devices Without Substrate Terminal</u>	331
<u>Calculations on Parallel and Series Resistor Devices width/length for LVS</u>	332
<u>Bipolar LVS Example</u>	333
<u>Hierarchical LVS</u>	337
<u>Basic Requirements</u>	338
<u>Selecting HLVS Hcells</u>	339
<u>Automatically Selecting HLVS Hcells</u>	340
<u>Hcell Pins</u>	340
<u>Hcell Text (Pin Names)</u>	341
<u>Adding to Text</u>	342
<u>Pin Text</u>	342
<u>Masking Hcells with Black Boxes</u>	343
<u>Hcells and the BASE-LAYER</u>	344
<u>Hcell Frames</u>	346
<u>Wire Types</u>	346
<u>Feedthroughs</u>	347
<u>GEN-TEXT Commands</u>	348
<u>HLVS Error Types</u>	350

Dracula Reference

<u>Sample HLVS Input Rules File</u>	350
<u>Inherited Connections for LVS in Dracula Verification</u>	351
<u>Example</u>	352

7

Extracting Electrical Parameters (LPE)

<u>LPE Overview</u>	355
<u>LPE Flow Diagram</u>	357
<u>LPE Flow Diagram (continued)</u>	358
<u>Extracting SPICE Parameters</u>	358
<u>MOS Transistors</u>	358
<u>Parasitic Diode Parameters</u>	362
<u>Parasitic Capacitor Parameters</u>	363
<u>Element Resistor Parameters</u>	364
<u>Bipolar Transistor Area and Perimeter</u>	369
<u>Flexible LPE</u>	372
<u>Geometric Primitives</u>	373
<u>Reserved Parameter Keywords</u>	374
<u>User-Defined Parameters</u>	374
<u>Selective LPE</u>	388
<u>Selective LPE Batch Example</u>	389
<u>Selective LPE Incremental Example</u>	389
<u>LPE Examples</u>	390
<u>Features of the Example</u>	391
<u>Rules File in a MOS Example</u>	392
<u>NMOS LPE Input File</u>	398
<u>CMOS LPE Input File</u>	399
<u>Hierarchical LPE</u>	401
<u>Cell Mode</u>	403
<u>Composite Mode</u>	403

8

Using Parametric Data (LPEPRO)

<u>LPEPRO Overview</u>	405
<u>Using LPEPRO</u>	406

Dracula Reference

<u>Terminal Commands</u>	407
<u>Administrative Commands</u>	408
<u>DISK</u>	409
<u>INPUT</u>	410
<u>RESET</u>	411
<u>SAVE</u>	412
<u>EXIT</u>	413
<u>Network and Gate Commands</u>	413
<u>GET</u>	414
<u>DELAY</u>	416
<u>Delay Value Command</u>	416
<u>CALCULATE</u>	417
<u>Path Analysis Commands</u>	418
<u>LPATH</u>	419
<u>SPATH</u>	420
<u>PATH</u>	421
<u>START</u>	422
<u>END</u>	423
<u>MASK</u>	424
<u>UNMASK</u>	425
<u>Report Commands</u>	425
<u>PRINT, TYPE, and STORE</u>	426
<u>Scanning Layout Parameters</u>	433
<u>Identifying Nodes in the Layout</u>	433
<u>MOS Subcommand</u>	434
<u>CAPACITANCE Subcommand</u>	437
<u>DIODE Subcommand</u>	438
<u>PARAMETER Subcommand</u>	438
<u>SPICE Generator</u>	439
<u>Whole Chip or Whole Cell</u>	439
<u>Selected Paths</u>	441
<u>Logic Gate Generator</u>	443
<u>Using the Logic Gate Generator</u>	443
<u>Gate Types</u>	444
<u>Wired-OR Examples</u>	460
<u>Calculating Delay</u>	461

Dracula Reference

<u>L/W Ratio, Capacitance, and Resistance</u>	462
<u>Using Delay Equation Options</u>	462
<u>Revising Schematic Data</u>	464
<u>Calculating Delays for Nonprimitive Gates</u>	466
<u>Reviewing Delay Information</u>	468

9

<u>Extracting Parasitic Resistance (PRE)</u>	471
<u>Overview of Parasitic Resistance Extraction</u>	471
<u>First PRE Pass</u>	472
<u>Second PRE Pass</u>	474
<u>Creating the Resistor and Resistor Terminal Layers</u>	478
<u>Using the CUT-TERM Command</u>	478
<u>Using a Command Sequence</u>	479
<u>Creating the Pad Terminal Layers</u>	479
<u>Examples</u>	481
<u>Error Conditions</u>	481
<u>Calculating Junction Resistance</u>	482
<u>Redistributing Parasitic Capacitances</u>	483
<u>Sample PRE Rules File</u>	485
<u>Considerations for Parasitic Resistance Extraction</u>	490
<u>Sample Data and Results</u>	492
<u>Hierarchical PRE</u>	495
<u>SPF File Overview</u>	495
<u>DSPF Format</u>	496
<u>RSPF Format</u>	501
<u>Dracula, CDC, DSPF and RSPF Flow</u>	506

10

<u>Using DRAC2CDL Netlist Generator</u>	507
<u>Overview</u>	507
<u>Running DRAC2CDL</u>	507
<u>Interactive Mode</u>	507
<u>Batch Mode</u>	509
<u>Examples</u>	509

11

<u>Description Block Commands</u>	513
<u>ABORT-MULT-STAMP</u>	514
<u>ABORT-P-G-SHORT</u>	516
<u>ABORT-SOFTCHK</u>	517
<u>ACUTE-CLIP</u>	518
<u>ALLOW-RECONNECT</u>	520
<u>ANGLED-EDGE-TOL</u>	521
<u>ARRAY</u>	522
<u>ARRAY-ENABLED</u>	524
<u>AUTOMATCH</u>	525
<u>BLACKBOX-FILE</u>	526
<u>BLOCK-NAME-ONLY</u>	529
<u>BOUNDING-BOX</u>	530
<u>BOXMAP</u>	531
<u>BOX-M-COMPARE</u>	533
<u>BOX-M-FACTOR</u>	534
<u>BOX-M-IGNORE</u>	535
<u>CAP-POLARITY</u>	536
<u>CARE-SPLIT-ORDER</u>	537
<u>CELLBOX-LAYER</u>	539
<u>CELL-CHILD-TEXT</u>	540
<u>CELL-ERROR-REP</u>	541
<u>CELL-LIBRARY</u>	544
<u>CHECK-MODE</u>	545
<u>CHECK-PATH</u>	548
<u>CMOS-NTYPE</u>	549
<u>CMOS-PTYPE</u>	550
<u>CNAMES-CHK-OFF</u>	551
<u>CNAMES-CSEN</u>	552
<u>CONN-ALL-WIRE</u>	554
<u>CONN-INTER-WIRE</u>	555
<u>CONN-OPEN-WIRE</u>	556
<u>CONVERT-DATABASE</u>	557
<u>CPOINT-FILE</u>	559

Dracula Reference

<u>CSH-F-OPTION</u>	561
<u>DATAFORMAT</u>	562
<u>DD-RSH-COMMAND</u>	563
<u>DELCEL</u>	564
<u>DEL-VIR-WIRE</u>	565
<u>DIODE-P-TO-G</u>	566
<u>DIODESEQ</u>	567
<u>DRACBATCH-EMAIL</u>	568
<u>EBOX-SHORT-PIN</u>	569
<u>EMPTY-BLACKBOX</u>	570
<u>EMPTY-ENC</u>	571
<u>END-MACRO</u>	572
<u>ENVIRONMENT-MAX</u>	573
<u>ERROR-PATH-WIDTH</u>	575
<u>EXPAND-GATE-DELI</u>	576
<u>EXPAND-INST-DELI</u>	577
<u>EXPAND-NODE-DELI</u>	578
<u>EXTERM-NOWARN</u>	579
<u>FASTSIZE</u>	580
<u>FDELIMITER</u>	581
<u>FILTER Commands</u>	582
<u>FILTER-LDD</u>	594
<u>FILTER-REPORT</u>	595
<u>FIX-INPUT-ORDER</u>	596
<u>FLAG-ACUTEANGLE</u>	597
<u>FLAG-NON45</u>	598
<u>FLG-EXPTH-OFFGRD</u>	599
<u>FLAG-OFFGRID</u>	601
<u>FLAG-PTH-OFFGRID</u>	603
<u>FLAG-SELFINTERS</u>	604
<u>FLAG-SELFTOUCH</u>	606
<u>FLAT-GDS-SMALL</u>	608
<u>FLATTEN-PWRGND</u>	609
<u>FNODE-CONNECT</u>	610
<u>GATE-RES-EXT</u>	612
<u>GEN-TEXT-FILE</u>	614

Dracula Reference

<u>GEN-TEXT-FLTNODE</u>	615
<u>GEN-TEXT-FRAME</u>	617
<u>GEN-TEXT-WIRE</u>	619
<u>GEN-XRF-RPT</u>	620
<u>GROUND-NODE</u>	621
<u>HCELL</u>	622
<u>HCELL-COLUMN-1</u>	625
<u>HCELL-FILE</u>	626
<u>HCELL-IN-HCELL</u>	627
<u>HCELL-MAX-PLACEMENTS</u>	629
<u>HCELL-MAX-SEGMENTS</u>	630
<u>HCELL-OPTIMIZE-PLACEMENTS</u>	631
<u>HCELL-RULE</u>	632
<u>HIER-OUTDISK</u>	634
<u>IGNORE-SCH-SUB</u>	635
<u>INAME-TOPDOWN</u>	636
<u>INDISK</u>	637
<u>INDISK-FILE</u>	638
<u>INSTPIN-FILE</u>	641
<u>INTERNAL-NODE</u>	643
<u>INTERNODE-PREFIX</u>	644
<u>KCELL-FILE</u>	645
<u>KEEPDATA</u>	646
<u>KEEP-INST-FPIN</u>	648
<u>KEEP-SHORT-BJT</u>	649
<u>KEEP-SHORT-CAP</u>	650
<u>KEEP-SHORT-DIO</u>	651
<u>KEEP-SHORT-MOS</u>	652
<u>KEEP-SHORT-RES</u>	653
<u>KEEP-TEXT</u>	654
<u>LAYER-FILE</u>	655
<u>LAYER-FILE-TYPE</u>	656
<u>LEGAL-CHAR-SET</u>	657
<u>LIBRARY</u>	658
<u>LIMIT-DRC-ERROR</u>	659
<u>LIMIT-RUN-TO</u>	660

Dracula Reference

<u>LISTERROR</u>	663
<u>LONG-SUM-LEN</u>	665
<u>LOOP-TWO-CAP</u>	666
<u>LOOP-TWO-RES</u>	667
<u>LPE-FORMAT</u>	668
<u>LPE-GROUND</u>	669
<u>LPE-PRECISION</u>	670
<u>LPE-QUERY</u>	671
<u>LUMPCAP</u>	673
<u>LVL-PART</u>	675
<u>LVS-OPTION</u>	676
<u>LVS-RPT-ONLY</u>	678
<u>M-Factor Commands</u>	679
<u>M-Factor Tolerance Commands</u>	681
<u>MAG-BEF-GRID</u>	684
<u>MAGNIFY-IN</u>	685
<u>MAGNIFY-OUT</u>	686
<u>MAP-TEXT-FILE</u>	687
<u>MATCH-P-G-BY-TYP</u>	688
<u>MEMORY-CORE</u>	689
<u>MINI-SUMMARY</u>	692
<u>MIRROR</u>	693
<u>MODEL</u>	694
<u>MOS-CAP</u>	695
<u>MOS-M-COMPARE</u>	696
<u>MULTICPU</u>	697
<u>MULTI-GDS2-IN</u>	698
<u>MULTI-PIN-FILE</u>	700
<u>MULTIPLE-DISK</u>	702
<u>NO-SUB-TERM</u>	704
<u>NOT-HCELL</u>	705
<u>NRX-EXTRACT</u>	706
<u>NRX-SAVE</u>	713
<u>OAOUTTYPE</u>	714
<u>OPTIMIZE-RULE</u>	715
<u>OUTCELL</u>	716

Dracula Reference

<u>OUTDISK</u>	717
<u>OUTLIB</u>	718
<u>OUTPUT-ONE-LAYER</u>	719
<u>OVPR-TOLERANCE</u>	720
<u>PARALLEL-FILE</u>	721
<u>PAR-COMPARE</u>	723
<u>PAROUTNAM</u>	725
<u>PARSET</u>	731
<u>PATH-W-OFFGRID</u>	735
<u>PATH-W-ROUND</u>	737
<u>PINCAP-FILE</u>	739
<u>POINT-CONNECT</u>	740
<u>POWER-NODE</u>	741
<u>PREFIX-PARASITIC</u>	742
<u>PREINTERACTIVE</u>	743
<u>PRIMARY</u>	745
<u>PRINTFILE</u>	746
<u>PROB-GEOM-LIST</u>	747
<u>PROCESS-VAR</u>	748
<u>PROGRAM-DIR</u>	750
<u>PULL-DOWN</u>	751
<u>PULL-UP</u>	752
<u>RAM-CELL</u>	753
<u>RAM-CORE</u>	754
<u>RCX-MODE</u>	755
<u>REDUCER</u>	756
<u>REP-UNM-INST-PG</u>	758
<u>RERUN-FILTER</u>	759
<u>RESOLUTION</u>	760
<u>RESWSMASH</u>	762
<u>ROTATION</u>	763
<u>ROUND-OFF-TOLER</u>	764
<u>SCALE and REV-SCALE</u>	765
<u>SCALEOUT and REV-SCALEOUT</u>	766
<u>SCHEMATIC</u>	767
<u>SELECT-MODE</u>	768

Dracula Reference

<u>SIZE-MAX-VERTEX</u>	769
<u>SIZE-MIN-WIDTH</u>	770
<u>SIZENOT</u>	772
<u>SKILL-LOADER</u>	773
<u>SMART-LPE</u>	774
<u>SMASH-CAP-TYP</u>	779
<u>SMASH-FLOAT-RES</u>	780
<u>SMASH-MOS-TYP</u>	781
<u>SMASH-RES-TYP</u>	785
<u>SMOOTH-SIZE</u>	786
<u>S-OPTION-CLASS</u>	788
<u>SPF-MAP-FILE</u>	789
<u>START-MACRO</u>	791
<u>STATUS-COMMAND</u>	792
<u>SUBCKT-NAME</u>	793
<u>SUBNODE-DELIM</u>	794
<u>SUB-PROGRAM-NAME</u>	795
<u>SUBTYPE-CSENS</u>	797
<u>SUMMARY-ONLY</u>	798
<u>SVS-LAYOUT</u>	799
<u>SVS-SCHEMATIC</u>	800
<u>SYSOUT</u>	801
<u>SYSTEM or SYSIN</u>	803
<u>TEXT-HEIGHT</u>	805
<u>TEXT-LEVEL</u>	806
<u>TEXT-PRI-ONLY</u>	807
<u>TNAMES-CSEN</u>	808
<u>TRANSISTOR</u>	809
<u>UNIT</u>	810
<u>UNSPEC-LAY-PARA</u>	811
<u>UNSPEC-PARA</u>	812
<u>UNSPEC-SCH-PARA</u>	814
<u>WINDEL</u>	815
<u>WINDOW</u>	817
<u>WINDOW-CUT</u>	819
<u>WORK-DIR</u>	820

<u>ZERO-SPAC-F-EQU</u>	822

12

Input-Layer Block Commands

<u>Introduction</u>	825
<u>Using Text</u>	825
<u>BASE-LAYER</u>	833
<u>CELLBNDY</u>	835
<u>CONNECT-LAYER</u>	838
<u>CTEXT and ATTACH</u>	841
<u>EXPORT</u>	843
<u>FRAME-EXCLUDE-LAYER</u>	845
<u>FRAME-HOLE-LAYER</u>	846
<u>GEN-TEXT-LAYER</u>	847
<u>IDTEXT</u>	848
<u>IMPORT-LAYERS</u>	849
<u>KTEXT</u>	850
<u>Layer Name Definition</u>	851
<u>OUT-ANNOTATED-LAYER</u>	855
<u>PAD-LAYER</u>	856
<u>PIN-TEXT-LAYER</u>	859
<u>RCONNECT-LAYER</u>	861
<u>RCX-LAYERS</u>	862
<u>SUBSTRATE</u>	863
<u>TEMPORARY-LAYER</u>	865
<u>TEXTSEQUENCE</u>	867

13

Operation Block Commands

<u>Introduction</u>	869
<u>Logical Operations</u>	869
<u>Electrical Node Extraction</u>	870
<u>Sizing Operations</u>	870
<u>Spacing Checks</u>	870
<u>Circuit Element Extraction</u>	871

Dracula Reference

<u>Electrical Rule Checking</u>	871
<u>Layout Versus Schematic</u>	871
<u>Layout Parameter Extraction</u>	872
<u>AND</u>	873
<u>ANDNOT</u>	875
<u>ANGLED-EDGE-TOL</u>	876
<u>AREA</u>	878
<u>ATTACH</u>	881
<u>ATTRIBUTE CAP</u>	884
<u>ATTRIBUTE RES</u>	891
<u>BBOX</u>	893
<u>*BREAK</u>	900
<u>CALCULATE</u>	902
<u>CAT</u>	905
<u>CHKPAR</u>	907
<u>COEFFICIENT CAP</u>	910
<u>COMPUTE</u>	911
<u>CONNECT</u>	913
<u>CORNER</u>	915
<u>COVERAGE</u>	920
<u>CUT</u>	928
<u>CUT-TERM</u>	930
<u>DENC</u>	936
<u>DEVTAG</u>	946
<u>drcAntenna</u>	951
<u>ECONNECT</u>	962
<u>EDGECHK</u>	965
<u>EDTEXT</u>	966
<u>ELCOUNT</u>	968
<u>ELEMENT BJT</u>	971
<u>ELEMENT BOX</u>	973
<u>ELEMENT CAP</u>	975
<u>ELEMENT DIO</u>	977
<u>ELEMENT IN?</u>	979
<u>ELEMENT LDD</u>	981
<u>ELEMENT MOS</u>	983

Dracula Reference

<u>ELEMENT PAD</u>	985
<u>ELEMENT RES</u>	986
<u>ENC</u>	988
<u>ENCBASEDOVLP</u>	1000
<u>ENCRECT</u>	1007
<u>EQUATION</u>	1009
<u>EXPLODE</u>	1012
<u>EXT</u>	1014
<u>extractParasitic</u>	1030
<u>FLATTEN</u>	1037
<u>FLOATCHK</u>	1039
<u>FRINGE CAP</u>	1040
<u>GENRECT</u>	1043
<u>GLOBAL-SCONNECT</u>	1045
<u>GPATHCHK</u>	1046
<u>GPATHDEF</u>	1048
<u>GROW</u>	1050
<u>HEDTEXT</u>	1052
<u>HIERARCHEN</u>	1054
<u>HOLE</u>	1055
<u>INT</u>	1060
<u>KEDTEXT</u>	1068
<u>LCONNECT</u>	1069
<u>LENGTH</u>	1071
<u>LEXTRACT</u>	1075
<u>LINK</u>	1083
<u>LPECHK</u>	1084
<u>LPESELECT</u>	1093
<u>LVSCHK</u>	1103
<u>MULTILAB</u>	1117
<u>MULTI-SHEETRES</u>	1119
<u>NDCOUNT</u>	1120
<u>NEIGHBOR</u>	1123
<u>NODE-FILE</u>	1124
<u>NODE-SELECT</u>	1126
<u>NOT</u>	1128

Dracula Reference

<u>OCTBIAS</u>	1132
<u>OR</u>	1134
<u>OVERLAP</u>	1137
<u>OVL</u>	1138
<u>PARAMETER CAP</u>	1139
<u>PARAMETER RES</u>	1141
<u>PARASITIC CAP</u>	1143
<u>PARASITIC DIO</u>	1146
<u>PARASITIC RES</u>	1150
<u>PATHCHK</u>	1152
<u>PERI</u>	1155
<u>PLENGTH</u>	1156
<u>POSTENC</u>	1158
<u>PROBE</u>	1159
<u>RCONNECT</u>	1161
<u>RECTCHK</u>	1163
<u>RELOCATE</u>	1166
<u>RLENGTH</u>	1168
<u>RSPFSELECT</u>	1170
<u>SAMELAB</u>	1171
<u>SCONNECT</u>	1173
<u>SELECT</u>	1176
<u>SHRINK</u>	1190
<u>SIZE</u>	1192
<u>SNAP</u>	1205
<u>SOFTCHK</u>	1206
<u>SPFSELECT</u>	1208
<u>STAMP</u>	1212
<u>TRIANGLE</u>	1215
<u>WIDTH</u>	1216
<u>XBOX</u>	1225
<u>XCELL</u>	1227
<u>XDEVICE</u>	1229
<u>XOR</u>	1231
<u>XVIA</u>	1234

Dracula Reference

A

<u>Supported Database Records</u>	1237
<u>Introduction</u>	1237
<u>GDSII Records</u>	1237

B

<u>Data Structure Translator</u>	1239
<u>Running the Translator</u>	1239
<u>Examples</u>	1240

C

<u>Running Dracula</u>	1241
<u>Running a Dracula DRC/ERC Job</u>	1241
<u>Submitting the jxrun.com File</u>	1241
<u>Running Other Dracula Programs</u>	1243
<u>UNIX Commands</u>	1243
<u>System Requirements</u>	1244
<u>UNIX Case-Sensitivity Enhancements</u>	1245
<u>PDRACULA</u>	1245
<u>LOGLVS</u>	1248
<u>Batch Job Queuing Utility</u>	1249

<u>Glossary</u>	1255

<u>Index</u>	1263

Preface

This manual assumes that you are familiar with the development and design of integrated circuits. It contains reference information about the Dracula® standalone verification product.

The preface discusses the following:

- [Related Documents](#)
- [Typographic and Syntax Conventions](#)

Related Documents

The *Dracula Reference* is often used with other Cadence products, or requires knowledge of special language (SKILL, TLF, DEF, Verilog). The following documents give you more information about these tools and languages.

- See the [Virtuoso Layout Editor User Guide](#) if you want to use the Cadence layout editor to enter your design.
- See the [Cadence Installation Guide](#) for information about how to install the product.
- See the [Dracula User Guide](#) for detailed instructions on how to perform Dracula tasks.
- See the [Virtuoso Design Environment User Guide](#) for information on how to
 - ☐ Start the Cadence software
 - ☐ Get around in the Cadence software
 - ☐ Get help on a command
 - ☐ Use the mouse
 - ☐ Use menus
 - ☐ Use standard and options forms
 - ☐ Start a design session

Typographic and Syntax Conventions

This section explains the conventions that this manual uses when describing commands.

`text`

Text in `courier` font indicates text that you must type exactly as it is presented.

argument

Text in *italic* font indicates text that you must replace with an appropriate argument.

`{ }`

Surrounds optional input.

`/`

Separates a choice of options.

`...`

Indicates that you can repeat the previous argument.

The following syntax shows a choice between two items:

```
ABORT-SOFTCHK = YES/NO
```

With this syntax, you can type one of the following:

```
ABORT-SOFTCHK = YES
ABORT-SOFTCHK = NO
```

The following syntax shows how this manual represents Dracula options:

```
ELEMENT MOS {[option]} layer-a layer-b layer-c {layer-d}
```

Here is an example of how you might type this command:

```
ELEMENT MOS[N] ngate poly nsd pwell
```

Type the command name, `ELEMENT MOS`. You then have the option of typing a code, represented in the syntax by the variable *option*, that specifies the type of the MOS device. The command does not require a device type to run properly, but you can specify it if you want. You must put brackets around the device type.

The names of *layer-a*, *layer-b*, and *layer-c* are all required. However, the name of *layer-d*, in braces, is optional.

All Dracula commands are available for all checking methods unless otherwise stated in the command description. Checking methods are *modes* (flat, hierarchical, cell, and composite)

Dracula Reference

Preface

and Dracula applications (DRC, ERC, LVS, LPE, PRE). Commands that do not offer all checking methods include a “Checking Methods” section that lists the modes and applications available with that command.

An  icon in the margin indicates an option that you can use only with hierarchical Dracula.

Dracula Reference

Preface

Overview

The purpose of this chapter is to provide an introduction to the Dracula[®] standalone physical verification product.

This chapter focuses primarily on the following:

- [“Introducing Dracula”](#) on page 29
- [“Introducing Hierarchical Dracula”](#) on page 34

Introducing Dracula

The Dracula product is a suite of software tools that is acknowledged as an industry standard for Integrated Circuit (IC) design verification. The following features describe Dracula’s advantages over other verification products:

- Accuracy

Dracula software locates true errors and eliminates false ones. Dracula’s accuracy is accomplished with the following:

- ☐ Electrical node description and label stamping
- ☐ Element, device, and circuit descriptions
- ☐ All-angle geometry checking
- ☐ Advanced checking to verify state-of-the-art deep submicron designs
- ☐ Exhaustive testing by Cadence and its customers

Dracula software represents all-angle geometries as trapezoidal data. This data representation technique is applied in hierarchical form to the disk storage, memory, and checking and extraction processes.

- Speed and efficiency

Dracula Reference

Overview

Dracula features a trapezoidal database that gives the checking algorithms fast, efficient access to the layout data.

A band scan approach to data processing increases performance and makes resource requirements directly proportional to the size of the IC layout.

■ Interactive processing

An interactive preprocessor drives Dracula. The following features are easy to use and can increase your productivity:

- ☐ Free format command entry in a block-oriented language
- ☐ User-defined names for IC layers
- ☐ Online interactive diagnostics
- ☐ Complete set or subset of rules
- ☐ Break point and restart capability
- ☐ Commands that work in all Dracula applications

■ A fully integrated system

Dracula offers a full tool suite to meet most verification needs:

- ☐ DRC to ensure the chip is processed correctly
- ☐ LVS and ERC to ensure the chip is electrically correct
- ☐ LPE and PRE parasitic data that is used by simulators to ensure the chip performs to specifications
- ☐ Mask generation for accuracy

These modules are integrated and work from the same database to ensure a smooth transition through each phase of IC verification.

Flat and Hierarchical Databases

Dracula offers three approaches to design verification: flat Dracula, hierarchical Dracula, and distributed Dracula.

■ Flat Dracula

Best for verifying small to large databases. Sold as Dracula II.

■ Hierarchical Dracula

Best for verifying large hierarchical databases. Sold as Dracula III.

- **Distributed Dracula**
Best for very large flat databases. It flattens hierarchical databases.

All three products support all Dracula applications (DRC, LVS, ERC, LPE, PRE) and require similar rules files. All features described in this manual apply to hierarchical Dracula. Some apply to hierarchical Dracula only, but not to flat Dracula. Information that applies to hierarchical Dracula only is identified in the section title or by an  icon in the margin.

For more information about hierarchical checking, refer to the section, [“Introducing Hierarchical Dracula”](#) on page 34.

Dracula Applications

Dracula offers a complete set of integrated applications for verifying IC layout designs. This section describes each of the Dracula applications.

- **Design Rules Checker (DRC)**

DRC performs geometric spacing checks on the layout to ensure it can be manufactured under a specified IC process technology. The DRC system is process-technology-independent; you can write different Dracula rules files for different process technologies.

DRC has a resizing capability that performs complex resizing and checking before producing the final mask geometries. Dracula can resize layouts based on mask or silicon size and can accommodate the design rule requirements of new or modified process technologies.
- **Electrical Rules Checker (ERC)**

ERC checks for shorted circuits, open circuits, and floating nodes. ERC isolates errors down to the shortest connection path for shorted circuits. ERC performs process-technology-independent and process-technology-dependent electrical rule checks. ERC detects the following process-dependent errors:
 - ☐ Invalid devices
 - ☐ Improper implant types
 - ☐ Improper substrate bias
 - ☐ Improper power and ground connections
 - ☐ Singular nodes
- **Layout Versus Schematic (LVS)**

Dracula Reference

Overview

LVS compares the IC layout to the schematic and reports discrepancies in the layout interconnection. LVS is process-independent and interfaces to logic simulation languages and schematic entry systems. A trace capability feature facilitates error isolation and identification.

■ Layout Versus Layout (LVL)

LVL compares two mask layout databases at the device or gate level and reports discrepancies in the interconnections and in specified device parameters.

■ Schematic Versus Schematic (SVS)

SVS compares two schematics.

■ Layout Parameter Extraction (LPE)

LPE computes and extracts the following key electrical parameters from IC layouts:

- ☐ Fixed capacitance of nodes
- ☐ Areas and perimeters of diodes
- ☐ Gate size of transistors
- ☐ Areas and perimeters of bipolar transistors
- ☐ Beta ratios

LPE uses extracted parameters from LVS consistency checks. It automatically flattens specified portions of the design hierarchy.

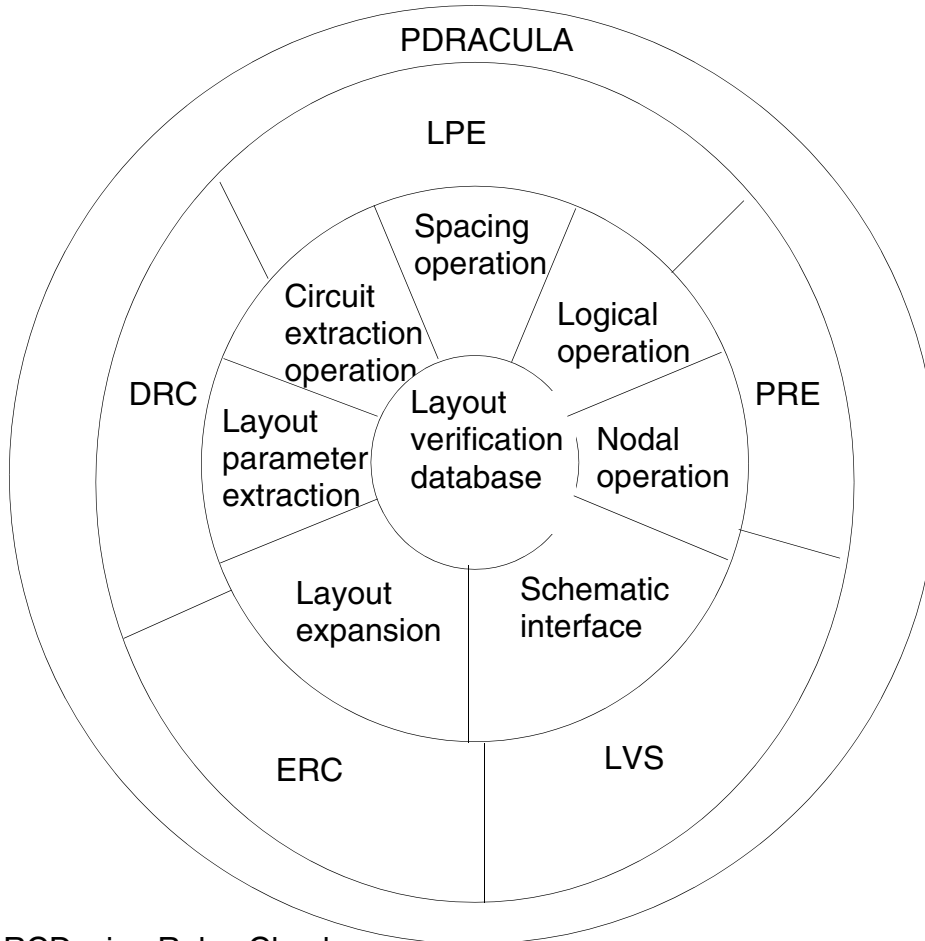
■ Parasitic Resistance Extraction (PRE)

PRE enhances LPE by extracting parasitic resistance created by any of the conduction layers in the layout. With LPE and PRE, you can extract resistance and capacitance parameters in the interconnect layers of the layout and thus generate accurate simulations for the most critical designs.

You run the Dracula applications using the preprocessor, PDRACULA. PDRACULA controls execution and most of the file management functions, eliminating the need for complex job control languages. PDRACULA provides tools for both the IC process engineer who maintains the design rules and the designer who runs Dracula and interprets the results.

The following figure shows the interaction between the preprocessor and Dracula application programs. PDRACULA is the outermost shell of the verification system. It controls the Dracula applications that perform verification operations on the layout database.

Relationship Between the Preprocessor and Dracula Applications



DRC Design Rules Checker
LPE Layout Parameter Extraction
ERC Electrical Rules Checker
PRE Parasitic Resistance Extraction
LVS Layout Versus Schematic

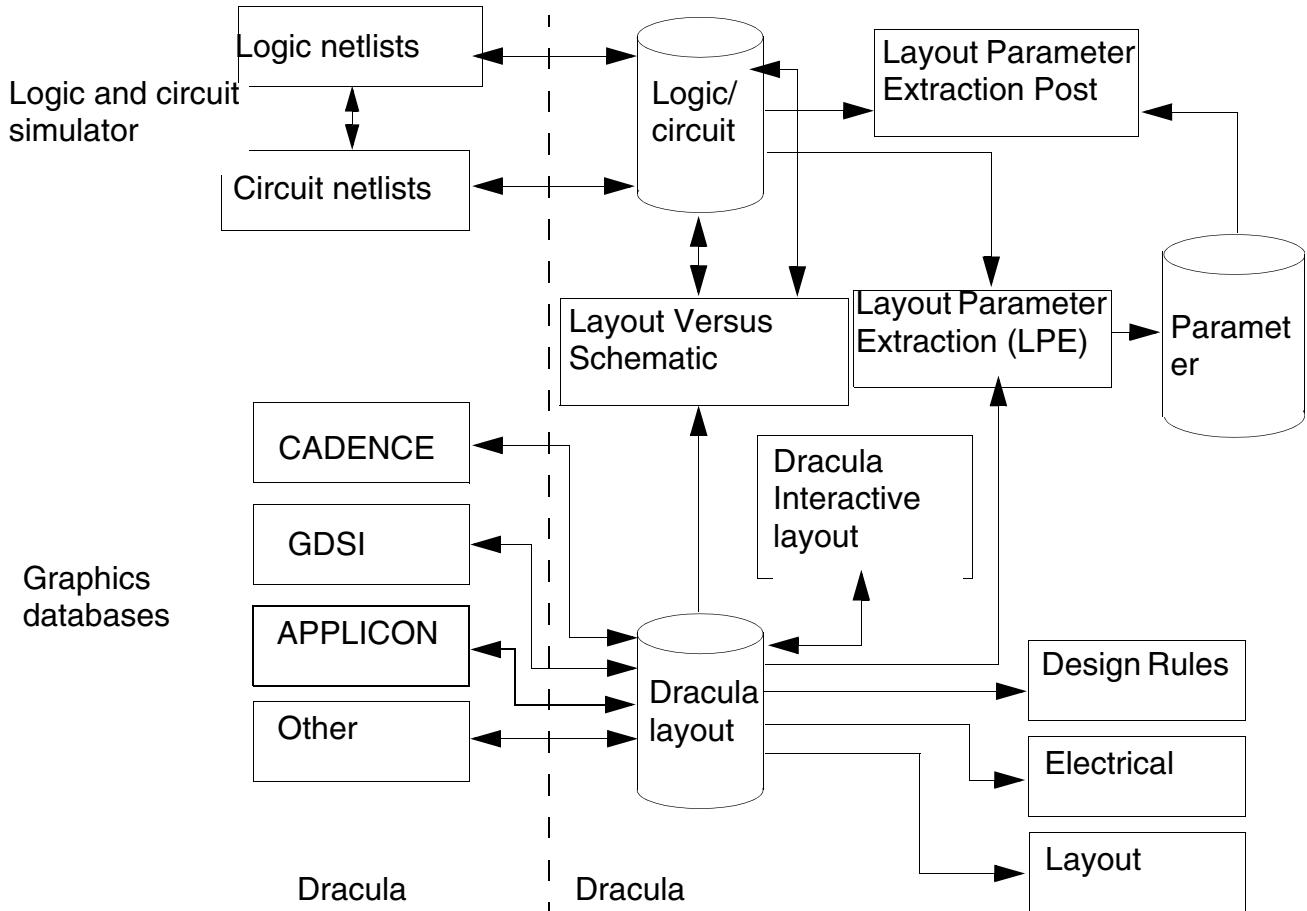
Dracula Interfaces

Dracula offers a full set of software that links the major steps of the IC design process. Interfaces are available for all commonly used logic and circuit simulators, graphics systems, and workstations.

The following figure shows the interfaces that Dracula supports and illustrates how the interfaces work with the Dracula tool set.

Dracula Reference Overview

Dracula Interfaces and Tools



Introducing Hierarchical Dracula

Hierarchical Dracula is a complete IC layout verification program that uses the IC design hierarchy to improve verification performance and throughput. Hierarchical Dracula supports the following Dracula applications:

- Hierarchical Design Rules Checking (HDRC)
- Hierarchical Electrical Rules Checking (HERC)
- Hierarchical Layout Versus Schematic (HLVS)
- Hierarchical Layout Parameter Extraction (HLPE)
- Hierarchical Parasitic Resistance Extraction (HPRE)

- New Hierarchical Layout Versus Schematic (NHLVS)
- Hierarchical Netlist Versus Netlist (HNVN)



The Dracula 4.5 release uses a database compression technique that was introduced into Dracula with version 4.1. Dracula 4.5 reads in all databases created with Dracula versions 4.1 through 4.3 inclusive.

Hierarchical Structure

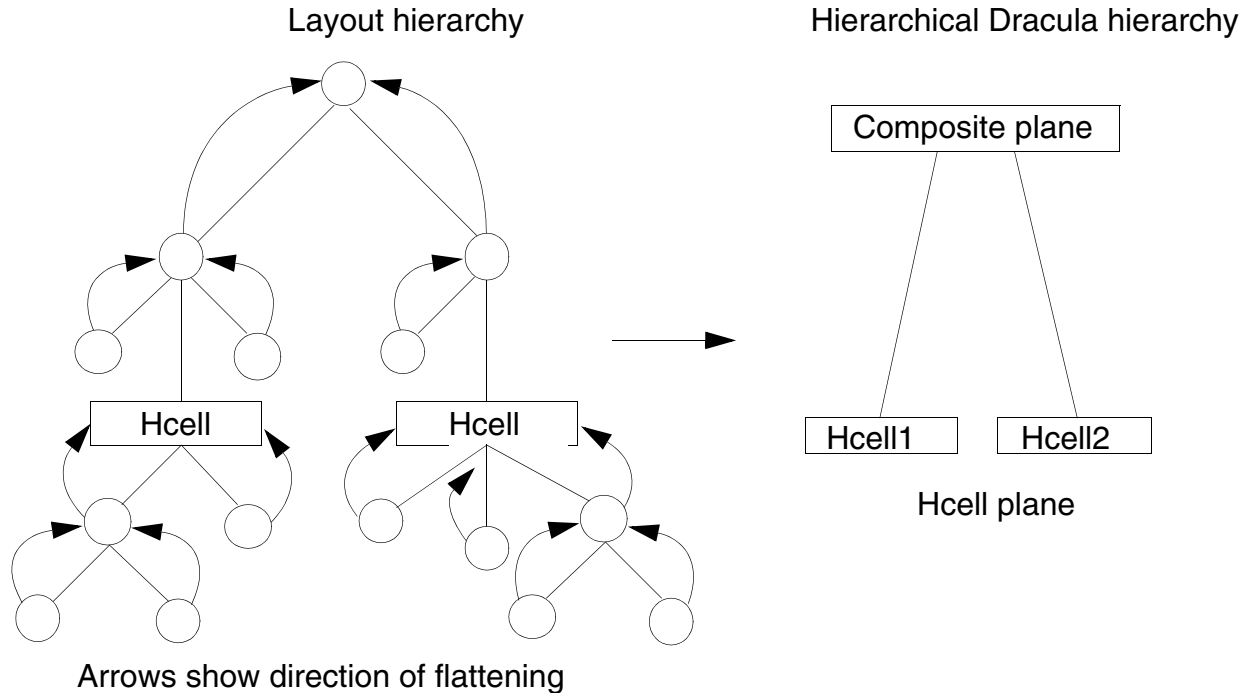
Hierarchical Dracula uses two levels of hierarchy: the *Hcell plane* and the *composite plane*.

The Hcell plane contains a set of cells chosen from the cell hierarchy. The system can choose these Hcells automatically, or the designer can choose them. The selected cells, referred to as *Hcells*, can be intermediate- or bottom-level cells in the layout hierarchy. Dracula expands all cells nested in an Hcell and brings them up to the Hcell plane.

The composite plane is the top level of the hierarchy. Dracula expands all cells that are not Hcells and are not contained in Hcells and brings them to the top level. This results in two levels of hierarchy, regardless of the number of levels of hierarchy in the original database.

The composite plane does not need to be the top level of your design. It can be an intermediate cell that is verified as a composite plane in one Dracula run and verified as an Hcell plane in another run.

Forming the Hcell Hierarchy



Hierarchical Dracula uses the hierarchy of the layout database to reduce redundant operations in the following ways:

- Dracula performs operations in Hcells only once, even though these cells can be placed many times in the layout.
- Dracula recognizes the same cell-to-cell and cell-to-composite relationships and processes them once for HDRC.

This reduction gives a more concise DRC error summary by eliminating redundant error reports in multiple cell placements and reducing the time needed for verification.

HDRC operations do not assume or impose special restrictions, design methodologies, or layout limitations. These operations allow overlaps between cells so that geometries from the top level or from a given cell can penetrate or completely cross through without requiring protection frames or cell boundaries around each cell.

Note: If your layout database is flat or contains many overlapping cells, the performance and error reporting benefits of hierarchical Dracula might not be obvious.

Functions Not Available in Hierarchical Rules Files

Although the rules file for hierarchical and flat mode Dracula are similar, you cannot use the following functions in hierarchical rules files:

- LENGTH
- EXT G option (You can specify all other EXT options in a hierarchical rules file.)
- RELOCATE

Hierarchical Dracula Modes

Hierarchical Dracula operates in four different modes: flat, cell, hierarchical, and composite. To specify the mode you want, use the CHECK-MODE command in the Description block.

- Flat Mode

If you do not use the CHECK-MODE command, flat mode is the default. Flat mode provides all Dracula capabilities and accepts rules files written for flat mode Dracula.

If you are new to Dracula, begin in flat mode. After you develop proficiency in DRC, ERC, LVS, and LPE, use the other modes for hierarchical operations.

- Cell Mode

Cell mode verifies a group of selected Hcells and reports errors on a cell-by-cell basis.

Use cell mode to check selected cells only. Do not use cell mode to check an entire layout.

- Hierarchical Mode

In hierarchical mode, HDRC checks the Hcells and the composite plane, as well as all Hcell-to-Hcell interfaces and Hcell-to-composite interfaces. Use hierarchical mode for HDRC only.

- Composite Mode

Composite mode extracts interconnect information from cells then processes composite, cell, cell-to-cell, and composite-to-cell interconnect information for HERC and HLVS. It processes only composite information for HLPE. Running HLVS in both cell mode and composite mode achieves a complete layout-to-netlist check.

Use composite mode for HDRC, HERC, HLVS, HLPE, and HPRE.

Note: You cannot use composite mode HDRC and HLVS in the same Dracula run.

Dracula Reference Overview

Using Dracula

The focus of this chapter is on using Dracula to perform the following:

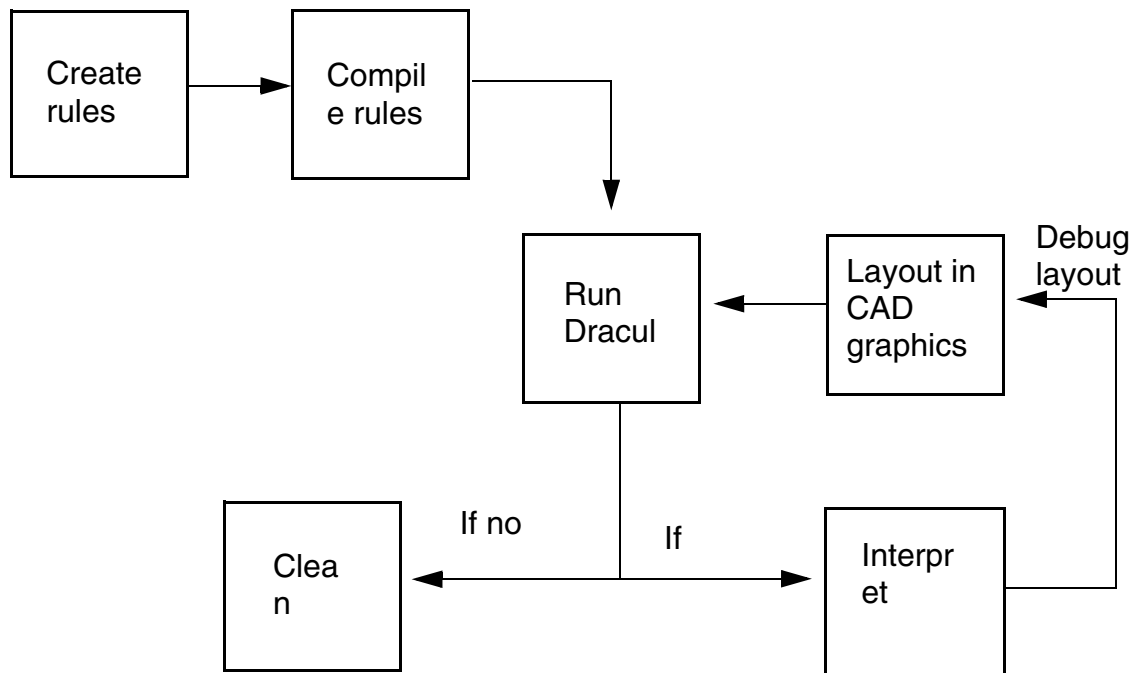
- [Verifying a Design with Dracula](#) on page 39
- [Creating a Rules File](#) on page 40
- [Compiling the Rules File](#) on page 45
- [Running Dracula](#) on page 62
- [Interpreting Dracula Output](#) on page 63
- [Dracula Distributed Processing](#) on page 65
- [Running Dracula in 32-bit or 64-bit Modes](#) on page 77

Verifying a Design with Dracula

Before running Dracula to verify a design, you must write a rules file. The rules file identifies the layers in your design and the verification operations you want Dracula to perform. To verify a layout, compile your rules file with the PDRACULA preprocessor, then run the Dracula program.

If Dracula finds errors, it generates error reports and an error database that contains information you can use to debug the layout. After debugging, you can resubmit the verification job and continue debugging until you have a clean layout.

Dracula Layout Verification Process



Creating a Rules File

A rules file is an ASCII file you create with a UNIX® text editor such as *vi* or *emacs*. The rules file contains Dracula commands that identify layers in your design and the verification operations you want Dracula to perform.

Rules File Structure

A Dracula rules file has the following parts:

- Description block
- Input-Layer block
- Operation block

Dracula Reference

Using Dracula

Each block begins with a statement that identifies the block and ends with an *END statement. The *END statement is always the last line in the block. The block statements begin with an asterisk (*) in the first column.

Description Block

The Description block describes the system on which Dracula runs. This block also contains information about the circuit to be checked, including the following:

- Execution mode
- Primary cell name
- Input/output devices, file names, and formats
- Graphics scale factor
- Grid increments
- Output print file
- Display of warning messages

The *DESCRIPTION statement marks the beginning of the Description block. See [Chapter 11, “Description Block Commands,”](#) for descriptions of the commands you can include in the Description block.

Input-Layer Block

The Input-Layer block relates your layout layer numbers or names to Dracula layer names and specifies other information Dracula needs about layers. This block specifies the following:

- Layers to import
- Layer names
- Mask order
- Layers to export
- Text layers

The *INPUT-LAYER statement marks the beginning of the Input-Layer block. See the Introduction section of Chapter 12 for descriptions of the commands you can include in the Input-Layer block.

Operation Block

The Operation block defines the operations and applications to be run and flags errors. The types of operations it defines are

- Logical
- Electrical node
- Resizing
- Parameter
- Spacing
- ERC, DRC, LVS, LPE, and PRE

The *OPERATION statement marks the beginning of the Operation block. See the Introduction section of Chapter 13 for descriptions of the commands you can include in the Operation block.

Dracula Reference

Using Dracula

Syntax

The following table shows the special characters used in Dracula rules files.

Character	Description	Example
Blank space	Separates arguments.	AND NDIFF POLY NGATE
, Comma	Separates arguments.	GROUND-NODE = VSS1, VSS2, GND
= Equal sign	Separates arguments.	AUTOMATCH = YES
* Asterisk	Indicates a control statement, a block definition, or a break point. Must be in the first column.	*DESCRIPTION . *END
& Ampersand	Indicates a conjunctive rule. Must be the last character on a line, except when used as a conjunct rename within a conjunct rule.	EXT lay1 lay2 LT 1 & EXT lay1 lay3 LT 1 & EXT lay1[O] lay3 LT 2 OUTPUT err1 23
[] Brackets	Surround an option. Do not put a space on either side of an opening or closing bracket.	ENC[T] ptie pwell SELLT 10 badptie
; Semicolon	Introduces a comment. Can be on a command line or on a separate line. The input to the right of the semicolon is a comment.	*INPUT-LAYER diff = 1 ; diffusion input layer poly = 5 ; polysilicon input layer metal = 7 ; metal input layer

The first word of each command line must be a Dracula command. Each line can contain only one command. All commands within each block are free format; you can place them in any order you want. Blank spaces can precede a command.

You must separate all data items in a command line with one of these delimiters: a blank space, a comma, or an equal sign. For example:

```
INDISK = chip ; optional comment
```

Do not use a plus sign (+) to indicate a positive number. Do not put a space between a minus sign (-) and a negative number.

For additional information on creating Dracula rules files, see the [Writing Rules for Dracula](#) chapter in the *Dracula User Guide*.

Rules File Example

The following example shows a Dracula rules file:

```
*DESCRIPTION
;
; System description data input section
;
PRIMARY    =  iomux
SYSTEM     =  GDS2
INDISK     =  1234
OUTDISK    =  5678
SCALE      =  .001 MICRON
MODE       =  EXEC NOW
RESOLUTION =  .25 MICRON
.
.

*END
;
*INPUT-LAYER
;
; Layer mnemonic name definition section
;
poly       =  5
diff       =  7
implant    =  2
metal      =  9
mc1        =  10
epi        =  11
.
.

CONNECT-LAYER =  diff  poly  metal
PAD-LAYER =  vapox
*END
;
*OPERATION
;
; Logical, resizing, connection, and spacing operations
;
AND        poly      diff      gate
SIZE       gate      BY 1      ovgate
AND        diff      ovgate     difgate
ENC[0]     difgate   implant    LT 4   OUTPUT rule01 5
.
.
CONNECT    metal     poly      BY mc1
CONNECT    poly      diff      BY epi
CONNECT    metal     diff      BY mc1
.
.
*END
```

Compiling the Rules File

After you create your rules file, use the PDRACULA preprocessor to compile it. PDRACULA performs three tasks:

- Checks for syntax errors in the rules file
- Compiles the error-free rules file and stores it in the *jxrun.com* file

The *jxrun.com* file contains the commands that submit a Dracula job.

- Creates a symbolic link from the main library to the run directory and puts it in the *jxrun.com* file

All reference libraries must be in the run directory or symbolically linked to the run directory.

If there is a problem compiling the rules file, PDRACULA issues either a *warning* or an *error* message indicating the line in question, the problem, and sometimes a possible solution. You must correct all errors and recompile the file. If you understand the implications of a warning message, you can disregard it.

PDRACULA is interactive. While PDRACULA runs, you can enter PDRACULA commands, which are described in the following sections. Here are some of the things you can do with the PDRACULA commands:

- View error messages
- Interactively edit and store changes to the rules file
- Compile multiple rules files to one run file

Note: There is no limit to the number of steps for Dracula, but there is an internal limitation for PDRACULA. The maximum number of commands in the Operation Block is 12000. The maximum number of characters in a line is 256.

In addition, during the compilation of a rule file, PDRACULA splits each command into terms separated by delimiters, such as commas, spaces, and equal signs. However, this does not apply to semicolons. The maximum number of terms is limited to 20, which is enough for all commands except LVCHK and LPECHK. Exceeding the maximum number of terms will return an error message.

When necessary, you can overcome the 20 term limitation for LVCHK and LPECHK by writing the commands as two commands with complementary arguments. For example, the command:

Dracula Reference

Using Dracula

```
lvschk[cpur] weffect=.125 muldelw=-2 wpercent=2 lpercent=2 capval=1  
resval=1 eaper=1 dioarea=1 printline=200 listunmatch=200 dioperi=4
```

can be rewritten as:

```
lvschk[cpur] weffect=.125 muldelw=-2 wpercent=2 lpercent=2 capval=1 resval=1  
lvschk[cpur] eaper=1 dioarea=1 printline=200 listunmatch=200 dioperi=4
```

Running PDRACULA

To compile your rules file, follow these steps:

1. To start PDRACULA on a UNIX workstation, type the following command:

PDRACULA

2. Type PDRACULA commands at the PDRACULA command prompt, which is a colon (:). For example, you can

- ☐ Use the /GET command to indicate the rules files to compile and to display any error or warning messages.
- ☐ Use the /NEXT command to run multiple Dracula jobs.
- ☐ Use the /EDIT command to bring up your rules file in the *vi* or *emacs* text editor so you can make corrections.

See the PDRACULA command descriptions on the following pages for details about using individual PDRACULA commands.

3. Use the /FINISH command to compile the rules file and end the PDRACULA session.

If the rules file contains errors, the PDRACULA command prompt returns and PDRACULA prompts you to correct the errors. If your rules file compiles without any errors, the PDRACULA session ends and the system command prompt appears.

4. If there are errors, make any necessary changes to the rules file and rerun PDRACULA.

Using PDRACULA Commands

To compile your rules file, view error output, and make corrections to your rules file, type the PDRACULA commands interactively at the PDRACULA command prompt. Precede each of the commands with a slash (/). To save time, type the first three letters (for example, /RES for /RESTART) or the first character (for example, /G for /GET).

The following pages describe the PDRACULA commands. Some of the following command descriptions use the term *active input sequence*. The active input sequence is the PDRACULA commands you have typed so far in the current PDRACULA session.

/ABORT

/ABORT

Description

Stops a PDRACULA run and closes all temporary files. Use /ABORT to stop all processing.

Example

/ABORT

/CLEAR

/CLEAR

Description

Deletes all currently active commands. Use /CLEAR to free your UNIX window and start a new process without exiting PDRACULA.

Example

/CLEAR

/DCL

/DCL

Description

Communicates with your host operating system without exiting the preprocessor. After issuing a /DCL command, you can use any of the host commands.

To return to PDRACULA, position the cursor on a blank line and press the Return key.

Example

```
/DCL  
ps ax  
/DCL  
ls *.COM
```

/FINISH

`/FINISH {jcl-filename}`

Description

Ends the command sequence and starts the PDRACULA preprocessor.

PDRACULA checks the Description block for missing data and the rules file for empty Input-Layer or Operation blocks. If any required parameters are missing from the Description block, or if any layers are missing from the Input-Layer block, PDRACULA issues a warning.

If the rules file contains errors, PDRACULA prompts you to correct the errors and displays the PDRACULA command prompt. If the rules file has no errors, PDRACULA finishes and your system command prompt appears.

Note: If PDRACULA finds errors on a UNIX workstation, the status returns to status 1 after the /FINISH command. For information about PDRACULA status codes, refer to the [STATUS](#).

Arguments

jcl-filename

Specifies the name of the file generated by the preprocessor. The default file name is *jxrun.com*.

Example

The /GET command reads in and displays the contents of the rules file, *myfile.rul*. After PDRACULA displays the file, /FINISH compiles the rules file, then ends the PDRACULA session.

```
/GET myfile.rul
*DESCRIPTION
SYSTEM      = GDS2
PRIMARY     = iomux
MODE        = exec now
INDISK      = 1234
OUTDISK     = 5678
SCALE       = .001 micron
RESOLUTION  = .25 micron
*END
*INPUT-LAYER
diff        = 1
poly        = 2
*END
*OPERATION
```

Dracula Reference

Using Dracula

```
AND poly diff gate
EXT gate LT 5 OUTPUT rule03 5
*END
/FINISH
```

/GET

`/GET input-filename {Y/N} {FILE} {MULTI}`

Description

Specifies which rules file or files to compile.

Arguments

input-filename

Specifies the name of the file to be compiled.

Y/N

Y displays the input file and all warning and error messages on the screen.

N suppresses display of the input file on the screen. Only warning and error messages appear on the screen. In Dracula 4.8 and subsequent releases, N is the default.

FILE

When used with Y, sends the input file and all warning and error messages to a file. The name of the file is the rules file name with a *.lis* extension. When used with N, sends only warning and error messages to the *.lis* file.

MULTI

Used for multilevel DRC only. Creates a two stage *jxrun.com* file to analyze and select multilevel Hcells. Creates the MULTICEL.TAB file after the *jxrun.com* file completes.

Examples

To display only warnings and error messages on the screen, type

```
/GET abc.rul
```

or

```
/GET abc.rul N
```

To display the contents of `abc.rul` file and messages on the screen, type

```
/GET abc.rul Y
```

To send warnings and error messages to a file named `abc.lis`, type

```
/GET abc.rul FILE
```

or

Dracula Reference

Using Dracula

```
/ GET abc.rul N FILE
```

To send all the contents of `abc.rul` file and messages to a file named `abc.lis`, type

```
/GET abc.rul Y FILE
```

Multilevel Example

```
/GET drc.rul MULTI
```

The contents of *drc.rul* appear on the screen and PDRACULA creates the two stage *jxrun.com* file to perform data analysis and multilevel Hcell selection.

After you run the *jxrun.com* file, DRACULA writes the selected multilevel Hcells to the MULTICEL.TAB file, then you rerun PDRACULA with the following command:

```
/GET drc.rul N
```

PDRACULA does the following:

- Creates a subdirectory for each cell name in the MULTICEL.TAB file
- Creates a *jxmult.com* file for the root directory and a *jxrun.com* file for each subdirectory
- Symbolically links the MULTICEL.TAB file to each subdirectory
- Chooses Hcells suitable for multilevel hierarchy and creates directories for checking each Hcell (CHECK-MODE = MULTI)

After PDRACULA completes, you execute the *jxmult.com*, which executes the *jxrun.com* files in each subdirectory.

PDRACULA selects cells for Hcells based on the following criteria:

- Cells must exceed the requirements specified in the HCELL-RULE command. The default number of placements is four. The default number of segments is 1,024 times the number of layers.
- If both parent and child qualify as Hcells, DRACULA checks if the total number of line segments (after expansion) of the child exceeds 90 percent of that of the parent. If so, DRACULA selects only the child cell as an Hcell.
- Does not violate overlap criteria set by the CHECK-MODE command.
- DRACULA writes multilevel cells with over 100,000 line segments to the MULTICEL.TAB file.

Dracula Reference

Using Dracula

/HELP

`/HELP {ACTION/EXECUTIVE/FILES-CREATED} item`

Description

Displays information about PDRACULA. Use HELP to access online information for quick reference.

Arguments

ACTION

Specifies the subtopic under which preprocessor commands are listed.

EXECUTIVE

Specifies the subtopic under which input block information is listed.

FILES-CREATED

Specifies the subtopic.

item

Can be any keyword in the help dictionary under the headings of ACTION, EXECUTIVE, or FILES-CREATED. After choosing one of these, you can choose from the categories listed on your screen.

Note: You must set the DRACHELP4 environment variable to point to the directory where the Dracula help files are located. The following is the UNIX command for setting the environment:

```
setenv DRACHELP4 drac_dir/etc/drac_help/
```

Examples

`:/HELP`

The following text appears on your screen. The text you type is shown in **bold**.

```
PDRACULA is the Dracula compiler that takes input in an English-like language and
converts it to the JCL-like control language that drives the Dracula system.
Additional information available:
ACTION    EXECUTIVE  FILES-CREATED
PDRACULA SUBTOPIC? ACTION

PDRACULA
    ACTION
```

Dracula Reference

Using Dracula

ACTION DIRECTIVES

The EXECUTIVE CONTROL ACTION DIRECTIVES will cause some specific action to take place during the input processing phase of the program. These directives can be invoked anywhere within the input sequence of the program. The input processing system recognizes these directives by a slash '/' appearing as the first character in the input and followed directly by a keyword.

NOTE: Input that appears in curved brackets (that is, {OUTPUT c-name l-num}) indicates that this input data item is optional and depends on the user's specific requirements.

Additional information available:

/ABORT	/CLEAR	/COMMANDS	/CREATE-JCL
/CURRENT-VERSION	/DCL	/EDIT	/FINISH
/GET	/HELP	/JCL-FILENAME	/LIST
/NEXT	/OFF	/ON	/PRINT
/QUIT	/RESTART	/STATUS	/STORE
/SUBMIT	"?"	HELP	"UNSORTED"

PDRACULA ACTION Subtopic ? **/GET**

PDRACULA

ACTION

/GET

/GET {file-name {NOLIST | LIST} {FILE}}

The GET Action Directive will cause the input processing system to go out and read a previously generated file and then insert it into the input sequence. This function lets you use previously generated processes and card input streams to perform Design Rule Checks. NOLIST prevents the card images in file name from being echoed. LIST causes the card images in file name to be echoed. If the keyword FILE is specified, then all error and warning messages will be written to a file named *filename.LIS*

Additional information available:

ARGUMENT EXAMPLES

PDRACULA ACTION /GET Subtopic ? **[RETURN]** ;Leave /GET
;subtopic.

PDRACULA ACTION Subtopic ? **[RETURN]** ;Leave Action
;subtopic.

PDRACULA Subtopic ? **[RETURN]** ;Leave HELP
;facility.

:

/LIST

```
/LIST block {UNSORTED/SORTED}
```

Description

Lists all or part of the active input sequence.

Arguments

block

Defines what to list: the input or output file names, or one of the following blocks:

```
ALL  
DESCRIPTION  
INPUT-LAYER  
OPERATION
```

UNSORTED

Indicates that the output listing is not sorted.

SORTED

Expands free-format command lines into predetermined fields that make the command sequence easier to read and check. This option removes comment statements. Default.

Examples

```
/LIST myfile UNSORTED  
/LIST DESCRIPTION  
/LIST INPUT  
/LIST OPERATION UNSORTED
```

/NEXT

/NEXT

Description

Marks one of the termination points in your command sequence. /NEXT creates the *jxrun.com* file and starts the preprocessor again. Use /NEXT to generate multiple *jxrun.com* files so you can run multiple Dracula jobs sequentially in the same directory. This command prompts for any data missing from the Description block.

/NEXT ends a prior /GET command and prevents unwanted concatenation. To end your entire sequence, use the /FINISH command.

Note: You *should not* use the /NEXT command when compiling a Dracula rule file for Dracula distributed processing with PDRACULA.



Do not use the /NEXT command on multiple LPE runs. The /NEXT command produces conflicts between the LPEDIR.TAB directory files.

Example

Enter the following commands to create three different *jxrun.com* files and execute the jobs in sequence:

```
/GET      drc1.com
/NEXT     ;creates jxrun.com
/GET      drc2.com
/NEXT     ;creates jxrun1.com
/GET      drc3.com
/FINISH   ;creates jxrun2.com
```

/PRINT

```
/PRINT block {UNSORTED/SORTED}
```

Description

Lists all or part of your active input sequence in a temporary file.

Note: The temporary file on the UNIX operating system is *fort.10*.

Arguments

block

Specifies which part of your input sequence to list in a temporary file. The list can include the input or output file name or one of the following blocks:

```
ALL  
DESCRIPTION  
INPUT-LAYER  
OPERATION
```

UNSORTED

Indicates that the list is not sorted.

SORTED

Expands the free-format command lines into predetermined fields that make the command sequence easier to read and check. This option removes comment statements. Default.

Examples

```
/PRINT myfile UNSORTED  
/PRINT DESCRIPTION  
/PRINT INPUT  
/PRINT OPERATION UNSORTED
```

/RESTART

/RESTART

Description

Executes part of your *jxrun.com* file without running the entire job again. This command restarts Dracula from a breakpoint in the *jxrun.com* file. PDRACULA prompts for "From" and "To" breakpoints.

On a UNIX workstation, you must manually run Dracula again using *jxprechk.com*. /RESTART affects the *jxprechk.com* file, which runs the portion of *jxsub.com* that lists your "From" and "To" breakpoints.

For details about break points and the BREAK command, see the "Introduction" section of Chapter 13.

Examples

```
/GET myfile.com NO
/RES
From: plot
To:   end
```

Reserved Breakpoints

You can use the automatic breakpoints listed below in the *jxrun.com* file, or you can insert other break points. Do not use these reserved words for breakpoints you define with the BREAK command.

```
BEGIN:
EXPAND:
SORT:
MERGE:
LVSNET:
SCHMIN:
SYSOUT:
DELETE: (if KEEPDATA = NO/SMART is specified)
END:
```

To use the /RESTART command from a breakpoint other than *BEGIN* on a previous Dracula job, use the KEEPDATA=YES command. If you specify KEEPDATA=NO or KEEPDATA=SMART in the previous job, the job stops.

Dracula Reference

Using Dracula

/STATUS

/STATUS

Description

Displays your active status so you can identify the block being processed and the current status of processing. To determine which block Dracula is processing, use /STATUS following a /GET command.

Example

```
:/STATUS
```

```
YOU ARE NOT IN ANY BLOCK
```

```
  2 INPUT-LAYER      CARDS
  8 DESCRIPTION      CARDS
  4 OPERATION        CARDS
  0 PLOT-OUTPUT      CARDS
```

CATEGORY	NUMBER OF LINES	MAXIMUM ALLOWED
LOGICAL	1	1000
SIZE	3	600
INPUT LAYERS	2	256
OUTPUT CELLS	3	2000
TOTAL OPERATIONS	4	2000

/STORE

`/STORE filename {UNSORTED/SORTED}`

Description

Stores your active input sequence in a file. To simplify checking, the stored file is sorted and the data is placed in predetermined fields. The /STORE command lets you build processes and procedure files interactively.

Arguments

filename

Specifies the name of the mass storage file created from your active input sequence.

UNSORTED

Indicates that the output file is not sorted.

SORTED

Expands free-format command lines into predetermined fields that make the command sequence easier to read and check. This option removes comment statements. Default.

Examples

```
/STORE myfile  
/STORE process UNSORTED
```

Running Dracula

The PDRACULA preprocessor produces four executable files that you can use to run Dracula.

- `jxrun.com` runs Dracula in the foreground.
- `jxsub.com` runs Dracula in the background.
- `jxmuilt.com` runs Dracula in multilevel mode.
- `jxexe.com` and `jxpar.com` run distributed Dracula in the background.

The `jxrun.com` file is a C shell script that contains all information from your rules file in an executable format. The file is divided into “stages.” The first stage reads in your database. The next stages process the layers in your design and perform all operations defined in your rules file. PDRACULA can process over 80 temporary layers. The last stage converts your

database back to the original layout database format. As *jxrun.com* runs, the run messages it generates appear on the screen.

The *jxsub.com* file is a brief C shell script that executes *jxrun.com* in the background. This script saves all run messages in a log file. The following section, “Interpreting Dracula Output,” describes the log file and other Dracula output files.

The *jxmult.com* file is a C shell script that executes the *jxrun.com* scripts in each run directory when you run multilevel hierarchy. Dracula defines the primary cell for each run by the cells designated in MULTICEL.TAB. Dracula consecutively executes the *jxrun.com* in each subdirectory.

The *jxpar.com* and *jxexe.com* files are C shell scripts that Dracula uses when you run a distributed Dracula process. For information about the using these files, refer to the [Dracula Distributed Processing](#) section in this manual.

Interpreting Dracula Output

Dracula produces printed results of all major Dracula applications. If you run Dracula in the foreground, the results appear on the screen while Dracula runs. If you run Dracula in the background, the results are saved in files.

Each Dracula run, regardless of the applications it includes, might produce the following files and extensions:

- A log file with the *.log* extension
- A summary file with the *.sum* extension
- A system error file with the *.mlg* extension
- A warning file with the *.msm* extension

The PRINTFILE command in the Description block defines the file name. The default is *printf*, which produces the files *printf.log*, *printf.sum*, *printf.mlg*, and *printf.msm*.

After each Dracula run, check the log file to make sure the job was completed properly. The last stage listed in the log file shows a module called GDS2OUT, APPLEOUT, or a similar module depending on your database format. If the Description block of the rules file includes the command PREINQUERY = YES, the last module listed is PREINQUERY.

Note: In the 4.8 version includes when you specify PREINTERACTIVE=YES, PDRACULA adds the module PREINQUERY at the end of the *jxrun.com* file, just like how the command PREINQUERY = YES does. However, the original command PREINQUERY = YES also works. If the log file lists something other than these modules in the last stage, refer to the

Dracula Reference

Using Dracula

system error file (*.mlg*). The system error file lists fatal error messages and log-off statistics, such as the amount of CPU time used for the run. If you include the EXTERM-NOWARN=NO command in the Description block, the log file lists the “unconnected pins in cell” messages.

After each Dracula run, you should also check the summary file for cells that PDRACULA generated or skipped, depending on the errors found. The summary file also lists the number of errors found and the x and y coordinates of any problem polygons.

The warning file lists all warnings and error messages generated during the run.

Dracula Distributed Processing

Dracula distributed processing shortens Dracula run times by using multiple workstations running simultaneously. Distributed processing is designed to be easy to use, to minimize the communication overhead between machines, and to maintain the integrity of your database. Distributed processing works on heterogeneous networks that support Sun *rpc*, UNIX BSD *rsh*, and *automount*.

- Dracula distributed processing runs
 - ❑ On multiple workstations connected through an NFS network. It can run on a combination of SUN, HP, IBM.
 - ❑ On the different CPUs of a single workstation.
- Dracula distributed processing
 - ❑ Does not require an optional license
 - ❑ Gives you the capability to run Dracula on multiple CPUs using a single Dracula license, independent of the number of CPUs used
 - ❑ Allows you to run multiple multi-CPU jobs, equal to the number of Dracula licenses that you own
- Dracula distributed processing runs the following applications in flat mode:
 - ❑ DRC
 - ❑ ERC
 - ❑ LVS (Dracula distributed processing also runs composite mode LVS.)
 - ❑ LPE
 - ❑ PRE
- Dracula distributed supports hierarchical DRC and composite mode LVS. (Dracula 4.7 supports LPE/PRE).
- `dracq` does not support Dracula.

The next sections describe how distributed processing works and the control scripts and files you need to run Dracula distributed. The remaining sections explain how to set up distributed runs and how to run Dracula distributed.

How Dracula Distributed Works

In a Dracula distributed run, each CPU on a MULTICPU or on a workstation is called a *node*. You start the job from the *master node*, which distributes tasks to the *slave nodes*.

A Dracula daemon, *DRACD*, manages the distribution of tasks between CPUs. Synchronization processes take place at these stages:

- After the last EXPAND
- Just before GDS2OUT

During a Dracula distributed run, the daemon creates and maintains the following two directories on each node:

- The `drac-dir` directory contains intermediate files generated during the run. Files must remain until being finally merged, and can be used if the job is restarted.
- The `drac-link` directory contains files that Dracula uses on the master node to identify all files from the previous run that were copied to the slave nodes.

In addition, the daemon maintains a history file, called `STATUS.TAB`, in the master node's working directory. This history file remains on the master node after the run is complete.

Machines/CPU's which can be used to run the tasks are described in a *node configuration file*.

Setting Up for Dracula Distributed Processing

Before you start a Dracula distributed run, you must do all of the following:

- Use the `MULTICPU` command to start the Dracula distributed processing in a multiple CPU workstation. This is a new command in the 4.8 release.
- In the master node, create the node configuration file. For example:

```
master/working1/executable_pathpower_rankmount_type
slave1/working10/executable_pathpower_rankmount_type
slave2/working20/executable_pathpwoer_rankmount_type
```

While creating the node configuration file, follow these guidelines:

- ☐ List the master node first.
- ☐ *Do not use symbolic links:* Make sure that the working directory names in the node configuration file are not links to the physical directory, but are names of the physical directory itself.

Dracula Reference

Using Dracula

- ❑ *Do not include blank lines:* Make sure that the node configuration files do not contain any blank lines (at the beginning, at the end, or between lines).

- Edit the Dracula rules file to add the following line in the DESCRIPTION block:

```
PARALLEL-FILE = node_configuration_file
```

- Create a run (working) directory on each slave node as listed in the node configuration file.

Note: You must create all working directories before you start the run.

- Check access and permission. This can require system privileges or the presence of a system administrator.

- ❑ From the master node, you must be able to `rlogin` to slave nodes *without being asked for a password* and vice versa.

- ❑ From the master node's working directory, you must have write permission to slave nodes' working directories. To check, enter the following:

```
master% touch /net/slave[i]/working_directory/dummy
slave[i]% touch /net/master/working_directory/dummy
```

- Make sure working files are in the directory of the master machine, or Dracula issues an error message.
- Add the `PARALLEL-FILE` command or the `MULTICPU` command to the Description block of the rules file.

Creating a Node Configuration File

To specify the node configuration file to use in a Dracula distributed run, use the PARALLEL-FILE command. Each line in the node configuration file lists a node to be used in the run and contains the following six fields:

Note: The master node must be the first node in the PARALLEL-FILE command.

Field 1

Name of a workstation to be used in the run.

Field 2

Working directory on this workstation. The working directory must be different for each entry because no two workstations can share identical disk space.

Field 3

Node type. The possible values are 1, 0, and asterisk (*). For the master node, specify 1. For a slave node, specify 0 or *. The master node must be the first entry.

Field 4

Dracula executable path for this workstation. An asterisk specifies the path listed by the PROGRAM-DIR command in the rules file. The path for the PROGRAM-DIR command must be the explicit path.

Field 5

Power rank of this workstation in relation to the other workstations being used in this run. The most powerful workstation has the highest number. An asterisk specifies a rank of one.

Field 6

Mount type, which can be one of the following:

0	Soft mount. Appends <i>/net/hostname/</i> in front of the working directory.
1	Hard mount. Appends <i>/dracnet/hostname</i> in front of the working directory.
2	Suppresses automount and keeps the original working directory for each machine.

Dracula Reference

Using Dracula

Field 7

A # sign followed by the module name, which is used to assign critical processes to a powerful machine.

Example 1

This parallel file example assigns critical processes to a dedicated node:

```
nickel /net/nickel/dracula_lvs 1 $DRACBIN 4 0 # NODAL LVSCHM
iron /net/iron/dracula_lvs 0 $DRACBIN 1 0
silicon /net/silicon/dracula_lvs 0 $DRACBIN 1 0
```

In this parallel file, modules NODAL and LVSCHM must run on the machine *nickel*.

Example 2

The following example shows a node configuration file.

```
iron      /dracula/test  1  *                               4 0
silicon   /dracula/test  0  *                               3 0
nickel    /dracula/test  *  /net/nickel2/dracula/bin         3 0
```

In this example, *nickel* has an executable path that is different from the path specified in the PROGRAM-DIR command.

Running Dracula Distributed Processing

To run a Dracula distributed processing, follow these steps given below.

Starting a Run

Perform these steps on the master node:

1. Compile the command file using PDRACULA.

PDRACULA generates the following control scripts you can use to run Dracula distributed processing:

- ❑ The *jxpar.com* control script sets up the tasks to send to each node. It copies the necessary files to each node used in the run and creates the *drac-dir* and *drac-link* subdirectories.
- ❑ The *jxexe.com* control script starts a Dracula distributed run.
- ❑ The *jxclean.com* control script clears processes left running after you stop a distributed run with the UNIX *kill -9* command.

PDRACULA uses your node configuration file to generate the following files that control Dracula distributed runs:

- ❑ DRAC-FILE.TAB lists each file needed in the run.
- ❑ DRAC-TASK.TAB lists each task to be performed in the run.
- ❑ DRAC-NODE.TAB lists the nodes to be used in the run.
- ❑ DRAC-SYNC.TAB lists the synchronization points for the run.

For more information, see the “[Distributed Dracula Files](#)” section.

2. To set up the distributed run, type

```
jxpar.com
```

Dracula sets up the tasks to send to each node, taking into account the dependencies listed in DRAC-TASK.TAB and the workstation power ranks listed in DRAC-NODE.TAB.

3. To start the run, type

```
jxexe.com
```

This command starts the run from the beginning.

Note: Most error messages are found in the *.parlog file. From version 4.6.0998 and subsequent versions, the error messages are found in the drac_dir/DRACCLNT.log.

Restarting a Run

If a run stops before it is complete, you can restart it in one of two ways:

- To continue from where the previous run stopped, type

```
jxexe.com 0
```

- To restart from a specific task, enter the task number after the *jxexe.com* command. The `DRAC-TASK.TAB` file lists the task numbers. For example, to start at the third task, type

```
jxexe.com 3
```

Clearing Processes After a Run

After a normal Dracula distributed run, no processes are left running and cleaning is done automatically. The results are copied back into the master working directory.

However, if you use the UNIX *kill -9* command to stop a distributed run, some processes might continue running on the slave nodes.

- To clear any processes still running after you stop a Dracula distributed run, type the following on the master node:

```
jxclean.com
```

Distributed Dracula Files

The following sections describe each of the Dracula distributed files.

DRAC-FILE.TAB File

Each line in the `DRAC-FILE.TAB` file lists a file needed in the run and contains the following two fields:

Field 1

The index number associated with the file.

Field 2

The name of the file.

Here is a sample `DRAC-FILE.TAB` file:

```
1 INPUT.DB
2 9ACTIVE.DAT
```

Dracula Reference

Using Dracula

```
3 9POLY.DAT
4 9GATE.DAT
. . .
. . .
15 ACTIVE.DAT
16 POLY.DAT
17 GATE.DAT
. . .
```

DRAC-TASK.TAB File

Each line in the `DRAC-TASK.TAB` file lists a task to be performed in the run and contains the following six fields:

Field 1

Task number.

Field 2

Task numbers of any tasks that must be completed before this task can begin. This field must end with an asterisk (*).

Field 3

Index numbers of any input files to this task. The `DRAC-FILE.TAB` file lists the files with their index numbers. This field must end with an asterisk (*).

Field 4

Index numbers of files this task generates. These are the same index numbers listed in `DRAC-FILE.TAB`. This field must end with an asterisk (*).

Field 5

Index numbers of files the daemon must copy from the slave to the master server when this task is complete. These are the same index numbers listed in `DRAC-FILE.TAB`. This field must end with an asterisk (*).

Field 6

Broadcast flag. The possible values are 1 and 0. To update `DRACINFO.TXT`, use 1.

Here is a sample `DRAC-TASK.TAB` file:

```
1      *      1*      2*      *      0
2      *      1*      3*      *      1
3      *      .      .      .      1
.      .      .      .      .      .
13     1 4 7*  15 16*  17*      *      .
```


Dracula Reference

Using Dracula

Task number 13 in this sample file has 3 prerequisite tasks: tasks 1, 4, and 7. Its input files are ACTIVE.DAT and POLY.DAT, which are listed with the indexes 15 and 16 in the DRAC-FILE.TAB.

AC-FILE.TAB file. This task generates the GATE.DAT file. The daemon does not need to copy any files back to the master node when task 13 completes.

DRAC-NODE.TAB File

Each line in DRAC-NODE.TAB lists a node to be used in the run and contains the following seven fields:

Field 1

Name of the workstation to be used in the run.

Field 2

Working directory on the workstation to be used in the run.

Field 3

Node type. The possible values are 1, 0, and asterisk (*). For the master node, specify 1. For a slave node, specify 0 or *. The master node must be the first entry.

Field 4

Dracula executable path for this workstation. An asterisk specifies the path listed by the PROGRAM-DIR command in the rules file.

Field 5

Security path. A slash (/) indicates the path listed by the DRACSEC4 environment variable.

Field 6

Power rank of this workstation in relation to the other workstations being used in this run. The most powerful workstation has the highest number. An asterisk specifies a rank of one.

Field 7

Mount type. A soft mount is zero. A hard mount is one.

The following example shows a sample DRAC-NODE.TAB file (explicit soft mount):

```
nickel /net/nickel/dracula/test      * /net/nickel2/dracula/bin / 3 0
iron   /net/iron/dracula/test        1 * / 4 0
silicon /net/silicon/dracula/test    0 * / 3 0
```

Dracula Reference

Using Dracula

The following example shows a sample DRAC-NODE.TAB file (implicit soft mount):

```
nickel /dracula/test * /net/nickel2/dracula/bin / 3 0
iron   /dracula/test 1 * / 4 0
silicon /dracula/test 0 * / 3 0
```

The following example shows a sample DRAC-NODE.TAB file (hard mount):

```
nickel /net/dracula/test * /nickel/dracula/bin / 3 1
iron   /net/dracula/test 1 * / 4 1
silicon /net/dracula/test 0 * / 3 1
```

These sample files list three nodes: *nickel*, *iron*, and *silicon*. Each node has a working directory called *test* under a directory called *dracula*. *Iron* is the master node, identified by the value *1* in the third field.

Iron is also the most powerful workstation because the power rank value in the sixth field, *4*, is higher than the power rank values for the other two nodes.

The asterisk in the fourth field indicates that *iron* uses the path specified by the PROGRAM-
DIR command as its executable path.

The slash in the fifth field indicates that *iron* uses the security path specified by the
DRACSEC4 environment variable.

DRAC-SYNC.TAB File

During a distributed run, Dracula must synchronize twice. During synchronization, Dracula waits for all tasks currently running to complete before beginning the next task.

The first synchronization point is after the last EXPAND. It handles the design's substrate and boundary. Dracula must generate all pseudo layers before it can continue with more advanced operations.

The second synchronization point is immediately before GDS2OUT, APPLEOUT, or CIFOUT. Dracula collects and sorts the summary files generated by each task. Dracula must complete all tasks before it can convert the database.

Each line in the DRAC-SYNC.TAB file lists a synchronization point and contains the following four fields:

Field 1

Synchronization point type: either 1 or 2.

Field 2

Number of the task at which to synchronize.

Field 3

Number of the stage at which to synchronize. This is the stage number listed in the *jxrun.com* executable file.

Field 4

Number of the synchronization file as listed in DRAC-FILE.TAB.

The following example shows a sample DRAC-SYNC.TAB file:

1	11	9	18*
2	131	0	*

Dracula Distributed Processing and the Dracula Graphical User Interface Tool

The Dracula graphical user interface (the `draculaInteractive` tool) supports Dracula distributed processing.

- You need to set the Dracula data path correctly while running the Dracula interactive debugging tool. To get the Dracula data, you have to set the path to `master_node/drac_link` where `master_node` is the directory from which you ran the Dracula distributed job.
- *You should not* use the `/NEXT` command when compiling a Dracula rule file for Dracula distributed processing with PDRACULA.

See the “Note” under the [`KEEPDATA`](#) command section of this manual.

Dracula Distributed Processing in the Dracula 4.51 Release

- Users can specify their own queuing system (`dracq` does not support DD)
- Dracula distributed processing groups layers for SPACING, LOGICAL and SELECT types of operations in order to reduce the overhead.

Dracula Distributed Processing in the Dracula 4.6 release

Dracula distributed processing supports multiple disks storage on master and slave nodes.

Dracula Distributed Processing in the Dracula 4.7 release

A Dracula distributed solution is available for LPE/PRE.

Log file details individual run times for each stage as well as total run time. Throughput of each machine/CPU is displayed for statistics.

Jobs can be optimized by assigning most critical steps to most powerful CPUs.

Dracula Distributed Processing in the Dracula 4.8 release

A new command in this 4.8 release is the MULTICPU command that allows you to run Dracula distributed processing on a multiCPU machine.

Running Dracula in 32-bit or 64-bit Modes

The Dracula directory structure is as follows:

- The wrappers are stored in:

`tools/dracula/bin`

- 64-bit executables are stored in:

`tools/dracula/bin/64bit`

- 32-bit executables with or without a 64-bit equivalent are stored in:

`tools/dracula/bin/32bit`

To enable 64-bit applications use the UNIX environment variable `CDS_AUTO_64BIT` {ALL | NONE | "INCLUDE:list" | "EXCLUDE:list" | "list" }. This environment variable enables you to select 32-bit or 64-bit executables using the following values:

- **ALL:** Invokes all applications as 64-bit, where available. The list of executables available are in your `install_dir/bin/64bit`.
- **NONE:** Invokes all applications as 32-bit.
- **"INCLUDE:list":** Invokes all applications in the list as 64-bit, where available.
- **"EXCLUDE:list":** Invokes all applications as 64-bit, where possible, except the applications in the list.
- **"list":** Invokes only the executables included in the list, if available, as 64-bit. "list" is a list of case-sensitive executable names delimited by a semi-colon(;) or a colon(:). Providing the list without a preceding keyword works the same as using the keyword `INCLUDE`.

Examples:

```
setenv CDS_AUTO_64BIT "pipo;layout.exe"
setenv CDS_AUTO_64BIT "EXCLUDE:si"
```

All 64-bit executables are controlled by a wrapper executable. The wrapper invokes the 32-bit or 64-bit executables depending on how the `CDS_AUTO_64BIT` environment variable is set, or whether the 64-bit executable is installed. The wrapper also adjusts the paths before invoking the 32-bit or 64-bit executables.

If you do not set the `CDS_AUTO_64BIT` environment variable then the 32-bit executable is invoked by default.

Dracula Reference

Using Dracula

Additionally, Dracula uses a separate environment variable called `DRACULA_AUTO_64BIT` to enable 64-bit mode. This environment variable specifies the addressing mode only for Dracula, and overrides the `ALL` or `NONE` settings of the `CDS_AUTO_64BIT` environment variable for Dracula. `DRACULA_AUTO_64BIT` recognizes the following variable values:

- `ALL`: Invokes all applications as 64-bit, where available. The list of executables available are in your `install_dir/bin/64bit`.
- `NONE`: Invokes all applications as 32-bit.
- `"INCLUDE:list"`: Invokes all applications in the list as 64-bit, where available.
- `"EXCLUDE:list"`: Invokes all applications as 64-bit, where possible, except the applications in the list.
- `"list"`: Invokes only the executables included in the list, if available, as 64-bit. `"list"` is a list of case-sensitive executable names delimited by a semi-colon(;) or a colon(:). Providing the list without a preceding keyword works the same as using the keyword `INCLUDE`.

Examples:

```
setenv DRACULA_AUTO_64BIT "SPACING;LOGLVS;EXPAND"  
setenv DRACULA_AUTO_64BIT "EXCLUDE:PDRACULA"
```

Important

Some important points are:

- ❑ Entry points to both the 32-bit and 64-bit addressing executables are exclusively through the wrapper programs.
- ❑ You must not include directories containing the associated "real" non-wrapper executables in their UNIX path.
- ❑ There is a consistent use model in both 32 bit and "automatic high capacity" modes (for example, the same UNIX path).
- ❑ Prevents pilot errors (for example, attempting to run a 64-bit executable in a 32-bit environment).
- ❑ Consistent with Solaris 7 32 or 64-bit model.
- ❑ Requires the same licenses for a 32-bit or 64-bit operation

Checking Design Rules (DRC)

This chapter discusses Design Rule Checking in Dracula, focusing on the following topics:

- [“Design Rules Checker”](#) on page 79
- [“DRC Example”](#) on page 81
- [“Hierarchical and Multilevel Hierarchical Design Rules Checker”](#) on page 110

Design Rules Checker

The Dracula Design Rules Checker (DRC) performs checks on layout geometries to ensure the layout can be manufactured under a specific IC process technology. Using DRC during the IC design process guarantees manufacturability of the layout and offers high yield for the process used.

This chapter describes the DRC commands and shows a detailed example of a typical DRC run. It also describes hierarchical DRC (HDRC) and multilevel hierarchical DRC (MDRC).

For more information about how to write different types of DRC checks, see [“Writing DRC Checks”](#) in the “Writing Rules for Dracula” chapter of the *Dracula User Guide*.

DRC Commands

DRC includes an extensive set of design rules, including conjunctive rules. DRC commands are process technology independent, so you can write different Dracula rules files for different process technologies. The DRC commands can be grouped into three categories:

- Logical operation commands create new layers using combinations of original layers, previously created layers, or imported layers.
- Sizing commands perform complex resizing and checking before producing final mask geometries. Dracula can resize layouts based on mask or silicon size and can accommodate the design rule requirements of new or modified process technologies.

Dracula Reference

Checking Design Rules (DRC)

- Spacing commands check geometric spacing and sizing on polygons in the original and created layers. Dracula can flag and process errors further using conjunctive rules, the *R* option, or the *R'* options.

The tab divider page at the beginning of this chapter lists all DRC commands. For detailed information about each DRC command, refer to [Chapter 13, “Operation Block Commands.”](#)

Conjunctive Rules

You can join some commands so that one command processes the errors output by a prior command. A conjunctive rule set is a series of commands that generates a single output.

To write a conjunctive rule, put an ampersand (&) at the end of a command line. The output of that command is then available to any command that follows in the conjunctive set. Therefore, you can write a series of dependent checks. You must end a conjunctive rule set with an OUTPUT statement.

The ampersand generates error flags for the corresponding input layers. Flags for *layer-a* segments are marked on *layer-a*, and flags for *layer-b* segments are marked on *layer-b*. These are conjuncted layers. Subsequent checks on a conjuncted layer filter these errors until any remaining errors are output. Use the renamed conjuncted layer in later checks so you do not have to recompute these error flags. Except for layers you rename, conjuncted layers revert back to the original layer after the conjunctive rule is completely processed.

The ampersand (&) is a reserved character, and anything that follows it on the same line is considered a comment. You cannot specify an output cell on the same line as an ampersand.

The following is an example of a conjunctive rule using spacing checks:

```
EXT METAL LT 4.0 &  
LENGTH METAL GT 5.0 OUTPUT 10
```

The EXT command checks for geometries on the METAL layer with an external spacing of less than 4.0 units and flags them as errors. The LENGTH command then checks the error flags generated by the EXT command, finds the error flags that are greater than 5.0 units in length, and outputs them to layer 10.

DRC Example

This section shows an example of a typical DRC run, including descriptions of the technology, the Description, Input-layer, and Operation block commands, and printed and graphic DRC output.

This DRC example uses one cell of a memory chip. The following table lists information about the cell:

Technology	3.5-micron NMOS
Physical size	410 x 260 microns
Number of transistors	160

Dracula Reference

Checking Design Rules (DRC)

Number of polygons	1100
CAD graphics system	Applicon 860
Graphics unit scale	0.0000152588 micron (1 micron = 65536 VLSI units)

Description Block

The Description block contains the input/output devices and file names, CAD layout graphics system, graphics unit scale factor, cell (or portion of cell) under investigation, and execution mode. The commands for the Description block are as follows:

```
DESCRIPTION
;
PRIMARY = demo           ; Primary cell name
SYSTEM = cadence         ; CAD graphics system
INDISK = demodrc         ; Input file name
OUTDISK = demout         ; Output file name
SCALE = 0.0000152588 mic ; Graphics scale factor
                        ; (or Rev-Scale = 65536 mic)
PRINTFILE = drcprt       ; Output print file
MODE = exec now          ; Execution mode
LISTERROR = yes          ; Extended error print
CHECK-PATH = yes         ; Checks expanded paths
RESOLUTION= 0.125 mic    ; Minimum layout increment
*END
```

Input-Layer Block

The Input-Layer block equates the system layer number with your own symbolic name and defines the mask sequence for the major layers in the technology process.

Note: The *srcdrn* layer is not yet defined. A logical operation in the Operation block creates this layer.

```
*INPUT-LAYER
;
diff      = 1   ; Diffusion input layer
epi       = 4   ; Epi contact input layer
poly      = 5   ; Polysilicon input layer
cont      = 6   ; Metal contact input layer
metal     = 7   ; Metal input layer
natural   = 9   ; Natural transistor implant layer
harddep   = 10  ; Hard depletion transistor implant
softdep   = 11  ; Soft depletion transistor implant
;

CONNECT-LAYER = srcdrn poly metal ;order of layer process
;
*END
```

Operation Block

As a result of the various chemical processes in the IC fabrication process, the silicon tends to expand or contract. Therefore, the IC masks might have to resize particular geometries to produce the after-process silicon geometries specified by each IC fabrication process. Because of the difference between the final silicon geometries and the IC masks, there are two ways to produce the IC layout.

- Drawing based on mask size

The designer develops the IC layout based on the mask size required to produce the final after-process silicon geometries.

- Drawing based on silicon size

The designer develops the IC layout based on the final after-process silicon geometries.

Using the mask-size approach, the process engineer generates a set of mask design rules for the layout designer. These rules are based on the silicon design rules and the process deviation factor on the IC masks. The advantage of the mask-size approach is that the mask shop can use the layout database to produce the IC masks without resizing. The disadvantages of this approach are that mask design rules tend to be more complicated, layout has less physical meaning, and electrical parameters such as capacitance, resistance, and W/L become more difficult to compute.

The design rules in the following examples are based on the mask-size approach. In the following rules, the dimensions for spacing are different in the “field” and “source-drain” directions because the polysilicon layer and the diffusion layer have different oversize and undersize amounts during the IC process.

The following Operation block examples are based on a 3.5 NMOS process. The design rules and a sample rules file are shown for each example.

Gate and Depletion Rules

Typical mask design rules for the depletion implant mask (or enhancement blocking mask) and the gates are as follows:

- Minimum overlap of transistors and capacitors for depletion implant mask (or enhancement blocking mask)

In the field direction	2.0 microns
In the source or drain direction	3.0 microns

Dracula Reference

Checking Design Rules (DRC)

- Minimum external spacing between depletion implant mask (or enhancement blocking mask) and transistors

In the field direction	2.5 microns
In the source or drain direction	3.5 microns

Typical mask design rules for the *epi* contacts and transistors are as follows:

Minimum *epi*-contact spacing to a transistor edge

In the field direction	3.0 microns
In the source or drain direction	4.0 microns

To specify these rules, use the following commands in the Operation block:

```
*OPERATION
;
; FORM TRANSISTORS OVERSIZED IN SOURCE/DRAIN DIRECTION
;
SIZE poly BY 1 ovpoly ; Oversize polysilicon to expand channels in s/d
direction.
AND ovpoly diff ovpodif ; Form ovpodif channels oversized in s/d direction.
NOT ovpodif epi xtor ; Form xtor transistors oversized in s/d direction.
;
; FORM DEPLETION IMPLANT AND ENHANCEMENT BLOCKING
;
OR natural harddep eblk ; Form enhancement blocking layer
OR harddep softdep depl ; Form depletion implant layer
;
ENC[T] xtor eblk LT 2.0 OUTPUT drc02 41 ; Enhancement
ENC[T] xtor depl LT 2.0 OUTPUT drc03 51 ; Depletion
;
EXT[T] eblk xtor LT 2.5 OUTPUT drc02 42 ; Enhancement
EXT[T] depl xtor LT 2.5 OUTPUT drc03 52 ; Depletion
;
EXT[T] epi xtor LT 3.0 OUTPUT drc04 61
;
ENC[T] xtor&pxtor ovpoly LT .001 & ; Or in the direction of diffusion.
EXT[T] &pxtor epi LT 3.0 OUTPUT drc05 62
; diffusion conjunctive rename of xtor layer to &pxtor fs
```

Diffusion Separation Rules

Typical mask design rules for the external separation of the diffusion geometries are as follows:

Dracula Reference

Checking Design Rules (DRC)

Minimum diffusion external separation for the diffusion (field) mask when overlapped by an *epi* contact

Limiting case, no *epi* overlap 4.0 microns

When either of the diffusion lines has an *epi* contact 4.5 microns

When both of the diffusion lines have an *epi* contact 5.5 microns

To specify these rules, use the following commands in the Operation block:

```
; Operation block, continued
;
; FORM DIFFUSION EPI-CONTACT OVERLAP LAYER
; diff = Diffusion layer
; epi = Epi contact layer
;
AND diff epi diffep
;
EXT diff LT 4.0 OUTPUT drc01 32
EXT diff diffep LT 4.5 OUTPUT drc01 33
EXT diffep LT 5.5 OUTPUT drc01 34
```

Spacing Rules

This class of design rules includes the minimum spacing required to avoid shorts caused by the IC fabrication process. These rules generally involve external spacing checks between geometries, which tend to produce many false errors unless there is an effective electrical node capability. You can ignore these checks if the geometries are part of the same electrical node, even if the spacing is less than the allowable dimension.

Typical external spacing design rules are as follows:

- Minimum polysilicon-to-diffusion spacing is 1.0 micron (if not in the same electrical node).
- Minimum polysilicon metal contact-to-diffusion spacing is 4.0 microns (if not in the same electrical node).
- Minimum diffusion metal contact-to-polysilicon spacing is 3.5 microns (if not in the same electrical node).
- Minimum diffusion metal contact-to-transistor spacing in the direction of diffusion is 4.0 microns.

To specify these rules, use the following commands in the Operation block:

```
; Operation block, continued
;
; FORM THE SOURCE-DRAIN LAYER
; poly = Polysilicon layer
; diff = Diffusion layer
```

Dracula Reference

Checking Design Rules (DRC)

```
; cont = Metal contact layer (mc)
; epi  = Epi contact layer
;
NOT diff poly srcdrn  Form srcdrn = diff - poly
;
; ELECTRICAL NODE DEFINITION
;
CONNECT      metal      poly      BY cont
CONNECT      metal      srcdrn    BY cont
CONNECT      poly       srcdrn    BY epi
;
EXT[N] poly srcdrn LT 1.0 OUTPUT drc05 73
;
; FORM LAYERS OVERLAPPING THE METAL-CONTACTS
;
AND poly      cont mc poly  Form mcpoly polysilicon metal contact
AND srcdrn    cont mc srdrn Form mcsrdrn source-drain metal contact
;
EXT[N]        mc poly      srcdrn      LT 4.0  OUTPUT drc06 86
EXT[N]        mc srdrn     poly        LT 3.5  OUTPUT drc06 87
;
EXT[T]        &pxtor       mc srdrn     LT 3.0  OUTPUT drc06A 87
```

Enclosure Rules

Sometimes the dimension of the enclosure of a metal contact by the metal layer is larger in the direction of current than in the other direction.

The design rule is the minimum polysilicon-metal contact enclosed by polysilicon enclosure spacing and the minimum metal spacing is 6.0 microns.

In the direction of current	2.0 microns
In all other directions	1.0 micron

To specify this rule, use the following commands in the Operation block:

```
; Operation block, continued
;
; poly      = Polysilicon layer
; metal     = Metal layer
; cont      = Metal contact layer (mc)
;
ENC[TO]      cont poly      LT 1.0  OUTPUT drc06 84
;
SIZE         cont BY 3.0  ovmc ; Form metal-entry into metal-contact
AND          ovmc metal   xcmet ; Form metal-direction
ENC[PC]      cont xcmet   RANGE 2.99 3.01 &
ENC[PCTO]    cont poly    LT 2.0  OUTPUT drc06 85
```

Polysilicon-to-Diffusion Spacing Rules

Most processes require design rules that specify the minimum external spacing between the polysilicon and diffusion geometries. However, when metal crosses adjacent polysilicon and diffusion, the spacing requirement increases. This increase prevents metal from breaking during processing when it runs over the valleys formed by diffusion and polysilicon layers.

Specify the design rule minimum polysilicon-to-diffusion spacing as follows:

When crossed by metal layer	2.0 microns
All other instances	1.0 micron

To specify this rule, use the following commands in the Operation block:

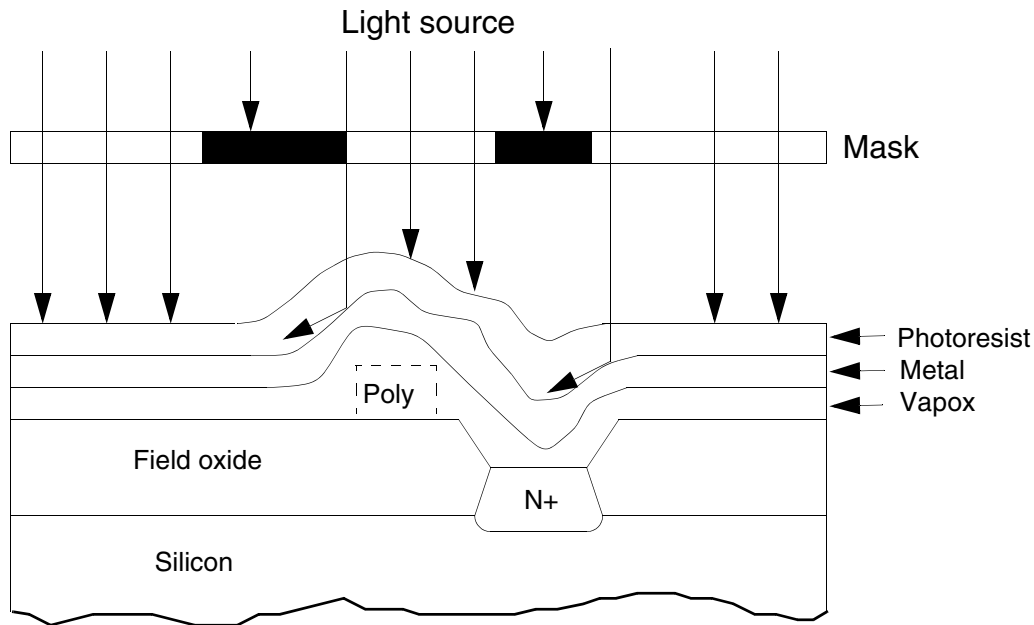
```
; Operation block, continued
;
; poly  = Polysilicon layer
; diff  = Diffusion layer
; metal = Metal layer
;
; FORM LAYERS OVERLAPPING THE METAL LAYER
;
EXT[R] poly diff LT 2.0 valley ; Form the valley layer
AND metal valley metval OUTPUT drc05 74
```

Metal Reflection Rules

The next class of design rules includes checks necessary to prevent adverse effects caused by reflecting light during the IC fabrication process. An adverse effect is often referred to as a “misalignment” or a “hazard.”

As layers of various materials build up during the IC fabrication process, peaks and valleys forming on the IC cause angular light reflection. For example, if shaded areas of the IC mask for the metal layer represent the after-process metal geometries, light might reflect off the deposited metal when light passes through the clear areas in the mask. The following figure, “Cause of Metal Reflection,” shows this lighting effect.

Cause of Metal Reflection

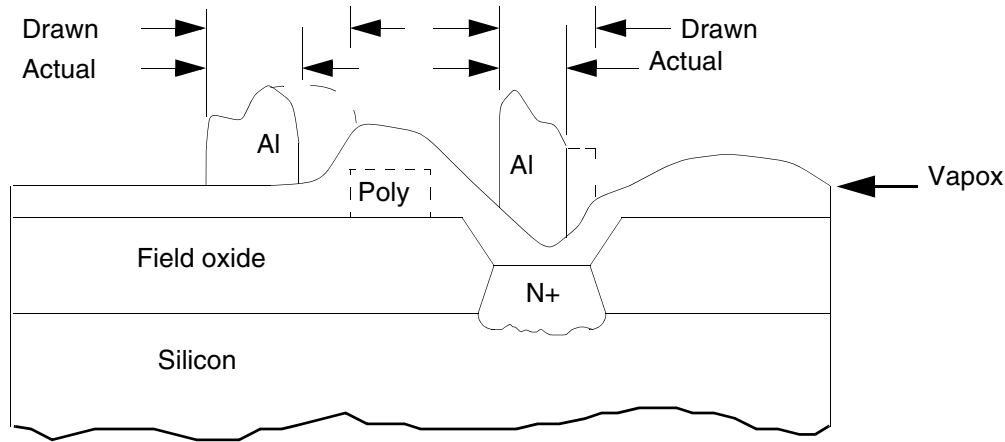


In the following figure, “After-Process Geometries,” the areas under the shaded portion of the IC mask are exposed to light, which results in undersized geometries. The rules used in this example are based on the requirements of a positive photoresist process.

Dracula Reference

Checking Design Rules (DRC)

After-Process Geometries



The mask design rules for metal reflection define the minimum width of the metal as follows:

Limiting case (no reflection)	5.0 microns
When metal overlaps polysilicon by less than 2.0 microns	6.0 microns
When external separation between polysilicon and metal is less than 2.0 microns	6.0 microns
When diffusion extension over metal is less than 2.0 microns	6.0 microns
When enclosure spacing of metal to diffusion is less than 2.0 microns	6.0 microns

To specify these rules, use the following commands in the Operation block:

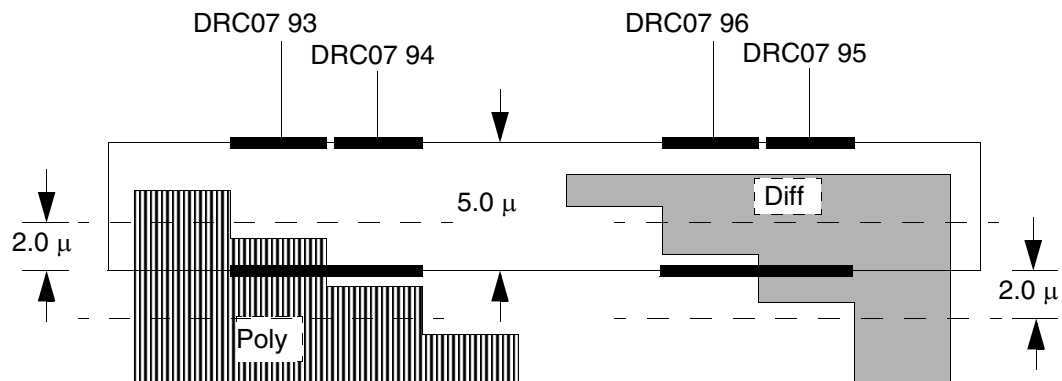
```
; Operation block, continued
;
; poly = Polysilicon layer
; diff = Diffusion layer
; metal = Metal layer
; polymet = Polysilicon and metal layer
;
; LIMITING CASE NO REFLECTION
;
WIDTH      metal      LT 5.0      OUTPUT drc07 92
;
; METAL OVER POLYSILICON REFLECTION CHECK
;
INT[P]      poly      metal      LT 2.0 &
INT[P]      metal     metal[O]   LT 6.0      OUTPUT drc07 93
```

Dracula Reference

Checking Design Rules (DRC)

```
EXT[TP]    metal    poly      LT 2.0 &  
INT[P]     metal    metal[O]  LT 6.0  OUTPUT drc07 94  
;  
; METAL OVER DIFFUSION REFLECTION CHECK  
;  
ENC[TP]    metal    diff      LT 2.0 &  
INT[P]     metal    metal[O]  LT 6.0  OUTPUT drc07 95  
;  
ENC[TP]    diff     metal     LT 2.0 &  
INT[P]     metal    metal[O]  LT 6.0  OUTPUT drc07 96
```

Metal Reflection Rules



Dracula Reference

Checking Design Rules (DRC)

Executing DRC from a Design Rule Library

The design rules necessary to comply with the technology in this example are shown in the following Operation block:

```
*OPERATION
;
; FORM THE SOURCE-DRAIN LAYER
;
NOT      diff      poly      srcdrn
;
; ELECTRICAL NODE CONSTRUCTION
;
CONNECT  metal      poly      BY cont
CONNECT  metal      srcdrn    BY cont
CONNECT  poly       srcdrn    BY epi
;
; LAYER RESIZING OPERATIONS
;
SIZE      poly      BY 1.0     ovpoly
STAMP     ovpoly    BY        poly
SIZE      epi       BY 3.0     ovepi
;
; FORM TRANSISTORS OVERSIZED IN THE SOURCE-DRAIN DIRECTION
;
AND       ovpoly    diff      ovpodif
NOT       ovpodif   epi       xtor
;
; FORM DEPLETION IMPLANT MASK AND ENHANCEMENT BLOCKING MASK ; LAYERS
;
OR        harddep   softdep   depl
OR        natural   harddep   eblk
;
; DESIGN RULE CHECKS
;
; DIFFUSION MASK DRC
;
WIDTH     diff      LT 4.0     OUTPUT drc01 31
EXT       diff      LT 4.0     OUTPUT drc01 32
;
; DIFFUSION WITHIN EPI-CONTACT DRC
;
AND       diff      epi       diffep
;
EXT[P]    diff      diffep    LT 4.5  OUTPUT drc01 33
EXT[P]    diffep    LT 5.5    OUTPUT drc01 34
;
; ENHANCEMENT BLOCKING MASK DRC
;
ENC[T]    xtor      eblk      LT 2.0  OUTPUT drc02 41
EXT[T]    eblk      xtor      LT 2.5  OUTPUT drc02 42
EXT[P]    eblk      LT 3.0    OUTPUT drc02 43
;
; DEPLETION IMPLANT MASK DRC
;
AND       poly      diffep    podifep
;
ENC[T]    xtor      depl      LT 2.0  OUTPUT drc03 51
EXT       depl      xtor      LT 2.5  OUTPUT drc03 52
EXT[P]    depl      LT 3.0    OUTPUT drc03 53
```

Dracula Reference

Checking Design Rules (DRC)

```

AREA          podifep      RANGE 0.0 20.0  OUTPUT drc03 54
ENC           diffep       epi          LT 1.0  OUTPUT drc03 55
;
; EPI-CONTACT DRC
;
AND           poly         ovepi         ovepi
AND           srcdrn       ovepi         sdovepi
;
EXT[T]        epi          xtor          LT 3.0  OUTPUT drc04 61
EXT[O]        epi          mc            LE 0.0  OUTPUT drc04 62
EXT[P]        epi          LT 3.0        OUTPUT drc04 63
EXT[N]        ovepi        sdovepi       LT 3.0  OUTPUT drc04 64
ENC[T]        xtor&pxtor   poly          LT.001 &
EXT[T]        &pxtor       epi          LT 3.0  OUTPUT drc05 62
;
; POLYSILICON MASK DRC
;
WIDTH         poly         LT 3.5        OUTPUT drc05 71
EXT           poly         LT 4.0        OUTPUT drc05 72
EXT[N]        poly         srcdrn        LT 1.0  OUTPUT drc05 73
;
; METAL BREAK
;
EXT[R]        poly         diff          LT 2.0  valley
AND           metal        valley        metval  OUTPUT drc05 74
EXT           metval       LT 2.0        OUTPUT drc05 74
;
ENC           xtor         poly          LT 3.0  OUTPUT drc05 75
;
; METAL-CONTACT DRC
;
AND           poly         cont          mcpoly
AND           srcdrn       cont          mcsrdn
;
AREA          cont         RANGE         0. 9.  OUTPUT drc06 81
ENC           cont         metal        LT 1.0  OUTPUT drc06 82
ENC           cont         diff         LT 2.5  OUTPUT drc06 83
ENC           cont         poly         LT 1.0  OUTPUT drc06 84
SIZE          cont         BY 3.0        ovmc
AND           ovmc         metal        xcmet
ENC[PC]       cont         xcmet        RANGE 2.99 3.01 &
ENC[PCTO]     cont         poly         LT 2.0  OUTPUT drc06 85
WIDTH         xcmet        LT 6.0        OUTPUT drc06 86
EXT[N]        mcpoly       srcdrn        LT 4.0  OUTPUT drc06 87
EXT[N]        mcsrdn       poly         LT 3.5  OUTPUT drc06 88
EXT[T]        &pxtor       mcsrdn       LT 3.0  OUTPUT drc06a 87
EXT           cont         LT 4.0        OUTPUT drc06 89
;
; METAL DRC
;
EXT           metal        LT 4.0        OUTPUT drc07 91
WIDTH         metal        LT 5.0        OUTPUT drc07 92
;
; METAL OVER POLYSILICON REFLECTION
;
INT[P]        poly         metal        LT 2.0  &
INT[P]        metal        metal[O]     LT 6.0  OUTPUT drc0 7 93
EXT[TP]       metal        poly         LT 2.0  &
INT[P]        metal        metal[O]     LT 6.0  OUTPUT drc0 7 94
;
; METAL OVER DIFFUSION REFLECTION DRC

```

Dracula Reference

Checking Design Rules (DRC)

```
;
ENC[PT]      metal      diff      LT 2.0 &
WIDTH        metal      metal[O]   LT 6.0  OUTPUT drc07 95
ENC[PT]      diff       metal      LT 2.0 &
WIDTH        metal      metal[O]   LT 6.0  OUTPUT drc07 96
;
* END
```

After you create the Description, Input-Layer, and Operation blocks, you can store the blocks in one or more files. The following examples are files you can use:

Description block *drcdesc.com*
Input-Layer block *drcinpl.com*
Operation block *drcoper.com*

You can then read the files into the preprocessor using the following PDRACULA commands:

```
/GET drcdescNOLIST
/GET drcinplNOLIST
/GET drcoperNOLIST
/FINISH
```

Alternatively, you can concatenate these three files into a single file, such as *drc.com*, and enter the file as follows:

```
/GET drcNOLIST
/FINISH
```

Direct user interface in the Dracula system takes place in the preprocessor, PDRACULA, which is the outermost shell of the verification system. The preprocessor provides execution control and most file management functions.

To start PDRACULA, type the following command:

```
PDRACULA
```

The following example shows how to submit a DRC job to PDRACULA. Text you type is shown in **bold**.

```
*****
*/N* DRACULA3 (REV. 4.3 /SUN-4 /GENDATE: 14-AUG-95/07)
*** (Copyright 1995, Cadence) ***
*/N* EXEC TIME =11:35:40 DATE = 16-AUG-95
*****
:GET drc NO
:FINISH
**Note: CREATING DEFAULTING CNames-CSEN TO "NO" WHEN SYSTEM=APPLICON
**CREATING : COMMAND FILE: jxrun.com
```

Dracula Reference

Checking Design Rules (DRC)

**NOTE: THIS JOB HAS 28 STAGES

END OF DRACULA COMPILATION

Developing and Executing a Design Rule

The preprocessor contains a command language with which you can develop technology processes for future use in production environments. The following example demonstrates the development of the metal-over-polysilicon reflection check. This example is an interactive session with the preprocessor.

Note: You must read in a file or type in the Dracula commands before you can output this listing.

Text you type is shown in **bold**.

To start PDRACULA, type the following:

```
PDRACULA

:/LIST ALL UNSORTED

*INPUT-LAYER
POLY      =      5
METAL     =      7
*END
*OPERATION
AND       POLY      METAL  POLYMET
WIDTH     POLYMET   LT 2    &
ENC[T]    POLYMET   METAL  LT 0.001  &
WIDTH     METAL     LT 6    OUTPUT   drc07 93
*END

:/STORE yourfile

FILE: YOURFILE EXISTS, OVERWRITE (Y/N) ? Y

:/FINISH

SCALE : .0000152588 MIC (or Rev-Scale = 65536 MIC)
RESOLUTION : .125 MIC
PRIMARY CELL : demo
INPUT-FILE-NAME
(INDISK NAME) : INDISK=demodrc
OUTPUT-FILE-NAME
(OUTDISK NAME) : OUTDISK=demout
SYSTEM : CADENCE
MODE : EXEC NOW

Note:   CREATING PRECHK FILE      :JXPRECHK.COM
        AND RULES FILE           :JXRUN.COM

END OF DRACULA COMPILATION
FORTRAN STOP
```

Dracula Reference

Checking Design Rules (DRC)

```
Previous logical name assignment replaced
Job 456 entered on queue SYS$BATCH
WITHOUT PRECHECK, THE DRACULA JOB IS SUBMITTED
```

Viewing the Results

This section contains illustrations of system-generated printed and graphic output of violations. Ten violations are shown. The output cells containing the violations are shown in the examples on the following pages. The violations are the following:

- DRC0132 External Spacing of Diffusion
- DRC0351 XTOR Enclosed by Depletion
- DRC0352 Depletion to Enhanced XTOR in Field Dir
- DRC0354 Minimum Gate Area When within EPI
- DRC0464 Poly to Diffusion Near EPI
- DRC0573 Poly to Diffusion (Node Option)
- DRC0574 Poly to Diffusion When Crossed by Metal
- DRC0685 Metal-Contact between Metal and Poly
- DRC0687 Metal-Contact within Diffusion to Poly
- DRC0792 Width of Metal

Viewing the Summary File

The first step in evaluating a DRC analysis is to scan through the Violation/ Output Cell Summary using an online editor like *vi* or *emacs*. The summary contains a definition of the primary cell, cell name, layer number, windowed location in physical units, and number of line-segment violations.

Text you type is shown in **bold**. To view the file in *vi*, type the following command at a UNIX command line:

```
vi Printf.sum
```

Dracula displays the following information:

```
*****
* * * * *          D R A C U L A          * * * * *
* * * * *          *** COPYRIGHT 1995***   * * * * *
```


Dracula Reference

Checking Design Rules (DRC)

* * * * * RELEASE 4.3 / 1995 * * * * *
* *

BEGIN AT TIME=12:00:00 DATE=17-AUG-95
CELL DRC0241 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0243 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0353 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0355 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0462 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0463 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0562 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0682 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0684 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0686 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0688 SKIPPED BECAUSE OF NO OUTPUT DATA
CELL DRC0796 SKIPPED BECAUSE OF NO OUTPUT DATA

----- OUTPUT CELL SUMMARY -----

CELL NAME	LAYER #	# OF POLYGON W I N D O W	LINE SEGMENTS			
DRC0131	31	17.000	25.500	282.500	233.500	70
DRC0132	32	90.500	40.500	112.500	172.000	10
DRC0133	33	30.000	20.000	275.500	242.000	42
DRC0134	34	111.000	15.500	124.500	156.500	24
DRC0242	42	123.000	184.500	125.000	208.500	4
DRC0351	51	41.000	156.500	42.500	174.500	2
DRC0352	52	123.000	184.500	125.000	208.500	4
DRC0354	54	98.000	29.000	351.500	169.500	3
DRC0461	61	30.000	92.000	31.625	95.500	2
DRC0464	64	29.500	19.500	275.500	161.500	34
DRC0571	71	302.000	.000	350.000	262.000	6
DRC0572	72	41.000	7.000	286.000	257.000	24
DRC0573	73	261.000	35.500	266.500	226.500	20
DRC0574	74	9.000	15.000	150.500	247.000	28
DRC0575	75	30.000	90.625	36.000	94.125	4
DRC0681	81	86.500	.000	348.500	262.000	14
RC0683	83	36.000	16.000	288.000	265.000	14
DRC0685	85	369.000	6.000	370.000	256.000	8
DRC0687	87	41.000	44.500	321.500	257.000	10
DRC0791	91	142.000	.000	411.000	265.000	10
DRC0792	92	69.000	.000	395.000	262.000	14
DRC0793	93	364.000	128.500	393.500	133.500	4
DRC0794	94	75.000	79.000	267.375	221.000	6
DRC0795	95	12.000	152.500	18.000	160.000	4
PRIMARY CELL=	OUTDEMO					
WINDOW	=	9.000	.000	411.000	265.000	

***** PROBLEM GEOMETRIES LISTING *****

CELL: MB297 LAYER: 2 PROBLEM GEOMETRIES:ACUTE -PROCESSED
-1123.00 233.00 -1123.00 187.00 -1254.00 187.00 -1254.00 202.00
-1275.00 202.00 -1275.00 187.00 -1384.00 187.00 -1384.00 233.00
-1373.00 233.00 -1373.00 227.00 -1293.00 227.00 -1293.00 233.00
-1248.00 233.00 -1248.00 227.00 -1168.00 227.00 -1168.00 233.00
-1168.00 228.00 -1168.00 233.00 -1123.00 233.00
CELL: BLK1 LAYER: 9 PROBLEM GEOMETRIES:OPEN POLYGON -DISCARDED

- 380.00 282.00 -379.90 282.00 -1337.00 299.00 -1127.90 399.00
-1108.90 380.00

***** END OF PROBLEM GEOMETRIES LISTING *****

Dracula Reference

Checking Design Rules (DRC)

Viewing the Log Files

To print a detailed listing of the design rule violations in the *.log* and *.drc* files, use the LISTERROR command in the Description block.

The listing of design rule violations shown below lists the type of checks performed, names of the layers, value of the checks, output layer numbers, and output cell names. For each violation, the x,y coordinates of the two line segments are printed in physical units.

```
EXTERNAL CHECK - DIFF SPACING = 4.000 ( 32.0 DBU)
OUTPUT LAYER # = 32 OUTPUT CELL NAME = DRC0132
```

```
ERR  --- LAYER DIFF ---          --- LAYER DIFF ---

#   X1      Y1      X2      Y2      X1      Y1      X2      Y2
1  90.500  40.500  90.500  43.500  94.000  44.000  94.000  44.500
2  90.500  40.500  90.500  43.500  94.000  44.500  94.000  56.500
3  90.500  87.500  90.500  90.500  94.000  74.500  94.000  86.500
4  90.500  87.500  90.500  90.500  94.000  86.500  94.000  87.000
5 109.000 160.000 109.000 172.000 112.500 157.500 112.500 162.000
```

Viewing the Violations

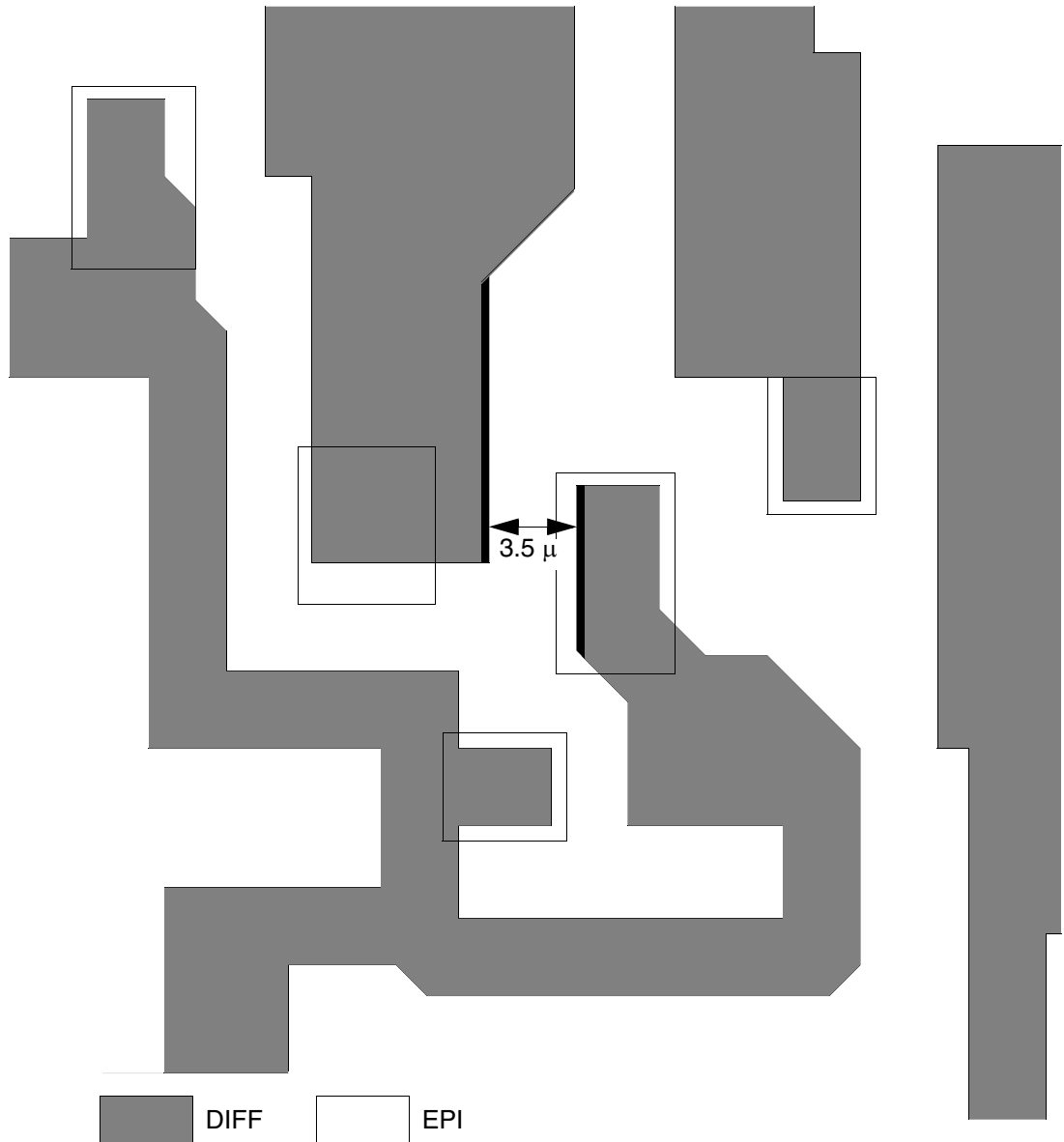
The examples on the following pages show the graphic output for the violations previously defined. Each page shows an example of a layout violation and the commands that flagged the violation.

Dracula Reference

Checking Design Rules (DRC)

DRC0132 External Spacing of Diffusion

EXT diff LT 4.0 OUTPUT drc01 32

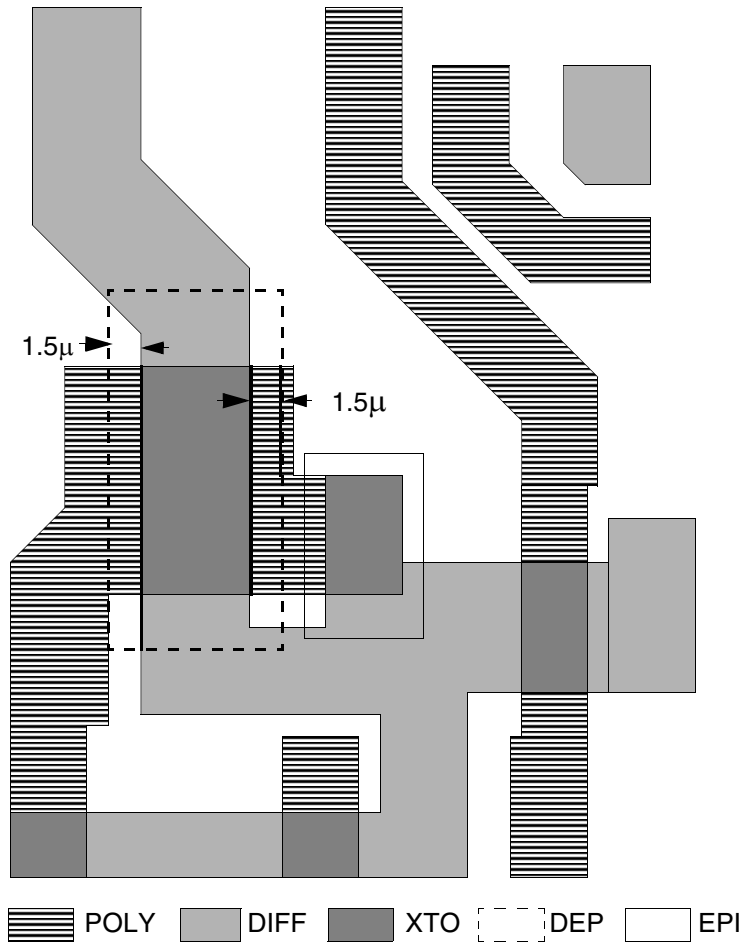


Dracula Reference

Checking Design Rules (DRC)

DRC0351 XTOR Enclosed by Depletion

ENC[T] xtor depl LT 3.0 OUTPUT drc03 51



Dracula Reference

Checking Design Rules (DRC)

DRC0352 Depletion to Enhanced XTOR in Field Dir

EXT diff xtor LT 2.5 OUTPUT drc03 52

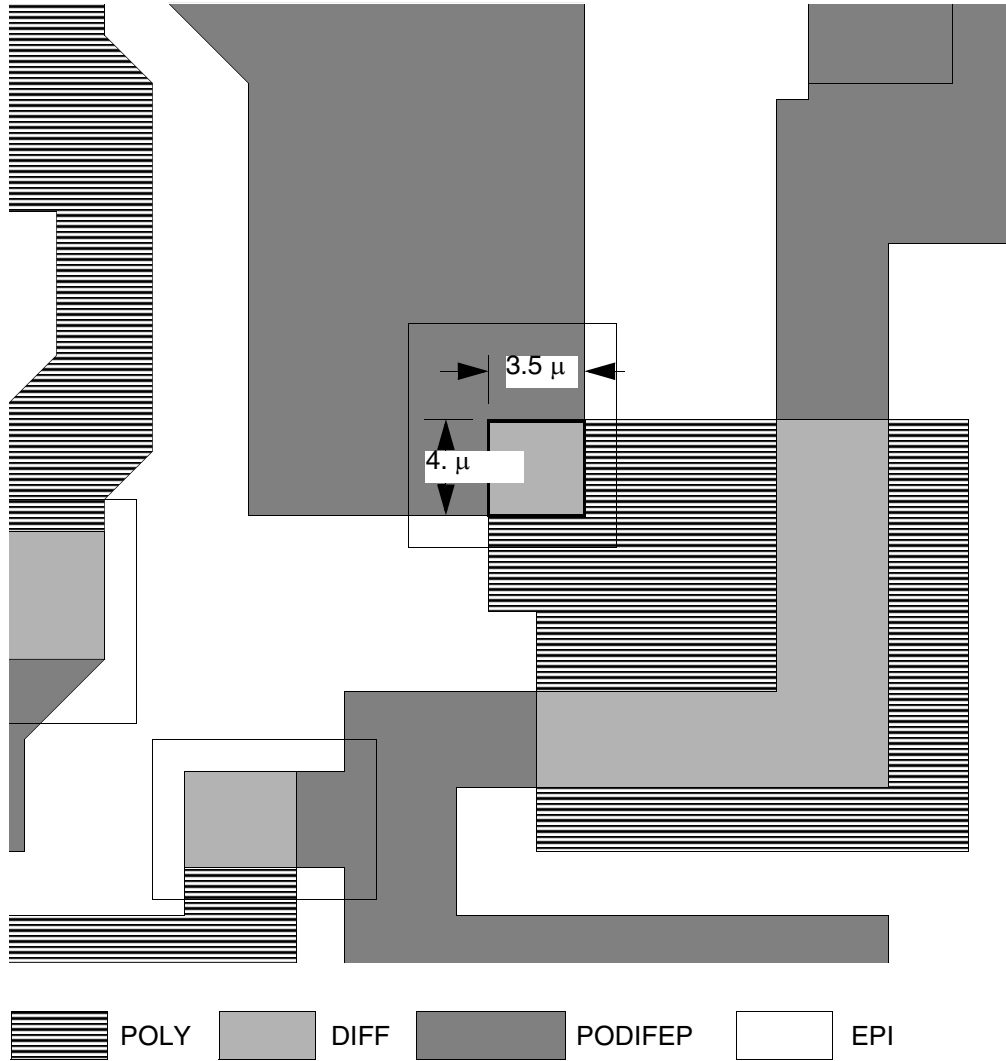


DRC0354 Minimum Gate Area When Within EPI

Dracula Reference

Checking Design Rules (DRC)

AREA podifep RANGE 0 20 OUTPUT drc03 54

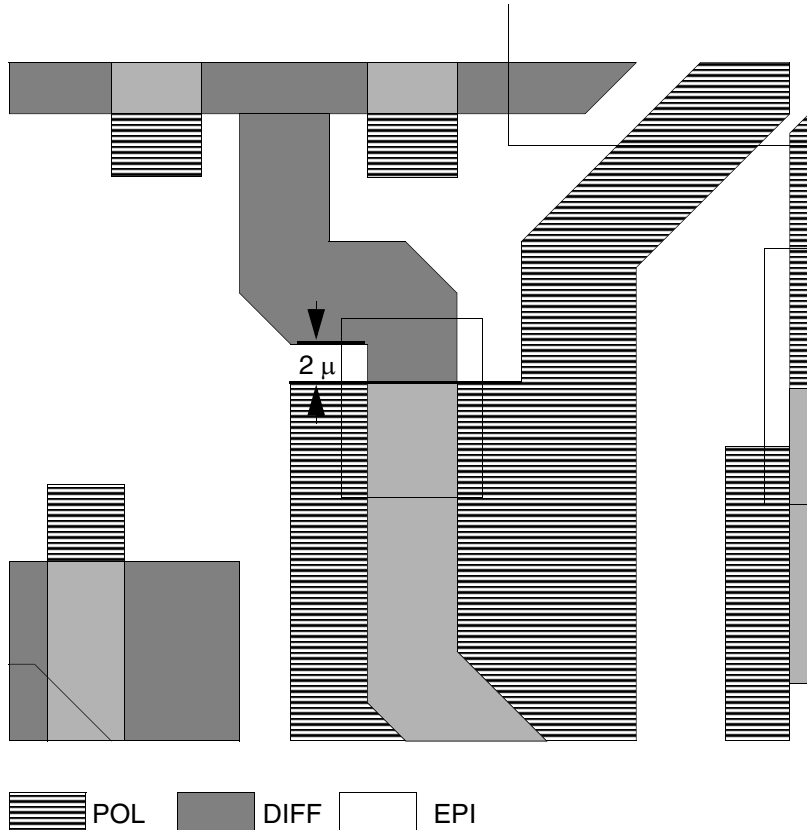


DRC0464 Poly to Diffusion Near EPI

Dracula Reference

Checking Design Rules (DRC)

EXT[N] ovepipe sdovepi LT 3.0 OUTPUT drc04 64

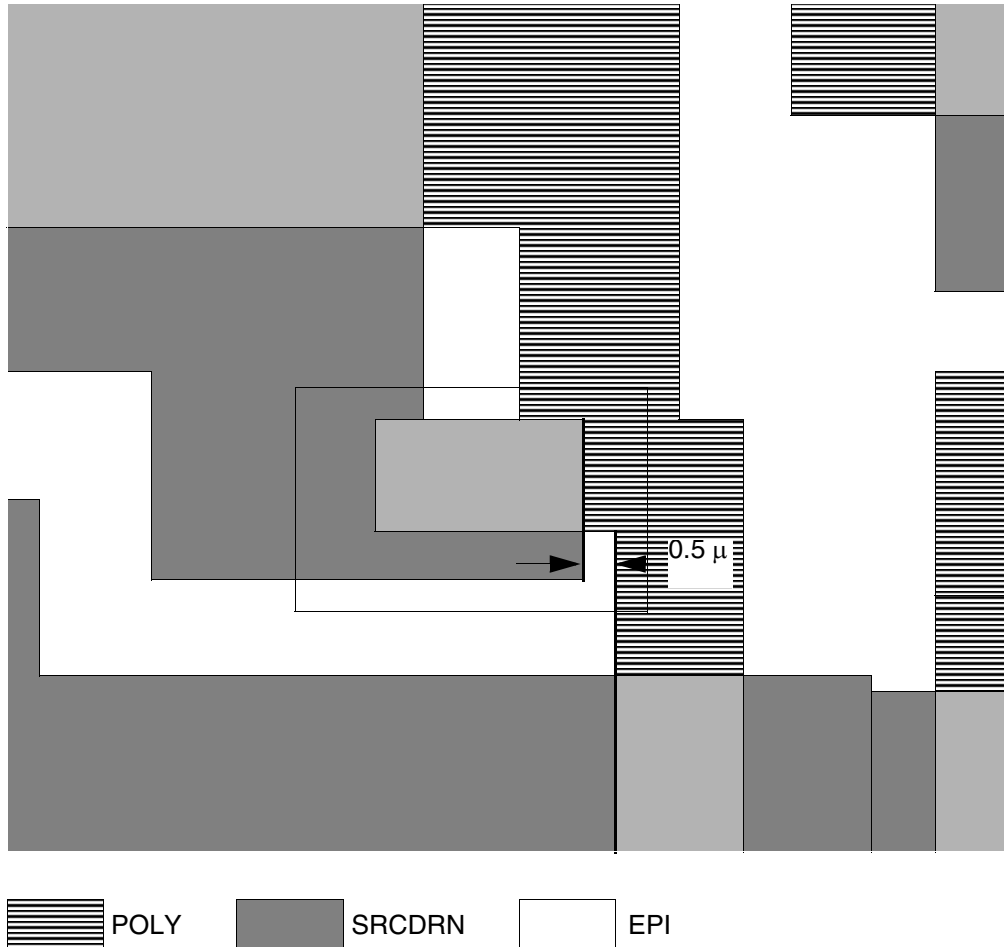


DRC0573 Poly to Diffusion (Node Option)

Dracula Reference

Checking Design Rules (DRC)

EXT[N] poly srcdrn LT 1.0 OUTPUT drc05 73



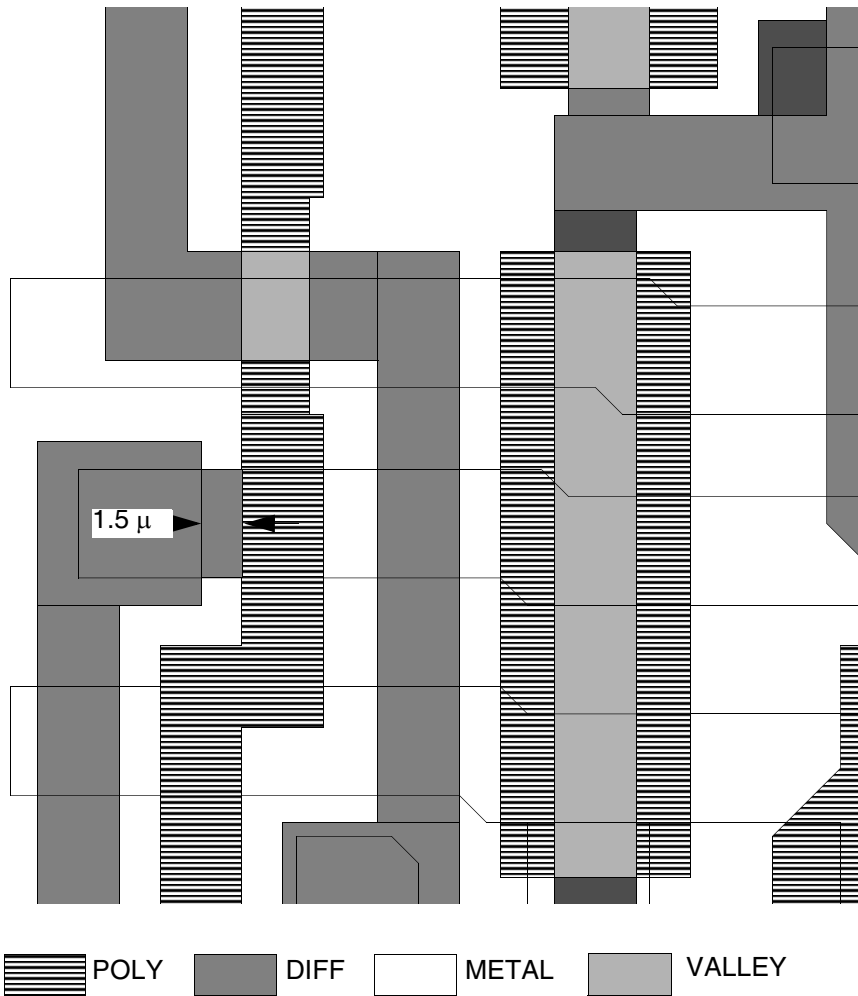
Note: Because of the *N* option, this error violation is not flagged.

Dracula Reference

Checking Design Rules (DRC)

DRC0574 Poly to Diffusion When Crossed by Metal

EXT[R] poly diff LT 2.0 valley
AND metal valley metval OUTPUT drc05 74



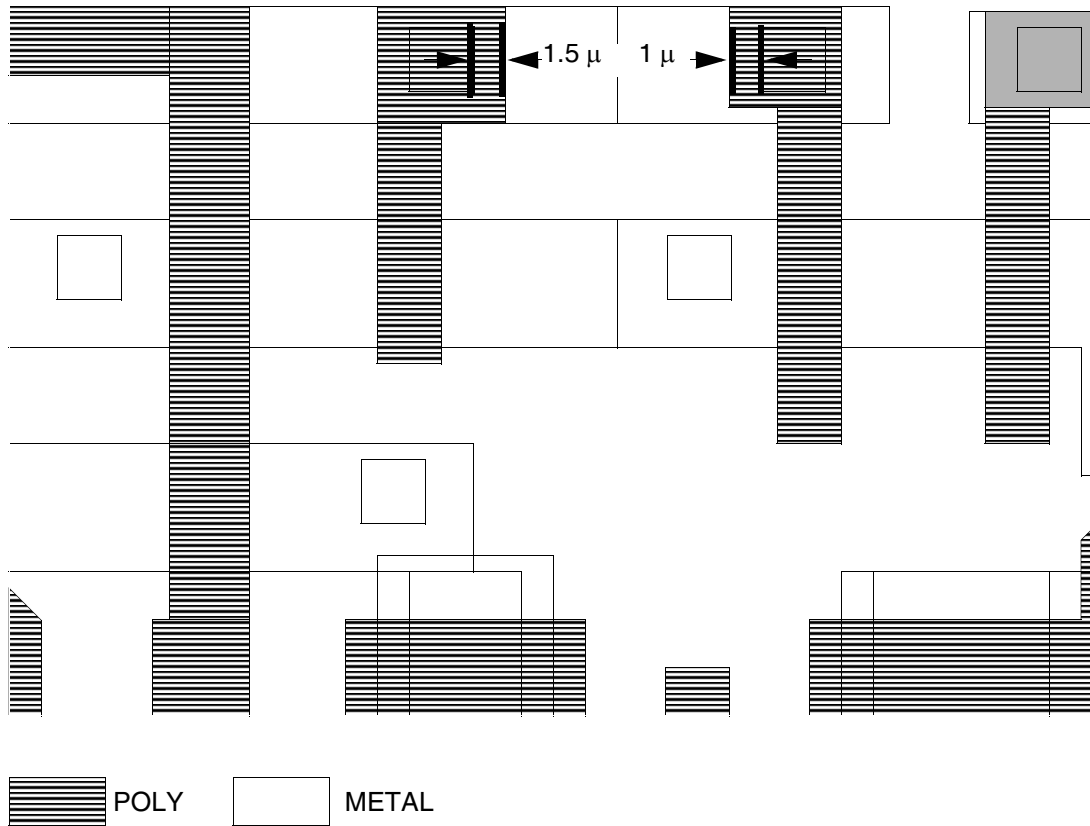
DRC0685 Metal-Contact Between Metal and Poly

SIZE cont by 3.0 ovmc
AND ovmc metal xcmct
ENC[PC] cont xcmct RANGE 2.9 3.1 &

Dracula Reference

Checking Design Rules (DRC)

ENC[PCTO] cont poly LT 2.0 OUTPUT drc06 85

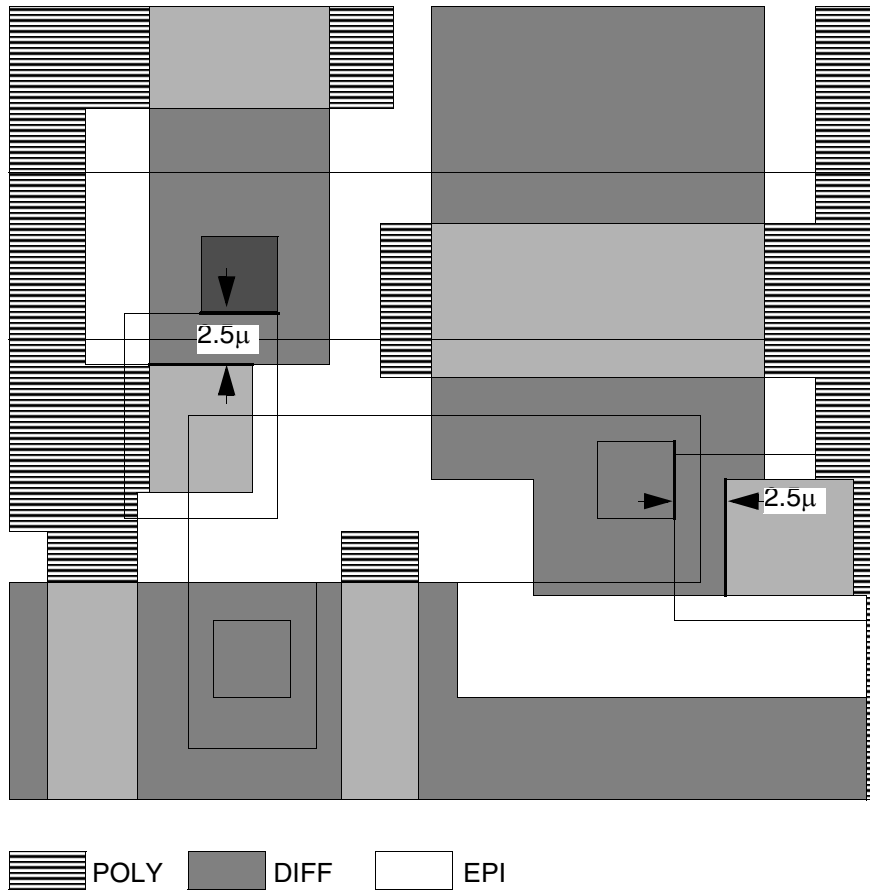


Dracula Reference

Checking Design Rules (DRC)

DRC0687 Metal-Contact Within Diffusion to Poly

EXT[N] mcsdrn poly LT 3.5 OUTPUT drc06 87



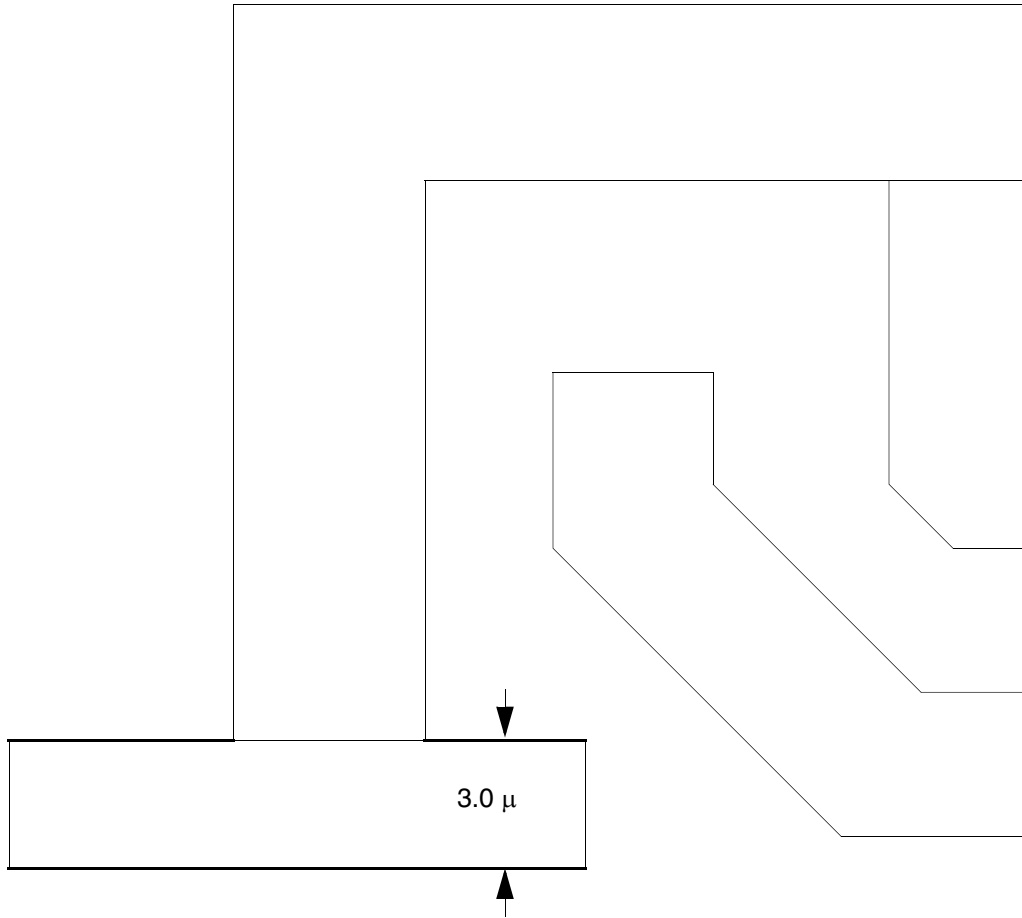
Note: Because of the *N* option, this error violation is not flagged.

Dracula Reference

Checking Design Rules (DRC)

DRC0792 Width of Metal

WIDTH metal LT 5.0 OUTPUT drc07 92



Dracula Reference

Checking Design Rules (DRC)

Database Comparison

The following rules file demonstrates a DRC check that compares digitized layers of one database to layers of another database. The output is in a graphic format.

compare1.com	compare2.com
*DESCRIPTION	*DESCRIPTION
INDISK=lay1.db	INDISK=lay2.db
PRIMARY=top1	PRIMARY=top2
.	PRINTFILE=comprt
.	OUTDISK=comout
.	.
*END	*END
*INPUT-LAYER	*INPUT-LAYER
meta=7	metb=7
EXPORT meta	IMPORT-meta
*END	.
	*END
	*OPERATION
	NOT metb meta err1 OUTPUT err1 50
	NOT meta metb err2 OUTPUT err2 50
	.
	;or
	.
	XOR meta metb err1 OUTPUT err1 50
	*END

To submit the comparison job, type the following:

```
% PDRACULA
  :/GET compare1.com n
  :/NEXT
  :/GET compare2.com n
  :/FIN
```

This database comparison first runs the *compare1.com* job to read in the first database layers. Then the database comparison runs the *compare2.com* job to read in the second database layers and perform the graphic comparison. The summary report file is *comprt.sum*, and the graphic output file is *comout.dat*.

Hierarchical and Multilevel Hierarchical Design Rules Checker

HDRC provides the design rule checking function for hierarchical Dracula. Before reading this section, you should be familiar with the information about hierarchical Dracula found in the [Introducing Dracula](#) section in chapter 1 of this manual.

As chip complexity and device count increase, the need for CPU time and disk space also increases. Dracula supports two-level hierarchical design rule checking (HDRC) and multilevel (MDRC) checking. MDRC supports multilevel design hierarchy so you can reduce the verification time and disk storage. Dracula has the following check modes: flat, cell, composite, hierarchical, and multilevel.

The following information is described in this section:

- Hcell selection
- Hcell creation, Hcell environments, and the composite plane
- HDRC modes
- HDRC error reports
- Sample HDRC rules file

For more information about running HDRC, see “[Running a Hierarchical Design Rule Check](#)” in the “Setting Up Hierarchical Dracula” chapter of the *Dracula User Guide*.

Prerequisites

You need the following data to run HDRC:

- An HDRC rules file
- A database file in one of the database formats that Dracula supports

For a list of the supported database formats, refer to the [SYSTEM or SYSIN](#) section in Chapter 11 of this manual.

Selecting HDRC Hcells

In both hierarchical and cell modes, HDRC automatically selects Hcells according to internal criteria. This automatic selection of Hcells applies only to HDRC. The selection process initially assumes that all cells in the layout are Hcell candidates, except for those you declare in NOT-HCELL commands. Dracula applies each of the criteria in the following table, “Criteria for HDRC Hcell Selection,” to the Hcell candidates, successively eliminating candidates from the group. Dracula uses only cells that meet all five criteria.

In addition, you can select Hcells by using the HCELL and HCELL-FILE commands. Or, in hierarchical mode only, you can exclude Hcells with the NOT-HCELL command. These commands override the default selection. However, these commands can affect the selection of other Hcells.

In hierarchical mode only, you can modify the automatic Hcell selection process by changing the parameters with the HCELL-RULE command. If you do not know the layout hierarchy, run the default Hcell selection first. Then use the HCELL-RULE command to set your own parameters. You can also use the RAM-CELL and ROM-CELL commands to improve the run performance when RAM and ROM cells are present.

To turn off automatic cell selection, specify an HCELL-RULE command with parameters that prevent the automatic selection of Hcells. For example, $N = 9999$ and $M = 9999$. In this case, the HCELL command determines which Hcells to use. The following table shows the N and M parameters.

Criteria for HDRC Hcell Selection

Selection criteria	Description
Number of placements	A cell must be located in the layout database at least N times (default $N = 2$).
Size of cells	Cells must have a height or length more than twice the ENVIRONMENT-MAX value. HDRC allows variable cell heights as an option to HDRC.
Number of edges	A cell must consist of at least M geometrical edges inside the cell (default $M = 4 * i$, where i is the total number of input layers).
Ascending/descending relationship	An Hcell cannot be located within another Hcell.
Number of placements	A cell must be located in the layout database at least four times.

Dracula Reference

Checking Design Rules (DRC)

Criteria for HDRC Hcell Selection, *continued*

Selection criteria	Description
Size of cells	Cells must have a total number of layers less than 1024.
Selection of cells	Dracula writes cells that have more than 100,000 line segments to the MULTICEL.TAB file.
Number of edges	A cell must consist of at least M geometrical edges inside the cell (default $M = 4 * i$, where i is the total number of input layers).
Ascending/descending relationship	An Hcell can be located within another Hcell. The total number of line segments of the child after expansion cannot exceed 90% of the line segments of the parent cell.
Overlapping area between Hcells	Dracula does not select a cell if more than half its area overlaps other cells.

Creating Hcells, Hcell Environment, and Composite Plane

To perform HDRC, the Dracula program first builds the Hcells, Hcell environments, and composite plane data structures.

After Dracula defines an Hcell, the program generates a horizontally aligned rectangular box large enough to encompass all the flattened geometries in that Hcell. Dracula then outputs the Hcell box created to the CELLBOXX layer for later reference. You can access this data through the CELLBOXX.DAT file.

Hierarchical mode HDRC uses the Hcell environment. This environment can be thought of as a box that encompasses the Hcell and extends beyond it. Any geometries (from the composite level or from another Hcell) within an Hcell environment are considered part of that instance of the Hcell (refer to the following figure, “Building the Hcell Box and Hcell Environment”). By default, the Dracula program sets the environment equal to the largest spacing value specified in any spacing check. However, if you use the ENVIRONMENT-MAX command and the preprocessor detects a spacing value greater than the ENVIRONMENT-MAX value, it warns you of the error.

Dracula Reference

Checking Design Rules (DRC)

If any DRC rule specifies the S option, the value affects the environment calculation. By default, the environment value is

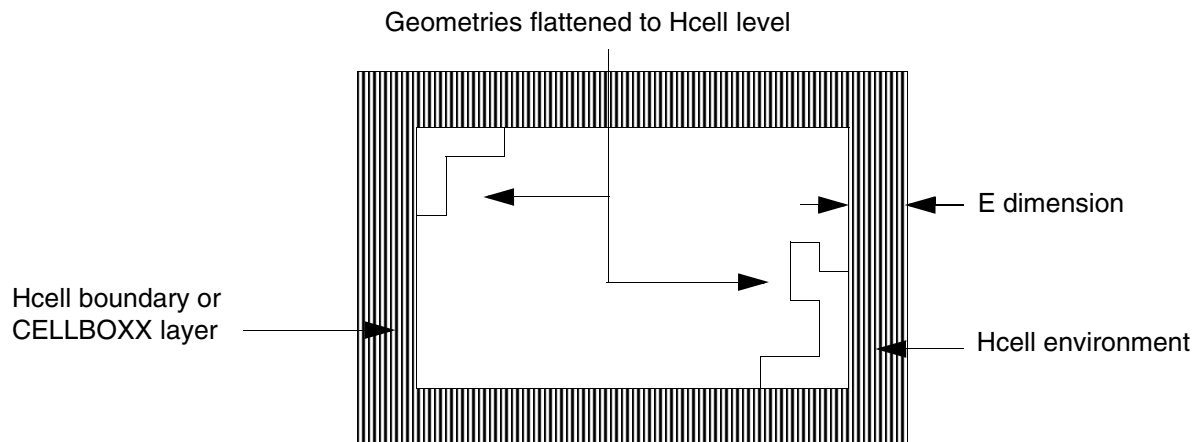
```
max(drc rules)
```

If any DRC rule specifies the S option, the environment value is

```
max(drc rules, sqrt(2)*(drc rule with S option))
```

If you have already set the ENVIRONMENT-MAX value, the value specified with the S option overrides the final environment value.

Building the Hcell Box and Hcell Environment



The Hcell plane is the collection of Hcells and their environment information. You can refer to individual layers or all layers in the Hcell plane. The composite plane contains all geometries not included in any Hcell boxes.

Dracula uses composite plane data and Hcell environment data in hierarchical mode only.

HDRC Modes

HDRC runs in five modes: flat, cell, composite, hierarchical, and multilevel.

Flat Mode

Flat mode HDRC processes input data and output error cells in the same manner as flat Dracula. Dracula expands layers and does not retain hierarchical information for error reporting. However, HDRC attempts to group like commands together to reduce overhead time in loading layers, and as a consequence can have faster run times.

Flat mode is the default mode.

Cell Mode

Cell mode HDRC verifies only cells that qualify as Hcells, not nonHcells or the interconnects between Hcells. This mode generates Hcell-based output, with a single output error cell for each Hcell. These error cells contain all the errors associated with that Hcell and any pseudo-layers created and output by a LOGICAL, SIZE, or SELECT operation. You need to separate different error flags with unique output layer numbers, so Dracula can isolate the error flags in the Hcell output.

To run a job in cell mode, use the `CHECK-MODE=cell` command in the Description block.

Composite Mode

Composite mode HDRC checks the interconnections between Hcells, and between Hcells and composite-level cells. Dracula treats Hcells as “black boxes” and does not check inside Hcells.

To run a job in composite mode, you use the `CHECK-MODE=composite` command in the Description block.

Hierarchical Mode

Hierarchical mode HDRC completely verifies the circuit and outputs the results in hierarchical form. All cells and interconnects are checked. This mode is best for circuits containing a high degree of hierarchy. This mode can also generate Hcell-based output, with a single output error cell for each Hcell.

To run a job in hierarchical mode, you must use the `CHECK-MODE=hier` command.

The following limitations apply to hierarchical mode HDRC checks:

Nodal-related type commands

```
CONNECT
STAMP
LINK
SELECT with LABEL
SELECT[N]
EXT[N]
ENC[N]
INT[N]
```

These commands check hierarchical layers when you use the EXCEPTION-ON=AUTO-FLATTEN command in the Description block of your rule file or if you first flatten the layers with the FLATTEN command. The AUTO-FLATTEN option automatically flattens all layers you use in a CONNECT command.

Dimensional-Spacing Checks

Do not use values larger than 10 microns in dimensional checks, such as ENCLOSURE, EXTERNAL, WIDTH, INTERSECT, and LENGTH commands. Large values slow down the checking. If you want a few large dimensional checks, code these checks differently to avoid performing large dimensional checks on hierarchical layers. For example, consider the following checks:

```
EXT pad LT 50 OUTPUT err1 1
EXT pad metal LT 40 OUTPUT err2 33
```

You can rewrite the commands as shown in the following examples.

```
FLATTEN pad fpad ; FPAD is a flat layer
EXTERNAL fpad LT 50 OUTPUT err1 1

FLATTEN metal fmet1 ; FMET1 is a flat layer
EXTERNAL fpad fmet1 LT 40 OUTPUT err2 22
```

The FLATTEN *metal fmet1* command can use a great deal of CPU time because the metal layer can be very large or dense. A better approach is to take advantage of hierarchical commands and of the pad layer, which is sparsely populated, as shown in the following example:

```
SIZE pad BY 40 spad ; this is a hierarchical
                    ; SIZE and very little cpu
                    ; time is required because
                    ; PAD is a low density layer.
AND spad metal smet ; SMET is a hierarchical
                    ; layer and only has data
                    ; around the PADS.
FLATTEN smet fmet2 ; FMET2 is a flat layer
```

Dracula Reference

Checking Design Rules (DRC)

```
EXTERNAL fpad fmet2 LT 40 OUTPUT err3 33
```

When doing external spacing checks, the resolution value must be at least one-half your minimum DRC value. For example, if you have a DRC spacing value of 5.25, the resolution must be .125.

Multilevel Mode

Multilevel hierarchical DRC (MDRC) is different from full hierarchical DRC in that not every cell in your design hierarchy is used as a verification Hcell. Dracula chooses only a subset of cells to be Hcells. The major difference between Dracula hierarchical mode and Dracula multilevel mode is that an Hcell in multimode can contain another Hcell.

MDRC consists of two phases: the pre-analysis phase and the analysis phase. The pre-analysis phase selects suitable Hcells by thoroughly analyzing your design data. The output of the pre-analysis phase is the MULTICEL.TAB file, which contains the selected Hcells. Each Hcell in the file is a checking cell. To run a job in multilevel mode, use the CHECK-MODE=*multi* command in the Description block.

For more information on using MDRC, see [“Running a Multilevel Design Rule Check”](#) in the “Setting up Hierarchical Dracula” chapter of the *Dracula User Guide*.

Multilevel DRC Example

1. Create the *jxrun.com* file using the following command:

```
/GET drc.rul MULTI
```

The contents of *drc.rul* appear on the screen and Dracula creates the two-stage *jxrun.com* file used to perform data analysis and multilevel Hcell selection.

2. Execute *jxrun.com*.

Dracula writes the selected multilevel Hcells to the MULTICEL.TAB file.

3. Rerun PDRACULA using the following command:

```
/GET drc.rul N
```

PDRACULA does the following:

- ☐ Creates a subdirectory for each cell name in the MULTICEL.TAB file
- ☐ Creates a *jxmult.com* file for the root directory and a *jxrun.com* file for each subdirectory
- ☐ Symbolically links the MULTICEL.TAB file to each subdirectory

4. After PDRACULA completes, execute *jxmult.com*.

Dracula executes the *jxrun.com* files in each subdirectory consecutively.

Hcell Selection

PDRACULA uses the following criteria to select cells for Hcells:

- Cells must exceed the requirements specified in the HCELL-RULE command.

The default number of placements is four. The default number of segments is 1,024 times the number of layers.
- If both parent and child qualify as Hcells, PDRACULA checks if the total number of line segments (after expansion) of the child exceeds 90 percent of that of the parent. If so, Dracula selects only the child cell as an Hcell.
- Cells must not violate overlap criteria set by the CHECK-MODE command.
- Dracula writes multilevel cells with over 100,000 line segments to the MULTICEL.TAB file.

HDRC Error Reports

HDRC error reports include the following:

- HDRC Output Error Format
- HDRC Error Cell
- HDRC Error Summary File

HDRC commands that have an output specification create an error cell similar to flat Dracula if error segments are generated in the composite plane or in flat mode. When run in hierarchical mode, these commands output Hcell error segments to a new or existing Hcell error cell. Hcell error cells can store error segments hierarchically. Dracula stores Hcell-related error segments in either error cell depending upon the uniqueness of the violation. The “HDRC Output Error Format” section in this chapter has more details on the various types of error cells.

Error cells and Hcell error cells have the same polygon format as the input database (GDS II) unless you explicitly specify a different output format with the SYSOUT command. This means you can take these error cells back to the original graphics system, overlay them on the original layout, and see the location of the design rule violations.

The Error Summary file (PRINTF.SUM) contains a summary of the composite plane and Hcell plane error segments. This file is described in the “HDRC Error Summary File” section in this chapter.

HDRC Output Error Format

Dracula sends output error segments from HDRC/DRC spacing checks to an output error file on disk. The OUTDISK command in the Description block determines the name of the output file. The output error file contains a primary output cell that points to the various error cells created. Dracula stores these error cells in files in your host system.

Dracula can store HDRC error segments in two different types of error cells: composite error cells and Hcell error cells.

The following factors determine what kind of error cells contain the error segments:

- The mode, whether flat, cell, composite, hierarchical, or multilevel
- The selected CELL-ERROR-REP option
- The location of the error

Hcell Errors

You select the CELL-ERROR-REP option when you run an HDRC in hierarchical mode. These options establish what to do with errors associated with all instances of an Hcell or all instances of an Hcell environment. The following options are available:

ONCE

Displays Hcell error segments in the composite plane at the first Hcell placement starting from the bottom left of the layout. This is the default.

ALL

Places the Hcell error segments at all instances in the composite, as would be done in a flat mode DRC.

HIER

Displays Hcell error segments in a hierarchical manner at all placements in the composite. Outputs error segments to one of the following error cells according to the rules described in the following sections.

ORIG-ALL

Reports error cells and output cells generated by the layer operations (for example, SIZE and SELECT operations) in the original design hierarchy.

ORIG-DRC

Reports DRC error flags in the original cell.

cell-name

Reports error flags only from the cell name you specify for the output error cell.

Composite Error Cell

Dracula uses this type of error cell for error segments generated in flat mode, which is the same as in flat Dracula. For hierarchical mode, the cell receives error segments when any of the following conditions occur:

- The error segment is in the composite plane
- CELL-ERROR-REP = ONCE or ALL
- CELL-ERROR-REP = HIER and the error segment is not common for all placements of the Hcell or Hcell environment

Hcell Error Cell

The Hcell error cell receives error segments for cell mode and hierarchical mode when CELL-ERROR-REP=HIER and Dracula finds error segments within an Hcell or its environment for all placements of the Hcell.

Duplicate error segments in error cell types can occur in the following cases:

- An error segment occurring only in the environment of an Hcell for all placements of that Hcell appears both in a composite error cell and an Hcell error cell
- An error segment in an area overlapped by two cells for all placements of the two cells appears in both Hcell error cells

The output error file is organized by an output primary error cell (*OUTname*) containing the two types of error cells. For naming conventions for the output primary cell, see the [OUTLIB](#) section in Chapter 11 of this manual. The following example details the organization of the output primary cell.

Primary Error Cell Names

Error-Cell-1

Composite error cells generated for:

Error-Cell-2 flat mode or hierarchical mode

...

Error-Cell-n1

HCELL-Error-Cell-1

Hcell error cells generated for:

HCELL-Error-Cell-2 cell mode or hierarchical mode

...

HCELL-Error-Cell-n3 with CELL-ERROR-REP=HIER

Each error cell receives its data from the output of a different HDRC spacing check command. For example:

Dracula Reference

Checking Design Rules (DRC)

```
WIDTH METAL LT 5 OUTPUT ERR2 56
```

generates error segments destined to a composite error cell (ERR256). Dracula creates this error cell and stores the segments in layer 56.

Error Cell File Names

The first part of error cell file names is obtained by concatenating the spacing check command error cell name (*c-name*) and the layer number (*l-num*). In the example above, the error cell name is ERR256.

The concatenation of the error cell name and layer number specified in the spacing check command must be unique, the same as in flat Dracula. The layer number, however, can be the same for the various commands.

Dracula puts error segments in error cells relative to the 0,0 coordinates of the primary cell in the original database. Therefore, if you overlay the error cells on your original database, whether you place one cell at a time or all cells together, you must place them at coordinates 0,0.

Hcell error cells can contain error segments from more than one HDRC spacing check, but only from error segments associated with one Hcell. Because HDRC spacing checks are output on different layers, each Hcell error cell can contain many layers. For example:

```
WIDTH METAL LT 5 OUTPUT ERR1 56
WIDTH POLY LT 5 OUTPUT ERR2 57
WIDTH MC LT 5 OUTPUT ERR3 58
```

All three commands generate error segments for an Hcell named *abc* and an Hcell named *bcd*. Dracula generates error segments for an Hcell error cell corresponding to *abc* and *bcd* under the three different layers (56, 57 and 58). Each layer contains the error segments generated by each of the corresponding WIDTH commands. No error cell structure exists for cells. You must separate errors by layer number.

Dracula creates an Hcell error cell file for cell ABC and names it ABC_CEL for GDS II or ABC_CEL;1 for Applicon. Dracula adds _CEL to the GDS II Hcell error cell name to avoid a cell name conflict with *abc* from your database.

Note: Dracula creates a new error cell file for each HDRC spacing check that generates an error, but it only creates an Hcell error cell file if it does not already exist for those Hcell errors generated.

HDRC Error Summary File

The first item in the Error Summary file (PRINTF.SUM) shows the type of error reporting you requested in the CELL-ERROR-REP command.

The “All Error Cells” section lists all output error cells specified in an HDRC or DRC spacing check command, whether or not Dracula creates any error cells.

The “Output Cell Summary” section of the error summary file has entries containing the following items:

- Error cell names and corresponding layer numbers generated by HDRC/DRC spacing checks that generate error segments to be sent to the composite plane
- The minimum and maximum *X* and *Y* composite plane coordinates of the smallest window containing the error segments
- The number of error segments output to the error cell. In flat mode, Dracula sends all error segments to error cells. While in hierarchical mode, Dracula outputs some error segments to error cells and some to Hcell error cells, as explained in the “HDRC Output Error Format” section.

The “Detailed Summary for Cell and Cell-Comp Errors” section (provided for hierarchical mode HDRC runs) summarizes the error segments related to Hcells. (See the [Example of HDRC Error Summary for Hierarchical Mode](#) in this chapter). These error segments are either fully enclosed by an Hcell, or are located in the Hcell environment, while having a complementary error segment within the Hcell. Dracula reports two categories of errors:

- **Hcell Errors**

Errors associated with all placements of an Hcell. Typically, these are errors fully contained in the Hcell.

- **Cell-Comp Errors**

Errors associated with only some placements of an Hcell. The name “Cell-Comp error” reflects that these error segments are always output on the composite plane (see the [Hcell Error Cell](#) section in this chapter). These errors are typically found between cell geometries and their environments.

The report divides the errors along two dimensions. The columns reflect the subset of the errors for the various Hcells. The rows reflect the subset of the errors according to the error cells you specified in the OUTPUT option of the HDRC/DRC spacing check commands.

Dracula labels the columns with the file name of the Hcell. Dracula labels the rows with the file name of the error cell or with the label Cell-Comp under the error cell file name.

Dracula Reference

Checking Design Rules (DRC)

A number at the intersection of an Hcell and an error cell name reflects the number of Hcell error segments that Dracula outputs to this error cell. For this case, Dracula counts each error segment only once, independent of the number of placements of the Hcell.

If the number in a row is labeled "Cell-Comp," Dracula indicates the number of Cell-Comp error segments associated with the Hcell column and the error cell named in the preceding row. Dracula counts this number as one error per placement.

Example of HDRC Error Summary for Hierarchical Mode

The report has two column totals, one for the Hcell errors and another for Cell-Comp errors. The report also has row totals.

The summary report ends with the primary error cell name, as well as the x, y coordinates of the smallest window that includes all the errors found. The summary ends with the time and date the run ended.

```
*****/
*/N* DRACULA3 (REV. 4.3 /SUN-4 /GENDATE: 16-AUG-95/07)
*** (Copyright 1995, Cadence) ***
*/N* EXEC TIME =12:21:30 DATE = 18-AUG-95
*****

----- ALL ERROR CELLS LISTING -----
D529829      D5610B32      PPCPW28      D5314A30      D5314B30
D52729      D531330      BAD1FF28     D524A29      D523A29
D52629      D57133      D52229      D523B29      D524B29
D52529      D52829      D531A30      D531B30      D532A30
D532A230     D53330      D534A30      D535A30      D536A30
D536B30      D536C30      D53930      D55131      D55231
D55331      D55431      D55531      D56132      D561B32
D56232      D56332      D561132     D56732      D56432
D561232     D56832      D569A32     D569B32     D5610A32
D57233      D573A33     D57433      D57533      D576A33
D576833     D58234      D58334      D58434      D58534
D59135      D928D35     D593A35     D593B35     D596A35
D596B35      D599A35     D510135     D510235     D5103A35
D5103B35     D5103C35     D5103D35     D510535     D510735
D510835     D5131A36     D5131B36     D5132A36     D5132B35
D5132C36     D5133A36     D513336     D5133B36     D133C136
D133C236     D5134A36     D5134B36     D5134C36     D513536
D5136A36     D5136B36     D513836     D5139A36     D5139B36
D1310A36     D1310B36     D5131136     D1312A36     D1312B36
D131436      D515137     D515237     D515337     D5133C37
D514437

BEGIN      AT      TIME      = 06:11:51      DATE = 16-AUG-95

----- OUTPUT CELL SUMMARY -----
CELL-NAME  LAYER #      WINDOW -----
            DATATYPE      # OF POLYGONS TEXTS
                        (LINE SEGMENTS)
```

Dracula Reference

Checking Design Rules (DRC)

D5314B30	30	-703.500	-1015.500	865.000	752.00	2
D57133	33	-651.000	-1165.000	830.500	-978.50	34
D523B29	29	-1301.500	-1293.000	1049.500	1143.00	538
D534A30	30	-938.500	-1309.500	1310.000	907.50	238
D535A30	30	-1037.500	-1309.550	1310.050	917.00	280
D536A30	30	-1179.000	-1258.500	1211.000	1286.50	2518
D536C30	30	-651.000	-1161.400	830.500	-981.50	68
D561B32	32	-680.600	-926.050	882.300	719.00	26612
D56732	32	-871.900	-1162.500	886.000	725.00	170416
D57433	33	-653.000	-1166.000	832.500	-977.00	533
D58234	34	-659.500	-1161.500	839.000	-981.50	66
D599A35	35	-445.500	-1141.000	21.000	-1049.004	
D5103A35	35	-652.500	-1168.500	888.500	722.50	304
D510835	35	-333.500	-1181.500	799.000	-1129.50	10
D5131A36	36	-664.800	-923.500	848.800	-922.50	672
D5131B36	36	-869.500	-885.500	-846.000	-884.00	2
D5133A36	36	864.000	-943.500	880.000	-936.50	8
D1310A36	36	-643.000	-1168.500	822.500	-975.00	100
D1310B36	36	-652.500	-1168.500	832.000	-975.00	68
D1312A36	36	-643.000	-1165.000	822.500	-978.50	106

OUTDISK OPTION IS:

EXPLODE CELL GEOMETRIES INTO PRIMARY CELL FOR ALL PLACEMENTS (CELL-ERROR-REP = ALL)

--- DETAILED SUMMARY FOR CELL AND CELL-COMP ERRORS
(NO. OF POLYGONS) ---

NOTE: THE FOLLOWING POLYGON COUNTS DO NOT INCLUDE ERRORS CAUSED ONLY BY COMPOSITE GEOMETRIES

H	T	A	A	A	A	A	A	A	A	A	A	A
C	O	D	D	D	D	D	D	D	D	D	D	D
E	T	0	5	0	0	0	2	6	7	7	7	7
L	A	5	0	1	2	7	0	0	0	0	0	0
L	L	.	1	.	.	.	1	2	2	5	3	4
S		C	.	C	C	C	.	.	.	.	.	.
		E	C	E	E	E	C	C	C	C	C	C
		L	E	L	L	L	E	E	E	E	E	E
		;	L	;	;	;	L	L	L	L	L	L
		0	;	0	0	0	;	0	;	;	;	;
			0				0	0	0	0	0	0

ERRCELL

D523B29	CEL:	36	2	14	0	0	6	2	12	0	0	0	0
CELL-COMP:		4	0	0	0	4	0	0	0	0	0	0	
D535A30	CEL:	3	0	0	0	0	3	0	0	0	0	0	0
D536A30	CEL:	68	0	26	6	6	4	0	26	0	0	0	0
D536C30	CEL:	4	0	0	0	0	0	0	0	4	0	0	0
D561B32	CEL:	108	0	0	0	0	0	0	0	48	12	24	12
D56732	CEL:	406	0	0	0	0	0	0	119	88	39	80	
CELL-COMP:		6	0	0	0	0	6	0	0	0	0	0	0
D57433	CEL:	32	0	0	0	0	0	0	0	32	0	0	0
D58234	CEL:	4	0	0	0	0	0	0	4	0	0	0	
D5103A35	CEL:	18	0	0	0	0	0	0	0	12	6	0	0
D5131A36	CEL:	14	0	0	0	0	0	0	0	14	0	0	0
CELL-COMP:		448	0	0	0	0	0	0	0	0	0	0	
D5131B36	CEL:	0											

Dracula Reference

Checking Design Rules (DRC)

```
CELL-COMP:      1    0    0    0    0    1    0    0    0    0    0    0
D1310A36  CEL:   6    0    0    0    0    0    0    0    6    0    0    0
D1310B36  CEL:   4    0    0    0    0    0    0    0    4    0    0    0
D1312A36  CEL:   6    0    0    0    0    0    0    0    6    0    0    0
-----
```

TOTAL

```
CELL:          709  2  40  6  6  13  2  38  249  106  63  92
CELL-COMP:     459  0  0  0  0  11  0  0  0  0  0  0
```

```
PRIMARY CELL:   OUTAD000
WINDOW:        -1301.500    -1309.550    1310.050    1286.500
ENDED  AT  TIME  = 06:12:25    DATE  = 16-AUG-95
```

HDRC Error Summary for Cell Mode

In cell mode, an HDRC run produces a different summary report (PRINTF.SUM) than in hierarchical and flat modes. The summary is divided into two sections and summarizes the number of error segments found by the HDRC checking commands for each Hcell.

The “All Error Cells Listing” section contains the name of all the Hcells used and the time and date the run started.

The “Individual Cell Summary” section contains information about the number of error segments generated for every error cell output specification under each Hcell. Each entry has a line with an Hcell name, followed by one or more lines with an error cell name, a corresponding layer number, data type, and total number of error segments under this Hcell and this error cell. A final line shows the total number of error segments for error cells under this Hcell and the X, Y coordinates of the smallest window in the Hcell that contains all the errors. These window coordinates are relative to the coordinates of the Hcell in the original database.

Example of HDRC Error Summary for Cell Mode

```
-----ALL ERROR CELLS LISTING-----
DD01      DD02
          BEGIN  AT  TIME  = 12:20:23          DATE  = 16-AUG-95
----- INDIVIDUAL CELL SUMMARY-----
CELL NAME:  SDFF4_CEL
ERROR-CELL:DD02LAYER/DATATYPE:2/      0  # OF POLYGONS:22
WINDOW:    -7.0009.000220.000121.000TOTAL POLYGONS:22
CELL NAME:  SB1_CEL
ERROR-CELL: DD02LAYER/DATATYPE:2/      0# OF POLYGONS:1
```

Dracula Reference

Checking Design Rules (DRC)

WINDOW: -6.000-11.00066.50011.000 TOTAL POLYGONS:1

CELL NAME: ROM_CEL

ERROR-CELL: DD02LAYER/DATATYPE: 2/ 0# OF POLYGONS:6

WINDOW: -7.000-7.000105.00051.500TOTAL POLYGONS:6

CELL NAME: SB2_CEL

ERROR-CELL: DD02LAYER/DATATYPE:2/ 0# OF POLYGONS:1

WINDOW: -6.000-11.00066.50011.000TOTAL POLYGONS:1

CELL NAME: SDFF1_CEL

ERROR-CELL: DD02LAYER/DATATYPE: 2/ 0# OF POLYGONS:57

WINDOW: -9.000-17.000250.000206.500TOTAL POLYGONS:57

PRIMARY CELL: OUTDEMO

WINDOW: -9.000-17.000250.000206.500

Dracula Reference

Checking Design Rules (DRC)

Sample HDRC Rules File

*DESCRIPTION

```
CHECK-MODE = hier
ENVIRONMENT-MAX = 7 mic
;
MODE      = exec no
SCALE     = .001 mic
RESOLUTION= .25 mic
PRINTFILE = hdrcl
PROGRAM   = drac3:
SYSTEM    = gds2
PRIMARY   = demo
INDISK    = demo.db
OUTDISK   = hdrclout
*END
*INPUT-LAYER
diff      = 1
epi       = 7
poly      = 4
metal     = 6
pad       = 17
PIN-TEXT-LAYER = pinlay
CONNECT-LAYER = fpol fsd ; PREPARE FOR NODAL CHECK IN HDRC
*END
*
```

```
AND      poly      diff      gate
NOT      diff      gate      srcdrn
WIDTH    diff      LT 5      OUT      w5001  30  3
EXT[H]    diff      LT 5.5    OUT      x5002  31  4
WIDTH    poly      LT 4.5    OUT      w5003  32
EXT      poly      LT 4.0    OUT      x5004  33
EXT      poly      srcdrn    LT 2.0    OUT      x5013  41
EXT[T]    poly      diff      LT .001    OUT      x5013  42
WIDTH    srcdrn    LT 5.5    OUT      w5013  42
WIDTH    gate      LT 5.0    OUT      w5016  45
WIDTH    epi       LT 7.0    OUT      w5020  52  4
ENC[T]    diff      poly      LT 4.0    OUT      e5301  57  7
;**** NEW ***** ; PREPARE FOR LARGE DIMENSION CHECK
SIZE      pad      BY 30      spad
AND      spad      metal     smet
FLATTEN   pad      fpad
FLATTEN   smet      fmet
EXT      fmet      fpad      LT 30.0  OUT sp1  63  1
;**** NEW ***** ; PREPARE FOR NODAL SPACING CHECK
FLATTEN   poly      fpol      ; FLATTEN POLY LAYER FOR CONNECTION
FLATTEN   srcdrn    fsd       ; FLATTEN SRCDRN LAYER ALSO
FLATTEN   epi       fepi      ; FLATTEN SRCDRN LAYER ALSO
;
CONNECT   fpol      fsd  BY    fepi
EXT[TN]   fpol      fsd      LT 2.0  OUTPUT sp2  58
```

*END

Dracula Reference

Checking Design Rules (DRC)

Checking Electrical Rules (ERC)

This chapter discusses Electrical Rules Checking in Dracula, focusing on the following topics:

- [“Electrical Rules Checker”](#) on page 129
- [“ERC Examples”](#) on page 135
- [“Hierarchical ERC”](#) on page 143

Electrical Rules Checker

The Dracula Electrical Rules Checker (ERC) checks for electrical rule violations. ERC isolates the exact locations at which shorts occur, eliminating the time-consuming process of finding shorts between global signals. ERC detects the following kinds of errors:

- Process-independent errors
 - ☐ Short circuits
 - ☐ Open circuits
 - ☐ Floating nodes
- Process-dependent errors
 - ☐ Invalid devices
 - ☐ Improper implant types
 - ☐ Improper substrate bias
 - ☐ Improper power and ground connections
 - ☐ Singular nodes

This chapter describes the ERC commands and shows detailed examples of the commands used in an ERC run. This chapter also describes hierarchical ERC (HERC).

ERC Commands

The ERC Input-Layer block commands CONNECT-LAYER and TEXTSEQUENCE determine the mask order of conductor layers and the order in which Dracula attaches text. For details about how Dracula handles text, see [Chapter 12, “Input-Layer Block Commands.”](#)

The ERC Operation block commands can be grouped into the following three categories:

- Electrical node extraction
- Circuit element extraction
- Electrical rules checking

Note: Dracula allows a maximum of 1,000 ERC commands in the rule file.

For details about each of the Operation block commands, refer to [Chapter 13, “Operation Block Commands.”](#)

Electrical Node Extraction Commands

The electrical node extraction commands define the connectivity between layers where the connection physically derives from the contacts. The CONNECT-LAYER command in the Input-Layer block defines the sequence. You define the contacts that connect the prescribed layers in the Operation block.

Electrical nodes are formed by the layer processing sequence and the interlayer contacts. Dracula forms and uniquely labels each electrical node formed by the CONNECT-LAYER and CONNECT commands. This process is called “stamping.” Dracula can carry the stamping information to newly generated layers in two ways:

- Dracula automatically stamps layers produced by the AND and NOT commands. Dracula stamps the output layer with the node label of the first specified layer.
- The STAMP command stamps the node labeling from one layer to another. Stamping occurs only for stamped layer polygons that overlap nodes in the stamping layer.

Circuit Element Extraction Commands

The circuit element extraction commands extract MOS devices, pads, and electrical parameters such as widths and lengths. When you define electrical nodes with the CONNECT command, the circuit element extraction commands automatically generate a circuit netlist. You can use this extracted circuit for ERC, LPE, and LVS operations.

Dracula Reference

Checking Electrical Rules (ERC)

Dracula checks the devices defined by ELEMENT commands to verify that the devices are correctly formed. If the terminals of a device are missing or not formed properly, Dracula reports the coordinates of the device in the .ERC file.

Electrical Rules Checking Commands

The electrical rules checking commands check for incorrect devices and gross continuity errors. Some of these rules are global and some are local.

After completing your ERC checks, Dracula flags electrical nodes and elements as potential violations and generates graphic output. For electrical nodes, Dracula traces continuity through the contacts to all areas of layers with the same electrical potential. Dracula uses a definition area to identify elements. For example, the channel area can represent the MOS device to help locate specific violations.

Because ERC operates on the internal data structure that represents the layout by trapezoids, and not polygons, the graphic output and the original layout might look different even though the covered area is the same. If multiple layers are involved in a violation, Dracula assigns consecutive output layer numbers with the lowest number being the one you specify in the command. In the following example, the MULTILAB command outputs errors to layer 50.

```
* INPUT-LAYER
;
diff      = 1          ; Diffusion input layer
epi       = 4          ; Epi-contact input layer
poly      = 5          ; Polysilicon input layer
cont      = 6          ; Metal-contact input layer
metal     = 7 TEXT 60   ; Metal input layer
vapox     = 8          ; VAPOX input layer for pads
natural   = 9          ; Natural XTOR implant layer
harddep   = 10         ; Hard depletion XTOR implant
softdep   = 11         ; Soft depletion XTOR implant
TEXT      = 60         ; Text layer
;
CONNECT-LAYER = srcdrn poly metal
*END
*OPERATION
CONNECT metal poly by mc
CONNECT metal srcdrn by mc
MULTILAB OUTPUT mullab 50
*END
```

Because multiple layers are involved, Dracula assigns graphic output layers as follows:

```
metal    to layer 50
poly     to layer 51
srcdrn   to layer 52
mc       to layer 53
```

Dracula Reference

Checking Electrical Rules (ERC)

Note: PDRACULA issues an error message if any layer number exceeds 127. To avoid this error, use a smaller layer number, a different cell name, or a different data type.

Dracula Reference

Checking Electrical Rules (ERC)

ERC Output

ERC summaries are put in a file with a.ERC extension and the same root name as the.SUM file. The.ERC file contains the number of nodes and regions and all successfully attached text information, including the text name, its coordinate, node number, and text type. The file also records text not attached to any layer, text that caused open or short circuits, soft connection violations, and all ERC results.

To do electrical rules checks, ERC uses electrical nodes derived from the layout and text input through your CAD system, EDTEXT, or both. If your ERC run generates error output cells from MULTILAB or SAMELAB commands, you must resolve these errors before inspecting any other output.

Before loading the graphic error output cells onto your CAD system:

- Inspect module IDLAB, which is the entire text/label read in. Check the conversion of label names and their x,y coordinates.
- Inspect module POSATT, which is the text/label processed for attachments. Dracula groups together text that must be attached to particular conductor layers by CONNECT-LAYER and lists these attachments in the far right column. The attachment might indicate whether the x,y coordinates of the text are correct. Node numbers are also listed.

POSATT shows which group of text/labels is shorted to each other (MULTILAB) and which group of text/labels is opened (SAMELAB). You find these groups listed under the warning in the.ERC summary file.

- Find the layer assignment by searching for the error output name, such as SHORTS40 or OPENS40. Because MULTILAB and SAMELAB both output nodes, the error output cell layer assignments are listed under the ERC checks in the.ERC summary file.

The following is an abbreviated example of a .ERC summary file.

```
*****
*/N   DRACULA   (REV. 4.0.14 / SUN-4 4.X /GENDATE: 8-MAY-95/01
      *** (Copyright 1995, Cadence) ***
*/N*   EXEC TIME =16:37:40   DATE = 8-JUNE-95
*****
```

INDISK PRIMARY CELL : CADENCE

```
*****ERC SUMMARY REPORT*****
      DATE:    5-May-95
      TIME:    09:30:19
```

(COMPOSITE PLANE)

```
*/I*   690 NODES ARE IDENTIFIED FROM    6125 REGIONS
```

Dracula Reference

Checking Electrical Rules (ERC)

*****COMP TEXT SUMMARY*****

OEBP	X	728.50	Y	4948.25	NODE	217	ATTACH	MT2
IO4TOP	X	709.25	Y	4967.50	NODE	272	ATTACH	MT2
IO5TOP	X	721.50	Y	5193.50	NODE	275	ATTACH	MT2
IO6TOP	X	720.50	Y	5417.50	NODE	280	ATTACH	MT2
IO7TOP	X	715.75	Y	5643.25	NODE	299	ATTACH	MT2
WE3B	X	673.75	Y	6306.50	NODE	86	ATTACH	MT2

.
.

/W	WARNING	** TEXT : VCC	1	SHORT	DISCARDED
/W	WARNING	** TEXT : VSS	1	SHORT	DISCARDED
/W	WARNING	** TEXT : IO7BOT	106	SHORT	DISCARDED
/W	WARNING	** TEXT : IOT7	106	SHORT	DISCARDED

.
.

.. LIST OF EXTRACTED PAD NAMES AND NODE NUMBERS :

A0	243
A0TOP	356
A1	236
A1TOP	349

.
.

VCC	E	DELETED
VSS	E	DELETED
WE3B	86	
WEBP	15	

***** ERC ERROR SUMMARY FOR CELL : SHORT30 *****

NUMBER OF	NUMBER OF	NUMBER OF
ERRORS FOUND	ERRORS LISTED	OUTPUT TRAPEZOIDS
2	0	42779

***** ERC ERROR SUMMARY FOR CELL : OPEN30 *****

NUMBER OF	NUMBER OF	NUMBER OF
ERRORS FOUND	ERRORS LISTED	OUTPUT TRAPEZOIDS
0	0	0

***** MULTILAB SUMMARY *****

43	TRAPEZOIDS ARE SELECTED FROM 3MT2	TO SHORT30	AT LAYER	30
91	TRAPEZOIDS ARE SELECTED FROM 3MET1	TO SHORT30	AT LAYER	31
0	TRAPEZOIDS ARE SELECTED FROM 3PL2	TO SHORT30	AT LAYER	32
1	TRAPEZOIDS ARE SELECTED FROM 3POLY1	TO SHORT30	AT LAYER	33

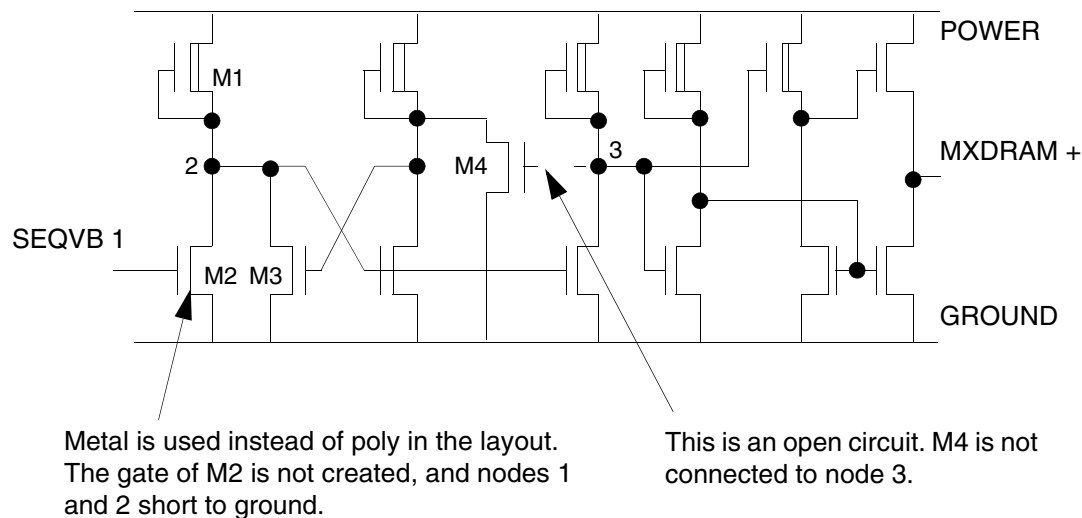
0 TRAPEZOIDS ARE SELECTED FROM 3NSD TO SHORT30 AT LAYER 34

ERC Examples

This section shows an input file that demonstrates the operation and features of ERC. One cell of a 3.5-micron NMOS technology memory chip is used for these examples. The following figure shows the layout and schematic of this cell.

For information about debugging an ERC run, refer to [Chapter 13, “Operation Block Commands.”](#)

Schematic Representation of the Sample Cell Layout



Dracula Reference

Checking Electrical Rules (ERC)

Description Block

In the Description block, you define the input or output devices and file names, CAD layout graphics system, graphics unit scale factor, cell (or portion of cell), and execution mode.

```
*DESCRIPTION
;
PRIMARY      = demo      ; Primary cell name
SYSTEM       = GDS2      ; CAD graphics system
INDISK       = demoerc   ; Input file name
OUTDISK      = demout    ; Output file name
SCALE        = 0.001 mic ; Graphics scale factor
PRINTFILE    = ercprt    ; Output print file
MODE         = exec now  ; Execution mode
RESOLUTION   = 0.125 mic ; Minimum layout increment
;
; *END
```

The statistics of the memory cell in this example are as follows:

```
Technology          = 3.5-micron NMOS
Number of transistors = 14
CAD graphics system  = GDS II
Graphics unit scale  = 0.001 micron
```

Input Layer Block

In the Input-Layer block, you correlate the layout layer number with your own symbolic name and define the mask sequence for the major layers in the technology process.

```
* INPUT-LAYER
;
diff      = 1      ; Diffusion input layer
epi       = 4      ; Epi-contact input layer
poly      = 5      ; Polysilicon input layer
cont      = 6      ; Metal-contact input layer
metal     = 7 TEXT 60 ; Metal input layer
vapox     = 8      ; VAPOX input layer for pads
natural   = 9      ; Natural XTOR implant layer
harddep   = 10     ; Hard depletion XTOR implant
softdep   = 11     ; Soft depletion XTOR implant
TEXT      = 60     ; Text layer
;
CONNECT-LAYER = srcdrn poly metal ;Layer process order
;
*END
```

Note: The *surgery* layer is not yet defined; it is formed by a logical operation in the Operation block.

The reverse order of the CONNECT-LAYER sequence is the default TEXTSEQUENCE that is used to attach text to the layout, that is, to *metal*, then to *poly*, and then to *surgery*.

Dracula Reference

Checking Electrical Rules (ERC)

Operation Block

You specify your electrical rules in the Operation block. The following example describes the set of rules you need to comply with ERC specifications.

```
*OPERATION
;
; ***ELECTRICAL NODE DESCRIPTION***
;
; FORM TRANSISTORS
;
AND    poly      diff      podiff
;
NOT     podiff    epi       xtor
;
OR      harddep   softdep   depl
OR      natural   depl      eblk
;
AND     xtor      depl      chnld
AND     xtor      natural   chnln
NOT     xtor      eblk      chnle
;
; FORM SOURCE-DRAIN LAYER
;
NOT     diff      xtor      srcdrn
;
; ELECTRICAL NODE CONSTRUCTION
;
; FORM PAD WINDOWS
;
AND      metal      vapox    padwin
CONNECT  metal      poly     BY cont
CONNECT  metal      srcdrn   BY cont
CONNECT  poly       srcdrn   BY epi
;
; ***NETWORK CONSTRUCTION***
;;
; FORM NETWORK
;
ELEMENT  MOS[D]     chnld    poly     srcdrn
ELEMENT  MOS[N]     chnln    poly     srcdrn
ELEMENT  MOS[E]     chnle    poly     srcdrn
ELEMENT  PAD        padwin   metal
;
; ***ELECTRICAL RULE CHECKS***
;
PROBE    vcc        OUTPUT   vcc      50
PROBE    vss        OUTPUT   vss      50
;
MULTILAB OUTPUT     shorts   50
;
SAMELAB  OUTPUT     opens    50
;
PATHCHK  LEVEL 1    OUTPUT   path1r   50
PATHCHK  LEVEL 2    OUTPUT   path2r   50
PATHCHK  LEVEL 3    OUTPUT   path3r   50
;
ELCOUNT mos        all      EQ 0     OUTPUT   float    50
;
NDCOUNT  mos        all      EQ 1     OUTPUT   mosnil   40
```

Dracula Reference

Checking Electrical Rules (ERC)

```
NDCOUNT      mos      srcdrn  EQ 1      OUTPUT  moscap  40
;
ECONNECT     mos[D]    srcdrn  CONN vss  OUTPUT  mdconn  40
; *END
```

After you create the Description, Input-Layer, and Operation blocks, you can store the blocks in one or more files, such as the following:

Description block	<i>ercdesc.com</i>
Input-Layer block	<i>ercinpl.com</i>
Operation block	<i>ercoper.com</i>

You can read these files into the preprocessor with the following commands:

```
/GET ercdesc NOLIST
/GET ercinpl NOLIST
/GET ercoper NOLIST
/FINISH
```

NMOS ERC Rules File

The following sample input file is for NMOS circuits. Because ERC checks for standard MOS digital circuit designs, some of the ERC checks are not useful for memory or linear circuits.

```
;*****
;               SILICON GATE NMOS ERC CHECKS
;               FOR STANDARD DIGITAL DESIGN TECHNIQUES
;***** ;
;
*DESCRIPTION
;
INDISK = inchip ; Graphic database file
OUTDISK = ercout ; Graphic error database file
PRIMARY = top ; Top cell name
MODE = exec now ; Mode execution
SYSTEM = gds2 ; Graphic CAD database format
PROGRAM-DIR = drac: ; Program directory of Dracula modules
SCALE = .001 micron ; Graphic database scale factor
RESOLUTION = .5 micron ; Graphic digitized grid increments
PRINTFILE = ercprt ; Print-file name ERC run
;
*END
;
;*****
; INPUT-LAYER BLOCK
;*****
*INPUT-LAYERS
;
diff =1 ; Thin oxide region
depl = 2 ; Depelction VT adjust implant
bcon = 3 ; Buried contact for poly to diffusion
poly =4 ; Polysilicon
mcon = 5 ; Metal contacts
metal =6 TEXT 30 ; Metal & Text layer 30
pad = 7 ; Pad opening
;
```

Dracula Reference

Checking Electrical Rules (ERC)

```
CONNECT-LAYER= srcdrn poly metal
;*** The conductor layers of the NMOS process ***
;
*END

;*****
;OPERATION BLOCK
;
;    CREATE THE DEVICE LAYERS AND CONDUCTOR LAYERS
;*****
*OPERATION
;
AND diff poly gbc
NOT gbc bcon gate ; All transistors
NOT diff gate srcdrn ; N + source/drain layer
AND gate depl dgate ; Depletion devices
NOT gate dgate egate ; Enhancement devices
;
AND pad metal padwin ; Bonding pad
;
; EDTEXT abc.txt ; ASCII texting file ( optional )
;
;*****
;
;    CONNECT OPERATION FOR NMOS PROCESS
;*****
;
CONNECT metal poly BY mcon
CONNECT metal srcdrn BY mcon
CONNECT srcdrn poly BY bcon
;
;*****
;
;    DEFINING ELEMENTS OF NMOS PROCESS
;*****
;
ELEMENT MOS[D] dgate poly srcdrn ; Depletion device
ELEMENT MOS[E] egate poly srcdrn ; Enhancement device
ELEMENT PAD padwin metal ; Pad element
;
;*****
;
;    ERC CHECKS
;*****
;
MULTILAB OUTPUT shorts 30 ; Short test on all labelled nodes.
SAMELAB OUTPUT opens 30 ; Opens test on all labelled nodes.
;
PATHCHK LEVEL 1 OUTPUT novdd 30 ; Nodes that have no path to
; power.
PATHCHK LEVEL 2 OUTPUT novss 30
```

Dracula Reference

Checking Electrical Rules (ERC)

```
; Nodes that have no path to ground.
PATHCHK LEVEL 3 OUTPUT nopwr 30
; Nodes that have no path to power and ground.
PATHCHK LEVEL 4 OUTPUT noall 30
; Nodes that have no path to power, ground, and other texted
; pads.
;
ECONNECT MOS srcdrn CONN vdd &
ECONNECT MOS srcdrn CONN vss OUTPUT vddvss 31
; Transistors that are connected directly across power supply.
;
NDCOUNT MOS srcdrn EQ 1 OUTPUT allcap 32
; Transistors with source/drain count equal to one( capacitor? )
;
NDCOUNT MOS all EQ 1 OUTPUT mixup 33
; Transistors with their source, drain, and gate connected
; together.
;
NDCOUNT MOS srcdrn GT 2 OUTPUT sdx 35
; Transistor with source/drain count greater than two
; ( illegal MOS device ?MOS ).
;
ELCOUNT MOS all EQ 0 OUTPUT flonod 30
; Node with no MOS device connected.
ELCOUNT MOS all EQ 1 OUTPUT sindev 30
; Node with only one device connected.
;
ECONNECT MOS[D] poly CONN vss OUT dgtvss 36
; Depletion transistors with their gates connected to VSS.
ECONNECT MOS[E] poly CONN vdd OUT egtvdd 37
; Enhancement transistors with their gates connected to VDD.
;
ECONNECT MOS[E] srcdrn CONN vdd &
ECONNECT MOS[E] poly DISC vdd &
NDCOUNT MOS[E] all LT 3 &
NDCOUNT MOS[E] srcdrn EQ 2 OUTPUT noimp 38
; Enhancement transistor connected in a depletion mode
; configuration.
;
ELCOUNT MOS[D] srcdrn GT 1 OUTPUT gtdepl 30
; Node with more than one depletion load device.
;
; *BREAK lvs
; LVSCHK ; Include this if you want to run LVS
*END
```

Dracula Reference

Checking Electrical Rules (ERC)

CMOS ERC Rules File

The following example is an ERC rules file for a silicon-gate CMOS technology.

```
;*****
;
;           SILICON GATE CMOS ERC CHECKS
;           FOR STANDARD DIGITAL CMOS DESIGN TECHNIQUES
;***** *DESCRIPTION
;
INDISK = inchip ; Graphic database file name
OUTDISK = ercout ; Graphic error cell file name
PRIMARY = top ; Top cell name
MODE = exec now ; Mode execution
SYSTEM = gds2 ; Graphic CAD database format
PROGRAM-DIR = drac: ; Program directory of Dracula modules
SCALE = .001 micron ; Graphic database scale
RESOLUTION = .5 micron ; Database digitized grid unit
PRINTFILE = ercprt ; Print file name of DRACULA run
;
*END

;*****
;           INPUT-LAYER BLOCK
;*****
*INPUT-LAYERS
diff = 1 ; Thin oxide regions
pwell = 2 ; P-well regions
poly = 3 ; Polysilicon
nplus = 4 ; N + implant
pplus = 5 ; P + implant
cont = 6 ; Metal contacts
metal = 7 ; Metal
vapox = 8 TEXT=28 ; Pad opening & Text layer 28
SUBSTRATE = bulk 63 ; Layer created for reverse mask
;
CONNECT-LAY = nsub pwell psd nsd poly metal
;*** The conductor layers of the CMOS Process. ***
;
*END

;*****
;           OPERATION BLOCK
;           CREATE THE DEVICE LAYERS AND CONDUCTOR LAYERS
;*****
*OPERATION
AND diff nplus nregion ; All active N + regions
AND poly nregion ngate ; N-Channel devices
NOT nregion ngate nsd ; N + source/drain & taps
;
AND diff pplus pregon ; All active p + regions
AND poly pregon pgate ; P-Channel devices
NOT pregon pgate psd ; P + source/drain & taps
;
AND vapox metal bpad ; Bonding pad device
;
NOT bulk pwell nsub ; Created N-Substrate layer
;
AND psd pwell ptwell ; Pseudo P + tap for P-well
;
```

Dracula Reference

Checking Electrical Rules (ERC)

```
NOT nsd pwell ntsub      ; Pseudo N + tap for N-Substrate
;
;*****
;          CONNECT OPERATION FOR CMOS PROCESS
;*****
;
CONNECT metal poly BY cont
CONNECT metal nsd BY cont
CONNECT metal psd BY cont
CONNECT psd pwell BY ptwell
CONNECT nsd nsub BY ntsub
;
;*****
;          DEFINING ELEMENTS OF CMOS PROCESS
;*****
;
ELEMENT MOS[N] ngate poly nsd pwell      ; N-Channel device
ELEMENT MOS[P] pgate poly psd nsub       ; P-Channel device
ELEMENT PAD bpad metal                   ; Pad element
;
;*****
;          ERC CHECKS
;*****
;
MULTILAB OUTPUT shorts 40
; Shorts test on all labelled nodes.
;
SAMELAB OUTPUT opens 40
; Opens test on all labelled nodes.
;
NDCOUNT MOS[N] nsd GT 2 OUTPUT dverr1 45
NDCOUNT MOS[P] psd GT 2 OUTPUT dverr2 45
; Transistors with source/drain count greater than two
; ( illegal MOS device, ?MOS ).
;
ELCOUNT MOS all EQ 0 OUTPUT float 40
; Nodes with no device connected.
;
ECONNECT MOS[N] nsd CONN vdd &
ECONNECT MOS[N] nsd CONN vss OUTPUT vdvsn 48
ECONNECT MOS[P] psd CONN vdd &
ECONNECT MOS[P] psd CONN vss OUTPUT vdvsp 48
; Transistors that are connected directly across power supply.
;
ECONNECT MOS[P] poly CONN vdd OUTPUT gatvdd 49
; P-Channel transistors with their gates connected to VDD(OFF!).
ECONNECT MOS[N] poly CONN vss OUTPUT gatvss 49
; N-Channel transistors with their gates connected to VSS(OFF!). ;
ECONNECT MOS[P] psd CONN vss OUTPUT psdvss 50
; P-Channel transistors with their source/drain connected to VSS
;( inverted CMOS gate ).
ECONNECT MOS[N] nsd CONN vdd OUTPUT nsdvdd 50
; N-Channel transistors with their source/drain connected to VDD
;( inverted CMOS gate ).
;
PATHCHK LEVEL 1 OUTPUT novdd 40
; Nodes that have no path to ground.
PATHCHK LEVEL 2 OUTPUT novss 40
; Nodes that have no path to power.
```

Dracula Reference

Checking Electrical Rules (ERC)

```
PATHCHK LEVEL 3 OUTPUT nopwgr 40
; Nodes that have no path to power and ground.
PATHCHK LEVEL 4 OUTPUT noall 40
; Nodes that have no path to power, ground and other texted pads.
;
ELCOUNT MOS all EQ 1 OUTPUT onedev 40
; Nodes with only one device connected.
;
NDCOUNT MOS[N] nsd EQ 2 &
NDCOUNT MOS[N] all EQ 2 OUTPUT ndepl 55
; N-Channel Transistors connected in depletion mode
; configuration.
NDCOUNT MOS[P] psd EQ 2 &
NDCOUNT MOS[P] all EQ 2 OUTPUT pdepl 55
; P-Channel Transistors connected in depletion mode
; configuration.
NDCOUNT MOS all EQ 1 OUTPUT mixup 56
; Transistors with their source, drain and gate connected
; together.
;
LCONNECT pwell DISC vss OUTPUT flowel 57
; Any P-well not connected to VSS.
;
ELCOUNT MOS[N] nsd GT 1 &
ELCOUNT MOS[P] psd EQ 0 &
ELCOUNT MOS poly GT 0 OUTPUT badngt 40
; a CMOS NAND gate that is driving another gate through the
; intermediate node of the stacked N-Channel pull-downs.
;
ELCOUNT MOS[P] psd GT 1 &
ELCOUNT MOS[N] nsd EQ 0 &
ELCOUNT MOS poly GT 0 OUTPUT badpgt 40
; a CMOS NOR gate that is driving another gate through the
; intermediate node of the stacked P-Channel pull-ups.
;
; *BREAK LVS
; LVSCCHK ; Include if you want to run LVS
*END
```

Hierarchical ERC

HERC is the electrical rule checking function for hierarchical Dracula. Before reading this section, you might want to be familiar with the information about hierarchical Dracula in Chapter 1, “Overview.”

HERC works like flat Dracula ERC, with the addition of checking in cell and composite modes. Circuits that contain a high level of repetitive cell placements with minimal cell overlap benefit most from HERC.

You can mix HERC commands with HLVS and HLPE commands and use the same mechanism to define Hcell and cell text. Most of the concepts and commands related to HLVS apply also to HERC. Information specific only to HERC is provided in this section. For information about hierarchical operations, refer to [“LVS Overview”](#) on page 275.



The HLVS commands `FRAME BY` and `CELL-CHILD-TEXT` mask HERC errors and should not be used in an HERC run.

For more information about running HERC, see “[Running a Hierarchical Electrical Rules Check](#)” in the “Setting up Hierarchical Dracula” chapter of the *Dracula User Guide*.

HERC Check Modes

You can run HERC in one of three modes: flat, cell, or composite. In flat mode, you can combine HERC with DRC, LVS, or LPE in the same rules file. In cell or composite modes, HERC can run in the same rules file with HLVS and HLPE only.

To specify the mode for a job, you must include the CHECK-MODE command in the Description block of your rules file. The default is flat mode.

Flat Mode

Flat mode HERC checks the electrical rules of the circuit in the same manner as flat mode ERC. To run a job in flat mode, you do not need to edit the ERC rules file.

Cell Mode

Cell mode HERC checks the cells that you define as Hcells and outputs trapezoid files with the extension CET. These files are duplicates of HERC. CEL polygon output files. These files save the trapezoid format for viewing in Dracula Interactive. Dracula Interactive lets you view only cell-mode trapezoid files.

Composite Mode

Composite mode HERC checks the interconnections between Hcells and between Hcells and composite level cells. Dracula treats Hcells as “black boxes” and checks only the cell pins. Composite mode does not check inside Hcells.

HERC Limitations

HERC has the following limitations:

- The maximum number of Hcells is 2,000.

Dracula Reference

Checking Electrical Rules (ERC)

- No hierarchy can exist among the Hcells.
- Neither Hcells nor the composite can have floating contacts. Therefore, at least one of the connect layers for the contact must be present to avoid unrecognizable nodes.
- You cannot form devices in the composite-to-cell overlap area, except with the BASE-LAYER command.
- To complete an HERC job, run cell mode checks before composite mode checks. Also, for composite mode HERC checks, you must repeat the checking commands in the previous cell mode run. Every HERC function in composite mode must be the same as cell mode.

HERC Commands

The HERC commands are MULTILAB, ELCOUNT, PATHCHK, GPATHCHK, and FLATTEN-PWRGND. These commands are described in detail in the following sections.

MULTILAB

To view the composite mode multiple label error output on Dracula Interactive or another graphic editor, you must use both the MULTILAB error file and the SHORTBOX.DAT file. The original MULTILAB error output shows the short polygons on the composite plane, while SHORTBOX.DAT shows all cell boxes through which the shortest path passes. Both the MULTILAB error file and SHORTBOX.DAT are in the OUTDISK database.

For cell mode MULTILAB and all other cell mode ERC commands, Dracula Interactive cannot display the original error output .CEL file because of its polygon format. To display the shortest path, display the corresponding .CET short path file with the same file name.

ELCOUNT

In composite mode HERC, the ELCOUNT command does a complete check, first counting the number of elements on the Hcell boundary, then counting the elements in the composite plane.

If you declare the element pads along the boundary in cell mode HERC, the ELCOUNT command checks the elements inside the Hcell and ignores the Hcell boundary. If you do not declare element pads along the boundary, Dracula generates false errors.

GPATHCHK

The GPATHCHK command checks MOS circuits where a node is supposed to have a path to both the power and ground through one or more source/drains of the MOS device and resistor device. This command assigns an error code to nodes without a proper path to either power or ground.

Before using the GPATHCHK command, you must use the GPATHDEF command to define the connectivity flow for your devices.

PATHCHK

The PATHCHK command in composite mode first traces the path inside each Hcell, then all cell paths on the composite plane to enforce a complete path check.

In cell mode, the PATHCHK command traces only the paths inside Hcell.

FLATTEN-PWRGND

If you digitize text directly down to cells without passing through the composite plane, you can use the FLATTEN-PWRGND=YES command to explode cell power or ground onto the composite plane to detect the shorts formed between cell and composite levels.

Compiling Network Descriptions (LOGLVS)

The main focus of Chapter 5 is on LOGLVS in Dracula. It covers the following:

- [“Overview”](#) on page 147
- [“Preparing Netlists”](#) on page 151
- [“LOGLVS Command Sequence”](#) on page 153
- [“CDL Control Commands”](#) on page 156
- [“SPICE and HSPICE Commands”](#) on page 193
- [“Preparing Data Files”](#) on page 216
- [“Running LOGLVS”](#) on page 221
- [“Using Hierarchical LOGLVS”](#) on page 267
- [“LOGLVS Examples”](#) on page 271

Overview

To run an LVS comparison, you must convert a schematic netlist into a transistor-level network data file. You can use LOGLVS to convert many different database formats, as shown in the LVS flow diagram on the following page.

LOGLVS compiles the logic network data, expands the macros, and integrates the circuit definition with the logic network data.

A logic schematic is concerned only with an element’s logical function—NAND, NOR, AND, OR, and so forth—not with how this function is implemented in transistors. Therefore, the schematic does not provide enough information to perform a consistency check on gates that have the same logic function but different implementations. To solve this problem, Dracula reads a subcircuit description data file you create with the Circuit Description Language

(CDL) described in the [“Preparing Netlists”](#) section in this chapter. Use CDL to modify an existing circuit simulator input data file, such as a SPICE™ file.

When LOGLVS integrates a circuit definition with logic network data, it produces transistor-level network data. Each type of primitive logic element must have a corresponding subcircuit description definition. Logic elements with the same name and the same number of inputs and outputs, but with different implementations, require different CDL definitions. CDL uses the name extension to accomplish this. If a logic element does not have a CDL description, it is treated as a *box* with a logic meaning.

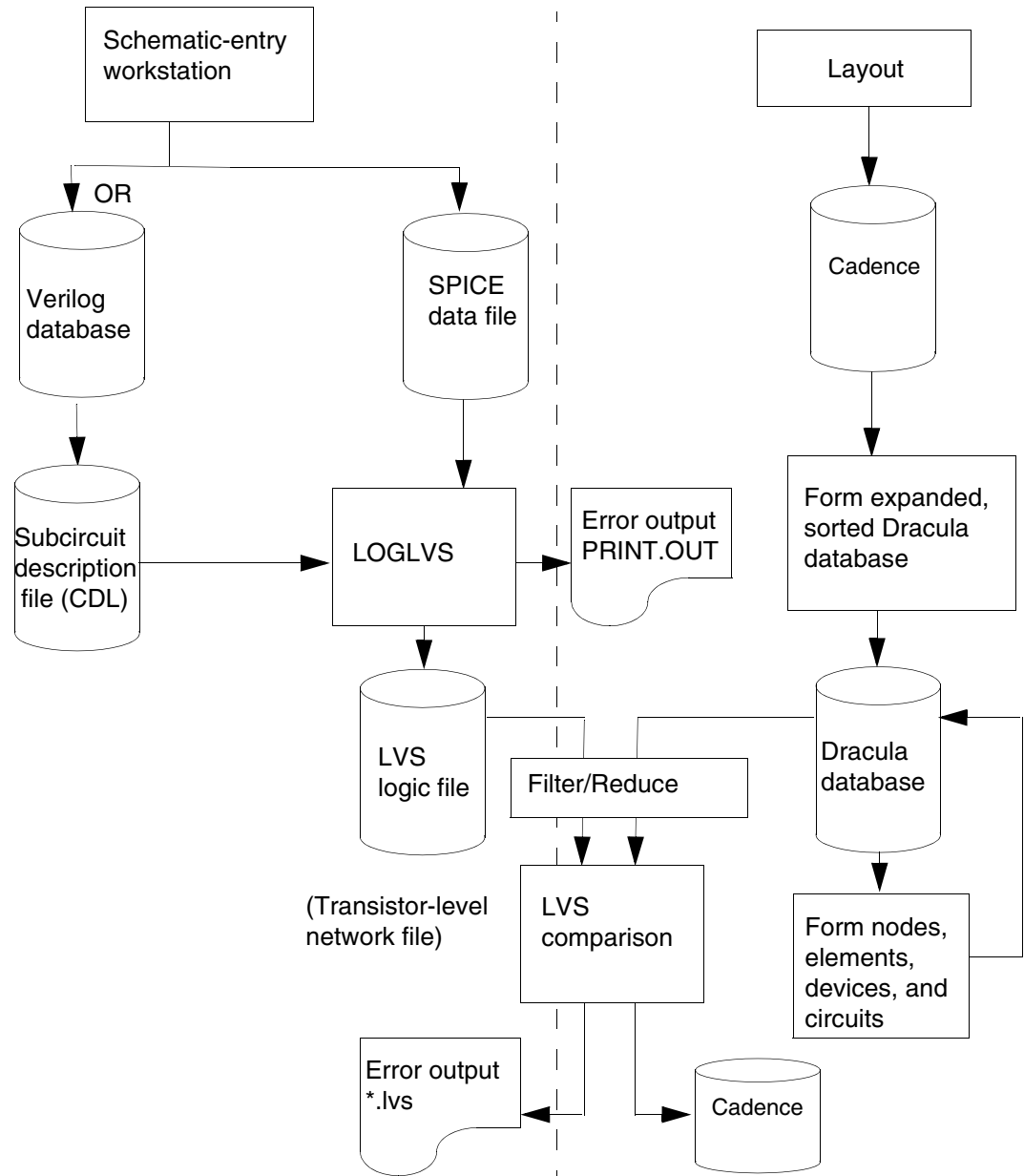
Dracula compares the elements in the logic schematic with the elements in the circuit library. If both name extensions and I/O counts match, Dracula expands the logic element into transistors. After checking and expanding all logic elements in the schematic, Dracula creates an LVS logic file. This file is also known as the transistor-level network file, although it can contain a non-transistor box.

Dracula also expands SPICE data files into transistor-level network files for LVS comparison. The line length is limited to 128 characters, and the module/signal/instance name length is limited to 80 characters.

Note:

- To turn on MULTIPLE-DISK mode, you only need to specify MDISKIO.TAB in the LOGLVS run directory.
- In Dracula 4.81 and subsequent versions there is no maximum limit currently to the number of commands in LOGLVS. LOGLVS writes the commands into JOURNAL.LGS directly. The only limitation is that the size of JOURNAL.LGS cannot be larger than your system's default file size.

LVS Flow Diagram



Initial Correspondence Nodes

LVS requires some initial correspondence node pairs for comparison. Dracula checks the names from the layout and the network to find the initial correspondence pairs. Each network description language has its own way of specifying initial correspondence nodes. The following table summarizes the node names for each data description format.

Dracula Reference

Compiling Network Descriptions (LOGLVS)

Note: Internal correspondence pairs are not required when you use the AUTOMATCH command in the Description block.

	Node names	Definitions
SPICE	GLOBAL I/O list in <i>.SUBCKT</i> or <i>.MACRO</i>	VCC and VDD are keywords for power VSS and GND are keywords for ground Node names defined as the highest level <i>.SUBCKT</i> or <i>.MACRO</i> are I/O for the entire chip
Verilog	supply1 supply0 input output	POWER GROUND INPUT OUTPUT

Node Names

After macro expansion, each node has a new name inside the macro. LOGLVS uses the following format for a node name or cell designator in a logic-level or circuit-level network:

name1-name2-name3...-namen

name1

Highest level module designator

namen

Lowest level node name or designator

LOGLVS uses the following format for the expansion of a SPICE element name:

Mnum1Xnum2...Xnumn

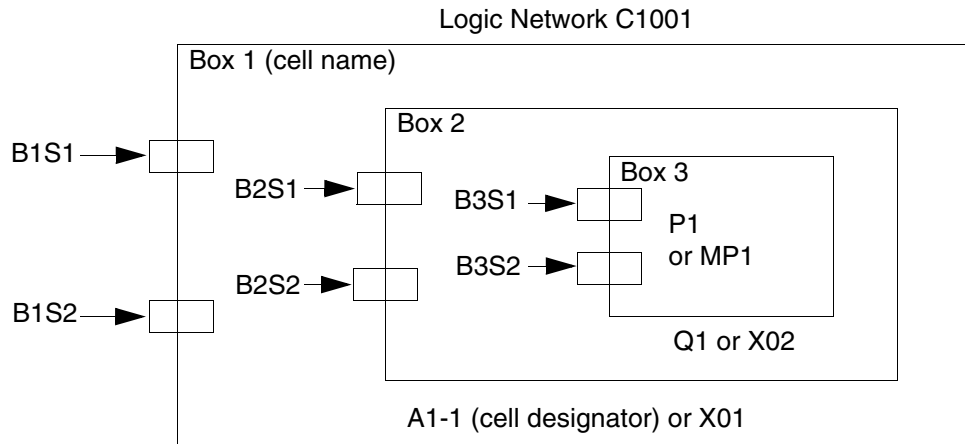
Mnum1

Lowest level element name

Xnum2...Xnumn

Subcircuit element names of the subcircuit call listed in the reverse order of the hierarchy

For example, the logic network has an element *A1-1* as follows:



Dracula expands the node name *P1* in *BOX3* as follows:

`A1-1-Q1-P1`

If the preceding figure is a SPICE subcircuit, Dracula expands the name of the element *MP1* as follows:

`MP1X02X01`

The LVS error report uses the schematic node name. LOGLVS creates the IMAGE.LIS file, which stores the node name directory.

Preparing Netlists

LVS can use a variety of netlists. However, a logic simulator netlist describes only an element's logical function (NAND, NOR, and so forth) and not the transistors that make up the logic symbol. Therefore, you must also provide a SPICE or CDL file that describes, at the transistor level, what comprises the logical elements (primitives). LOGLVS uses both of these files to create a transistor-level netlist for LVS.

Every type of primitive logic element called by the schematic netlist must have a corresponding subcircuit definition in SPICE or CDL format. Logic elements with the same name and the same number of inputs and outputs, but with different implementations, must have different subcircuit definitions. You can either use a different name or use the name extension feature available in CDL.

LOGLVS accepts CDL netlist files in random .SUBCKT order.

The CDL format is a subset of the SPICE format but differs from it in the following ways:

- You can use both node names and node numbers.
- When defining MOS devices, you must give all four terminals (source, gate, drain, and substrate).

To run LVS successfully, you must make minor additions to the basic SPICE or CDL subcircuit netlist.

LOGLVS For Analog or RF Designs

In this release and subsequent releases, LOGLVS can read the netlist with primitive devices defined in instance forms and LOGLVS can also store parameters for box elements. You usually use these two features in Analog or Radio Frequency (RF) designs. You can use box elements to specifically describe the inductor or other design components in the netlist.

Instance Primitive Device Example

Given below is an example netlist with instance primitive devices.

```
.SUBCKT nmos_rf d g s b
.ENDS
.SUBCKT TEST1_N S_B D G
X1 D G S_B S_B nmos_rf wr=2.5e-06 lr=1.8e-07 nr=4
.ENDS
```

The `subckt nmos_rf` represents an NMOS device in the RF design. You use an empty `.SUBCKT` to define the NMOS device. You use a `.SUBCKT` call (`X1`) to instantiate an instance of this device. Refer to the [Running LOGLVS](#) section for the [DEVMAP](#) control command to find out how LOGLVS reads this kind of netlist.

Box Element Example

The following shows an example to define a box element (an inductor element) in the CDL netlist.

```
.SUBCKT spiral_rad T1 T2
.ENDS
.SUBCKT TEST1 P1 P2 P3 P4
X2 P1 P2 spiral_rad nr=4.5 rad=6e-05 w=1.5e-05 s=1.5e-06
.ENDS
```

You use an empty `.SUBCKT` definition to define a box element. You use an empty `.SUBCKT` call (`X2`) to instantiate an instance of the element. You can specify parameters for instances of a box element. In the above example, `nr`, `rad`, `w`, and `s` are parameters.

Note: The box element should be defined with an empty `.SUBCKT` statement before it is instantiated.

LOGLVS can store parameters of a box element instance for the Dracula LVS parameter comparison. The order of the parameters and the number of parameters for instances of the same primitive device types need not be the same. However, the number and order of the parameters for box elements should be the same. The maximum number of parameters is limited to 5.

For the parameter comparison of box elements, refer to the PAR-COMPARE command in the “Description Block Commands” chapter.

LOGLVS Command Sequence

This section describes the LOGLVS commands and shows examples for using them in different types of netlists.

LOGLVS commands can be grouped into the following categories:

- Control commands
- Parse commands
- Expand or flatten commands

For some commands, the order is important. If you specify commands in the wrong order, Dracula issues a warning message.

Control Commands

Control commands are optional and must appear at the beginning of the netlist.

CASE

Turns on case sensitivity. You must specify this command to invoke case sensitivity.

TRANSISTOR

Allocates virtual memory

RESISTOR

Allocates virtual memory for resistor shortages

FPIN

Turns on explicit pin assignment and implicit floating pin handling.

KFPIN

Suppresses the filtering of floating nodes in cell instances when running in composite mode.

DATAFORMAT

Turns on 4.2 data compression

DXF

Generates EXPELE.LIS/EXPELE.CEL

HTV/DRE

Generates Dracula Interactive files

NO_WARNING

Turns off connection checking

NO_WARNING EQUIV_CHK

Suppresses warning messages from LOGLVS on non-global names which are "re-equived", for example *.EQUIV P=pch P=p5v.

PINORDER

<filename>Generates pin order file to be used in QRC. Default filename is PINORDER.TXT but you may change the name by specifying filename. For example,
PINORDER my_pin_ord

Parse Commands

To parse the netlist, you must specify parse commands.

NVER

Compiles library produced by VAN (Verilog Analyzer)

CIRCUIT

Compiles SPICE or CDL netlist

CELL/BBOX

Drops netlist statements from all Hcells. Works only in composite mode for SPICE files.

Expand or Flatten Commands

Expand or flatten commands are required and let you expand or flatten the netlist from the parsed result for the primary cell.

GENPAD

Generates a 6GPADS.DATA file and must precede the CONV command

LINK

Expands the logic network. For VERILOG netlists only.

CONVERT

Converts logic network into a LVSLOGIC.DAT file.

SUMMARY

Prints the ELEMENT summary by types and must follow the CONV command

EXIT

Exits the program

LOGLVS Examples

CDL example:

```
LOGLVS !<<
TRANS 200000
RESI 30000
DXF
CELL cell.tab      ;necessary if running cell mode. Use the
                   ;CELL/bbox command if running composite
                   ;mode.
```

```
CIR cdl                ;parse CDL netlist first
CON top
X
!
```

Verilog example (Assuming that line "define mylib lib_dir" exists in file cds.lib and directory lib_dir exists):

```
van -lib mylib Verilog.net
```

```
LOGLVS !<<
CASE ;VAN is case-sensitive
CELL cell.tab ;necessary if running cell mode. Use the
;CELL/bbox command if running composite
;mode.
CIR cdl ;parse CDL netlist
NVER mylib top_cell ;parse Verilog netlist
CON
SUM
X
!
```

CDL Control Commands

You can add commands to the CDL netlist that can be read by Dracula software but not by the simulator.

- Use an asterisk (*) preceding a command to specify a line to be read by Dracula but not by the circuit simulator.

Note: If the first two characters of the line do not constitute a Dracula control statement, Dracula ignores the line. Spaces are not allowed between these characters.
- In basic element definition use a dollar sign (\$) preceding a field to specify a field to be read by Dracula but not by the circuit simulator.
- In subcircuit headers and in subcircuit calls use a dollar sign after a space character (\$) to mark the beginning of a comment.
- Use a dollar sign (\$) preceding a command to indicate that the line is a comment, and Dracula ignores the line.

***.BIPOLAR**

*.BIPOLAR

Description

Preserves and compares any analog elements coded in the netlist (capacitors, diodes, and resistors). If you do not use the *.BIPOLAR command, LOGLVS ignores all resistors, capacitors, and diodes.

When you use *.BIPOLAR, use the *.CAPA command to tell Dracula to ignore capacitors. Use *.DIODE to tell Dracula to ignore diodes.

To cause Dracula to ignore resistors, use the *.RESI command and specify a threshold resistance below the resistors you want Dracula to ignore. This effectively “shorts” the resistor nodes.

When you use the *.BIPOLAR command, Dracula checks only the connectivity of capacitors, diodes, and resistors. Dracula does not check their size parameters unless you specify in the netlist, respectively,

- *.RESVAL or *.RESSIZE
- *.CAPVAL or *.CAPAREA
- *.DIOAREA
- *.DIOPERI

If you use the RESVAL command in the netlist file, you cannot also use the RESSIZE command in the netlist. Likewise, if you use the CAPVAL command, you cannot also use the CAPAREA command.

Be sure your rules file checks for parameter mismatches for each device. Dracula either checks for all resistance mismatches or for width/length mismatches. However, Dracula cannot check for both of these simultaneously.

Note: Make sure that every element in the CDL file corresponds to an element in the layout.

***.BUSDELIMITER**

**.BUSDELIMITER delimiterChar*

Description

Specifies delimiter characters for pins in a bus in CDL. Pins in a bus must be grouped together in ascending or descending order in CDL. The order of the net connection to its corresponding pin is determined by its position in the .SUBCKT statement. For example, a RAM cell is defined in CDL as follows:

```
.SUBCKT RAM A[0] A[1] B C[8] C[7]
```

The reference for the RAM cell in Verilog is as follows:

```
MODULE TOP (CLK, IN, ...  
  INPUT [1:2] CLK;  
  INPUT [1:0] IN;  
  ...  
  RAM XR1 (.B(net78), .C(CLK), .A(IN));
```

The pin C[8] of instance XR1 is connected to CLK[1].

The pin C[7] of instance XR1 is connected to CLK[2].

The pin A[0] of instance XR1 is connected to IN[1].

The pin A[1] of instance XR1 is connected to IN[0].

The *.BUSDELIMITER command tells LOGLVS what bus delimiter to look for in CDL files so that LOGLVS correctly matches the CDL definition and its corresponding Verilog reference.

Arguments

delimiterChar

The delimiter can be one of the following: a bracket ([]), a less than sign (<), a curly bracket ({}), or an underscore (_). Specify only the leftmost character with the *.BUSDELIMITER command.

Example

The following example specifies a bracket ([]) as the delimiter character.

```
*.BUSDELIMITER [
```

***.CAPA**

`*.CAPA`

or

`*.CAPA model-name1 model-name2 ...`

Description

Use the `*.CAPA` command only when you specify `*.BIPOLAR` but do not want the capacitors from the SPICE file to be included in the LVS check.

If you specify model names, LOGLVS drops capacitors with the specified model names. LOGLVS also drops capacitors that do not specify model names. You can specify wildcards with the following exceptions:

- You must specify “*” with another character. It cannot be specified alone because “*” is treated as a comment in CDL/SPICE.
- You cannot specify wildcards on the cap statement. For example, you cannot specify the following statement:

```
C3 il vss 3.53898E-02 $.MODEL = P*
```

- `*.CAPA` only pay attention for first four characters of model-name.

Arguments

`model-name1 ...`

LOGLVS drops capacitors with the model names you specify.

Example

In the following example, LOGLVS drops C1 and C2.

```
*.global vdd vss
*.BIPOLAR
*.RESI=150 [ry] [rx]
*.CAPA PA
.param
.subckt tlow in out
C1 in out 3 $.MODEL = PA
.ends tlow
.SUBCKT inv Z / A WN=1 LN=1 WP=4 LP=2
* INVERTER
MP1 Z A vdd vdd P WP LP
MN1 vss A Z vss N WN LN
.ends inv
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

```
.subckt top out1 in1 V=1K
x1 o1 i1 /inv WN=13 LN=3
x2 i1 in1 /inv WN=13 LN=3
x3 out1 o1 /inv WN=13 LN=3
x4 o1 vss /tlow
C2 i1 vss 3.53898E-02
C3 i1 vss 3.53898E-02 $.MODEL = PB
.ends
```


***.CAPAREA**

*.CAPAREA

Description

Use this command to specify the area of the capacitor.

Example

```
C0 NET2 GND 4.5 $[MP]
```

indicates that the area of the capacitor body is 4.5 squares.

To have LVS compare the area of the capacitors between the schematic and layout, specify the CAPAREA option in the LVSCCHK command.

***.CAPVAL**

*.CAPVAL

Description

Use this command to specify the capacitance value of the capacitor.

Example

```
C0 NET2 GND 4.5p ${MP}
```

indicates that the capacitance value of the capacitor is 4.5 picofarads.

To have LVS compare the area of the capacitors between the schematic and layout, specify the CAPVAL option in the LVSCHK command.

***.DEFAULT**

```
*.DEFAULT {W=width L=length} {QW=qwidth QL=qlength}  
          {RW=rwidth RL=rlength}
```

Description

Assigns a default width and length to all transistors and resistors in the netlist that have no width and length specification.

Arguments

width

Width of transistors.

length

Length of transistors.

qwidth

Width of BJTs.

qlength

Length of BJTs.

rwidth

Width of resistors.

rlength

Length of resistors.

Specify both width and length. If you specify only one, the other defaults to zero.

***.DIOAREA**

*.DIOAREA

Description

Use this command to specify the area of the diode.

Example

```
D10 NET2 NET7 DP 21 20
```

indicates that the area of the diode body is 21 squares.

To have LVS compare the area of the diodes between the schematic and layout, specify the `DIOAREA` option in the LVSCHK command.

***.DIOPERI**

*.DIOPERI

Use this command to specify the perimeter of the diode.

Example

```
D10 NET2 NET7 DP 21 20
```

indicates that the perimeter of the diode body is 20.

To have LVS compare the perimeter of the diodes between the schematic and layout, specify the `DIOPERI` option in the LVSCHK command.

***.DIODE**

*.DIODE

Description

Use the *.DIODE command only when you specify *.BIPOLAR but do not want the diodes from the SPICE file to be included in the LVS check.

***.EQUATION**

*.EQUATION

Description

Specifies that LOGLVS evaluates and passes parameter expressions down through the netlist hierarchy. You must also specify the .PARAM command.

Note:

- If case-sensitivity is on, all function names, for example, SORT and LOG, must be in uppercase to be recognized by LOGLVS.
- Units are not allowed inside the equation.
- Blank spaces and line breaks are allowed only around equal sign and inside parentheses in equations in element definitions. Blank spaces are not allowed inside an equation that is not in the element definition.
- A subcircuit header cannot contain an equation.

Example

```
*.BIPOLAR
*.GLOBAL GND gnd vdd
*.PIN GND gnd vdd
*.EQUATION
*.RESI
*.RESVAL
.PARAM
.subckt zinv zoutp zinp pl=2.0 pp=5.2
MQ1 zoutp zinp vdd vdd P W=pl*2+(pp-1) L=LOG(pp) ...
R2 zoutp zinp (2000
+ +
+ pp) $W=MAX(pl,pp) $L=MIN(pl,pp)
.ends zinv
.subckt ztest ztout ztin S1=10 S2=90
XI1 ztout net2 zinv pl=4*2 pp=(S1*10)
XI0 net2 ztin zinv pl=EXP(S1-9)/2 pp=(S1+S2)
.ends ztest
```

After expansion, for the devices of instance *XI1*:

```
MQ1 width=115 and length=4.61
MQ0 width=10.49 and length=10.49
R2 resistance value=2100, width=100, and length=8
```

For the devices of instance *XI0*:

```
MQ1 width=101.72 and length=4.61
MQ0 width=10.49 and length=10.49
R2 has resistance value=2100, width=100, and length=1.35
```

***.EQUIV**

`*.EQUIV new-name = old-name`

`*.EQUIV new-model = old-model`

Description

The *.EQUIV command works in one of three ways:

- Equates node names in a layout database to the corresponding node number in the netlist
- Equates element subtypes in a Dracula rules file to model names in a CDL file
- Equates power and ground nodes. For example,

`*. EQUIV VDD=vdd1 VDD=vdd2`

- Equates an element subtype in the Dracula rules file to its corresponding instance primitive .SUBCKT name in the netlist. For example,

```
*.EQUIV N=NMOS_RF
.SUBCKT nmos_rf d g s b
.ENDS
.SUBCKT TEST1_N S_B D G
X0 D G S_B S_B nmos_rf wr=2.5e-06 lr=1.8e-07 nr=4
.ENDS
```

This equates the instance primitive device NMOS_RF in the schematic to the N type of MOS device in the layout (ELEMENT MOS[N]). You use this information for the LVS comparison, that is, you can compare the layout instances of MOS[N] with the schematic instances of NMOS_RF.

- Equates a schematic subcircuit name to its corresponding layout box element name. Just as in the above case, you use this information for the LVS comparison. For example,

```
*.EQUIV IN1=spiral_rad
.SUBCKT spiral_rad T1 T2
.ENDS
.SUBCKT TEST1 P1 P2
X0 P1 P2 spiral_rad nr=4.5 rad=6e-05 w=1.5e-05 s=1.5e-06
.ENDS
```

The corresponding schematic box element for layout box element IN1 is spiral_rad. The above example compares the instances of the layout box element, IN1 with the instances of the schematic box element, spiral_rad.

In an LVS comparison, all the layout text names must start with an alphabetic character and must match the netlist I/O names. The *.EQUIV command can assign a new name to a number in the SPICE netlist. The order in which you enter these parameters is important.

You can also use the *.EQUIV command to equate an element subtype to a model name or `subckt` name of an instance primitive device or a box element. Dracula reads the first two characters of the model name, and if they are not the same as the subtype of the element defined, it looks for an *.EQUIV command.

If your *.EQUIV command extends to more than one line, use an asterisk-plus sign (*+) at the beginning of each additional line to indicate continuation of the previous line.

Note: You cannot use the *.EQUIV command in the following ways:

- To double-translate node names

For example, `ABC = XYZ` and `XYZ = DEF` does not equate to `ABC` and `DEF`.

- To rename a SPICE node number

For example, `CBS=99` where `CBS` is an assigned text in the layout database or in `EDTEXT`.

Arguments

new-name

Layout text.

old-name

Schematic node name or SPICE or CDL node number.

new-model

Element subtype.

old-model

SPICE/CDL model name.

In this version and subsequent versions you can use the EQUIV command (from CDL) also as a LOGLVS command and it works similar to the GLOBAL command. And, you do not have to edit your netlist when you use this command in this way, but you have to limit your command length by 78 characters. When you use EQUIV as a LOGLVS command, it overrides subsequent *.EQUIV statements in your CDL files (without changing them). Refer to the “Running LOGLVS” section for the control command.

Example: Equating Node Names with Node Numbers

The following example equates node numbers in a SPICE file (99, 0, 43) to text names in a layout (VSS, VCC, AIN).

Layout text:

```
VCC VSS AIN
```

SPICE:

```
*.EQUIV VCC=99 VSS=0 AIN=43
M1 99 43 8 0 NC W=20 L=5
```

Example: Equating Subtype Elements with Model Names

In the following example, for each model name *N15L* in the netlist, the layout database must contain a MOS[N] subtype element. Also, for each MOS[N] element in the layout, a corresponding model name *N15L* must be coded in the netlist.

Dracula:

```
ELEMENT MOS[N]
```

SPICE:

```
*.EQUIV N=N15L
M1 D G S B N15L W=10 L=1.5
```

Example: Using Node Numbers in a SPICE File

The following is a correct SPICE file:

```
*.SPICE
*.GLOBAL 99:P 0:G
*.EQUIV VCC=99 VSS=0 A=8 B=7 C=98
.SUBCKT TOP 8 7 98
M1 99 8 7 99 P 20 5
M2 0 8 7 0 N 10 5
M3 7 98 9 99 P 10 5
M4 7 98 9 0 N 5 5
.ENDS
```

Here is an example of an incorrect **SPICE** file:

```
*.SPICE
*.EQUIV A=8 B=7 C=98 VCC=99 VSS=0
.SUBCKT TOP A B C
M1 99 8 7 99 P 20 5
M2 0 8 7 0 N 10 5
M3 7 98 9 99 P 10 5
M4 7 98 9 0 N 5 5
.ENDS
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

The SPICE file is incorrect because the I/O's (*A*, *B*, and *C* in the .SUBCKT command) should be *8*, *7*, and *98*. The *.EQUIV command does not equate names and numbers within the SPICE file.

The incorrect SPICE file can also be changed to a correct CDL file as follows:

```
*.GLOBAL VCC VSS
.SUBCKT TOP A B C
M1 VCC A B VCC P 20 5
M2 VSS A B VSS N 10 5
M3 B C 9 VCC P 10 5
M4 B C 9 VSS N 5 5
.ENDS
```

No *.EQUIV command is needed because this is a CDL file. The .SUBCKT I/O names assign node names to the layout text I/O names for initial correspondence in LVS.

Note: When you use your SPICE or CDL data to run LVS, the *.EQUIV command assigns a name to a node number. It cannot connect two nodes. For example, node 1 and 2 are two nodes in the SPICE data file. If you specify *.EQUIV VCC=1, VCC=2, Dracula assigns the name VCC to both node 1 and node 2. These nodes are still two different nodes. For an example of an exception to this global node handling, see the following example.

Example: Equating Different Power Nodes in a SPICE File

```
*.GLOBAL VDD, VSS, VDD1:P, VDD2:P, VSS1:G, VSS2:G
*.PIN VDD VSS
*.EQUIV VDD=VDD1 VDD=VDD2 VSS=VSS1 VSS=VSS2
.SUBCKT NA2 Z A B
* 2 INPUT NAND
MP1 Z A VDD1 VDD P WP=21.0 LP=21.0
MP2 Z B VDD VDD P W=1.0 L=1.0
MN1 Z A 5 VSS2 N W=1.0 L=1.0
MN2 5 B VSS VSS N W=1.0 L=1.0
.ENDS NA2

.SUBCKT NA3 Z C B A
* 3 INPUT NAND
*.SWAP A B C
MP1 Z A VDD2 VDD P W=21.0 L=21.0
MP2 Z B VDD VDD P W=1.0 L=1.0
MP3 Z C VDD VDD P W=1.0 L=1.0
MN1 Z A 6 VSS1 N W=1.0 L=1.0
MN2 6 B 7 VSS N W=50.62 L=4.53
MN3 7 C VSS VSS N W=1.0 L=1.0
.ENDS NA3
```

***.GNONSWAP**

**.GNONSWAP mname*

Description

Declares all MOS devices with a specific model name as *.NONSWAP devices.

Argument

mname

The model name reference. Must be one or two characters.

Example

```
*.GLOBAL VDD VSS
*
*.GNONSWAP P
*
.SUBCKT NA2 Z / A B
MP1 Z A VDD VDDP WP=12 LP=14
MN1 Z B VDD VDDN WN=12 LN=14
MP2 Z 5 VSS VSS P WP=12 LP=14
MN2 5 B VSS VSS N WN=12 LN=14
.ENDS
```

***.LDD**

*.LDD

Description

Checks LDD devices. You must place this command at the beginning of your CDL file. For more information, refer to the MOSFET description in the “CDL Element Definition” section of this chapter.

Example

```
*.GLOBAL VDD VSS
*.LDD
.
.SUBCKT TOP A B
.
M1 A B VSS NLDD L=2.0 W=10.0 $LDD[N]
.
.ENDS
```

***.MEGA**

*.MEGA

Description

You specify *.MEGA to invoke case sensitivity for *m* (milli) and *M* (mega). If you do not specify *.MEGA, both *m* and *M* mean “milli.” The *.MEGA command works only if you specify the *.SCALE command in the CDL netlist and set the case sensitivity in LOGLVS using the CASE command in LOGLVS.

Note: When you do not use the CASE command, uppercase is the default. Specify the CASE command prior to lines that affect parsing.

If you specify the *.MEGA command, *53m* means 53×10^{-3} , and *53M* means 53×10^6 . If you do not specify *.MEGA, *53m* and *53M* both mean 53×10^{-3} .

Example

```
*.BIPOLAR
  *.RESI = 0
  *.RESVAL
  *.CAPVAL
  *.DIODE
  *.EQUATION
  *.SCALE METER
  *.mega
  .PARAM

*.GLOBAL vdd!

*.PIN vdd

.SUBCKT px A Z
  MM0 Z A vdd! vdd! PM W=Wx L=1.5u M=1
  .ENDS px

.SUBCKT TOP IN OUT VM VP
  RR9 OUT VP 53K $[RP]
  RR11 IN VM 53M $[RP]
  XI0 IN OUT / px Wx=12u
  .ENDS TOP
```

***.NONSWAP**

**.NONSWAP mname*

or

```
Mxxx drain gate source bulk mname {W=width L=length} {m=multiplier} {$LDD[type]}  
$NONSWAP
```

Description

Specifies all logic gate, series, and parallel structures are swappable by default. The *.NONSWAP command lets you control the swappability in different applications.

This command applies to MOS devices only.

Arguments

mname

The model name reference. Must be one or two characters.

Mxxx

The MOSFET element name. The name must begin with *M*.

drain

The drain terminal node number/name.

gate

The gate terminal node number/name.

source

The source terminal node number/name.

bulk

The bulk terminal node number or name.

width

The width. If you specify *W=*, it indicates a width specification. Dracula does not check the width if you do not include this argument.

length

The length. If you specify *L=*, it indicates a length specification.

multiplier

Specifies multiple device.

type

The LDD designator, specified as a comment. Declares that the source and drain terminals have different characteristics and cannot be swapped.

Examples

Specify non-swappable devices in device statements for a SPICE or CDL file as follows:

```
.SUBCKT TOP N1 N2
.....
.....
MD1 out N1 10 VSS MN L=10 W=20 $NONSWAP
MD2 10 N2 20 VSS MN L=10 W=20 $NONSWAP
MD3 20 N3 VSS VSS MP L=10 W=20
.....
.....
.ENDS
```

Specify non-swappable devices by subtype in a SPICE or CDL file as follows:

```
.SUBCKT TOP N1 N2
*.NONSWAP MN
.....
.....
MD1 out N1 10 VSS MN L=10 W=20
MD2 10 N2 20 VSS MN L=10 W=20
MD3 20 N3 VSS VSS MP L=10 W=20
.....
.....
.ENDS
```

You must specify the *.NONSWAP command after the .SUBCKT command.

***.NOPIN**

**.NOPIN signalName ...*

Description

Specifies the exclusion of global pins that were listed in **.PIN* command from HCELL subcircuit. The **.NOPIN* command can only be used inside the *.SUBCKT* statement, immediately following the header of the *.SUBCKT* statement.

Arguments

signalName

The name of the net for the pin.

***.NOSUB_M**

*.NOSUB_M

Description

Ignores substrate terminal for MOS in the LOGLVS schematic side.

***.PIN**

**.PIN signalName ...*

Description

Specifies the insertion of pins that are not in the I/O list to the primary cell or Hcell instances. The *.PIN command specifies global pins if you specify it outside the .SUBCKT statement. If you specify the *.PIN command inside a .SUBCKT statement, *.PIN specifies local pins. See [Using Hierarchical LOGLVS](#) on page 267 for more information.

Arguments

signalName

The name of the net for the pin.

Example

The following example adds a pin A to the subcircuits:

```
*.PIN A
```

When *.PIN A is used outside of the .SUBCKT block, it adds pin A to all subcircuits except those defined with *.NOPIN A. When *.PIN A is used inside the .SUBCKT block, it adds pin A to the current subcircuit only. Node A should be defined as a global node to be used in *.PIN.

***.PININFO**

**.PININFO signalName:pinType ...*

Description

Specifies which pins act as drivers (outputs) and which pins act as loads (inputs). This command adds pin types to pins listed in the .SUBCKT statement only and should be placed after the .SUBCKT statement. Global nodes are ignored if they are listed in the *.PININFO statement.

By default, if a “/” appears in the .SUBCKT header:

- The pins preceding the “/” are considered output nodes.
- The pins following the “/” are input nodes.

If no “/” exists or you specify an unknown pin type, the first pin is an output node and the others are input nodes.

Note: When running LVS, pin types are ignored, except for power or ground pads. Pin type is very important in DSPF (an output of HPRE). Refer to [“Overview of Parasitic Resistance Extraction”](#) on page 471 for more details.

Arguments

signalName

The name of the net for the pin.

pinType

Specifies one of the following pin types:

B	Input/Output (bi-directional)
C	Clock
E	Enable
G	Ground
I	Input
J	Jumper
O	Output

Dracula Reference

Compiling Network Descriptions (LOGLVS)

P	Power
S	Switch
U	Unused

Example 1

In the following example, *Y* is an output signal. *A* is bi-directional. Pins *B* and *C* are input signals.

```
*.PININFO Y:O A:B B:I C:I
```

Example 2

In the following example, if you do not specify power and ground, they are treated as input.

```
.SUBCKT TOP A1 A2 A3 A4 T1 T2 T3 T4 ANA_VDD ANA_VSS DIG_VDD
+ DIG_VSS
*.PININFO ANA_VDD:P ANA_VSS:G DIG_VDD:P DIG_VSS:G
XNA2 A1 A2 A3 ANA_VDD ANA_VSS NA2
XNA4 A1 A2 A4 DIG_VDD DIG_VSS NA2
XNA3 T1 T2 T3 T4 DIG_VDD DIG_VSS NA3
.ENDS
```

Example 3

For a .SUBCKT with long list of signals, the syntax is:

```
*.PININFO O14:O O13:O O12:O O11:O O10:O O9:O O8:O O7:O
*.PININFO O6:O O5:O O4:O O3:O O2:O O1:O B10:B B9:B B8:B B7:B
*.PININFO B6:B B5:B B4:B B3:B B2:B B1:B I10:I I9:I I8:I I7:I
*.PININFO I6:I I5:I I4:I I3:I I2:I I1:I
```

or

```
*.PININFO O14:O O13:O . . . O O7:O
*.PININFO O6:O O5:O . . . B7:B
*.PININFO B6:B B5:B . . . I7:I
*.PININFO I6:I . . . I1:I
```

***.RESI**

```
*.RESI { = value } {[modelName] ... }  
*.RESI { value } {[modelName] ... }  
*.RESI {[modelName] ... }
```

Description



You must also specify the `*.RESVAL` command when using the `*.RESI` command. However, do not specify the `*.RESSIZE` command when using the `*.RESI` command as it may result in unpredictable behavior.

Specifies the threshold value of the shorted resistors. If the resistance between any two nodes in the resistor statement is less than or equal to the threshold value, Dracula connects the two nodes.

You can also short a resistor by specifying its model name. Dracula treats all resistors with the same model name as shorts.

You cannot short two global nodes together.

You can use resistors in SPICE format for simulating both intentional and parasitic resistors. For LVS purposes, you must short the parasitic resistors.

You must include a `*.BIPOLAR` command. Otherwise, Dracula ignores `*.RESI` and all resistors. You must include `*.RESI` before any SPICE element statement.

Note: If multiple `*.RESI` lines are specified, LOGLVS will use the value of the last `*.RESI` line (this includes the default value of 2000).

Arguments

value

The threshold value of the resistors to be shorted. You can indicate the units with a *K* or an *M*, or you can type the full number. You can use the SPICE scale factor for *value*. The default is 2K.

modelName

The model name of the resistors you want to short. The *modelName* can be a maximum of two characters. LOGLVS always truncates modelNames on a `*.RESI` line if they are

longer than four characters. Do not put spaces between the square bracket and the model name.

Example 1

In the following example, Dracula shorts both *R1* and *R2*.

```
*.Global VDD VSS
*.Bipolar
*.RESI=3K [RA]

.SUBCKT TOP A B C
M1 A B VDD VDD P
R1 C A 5K $[RA]
R2 A B 2.5K
.ENDS
```

Example 2

In the following example, the X4 placement of SHI1 shorts the internal net 01 to VSS. The X5 placement of SHI2 shorts its two input terminals, OUT1 and IN1, together. The short effect propagates through the network.

```
*.GLOBAL VDD VSS
*.BIPOLAR
*.RESI
.PARAM

.SUBCKT INV Z / A WN=1 LN=1 WP=4 LP=2

* INVERTER
MP1 Z A VDD VDD P WP LP
MN1 VSS A Z VSS N WN LN
.ENDS INV

* SHORT INPUT TO VSS
.SUBCKT SHI1 PORTA
R1 PORTA VSS 1.5K
.ENDS

* SHORT TWO I/Os
.SUBCKT SHI2 A B
R1 A B 1K
.ENDS

.SUBCKT TEST OUT1 IN1
X1 I1 IN1 /INV WN=13 LN=3
X2 O1 I2 /INV WN=13 LN=3
X3 OUT1 O1 /INV WN=13 LN=3
X4 O1 VSS /SHI1
X5 OUT1 IN1 /SHI2
.ends
```

***.RESSIZE**

*.RESSIZE

Description



Do not use the *.RESVAL and *.RESSIZE commands simultaneously in the netlist. If both commands are specified and *.RESVAL is the last command, it will be used and can result in unpredictable behavior.

Use this command to specify the body size of a resistor as the number of squares in the resistor, or to specify the width and length of the resistor body.

Example 1

```
*.RESSIZE
R3 NET2 OUT 2.4 $[S]
```

indicates the body size is 2.4 squares.

To have LVS compare the resistor's body size between the schematic and layout, specify the RESSIZE option in the LVSCHK command.

Example 2

```
*.RESSIZE
R3 NET2 OUT $[S] $W=5 $L=12
```

indicates the width of the resistor is 5 microns and the length is 12 microns.

To have LVS compare the width and length of a resistor body between the schematic and layout, you need to specify the RESWPERCENT and RESLPERCENT options respectively in the LVSCHK command.

***.RESVAL**

*RESVAL

Description



Do not use the *.RESVAL and *.RESSIZE commands simultaneously in the netlist. If both commands are specified and *.RESSIZE is the last command, it will be used and can result in unpredictable behavior.

Use this command when specifying the resistance value of the resistor.

Example

*.RESVAL

R3 NET2 OUT 2.4 \${S}

indicates the resistance value is 2.4 ohms.

To have LVS compare the resistance value between the schematic and layout, specify the RESVAL option in the LVSCHK command.

***.REVERSE**

*.REVERSE

Description

Swaps MOS width and length in the SPICE netlist. The standard SPICE MOS statement specifies width in the first field and length in the second field. If these two fields are swapped, you must use *.REVERSE to ensure correct device size checking.

Example

In the following example, the width and length are swapped in the MOS statements, and *.REVERSE alerts LOGLVS of this. If you specify *W=* and *L=* in the MOS statements, *.REVERSE has no effect.

```
***TRUE SPICE FILE***
*.SPICE
*.REVERSE
*.GLOBAL      99      0
*.EQUIV      IN=1      OUT=2      VCC=99      VSS=0
.SUBCKT      INV      1      2
M1      99      1      2      99      P      2      20
M2      0      1      2      0      N      2      10
.ENDS
```

***.SCALE**

`*.SCALE [meter]`

Description

Specifies that Dracula read scales in the CDL netlist, including scales for all parameters. If you do not specify this command, scales in the CDL netlist are read only for values of resistor and capacitor instances (note that this is not the case when you use named parameters to set them).

Note: Dracula ignores all units in the subcircuit headers and instance invocations. For the expression $w=2.4\mu m$, u is the scale and m is the unit. When `*.scale` is used without “meter”, $0.25u$ becomes $0.25e-12$ meter, which is correct “by rule”, but too small to be an intended value. Dracula issues a warning and scales by $1.0e6$. The final value is equal to $0.25e-6$ meter.

The following table lists the scaling factors.

Character (case-insensitive)	Name	Multiplier
T	Tera	10^{12}
G	Giga	10^9
MEG	Mega	10^6
K	Kilo	10^3
M	Milli	$1e-3$
U	Micro	$1e-6$
N	Nano	$1e-9$
P	Pico	$1e-12$
F	Femto	$1e-15$

Arguments

meter

Changes the default database unit from microns to meters. Parameters that represent distance, such as those for diode, MOSFET, BJT, and function MOSFET instances, are multiplied by $E+6$ internally to change the unit back to micron. Parameters for resistor and capacitor instances are not adjusted if they do not represent distance.

Example 1

This example represents default behavior.

```
*.BIPOLAR
*.CAPVAL
.PARAM
*.GLOBAL GND:G VDD:P
.SUBCKT INV IN OUT
C1 IN GND 59K * 59000
M1 OUT IN VDD VDD P w=.5u l=.25u * .5 .25
M2 OUT IN GND GND N w=.5u l=.25u * .5 .25
.ENDS
```

Example 2

In the following example, **.SCALE* is specified.

```
*.BIPOLAR
*.CAPVAL
*.SCALE
.PARAM
*.GLOBAL GND:G VDD:P
.SUBCKT INV IN OUT
C1 IN GND 59K * 59000
M1 OUT IN VDD VDD P w=500000u l=.25 * .5 .25
M2 OUT IN GND GND N w=500000u l=.25 * .5 .25
.ENDS
```

Example 3

In the following example, **.SCALE [meter]* is specified.

```
*.BIPOLAR
*.CAPVAL
*.SCALE meter
.PARAM
*.GLOBAL GND:G VDD:P
.SUBCKT INV IN OUT
C1 IN GND 59K * 59000
M1 OUT IN VDD VDD P w=.5e-6 l=.25u * .5 .25
M2 OUT IN GND GND N w=.5e-6 l=.25u * .5 .25
.ENDS
```

***.SPICE**

*.SPICE

Description

Indicates a standard SPICE file. This command lets you use an unmodified SPICE file that contains only node numbers instead of node names. When you use a *.SPICE command, Dracula assumes all node names in the CDL or SPICE commands use numbers instead of names. Dracula issues a warning when it encounters a node name instead of a number.

You must use this command before using a SPICE element statement. If you omit the *.SPICE command, you must place a backslash before the name of the subcircuit being called. Without the backslash, Dracula assumes that the subcircuit is defined in CDL format.

***.UNSPEC**

*.UNSPEC

Description

When you use the *.UNSPEC command in your SPICE or CDL netlist, LOGLVS detects devices that do not have a width or length definition. Dracula issues a warning in the PRINT.OUT file.

In Dracula 4.9 version, when *.UNSPEC and *.BIPOLAR are set, BJTs are checked for the emitter area value only, and not for width of length (as before).

Use this command to check RES, BJT, JFET, and MOS devices.

Examples

Example 1

```
*.GLOBAL VDD, VSS
*.PARAM WP=2 WN=2
*.UNSPEC

.SUBCKT NA2 Z / A B
* 2 INPUT NAND
MP1 Z A VDD VDD P
MP2 Z B VDD VDD P WP=12 LP=14
MN1 Z A 5 VSS N
MN2 5 B VSS VSS N WN=12 LN=14
.ENDS NA2
```

When you place the *.UNSPEC command in the SPICE or CDL file before the .SUBCKT command, you get warning messages for the devices *MP1* and *MN1* as follows:

```
***WARNING: MOS HAVE WIDTH OR LENGTH WITH 0
LINE 7 : MP1 Z A VDD VDD P

***WARNING: MOS HAVE WIDTH OR LENGTH WITH 0
LINE 9 : MN1 Z A 5 VSS N
```

Example 2

```
*.GLOBAL VDD, VSS
*.BIPOLAR
*.UNSPEC

.SUBCKT ENET D G S SUB C B E
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

```
***  
M1 D G S SUB MP  
Q1 C B E PNP  
.END
```

When you place the *.UNSPEC and *.BIPOLAR commands in the SPICE or CDL file before the .SUBCKT command, you get warning messages for the device M1 as follows:

```
PROCESSING INPUT FILE: ./test.cdl  
*** WARNING: ELEMENT HAS WIDTH OR LENGTH WITH 0  
LINE      8:M1 D G S SUB MP  
*** WARNING: BJT HAS EAREA EQUAL 0  
LINE      9:Q1 C B E PN  
*** WARNING: ELEMENT HAS WIDTH OR LENGTH WITH 0  
NUMBER      1:M1              0              0  
*** WARNING: BJT HAS EAREA WITH 0  
NUMBER      2:Q1
```

***.UNSPEC-MOS**

*.UNSPEC-MOS

Description

When you use the *.UNSPEC-MOS command in your SPICE or CDL netlist, LOGLVS detects devices that do not have a width or length definition. Dracula issues a warning in the PRINT.OUT file.

You use this command only for checking MOS transistors.

Example

```
*.GLOBAL VDD, VSS
*.UNSPEC-MOS
```

```
.SUBCKT ENET D G S SUB C B E
***
M1 D G S SUB MP
Q1 C B E PNP
.END
```

When you place the *.UNSPEC-MOS command in the SPICE or CDL file before the .SUBCKT command, you get warning messages for the device M1 as follows:

```
*** WARNING: ELEMENT HAS WIDTH OR LENGTH WITH 0
LINE          7:M1 D G S SUB MP
*** WARNING: ELEMENT HAS WIDTH OR LENGTH WITH 0
NUMBER        1:M1
```


SPICE and HSPICE Commands

This section describes the HSPICE™ and SPICE commands you can use in a netlist file.

.ENDS or .EOM

`.ENDS / .EOM`

Description

Terminates a subcircuit definition.

The `.SUBCKT` or `.MACRO` commands begin a subcircuit definition. A subcircuit definition that begins with `.SUBCKT` must end with `.ENDS`. A subcircuit definition that begins with `.MACRO` must end with `.EOM`.

.GLOBAL

`.GLOBAL name... {:P/G}`

Description

Defines global node names such as power, ground, clock, or any node not passed in and out of subcircuits through the subcircuit I/O. If your version of the SPICE format does not have a `.GLOBAL` command, you can use `*.GLOBAL`. However, even with `*.GLOBAL`, any continuation lines must begin with a plus sign (+), not an asterisk-plus sign (*+).

Arguments

name

Specifies a list of global node names. The names that appear here must be the node names in the logic network.

:P/G

You can assign a pad type to a global name by adding `:P` or `:G` to the name. *P* stands for power, and *G* stands for ground.

The pad type must always attach to a global name or number even though this global name or number is later replaced by a new name with an `*.EQUIV` command.

Examples

In this example, VBB is a power pad.

```
.Global VSS, VDD, VBB:P
```

Two ground and power pads have unique names. In this application, VBB is a ground pad.

```
*.GLOBAL 1:P 0:G 99:G 2:P  
*.EQUIV VSS1=0, VDD1=1, VDD2=2, VBB=99
```

.INCLUDE

`.INCLUDE filename`

Description

Includes a file in the SPICE or CDL netlist. This is a standard SPICE statement.

Argument

filename

The name of a file to be included in the netlist.

Examples

```
**CDL/SPICE FILE
.INCLUDE MODEL.SPI
.INCLUDE QUAD1.SPI
.INCLUDE QUAD2.SPI
.INCLUDE QUAD3.SPI
.INCLUDE QUAD4.SPI
```

.PARAM

```
.PARAM parameter = value...
```

Description

LOGLVS supports HSPICE parameter usage. You must use this command with parameters in the HSPICE file even if you do not declare any global parameters.

LVS compares the following parameters only:

- Width and length of MOS and BJT devices
- Values and areas of capacitors
- Values and size of resistors
- Area and perimeter of diodes

Do not place an asterisk (*) in front of the .PARAM command. If you use an asterisk, Dracula ignores this command. To use the .PARAM command, set local parameters in the .SUBCKT command as shown below:

```
.SUBCKT AND 3 1 2 PA1=10, PA2=5
```

PA1 and *PA2* assign a default value to the parameters inside the subcircuit in case a global assignment is not made. You can use only numbers in the I/O listing. You can also set the parameters in the subcircuit call as follows:

```
X1 3 1 2 AND PA1=20 PA2=10
```

The value of *PA1* and *PA2* overrides a value assigned in the .SUBCKT command but does not override a globally set value.

You can use the parameters to replace the width and length specification in an MOS statement as follows:

```
.MAC NAND 1 2 3 WP=10 LPn=5  
M1 3 1 99 0 P WP LPn  
.  
.  
.  
.EOM
```

Note

If a parameter value is missing, LOGLVS issues an error message. In the following example the parameter value is missing in the subcircuit definition:

```
.PARAM
.SUBCKT TOP A B PW=
...
.ENDS
```

In this case the PRINT.OUT file contains the message:

```
2: .SUBCKT TOP A B PW=
*** ERROR: SYNTAX ERROR IN EQUATION
```

In other words, in the case where an undefined parameter is used in an expression, the value of the undefined parameter defaults to zero.

Arguments

parameter

Parameter names. Names must begin with an alphabetic character. There is no limit to the length of parameter names, but only the first four characters in a parameter name determine its uniqueness. For example, ABCDE is considered the same as ABCDF because the first four characters in both names are identical. If a parameter name over four characters appears more than once in the netlist, it is still counted only once.

Note: When the parameter name begins with a W or L character, it has special meaning as shown in the following example:

```
.PARAM
.SUBCKT A X Y Z WL=1 LW=2
M1 X Y X Y P WL LW
M2 Z Y Y Z N WL LW
.ENDS
```

In this case, both MOS M1 and M2 get Width=1 and Length=2.

value

Real numbers. The units of these numbers are ignored. To have the scale taken into consideration, specify the *.SCALE command.

Example

The following example shows a CDL netlist with parameter passing:

```
*TWO-SOME  THREE-SOME  INVERTER  CHAIN
.PARAM
*.GLOBAL  VDD  VSS
.SUBCKT    INV  OUT  IN   pw=10  pl=2  nw=5  nl=2
M1  VDD  IN  OUT  VDD  P  pw  pl  $  or  w=pw  l=pl
M2  OUT  IN  VSS  VSS  N  nw  nl  $  or  w=nw  l=nl
.ENDS
.SUBCKT    CHAIN  OUT  IN   pw1=10  nw1=5  pw2=30
+ nw2=15  pw3=90  nw3=45  nl1=2
X1  OUT  SIG2  INV  pw=pw3  nw=nw3  nl=nl1
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

```
X2  SIG2  SIG1  INV    pw=pw2  nw=nw2  nl=nl1
X3  SIG1  IN    INV    pw=pw1  nw=nw1  nl=nl1
.END
.SUBCKT  TOP    OUTA  OUTB  INA  INB
XA  OUTA  INA    CHAIN
XB  OUTB  INB    CHAIN  pw1=15  nw1=7.5  pw2=15
+ nw2=7.5  pw3=15  nw3=7.5  nl1=2.5
.ENDS
```

In the previous example, *OUTA*, *OUTB*, *INA*, *INB*, *VDD*, and *VSS* are the names for initial correspondence in LVS.

In inverter chain *A*, the output instance *P* device is 90/2 (W/L) and the *N* device is 45/2. The middle instance *P* device is 30/2 and the *N* device is 15/2. The input instance *P* device is 10/2 and the *N* device is 5/2.

In the inverter instance *B*, the three instances of *P* device and *N* device are 15/2 and 7.5/2.5, respectively. The parameter passing feature lets you specify the device size without many unique subcircuit definitions.

.SUBCKT or .MACRO

```
.SUBCKT/.MACRO subname[#ext] output-node... {/}  
    input-node...
```

Description

Begins a subcircuit definition. The two commands are equivalent.

The .ENDS or .EOM command ends a subcircuit definition. A subcircuit definition that begins with .SUBCKT must end with .ENDS. A subcircuit definition that begins with .MACRO must end with .EOM.

Note: Any subcircuit that does not contain a device is considered an element box and must have a corresponding element box definition in your Dracula rules file.

If you want to use your original SPICE data to run LVS, note the following:

- If you use an *.EQUIV command and the circuit file is a standard SPICE file (only node numbers and no node names), the .SUBCKT I/O must be numbers and not names.
- LVS does not allow nested .SUBCKT definitions. Dracula allows subcircuit calls inside .SUBCKT definitions, but not .SUBCKT definitions inside .SUBCKT definitions.
- Node numbers defined in the global statement cannot be included in the .SUBCKT I/O listing, except for a ground node, which is 0.

Arguments

subname

The element name.

#ext

The extension associated with this element. The pound sign (#) indicates that an extension follows. The extension is alphanumeric and can be up to eight characters long. It distinguishes elements that have the same name and I/O count but different implementations. The output node name of a device must have the same *#ext* string appended at the end.

output-node

The output node names of the element. For single output devices, only the first name is considered output. The rest is considered input.

Dracula Reference

Compiling Network Descriptions (LOGLVS)

/

Separates output names from input names.

Note: When parsing .SUBCKT names, LOGLVS treats “/” as a normal character instead of as a separator.

input-node

Input node names.

Example

```
.SUBCKT NAND#1 OUT / IN1 IN2
```

To specify a node as input or output, use the backslash (/) delimiter. LVS does not use of this information, but LPE does.

.SWAP and Special Elements

```
.SWAP pin-name...  
.SWAP (f/p {(f/p pin-name...)} pin-name... )...
```

Description

LOGLVS lets you specify a special element as a box. The .SWAP command specifies which I/O pins can be swapped.

In composite mode, the netlist for LVS, whether compiled from your schematic netlist or extracted from the layout database, contains a list of interconnected Hcell placements and a list of composite-level pins. The .SWAP command determines whether pins can be swapped on the Hcell placements.

Use the first syntax shown above for a one level swap. Use the second syntax line for a multiple level swap.

In a one-level swap, a group of pins can be interchanged among themselves, meaning that LVS processes all pins in the group the same way.

In a two-level swap, two groups of pins are interchangeable. The pins within each group can be interchangeable.

In a multiple level swap, two subtrees of pins are interchangeable. A subtree is a branch of a tree that represents how pins are grouped. Subtrees within each subtree can be interchangeable.

The following limitations apply to multiple level swappability:

- All limitations of one level swappability apply.
- Swappable trees must have the same structure.
- Multiple level swappability applies only to pins in an Hcell.

Argument

pin-name...

I/O names of the subcircuit. You must enclose the .SWAP specification by the .SUBCKT and .ENDS or .MACRO and .EOM commands.

You must designate whether the pin order is fixed (*f*) or permutable (*p*). If you do not want the pin order to change in processing, specify *f* before the list of pin names. If the pin order can change during processing, specify *p* before the list of pin names.

Note: There can be no element specification other than another `.SWAP`. A plus sign (+) indicates continuation to the next line. If you use a `.NOPIN` statement, it must precede the `.SWAP` statement.

Examples: One-Level Swap

More than one `.SWAP`:

```
.SUBCKT AX1 1 2 3 4
*.NOPIN pin1 pin2
.SWAP 1 2
.SWAP 3 4
.ENDS
```

Nodes *1* and *2* are class 1 terminals. Nodes *3* and *4* are class 2 terminals.

The following specification is illegal. The MOS element specification *M1* is not recognized when you use `.SWAP`.

```
.SUBCKT A1 OUT I1 I2 I3
M1      OUT1 I1 I2 O N
*.NOPIN pin1 pin2
.SWAP    I1    I2 I3
.ENDS
```

Examples: Multi-Level Swap

The circuits in the following examples use this `.SUBCKT` definition:

```
.SUBCKT SAMPLE A B C D E F G H I J K L
```

In the following example, pins *A* and *B* are not swappable. Pins *C* and *D* are not swappable. However, groups (*A B*) and (*C D*) are swappable. Pin *A* is at the same position as pin *C*, so they are swappable. Pins *B* and *D* are swappable.

```
CIR1:
.SWAP (p (f A B) (f C D))
```

In the following example, pins *A* and *B* are swappable, pins *C* and *D* are swappable, and groups (*f (p A B) E*) and (*f (p C D) F*) are swappable.

```
CIR2:
.SWAP (p (f (p A B) E) (f (p C D) F))
```

Subcircuit Calls

Xyyy name... {/} subname

Description

A description of the input and output pins of a circuit followed by the subcircuit name.

The .SUBCKT or .MACRO command begins a subcircuit call. The .ENDS or .EOM command ends a subcircuit call.

If the netlist file does not define the subcircuit, Dracula generates a warning message in the PRINT.OUT file that includes the cell name and its corresponding line number.

Arguments

Xyyy

The subcircuit call reference beginning with an X.

name

Lists the external reference node names. Dracula does not distinguish between input and output node names. Dracula refers to the names in the order you define in the subcircuit definition.

/

Indicates the end of the external node name specification.

subname

The subcircuit element name. If the original definition uses an extension, *subname* must include that extension. You do not need to define the subcircuit before listing it. LOGLVS accepts randomly ordered netlists.

Examples

The following example shows subcircuit definitions in a SPICE netlist file.

```
*.SPICE
.
.
.SUBCKT GRAYCODE 3 4 5 6 7
.
.
X1 14 8 INV
X2 16 9 INV
.
.
.ENDS
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

The following netlist file has missing subcircuit definitions:

```
*.GLOBAL VDD
.SUBCKT sub0 o i
M1 I O VDD VDD P L=5 W=10
.ENDS
.SUBCKT sub1 ao1 ai1
M1 VDD Ao1 1 VDD P L=5 W=10
M2 VDD Ai1 1 VDD P L=5 W=10
.ENDS
.SUBCKT sub2 bo1 bi1
M3 1 bo1 PP VDD P L=5 W=10
M4 1 bi1 2 VDD P L=5 W=10
Xsub12 PP 2 /subxxxxx
.ENDS
.SUBCKT TOP t1 t2
Xsub1 T1 ii /sub1
Xsub98790 t2 ii /sub99999999
.ENDS
```

The PRINT.OUT file contains the following error messages:

```
*** ERROR: SUBCIRCUIT :SUBXXXXX      NOT DEFINED at LINE: 12
*** ERROR: SUBCIRCUIT :SUB99999999   NOT DEFINED at LINE: 16
1) *** UNDEFINED ELEMENT SUB9 : XSUB98790
```

Pin Count Discrepancies

If the pin count in the .SUBCKT call is different than the number of pins in the subcircuit, LOGLVS outputs the following error in the PRINT.OUT file:

```
** ERROR : DEVICE TYPE JKFF1/GA IS DEFINED WITH 7 PINS, NOT 8 PINS
** ERROR ** DEVICE TYPE : JKFF1 OF ELEMENT N1A IS NOT DEFINED
```

If you ignore this error message, LVS might fail in LVSCHEM.

CDL Element Definition

The CDL defines the following elements:

- Capacitors
- Diodes
- Resistors
- Three types of transistors
 - Bipolar junction transistors (BJTs)
 - Junction field effect transistors (JFETs)
 - Metal oxide semiconductor field effect transistors (MOSFETs)

To define these elements, CDL uses the formats described in this section. Dracula ignores other elements.

Capacitor Syntax

```
Cxxx npositive nminus {{mname} value} {/mname} {M=multiplier} {$[mname]}  
      {$MODEL=mname} {$SUB=substrate}
```

Note: C= and AREA= parameters are not supported.

Arguments

Cxxx

The capacitor element. Must begin with the letter C.

npositive

The positive terminal node number/name.

nminus

The negative terminal node number/name.

mname

Specifies the model name reference. The name must be one or two characters. If you want model/subtype checking, you must specify the *mname* in the CDL netlist and specify the ELEMENT CAP subtype option in your rules file.

value

The capacitance value or the area, depending upon *.CAPVAL or *.CAPAREA commands. Dracula does not check the value if this specification is missing.

multiplier

The number of elements expressed as an integer greater than or equal to one.

substrate

Specifies substrate terminal or node name.

Note: If you require a polarity check, you cannot swap the *npositive* and *nminus* terminals with the ELEMENT CAP *layer-b* and *layer-c*. You must specify the LVSCCHK[P] command in your rules file. If the CDL netlist contains capacitors to be checked by LVS, you might want to use the *.BIPOLAR command. From the 4.7 release, Dracula will support *multiplier* for Capacity. If you specify *multiplier* Capacity in the CDL netlist and do not want to check M-Factor in LVSCCHK, the parameters of these devices will be recalculated so that the effective values will be processed in the LVS flow.

Example

```
C23  neta netb 3.5p M=2 $.MODEL=CX
```

Diode Syntax

```
Dxxx npositive nminus mname {area {periphery}} {AREA=area} {PJ=periphery}  
      {M=multiplier} {$SUB=substrate}
```



AREA and PJ are read only when *.DIOAREA and *.DIOPERI keywords are used.

Arguments

Dxxx

The diode element name. Must begin with the letter *D*.

npositive

The positive terminal node number/name.

nminus

The negative terminal node number/name.

mname

The model name reference. The name must be one or two characters. For model/subtype checking, you must specify the *mname* in the CDL netlist and specify the ELEMENT DIO subtype option in your rules file. You cannot swap *npositive* and *nminus* terminals.

area

The diode area specification.

periphery

The periphery of junction.

multiplier

The number of elements expressed as an integer greater than or equal to one.

substrate

The substrate terminal or node name.

Note: If the CDL file contains diodes to be checked by LVS, you might want to use the *.BIPOLAR command. In the 4.7 release, Dracula will support *multiplier* for Diode. If you specify *multiplier* Diode in the CDL netlist and do not want to check M-Factor in LVSCCHK, the parameters of these devices will be recalculated so that the effective values will be processed in the LVS flow.

Example

```
D100 net_plus net_minus DD
```

Resistor Syntax

```
Rxxx term1 term2 {{mname} value} {/mname} {M=multiplier} {{W=width} {{L=length}  
    {[mname]} {$.MODEL=mname} {$SUB=substrate}}
```

Arguments

Rxxx

The resistor element name. Must begin with the letter *R*.

term1

The first terminal node number/name.

term2

The second terminal node number/mname.

Dracula Reference

Compiling Network Descriptions (LOGLVS)

mname

The model name reference. The name must be one or two characters. For model/subtype checking, you must specify the *mname* in the CDL netlist and specify the ELEMENT RES command with a subtype option in your rules file.

value

The resistance value or the size, depending upon *.RESVAL or *.RESSIZE commands. Dracula does not check the value if this specification is missing.

multiplier

The number of elements expressed as an integer greater than or equal to one.

width

The width of the resistor.

length

The length of the resistor.

substrate

The substrate terminal or node name.

Note: If the CDL file contains resistors to be checked by LVS, you can use the *.BIPOLAR command. From the 4.7 release, Dracula will support *multiplier* for Resistor. If you specify *multiplier* Resistor in the CDL netlist and do not want to check M-Factor in LVSCCHK, the parameters of these devices will be recalculated so that the effective values will be processed in the LVS flow.

Example

```
R11 net3 net7 5.9 M=3 $[RX]
```

Transistor (BJT) Syntax

```
Q{name} coll base emitter {nsub} {/}mname {M=multiplier} {$SUB=substrate}  
    {$EA=value} {$L=length} {$W=width}
```

Note: \$EA value may be set using AREA parameter or inline when *.SPICE command is used:

```
Q{name} coll base emitter {nsub} {/}mname {{AREA=}value} {M=multiplier}  
    {$SUB=substrate} {$EA=value} {$L=length} {$W=width}
```

Arguments

Qxxx

The BJT element name. Must begin with the letter *Q*.

coll

The collector terminal node name.

base

The base terminal node number/name.

emitter

The emitter terminal node number/name.

nsub

Substrate terminal node name or number.

mname

The model name reference. The name can be one or two characters.

multiplier

Specifies an integer greater than or equal to one. You must place *multiplier* in the description before you specify any value.

value

Specifies an emitter size, coded as a comment.

length

The length of the BJT.

width

The width of the BJT.

substrate

The substrate terminal or node name.

M-factor commands define multipliers globally. The information here lets you specifically define multipliers for different circuit elements. See [Chapter 11, “Description Block Commands,”](#) for information about M-factor commands.

Example

```
.PARAM
*.GLOBAL VCC VSS
*.BIPOLAR
.SUBCKT TOP A B AB
....
```

```
Q1 2 5 1 LP M=4 $EA=30 $W=10 $L=3
....
....
.ENDS
```

Transistor (JFET) Syntax

```
Jxxx drain gate source {/}mname {M=multiplier} {W=width} {L=length}
```

Arguments

Jxxx

The JFET element name. Must begin with the letter *J*.

drain

The drain terminal node number/name.

gate

The gate terminal node number/name.

source

The source terminal node number/name.

mname

The model name reference. The name can be one or two characters.

multiplier

Specifies an integer greater than or equal to one. You must place *multiplier* in the description before you specify any value.

width

The width. *Width* is a real number. Dracula does not check the width if it is not specified. *W=* is optional.

length

The length. *Length* is a real number. Dracula does not check the length if it is not specified. *L=* is optional.

Note: You can specify the width and length with two real numbers following the model name. The first real number is width, the second number is length.

Note: To use parameters as discussed in HSPICE format, use the .PARAM command.

M-factor commands define multipliers globally. The information here lets you specifically define multipliers for different circuit elements. See [Chapter 11, “Description Block Commands,”](#) for information about M-factor commands.

Example

```
J01 net0 net1 net JP W=11 L=2
```

Transistor (MOSFET) Syntax

```
Mxxx drain gate source bulk {/}mname {width {length}} {W=width} {L=length}  
    {M=multiplier} {$LDD[type]} {$NONSWAP}
```

Arguments

Mxxx

The MOSFET element name. It must begin with the letter *M*.

drain

The drain terminal node number/name.

gate

The gate terminal node number/name.

source

The source terminal node number/name.

bulk

The bulk terminal node number/name.

mname

The model name reference. Must be one or two characters.

multiplier

Specifies an integer greater than or equal to one. You must place *multiplier* in the description before you specify any value.

width

The optional width. Dracula does not check the width if this specification is missing.

length

The optional length. Dracula does not check the length if this specification is missing.

type

The LDD designator, specified as a comment. Declares that the source and drain terminals have different characteristics and cannot be swapped.

M-factor commands define multipliers globally, while the information here lets you specifically define multipliers for different circuit elements. For more information about M-factor commands, refer to [Chapter 11, “Description Block Commands.”](#)

Example 1

```
*.GLOBAL VCC VSS
*.BIPOLAR
.SUBCKT TOP A B AB
.....
M1 A B AB LP N M=4 W=10 L=3
```

Example 2

```
M1 A B VSS N LDD L=2.0 W=10.0 $LDD[N]
```

You can stop generating pseudo gates when running an LVS check by using model name type *X-*. Dracula prevents any model name (*mname*) with an *X* as its first character from forming pseudo gates. This lets LVS locally stop the formation of pseudo gates in the schematic and allows LVS checking of mixed analog and digital circuits. This special model type works with the ELEMENT MOS[*X-*] command.

CDL Macro Expansion

CDL has its own macro expansion capability. Although LOGLVS allows more than 30 levels of macro expansion, it stores element names up to 125 characters and truncates characters over the limit.

Dracula expands macros as follows. A .SUBCKT command defines element *X001* as Box1. In Box1 there is another subcircuit called Box2 with the element name *X002*. In Box2 there is a third subcircuit called Box3 with the element name *X003*. The name of the transistor in Box3 is given in reverse order, from the lowest to the highest level:

```
J001X003X002X001
```

In addition to the element name-- *JXXX*, *MXXX*, *QXXX*, or *XYYY*-- every node in the subcircuit definition has a node name or a node number. Dracula expands the node name differently. If the subcircuit is called A1 in the logic network, the name of *node 1* inside Box 3 is given from the highest to the lowest level:

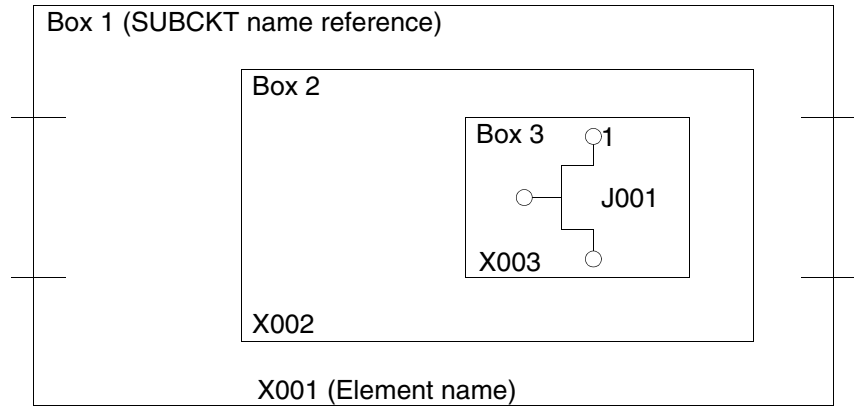
```
A1-X001-X002-X003-1
```

This node name expansion is consistent with the name expansion in the logic network.

Dracula Reference

Compiling Network Descriptions (LOGLVS)

Macro Expansion



CDL File Example

Here is a partial example of a CDL file. Remember that in CDL you must list all four terminals (drain, gate, source, and body) when you define MOS devices.

```
*.GLOBAL VDD,VSS
*
* NA2
.SUBCKT NA2 Z / A B
* 2 INPUT NAND
*TRANSNAME DRAIN GATE SOURCE BODY MODEL
MP1 Z A VDD VDD P WP=12 LP=14
MP2 Z B VDD VDD P WP=12 LP=14
MN1 Z A 5 VSS N WN=12 LN=14
MN2 5 B VSS VSS N WN=12 LN=14
.ENDS NA2
*
*
* NA3
.SUBCKT NA3 Z / A B C
* 3 INPUT NAND
MP1 Z A VDD VDD P
MP2 Z B VDD VDD P
MP3 Z C VDD VDD P
MN1 Z A 6 VSS N
MN2 6 B 7 VSS N
MN3 7 C VSS VSS N
.ENDS NA3
*
*
* NA4
.SUBCKT NA4 Z / A B C D
* 4 INPUT NAND
MP1 Z A VDD VDD P
MP2 Z B VDD VDD P
MP3 Z C VDD VDD P
MP4 Z D VDD VDD P
MN1 Z A 7 VSS N
MN2 7 B 8 VSS N
MN3 8 C 9 VSS N
MN4 9 D VSS VSS N
.ENDS NA4
* NOR
.SUBCKT NOR Z / A B
* 2 INPUT NOR
MP1 5 B VDD VDD P WP=22 LP=24
MP2 Z A 5 VDD P WP=22 LP=24
MN1 Z A VSS VSS N WN=12 LN=14
MN2 Z B VSS VSS N WN=12 LN=14
.ENDS NOR
*
* IV
.SUBCKT INV Z / A
* INVERTER
MP1 Z A VDD VDD P WP=12 LP=14
MN1 Z A VSS VSS N WN=12 LN=14
.ENDS INV
*
.SUBCKT JKFF1 Q QN / CL J K S R
MP1 6 CL VDD VDD P WP=12 LP=14
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

```
MN2 6 CL VSS VSS N WN=12 LN=14
MP3 5 6 VDD VDD P WP=12 LP=14
MN4 5 6 VSS VSS N WN=12 LN=14
MP5 9 6 7 VDD P WP=12 LP=14
MN6 9 5 7 VSS N WN=12 LN=14
MP7 44 5 7 VDD P WP=12 LP=14
MN8 44 6 7 VSS N WN=12 LN=14
MP9 8 7 VDD VDD P WP=12 LP=14
MP27 8 R VDD VDD P WP=12 LP=14
MN10 8 7 17 VSS N WN=12 LN=14
MN28 17 R VSS VSS N WN=12 LN=14
MP11 9 8 VDD VDD P WP=12 LP=14
MN12 9 8 10 VSS N WN=12 LN=14
MP13 9 S VDD VDD P WP=12 LP=14
MN14 10 S VSS VSS N WN=12 LN=14
MP15 QN 5 11 VDD P WP=12 LP=14
MN16 QN 6 11 VSS N WN=12 LN=14
MP17 8 6 11 VDD P WP=12 LP=14
MN18 8 5 11 VSS N WN=12 LN=14
MP19 Q 11 VDD VDD P WP=12 LP=14
MN20 Q 11 13 VSS N WN=12 LN=14
MP21 Q S VDD VDD P WP=12 LP=14
MN22 13 S VSS VSS N WN=12 LN=14
MP23 QN Q VDD VDD P WP=12 LP=14
MP25 QN R VDD VDD P WP=12 LP=14
MN24 QN Q 18 VSS N WN=12 LN=14
MN26 18 R VSS VSS N WN=12 LN=14
MP37 39 J VDD VDD P WP=12 LP=14
MP38 39 QN VDD VDD P WP=12 LP=14
MP39 44 39 VDD VDD P WP=12 LP=14
MP40 43 QN VDD VDD P WP=12 LP=14
MP41 44 K 43 VDD P WP=12 LP=14
MN42 39 QN 42 VSS N WN=12 LN=14
MN43 42 J VSS VSS N WN=12 LN=14
MN44 44 39 45 VSS N WN=12 LN=14
MN45 45 QN VSS VSS N WN=12 LN=14
MN46 45 K VSS VSS N WN=12 LN=14
.ENDS JKFF1
*
```

Preparing Data Files

This section explains how to use the following data formats to describe your network:

- SPICE
- VERILOG

Preparing SPICE Files

To use the original SPICE data (no logic-level description) to run LVS, follow these guidelines:

- Enclose all element statements in a set of .SUBCKT and .ENDS or .MACRO and .EOM commands. You must also include all of the highest level SPICE data statements in a set

Dracula Reference

Compiling Network Descriptions (LOGLVS)

of .SUBCKT and .ENDS or .MACRO and .EOM commands. The I/O names of this macro are candidates for initial correspondence node pairs.

- Dracula does not allow nested .SUBCKT definitions. You can nest .SUBCKT calls inside a .SUBCKT definition but not a .SUBCKT definition inside another .SUBCKT definition.
- Except for a ground node, which is assumed to be 0, node numbers defined in the global statement cannot be included in the .SUBCKT I/O listing.
- The power name must be VCC or VDD. The ground node name must be VSS or GND. Dracula does not recognize any other name in the *.EQUIV command as power or ground.
- *.EQUIV assigns a name to a node number; it cannot connect two nodes. For example, nodes one and two are two nodes in the SPICE data file. If you specify *.EQUIV VCC=1, VCC=2, Dracula assigns the name VCC to both node one and node two, but they are still two different nodes.
- To connect two nodes in the SPICE file, specify the voltage between them as zero. For example,

```
v1 10,0,0
```

Dracula connects node 10 and node 0.

You can connect any independent node to a global node or an I/O node. You cannot connect global nodes to global nodes.

- Do not use the following characters in node names:

, ; : = / tab space

You can use a slash (/) when defining device names in the CDL netlist file.

- The dollar sign (\$) and asterisk (*) are comment indicators. They are also specially interpreted by DRACULA in several cases. For that reason, do not use the dollar sign (\$) and asterisk (*) in node/device/parameter names in CDL netlists. When it is necessary to use dollar and asterisk signs, follow the syntax described in “CDL Control Commands” and “CDL Element Definition”, or problems can occur.

SPICE File Example

The .GLOBAL command declares power and ground nodes.

The .SPICE command indicates that this is a pure SPICE file and nodes can only be represented by a number. Use the .EQUIV command to assign a name to a node number.

Dracula Reference

Compiling Network Descriptions (LOGLVS)

The power node must be VCC or VDD. A ground node can be either VSS or GND. Nodes on the I/O listing of a .SUBCKT command are used as initial correspondence nodes.

```

NETLIST FOR GRAYCODE IN SPICE
*FILE NAME = GRAYCODE.SPI
*.SPICE
.GLOBAL 1 0
*.EQUIV Q1B=3 Q2B=4 Q3B=5 Q4B=6 CO=7
*+CA=8 MC1=9 SCT1=10 SCT2=11 SCT3=12 *+SCT4=13
.SUBCKT GRAYCODE 3 4 5 6 7
+8 9 10 11 12 13
X1 14 8 INV
X2 16 9 INV
XN1A 17 3 14 1 1 17 16 JKFF1
X3 17 10 INV
X4 18 11 INV
XN1B 19 4 14 17 17 18 16 JKFF1
X5 20 17 19 NA2
X6 21 20 INV
XR1A 22 5 14 21 21 23 16 JKFF1
X7 23 12 INV
X8 24 17 19 22 NA3
X9 25 24 INV
XR1B 26 6 14 25 25 27 16 JKFF1
XGH3D 27 13 INV
XTU2 28 17 19 22 26 NA4
X10 29 28 8 NOR
XRS2C 7 29 INV
.ENDS

```

Preparing Verilog Files

LOGLVS does not read Verilog directly (obsolete VER command is disabled). LOGLVS can read CIR (Common Interface Representation data) format produced by VAN (Verilog analyzer) program. An example of preparing CIR data:

1. Create a directory for a library (mkdir <lib_name>):

```
$ mkdir mylib
```

2. Create or edit the `cds.lib` file to add the new library:
("DEFINE <lib_name> <lib_dir_name>")

```
DEFINE mylib ./mylib
```

3. Run VAN to create the design library:
(van -lib <lib_name> <verilog_netlist>)

```
$ van -notienamed -lib mylib ../top.v
```

The following is an example of running LOGLVS ("NVER <lib_name> <top_cell>"):

```
:case
:cir sub.cdl /* CDL/SPICE netlist files should be imported first */
:nver mylib top /* Verilog netlist files
:con
:x
```

For composite mode, the command script is as follows:

```
:case
:cell/bbox cell.tab /* Use the box option for composite mode jobs */
:cir sub.cdl /* CDL/SPICE netlist files should be imported first */
:nver mylib top /* Verilog netlist files */
:con
:x
```

Important

VAN is case sensitive by default. You should either use the `-caseinsensitive` option of VAN, or use the `CASE` command of LOGLVS and names in netlists should correspond considering case. Specify the `CASE` command prior to lines that affect parsing.

When running a Verilog netlist in LOGLVS, you must supply a SPICE or CDL netlist with the Verilog file even if the Verilog netlist has transistor-level definitions. The SPICE or CDL netlist supplies the p-transistor and n-transistor level definitions. When preparing your Verilog files, take the following precautions:

Dracula Reference

Compiling Network Descriptions (LOGLVS)

- Check that the maximum line length of the Verilog netlist file is 128 characters or fewer, and that module or signal instance names have 80 characters or fewer.
- Match the input and output pin order from the CDL or SPICE definition to the Verilog file.

For example, if the Verilog file is

```
FF2 A1 (Q1,Q2,I1,I2,I3)
```

the subcircuit definition is

```
.SUBCKT FF2 Q QB R S C
.....
.ENDS
```

The results include the following connections:

```
Q1 to Q
Q2 to QB
I1 to R
I2 to S
I3 to C
```

If you specify pin names, LOGLVS checks that the pin names match between the Verilog description and the CDL or SPICE description. For example,

```
FF2 A1 (.Q(Q1),.QB(Q2),.C(I3),.R(I1),.S(I2));
```

Regardless of the sequence you define in the subcircuit, the connections are the same as in the Verilog file.

By default, VAN maps all power definitions to a global signal "tie1" and all ground definitions to a global signal "tie0".

If there is only one power/ground pair, VAN can be used in this mode. To get the correct connectivity, these signals need to be assigned to the corresponding Verilog names using *.EQUIV command:

```
*.GLOBAL VSS VDD $ CDL globals
*.GLOBAL tie0:G tie1:P $ true Verilog globals
*.EQUIV VSS=tie0 VDD=tie1 $ equating globals
```

To process a netlist with multiple power/ground signals, run VAN with the -notienamed option:

```
van -lib <lib_name> -notienamed <verilog_netlist>
```

Note: When -notienamed is specified, VAN treats all supply signals as internal wires. To make LOGLVS treat these signals as global signals, they must be defined in CDL as global by *.GLOBAL command:

```
*.GLOBAL VSS VDD1 VDD2
```

Nets assigned to 1'b0 and 1'b1 will be tied to global nets "tie0" and "tie1" even when the `-notienamed` option is used:

```
module top ;
wire a, b ;
bufd1 I0 ( .A(1'b1), .X(a) );
bufd1 I1 ( .A(1'b0), .X(b) );
endmodule
```

These global signals must still be reassigned by `*.EQUIV` command to the corresponding global node names.

Limitations:

- ❑ All VERILOG global nodes must be explicitly defined as globals for LOGLVS by its command `GLOBAL` or CDL command `*.GLOBAL`.

Running LOGLVS

The Network Compiler, LOGLVS, runs interactively to compile schematic netlist data and combine it with circuit description data to create a transistor-level network description that Dracula can use for LVS comparison.

After loading both the schematic netlist and the circuit description files on the system, you are ready to run LOGLVS. To start running LOGLVS, type

```
LOGLVS
```

During a LOGLVS run, you can use the commands described in this section to control the run. You need to type only the first three characters of the command. You can type a maximum of 128 characters on each command line.

When LOGLVS finishes, it returns one of the following three status codes in the PRINT.OUT file:

```
0  no errors or warnings
1  warnings, but no errors
2  errors
```

If you want the status code to appear on the screen, use the following script:

```
LOGLVS <<!  
lib ecad.teg  
cir ecad.cir  
link  
con  
exit  
!  
set $status2 = $status  
if ($status2 = 1) then  
    echo "LOGLVS WARNING ***** exit $status2 *****"  
elseif ($status2 = 2) then
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

```
    echo "LOGLVS ERROR ***** exit $status2 *****"  
    exit  
endif
```

CASE

CASE

Description

Turns on case sensitivity for the netlist processing.

By default, Dracula converts all input files to uppercase before processing.

CELL

`CELL{/option} filename... {1}`

Description

Identifies the file that specifies Hcells. This file must contain the Hcell subcircuit names as well as the Hcell layout cell names. See the HCELL-FILE command description in [Chapter 11, “Description Block Commands,”](#) for information on this file.

The CELL command works only in the hierarchical modes; it does not work in flat mode.

Arguments

option

Directs the command output. Type one of the following:

<code>/print</code>	Directs output to the PRINT.OUT file.
<code>/type</code>	Directs output to the terminal, assuming that the terminal defaults to <i>file unit 6</i> .
<code>/store</code>	Directs output to the IMAGE.LIS file.
<code>/auto</code>	Automatically updates Hcell table files. If Hcells do not have a placement number and are not found in the logic-level or circuit files, the Hcell table files are added with a semicolon (;) comment character in column 1.
<code>/bbox</code>	Ignores netlist statements for all Hcells when compiling the LOGLVS netlist. Use this option for composite mode only.

Note: LOGLVS still supports the `/box` option, which is compatible with the previous versions of Dracula.

filename

Names of files containing Hcell table files. Each file name can have up to 80 characters, including the full path name.

1

Indicates that the first cell is to be considered a schematic cell name. You must include the “1” to compile a netlist that will be treated as the layout netlist in a Schematic Versus Schematic (SVS) check in either cell or composite mode.

Examples

In the following example, LOGLVS reads in Hcells from three files: *file-1*, *file-2*, and *file-3*.

```
CELL/type file-1 file-2 file-3
CELL hcell.tab 1
```

Dracula treats the first cell listed in HCELL.TAB as a schematic cell and performs an SVS check.

cell/bbox	hcell.tab
cir	files
con	cell-name
x	

Dracula drops the netlist statements from all Hcells.

CIRCUIT

`CIRCUIT{/option} filename...`

Description

Reads the circuit library file. The CIRCUIT command also compiles SPICE data files.

LOGLVS accepts circuit files in random .SUBCKT order.

Arguments

option

Directs the command output. Type one of the following:

<code>/print</code>	Directs output to the PRINT.OUT file.
<code>/type</code>	Directs output to the terminal, assuming that the terminal defaults to <i>file unit 6</i> .
<code>/store</code>	Directs output to the IMAGE.LIS file.
<code>/nodrop</code>	Does not drop any devices that do not have I/O pins. This option applies to all netlists compiled with the CIRCUIT command from the point you specify <i>nodrop</i> .

filename

Names of files containing the circuit library. If a subcircuit is defined twice, the second subcircuit definition overrides the first. Each file name can have up to 32 characters.

CONVERT

`CONVERT{/option}/notop {cellname}`

Description

Converts the logic network data into LVS format and saves it to disk. LOGLVS compares the elements in the logic schematic with the elements defined in the circuit library. If the names, extensions, and I/O counts match, LOGLVS expands the logic elements into transistors. After it checks and expands all logic elements in the schematic, it creates the LVSLOGIC.DAT file.

Arguments

option

Directs the command output. Type one of the following:

<code>/print</code>	Directs output to the PRINT.OUT file.
<code>/type</code>	Directs output to the terminal, assuming that the terminal defaults to <i>file unit 6</i> .

The default is one output file, LVSLOGIC.DAT.

notop

Specify this option in Cell mode to specify there is no top-level *.SUBCKT*. You use this option when verifying a standard library of cells that you have not yet placed in a design hierarchy.

cellname

The name of the top level subcircuit or macro to be converted. Use this option only when you use a SPICE data file with no logic network. When the cell name is absent, the logic network is converted.

DATAFORMAT

DATAFORMAT 4.2/ZLIB { *LEVEL* }

Description

Turns on the database compression algorithm. Dracula can take D3, 4.2, and ZLIB input files for the same module. D3 is the default.

Note: You must also specify this command in the Description block for LVS and non-LVS jobs.

Arguments

4.2

Database compression algorithm introduced in Dracula 4.2.

ZLIB

Zlib database compression algorithm. It has about 2 times better data compression ratio compared to 4.2 and up to 6 times better compared to the default compression method, depending on the testcase.

LEVEL

Zlib compression level. Set this value between 1 and 9. 9.9 is the maximum compression ratio and has less performance. 1 is the minimum compression ratio and has more performance. The default value is 6.

Note: The LOGLVS data compression algorithm is defined for LOGLVS output only and has no effect on jxrun.com run.

Example

DATAFORMAT = ZLIB

DELCKT

`DELCKT filename`

Description

Specifies that LOGLVS drops the .SUBCKT names you specify, whether the .SUBCKT is empty or not. LOGLVS does not create *box* elements for an empty .SUBCKT.

Argument

filename

Specifies the .SUBCKT names to delete.

Example

```
delckt delcel.tab
cir cdl
con top
x
```

The *delcel.tab* file looks like this.

```
tlow23    ---> subckt name to be dropped.
```

DELECEL

DELECEL *filename*

Description

Removes logic gates in logic netlists such as Verilog. The file format is as follows:

- The line starting with the string DELET is skipped.
- The string after the first “=” character is read in as the top-down hierarchical cell instance name and represents the gates and/or cells to delete.
- “/” is the hierarchical name delimiter. The string after the instance name is ignored.
- Blank is the delimiter.
- The maximum length of any line is 256 characters.

Argument

filename

Specifies the gates and/or cells to delete in the file hierarchy.

Example

```
N = ANITA/TPC/DC_10 T = AN4
N = ANITA/TPC/FD_10 T = FDN2P
N = ANITA/INR/IP0_1 T = IV
N = XX3939 T = BUF8A
N = XX3945 T = BUF8A
N = XX3943 T = BUF8A
N = XX3941 T = BUF8A
```

DEVMAP

DEVMAP *filename*

Description

Reads the device map file for instance primitive devices.

Argument

filename

A user-defined text file.

Each line in the map file contains the following mapping information:

- The mapping of an instance primitive device to its corresponding primitive device type.
- The mapping of the model of an instance primitive device to its corresponding layout element subtype. This information is "optional" and you can use it to overwrite the mapping specified by the CDL *.EQUIV command in the netlist.
- The mapping of the parameters defined for an instance primitive device to the pre-defined parameters of its corresponding primitive device. The parameter names defined on an instance primitive device can be different from those defined for its corresponding primitive device, but you need to provide the name mapping in order for LOGLVS to correctly interpret the parameters.

Given below is the syntax for each line in the map file.

```
subckt_name=primitive_device, {MODEL=layout_subtype,} {PARAMETER=
{user_defined_name1/pre-defined_parametername1,user_defined_name2/pre-
defined_parameter_name2, ... user_defined_nameN/pre-defined_parameternameN }
```

Where the names of the pre-defined parameters for the primitive devices are:

```
For MOS:   M   W   L
For BJT:   M   W   L   EA
For RES:   M   W   L   VRV (value)   SRV (size)
For CAP:   M   VCV (value)   ACA (area)
For DIO:   M   DA (area)   JPERI (periphery)
```

Example

Here's a netlist with two instance primitive devices `nmos_rf33` and `pmos_rf33`.

```
.GLOBAL vss vdd
*.EQUIV N=nmos_rf33 P=pmos_rf33
.subckt nmos_rf33 d g s b
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

```
.ends nmos_rf33
.subckt pmos_rf33 d g s b
.ends pmos_rf33
.subckt inv in out vdd vss
XMP0 out in vdd vdd pmos_rf33 lr=0.35u nr=32 wr=2.5u
XMN0 out in vss vss nmos_rf33 lr=0.30u nr=32 wr=2.5u
.ends inv
```

Given below is an example of the device map file for this netlist.

```
nmos_rf33=MOS, MODEL=N1, PARAMETER=lr/L, nr/M, wr/W
pmos_rf33=MOS, PARAMETER=lr/L, nr/M, wr/W
```

In the device map file, `nmos_rf33` and `pmos_rf33` are both MOS devices. The corresponding layout element subtype for `nmos_rf33` is `N1` which overwrites the subtype `N` defined by `*.EQUIV` in the netlist. The corresponding element subtype for `pmos_rf33` is `P` as specified by `*.EQUIV`. The parameter `lr` means MOS channel length (lr/L), `wr` means channel width (wr/W) and `nr` is m-factor (nr/M).

You need to provide this mapping information and the LVS comparison on instances of these instance devices are the same as the LVS comparison on conventional MOS devices.



You must use this command before any parse command.

DXF

DXF

Description

Generates a cross reference file, EXPELE.LIS, which contains device numbers followed by the full path name for that device.



You must use this command before inputting logic or circuit-level files.

ENV

`ENV new_number`

Description

Increases the user-defined module name from the default of 1847 to a new value greater than or equal to the new number. The maximum number you can set has been increased from 10,000 to 30,000.

Arguments

`new_number`

Number of modules you can set.

Example

`ENV 30000`

EQUIV

`EQUIV new-name = old-name`

`EQUIV new-model = old-model`

Description

The EQUIV command works in one of three ways:

- Equates node names in a layout database to the corresponding node number in the netlist
- Equates element subtypes in a Dracula rules file to model names in a CDL file
- Equates power and ground nodes. For example,

`EQUIV VDD=vdd1 VDD=vdd2`

Arguments

new-name

Layout text.

old-name

Schematic node name or SPICE or CDL node number.

new-model

Element subtype.

old-model

SPICE/CDL model name.

Example

In this version and subsequent versions, you can use the .EQUIV command (from CDL) also as a LOGLVS command and it works similar to the GLOBAL command.

`EQUIV VDD=TIE1 GND=TIE2`

`EQUIV VDD=TIE3 GND=TIE4`

`cir cdl`

Note: You do not have to edit your netlist when you use this command in this way, but you have to limit your command length by 78 characters. So when you use EQUIV as a LOGLVS command, the LOGLVS program overrides subsequent *.EQUIV statements in your CDL files without changing the statements.

EXIT

EXIT/X

Description

Stops the LOGLVS program. LOGLVS recognizes both EXIT and X.

When LOGLVS finishes, it returns one of the following three status codes in the PRINT.OUT file:

```
0   No errors or warnings
1   Warnings, but no errors
2   Errors
```

Example

The following is a C-shell example:

```
LOGLVS <<!
lib ecad.teg
cir ecad.cir
link
con
x
!
If ($status != 0 ) exit 1
```

FDELIMITER

`*.FDELIMITER string`

Description

Specifies delimiter characters for feedthrough pin names on the layout text. This function lets you specify separate feedthrough pin names on the layout and prevents Dracula from shorting all feedthrough pins. You can use `*.FDELIMITER` only in flat mode. You must specify all feedthrough pin names in the CDL netlist file at the `.SUBCKT` statement. Specify the `*.FDELIMITER` command before the input netlist. LOGLVS generates a SCHFPIN.DAT file containing the feedthrough pin information.

When running PRE, the feedthrough pin text must be on the pad layer.

Argument

`string`

One to four characters. The string cannot contain a space, a colon (:), a semicolon (;), or a question mark (?). If you specify a string longer than four characters, Dracula reports an error.

Example

In the following example, all pins with prefix `Z@@10` are shorted together. The resultant pin name is `Z@@10`.

```
LOGLVS !<<
FDELIMITER &&
CIR CDL.netlist
CON TOP
X
!
.SUBCKT TOP OUT Z@@10&&86 Z@@10&&87 Z@@13&&88 Z@@13&&89 IP
...
```

FPIN

FPIN

Description

Turns on the insertion of missing floating pins in instances when compiling Verilog netlist. The FPIN command is recommended whenever connection by explicit pin name is used in Verilog.

When using connections by explicit pin name in Verilog, an instance pin can be missing if it is floating. For example, in the following module, *BUF* has pins *I1*, *I2*, and *O*. When placed in another module, the *I1* pin is floating. The Verilog statement looks like this:

```
BUF X1 ( .I2(NET1), .O(NET2) ); /* instance pin I1 is
                                floating */
BUF X2 ( .I2(NET2), .O(SET), .I1(TET) )
```

To ensure the correct connection, use the FPIN command to enable the automatic floating node insertion function.

Use the following command sequence when you use the FPIN command:

1. Issue the FPIN command.
2. Compile the CDL or SPICE netlist.
3. Compile the Verilog netlist.

GENPAD

GENPAD

Description

Generates a special internal schematic initial correspondence file, 6GPADS.DAT, which forces the LVS REDUCE module to use only names that are in both 6PADL.DAT and 6GPADS.DAT. This eliminates additional layout text that would cause REDUCE to form devices differently than the schematic side.

Use GENPAD before using the CONVERT command.

GET

`GET data`

Description

Retrieves circuit library data or logic network data from disk files created by the SAVE command.

Arguments

data

One of the following three values:

<code>all</code>	Restores both circuit library and logic network data.
<code>circuit</code>	Restores only circuit library data.
<code>logic</code>	Restores only logic network data.

GLOBAL

```
GLOBAL name1[:[P|G]] name2[:[P|G]] ... nameN[:[P|G]]
```

```
GLOBAL pfx1*[:[P|G]] pfx2*[:[P|G]] ... pfxN*[:[P|G]]
```

Description

Lets you specify power and ground nodes in LOGLVS.

Note: The GLOBAL command accepts multiple usage. All GLOBAL commands are merged into a big *.GLOBAL command internally. The big internal *.GLOBAL command cannot be longer than 1280 characters.

Arguments

name

Specifies a list of global node names. The names that appear here must be the node names in the logic network.

:P/G

You can assign a pad type to a global name by adding *:P* or *:G* to the name. *P* stands for power, and *G* stands for ground.

The pad type must always attach to a global name or number even though this global name or number is later replaced by a new name with an *.EQUIV command.

Example

```
POWER-NODE=Vdd* Vcc*
```

```
GROUND-NODE=Gnd* Vss*
```

HELP

HELP command

Description

Displays information on your screen about a specified command.

Note: The HELP command might not display the most recent information about commands.

Arguments

command

The name of a LOGLVS command. You can abbreviate the command name to its first three letters. The commands for which you can get help are

CIRCUIT
CONVERT
COMMAND-FORM
EXIT
GET
INPUT
LIBRARY
LINK
NET
PRINT
RESET
SAVE
STORE
TYPE

Note: You must set the DRACHELP4 environment variable to the directory where the LOGLVS.HLP file is located (*path/etc/drac_help/*).

HTV/DRE

HTV **or** DRE

Description

Generates files Dracula Interactive LVS needs for cross-probing.

Note: You must use this command before you input logic or circuit-level files.

KCELL

KCELL <kcell_filename>

Description

Keeps cell texts in flat mode to improve LVSCHM performance.

Until the 4.8 release, Dracula kept only the top cell pin texts, but ignored all the pin texts of the cells below the top. This created two problems. The first problem being that without the information of the cell pin text, some of the parallel groups could be matched falsely. Secondly, without the cell pin text, the parallel groups tended to be bigger and slowed down the performance because breaking up of the parallel groups is costly.

In the 4.8 release, this new feature keeps the cell pin text in flat mode LVS thus eliminating false errors and improving performance. This feature is applicable to the flat mode *only*.

Arguments

kcell_filename

Specifies the name of the file containing the Kcell name tables

Example

```
KCELL ../myKcell.tab
CIR ../netlist.cdl
CON adder
```

KFPIN

KFPIN

Description

Suppresses the filtering of floating nodes in cell instances when running in composite mode. By default, Dracula assigns a floating pin on an HCELL instance a node number of zero. To maintain the original floating node number, you must specify the KFPIN command. When you specify the KFPIN command, LOGLVS assigns non-zero node numbers to the floating pins.

See also the KEEP-INST-FPIN command in Chapter 11 of this manual.

Example

Sample command script:

```
LOGLVS <<!  
KFPIN           ;Suppress the filtering of floating  
CELL HCELL.TAB ;nodes in cell instances.  
CIR lib.cdl  
LIB gray.teg  
LINK  
CONV  
X  
!
```

LINK

LINK {*top-module-name*}

Description

Links and expands the library data into a logic network.

Arguments

top-module-name

You must specify a top module name if you are using a Verilog netlist file.

MOS_EXTENDED_REP

MOS_EXTENDED_REP

Description

Disables the MOS/LDD M-factor/width adjustment.

By default, LOGLVS stores M-factor and width combined in one record. If the M-factor or width values are big enough to be stored unmodified, LOGLVS tries to recalculate their values accordingly and accuracy may be lost. For example, LOGLVS adjusts MOS/LDD M-factor to M=250 (and sets the width accordingly) if it encounters a MOS/LDD value of M>511. In some cases this situation is critical for LVS comparison.



This command must be used before the CONVERT command.

NO_WARNING

`NO_WARNING {argument}`

Description

Turns off the warnings from LOGLVS on ERC-type checks of the CDL netlist. The error messages are still printed.

To suppress warning messages about non-global names that are "re-equived" by the EQUIV command, specify the EQUIV_CHK argument.

Use NO_WARNING before the CIRCUIT command.

Arguments

EQUIV_CHK

Suppresses the warning messages from LOGLVS on non-global names which are "re-equived", for example *.EQUIV P=pch P=p5v. Otherwise, LOGLVS issues the following warning:

*** WARNING: <...> IS NOT A GLOBAL NAME. IT IS "REEQUIVED" AT LINE xx. IF IT IS A NET NAME, EQUIV WILL CREATE DIFFERENT NETS WITH THIS NAME***

NVER

```
NVER {/KEEP_BSLASH} library_name top_cell_name
```

Description

Used to parse Verilog netlists. NVER reads Common Interface Representation (CIR) generated by VAN instead of reading netlists directly.

Note: You do not need to run the LINK command after using NVER.

Arguments

/KEEP_BSLASH

Prevents Dracula from removing the leading backslash character ('\') from Verilog node names.

library_name

User-specified library name. If you do not specify a library name, LOGLVS uses the default name.

top_cell_name

User-defined cell name.

mylib

The default library name. If there are only two arguments, LOGLVS assumes mylib to be the default and uses the second argument as top_cell_name. LOGLVS also issues the following warning:

Library name missing. Assuming mylib.

Note: If mylib is the library name, LOGLVS continues after the warning. However, if mylib is not the library name, after the warning LOGLVS issues the following message and stops: Architecture not found in your design library.

Example

```
case
cir data_mux.cir
nver /KEEP_BSLASH lib_name bad_module
con/print bad_module
x
```

PRECISION

PRECISION #

Description

Lets you establish the precision by which to specify device parameters. The LVS precision cannot be better than LOGLVS.

Processes device parameters in .001 PRECISION for PRECISION 3, and .0001 for PRECISION 4.

Note: Outputs of this command such as LVSLOGIC.DAT can only be used by versions 4.7 or later. PRECISION 4 is only supported in the June 2005 release and later releases. You should run PDRACULA after LOGLVS., then the PRECISION value can be passed to LVS comparison when you use this command.

Arguments

#

Number of significant digits (2~4) after the decimal point. The default is 2. The total number of digits before the decimal point and after the decimal point cannot be larger than 6. In other cases Dracula does not guarantee precision for extra digit.

Precision in the DracToRCX Flow

In the case of the DracToRCX flow, the PRECISION command is not directly supported, but instead must be defined using an environment variable:

- `setenv LVS_PRECISION_3` (to define a three decimal place precision)
- `unsetenv LVS_PRECISION_3` (to restore the default value of two decimal place precision)

PRINT

`PRINT lib`

Description

Directs LOGLVS output to a printer.

Arguments

lib

One of the following values:

CIRCUIT	Acts on a circuit library.
LOGIC	Acts on the logic network.

RESET

RESET *data*

Description

Resets the circuit library data, logic network data, or the logic library data.

Arguments

data

One of the following values:

<code>all</code>	Resets both circuit library data and logic network data.
<code>circuit</code>	Resets only circuit library data.
<code>logic</code>	Resets only logic data.

RESISTOR

RESISTOR *value*

Description

Presets the resistor numbers. LOGLVS obtains swap spaces according to this resistor number.

Argument

value

The maximum number of resistors allowed in the circuit. The default is 10,000. If the resistor number is less than the default, the specified number is ignored.



This command automatically restarts LOGLVS without any warning.

SAVE

SAVE data

Description

Saves the circuit library data or the logic library data files in program-assigned files. The SAVE command provides a reference file for LPEPRO. If you do not plan to run LPEPRO, you do not need to use SAVE.

Arguments

data

Contains one of the following three values:

ALL	Saves both circuit library and logic network data.
CIRCUIT	Saves only circuit library data.
LOGIC	Saves only logic data network. You must issue the SAVE LOGIC command before the CONVERT command.

SET FANIN

`SET FANIN value`

Description

Specifies the maximum fan-in that an instance can have. If the value that you specify is bigger than the current, Dracula uses as the new value the closest power of 2 that is bigger than the specified value or equal to the specified value. Currently fan-in cannot be set bigger than 2^{21} (2097152). The larger the fan-in limit is, the lower the fan-out limit is.

Argument

value

Sets the fan-in value. The default is 2^{15} (32768).

Example

In the following example, the fan-in value is changed to 2^{19} (524,288):

```
: SET FANIN 300000
*/W* SET MAX FAN-OUT=      4096 MAX FAN-IN=      524288
```

In the next example, the fan-in value is not changed, because it is not bigger than the default value:

```
: SET FANIN 30000
```

SET FANOUT

`SET FANOUT value`

Description

Specifies the maximum fan-out that an instance can have. If the value that you specify is bigger than the current, Dracula uses the closest power of 2 that is bigger than the specified value or equal to the specified value as the new value. Currently fan-out cannot be set bigger than 2^{20} (1048576). The larger the fan-out limit is, the lower the fan-in limit is.

Argument

value

Sets the fan-out value. The default is 2^{16} (65536).

Example

In the following example, the fan-out value is changed to 2^{18} (262,144):

```
: SET FANOUT 200000
*/W* SET MAX FAN-OUT=      262144 MAX FAN-IN=      8192
```

In the next example, the fan-out value is not changed, because it is not bigger than the default value:

```
: SET FANOUT 20000
```


STOP

STOP {ON} [ERRORS | WARNINGS | NOTHING]

Description

This command is provided to better manage running LOGLVS from batch scripts. When new errors or warnings are found, LOGLVS will stop execution and prompt the user if it is “OK” to continue before proceeding. If the user does not specify “OK”, LOGLVS will exit.

Argument

{ON}

Optional Keyword used only to clarify the purpose of the command. Stop upon finding errors or warnings or nothing.

ERRORS

Stop execution and wait for user input after new errors, but not after new warnings.

WARNINGS

Stop execution and wait for user input after either new errors or new warnings.

NOTHING

Do not stop execution. This is the default value.

Example

```
LOGLVS << ! > loglvs.log
STOP ON WARNINGS
cir a.cdl
...
```

STORE

STORE *lib*

Description

Stores LOGLVS output in a file, or the circuit library.

Arguments

lib

One of the following values:

CIRCUIT	Acts on a circuit library.
LOGIC	Acts on the logic network.

SUMMARY

SUMMARY

Description

Generates a summary of devices according to the device types and subtypes. Use SUMMARY after the CONVERT command.

TNAME

TNAME

Description

Lets you use top-level names as internal wires in the top cell when your design is composed of Verilog and CDL netlists. If you do not specify this command, LOGLVS does not use the top-level names for internal wires.

TRANSISTOR

TRANSISTOR *value*

Description

Presets the transistor numbers. LOGLVS obtains swap spaces according to this transistor number. In order for this command to be effective, you must specify it first in the rules file.

Arguments

value

The number of transistors needed by the circuit. The default is 100,000. If the transistor number you specify is smaller than the default, this value is ignored.



This command automatically restarts LOGLVS without any warning.

TYPE

TYPE *lib*

Description

Directs LOGLVS output to the terminal.

Arguments

lib

One of the following values:

CIRCUIT	Acts on a circuit library.
LOGIC	Acts on the logic network.

UNLIMIT

UNLIMIT

Description

This command is no longer supported. By default, LOGLVS creates a cross reference file, CROSREF.LIS, for long texts and their encoded string. If the net names or cell names are long, CROSREF.LIS contains name mapping information. The CROSREF.LIS file is sorted and each entry appears only once.

The PRINT.OUT file no longer contains the cross reference list. A message appears in the PRINT.OUT file referring you to the CROSREF.LIS file for cross reference information.

Example

The following example shows what the cross-reference pair in the CROSREF.LIS file looks like.

```
*1    = longStringLongerThan12Chars
*2    = Xlevel1InTopCell
*3    = MDeletionDv201
```

Shell Environment Variables Used By LOGLVS

The following shell environment variables can be set to control specific processes in LOGLVS:

■ **CDL_BUSPORT_DESCEND**

Set this variable to give LOGLVS controls for interpreting busport order. For example, consider a Verilog module with the following description:

```
module ccu_ram ( clk , status_wen , tag_wen , data_wen , status_addr1, ...);  
...  
input [6:0] status_addr1 ;  
...  
sbit sbitb (.AA(status_addr1), ...);  
...  
endmodule
```

In addition, consider that in the CDL file there is the corresponding instance:

```
.SUBCKT sbit AA[0] AA[1] AA[2] AA[3] AA[4] AA[5] AA[6] ...  
...  
.ENDS
```

Since LOGLVS supports strictly positional busport matching between CDL and Verilog, pin AA[0] in CDL will be matched with status_addr1[6] in Verilog and so on. These situations can cause device matching problems during the LVS stage. Setting the CDL_BUSPORT_DESCEND variable to "setenv CDL_BUSPORT_DESCEND 1" will force LOGLVS to match pins in descending order and pin AA[i] will be matched with pin status_addr1[i] (for i=0,6).

■ **DRAC_EQUIVS**

Set this variable as the value for initial memory allocation for EQUIVs processing. To avoid the possibility of unnecessary autorestarts of LOGLVS, this variable is useful for netlists containing more than 10000 EQUIV statements. The legal value for DRAC_EQUIVS must be a positive integer number less than or equal to 500000.

■ **DRACHELP4**

Set this variable to point to the directory where the Dracula help files are located. This variable allows the HELP command to display information on your screen about a specified command. The following is the UNIX command for setting the environment:

```
setenv DRACHELP4 drac_dir/etc/drac_help/
```

■ **FNET_WARN_LIMIT**

Set this variable as the maximum number of floating net warnings which will be reported in the PRINT.OUT file. With very large designs, a large number of floating net warnings

can be reported in the PRINT.OUT file. Normally, these nets are expected to be floating by design. Having a large number of these warnings can disguise the important warnings and errors. The FNET_WARN_LIMIT environment variable can help to debug problems in these situations, but will not limit the overall number of warnings.

■ **MOS_EXTENDED_REP**

Set this variable to disable the MOS/LDD M-factor/width adjustment. See MOS_EXTENDED_REP on page 247 for more information.

Dracula Reference

Compiling Network Descriptions (LOGLVS)

LOGLVS Example

Start LOGLVS by typing

LOGLVS

Dracula displays the following:

```
*****
*/N*  LOGLVS      (REV. 4.9.06-2008          / LINUX          /GENDATE: xx-JUN/2008)
      *** ( Copyright 2008, Cadence ) ***
*/N*  EXEC TIME = 17:17:40    DATE = xx-JUN-2008    HOSTNAME =  mos102
*****
current xtrs estimate:      200000
```

```
DRACULA NETWORK COMPILER (LOGLVS) PROGRAM BEGINS ..
```

```
**** COMMANDS SUMMARY ****
```

```
---- COMPILE H-CELL TABLE FILE FIRST, -----
---- THEN SPICE FILE ----
```

- 1) DATAFORMAT [ZLIB [#]] : Output Compressed DataBase Format
- 2) TRANSISTOR # : pre-allocate virtual memory
- 3) HTV or DRE : generate draculaInteractive files
DXF : generate EXPELE.LIS/EXPELE.CEL
- 4) CASE : turn case sensitive on
- 5) NO_WARNING : turn connection checking off
- 6) GLOBAL : define global node names
- 7) PRECISION # : set # of dev parameter decimal points
- 8) CEL filename : compile H-CELL table file
CEL/AUTO fn : AUTO CELL selection
CEL/BBOX fn : HCELL becomes an empty SUBCKT (Blackbox) SPICE/CDL only
- 9) SET FANIN # : set maximum # of input pins
- 10) SET FANOUT # : set maximum # of output pins
- 11) ENVI # : set max. # of user defined cells/names (>2047)
- 12) CIR filename : compile SPICE/CDL netlist file
- 13) NVER libname cellname : read EDB compiled by VAN
- 14) LINK : expand logic network
- 15) GENPAD : generate 6GPADS.DAT file
- 16) CON : convert LOGIC NETWORK into XTR file

Dracula Reference

Compiling Network Descriptions (LOGLVS)

```
CON cellname : convert TOP CELL into XTR file
CON/NOTOP    : convert CELL into XTR file for DRAC3

17) SUMMARY   : print ELEMENT summary by types
18) EXIT/X/x  : EXIT

ENTER COMMAND
:
```

Using Hierarchical LOGLVS

When running LOGLVS for an HLVS run, you must first use the CELL command to input the Hcell file.

When you run hierarchical LOGLVS, a menu of commands appears on the screen. You can use the following commands with hierarchical LOGLVS. The “Running LOGLVS” section describes them in detail.

```
CASE
CELL
CIRCUIT
CONVERT
EXIT
INPUT
LIBRARY
LINK
NVER
```

The following examples show how to run hierarchical LOGLVS.

SPICE Examples

Flat mode

```
:CIR file-a ; Specifies CDL file.
:CON cell-name
:X
```

Cell mode

```
:CEL cell.tab
:CIR file-a ; Specifies CDL file.
:CON/NOTOP ; Indicates that file-a lists
           ; no top level circuits. This is optional.
:X
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

Composite mode

```
:CEL/bbox cell.tab
:CIR file-a ; Specifies CDL file.
:CON cell-name

:X
```

VERILOG Examples

Flat mode

```
:CIR file-b ; Specifies CDL file.
:NVER library top_cell ; Specifies Library and top_cell.
:CON ; As in flat mode Dracula.
:X
```

Cell mode

```
:CEL cell.tab ; cell.tab describes Hcells.
:CIR file-b ; Specifies CDL file.
:NVER library top_cell ; Specifies Library and top_cell.
:CON ; As in flat mode Dracula.
:X
```

Composite mode

```
:CEL/bbox cell.tab ; cell.tab describes Hcells.
:CIR file-b ; Specifies CDL file.
:NVER library top_cell ; Specifies Library and top_cell.
:CON ; As in flat mode Dracula.
:X
```

CDL and SPICE Formats

You can use a SPICE netlist for running cell mode and composite mode HLVS. The first character of each text item on the layout must be an alphabetic character. Because true SPICE format allows only numbers to specify nodes, use a CDL netlist containing names to identify nodes.

You can use the following commands in the CDL or SPICE subcircuit file:

```
*.PIN VDD VSS .... etc
```

Place the *.PIN command at the top of the CDL or SPICE subcircuit netlist to indicate that these global I/O terminals exist for each subcircuit.

If the *.PIN command for an HLVS netlist is missing from the CDL or SPICE file, LOGLVS issues a warning. LOGLVS also issues a warning if there are any unused outputs in a .SUBCKT command that contain multiple outputs. Here is an example:

```
.SUBCKT dff q qbar in clk clkbar set reset
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

If *q* is not used in the schematic netlist, Dracula issues a warning that *q* is used only once in the netlist.

```
*.NOPIN pin1 pin2 .... etc
```

This *.NOPIN command overrides the designation of the *.PIN command and specifies that these global I/O terminals are not part of this specific Hcell subcircuit.

HLVS accepts a true SPICE netlist. LOGLVS accepts a local *.EQUIV command in the .SUBCKT command. You must specify the *.EQUIV command before any device commands.

```
$ TRUE SPICE FILE $
*.SPICE
*.GLOBAL 0:G 99:P
*.PIN 0 99
*.EQUIV vss=0 vdd=99
*+ d6=11 d5=12 d4=13 d3=14 d2=15 d1=16
*+ g=23 o1=24 o2=25 o3=26 o4=27
*
.SUBCKT inv 3 6
*.EQUIV in=6 out=3
mnl 3 6 0 0 n 5 2
mpl 3 6 99 99 p 10 2
.ENDS inv

.SUBCKT top 11 12 13 14 15 16 23 24 25 26 27
.
.
x1 44 11 inv
x2 . . . . .
.
.
.ENDS top
```

Hcell Definition Examples

This section contains Hcell definition examples. The Hcell is named *HCELL1*. Its inputs are *A*, *B*, and *C*, and its outputs are *D*, *E*, and *F*.

Example 1

CDL

```
*.GLOBAL VDD,VSS
*.PIN VDD,VSS
.SUBCKT HCELL1 D E F A B C
(*.NOPIN ..... ) ( optional )
.
. netlist information in CDL format (may be empty).
.
.ENDS
```

Example 2

```
*****
**          SPICE LIBRARY FOR GRAYCODE          ***
*****

*.GLOBAL VDD,VSS
*.PIN VDD VSS
.SUBCKT NA2 Z A B
* 2 INPUT NAND
MP1 Z A VDD VDD P
MP2 Z B VDD VDD P
MN1 Z A 5 VSS N
MN2 5 B VSS VSS N
.ENDS NA2
*
.SUBCKT NA3 Z A B C
* 3 INPUT NAND
.SWAP A B C
MP1 Z A VDD VDD P
MP2 Z B VDD VDD P
MP3 Z C VDD VDD P
MN1 Z A 6 VSS N
MN2 6 B 7 VSS N
MN3 7 C VSS VSS N
.ENDS NA3
*
.SUBCKT NA4 Z A B C D
* 4 INPUT NAND
MP1 Z A VDD VDD P
MP2 Z B VDD VDD P
MP3 Z C VDD VDD P
MP4 Z D VDD VDD P
MN1 Z A 7 VSS N
MN2 7 B 8 VSS N
MN3 8 C 9 VSS N
MN4 9 D VSS VSS N
.ENDS NA4
```

Dracula Reference

Compiling Network Descriptions (LOGLVS)

```
*
.SUBCKT NOR Z A B
* 2 INPUT NOR
MP1 5 B VDD VDD P
MP2 Z A 5 VDD P
MN1 Z A VSS VSS N
MN2 Z B VSS VSS N
.ENDS NOR
*

.SUBCKT INV Z A
*INVERTER
MP1 Z A VDD VDD P
MN1 Z A VSS VSS N
.ENDS INV
*

.SUBCKT JKFF1 Q QN CL J K S R
MP1 6 CL VDD VDD P
MN2 6 CL VSS VSS N
MP3 5 6 VDD VDD P
.
.
.
.
.
MN43 42 J VSS VSS N
MN44 44 39 45 VSS N
MN45 45 QN VSS VSS N
MN46 45 K VSS VSS N
.ENDS JKFF1
*
```

Hcell file (hcell.tab)

```
JKFF JKFF1
INV INV
ND2 NA2
ND3 NA3
ND4 NA4
NO2 NOR
```

LOGLVS Examples

This section shows an example of a dual CMOS JK flip-flop. The following figure shows the schematic diagram of the JK flip-flop at the transistor level. The figures on the following pages show the same network redrawn at the logic level and transistor level implementation of cells. The network is composed of two identical flip-flops with only power and ground in common.

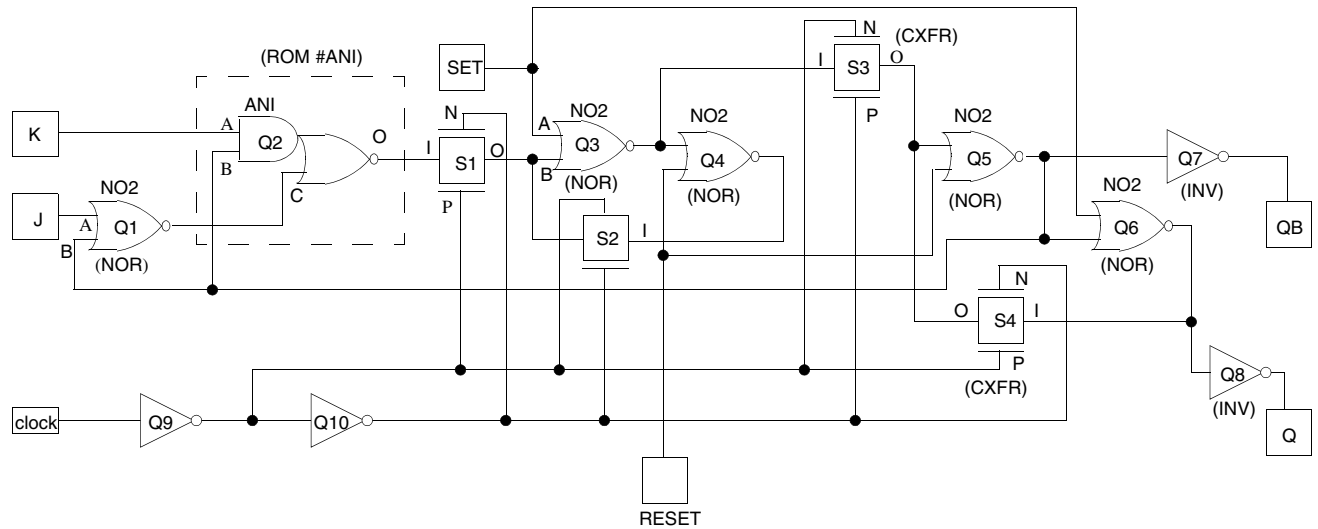
Each of the netlist examples in the sections on the SPICE formats describe the schematic in this example. .

Each cell name is listed at the top of the cell placement. The I/O pin names are also listed.

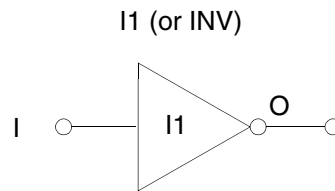
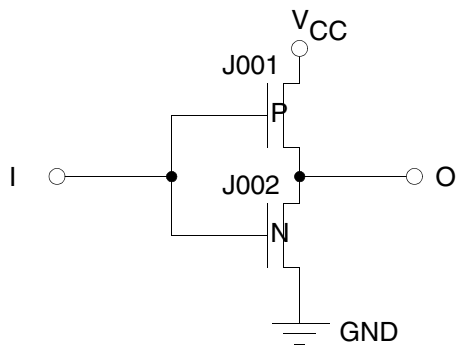
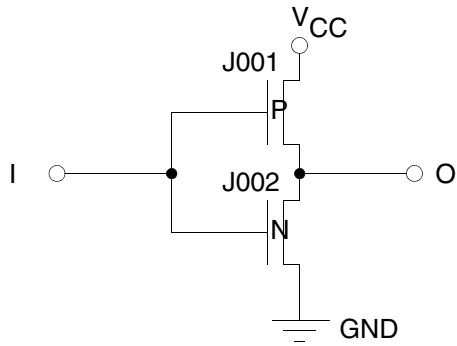
Dracula Reference

Compiling Network Descriptions (LOGLVS)

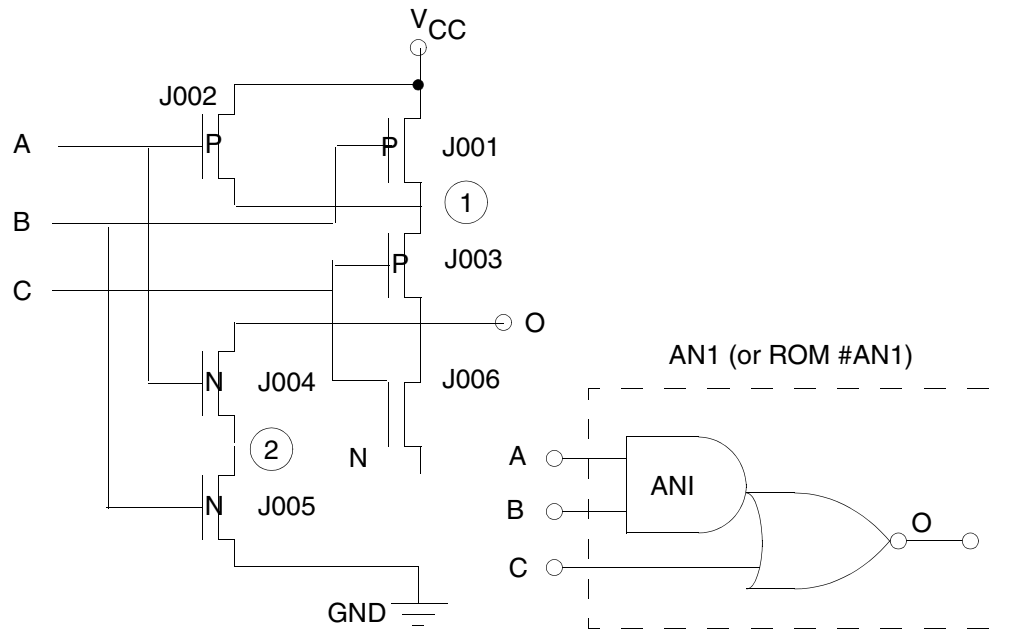
JK Flip-flop Logic Level



Transistor-Level Implementation



Transistor-Level Implementation (continued)



SPICE

In this example, the SPICE netlist is in the SPIC.CIR file. The cell name JKFF4627 in the CON command indicates that this cell will be converted.

```
DRACULA NETWORK COMPILER (LOGLVS) PROGRAM BEGINS...
ENTER COMMAND
:CIR    SPIC.CIR
READING SPIC.CIR
FILE    SPIC.CIR    INPUT
ENTER COMMAND
:CON    JKFF4627
TRANSISTOR-LEVEL FILE:  LVS LOGIC.DAT CREATED
NUMBER OF TRANSISTORS   (OR ELEMENTS) :92
ENTER COMMAND
:EXIT
```

Comparing Layouts and Schematics (LVS)

The focus of Chapter 6 is on LVS for Dracula. The following topics are covered:

- [“LVS Overview”](#) on page 275
- [“LVL Overview”](#) on page 281
- [“SVS Overview”](#) on page 285
- [“LVS Report Format”](#) on page 287
- [“Discrepancy Report Conventions”](#) on page 293
- [“LVS Error Types”](#) on page 295
- [“Device Recognition in LVS”](#) on page 306
- [“Isolating and Debugging Errors”](#) on page 324
- [“Interpreting an LVS Discrepancy”](#) on page 325
- [“Logical Equivalence in MOS Digital Circuits”](#) on page 327
- [“Calculations on Parallel MOS Devices for LVS”](#) on page 328
- [“Bipolar LVS Example”](#) on page 333
- [“Hierarchical LVS”](#) on page 337
- [“Inherited Connections for LVS in Dracula Verification”](#) on page 351

LVS Overview

Layout Versus Schematic (LVS) is the Dracula verification tool that confirms that your layout matches your original schematic. LVS compares the connectivity of the layout and schematic at the transistor level and reports any discrepancies. Dracula automatically generates the layout network using input data from major CAD graphics systems. LOGLVS, the Dracula

network compiler, prepares a schematic description for LVS in the form of a SPICE or logic simulation netlist.

LVS converts each network into a circuit model. Starting from the inputs and outputs of the circuit, LVS traces the two circuit models. Dracula uses heuristics to trace the circuit one step at a time. LVS traces I/O's first, then traces the paths that require the least backtracking. When LVS detects a match in the trace, Dracula assigns the matched devices or nodes a matched status. When LVS detects a mismatch, it stops tracing that path.

LVS considers the two networks consistent if it can assign all devices and nodes a matched status and there are no discrepancy points in the traced paths. When Dracula detects discrepancies, it presents them in both the output listings and in graphic form.

Besides comparing two networks, LVS also compares device subtypes (NMOS versus PMOS in CMOS circuits) and some device parameters, such as channel width and length in MOS circuits.

The LVS tool includes sections for checking

- Layout versus layout
- Schematic versus schematic
- Hierarchical layout versus schematic

Layout Versus Layout

Layout versus Layout (LVL) is the portion of LVS that compares two similar mask layout databases at the device or gate level and reports discrepancies in the interconnections and in certain device parameters. LVL requires the LVL-PART command in the Description block. For more information about LVL, refer to the [LVL Overview](#) section in this chapter.

Schematic Versus Schematic

Schematic versus Schematic (SVS) is the portion of LVS that compares two schematics. SVS requires the following commands in the Description block:

- HCELL-COLUMN-1
- SVS-LAYOUT
- SVS-SCHEMATIC

For more information about SVS, refer to the [SVS Overview](#) section in this chapter.

Hierarchical Layout Versus Schematic

Hierarchical Layout Versus Schematic (HLVS) works like LVS, except that it operates on a two-level hierarchy. For more information about HLVS, refer to the [Hierarchical LVS](#) section in this chapter.

Initial Correspondence Node Pairs

You can provide a set of initial correspondence nodes as starting points for the LVS trace operation. When a node in the layout database and a qualified node in the schematics have identical, unique labels, they become an initial correspondence node pair. A qualified schematic node is either a power node, ground node, top-level input node, top-level output node, or, depending on the schematic netlist language being used, an internal node. For details about assigning initial correspondence node pairs for the schematic netlist, refer to the "Compiling Network Descriptions (LOGLVS)" section in Chapter 5.

LVS chooses all power nodes, ground nodes, clock nodes, and at least one other node as minimum initial correspondence nodes. Because the program uses these initial correspondence pairs for trace operations, the more initial correspondence node pairs you provide, the more efficient the trace can be. However, LVS never questions the match of any correspondence node pairs, and if there are any errors in the initial correspondence node pairs, the program cannot check them, and the trace operation can be misled. Because manual texting on the layout database is an error-prone process, you need to provide the minimum number sufficient to do an effective check. As a guideline, assign all pads as initial correspondence node pairs. Additionally, you need to include important signal nodes, nodes with many device connections, or nodes of highly parallel circuits.

If Dracula does not find initial correspondence points, it turns on an automatic matching capability. For more information about automatic matching, refer to the [ARRAY-ENABLED](#) section in Chapter 11.

Trace and Discrepancy Points

Starting from the initial correspondence node pairs, LVS traces the extracted layout network and the schematic network one step at a time. Initially, LVS matches *all* correspondence node pairs. Whenever LVS finds a unique match between layout and schematic, it defines the objects as matched nodes or devices.

LVS ends the trace operation when it matches all nodes and devices or when discrepancies have stopped tracing all paths. This means that discrepancy points are the focus points for error isolation and error interpretation, instead of unmatched nodes or devices. To present the discrepancies, LVS reports matched or unmatched nodes and devices related to particular

discrepancies. The number of unmatched devices or nodes are not necessarily the same as the number of discrepancy points.

The sequence of the trace is dynamically determined in the program. Because error interpretations based on these assumptions might be incorrect, do not assume any trace sequences.

Using LVS

This section describes the following functional flow for using LVS:

1. Run DRC and ERC and correct all errors.
2. Compile the schematic netlist using LOGLVS.
3. Check the layout text.
4. Create and compile a rules file.
5. Run LVS.
6. View the discrepancy report.

Running DRC and ERC

You can run multiple verification operations such as DRC, ERC, LVS, and LPE at the same time. However, it is usually better to run operations one at a time, so you can correct only one type of error at a time. If you run ERC on the layout database first, you can correct the errors detected and reduce the run time and number of LVS discrepancies you must correct. When you run an ERC check, keep the following in mind:

- Check for illegal devices such as MOS diffusion connections to more than two electrical nodes or no electrical node.
- Using the MULTILAB command, check for shorted circuits between pads and major signals, such as VCC and VSS.
- Using the SAMELAB and ELCOUNT commands, check for open circuits associated with pads or major signals.
- Check text on the layout. Use composite text only, not cell level text.
- Identify major nodes and attach text to them if necessary. (This applies to nodes that connect to more than 150 elements.)

Compiling the Schematic Netlist

See the [Preparing Netlists](#) section in Chapter 5 of this manual for a detailed discussion of how to prepare and compile the schematic netlist.

Checking the Layout Text

You can provide a set of initial correspondence nodes as starting points for the tracing operations or use the AUTOMATCH command for internal correspondence point matching. Supply initial correspondence nodes by placing text on the layout database or by supplying text using an EDTEXT file. For detailed information about texting the layout, refer to [Chapter 12, “Input-Layer Block Commands.”](#)

When texting the layout, keep in mind the following guidelines:

- Text all input and output pads.
- Text power and ground nodes.
- Text important internal nodes.
- Make sure that the text strings on the database are the same as the node names in the netlist.
- Do not use too many texts per node.
- Check that the text origin is inside or at the edge of the geometry.

Creating and Compiling an LVS Rules File

If you have already run ERC, you can add LVS commands to your existing rules file. You can find a full description of each command in the “Description Block Commands” and “Operation Block Commands” chapters.

The following LVS commands are required:

SCHEMATIC

Calls the schematic netlist file created by LOGLVS. Enter this command in the Description block.

LVSCHK

Extracts the layout database for a consistency check between layout and schematic. Enter this command in the Operation block.

The following Description Block commands are optional:

Dracula Reference

Comparing Layouts and Schematics (LVS)

CPOINT-FILE

Inputs a file containing initial correspondence points for LVS matching. Lets you input initial correspondence points when node text names are different on the layout and schematic netlists. Lets you supply low-level net names as initial correspondence points.

FILTER-OPT

Sets filter options for both the layout and schematic.

FILTER-LAY-OPT

Sets filter options for the layout.

FILTER-SCH-OPT

Sets filter options for the schematic.

LVS-OPTION

Specifies LVS report options, letting you select the LVS error types you want listed.

LVS-RPT-ONLY

Generates a new LVS report from a previous LVS run.

SMASH-CAP-TYP

Lets you smash parallel capacitors even if they are of different types.

REMOVE-SCH-OPT

Removes FILTER stage from the schematic side after SCHMIN.

After you create the LVS rules file, use PDRacula to compile it. When the file is free of errors, you can run an LVS check.

Running LVS

Submit the LVS rules file using the *jxsub.com* command.

Viewing the LVS Report

Dracula creates a file containing all discrepancies found during the LVS comparison. The file name is *printfile.lvs*, where *printfile* is the name assigned by the PRINTFILE command in the Description block. Refer to later sections of this chapter for descriptions of the error messages and discrepancies.

LVL Overview

This section presents an overview of Layout Versus Layout (LVL) and rules files to demonstrate the verification of two similar IC layout databases. The output is in both graphic and listing formats.

LVL is the portion of LVS that compares one IC layout to a similar IC layout on the device/gate level (not on geometric polygon data) and reports discrepancies in the interconnections. (To compare polygon data from different layout databases, use the EXPORT and IMPORT commands with the logical NOT or XOR commands.) The system automatically generates the layout networks when you use the SCHEMATIC and GRAPHICS options for the LVL-PART command. LVL accepts layout databases from major CAD graphics systems.

In addition to the two layout networks (*6NET* files), the system produces a list of corresponding nodes for power, ground, clocks, inputs, and outputs (*6PAD* files) through the texting facility on both layouts. The texting on both layout databases must be the same.

LVL converts each network into internal model format (*6NET* files). Starting from the inputs and outputs of the layout circuit, LVL traces the two circuit models. Because the trace algorithm moves one step at a time, LVL uses heuristics. LVL traces all I/O's of the layout circuit and then traces the paths that require less backtracking. When the trace detects a match, LVL assigns a matched status to the devices or nodes. When the trace detects a mismatch, LVL stops tracing the path for that particular I/O.

If LVL assigns all devices and all nodes a matched status and there are no discrepancy points in the traced paths, the two layout networks are consistent. If LVL detects discrepancies, the system presents all discrepancy points in the *printfile.lvl* output listing.

In addition, LVL compares device subtypes and certain device parameters, such as the channel width and length of MOS devices.

Preparing for LVL

Before running LVL, perform the following steps:

1. Run ERC checks on both layout databases as follows:

- ☐ Check for illegal devices, such as connected MOS gate, source, and drain.

```
NDCOUNT MOS ALL EQ 1 OUTPUT ONENET 40
```
- ☐ To check for shorted circuits between pads and major signals, such as VCC and VSS, use the MULTILAB command.
- ☐ To check for open circuits, pads, and major signals, use the SAMELAB command.

2. Create a rules file for the reference layout database. This LVL rules file is the ERC rules file with the following two additions:

- ☐ Specify the LVL-PART command with the SCHEMATIC option in the Description block (such as SCHEMATIC in the LVS check).
- ☐ Specify the LVSCCHK command at the end of the Operation block to start the LVL check of the reference layout database.

3. Create a second rules file for the second layout database. Modify the ERC rules file as follows:

- ☐ Specify LVL-PART=GRAPHICS in the Description block.
- ☐ Specify the LVSCCHK command at the end of the Operation block to start the LVL check of the reference layout database against the layout database.

The LVL run provides a *printfile.lvl* discrepancy listing and *lvplot* graphic output database in the file specified by the OUTDISK command.

The LVL discrepancy listing is similar to the LVS discrepancy listing except that the program lists the x,y locations for the device in the reference layout database. LVL also lists device subtypes and substrate connections. If you specify the proper LVSCCHK options, LVL checks and lists MOS channel width and lengths.

4. To submit and execute the two LVL rules files, use the /NEXT command in the preprocessor.

The LVL-PART command alerts the preprocessor that LVL is performing an LVL discrepancy check. Because the LVL job is similar to an LVS job, use /NEXT to combine the two LVL rules files.

5. To control the formation of pseudo devices in the MOS process and to check MOS device sizes, use LVSCCHK options.

Dracula Reference

Comparing Layouts and Schematics (LVS)

If you do not specify *lvschk* options exactly the same in both rules files, the consistency check is different.

Rules Files

The following are examples of rules files to submit LVL runs. You can enter command names in either lowercase or uppercase letters.

Rules File for the Schematic

```
; **** "LVL1.COM" rules file for the Reference Layout ****
; ****                      Device (LVL-PART) ****
; **** NOTE: To run LVL, the consistency check must be ****
; **** between two (2) similar (process technology) ****
; **** databases. ****

;
;
*DESCRIPTION

.
PRIMARY = topl
MODE = exec now
SYSTEM = gds2
SCALE = .001 mic
RESOLUTION = .25 mic
PROGRAM-DIR = drac:
OUTDISK = lvllout
.
.
INDISK = layout1.in ;database for the reference device (SCHEMATIC)
PRINTFILE = lvllprt ;print file for the reference device run
LVL-PART = schematic ;specifies as LVL reference device
.
.
*END
*INPUT-LAYER
pwell = 1
ndiff = 2
pdiff = 3
poly = 4
contact = 5
metal = 6 TEXT = 26
pad = 7
SUBSTRATE = bulk 99
CONNECT-LAYER = nsub pwell psd nsd poly metal
*END
*OPERATION
AND poly ndiff ngate ; n-channel devices
NOT ndiff ngate nsd ; N source/drain diffusion
AND poly pdiff pgate ; p-channel device
NOT pdiff pgate psd ; P source/drain diffusion
NOT bulk pwell nsub ; N substrate
AND nsd nsub nsub ; pseudo contact for N substrate
```

Dracula Reference

Comparing Layouts and Schematics (LVS)

```
AND psd pwell ptwell ; pseudo contact for P well
;
CONNECT metal poly BY contact ; connecting the conductor
                                ; layers by contact
CONNECT metal nsd BY contact
CONNECT metal psd BY contact
CONNECT psd pwell BY ptwell
CONNECT nsub nsd BY nsub
;
ELEMENT MOS[n] ngate poly nsd pwell ; n-channel ELEMENT
ELEMENT MOS[p] pgate poly psd nsub ; p-channel ELEMENT
ELEMENT PAD pad metal ; pad ELEMENT
;
MULTILAB OUTPUT shorts 40 ; run the "shorts" test
SAMELAB OUTPUT opens 40 ; run the "opens" test
LVSCHK ; specify the LVL check
*END
```

Rules File for Graphics

```
; **** "LVL2.COM" rules file for the Layout Device ****
; **** to be checked (LVL-PART=GRAPHICS) ****
; **** NOTE: To run LVL, the consistency check must be ***
; **** between two similar (process technology) databases*
;
*DESCRIPTION
.
PRIMARY = top2
MODE = exec now
SYSTEM = gds2
SCALE = .001 mic
RESOLUTION = .25 mic
PROGRAM-DIR = drac:
OUTDISK = lvl2out
.
.
INDISK = layout2.in ; database for the device to be checked
PRINTFILE = lvl2prt ; print file for the device to be checked
LVL-PART = graphics ; specifies as LVL device to be checked
.
.
*END
*INPUT-LAYER
pwell = 21
ndiff = 7
pdiff = 22
poly = 24
contact = 5
metal = 10 TEXT = 13
pad = 63
SUBSTRATE = bulk 99
CONNECT-LAYER = nsub pwell psd nsd poly metal
*END
*OPERATION
AND poly ndiff ngate ; n-channel devices
NOT ndiff ngate nsd ; N source/drain diffusion
```

Dracula Reference

Comparing Layouts and Schematics (LVS)

```
AND poly pdiff pgate ; p-channel device
NOT pdiff pgate psd ; P source/drain diffusion
;
NOT bulk pwell nsub ; N substrate
AND nsd nsub nsub ; pseudo contact for N substrate
AND psd pwell ptwell ; pseudo contact for P well
;
CONNECT metal poly BY contact ; connecting the conductor
                                ; layers by contact

CONNECT metal nsd BY contact
CONNECT metal psd BY contact
CONNECT psd pwell BY ptwell
CONNECT nsd nsub BY nsub
;
ELEMENT MOS[n] ngate poly nsd pwell ; n-channel ELEMENT
ELEMENT MOS[p] pgate poly psd nsub ; p-channel ELEMENT
ELEMENT PAD pad metal ; pad ELEMENT
;
MULTILAB OUTPUT shorts 40 ; run the "shorts" test
SAMELAB OUTPUT opens 40 ; run the "opens" test
;
LVSCHK ; specify the LVL check
;
*END
```

Running LVL

To start an LVL run on a UNIX workstation, type the following command:

```
% PDRACULA
: /GET LVL1.COM
: /NEXT
: /GET LVL2.COM
: /FIN
```

LVL runs *lv1.com*, then runs *lv2.com*. The LVL discrepancy report file is *lv2prt.lvl*.

SVS Overview

Schematic Versus Schematic (SVS) compares two schematics. The two schematics must be in different directories. SVS requires the following commands in the Description block:

- SVS-LAYOUT specifies the layout data file generated by LOGLVS.
- SVS-SCHEMATIC specifies the schematic data file generated by LOGLVS.
- HCELL-COLUMN-1 specifies whether the first column in the Hcell table lists layout or schematic cells. This command is required for hierarchical Dracula only.

For more information about these commands, refer to [Chapter 11, “Description Block Commands.”](#)

Running SVS in Flat Mode

The following is an example of a rules file for flat mode SVS:

```
*DESCRIPTION
  PRINTFILE = lvsprt
  PROGRAM-DIR = /DRACULA/
  SVS-SCHEMATIC = LVSLOGIC.DAT
  SVS-LAYOUT = ../LVSLOGIC.DAT
*END
;
*OPERATION
  LVSCHK
*END
```

Running SVS in Cell or Composite Mode

Here is an example of a rules file for cell or composite mode SVS:

```
*DESCRIPTION
  PRINTFILE = lvsprt
  SVS-SCHEMATIC = LVSLOGIC.DAT
  SVS-LAYOUT = ../LVSLOGIC.DAT
  program-dir = /DRACULA/
  CHECK-MODE=comp
  HCELL-FILE=../cell.tab
*END
;
*OPERATION
  LVSCHK
*END
```

Running SVS in Hierarchical Mode

To run SVS in hierarchical mode, you must include the number 1 at the end of the LOGLVS CELL command. The number 1 identifies the first cell name in the Hcell file as a schematic. This example specifies that the first cell name listed in the Hcell file HCELL.TAB is a schematic.

```
CEL HCELL.TAB 1
```

If you do not include the HCELL-COLUMN-1 command, the default setting is *LAYOUT*. To switch the role of both circuits after running LOGLVS, use the HCELL-COLUMN-1 command. For example, on the first LOGLVS run, specify the command as follows:

```
SVS-SCHEMATIC = LVSLOGIC
SVS-LAYOUT = ../LVSLOGIC.DAT
```

Dracula Reference

Comparing Layouts and Schematics (LVS)

One the second run, switch the circuits without rerunning LOGLVS. Modify the rules this way:

```
SVS-SCHEMATIC = ../LVSLOGIC.DAT
SVS-LAYOUT = LVSLOGIC.DAT
HCELL-COLUMN-1 = SCH
```

LVS Report Format

The following is a brief overview of a sample LVS report. An LVS report contains information pertaining to an LVS job run. Depending on whether you run flat, cell, or composite mode, some of the major sections in the LVS reports can be the following:

- **Header:** The header contains the version identification of Dracula, the execution time of the job, and the name of the primary cell. This section is created by PDRACULA.
- **EXTERM:** If you include the EXTERM-NOWARN=YES command in the Description block, only the “unconnected pins in cell” messages are not displayed in the LVS report. This section is for composite mode LVS.
- **XDEVIC summary:** Reports illegal hierarchical devices.
- **LVSNET summary:** LVSNET prints warning messages in this section.
- **FILTER (layout) summary:** FILTER prints a summary of statistics for the layout side in this section. If you specify special filter options, they are listed in this section.
- **REDUCE (layout) summary:** This summary lists the options used in REDUCE. Depending on the FILTER-OPT command or the LVSCCHK or LPECHK[F or G option] command, the schematic REDUCE or the layout REDUCE may be different. The device reduction statistics follow the options listing.

The following types are described in the “Device Recognition” section in this chapter:

```
PDW AND SDWI SMID NOR
PUP OR PDWI PMID NAND
SDW AOI SUPI MOS
SUP OAI PUPI INV
```

MOSCAP represents the number of transistors reduced to MOS capacitors; that is, MOS transistors with source and drain tied together to either power or ground.

Other device types listed with statistics are the following:

```
BJT CELL DIODE LDD
CAP UND RES BOX
```

Note: The reference to “CELL” means the number of Hcells (placements or instances) that are placed in the network. “LDD” represents the lightly doped drain. “UND”

Dracula Reference

Comparing Layouts and Schematics (LVS)

represents the undefined device. For more information about “UND,” refer to the [Device Recognition in LVS](#) section in this chapter.

- FILTER (schematics) summary

FILTER prints a summary of statistics for the schematics side in this section. If you specify special filter options, they are listed in this section.

- REDUCE (schematics) summary

The statistics for the reduction of the schematics side are listed in this section. The meaning of each type of device is the same as the REDUCE (layout) section.

- Main LVS report body

This section is the core of the LVS report. It contains the date and time of execution of the LVS module and LVS options such as PRINTLINE, WPERCENT, and LPERCENT, the correspondence node pairs, LVS device match summary, the discrepancy points listing, the discrepancy points summary, the device matching summary by type, and the unmatched devices listing.

- Repeat summary

This section is at the end of the LVS report. It repeats the summary sections in the main report body to let you quickly browse LVS results.

Dracula Reference

Comparing Layouts and Schematics (LVS)

Sample LVS Report

The following sample report shows sections that you might have in your LVS report.

```
*****
*/N* DRACULA (REV. 4.4.X / 4.4 / GENDATE: 22-AUG-95/10)
    *** (Copyright 1995, Cadence) ***
    */N* EXEC TIME =22:55:37      DATE =29-AUG-95
*****
INDISK PRIMARY CELL :TEST
***** LVSNET SUMMARY REPORT *****
WEFFECT VALUE= 0.0000000
***** EXTERM SUMMARY REPORT *****
*CRCTEX* WARNING *--UNCONNECTED PINS IN CELL =      2  PLACE. =      31
PIN NAME      LAYER NAME      X      Y
Z2BAR         MET1           1344.00  -2402.40

***** FILTER (LAYOUT) SUMMARY REPORT *****
SET FILTERING OPTIONS TO BCDEFGHIJKO
ELIMINATED 509 OUT OF 1694 DEVICES FROM NETWORK
ELIMINATED 11 OUT OF 1185 DEVICES FROM NETWORK
***** REDUCE (LAYOUT) SUMMARY REPORT *****

***** STATISTICS BEFORE REDUCE *****
MOS   BJT   RES   DIODE   CAP   UND   BOX   CELL   LDD
1172   0     0     0       0     2     0     0     0
OPTION TO SMASH PARALLEL DEVICES IS -- ON
OPTION TO CONSTRUCT MOS PARALLEL/SERIES STRUCTURES IS -- ON
OPTION TO SMASH PSEUDO PARALLEL DEVICES IS -- OFF
OPTION TO FORM CMOS GATES IS -- OFF
OPTION TO FORM NAND/NOR GATES IS -- OFF
***** STATISTICS AFTER REDUCE *****
MOS   BJT   RES   INV   DIODE   CAP   SDWI   PDWI   SUPI
113   0     0    102   0       0     2     13    13
PUPI  SDW   PDW   SUP   PUP     AND   OR     AOI    NAND
2     58    23    56    58     0     0     0     0
NOR   OAI   UND   BOX   CELL   LDD   SMID   PMID   MOSCAP
0     0     2     0     0     0     0     0     0

***** FILTER (SCHEMATIC) SUMMARY REPORT *****
SET FILTERING OPTIONS TO FGHIJKO
ELIMINATED 0 OUT OF 798 DEVICES FROM NETWORK
***** REDUCE (SCHEMATIC) SUMMARY REPORT *****
***** STATISTICS BEFORE REDUCE *****
MOS   BJT   RES   DIODE   CAP   UND   BOX   CELL   LDD
798   0     0     0     0     0     0     0     0
***** STATISTICS AFTER REDUCE *****
```

Dracula Reference

Comparing Layouts and Schematics (LVS)

```

MOS      BJT      RES      INV      DIODE    CAP      SDWI    PDWI    SUPI
107      0        0        102     0        0        2       13     13
PUPI     SDW      PDW      SUP      PUP      AND      OR       AOI     NAND
2        58      23      56      58      0        0       0      0
NOR      OAI      UND      BOX      CELL     LDD      SMID     PMID    MOSCAP
0        0        0        0      0        0        0       0      0

```

***** LVS REPORT *****

DATE : 29-AUG-95
TIME : 22:57:57

PRINTLINE = 1000
MOS DEVICE WIDTH NOT CHECKED
MOS DEVICE LENGTH NOT CHECKED

1 *****

***** CORRESPONDENCE NODE PAIRS *****

SCHEMATICS		LAYOUT		PAD TYPE	
VDD	111	VDD		1	P
VSS	196	VSS		18	G
P1	188	P1		121	I
P10	2	P10		14	I
P11	388	P11		8	I
P12	368	P12		9	I
P13	392	P13		10	I
P14	364	P14		33	I
P15	396	P15	61	I	
P16	372	P16	95	I	
P17	400	P17	125	I	
P18	376	P18	166	I	
P19	1	P19	215	I	
P2	183	P2	110	I	
P21	404	P21	294	I	
P22	380	P22	328	I	
P23	8	P23	337	I	
P24	4	P24	301	I	
P25	143	P25	336	I	
P26	138	P26	335	I	
P27	133	P27	334	I	
P28	128	P28	333	I	
P29	123	P29	332	I	
P3	178	P3	79	I	
P30	118	P30	331	I	
P31	113	P31	330	I	
P32	107	P32	329	I	
P33	9	P33	405	I	
P34	6	P34	258	I	
P35	10	P35	327	I	
P36	7	P36	305	I	
P37	5	P37	321	I	
P38	384	P38	293	I	
P39	3	P39	224	I	
P4	173	P4	2	I	
P5	168	P5	3	I	
P6	163	P6	4	I	
P7	158	P7	5	I	
P8	153	P8	6	I	
P9	148	P9	7	I	

Dracula Reference

Comparing Layouts and Schematics (LVS)

NUMBER OF LAY. PADS READ = 40 DISCARDED = 0
NUMBER OF SCH. PADS READ = 40 DISCARDED = 0

NUMBER OF VALID CORRESPONDENCE NODE PAIRS = 38

1 *****

***** LVS DEVICE MATCH SUMMARY *****

NUMBER OF UN-MATCHED SCHEMATICS DEVICES = 0

NUMBER OF UN-MATCHED LAYOUT DEVICES = 8

NUMBER OF MATCHED SCHEMATICS DEVICES = 434

NUMBER OF MATCHED LAYOUT DEVICES = 434

1 *****

***** DISCREPANCY POINTS LISTING *****

***** DISCREPANCY 1*****

--- NODE C59

---WITH EXTRA LAY DEVICES-----

OCCURRENCE NAME C72.4

DEV926 PDW : DEV1805 PDW X=794 Y=1206
C72.4, C59, I509 C72.4, C59, I509

DEV927 SUP : DEV1806 SUP X=852 Y=1222
C72.4, C59, I509 C72.4, I509, C59

OCCURRENCE NAME C59

DEV928 SUP : DEV1794 SUP X=2252 Y=2342
C59, P26, P26 C59, P26, P26

DEV773 MOS N NOR M3 : DEV1543 MOS N X=2194 Y=2342
P26, C59, VSS P26, VSS, C59

OCCURRENCE NAME C64

DEV1055 PDWI : DEV1938 PDWI X=2594 Y=2182
C64, C59, C60 C64, C59, C60

DEV1057 SUPI : DEV1940 SUPI X=2652 Y=2182
C64, C59, C60 C64, C59, C60

OCCURRENCE NAME C72.7

DEV1095 SDW : DEV1786 SDW X=794 Y=1142
C72.7, C72.3, C59 C72.7, C59, C72.3

DEV1096 PUP : DEV1787 PUP X=852 Y=1126
C72.7, C72.3, C59 C72.7, C72.3, C59

OCCURRENCE NAME C64

DEV1130 SDW : DEV1967 SDW X=2594 Y=2262
C64-4, C59, C60 C64-4, C59, C60

DEV1131 PUP : DEV1968 PUP X=2652 Y=2246
C64-4, C60, C59 C64-4, C60, C59

***** UN-MATCHED ***** : ?DEV685 MOS P X=2452 Y=2278 C59, VDD, ?2192

***** UN-MATCHED ***** : ?DEV1523 MOS N X=2394 Y=2278 C59, VSS, ?2548

TOTAL 1 DISCREPANCY POINTS REPORTED

1 *****

***** DISCREPANCY POINTS SUMMARY *****

Dracula Reference

Comparing Layouts and Schematics (LVS)

1 MATCHED NODE TO EXTRA LAYOUT DEVICES

***** DEVICE MATCHING SUMMARY BY TYPE *****

TYPE	SUB-TYPE	TOTAL DEVICE SCH.	UN-MATCHED DEVICE LAY.	SCH.	LAY.
MOS	P	399	404	0	5
MOS	N	366	367	0	1

1 *****

***** UN-MATCHED LAYOUT DEVICES *****

***** (LIST UP TO 100) *****

: ?DEV491 MOS P X=2934 Y=1806
VSS, VDD, ?231
: ?DEV492 MOS P X=2950 Y=1822
VSS, ?231, ?234
: ?DEV493 MOS P X=2966 Y=1838
VSS, ?234, ?243
: ?DEV504 MOS P X=2982 Y=1854
VSS, VDD, ?243
: ?DEV685 MOS P X=2452 Y=2278
C59, VDD, ?2192
: ?DEV1523 MOS N X=2394 Y=2278
C59, VSS, ?2548

1 *****

***** LVS SUMMARY (REPEATED) *****

***** LVS DEVICE MATCH SUMMARY *****

NUMBER OF UN-MATCHED SCHEMATICS DEVICES = 0
NUMBER OF UN-MATCHED LAYOUT DEVICES = 8
NUMBER OF MATCHED SCHEMATICS DEVICES = 434
NUMBER OF MATCHED LAYOUT DEVICES = 434

***** DISCREPANCY POINTS SUMMARY *****

1 MATCHED NODE TO EXTRA LAYOUT DEVICES

Dracula Reference
Comparing Layouts and Schematics (LVS)

```
***** DEVICE MATCHING SUMMARY BY TYPE *****
*****
```

TYPE	SUB-TYPE	TOTAL DEVICE	UN-MATCHED DEVICE
		SCH. LAY.	SCH. LAY.
MOS	P	399 404	0 5
MOS	N	366 367	0 1

Discrepancy Report Conventions

LVS follows these conventions when listing the discrepancy report.

Number of matched and unmatched devices

LVS highlights the number of matched and unmatched devices. The total device count might not be the same as the total number of extracted devices because LVS can combine multiple devices into pseudo-devices and smash parallel devices into a single device.

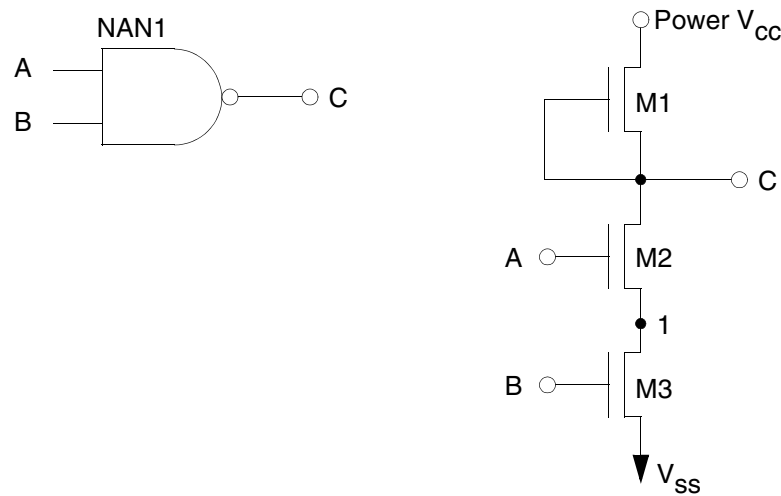
Schematic section against layout section

LVS divides an 80-column print line for a discrepancy point into two sections. The first 40 columns are the schematic section and the remaining columns are the layout section.

Schematic Section : Layout Section

Matched node identification

Use the schematic node name for both the schematic and layout nodes. If the schematic node is the local node, LVS uses the hierarchical node name from macro expansion to guarantee its uniqueness. For example, the following two-input NAND gate has five nodes with node 1 as the local node.



Dracula Reference

Comparing Layouts and Schematics (LVS)

These six node names are *VCC*, *VSS*, *A*, *B*, *C*, and *NAN1-1*, where *NAN1* is the occurrence name of the NAND gate and *NAN1-1* is the node name for node 1 after macro expansion.

Unmatched schematic node identification

If LVS finds a node in the schematic that does not have a corresponding layout node, LVS prefixes the node with a question mark (?). For example, ?A, ?B, ?C.

Matched device identifications

LVS always lists matched devices side by side with a schematic device on the left side of the center colon (:) and a layout device on the right side of the colon. Both the schematic device and the layout device have a unique internally generated device number, followed by a device type and an optional device subtype. A schematic device can also have a logic gate type and a device name within that particular gate, if applicable. A layout device has x or y locations in a natural unit.

LVS then follows the list of nodes connected to the matched device. For example:

```
Occurrence Name      NAN1
DEV123  MOS  D  NAND  M1 :  DEV255  MOS  D  X=100  Y=50
C,C,VCC               C,C,VCC
```

Note that the schematic device name is local to the gate, so LVS lists the occurrence name whenever possible to uniquely identify the device on schematics. In the above example, *M1* is the device within gate *NAND* with the occurrence name *NAN1*.

Unmatched schematic device identifications

If LVS finds a schematic device that is not referenced in the layout, LVS prefixes the device number with a question mark (?) and the corresponding layout device is (*****UNMATCHED*****).

```
?DEV123  MOS  D  NAND  M1 :  *****UNMATCHED*****
```

Unmatched layout device identification

If LVS finds a layout device that is not referenced in the schematic, LVS prefixes the device number with a question mark (?) and the corresponding schematic device is (*****UNMATCHED*****).

```
*****UNMATCHED*****  ?DEV255  MOS  D  X=100 Y=50
```

Connected Node List Conventions

Device type	Terminals listed
MOS/LDD	Gate, drain, source
BJT	Collector, base, emitter
RES/CAP/DIODE	Terminal-1, terminal-2
SDW	Output node, list of input nodes
SDWI	Output node, list of input nodes
SUP	Output node, list of input nodes
SUPI	Output node, list of input nodes
PDW	Output node, list of input nodes
PDWI	Output node, list of input nodes
PUP	Output node, list of input nodes
PUPI	Output node, list of input nodes
SMID/PMID	Output1, output2, list of input nodes
CELL	Composite node 1 --pin name 1
INV	Output node, input node
AND	Output node, list of input nodes
OR	Output node, list of input nodes
NAND	Output node, list of input nodes
NOR	Output node, list of input nodes
AOI	Output node, list of input nodes
OAI	Output node, list of input nodes
BOX	Terminal-1, terminal-2, terminal-3, terminal-4

LVS Error Types

Discrepancy points are the focus points for all LVS error isolations. Depending on the LVS error type, one discrepancy point can involve a single node, single device, or a subcircuit (group of nodes and devices). LVS lists each discrepancy point separately, and it always

Dracula Reference

Comparing Layouts and Schematics (LVS)

corresponds to one LVS error. If a subcircuit is involved in a discrepancy point, not all nodes and devices are necessarily in error.

The Dracula program defines LVS errors for each discrepancy point based on its key device or node against the rest of the nodes and devices. Note that LVS errors do not necessarily correspond exactly to human perception of errors. For example, when two signal lines swap for a portion of the interconnects and are considered only one error, LVS might generate two discrepancy points, one for each signal.

LVS categorizes discrepancies into the following 15 error types:

Type	Error description	Plot	Listing
1	Matched node to no device	Node	X
2	Matched device to unmatched node	Dev., Node	X
3	Inconsistently matched device	Dev., Node	X
4	Matched node to extra layout devices (missing schematic)	Dev., Node	X
5	Matched node to extra schematic devices (missing layout)	Node	X
6	Matched node to unmatched layout and schematic devices	Dev., Node	X
7	Other unmatched layout devices	Dev.	---
8	Other unmatched schematic devices	---	X
9	Device subtype mismatch (N-type, Ptype, etc.)	Dev.	X
10	Device size (W, L) mismatch	Dev.	X
11	MOS reversibility error	Dev.	X
12	Device substrate connection mismatch	Dev.	X
13	Device power connection mismatch (Multi-power supply situation)	Dev.	X
14	Reduced layout parallel MOS devices	Dev.	---
15	Filtered out layout MOS devices	Dev.	---

LVS Error Type 1

Matched node to no device.

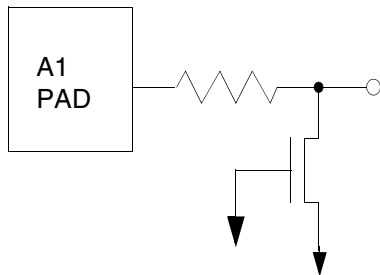
Key: matched node

Listed output: yes

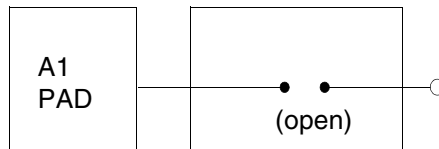
Graphic output: matched layout node

Initial correspondence nodes are not connected to devices. This is caused by misplaced layout text or input protection devices that are not properly defined.

Schematic



Layout



LVS Error Type 2

Matched device to unmatched node.

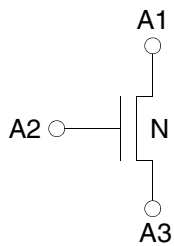
Key: matched device

Listed output: yes

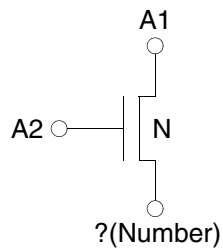
Graphic output: matched layout device, unmatched layout node

The device and some of its nodes are matched, but discrepancies are detected on other nodes.

Schematic



Layout



LVS Error Type 3

Inconsistently matched device.

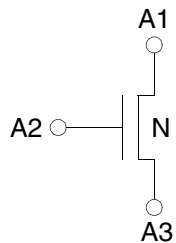
Key: matched device

Listed output: yes

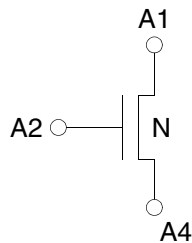
Graphic output: matched layout device, matched layout node

The device or nodes are matched locally first, and global discrepancies are detected later.

Schematic



Layout



LVS Error Type 4

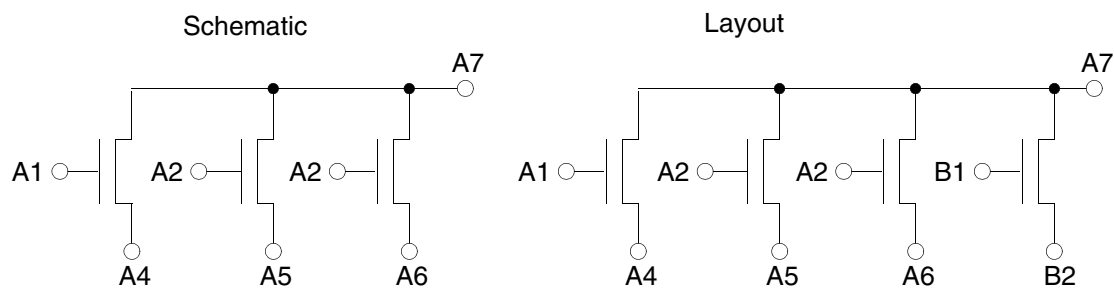
Matched node to extra layout devices (missing schematic devices).

Key: matched node

Listed output: yes

Graphic output: matched layout node, unmatched layout device

Extra layout devices are connected to the key matched node in addition to a number of matched devices.



LVS Error Type 5

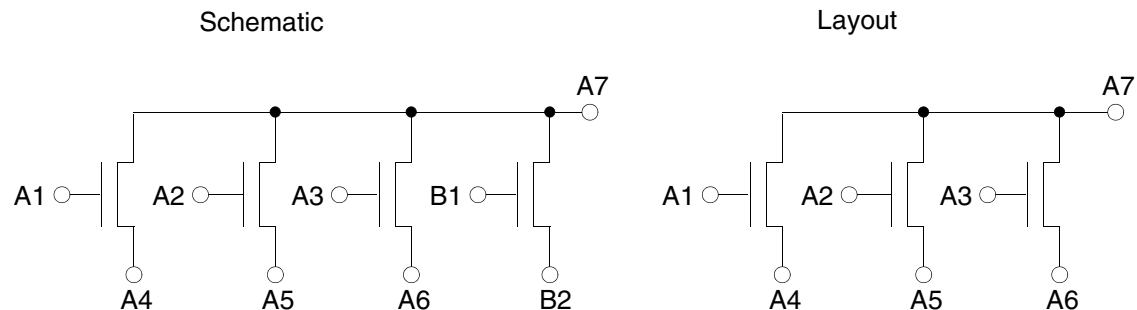
Matched node to extra schematic devices (missing layout devices).

Key: matched node

Listed output: yes

Graphic output: matched layout node

Extra schematic devices are connected to the key matched node in addition to a number of matched devices.



LVS Error Type 6

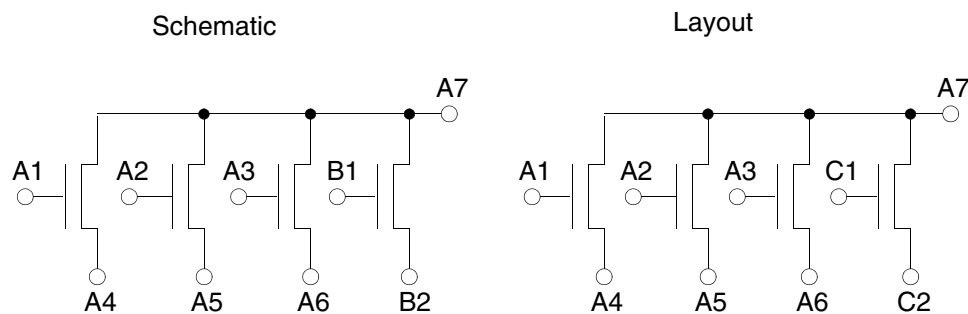
Matched node to unmatched layout and schematic devices.

Key: matched node

Listed output: yes

Graphic output: matched layout node, unmatched layout devices

Some of the schematic devices connected to the key matched node cannot be matched to the layout devices connected to the same node.



LVS Error Type 7

Other unmatched layout devices.

Key: unmatched layout devices

Listed output: yes

Graphic output: unmatched layout device

Unmatched layout devices are either totally separated from the rest of the circuit and cannot be reached from any initial correspondence node pairs or totally blocked by discrepancy points when trace operation stops.

LVS Error Type 8

Other unmatched schematic devices.

Key: unmatched schematic device

Listed output: yes

Graphic output: no

Dracula Reference

Comparing Layouts and Schematics (LVS)

Unmatched schematic devices are either totally separated from the rest of the circuit and cannot be reached from any initial correspondence node pairs or totally blocked by discrepancy points when trace operation stops. These devices are listed at the end of the discrepancy point listing.

LVS Error Type 9

Device subtype mismatch.

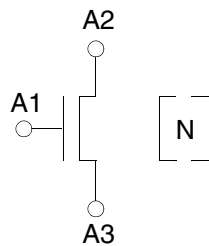
Key: matched device

Listed output: yes

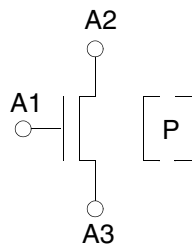
Graphic output: matched layout device

Two matched devices have different subtypes, which could be the implant types in the MOS circuit.

Schematic



Layout



LVS Error Type 10

Device size (W, L) mismatch.

Key: matched MOS device

Listed output: yes

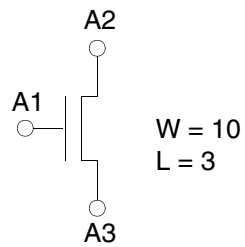
Graphic output: matched layout device

Dracula Reference

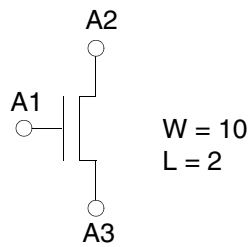
Comparing Layouts and Schematics (LVS)

Layout device size differs from the schematic device size by a specified percentage according to the LVCHK command options WPERCENT and LPERCENT.

Schematic



Layout



LVS Error Type 11

MOS reversibility error.

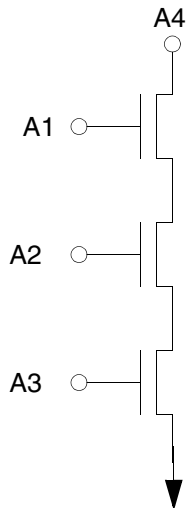
Key: matched MOS device

Listed output: yes

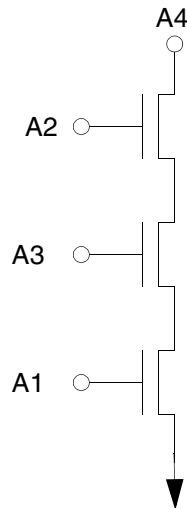
Graphic output: matched MOS devices

The physical locations of the layout MOS devices relative to outputs do not conform to the order specified in schematics.

Schematic



Layout



LVS Error Type 12

Device substrate connection mismatch.

Key: matched MOS device

Listed output: yes

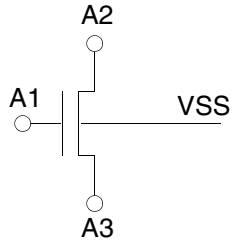
Graphic output: matched MOS device

Dracula Reference

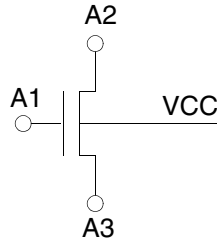
Comparing Layouts and Schematics (LVS)

MOS substrate connections are different between layout and schematic devices. This error usually applies to the CMOS circuit.

Schematic



Layout



LVS Error Type 13

Device power connection mismatch.

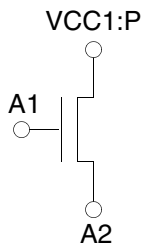
Key: matched device

Listed output: yes

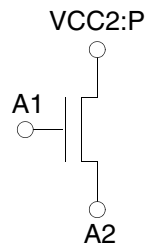
Graphic output: mismatched layout device

An incorrect power supply drives an output node in a multiple power supply situation.

Schematic



Layout



LVS Error Type 14

Reduced layout parallel MOS devices.

Key: no

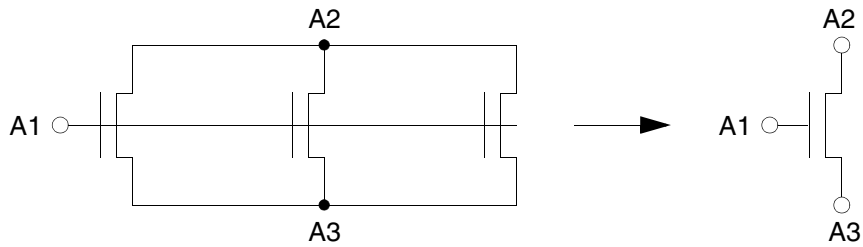
Listed output: no

Graphic output: reduced layout MOS devices

Dracula Reference

Comparing Layouts and Schematics (LVS)

When the LVSCCHK[K] option (keep parallel) is not turned on (as default for circuits), parallel MOS transistors are reduced into one MOS transistor. The remaining MOS transistors are not traced by LVS and are flagged as LVS error type 14. This condition should not be interpreted as an error.



LVS Error Type 15

Filtered out layout MOS devices.

Key: no

Listed output: no

Graphic output: filtered out MOS devices

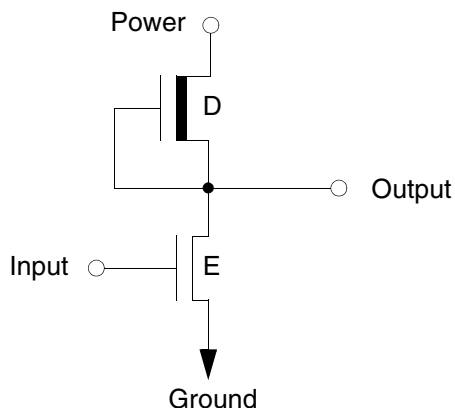
When the LVSCCHK[F] option (CMOS gate array filter) is turned on, unused CMOS devices are filtered out by LVS. Those devices can be flagged as LVS error type 15. Do not interpret this condition as an error.

Device Recognition in LVS

LVS recognizes two levels of devices. At the lowest level are all physical devices, such as MOS transistors, bipolar NPN transistors, bipolar PNP transistors, diodes, resistors, and capacitors. Second-level devices (pseudo gates and gates) are based on physical devices satisfying certain circuit structures. Typically, LVS uses a combination of both levels of devices in the trace operation. Therefore, the discrepancy points usually also include a combination of both levels of devices. LVS recognizes the logic gates and pseudo gates described in the following sections.

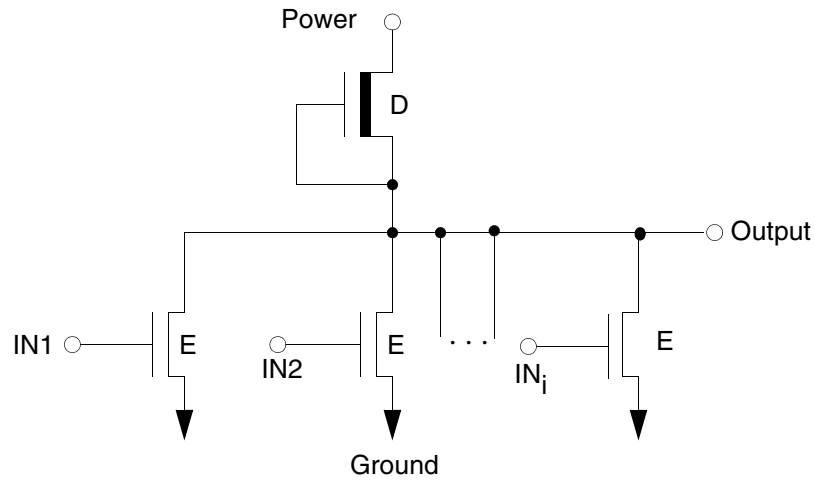
NMOS Inverter

INV Output, Input



NMOS NOR Gate

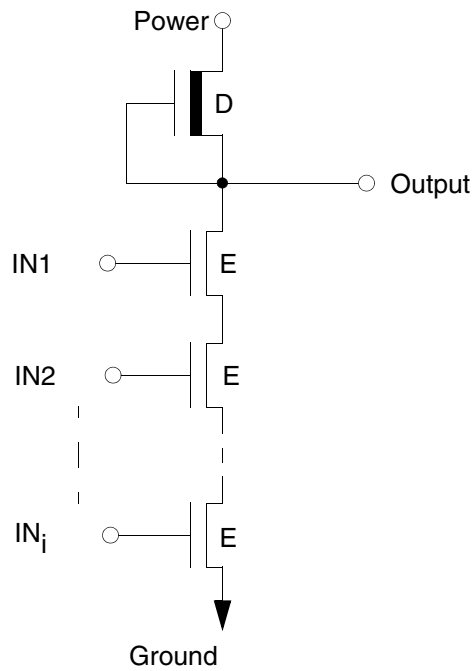
NOR Output, IN1, IN2,..., IN_i



Note: To create an NMOS OR gate, you must follow the NOR gate with an inverter.

NMOS NAND Gate

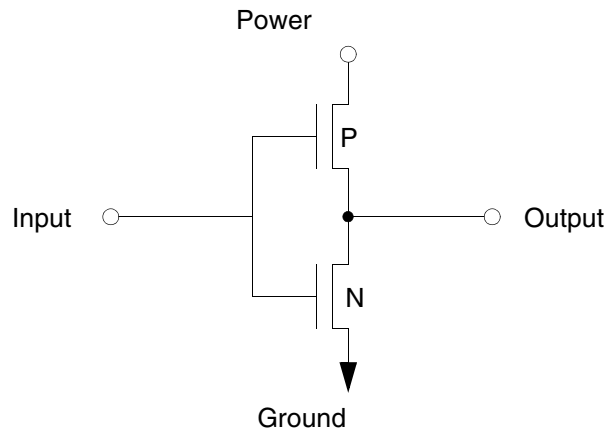
NAND Output, IN1, IN2,..., IN_i



Note: To create an NMOS AND gate, you must follow the NAND gate with an inverter.

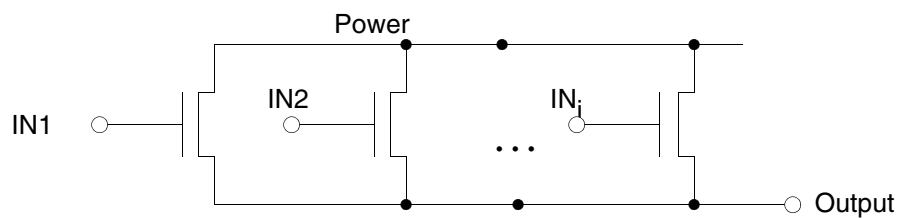
CMOS Inverter

Inverter Output, Input



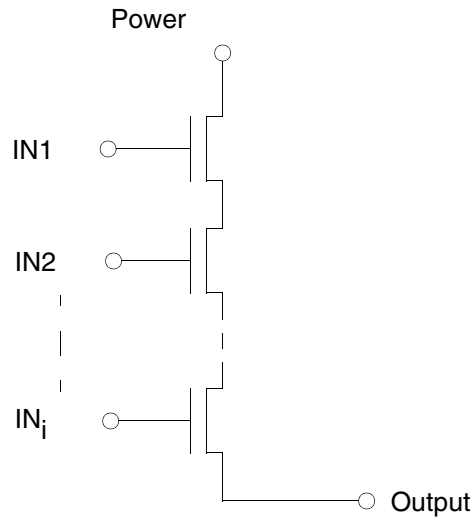
Simple Parallel Pull-Up (PUP) Structure

PUP Output, IN1, IN2,..., IN_i



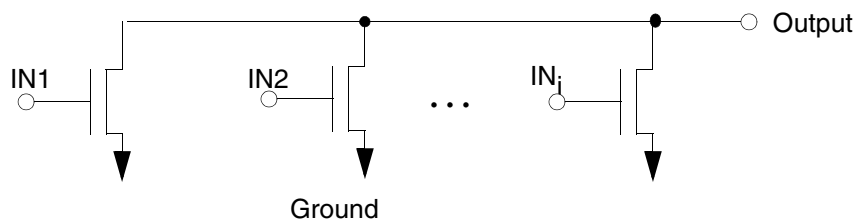
Simple Series Pull-Up (SUP) Structure

SUP Output, IN1, IN2,..., IN_i



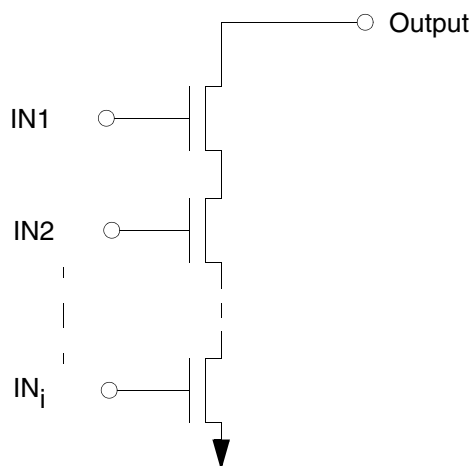
Simple Parallel Pull-Down (PDW) Structure

PDW Output, IN1, IN2,..., IN_i



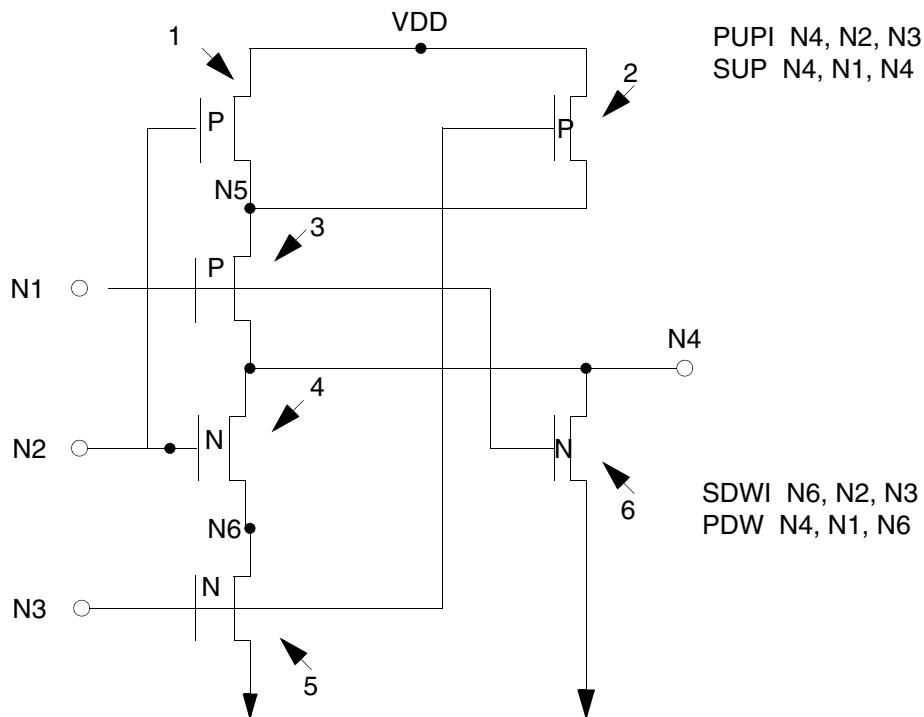
Simple Series Pull-Down (SDW) Structure

SDW Output, IN1, IN2, ..., IN_i



Complex Gate Including PUPI, SUPI, PDWI, and SDWI

Parallel pull-ups and pull-downs and series pull-ups and pull-downs are element structures of complex gates. When LVS encounters a complex structure, it first recognizes and represents the devices in the innermost structure, then uses these devices to drive the outer structure to form a cascaded structure of these elements. In the following example, CMOS circuit devices 1 and 2 form a PUPI element that together with device 3 form a SUP element driving the output node. Devices 4 and 5 form a SDWI element that together with device 6 form a PDW element driving the output node.

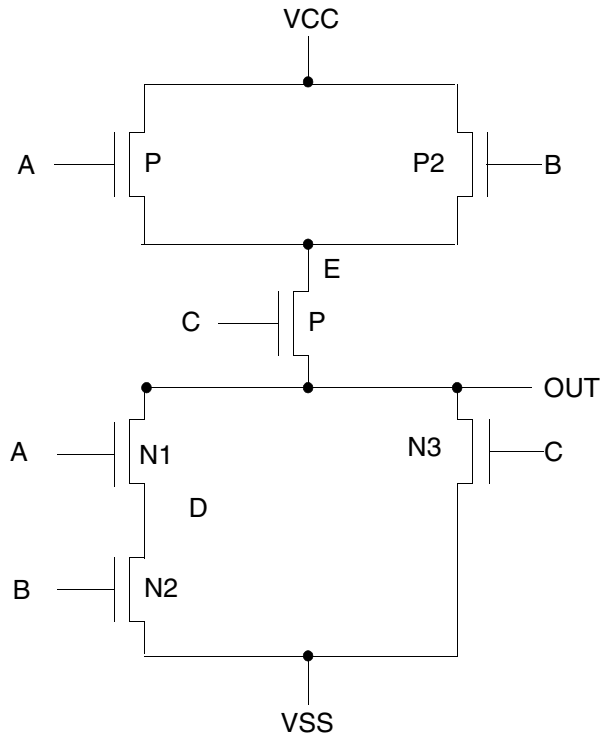


Because multiple series or parallel structures are involved in one complex structure, you must interpret the whole complex structure rather than just element structures in order to comprehend the true logic functions. In the outer structure (SUP and PDW in this example), LVS chooses the output node of the complex gate to represent itself. In the example, *N4* is this node.

Inside a parallel structure, LVS chooses the internal nodes that are closest to the output node in order to represent the associated series substructure. LVS chooses *N6* to represent the SDWI inside the parallel structure PDW.

Inside a series structure, LVS inherits its representation node to its parallel substructure. LVS chooses *N4* to represent the PUI under SUP. This process is applied from the outermost structure through the innermost structure.

PUPI and SDWI

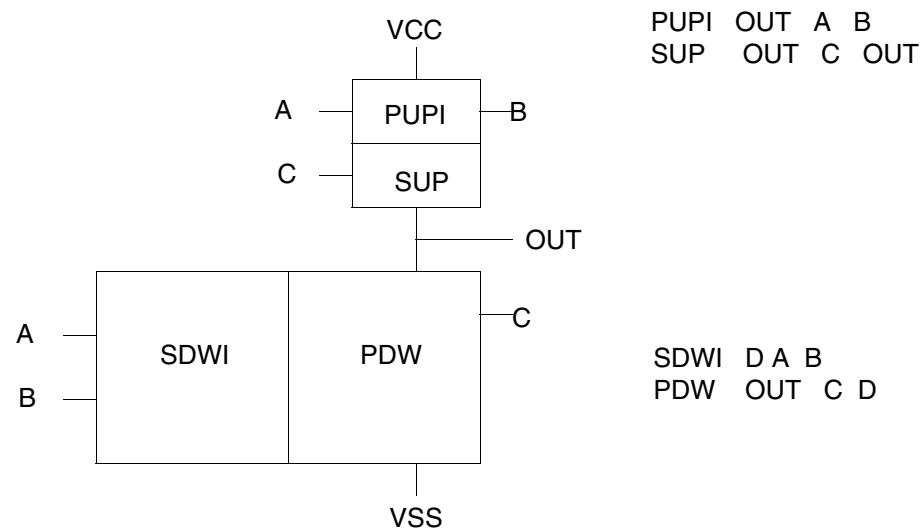


When PUP drives SUP, PUP is recognized as an incomplete PUP (PUPI) and the output node (in this example, OUT) is used as the output of the PUPI structure.

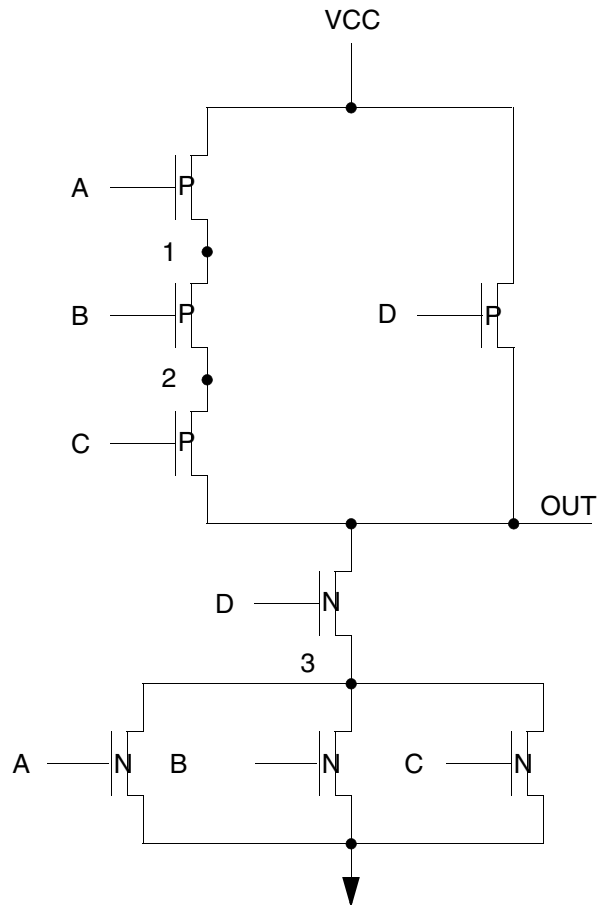
When SDWI drives PDW, use the internal node (in this example, *D*) to represent the output node of SDW that is recognized as an incomplete SDW (SDWI).

Dracula Reference
Comparing Layouts and Schematics (LVS)

The pull-down elements for the complex structure are represented as follows:



SUPI and PDWI



The pull-up elements for this complex structure are represented as follows:

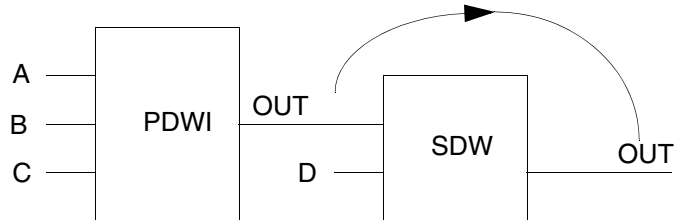
When SUP drives PUP, use node 2 as the output node of the SUP structure, SUP_I.

SUP _I	2,	A,	B,	C
PUP	OUT,	D,	2	

Dracula Reference

Comparing Layouts and Schematics (LVS)

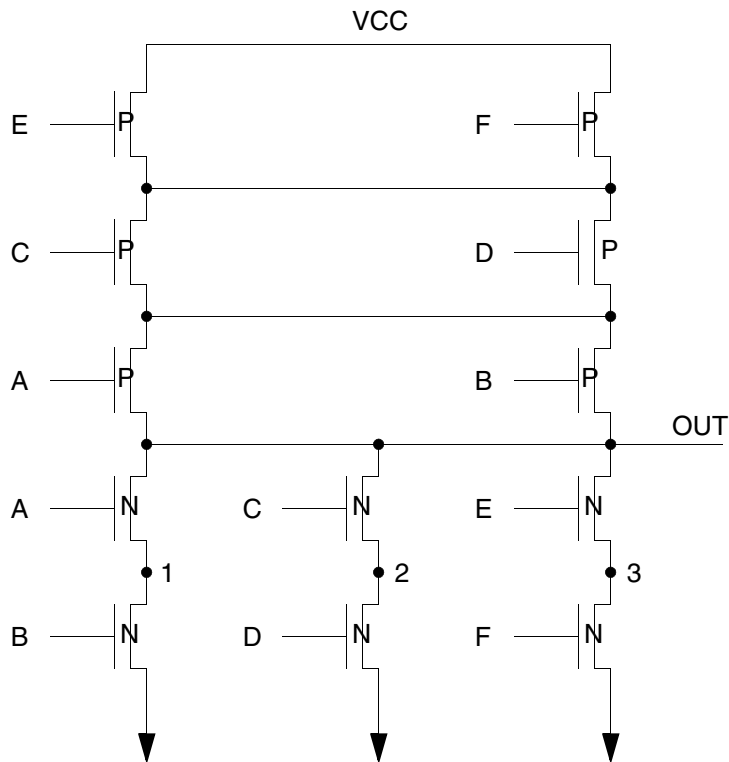
The pull-down elements for this structure are represented as follows:



When PDW drives SDW, use the output node of SDW to represent the output node of PDW, which is recognized as PDWI (incomplete PDW).

PDWI	OUT,	A,	B,	C
SDW	OUT,	D,	OUT	

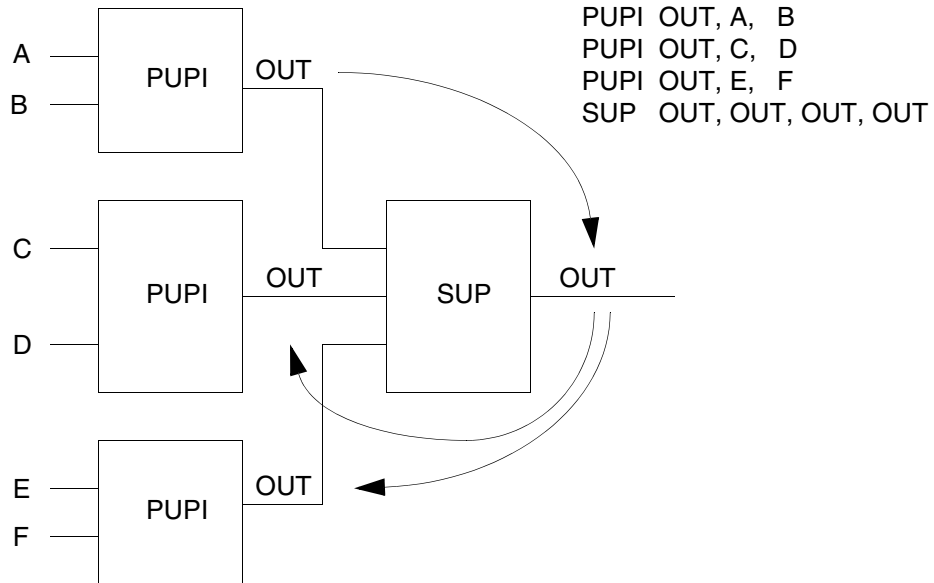
PUPI and SDWI



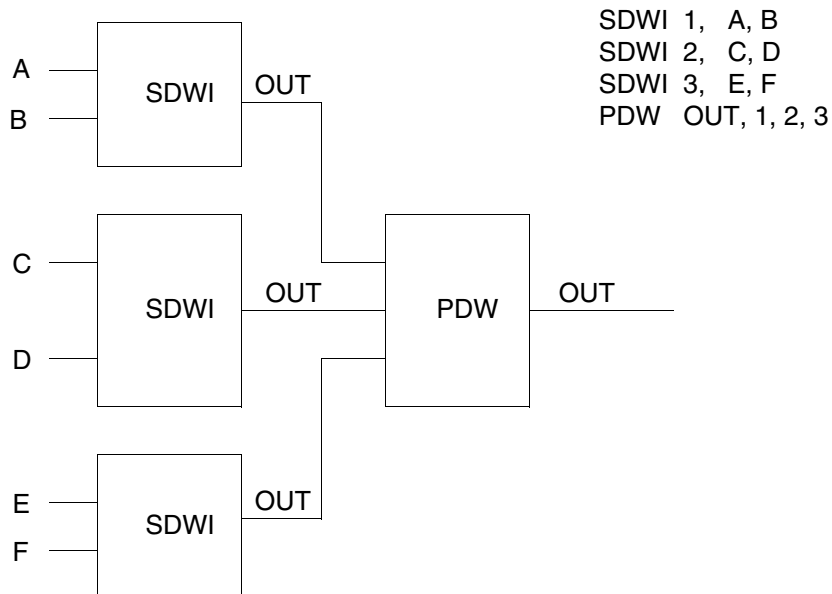
Dracula Reference

Comparing Layouts and Schematics (LVS)

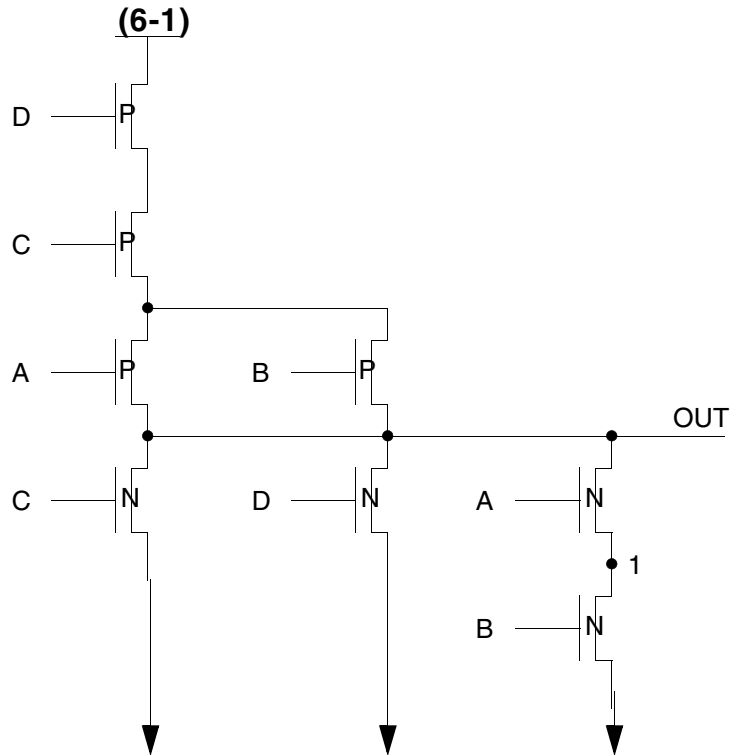
The pull-up elements for this complex structure are represented as follows:



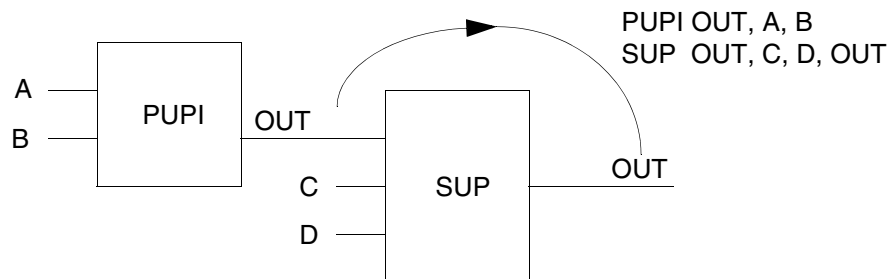
The pull-down elements for this complex structure are represented as follows:



PUPI and SDWI



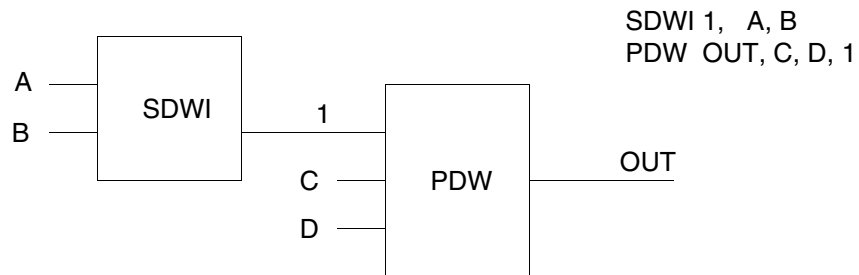
The pull-up elements for this complex structure are represented for as follows:



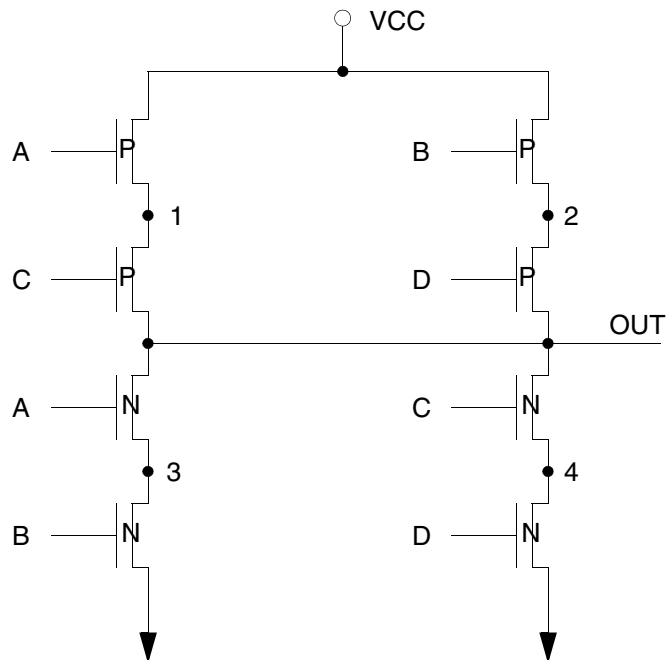
Dracula Reference

Comparing Layouts and Schematics (LVS)

The pull-down elements for this complex structure are represented as follows:

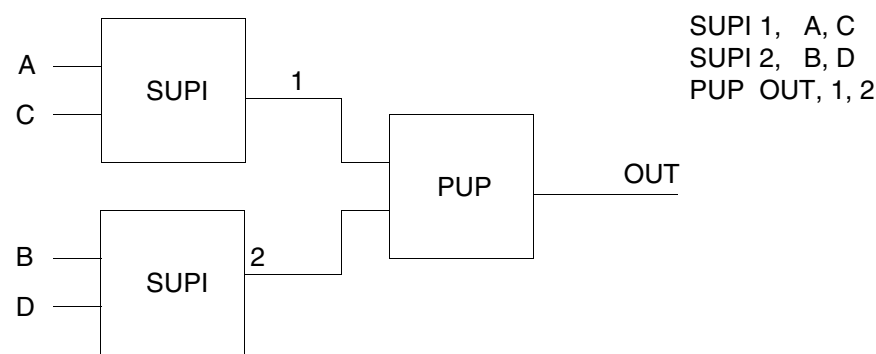


SUPI and SDWI

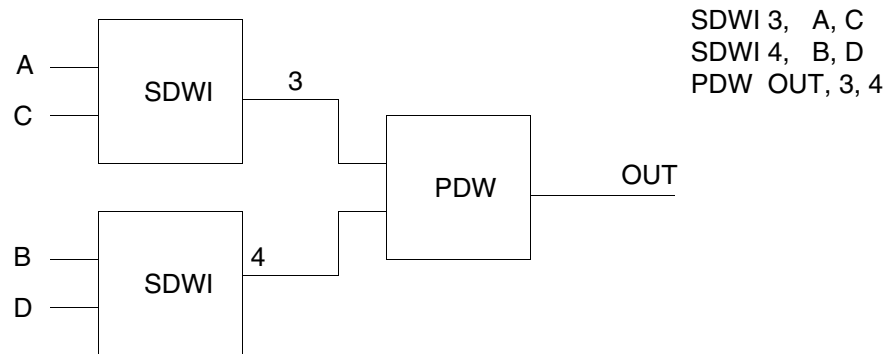


Dracula Reference
Comparing Layouts and Schematics (LVS)

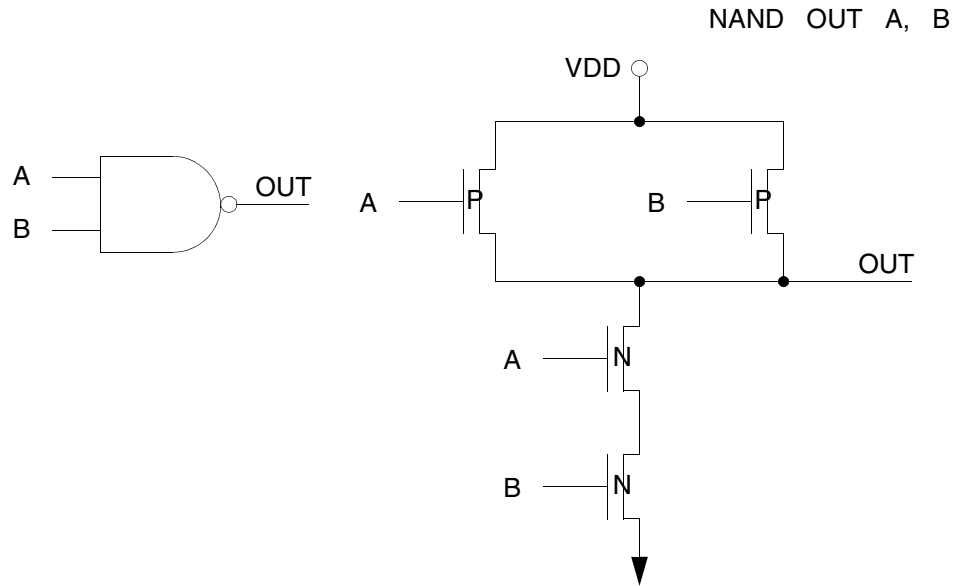
For pull-up elements, the structure can be as follows:



For pull-down elements, the structure can be as follows:

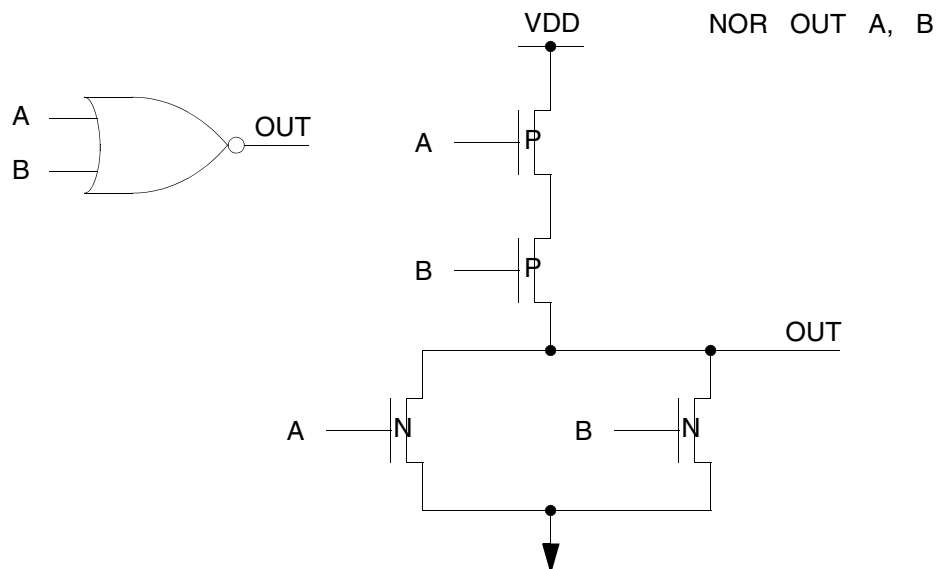


CMOS NAND



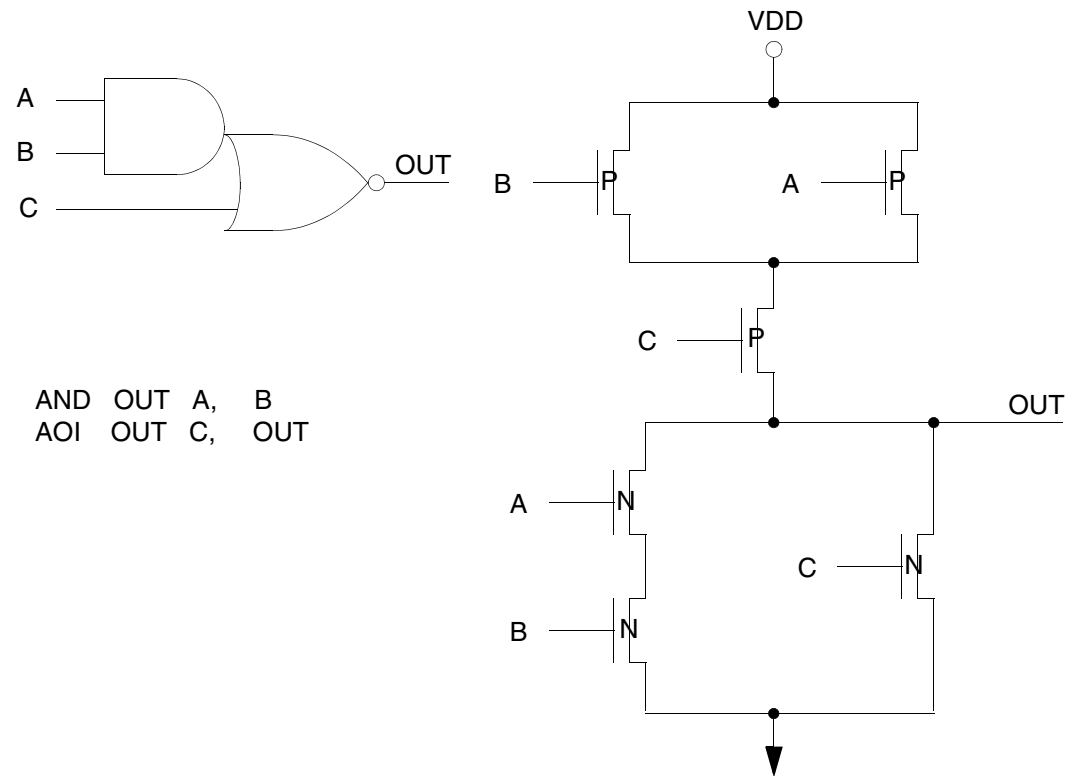
Note: To create a CMOS AND gate, you must follow the NAND gate with an inverter.

CMOS NOR

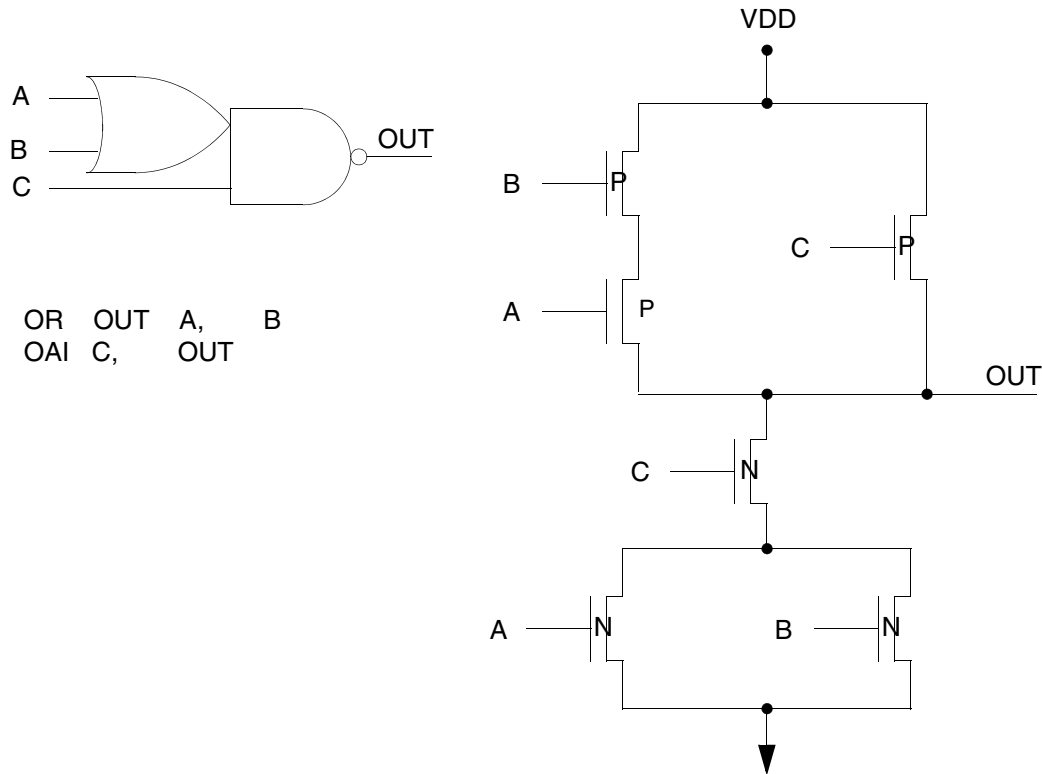


Note: To create a CMOS OR gate, you must follow the NOR gate with an inverter.

CMOS AOI

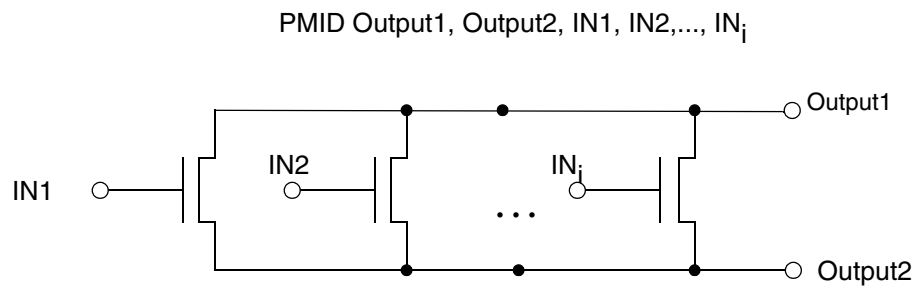


CMOS AOIs



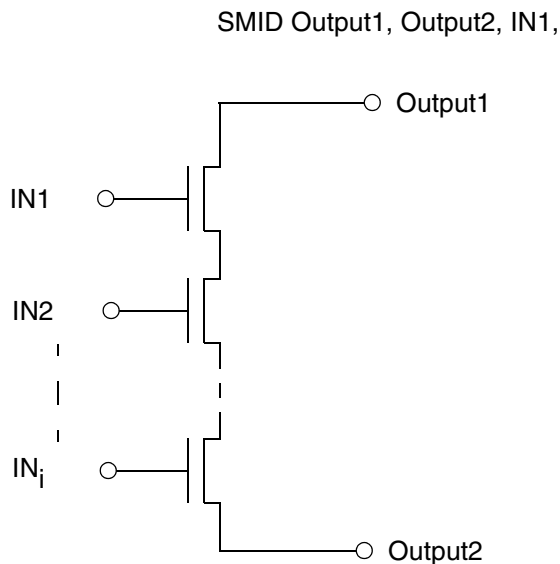
Simple Parallel Middle (PMID) Structure

This is a simple parallel middle (that is, neither pull-up nor pull-down, formed only when LVSCHK/LPECHK option [O] is used) structure of MOS transistors with the same implant.



Simple Series Middle (SMID) Structure

This is a simple series middle (that is, neither pull-up nor pull-down, formed only when LVSCHK/LPECHK option *O* is invoked) structure of MOS transistors with the same implant.



Undefined Elements

In the circuit extraction process of the layout, any fixed terminal element that encounters an undefined number of terminals automatically becomes an undefined element in LVS and is listed with a question mark (?).

?MOS: A MOS transistor with more than two source or drain terminals.

?BJT: A bipolar transistor with more than one base terminal.

?RES: A resistor with more than two terminals.

?CAP: A capacitor with more than two terminals.

?DIO: A diode with more than two node terminals.

Note: LVS encounters these undefined elements caused by unconventional layout techniques or an incorrectly specified *element* device layer.

Isolating and Debugging Errors

LVS uses expanded node names and device names to identify the schematic node, device, or subcircuit in error. LVS also identifies the higher level gate or functional block, of which these devices are a part. For the layout devices, LVS prints x and y locations in natural units.

Dracula Reference

Comparing Layouts and Schematics (LVS)

Furthermore, LVS error types can generate graphic output, which in some cases points to errors more directly than listings (like showing the redundant errors of arrays). To isolate the errors, follow these steps:

1. Focus on a discrepancy point. Identify its key device or node and understand the specific LVS error type.
2. Locate schematic gates of the functional block by their expanded names and layout locations.
3. Locate matched nodes or devices by their expanded names and layout locations.
4. Interpret unmatched nodes or devices by LVS error types.
5. Check the LVS graphic output by the same LVS error type.
6. View the layout data on the graphic workstation to verify the findings, or use Dracula Interactive.
7. Inspect the Dracula log file for the following modules:
 - ❑ The first REDUCE module reduces the layout side to second-level devices (logic gates and pseudo gates) for LVS discrepancy reporting. LVS lists a summary table of all devices at the end of this module. The summary is also included in the LVS report file.
 - ❑ The second REDUCE module reduces the schematic side in the same manner as the layout side. If you compare the schematic summary table at the end of the REDUCE module in the log file, the total number of logic gates, pseudo gates, and MOS devices generated match with the layout summary. If there are fewer logic gates or pseudo gates, this usually means that power or ground was lost (or partially lost) in either the layout side or schematic side. For example, if the layout side has 100 more MOS devices and 30 fewer logic gates, you can assume that somewhere in the layout side there was a loss of power or ground. The ERC PATHCHK command can locate the nodes that are stuck or floating. The summary is also included in the LVS report file.

Note: If the LVSCCHK [X] option is *on*, LVS does not reduce any MOS devices.

Interpreting an LVS Discrepancy

In this sample error summary file, discrepancies 1 and 2 are related. In discrepancy 1, LVS traces NODE 74 and in discrepancy 2, LVS traces NODE 84.

```
***** DISCREPANCY 1 *****
----- NODE      74 -----WITH EXTRA-LAYOUT DEVICES-----
```

Dracula Reference

Comparing Layouts and Schematics (LVS)

OCCURRENCE NAME XEN1-10

DEV15 MOS P---MP2XEN13z : DEV572 MOS P X=-322 Y=205
 XEN1-10, 74, XEN1-9 XEN1-10, XEN1-9, 74

OCCURRENCE NAME 84

DEV199 PUP : DEV734 PUP X=-170 Y=150
 84, 85, 74, 96 84, 74, 85, 96
 DEV201 SDWI : DEV738 SDWI X=-170 Y=61
 84, 95, 96, 74 97, 85, 96, 74
 DEV200 SDWI : DEV737 SDWI X=-31 Y=61
 84, 85, 96, 74 98, 85, 96, 74

**** UN-MATCHED **** DEV736 SDWI X=-170 Y=72
 99, 985, 96, 74

**** UN-MATCHED *** DEV735 SDWI X=-31 Y=72
 100, 85, 96, 74

***** DISCREPANCY 2 *****

----- NODE 84 -----

DEV199 PUP : DEV734 PUP X=-170 Y=150
 84, 85, 74, 96 84, 74, 85, 96
 ?DEV202 PDW : **** UN-MATCHED ****
 84, 97, 98

**** UN-MATCHED **** : ?DEV741 PDW X=-170 Y=14
 84, 97, 98, 99, 100

DEV201 SDWI : DEV738 SDWI X=-170 Y=61
 97, 95, 96, 74 97, 85, 96, 74
 DEV200 SDWI : DEV737 SDWI X=-31 Y=61
 98, 85, 96, 74 98, 85, 96, 74

**** UN-MATCHED **** : DEV736 SDWI X=-170 Y=72
 99, 985, 96, 74

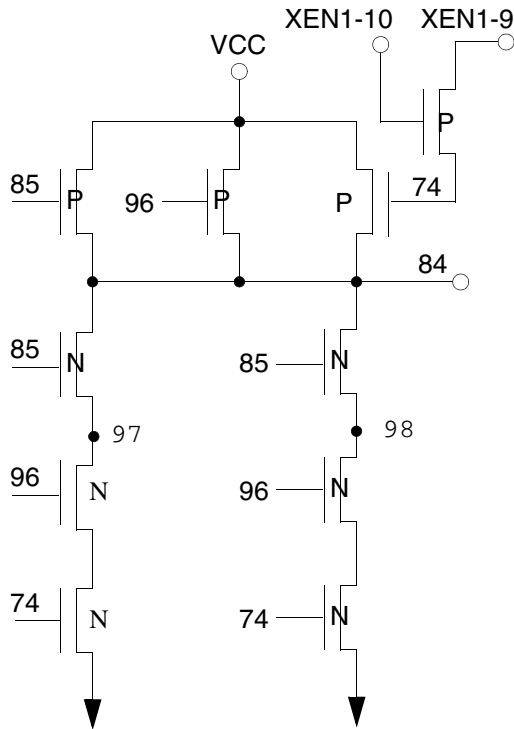
**** UN-MATCHED *** : DEV735 SDWI X=-31 Y=72
 100, 85, 96, 74

Dracula Reference

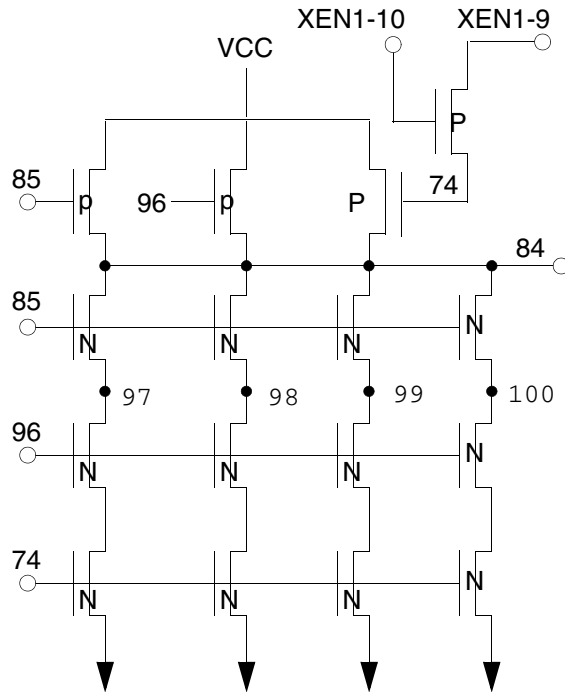
Comparing Layouts and Schematics (LVS)

The following is a schematic representation for discrepancy 1:

Schematic



Layout



Result:

The schematic side has two SDWI gates in parallel. The layout side has four SDWI gates in parallel. Discrepancy 2 is the repetition of discrepancy 1 traced from nodes 74 and 84.

Logical Equivalence in MOS Digital Circuits

For certain digital MOS circuits, the physical order of the inputs to the gates of the MOS devices in series or parallel are interchangeable because they are logically equivalent. LVS handles logical equivalences through the recognition of logic gates and pseudo gates. All inputs to these gates are interchangeable. LVS recognizes logic gates and pseudo gates by default. To check the physical order of MOS devices, you can use the following methods:

- To stop the formation of logic gates and pseudo gates to force LVS to trace all MOS transistors individually, turn on the LVSCCHK[X] option. This option is useful for analog circuits.

- Use the special MOS element type ELEMENT MOS[X-] and specify the subtype of MOS CDL transistor card of type X* to stop the formation of logic gates and pseudo gates locally. For example:

```
M12 12 18 1 1 X*
```

where * is any alphanumeric character.

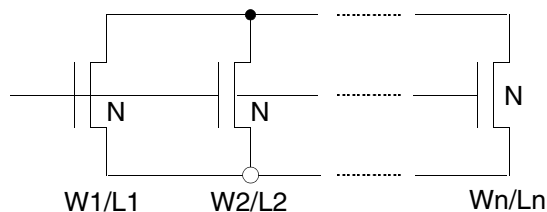
Calculations on Parallel MOS Devices for LVS

For some digital MOS circuits, you must place parallel transistors in the layout. Unless suppressed by an LVCHK[K] option, the program smashes these devices into one transistor. This section describes the formula LVS uses to combine the transistor sizes of these parallel devices.

For parallel transistors, you must specify both length and width. If you do not specify either length or width, the length and width of the parallel transistors is 0, which means unspecified.

W/L Calculation

These formulas compute the width and length of the transistor that replaces a group of n parallel transistors as the result of the smashing operation.



Compute the width and the length of the smashed transistor as follows:

$$W = \sqrt{A * C}$$

$$L = \sqrt{A / C}$$

where you compute the area, $A = W * L$, of the resulting transistor as the sum of the areas of the parallel transistors

$$A = a(1) + \dots + a(n)$$

$$= w(1)*l(1) + \dots + w(n)*l(n)$$

and you compute the width/length ratio $C = W/L$ of the resulting transistor as the sum of the width/length ratios of the parallel transistors.

$$\begin{aligned} C &= c(1) + \dots + c(n) \\ &= w(1)/l(1) + \dots + w(n)/l(n) \end{aligned}$$

This smashing operation preserves the electrical properties of the circuit.

For more information about the layout MOS device W/L calculation, refer to [“LPE Overview”](#) on page 355.

W_{eff} MULDELW Calculation

Because of process variations such as optical, etching effects, and implant/field oxide encroachment, you can compensate for changes in effective channel widths when laying out multiple parallel finger MOS transistors. Use the following equation to check schematic MOS device effective widths against compensated layout MOS device widths.

$$W_{\text{eff}} = W_{\text{actual}} + (N-1) * \text{MULDELW}$$

The value for MULDELW is typically negative. The default is 0.

MULDELW modifies the effective width (W_{eff}) of MOS devices that have multiple fingers in parallel. For both schematic devices and layout devices, calculate the effective widths as follows:

$$W_{\text{eff}} = W_{\text{actual}} + (N-1) * \text{MULDELW}$$

where

- W_{eff} is width used for LVS comparison.
- W_{actual} is sum of all widths of parallel fingered devices.
- N is the number of smashed parallel fingers of the transistor.

If a transistor has only one finger, then $N=1$ and $W_{\text{eff}}=W_{\text{actual}}$ as well. If you specify the K option, then N always is 1, because K prevents smashing of parallel legs of a device. If you

Dracula Reference

Comparing Layouts and Schematics (LVS)

specify the *S* option for split-gates, then *N* for these devices is calculated like normal parallel devices after these gates are smashed.

```
*OPERATION
.
.
LVSCHK    WPERCENT=2 MULDELW= -1
*END
```

Example 1

Schematic

M1 W=30 L=2

Layout

MOS Device 1a W=10 L=2

MOS Device 1b W=11 L=2

MOS Device 1c W=11 L=2

Calculations:

$$\begin{aligned}W_{\text{eff}} &= 30 + (1-1) * (-1) \\&= 30 + 0 \\&= 30\end{aligned}$$

$$\begin{aligned}W_{\text{eff}} &= 32 + (3-1) * (-1) \\&= 32 + (-2) \\&= 30\end{aligned}$$

No discrepancy.

Example 2

Schematic

M1 W=15 L=2

M2 W=16 L=2

Layout

MOS Device 1a W=15 L=2

MOS Device 1b W=16 L=2

Calculations:

$$W_{\text{eff}} = 31 + (2-1) * (-1)$$

$$W_{\text{eff}} = 31 + (2-1) * (-1)$$

Dracula Reference

Comparing Layouts and Schematics (LVS)

$$=31 + (-1) \qquad \qquad \qquad = \quad 31 + (-1)$$

$$=30 \qquad \qquad \qquad = \quad 30$$

No discrepancy.

Example 3

Schematic	Layout
M1 W=30 L=2	MOS Device 1a W=15 L=2
	MOS Device 1b W=15 L=2

Calculations:

$$\begin{aligned} W_{\text{eff}} &= 30 + (1-1) * (-1) \\ &= 30 + 0 \\ &= 30 \end{aligned}$$

$$\begin{aligned} W_{\text{eff}} &= 30 + (2-1) * (-1) \\ &= 30 + (-1) \\ &= 29 \end{aligned}$$

Discrepancy; >2% error.

MOS Devices Without Substrate Terminal

Some MOS devices are defined without substrate terminal. Currently Dracula would automatically output 0 as the substrate terminal if it is not defined in the rule file for layout-only run or LVS run without the S option specified in LPESELECT. Users expect to see a netlist without substrate terminal output if the above situation happens. The 4.8 version provides a solution to this problem by providing a new command, NO-SUB-TERM in the Description Block to offer the capability of not defining a substrate when extracting a device using LPE.:

Command Syntax

```
NO-SUB-TERM=mos_subtype1{mos_subtype2,...mos_subtypen}
```

Arguments

mos_subtype

is the sub-type of MOS which is specified in the ELEMENT MOS command. For example, "N" of the following element definition:

```
ELEMENT MOS[N] NDEVICE POLY NSD
```

The command would only be effective in the following condition, otherwise, it would be ignored:

- There is no MOS substrate terminal defined in the rule file
- In a layout only run or LVS run without S option specified in LPESELECT

Example

```
NO-SUB-TERM =N,P
```

```
.....
```

```
ELEMENT MOS[N] NGATE POLY NSD
```

```
ELEMENT MOS[P] PGATE POLY PSD
```

```
.....
```

An output example may look like the following:

```
M1      OUT  IN  VDD   P  L=3.00U  W=6.25U
```

```
M2  GND IN OUT  N L=3.00U  W=6.25U
```

Calculations on Parallel and Series Resistor Devices width/length for LVS

For some circuits, you have to place parallel or/and series resistors in the layout. Unless these are suppressed by an LVSCHK[K] option, Dracula smashes parallel devices into one resistor. LVSCHK[R] provides smashing series resistors into one resistor.

This section describes the expression LVS uses to estimate the effective resistor length (L_{eff}) and width (W_{eff}) when combining (or smashing) parallel and series resistors:

- For parallel resistors the effective values are calculated by formulas:

$$L_{eff} = \text{SQRT}(\text{AREA} / (W1/L1 + W2/L2))$$
$$W_{eff} = \text{AREA} / L_{eff}$$

Dracula Reference

Comparing Layouts and Schematics (LVS)

- For series resistors the effective values are calculated by formulas:

$$W_{eff} = \text{SQRT}(\text{AREA} / (L1/W1 + L2/W2))$$

$$L_{eff} = \text{AREA} / W_{eff}$$

Where:

W1 - the width of the first resistor,

L1 - the length of the first resistor,

AREA1 - the area of the first resistor,

W2 - the width of the second resistor,

L2 - the length of the second resistor,

AREA2 - the area of the second resistor,

AREA - the calculation of the combined AREA is controlled by the RESWSMASH command (see “RESWSMASH” on page 762):

- ❑ for RESWSMASH = NO (the default)

$$\text{AREA} = \text{AREA1} + \text{AREA2}$$

- ❑ For RESWSMASH = YES

$$\text{AREA} = 1 / (1/\text{AREA1} + 1/\text{AREA2})$$

The calculations are provided step by step.

Bipolar LVS Example

The following is an example of an input rules file in bipolar technology.

```
;*****
;
;          BIPOLAR LVS INPUT RULES FILE
;          P - SUBSTRATE AND DOUBLE LAYER METAL
;*****
;          ***** TYPE OF DEVICES; *****
;
;*
;* DEVICE/ELEMENT          TYPE
;* -----
;* NPN VERTICAL            [N]
;* PNP VERTICAL            [PV]
;* PNP LATERAL             [PL]
;* DIODE SCHOTTKY          [S]
;* CAPACITOR MET1/DEEP N+  [C]
;* RESISTOR DEEP N+         [DN]
;* RESISTOR SHALLOW P     [SP]
;* RESISTOR BURIED N+      [BN]
;* ;*****
;          DESCRIPTION BLOCK
```

Dracula Reference

Comparing Layouts and Schematics (LVS)

```
;*****
*DESCRIPTION
PROGRAM-DIR = /drac/bin      ; Program directory of Dracula modules
SYSTEM = gds2                ; Graphic CAD database format
PRIMARY = top                ; Top cell name
INDISK = bip.db              ; Graphic database file name
OUTDISK = lvsout             ; Graphic error cell file name
SCALE = 0.001 microns        ; Graphic database scale
RESOLUTION = 0.05 microns    ; Database digitized grid unit
MODE = exec now              ; Mode execution
PRINTFILE = lvsprt           ; Print file name of DRACULA job
;
SCHEMATIC = LVSLOGIC         ; Schematic netlist from LOGLVS compiler
;
*END

;*****
;                               INPUT-LAYER BLOCK
;*****
*INPUT-LAYER
bnn = 1      ; Buried N+ layer
nepi = 2     ; Epi region( Thin oxide region )
sp = 3       ; Shallow P diffusion
sn = 4       ; Shallow N+ diffusion
cont = 5     ; Metal to diffusion contact
met1 = 6     ; Bottom metal layer
vapox = 7    ; Pad opening
dn = 9       ; Deep N+ diffusion
dp = 10      ; Deep P+ diffusion
contact1    ; Top metal to bottom metal
met2 = 12    ; Top metal layer
;
SUBSTRATE= psub 63           ; Create p - substrate layer
;
CONNECT-LAYER= psub buried pplus enpn ncoll met1 met2
; *** The conductor layers of
;         the bipolar process ***
;
*END ;*****
;                               OPERATION BLOCK
;*****
*OPERATION
;
EDTEXT ram.txt ; Dracula text file
; ;*****
; LOGICAL OPERATION TO FIND THE VERTICAL NPN DEVICES
;*****
NOT sp sn cheese

SELECT sn hole cheese enpn      ; Emitter of the vertical npn
;                               ; transistor and device layer (also
;                               ; for the multiple emitter devices).

SELECT bnn ENCLOSE enpn blnpn ; Buried layer under the npn device
;*****
; LOGICAL OPERATION TO FIND THE LATERAL PNP DEVICES
;*****
AREA sp RANGE 224 226 epnp11 ; 15 by 15 micron sq. shallow p
AREA sp RANGE 255 257 epnp12 ; 16 by 16 micron sq. shallow p
;                               ; OR epnp11 epnp12 epnp13
SELECT epnp13 INSIDE nepi epnp14
```

Dracula Reference

Comparing Layouts and Schematics (LVS)

```
AND epnp14 bn epnp1 ; Emitter of the lateral pnp
;
; SIZE epnp1 BY 9 oepnp1      ; Create a layer to find the
;                               ; collectors
SELECT sp CUT oepnp1 cpnp1    ; Collector of the lateral pnp
;                               ; transistor and device layer (also
;                               ; for multiple collector devices)
; *****
; LOGICAL OPERATION TO FIND THE VERTICAL PNP DEVICES
; *****
SELECT sp OUTSIDE bnn epv1
SELECT epv1 OUTSIDE dp epnpv ; Emitter of the vertical pnp
;                               ; transistor
SIZE epnpv BY 9 oepnpv      ; Device layer for the vertical pnp
;                               ; transistor.
AND sp nepi pt1
AND pt1 dp pt2
AND pt2 cont ptpsub          ; Deep p+ contact to p substrate
;                               ; which is also the collector of the
;                               ; vertical pnp transistor.
; *****
; LOGICAL OPERATION TO CREATE THE N +
; CONDUCTOR LAYERS
; *****
; ENPN si already formed      ; ENPN is the shallow n+ interconnect
;                               ; layer
AND sn dn ncoll1
AND ncoll1 nepi ncoll        ; NCOLL is the deep n+ interconnect
;                               ; layer to buried n+ layer
; *****
; LOGICAL OPERATION TO FIND THE DEEP N RESISTORS
; *****
AND cont nepi rcont1
SIZE rcont1 BY 3 rcont      ; Oversized contacts used to separate
;                               ; resistors into discrete regions.
SELECT dn ENCLOSE nepi dnres1
AND dnres1 nepi dnres2
NOT dnres2 bnn dnres3
NOT dnres3 rcont dnres      ; Deep n+ resistor region and device
;                               ; layer.
; *****
; LOGICAL OPERATION TO FIND THE P+ RESISTOR DEVICE
; *****
SELECT sp ENCLOSE nepi spres1
AND spres1 nepi spres2
SELECT spres2 OUTSIDE enpn spres3
SELECT spres3 OUTSIDE dp spres4
NOT spres4 rcont spres      ; Shallow p+ resistor
;                               ; region and device layer
; *****
; LOGICAL OPERATION TO CREATE THE P +
; ( PPLUS ) CONDUCTOR LAYER
; *****
NOT sp spres4 spl
AND spl nepi pplus          ; Shallow p interconnect layer
; *****
; LOGICAL OPERATION TO FIND THE BURIED N +
; RESISTOR DEVICES
```

Dracula Reference

Comparing Layouts and Schematics (LVS)

```

;*****
SELECT nepi OUTSIDE sp bnpres1
NOT bnpres1 blnpn bnpres2
SELECT bnpres2 CUT sn bnpres3
NOT bnpres3 dn bnpres          ; Buried n+ resistor region
;                               ; and device layer
;*****;
        LOGICAL OPERATION TO CREATE THE
;        BURIED N+ CONDUCTOR LAYER
;*****
SELECT bnn ENCLOSE bnpres burl
NOT bnn burl buried          ; Buried n+ interconnect
;                               ; layer.
;*****;
        LOGICAL OPERATION TO FIND THE METAL1 TO
;        DEEP N+ CAPACITOR DEVICES
;*****

AND met1 dn cap1
AND cap1 nepi cap2
AND cap2 bnn cap3
NOT cap3 blnpn cap4
SELECT cap4 OUTSIDE cont cap5
SELECT cap5 OUTSIDE sn mncap   ; Metall to deep N+ capacitor
;                               ; region and device layer.
;*****;
        LOGICAL OPERATION TO FIND THE SCHOTTKY
;        DIODE DEVICES
;*****

AND met1 cont diode1
AND diode1 nepi diode2
AND diode2 bnn diode3
NOT diode3 sp diode4
NOT diode4 dn diode5
NOT diode5 sn sanode          ; Anode region of the schottky diode
;                               ; and device layer.
;*****;
        CONNECT OPERATION FOR THE BIPOLAR PROCESS
;*****

CONNECT met2 met1 BY via
CONNECT met1 enpn BY cont
CONNECT met1 ncoll BY cont
CONNECT met1 pplus BY cont
CONNECT ncoll buried BY bnn
CONNECT met1 psub BY ptpsub
;

STAMP cont BY met1          ; Contact now carries nodal information
STAMP enpn1 BY pplus
STAMP oepn1 BY enpn1        ; Oepn1 now carries nodal information
STAMP rcont BY met1         ; Rcont now carries nodal information
;
AND met2 vapox bpad          ; Bonding pad region and device layer
;*****;
        DEFINING TRANSISTORS, DIODE, CAPACITOR,
;        AND RESISTORS ELEMENTS
;*****
;        DEVICE TYPE  DEVICE-LAYER  COLLECTOR  BASE      EMITTER
;        -----

```


Dracula Reference

Comparing Layouts and Schematics (LVS)

```

;
ELEMENT BJT[N]          enpn          buried      pplus      enpn
ELEMENT BJT[PL]         cpnpl         p plus     buried     oepnpl
ELEMENT BJT[PV]         oepnpv        psub       buried     pplus
;
;
;          DEVICE TYPE   DEVICE-LAYER   ANODE       CATHODE
;          -----
ELEMENT DIO[S]         sa node         cont        buried
;
;
;          DEVICE TYPE   DEVICE-LAYER   TERMINAL
;          -----
ELEMENT RES[DN]        dnres          rcont
ELEMENT RES[SP]        spres          rcont
ELEMENT RES[BN]        bnres          ncoll
ELEMENT PAD            met2           bpad
;
;
;          DEVICE TYPE   DEVICE-LAYER   TERMINAL1   TERMINAL2
;          -----
ELEMENT CAP[C]         mncap          met1        buried

MULTILAB OUTPUT shorts 40      ; ERC Shorts test
SAMELAB OUTPUT opens 40       ; ERC Opens test
;
*BREAK LVS                      ; Breakpoint for restarting LVS for
;                                ; discrepancy report only(.LVS).
LVSCHK CAPVAL=5 DIOAREA=3      ; Layout versus schematic check
LVSCHK DIOPERI=2 RESSIZE=10
*END

```

Hierarchical LVS

Hierarchical Layout Versus Schematic (HLVS) works like LVS in flat Dracula, except that it operates on a two-level hierarchy like HDRC. Dracula processes specified Hcells and traces them separately during a cell mode HLVS run. After verifying the Hcells, Dracula HLVS is run in composite mode to verify the composite plane devices and Hcell connections.

Executing cell mode HLVS and then composite mode HLVS is the same as running flat mode LVS. Both do a completely verification of the layout against the schematic. Because of the reduced amount of data in a hierarchical structure, HLVS can handle very large chips. HLVS discrepancy reports (PRINTF.LVS) are concise because they do not replicate errors for every placement of an Hcell.

Even though HDRC and HLVS use a similar two-level hierarchy, Hcells in HLVS are modeled differently than Hcells in HDRC. Therefore, the databases of these two functions are incompatible. HDRC and HLVS are run as totally independent operations.

HLVS generates an Hcell box to define the boundary of every Hcell. This rectangular box is defined to cover the minimum area that encloses all geometries belonging to that Hcell.

Basic Requirements

To run HLVS, you must have the following data:

- A database file.

For a list of supported database formats, refer to the SYSTEM command description in [Chapter 11, “Description Block Commands.”](#)

- A netlist file (VERILOG, CDL, SPICE circuit file, etc).

Just as in flat Dracula, you use the CDL or SPICE circuit file to specify transistor level cells, and you can use it to specify the whole netlist. For more information about CDL netlist files, refer to the "Compiling Network Descriptions" (LOGLVS) section in Chapter 5.

- A composite text layer or EDTEXT file.

Refer to the EDTEXT command in the “Operation Block Commands” chapter.

- A cell text layer or HEDTEXT file.

Refer to the HEDTEXT command description in [Chapter 13, “Operation Block Commands.”](#)

- An Hcell table named by an HCELL-FILE command. This file contains statements with name pairs showing the correspondence between layout and the netlist Hcell names. The example below shows the format of this file. The name on the left is the layout Hcell name; the name on the right is the netlist Hcell name. You can also specify a FRAME option. You can specify individual Hcell commands to take the place of this file.

Here is a sample Hcell file:

QOINVERT	INVERTER
DW00NAND	NAND
DW001NOR	NOR

For more information about Hcell files, refer to the HCELL and HCELL-FILE command descriptions in [Chapter 11, “Description Block Commands.”](#)

- A Dracula rules file.

Selecting HLVS Hcells

You must manually select the Hcells for HLVS, HERC, HLPE, and HPRE, unlike HDRC, where you can have automatic Hcell selection. However, HLVS has a feature that automates Hcell selection.

You use the HCELL or HCELL-FILE command to select Hcells. The HCELL-FILE command specifies a file that contains a set of Hcell statements that are equivalent to Hcell commands. Hcell commands specify the layout cell name for the Hcell and the corresponding netlist cell name (LOGICNAME) that defines the Hcell subcircuit. Dracula needs both names to run HLVS.

More than one Hcell in the layout database can correspond to a single schematic subcircuit cell. For example, two Hcells, named *inv1* and *inv2*, can both correspond to a subcircuit cell named *inv*. This lets you lay out one circuit in different ways.

After you select the Hcells, HLVS flattens the hierarchy under the Hcell and composite plane for both the layout and netlist to establish a two-level hierarchy.

When selecting Hcells, select cells with many placements to reduce the amount of redundant circuitry to be traced. This reduces HLVS execution time. You must conform to certain restrictions when selecting Hcells. HLVS verifies that these constraints are met for all specified Hcells to guarantee that a cell mode and composite mode HLVS check gives a complete layout-to-schematic verification.

The following constraints apply to the selection of Hcells in HLVS:

- The maximum number of Hcells is 2,000.
- No Hcell can be a subcell of another Hcell.
- An Hcell cannot contain a floating contact of any kind. If you use programmable contacts, you must digitize at least one of the contact CONNECT layers with the contact layer to form a valid node.
- Each Hcell must have a corresponding schematic subcircuit. This subcircuit netlist must currently be in either CDL, SPICE, or Verilog as described in the “Compiling Netlist Descriptions” chapter.
- Geometries from the composite plane or other Hcells can overlap Hcells (Hcell box), but they cannot delete, add, or modify any devices within the Hcell, except by the BASE-LAYER command. In cases where such devices are formed by these overlapping geometries, Dracula flags them as “EXTRA CELL TO COMPOSITE” devices.

Automatically Selecting HLVS Hcells

Dracula can automatically select HLVS Hcells, as it does for HDRC. However, in HLVS you must cross reference layout and schematic cells. If layout cell names and their equivalent schematic representations are different, Dracula uses a cross-reference table to match them.

The *ref-filename* lists layout cell names with their corresponding schematic cell names. The format of the reference file is the same as for the HCELL-FILE command. Each line contains two names with blanks as delimiters. The first name is the layout cell name, and the second is the corresponding schematic cell name.

When selecting Hcells, Dracula starts with the cells listed in the reference file. First, Dracula selects Hcells according to the same criteria it uses to select HDRC Hcells. Then Dracula drops all Hcells not listed in the schematic netlist.

Note: You must specify a HCELL_FILE file. If you specify an empty HCELL_FILE file, Dracula considers all cells in the layout for selection.

To run HLVS using automatic Hcell selection, follow these steps:

1. To create the file *jxauto.com*, run PDRACULA.

```
%PDRACULA
%/get rulefile.rul n auto
%/fin
```

2. To generate an Hcell table, execute *jxauto.com*. The HCELL-FILE file specifies the file name for Hcell table.

```
%jxauto.com > jxauto.log
```

3. To examine the validity of the newly generated Hcells, run LOGLVS.

```
%LOGLVS
: CELL/AUTO HCELL-FILE
: .
: X
```

4. To create the final *jxrun.com*, run PDRACULA.

```
%PDRACULA
```

5. To run HLVS, execute *jxrun.com*.

```
%jxrun.com
```

Hcell Pins

Hcell external connections to other Hcells on the composite are allowed only at the pins (also called terminals). In schematics, this is the only way to designate terminal connections. You can identify a pin in the netlist subcircuit by declaring the name as an I/O pin. You can use

text to identify the node of a pin in the layout. In the Hcell layout, place text on a node where the Hcell connects with another Hcell geometry or with a composite geometry.

An Hcell can be a dummy cell with only its I/O pins and power pins digitized and assigned text. A corresponding netlist subcircuit with its I/O pins specified allows an HLVS composite mode run to check the correspondence of the layout block pins (with unspecified internal circuits) against the pins of the corresponding subcircuit in the netlist. For more information about dummy cells, refer to the BLACKBOX-FILE command in [Chapter 11, “Description Block Commands.”](#)

Interconnect between Hcells does not have to be at the Hcell boundary. You can locate pins within the Hcell and connect them using a pin of an overlapping Hcell or to an overlapping geometry from the composite plane.

Hcell Text (Pin Names)

HLVS, like LVS, identifies external pins of a circuit by using your text information. HLVS has texting requirements beyond those of flat mode LVS. To run a cell mode HLVS, text a minimum number of pins in the Hcell layout in addition to a minimum number of pins in the composite plane layout.

For cell mode HLVS, you must text power, ground, and at least one signal node for each independent circuit within the Hcell. An independent circuit is a net consisting of the set of all connections and components that have direct or indirect (that is, through a device) connections among themselves. For example, if two D flip-flops have a pin on the *D* input, a pin on their *Q* output, and no other connections, the two are independent circuits. But if the *D* input of one is tied to the *Q* output of the other, they are part of the same independent circuit.

For a cell mode HLVS run, use the GEN-TEXT command to eliminate manual texting of Hcell pins. GEN-TEXT generates complete Hcell pin text starting from the minimal text information required for a cell mode HLVS run and combines that with the pin names found in the netlist. For more information about texting Hcells, refer to the [GEN-TEXT Commands](#) in this chapter.

Adding to Text

You can add text for the flat or composite planes with either or both of the following methods:

- Read text from the input database by specifying the layer containing it in the `TEXT= layer` command in the Input-Layer block.
- Read text and coordinates from the file listed in the `EDTEXT` command.

You can provide Hcell text with either or both of the following methods:

- Read text from the input database as specified in the layers defined by `CTEXT= layer` commands in the Input-Layer block.
- Read text and text coordinates from the file specified in the `HEDTEXT- file` command. You can create this file with a text editor or the `GEN-TEXT` command. For details, refer to the `GEN-TEXT` command in [Chapter 12, “Input-Layer Block Commands.”](#)

In a composite mode HLVS run, you must assign text to all Hcell pins. An easy way to add text is first to provide minimum texting for the composite plane pins and for the Hcell pins. You can do this by adding text in the original database or by generating a HEDTEXT file. Next, run a cell mode HLVS with `GEN-TEXT` enabled, which creates a new HEDTEXT file with all the Hcell pin text included. Finally, run a composite mode HLVS using the HEDTEXT file created by `GEN-TEXT`.

Pin Text

You can create an additional physical layer for text layers so you can use logical operations on them. When you use the `PIN-TEXT-LAYER` command in the Input Layer block of the rule file, Dracula creates a square on the specified layer surrounding the text origins for each text layer on the composite plane.

The default size for the square is 2x2 database units. You can specify any size for the square; however, if the size is an odd number, the text origin is not in the center because text origins must be located on an integer.

Example

The following example uses the `PIN-TEXT-LAYER` command in the Input-Layer block of a rule file.

```
*DESCRIPTION
.....
*END
*INPUT-LAYER
PIN-TEXT-LAYER = PINLAY
```

```
POLY = 5 CTEXT= 21
.....
CONNECT-LAY = NSUB PWELL PSD NSD POLY METAL
*END
*OPERATION
; Logical Operations
;-----
AND PINLAY METAL LAY1
.....
; Perform Layer Interconnection
;-----
STAMP PINLAY BY LAY1
.....
*END
```

Masking Hcells with Black Boxes

In composite mode, you can mask an Hcell with a “black box.” A black box is a pin geometry-only replacement of an Hcell master whose original definition either does not exist or should be ignored. Define a black box when

- The Hcell master is another company’s proprietary design and you have only the pin information for the Hcell.
- The Hcell master has been verified by other means and you are not concerned about cell-to-master interactions other than the composite-to-cell pin connectivity. This use is similar to the FRAME BY command, except that you provide the frame.

Before defining a black box, you must be aware that HLVS cannot check a black box for violations such as connections from composite-to-cell internal nodes. Also, if the black box is specified incorrectly, it can cause serious problems. Dracula might overlook some LVS errors or report false mismatch violations.

Note: The purpose of using a black box is to reduce the amount of data processed during an HLVS cycle. To prevent missing errors that might be masked by a black box, run a full-chip LVS run as the last step in your verification process.

Use a black box file to define black boxes. The BLACKBOX-FILE command in the Description block identifies the file name. Each line in the black box file generates a pin geometry in the replacement Hcell master. Each line must list the following items:

- The text name
- The x,y coordinates where the pin geometry is to be centered
- The attached layer
- The name of the Hcell

Optionally, you can use width (w) and height (h) specifications to define the shape of the pin.

For more information about the black box file, refer to [“BLACKBOX-FILE”](#) on page 526.

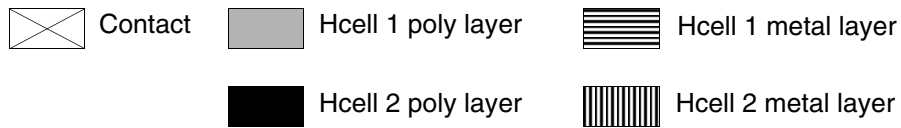
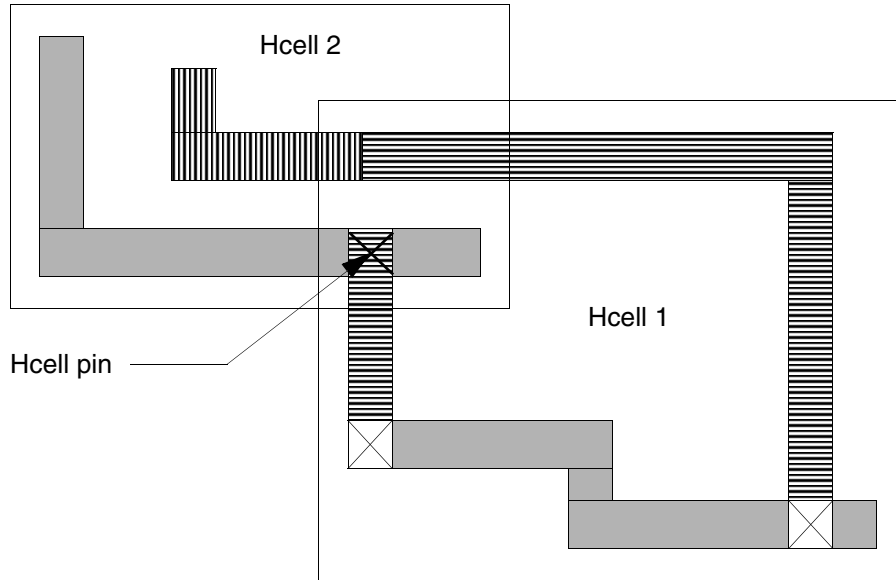
Hcells and the BASE-LAYER

HLVS lets Hcells overlap and connect at Hcell pins in the overlapping area. The composite plane can also have geometries that overlap or cross Hcells and connect at Hcell pins. The netlist must reflect this connectivity to avoid discrepancy errors. Dracula automatically checks devices created by composite plane geometries overlapping Hcells and lists them in the HLVS discrepancy report.

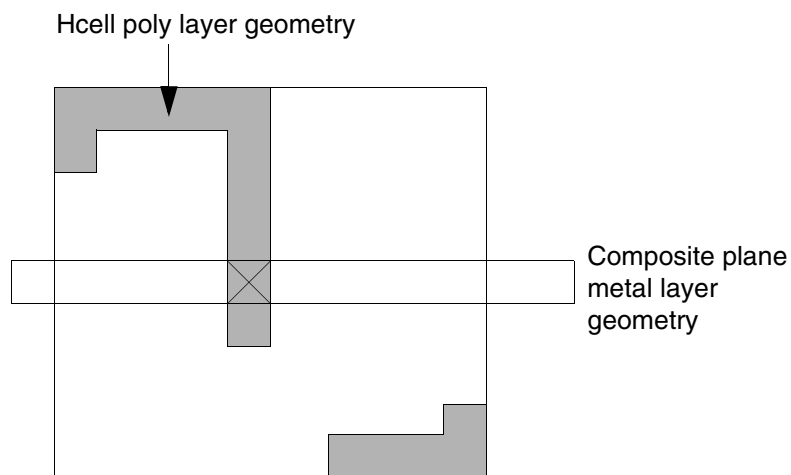
The BASE-LAYER command allows geometries digitized in the composite plane to be considered Hcell geometries. These geometries form devices and indicate connectivity. In a limited sense, the BASE-LAYER command handles geometries that are not identical for different placements. Geometries that are the same for each placement move into the cell. Unique portions of the geometries stay in composite level.

This is important for some gate array types of layouts, where composite plane layers (the base-layers) provide a set of symmetric geometries overlaid by Hcells that parameterize the semicustom circuit with transistors, contacts, and other connections. For details about this functionality, refer to the BASE-LAYER command description in [Chapter 12, “Input-Layer Block Commands.”](#)

Overlapping Hcells with Internal Pin Connections



Overlapping Composite Geometry with Hcell



Hcell Frames

You can specify *Hcell frames* for composite mode HLVS when you use the HCELL or HCELL-FILE commands. The *frame* is a rectangular area in the Hcell that tells Dracula to discard the geometry sections in it, leaving only the geometry sections in the area between the frame and Hcell box. This area along the boundary of the Hcell remains available to let composite mode HLVS determine the external pin connections of the Hcell by matching the geometries being connected and their text (pin name).

GEN-TEXT warns you of pins that are located in the empty region of the frame that cannot be assigned text. You must make the frame border large enough to include these pins. Otherwise, Dracula drops the text. When using Hcell frames and GEN-TEXT, be sure to put text in the proper places, as explained in the “GEN-TEXT Commands” section.

Wire Types

The concept of wire types is similar to virtual wires in flat Dracula except that it provides more flexibility as needed by the hierarchical environment. You can specify *MustJoin* wire types for pins by adding a semicolon (;) and a number after the *signal name* or pin text. For example, *Q;25*. There are two kinds of wire types: *MustJoin* and *Internally-Wired*.

MustJoin Wire Type

Pins with the same text name and different wire types are *MustJoin* wire types. These pins are assumed to be the same node even though they are not connected within their level (Hcell or composite plane). They are joined at some higher level. If the *MustJoin* wire type pins are in an Hcell, a composite mode HLVS verifies a connection between the two pins at the composite level. This *MustJoin* convention is equivalent to the *Virtual Wires* convention (*Q:* and *Q;*), but it is more general and precise in the hierarchical environment. For this reason, do not use virtual wires (:) for cell mode texts.

Internally-Wired Wire Type

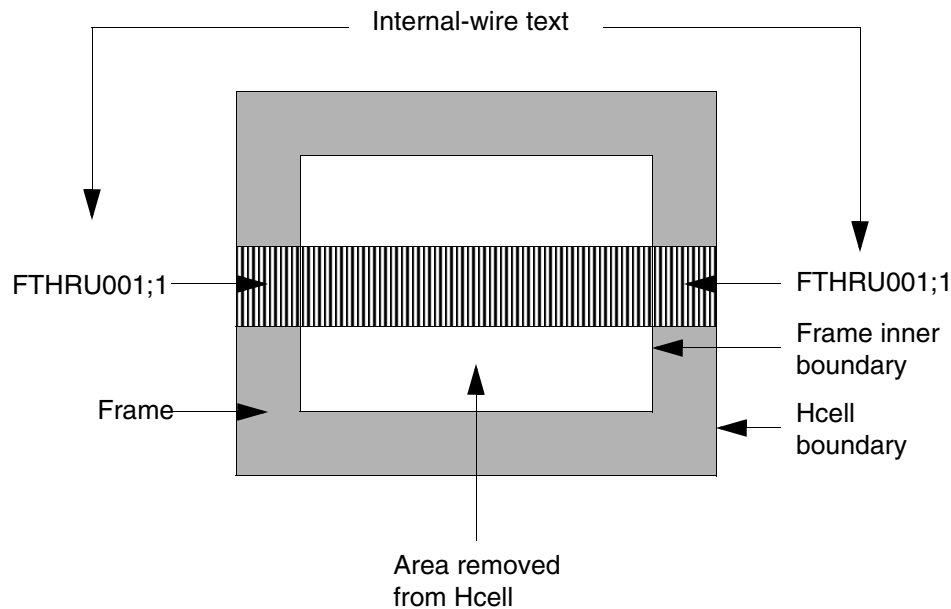
Dracula considers pins with the same text name and the same wire type, such as *Q;3* and *Q;3*, to be internally wired. This means that these wires must be connected within the Hcell in which they are found. A cell mode HLVS run verifies these connections. These pins are particularly important when you use a frame in composite mode to eliminate the layout within a cell, leaving only the pins. If you make two or more external connections to the same geometry at two different places, and if a frame cuts the geometry in the cell, you must label both connect-points of the geometry with an internally-wired wire type. Use a number between 1 and 3000 to name the wire type. Composite mode HLVS then knows that these

Dracula Reference

Comparing Layouts and Schematics (LVS)

two physically disjointed connections correspond to the same pin. Note that a wire name with no type (for example, *Q*) corresponds to a name with wire type of type “0 (zero)” (for example, *Q;0*).

When Dracula assigns a wire-type number, HLVS treats virtual wires as *MustJoin* wire types. Dracula accepts both wire type conventions and virtual wires for composite or flat data. Cell data must use the semicolon (;) convention.



Feedthroughs

Feedthroughs are interconnects that run through an Hcell without making a connection to any device in the Hcell. They route signals through a cell. Feedthroughs do not have a counterpart in the netlist Hcells; rather, they are part of the composite-plane interconnect. When feedthroughs exist, Dracula requires that you identify them in the Hcell layout by texturing them with a label starting with FTHRU followed by one or more digits. This label tells HLVS to ignore the feedthrough in a cell mode HLVS and to use it to connect nodes in a composite mode HLVS. The GEN-TEXT operation can identify feedthroughs and text them.

GEN-TEXT Commands

The GEN-TEXT operation reduces the amount of effort needed to text all Hcell pins as required for a composite mode HLVS operation. GEN-TEXT relies on HLVS to provide texting information for pins not assigned text. Hcells must have the required minimum texting so that a cell mode HLVS can run. Cell mode HLVS with GEN-TEXT enabled extracts text information from the netlist subcircuits and generates an HEDTEXT file containing all Hcell pin text. Subsequent HLVS runs can use this HEDTEXT file as input.

Use GEN-TEXT to save effort in texting Hcell pins, and, if the cell mode run completes successfully, ensures there are no differences between the names in the layout text and the corresponding netlist signal.

Running GEN-TEXT in flat mode is the same as running GEN-TEXT in cell mode with one Hcell. Because you can run GEN-TEXT in any mode, integrate the resulting text files from separate runs into one file for the final composite check.

Running GEN-TEXT in composite mode lets you run a job in composite mode more than once to verify the entire chip. For example, an Hcell (*cell-A*) at the chip level (*cell-chip*) might be so large that a flat or cell mode run might not process the LVS GEN-TEXT for *cell-A*. If you can identify the cells you place inside of *cell-A* as Hcells, then you can

- Run cell mode GEN-TEXT for cells inside of *cell-A*.
- Run composite mode GEN-TEXT on *cell-A*. Combine your GEN-TEXT file with the text created in the separate runs.
- Combine the text of all Hcells in *cell-chip*.
- Run composite mode LVS on *cell-chip*.

Note: You need to be sure that the nodes in *cell-A* to be assigned text with the GEN-TEXT command have geometries on the composite level, or else you need to use the BLACKBOX-FILE command to define the hidden geometries.

You can use the following commands to control the GEN-TEXT operation.

Description Block Commands

GEN-TEXT-FILE = *file-name*

This command starts the GEN-TEXT operation in HLVS. The *file-name* is the text file to be generated.

GEN-TEXT-FLTNODE = YES/NO

This command specifies whether GEN-TEXT should automatically identify feedthroughs and label them with *FTHRUxxxx* text labels.

```
GEN-TEXT-FRAME = xxx MICRONS
```

This command simulates a frame corresponding to *Hcell Frames* to be used in a composite mode HLVS. This simulated frame allows GEN-TEXT to determine whether errors exist in attempting to create text within the frame area.

```
GEN-TEXT-WIRE = YES/NO
```

This command finds pin nodes that must be labeled as *Internally-Wired* because they have been split by a frame (GEN-TEXT-FRAME command) and labels them with an *Internally-Wired* wire type label.

Input-Layer Block Command

```
GEN-TEXT-LAYER = layer...
```

This command specifies layers to be assigned text.

Using Frames with GEN-TEXT

You do not need a frame to run HLVS, but you can use a frame to reduce execution time. To specify a FRAME option on Hcells in a composite mode HLVS and to use GEN-TEXT to generate text, follow these steps:

During the cell mode HLVS run

- Enable GEN-TEXT with the GEN-TEXT-FILE command.
- Use GEN-TEXT-FRAME (FRAME BY in the Hcell table overwrites the GEN-TEXT-FRAME in the associated Hcell).
- Use GEN-TEXT-WIRE = YES.
- If Hcells have feedthroughs, use GEN-TEXT-FLTNODE = YES.
- Ensure that none of the Hcell commands or HCELL-FILE statements specify the FRAME option during the cell mode run.

During the composite mode run

- Provide a HEDTEXT command pointing to the file created by the GEN-TEXT-FILE command during the cell mode run.

- Make all frames specified in Hcell commands or HCELL-FILE statements have the same size as the frame specified in the GEN-TEXT-FRAME command. (Cell mode and composite mode runs can share the HCELL-FILE statements.)

These steps guarantee that if a frame splits a pin assigned text or a feedthrough with two connect locations, Dracula correctly assigns text to the pin or feedthrough using an *Internally-Wired* wire type. For details, refer to the [Wire Types](#) section in this chapter.

Example of GEN-TEXT Commands

The following file is a section of the Description and Input-Layer blocks in a rules file that uses GEN-TEXT with floating-node and wire type options and an 8-micron frame.

```
;
*DESCRIPTION
;
;   HEDTEXT file produced is gen.txt
;
;   GEN-TEXT-FILE = gen.txt
;
;   Use 8 micron frame.
;
;   GEN-TEXT-FRAME = 8 microns
;
;   Generate feedthrough text labels
;
;   GEN-TEXT-FLTNODE = yes
;
;   Distinguish wire types
;
;   GEN-TEXT-WIRE = yes
;
*END
*INPUT-LAYER
;
;   Specify layers for GEN-TEXT to text.
;
;   GEN-TEXT-LAYER = psub nwell poly metal
```

HLVS Error Types

LVS flags overhang geometries of two different cells, or different internal nodes of the same cell, that are connected as potential embedded errors rather than true errors.

Sample HLVS Input Rules File

This is a sample rules file for a PWELL CMOS standard cell.

```
*DESCRIPTION
CHECK-MODE = composite ; $$$ new $$$
```

Dracula Reference

Comparing Layouts and Schematics (LVS)

```
HCELL-FILE = hcell.tab ; $$$ new $$$
PRIMARY = chip
SYSTEM = gds2
INDISK = lvs.db
OUTDISK = lvsout
SCALE = .001 mic
RESOLUTION = .05 mic
PRINTFILE = lvsprt
MODE = exec now
PROGRAM-DIR = /dracula
schematic = LVSLOGIC
*END
;

*INPUT-LAYER
nplus=2
pplus=3
poly =4
met1=6 text=58 ; composite text
poly=23 CTEXT = 18 ; Hcell text $$$ new $$$
met1=24 CTEXT = 20 ; Hcell text $$$ new $$$
cont=5
pwell=1
SUBSTRATE=bulk 54
;
CONNECT-LAYER = nsub pwell psrdrn nsrdrn poly met1
;
*END
;

*OPERATION
HEDTEXT = pinnames.txts ; $$$ new $$$
NOT bulk pwell nsub
AND poly nplus ngate
AND poly pplus pgate
NOT nplus ngate nsrdrn
NOT nplus pgate psrdrn
;
CONNECT met1 nsrdrn BY cont
CONNECT met1 poly BY cont
CONNECT met1 psrdrn BY cont
CONNECT psrdrn pwell BY pwell
CONNECT nsrdrn nsub BY nsub
;
ELEMENT MOS[N] ngate poly nsrdrn pwell
ELEMENT MOS[P] pgate poly psrdrn nsub
;
LVSCHK
*END
```

Inherited Connections for LVS in Dracula Verification

Inherited connections is an extension to the connectivity model that allows you to create global signals and override their names for selected branches of the design hierarchy. This flexibility allows you to use

- Multiple power supplies in a design

- Overridable substrate connections
- Parameterized power and ground symbols

To learn about connectivity and naming conventions for inherited connections and how to add and edit net expressions in a schematic or symbol cellview, refer to the "Inherited Connections" sections in chapter 2 of the *Virtuoso Schematic Composer User Guide*.

Once parameters for a net are specified in the top (upper) level of a schematic or symbol cellview of a design, the instances (in the lower level) inherit the same connectivity from the parameters specified on the top level.

Example

If you have a logic that includes two inverter instances, I1 and I2, then to verify if the netlist in that design has inherited connectivity, follow these steps below.

1. Select the consistent logic (schematic) and layout data. You can run CDLOUT on one logic (schematic view) and correctly pick the corresponding layout.
2. Run CDLOUT on your logic data to get the needed netlist.
3. Check to see if in the top cell in your netlist you have two same instance, for example I1 and I2 that have been assigned different POWER and GROUND connections on the top level.

For example,

gnd and vdd for one instance

a_gnd and N33V for another instance

This means that the two same instance I1 and I2 have followed the "inherited connections" rule and have inherited different POWER and GROUND assignments from the top level respectively.

4. Run LVS for the above netlist and its corresponding stream.
You will get a *.DAT file, for example 6PADS.DAT file.
When you view this .DAT file, you see for example,

GND	1	G	0
VDD	2	P	0
N33V	3	O	0
A_GND	4	O	0
OUT1	5	O	0

Dracula Reference

Comparing Layouts and Schematics (LVS)

INI 6 I 0

Check to see if the GND (PAD 1) and A_GND (PAD 4) are assigned to the same instance respectively.

5. Check the LVSLOGIC file and view its contents and you see that there are two MOS elements (one for I1 and another for I2) in the instance being parsed by LOGLVS. And, their source node (connected to GROUND) is assigned PAD1 and PAD4 respectively.

This means that although this instance did not specify certain parameters for the POWER and GROUD originally, once the parameters are specified in the top (upper) level, the instances (in the lower level) are also inherited from the parameter specified on the top level.

Dracula Reference

Comparing Layouts and Schematics (LVS)

Extracting Electrical Parameters (LPE)

This chapter focuses on the following topics for LPE in Dracula:

- [“LPE Overview”](#) on page 355
- [“LPE Flow Diagram \(continued\)”](#) on page 358
- [“Flexible LPE”](#) on page 372
- [“Selective LPE”](#) on page 388
- [“LPE Examples”](#) on page 390
- [“Hierarchical LPE”](#) on page 401

LPE Overview

Using an integrated circuit (IC) layout, Layout Parameter Extraction (LPE) computes and extracts key electrical parameters, such as

- Node parasitic capacitances
- Defined resistors
- Diode areas and perimeters
- Bipolar device sizes
- Transistor gate sizes

You can use these extracted parameters in Layout vs. Schematic (LVS) consistency checking, in circuit and logic simulations, and in timing verification.

IC designers use circuit and logic simulations to predict product performance. You usually run simulations with estimated device sizes and loadings, and the simulations are not necessarily accurate reflections of layout performance. Substantial tolerances and safety margins are required to avoid design discrepancies. LPE gives you more precise values for physical devices and loadings so that simulations more accurately reflect layout performance.

Dracula Reference

Extracting Electrical Parameters (LPE)

LPE creates both the layout parameter database and the link between layout and schematic compiled by the Layout Versus Schematic Network Compiler (LOGLVS). For information about LOGLVS, refer to [Chapter 5, “Compiling Network Descriptions \(LOGLVS\).”](#)

You can use LPE as a standalone system or with the Design Rules Checker (DRC), Electrical Rules Checker (ERC), or LVS. LPE extracts the width and length of a MOS transistor, the areas and perimeters of a bipolar transistor, the areas and perimeters of a diode (source/drain diffusion), the capacitances of a node (with cross-coupling), and drawn resistance.

You can run LPE with hierarchical Dracula. For more information, refer to the [“Hierarchical LPE” section](#) in this chapter.

In batch mode, LPE reports in a SPICE-compatible format. You can submit the SPICE listing for circuit simulation with only minor editing, such as adding model information. You can select the following LPE reports:

- Program-assigned node/device names/numbers with cross-references to layout labels
- Program-assigned node/device names/numbers with cross-references to schematic I/O's and designators

LPEPRO performs the following functions interactively:

- Probes circuit parameters with your relational thresholds
- Manipulates circuit parameters with your formulas
- Selects your circuit path and outputs all devices along the path in SPICE format so that you can do simulation selectively
- Reproduces the logic gate formation from the layout
- Locates the longest and shortest paths
- Automatically adopts all element types, model names, parameters, units, and other descriptions specified in the LPE rules file

LPE follows the standard Dracula layout texting procedure. If you do not want to backannotate the circuit schematic with the extracted parameters, label only power and ground nodes. For a layout-to-schematic comparison, label the I/O's.

LPEQRY lets you extract devices and parameters associated with specified nets. It is both unnecessary and time consuming to run parameter extraction of all nets on a large, complex chip when you are interested in analysis of only the critical path.

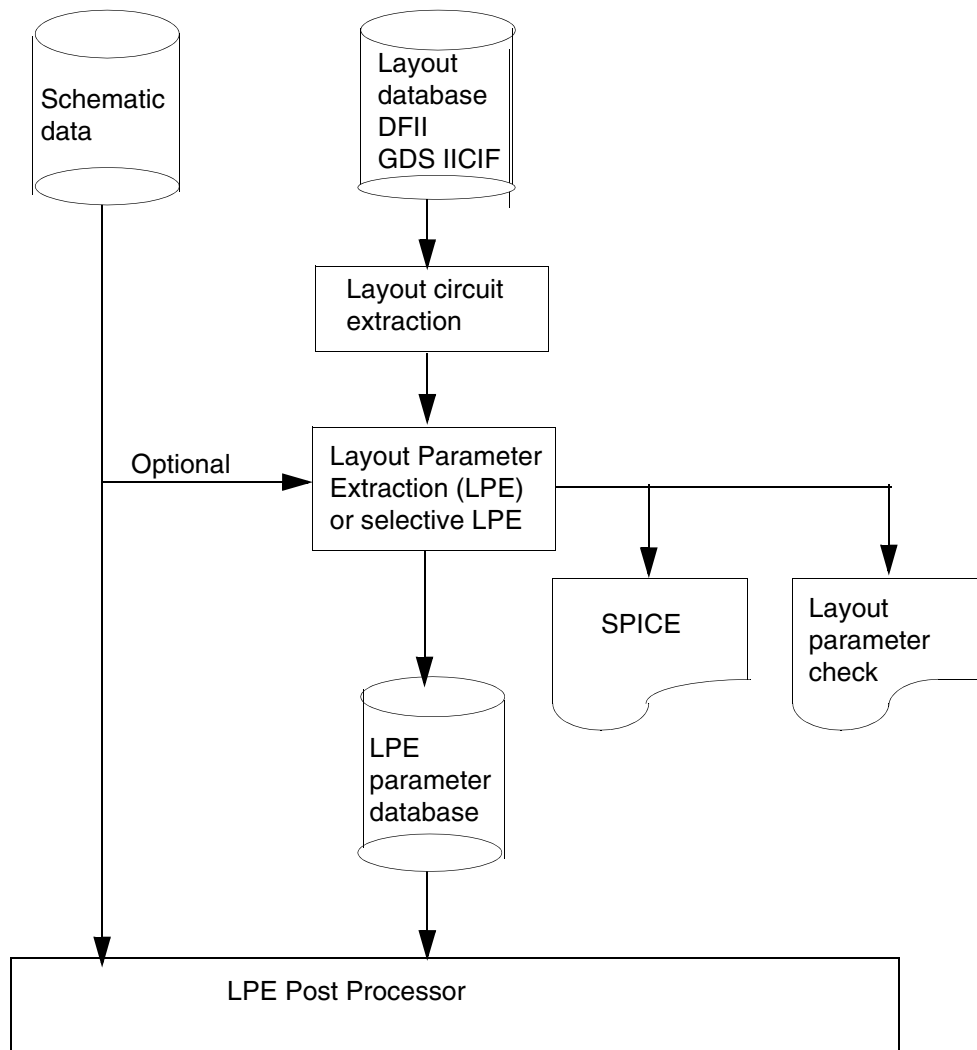
You must compile your schematics with the LOGLVS Network Compiler. LOGLVS accepts HSPICE, SPICE2, and Verilog schematic descriptions.

Dracula Reference

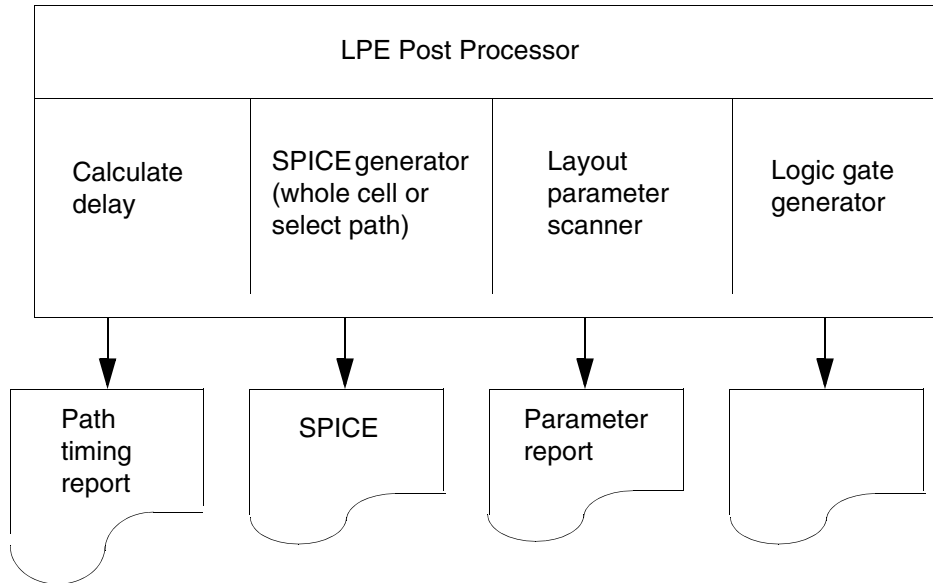
Extracting Electrical Parameters (LPE)

The following figure shows the relationship between LPE and LPEPRO.

LPE Flow Diagram



LPE Flow Diagram (continued)



Extracting SPICE Parameters

The following sections describe standard SPICE parameters and user-specified parameters. LPE extracts for MOS, parasitic diodes, parasitic capacitors, element resistors, and bipolar devices.

MOS Transistors

This section describes how Dracula calculates MOS transistor channel width and length in the graphics database.

Calculations Without Adjustments

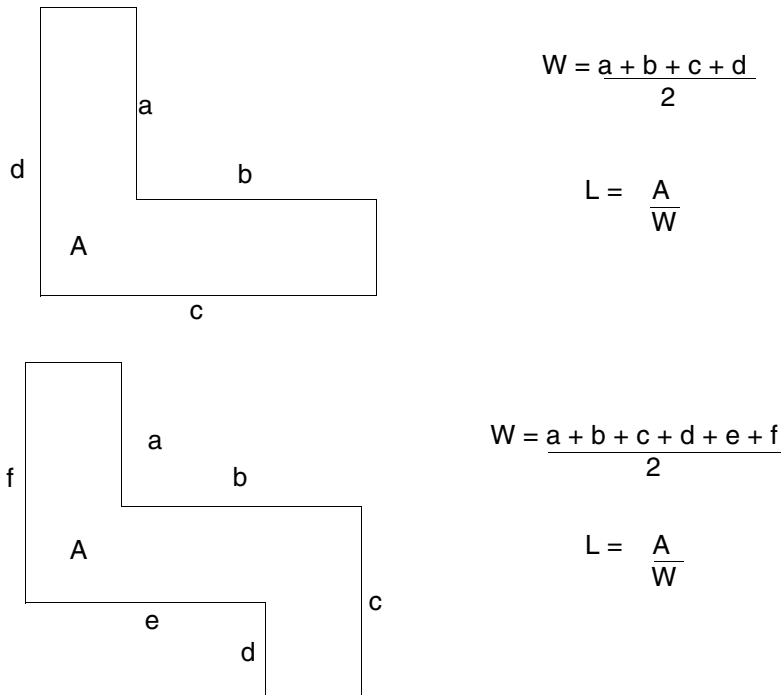
WEFFECT=0 or CORNER=0 (Default)

The transistor widths to the source and drain are the MOS transistor gate perimeters that touch or overlap the diffusion area. The average of the transistor widths at source and drain is the width for the gate area. The transistor length is the ratio of the total gate area to the average of the widths at source and drain.

Dracula Reference

Extracting Electrical Parameters (LPE)

LPE extracts parameters from the drawn geometry. If you can estimate the actual size of a transistor, you must undersize or oversize the gate area to the proper scale before parameter extraction. LPE does not automatically extract the gate to the substrate capacitance. You must define the extraction using the PARASITIC CAP command. You can use the ATTACH command to extract the area and perimeter of the diffusion and attach this information onto the MOS device. The following figure shows the default method for measuring width and length without adjustment.



Where $a+b+c+d+e+f$ represents the perimeter of the active gate area.

To extract MOS source/drain area, perimeter, and output as AS, AD, PS, and PD in SPICE, use the LEXTRACT and ATTACH commands. The following example illustrates this.

```
*DESCRIPTION
...
PARSET TEST AREA PERI
...
*END
...
*OPERATION
...
ELEMENT MOS[N] NCHNL METAL DIFFN OVPWELL;
ELEMENT MOS[P] PCHNL METAL DIFFP NSUB ;
...
LEXTRACT TEST DIFFN BY NODE PDIFFN
```

Dracula Reference

Extracting Electrical Parameters (LPE)

```
LEXTRACT TEST DIFFP BY NODE PDIFFP
ATTACH MOS[N] PDIFFN &
ATTACH MOS[P] PDIFFP

...
LPECHK
LPESELECT[S] MOS &
...
```

Calculations With WEEFFECT or CORNER Specified

You can specify a positive or negative value for WEEFFECT in the LVSCCHK or LPECHK command. The same restriction is true for CORNER in the LPESELECT command. When you encounter turns or corners, Dracula calculates the MOS channel width/length as follows:

For a pulldown device where $W > L$, each 15-degree turn in the transistor channel adds or subtracts $1/6$ of WEEFFECT/CORNER times the length, L , from the transistor default width, W . For a load device ($L \geq W$), the channel length is modified. The following examples illustrate these calculations (CORNER can be substituted for WEEFFECT).

- For a pulldown device (L is unmodified)

$$W_{\text{actual}} = W_{\text{init}} - (\text{WEEFFECT} * I * L) ; \text{ if } W_{\text{init}} \text{ is greater than } L$$

where

W_{actual} = the actual width of the gate

W_{init} = the initial width of the gate. This is always calculated by Dracula.

WEEFFECT = a compensation coefficient you supply to Dracula to correctly calculate the actual width/length of a transistor gate.

I = the total of a device's internal angles (in units of 90 degrees)

L = the length of the gate which is equal to the total area of the gate divided by the initial width of the gate (e.g. $\text{Area}/W_{\text{init}}$)

- For a load device (W is unmodified)

$$L_{\text{actual}} = L_{\text{init}} - (\text{WEEFFECT} * I * W) ; \text{ if } L_{\text{init}} \geq W$$

where

L_{actual} = the actual length of the gate

L_{init} = the length of the gate which is equal to the total area of the gate divided by the width of the gate (e.g. Area/W)

Dracula Reference

Extracting Electrical Parameters (LPE)

WEFFECT = a compensation coefficient you supply to Dracula to correctly calculate the actual width/length of a transistor gate.

I = the total of a device's internal angles (in units of 90 degrees, with each internal angle rounded to the nearest 15 degrees)

W = the width of the gate. This is always calculated for you by Dracula.

- For parallel MOS pulldown transistors (of the same type):

Transistor 1: $W_{\text{actual1}}, L_1 = A_1 / W_1$

Transistor 2: $W_{\text{actual2}}, L_2 = A_2 / W_2$

Total area: $A = A_1 + A_2$

Total ratio: $C = W_{\{\text{actual1}\}}/L_1 + W_{\{\text{actual2}\}}/L_2$

$W_{\text{combined}} = \sqrt{A \cdot C}$

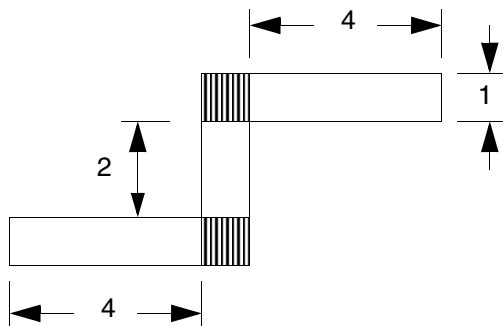
$L_{\text{combined}} = \sqrt{\frac{A}{C}}$

Example 1

$I/90 = 2$ [90 + 90 = 180 degrees]

For a pulldown device:
WEFFECT= 0.5, W = 12
L = 1, and $W_{\text{actual}} = 11$

For a load device:
WEFFECT= 0.5, W = 1
L = 12, and $L_{\text{actual}} = 11$



Example 2

$I/90 = 1$ [$45 + 45 = 90$ degrees]

For a pulldown device:

WEFFECT = 0.0, $W = 16$

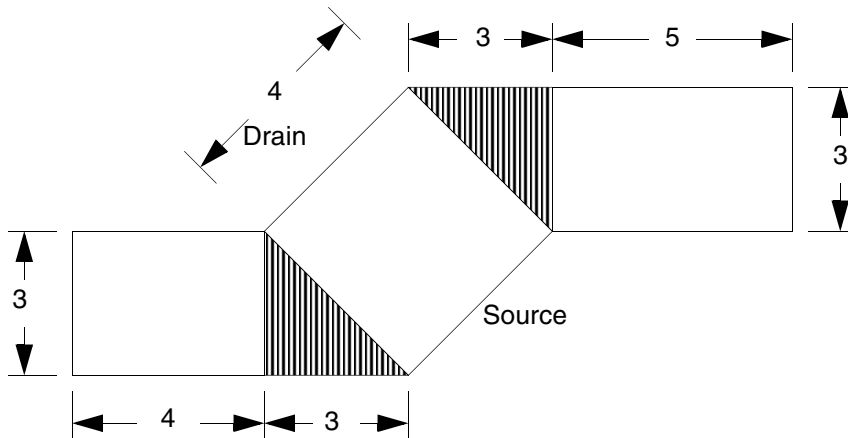
$L = 3.31$, and $W_{\text{actual}} = 16$

where WEFFECT = CORNER.

For another pulldown device:

WEFFECT = 1.0, $W = 16$

$L = 3.31$, and $W_{\text{actual}} = 12.69$



Parasitic Diode Parameters

LPE extracts the parameters of a positive-negative (P-N) junction diode. The source/drain diffusion area and the substrate/well on which it is diffused make up the two diode terminals. The polarities of the terminals are positive if P-doped and negative if N-doped.

You can extract as many as six types of diode parameters: A1, P1, A2, P2, A3, and P3. The choice of A2,P2 and A3,P3 depends on whether you specify the second and third argument layers in the PARASITIC DIO command. These two layers cannot overlap when both are needed to define a diode. For more information about the PARASITIC DIO or DIODESEQ commands, refer to the "Description Block Commands" and "Operation Block Commands" chapters.

To reduce the number of diodes for simulation, LPE lumps the parameters of the same diode type at the same node into one parameter before reporting them.

Parasitic Capacitor Parameters

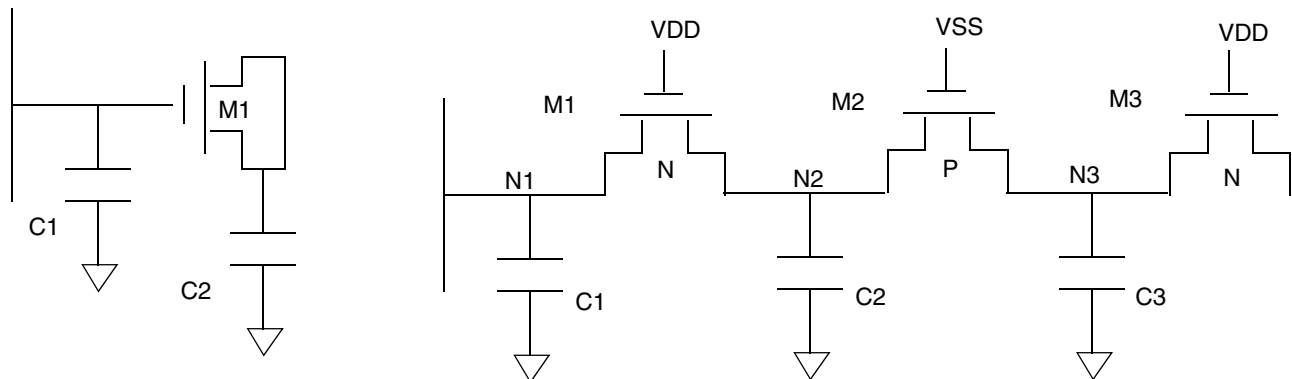
LPE defines most kinds of capacitance in any process technology with combinations of logical and sizing commands. After the connections of a circuit are established with CONNECT commands, the node numbers automatically pass from the first argument layer to the output layer on each logical AND/NOT command. The two layers that define a parasitic capacitor determine the connections between the two terminals of a capacitor.

LPE-extracted capacitance is always coupled from one node to another and skips the capacitors with both terminals at the same node. LPE reports the combined load capacitances of a node in lump sum format as the default unless you specify otherwise with the LPESELECT[C] command.

In addition to capacitance values, the parameter database includes the area and perimeter of the capacitor. You compute capacitance values by specifying the area and perimeter coefficients in the ATTRIBUTE CAP command. For details about the ATTRIBUTE CAP command, refer to [“ATTRIBUTE CAP”](#) on page 884.

Device Filtering

With device filtering on, if a filtered device has one terminal connected to one net of the circuit after filtering, the capacitance of the filtered device, which is related to the net, is taken into consideration. All parasitic capacitance related to the merged nodes is added to the resultant node. The following illustration shows how LPE handles filtered devices.



M1 is the filtered device.
C1 is taken into consideration.
C2 is not taken into consideration.

M1, M2, and M3 are the filtered devices and cause N1, N2, and N3 to short together.
The reported capacitance for the resultant node is equal to $C1+C2+C3$.

Element Resistor Parameters

To define the resistor device layer and terminals, use the ELEMENT RES command. To assign sheet resistance for a resistor device defined by the ELEMENT RES command, use the PARAMETER RES command.

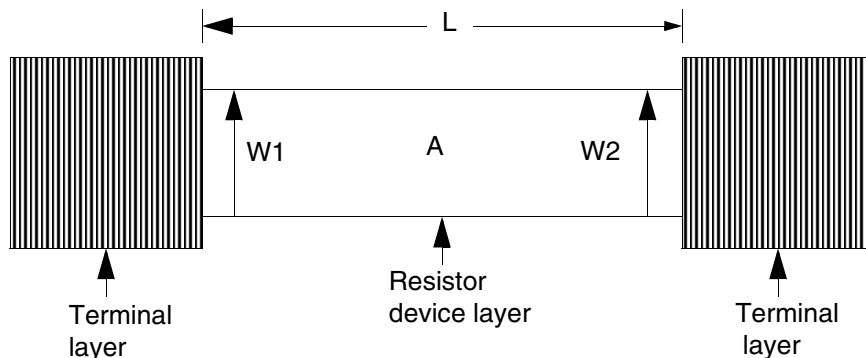
To define effective width, device size extraction uses the device layer where it touches terminals $W1$ and $W2$. Device size extraction uses area and effective width to compute the effective length of the resistor. You must properly size or logically create the terminal layer to ensure the correct resistance value.

To specify the scale factor of unit values to resistor value, use the UNIT command in the Description block. If you do not specify the unit name, LPE lists the parameter in natural units.

Example of a Typical Resistor

This example shows the equation for a typical resistor.

```
*DESCRIPTION
.
UNIT = capacitance,pf resistance,k
.
*END
```



$$W = (W1 + W2) / 2 \text{ and } L = A / W$$

where

W is the effective width of the resistor.

L is the effective length of the resistor.

A is the total area of the resistor device layer.

$$\text{Resistance, } R = (\text{value}) * (L / W)$$

Dracula Reference

Extracting Electrical Parameters (LPE)

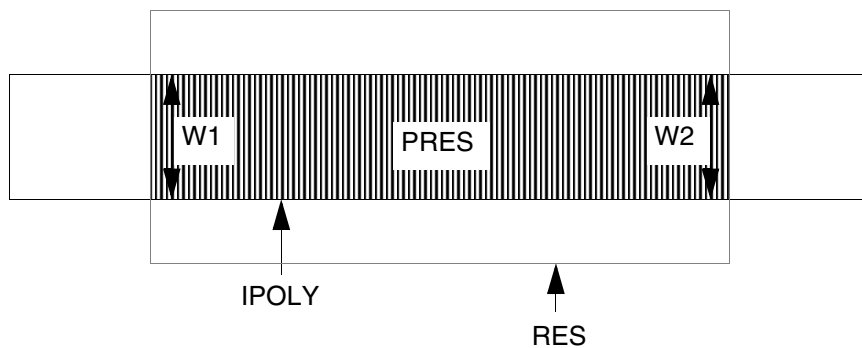
The value is the sheet resistance per square specified in the `PARAMETER RES` command.

This equation is applicable for a bent resistor with corners. The area adjusts the resistor length, which becomes a center line length.

Example of a Typical Polysilicon Resistor

This is an example of how you can use an extra digitized layer to identify an element. In this example, *poly*, the conductor layer created from the *ipoly* layer (complete polysilicon layer) and the *res* layer, acts as the resistor terminal. To extract a resistance value, you must specify the `PARAMETER` command.

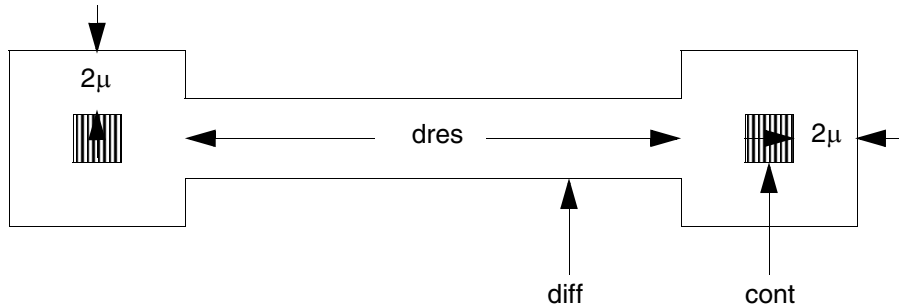
The *ipoly* layer is polysilicon, and the *res* layer is a digitized resistor layer.



```
*DESCRIPTION
.
UNIT = resistance,k
.
*END
*OPERATION
.
AND ipoly res pres      ; Device layer
NOT ipoly pres poly     ; Poly-si interconnect &
                        ; resistor terminals
.
ELEMENT RES[P] pres poly
PARAMETER RES[P] 200
.
LPECHK
LPESELECT res &
.
*END
```

Example of Diffusion Resistors

In this example, the *diff* layer is the complete diffusion layer, the *dres* layer is the diffusion resistor, and the *cont* layer is the metal-diffusion contact.



Note that the *diff* layer is a logical and/or select-generated layer. You must resize the *cont* layer to generate the “touch” of the device layer, *dres*.

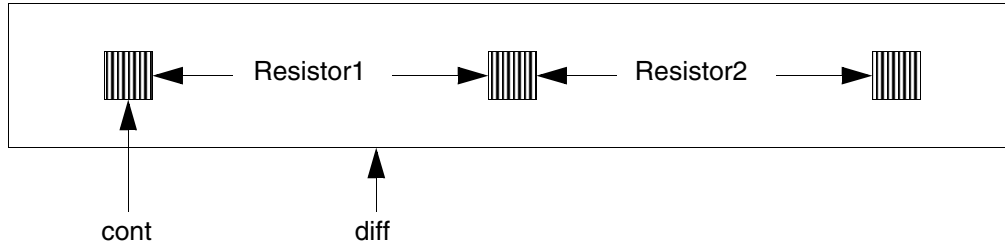
```
*DESCRIPTION
.
UNIT = resistance, ohm      ;In this case, the UNIT
                             ;command is not necessary
.
*END
*OPERATION
.
SIZE cont BY 2 ocont        ;Oversize CONT
NOT diff ocont dres         ;Device layer for resistor
NOT diff dres srcdrn        ;Source drain layer without
                             ;contact terminals
.
CONNECT metal srcdrn BY cont
.
.
ELEMENT RES[D] dres srcdrn
PARAMETER RES[D] 15
.
LPECHK
LPESELECT res &
.
*END
```

For more than two terminal-drawn resistors (Y-shaped, T-shaped, and so forth), you must generate an intermediate node (contact terminal). If you do not generate an intermediate node, you get an undefined resistor device (*?RES*).

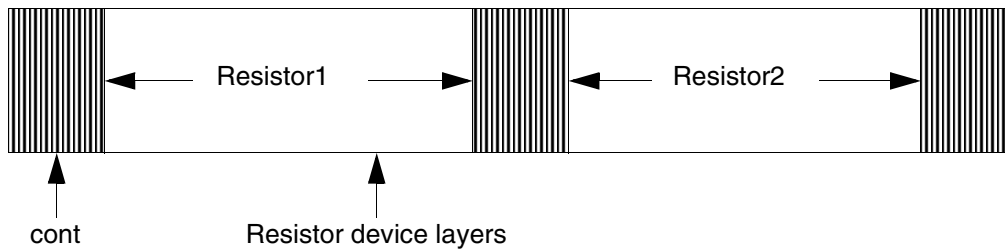
Dracula Reference

Extracting Electrical Parameters (LPE)

The following figure shows diffusion resistors in a series where the resistor bodies are not separated by contacts.



You must transform the process shown above by SIZE, LOGICAL, or SELECT operations. The following figure shows how to correct the extraction of the two resistors.

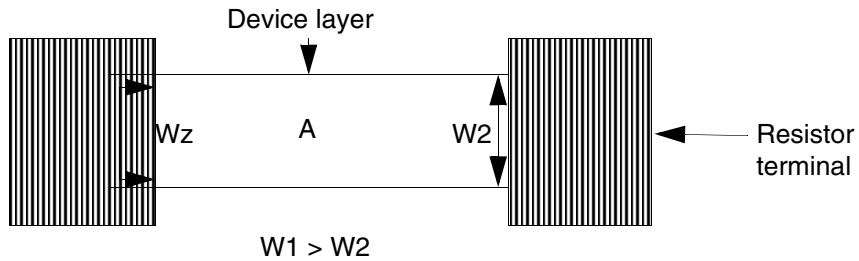
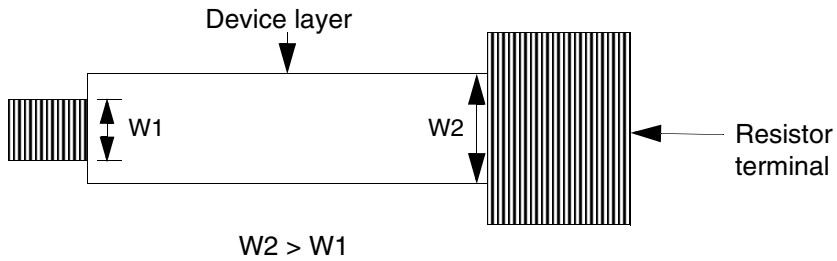
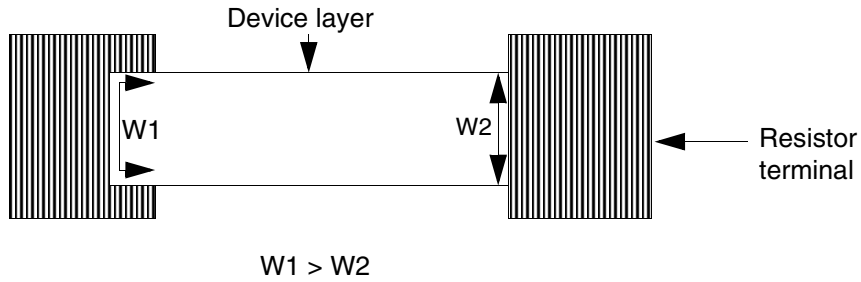


Examples of Problem Resistors

In the examples in the preceding section, the resistor terminals precisely touched the resistor device layer so that $W1$ and $W2$ measured the same. If the resistor terminals do not precisely touch the device layer, the calculated W and L might be slightly different.

Example of Width Not Measured Precisely

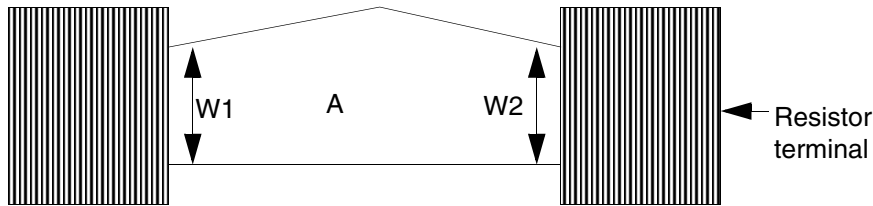
The following example shows where resistors width is not measured precisely.



A is the total area of the device layer.

Example of Length Not Measured Precisely

This figure shows an example of resistors where length is not measured precisely.



Area modifies the length, L .

LPE and LVS extract resistor devices and device sizes. The same resistor equation for W and L applies, but the extracted resistor device sizes vary depending on the interrelationship of the resistor device layer and the resistor terminals (precise touch expected).

Bipolar Transistor Area and Perimeter

This section shows you how to extract and measure parameters for bipolar transistors. LPE extracts bipolar device parameters and creates a CDL/SPICE netlist. The DEVTAG and LEXTRACT operation commands handle bipolar devices. With these commands, you can extract parameters from other associated layers or geometries of the same device (multiple collector or emitter bipolar devices). For more information about these commands, refer to [Chapter 12, “Input-Layer Block Commands.”](#)

Using Bipolar LPE

1. To define bipolar devices, use the ELEMENT BJT command.
2. To tag device numbers to layers that are not device layers but from which you want to extract parameters, use the DEVTAG command.
3. To specify the parameters (area and/or perimeter) you want to extract and the layers from which you want to extract them, use the LEXTRACT command.

The program supports the following parameters for bipolar transistors:

- a. Area and perimeter of emitters
- b. Area and perimeter of collectors
- c. Area and perimeter of base

Dracula Reference

Extracting Electrical Parameters (LPE)

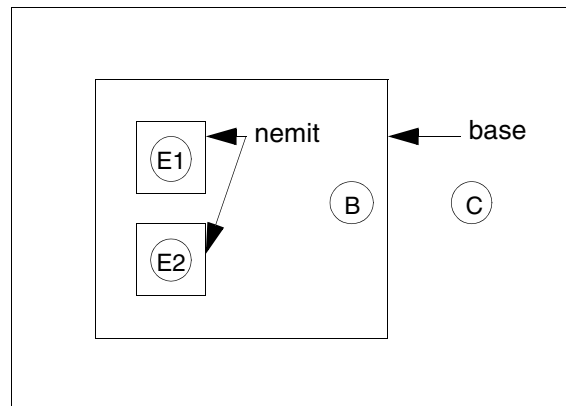
If you do not specify the LEXTRACT command, LPE extracts only area and perimeter from the device layer you specify in the ELEMENT BJT command. The default assumes the device layer is the emitter; the netlist reports the area and perimeter as “EA” and “EP.” If you want the area and perimeter of a terminal other than the device layer as output, use the DEVTAG and LEXTRACT commands.

4. To define the parameter set (area and/or perimeter) to be extracted, use the PARSET command.
5. To specify model names for bipolar transistors in the Description block, use the MODEL command.
6. To backannotate to schematics and to create the CDL/SPICE netlist, use the LPECHK and LPESELECT commands.

Example of Vertical *npn*

In this example, Dracula extracts the emitter area and the perimeter by default. The *nemit* layer is the device layer, *buried* is the device collector terminal, and *base* is the device base terminal. The *nemit* layer is also the device emitter terminal.

```
* DESCRIPTION
.
MODEL=BJT[n],npn
SCHEMATIC=lvslogic
.
* END
* OPERATION
.
ELEMENT BJT[n] nemit buried base nemit
.
LPECHK
.
LPESELECT[s] bjt OUTPUT spice
* END
```



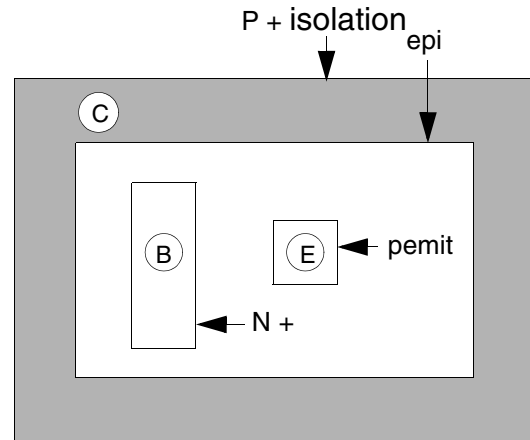
Dracula Reference

Extracting Electrical Parameters (LPE)

Example of Vertical *pnp*

In this example, Dracula extracts emitter areas, base areas, and perimeters and includes them in the *bjtv* parameter file. The *pemit* layer is the device layer, *psub* is the device collector terminal, *epi* is the device base terminal, and *pemit* is the device emitter terminal.

```
* DESCRIPTION
.
PARSET peb area peri ba bp ea ep
MODEL=BJT[vp],vpnp
SCHEMATIC=lvsllogic
.
* END
* OPERATION
.
ELEMENT BJT[vp] pemit psub epi pemit
.
DEVTAG BJT[vp] epi epil
LEXTRACT peb pemit BY BJT[vp] &
LEXTRACT peb epil BY BJT[vp]
.
LPECHK
LPESELECT[s] bjt OUTPUT spice
* END
```



Example of Lateral *pnp* with Possible Multiple Collector

In this example, Dracula extracts the emitter areas, collector areas, and perimeters and includes them in the *bjtl* parameter file. To tag the device numbers, Dracula must go back to *emit* (the true emitter) through *ovemit* (the intermediate layer oversized emitter). Dracula performs extraction through the *temit* file, which has device numbers linked by the DEVTAG command. No tagging is necessary for the *coll* layer because it is the device layer.

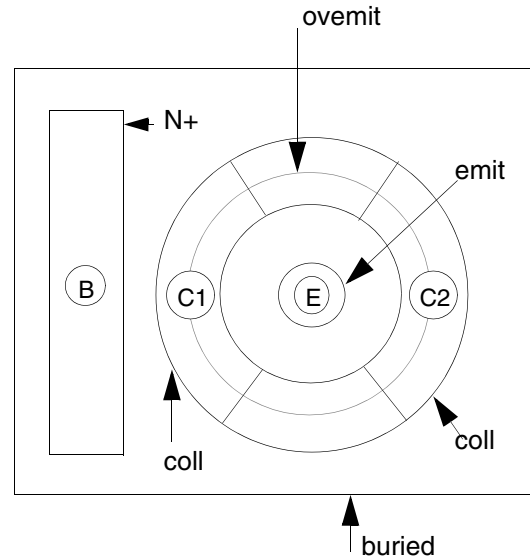
In this example, *coll* is the device layer and the device collector terminal, *buried* is the device base terminal, and *ovemit* is the device oversized emitter terminal. The *emit* layer is the true

Dracula Reference

Extracting Electrical Parameters (LPE)

size emitter; *emit1* and *temit* are files created by DEVTAG that contain device number information.

```
* DESCRIPTION
.
PARSET pec area peri ca cp ea ep
MODEL=BJT[lp],mpnp
SCHEMATIC=lvlogic
* END
* OPERATION
.
ELEMENT BJT[lp] coll coll buried overmit
.
DEVTAG BJT[lp] overmit emit1
DEVTAG[L] emit1 emit temit
LEXTRACT pec temit BY BJT[lp] bjt1p &
LEXTRACT pec col BY BJT [lp]
.
LPECHK
LPESELECT[S] bjt OUTPUT spice
* END
```



Note: The combined use of the LPESELECT BJT command and the LEXTRACT command generates a bipolar SPICE file. The file has the following format:

CA = area of the collector	CP = perimeter of the collector
BA = area of the base	BP = perimeter of the base
EA = area of the emitter	EP = perimeter of the emitter

Remember that the dollar sign (\$) is a SPICE comment delimiter.

```
Q1 out1 in1 gnd npn $ EA=100 EP=40.
Q2 vcc xfr1 out6 vpnp $ BA=900 BP=120 EA=100 EP=40
Q3 cout1 in3 vcc mpnp $ CA=400 CP=80 EA=400 EP=80
Q4 cout2 in3 vcc mpnp $ CA=400 CP=80 EA=400 EP=80
```

Flexible LPE

LPE supports standard CDL or SPICE parameters for devices. Flexible LPE lets you define parameters and extract them by specifying your equation. Flexible LPE supports all CDL or SPICE parameters (reserved parameter keywords), and you can output these parameters with the LPESELECT command in the SPICE file generated by LPE.

Dracula Reference

Extracting Electrical Parameters (LPE)

Flexible LPE has three types of parameters: geometric primitives, reserved parameter keywords, and user-defined parameters.

Geometric Primitives

The primitives defined below refer to the relation of a layer to itself and to other layers. To extract values from the relation of the specified layers to *layer-a*, use the LEXTRACT and PARSET commands.

ANG	Total internal angle
AREA	Area
PERI	Perimeter
W1	Overlapped perimeter to node 1 of second layer
W2	Overlapped perimeter to node 2 of second layer
OVAR	Overlapped area to second layer
OVPR	Overlapped perimeter to second layer
WIDT	Distance that the layer runs parallel
DEPT	Distance between the parallel geometries
TPR	Perimeter of geometries that cause fringe effect

Reserved Parameter Keywords

Reserved parameter keywords are for parameter values for devices you usually use for CDL or SPICE simulations. If you specify LEXTRACT for a device, you must specify an equation for each of these parameters in the PARSET command list. For more information about the PARSET list, refer to [Chapter 13, “Operation Block Commands.”](#) When extracting parameters for bipolar devices only, you can specify LEXTRACT without the EQUATION command.

W	MOS channel width
L	MOS channel length
A1	DIO area
P1	DIO perimeter
A2	DIO area
P2	DIO perimeter
A3	DIO area
P3	DIO perimeter
C	CAP value
R	RES value
CA	BJT collector area
CP	BJT collector perimeter
BA	BJT base area BP BJT base perimeter
EA	BJT emitter area
EP	BJT emitter perimeter
K	Fringe capacitance attribute used for piece-wise fringe capacitance
CLL	Sum of fringe effect related to TPR

User-Defined Parameters

Any names you define can have a maximum of four characters. The first character must be from A to Z. The remaining characters can be A-Z and 0-9. If you specify an LEXTRACT command for a device, you must specify an EQUATION command for each parameter in the PARSET list. When you do not specify LEXTRACT for the device type, the system does the default extraction.

Dracula Reference

Extracting Electrical Parameters (LPE)

Reserved Parameter Set (PARSET) Keywords

The reserved PARSET keywords are MOSD, DIOD, DIO2, DIO3, CAPD, CAPF, BJTD, RESD, LDDD, MOSS, CAPO, and BOXD. Do not use your own defined parameters with these reserved PARSET keywords.

The following are examples of default LPE parameter extraction for ELEMENT and PARASITIC devices:

```
ELEMENT MOS
ELEMENT LDD
ELEMENT DIO
PARASITIC DIO
ELEMENT CAP
PARASITIC CAP overlapped capacitances
PARASITIC CAP fringe capacitances
PARASITIC CAP overlapped capacitance with fringe effect
FRINGE CAP fringe capacitances including different layers
ELEMENT RES
PARASITIC RES
ELEMENT BJT
ELEMENT BOX
```

The LEXTRACT output parameter file name is enclosed in parentheses because the specification of the file name is illegal, but the name is generated internally by LPE.

ELEMENT MOS

LEXTRACT parameter set	MOSD
Layers	device layer MOSX**G Source and drain layer
Output file	7MOSX**M
EQUATION	$W=(W1+W2)/2$ $L=AREA/W$

The default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET MOSD AREA PERI ANG W1 W2 W L
.
```

Dracula Reference

Extracting Electrical Parameters (LPE)

```
*END
*OPERATION
.
.
ELEMENT MOS[N] NGATE POLY NSD PWELL
LEXTRACT MOSD NGATE NSD BY MOS[N] (7MOSXN8M) &
EQUATION W=(W1+W2)/2 &
EQUATION L=AREA/W
```

ELEMENT LDD

LEXTRACT parameter set	LDDD
Layers	Device layer LDDX**G Source and drain layer
Output file	7LDDX**L
EQUATION	$W=(W1+W2)/2$ $L=AREA/W$

The default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET LDDD AREA OVPR W L
.
.
*END
*OPERATION
.
.
ELEMENT LDD[A2] PCHNL METAL DIFFP DIFFN NSUB
LEXTRACT LDDD PCHNL DIFFP BY LDD[A2] (7LDDXA2L) &
EQUATION W=OVPR &
LEXTRACT LDDD PCHNL DIFFNBY LDD[A2] &
EQUATION W=(OVPR+W)/2 &
EQUATION L=AREA/W
```

ELEMENT DIO

LEXTRACT parameter set	DIOD
Output file	7DIOX **D
EQUATION	$A1=AREA$ $P1=PERI$

The default extraction is equivalent to the following:

Dracula Reference

Extracting Electrical Parameters (LPE)

```
*DESCRIPTION
PARSET DIOD AREA PERI A1 P1
.
.
*END
*OPERATION
.
.
ELEMENT DIO[N] NDIO PWELL NDIO
LEXTRACT DIOD NDIO BY DIO[N] (7DIOXN8D) &
EQUATION A1=AREA &
EQUATION P1=PERI
```

PARASITIC DIO

LEXTRACT parameter set	DIOD, DIO2, DIO3 for three, four, and five layers, respectively, on DIO
Layers	Device layer DIOP**G, fourth layer Device layer DIOP**G, fifth layer
Output file	7DIOP**D
EQUATION	A1=AREA-OVAR P1=PERI-OVPR A2=OVAR if fourth layer P2=OVPR A3=OVAR if fifth layer P3=OVPR A1=A1-A3 P1=P1-P3

If there is no fourth or fifth layer, the default extraction is the same as the default for ELEMENT DIO. If there is a fourth layer, the default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET DIO2 AREA PERI OVAR OVPR A1 P1 A2 P2
*END
*OPERATION
PARASITIC DIO[N] NDIO PWELL NDIO NGATE
LEXTRACT DIO2 NDIO NGATE BY DIO[N] (7DIOPN8D) &
EQUATION A1=AREA-OVAR &
EQUATION P1=PERI-OVPR &
EQUATION A2=OVAR &
EQUATION P2=OVPR
```

If there are fourth and fifth layers, the default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET DIO3 AREA PERI OVAR OVPR A1 P1 A2 P2 A3 P3
*END
*OPERATION
PARASITIC DIO[N] NDIO PWELL NDIO NGATE CONT
LEXTRACT DIO3 NDIO NGATE BY DIO[N] &
EQUATION A1=AREA-OVAR &
EQUATION P1=PERI-OVPR &
```

Dracula Reference

Extracting Electrical Parameters (LPE)

```
EQUATION A2=OVAR &
EQUATION P2=OVPR &
LEXTRACT DIO3 NDIO CONT BY DIO[N] (7DIOPN8D) &
EQUATION A3=OVAR &
EQUATION P3=OVPR &
EQUATION A1=A1-A3 &
EQUATION P1=P1-P3
```

ELEMENT CAP

LEXTRACT parameter set	CAPD
Layers	Device layer CAPX**G
Output file	7CAPX**C
EQUATION	C=attribute1*AREA +attribute2*PERI

The default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET CAPD AREA PERI C
.
.
*END
*OPERATION
.
.
ELEMENT CAP[A] MPOLY MET POLY
PARAMETER CAP[A] 0.00005 1.0E-4
LEXTRACT CAPD MPOLY BY CAP[A] (7CAPXA8C) &
EQUATION C=0.00005*AREA+1.0E-4*PERI
```

PARASITIC CAP

Overlapped capacitances:

LEXTRACT parameter set	CAPD
Layers	Device layer CAPP**G
Output file	7CAPP**C
EQUATION	C=attribute1*AREA +attribute2*PERI

The default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET CAPD AREA PERI C
```

Dracula Reference

Extracting Electrical Parameters (LPE)

```
.
*END
*OPERATION
.
.
PARASITIC CAP[A] MPOLY MET POLY
ATTRIBUTE CAP[A] 0.00005 1.0E-4
LEXTRACT CAPD MPOLY BY CAP[A] (7CAPPA8C) &
EQUATION C= 0.00005*AREA+1.0E-4*PERI
```

Overlapped capacitance with consideration of fringe effect:

LEXTRACT parameter set	CAPO
Layers	Overlap and fringe layers
Output file	7CAPP**C
EQUATION	C= areaCoeff*AREA + perimCoeff*(PERI-TPR) + CLL

The default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET CAPO AREA PERI TPR CLL C
.
.
*END
*OPERATION
.
.
PARASITIC CAP[O] M12 MET1 MET2
ATTRIBUTE CAP[O] 0.5 0.02 0.5 3.9
ATTRIBUTE CAP[O] 0.5 0.02 0.8 1.9
ATTRIBUTE CAP[O] 0.5 0.02 1.5 0.8
LEXTRACT CAPO M12 BY CAP[O] 7CAPPO8C &
EQUATION C= 0.5*AREA+ 0.02*(PERI-TPR)+ CLL
```

PARASITIC CAP

Fringe capacitances:

LEXTRACT parameter set	CAPF
Layers	Fringe layer
Output file	7CAPP**C
EQUATION	C=K*WIDT/DEPT

Dracula Reference

Extracting Electrical Parameters (LPE)

The default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET CAPF DEPT WIDT C K
.
.
*END
*OPERATION
.
.
PARASITIC CAP[A] METAL METAL METAL
ATTRIBUTE CAP[A] 0.5 0.02
LEEXTRACT CAPF METAL BY CAP[A] 7CAPPA8C &
EQUATION C=K*WIDT/DEPT
```

FRINGE CAP

Fringe capacitances:

LEEXTRACT parameter set	CAPF
Layers	Fringe layer
Output file	7CAPPA8C
EQUATION	C=K*WIDT/DEPT

The default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET CAPF DEPT WIDT C K
.
.
*END
*OPERATION
.
.
FRINGE CAP[A] METAL METAL
ATTRIBUTE CAP[A] 0.5 0.02
LEEXTRACT CAPF METAL BY CAP[A] 7CAPPA8C &
EQUATION C=K*WIDT/DEPT
```

Dracula Reference

Extracting Electrical Parameters (LPE)

ELEMENT RES

LEXTRACT parameter set	RESD
Layers	Device layer RESX**G
Output file	7RESX**R
EQUATION	$W=(W1+W2)/2$ $L=AREA/W$ $R=attribute*L/W$

The default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET RESD AREA PERI ANG W1 W2 W L R
.
.
*END
*OPERATION
.
.
PARASITIC RES[A] RDEV RTRM
ATTRIBUTE RES[A] 0.5
LEXTRACT RESD RDEV RTRM BY RES[A] (7RESPA8R) &
EQUATION W=(W1+W2)/2 &
EQUATION L=AREA/W &
EQUATION R=0.5*L/W
```

ELEMENT BJT

LEXTRACT parameter set	BJTD
Layers	Device layer BJTX**G
Output file	7BJTX**B
EQUATION	$EA=AREA$ $EP=PERI$

The default extraction is equivalent to the following:

```
*DESCRIPTION
PARSET BJTD AREA PERI EA EP
.
.
*END
*OPERATION
.
.
ELEMENT BJT[NP] EMIT COLL BASE EMIT
```

Dracula Reference

Extracting Electrical Parameters (LPE)

```
LEXTRACT BJTD EMIT BY BJT[NP] (7BJTXNPB) &  
EQUATION EA=AREA &  
EQUATION EP=PERI
```

Example of Parameter Extraction

The following are two examples of the extraction of your user-defined parameters.

To extract MOS widths and lengths and use a formula to adjust for bent gates, follow this procedure:

1. Create a rules file for a CMOS LPE run.
2. Define parameter set (*PA1*) to be extracted by the PARSET command.

```
AREA    =   Area of layer1  
OVPR    =   Perimeter of layer1 overlapping layer2  
ANG     =   Total internal angle of layer1 device  
W       =   Channel width  
L       =   Channel length
```

3. Define the ELEMENT MOS for N-channel devices.
4. Extract parameters in *PA1* parameter set for MOS[N] with the LEXTRACT command. The EQUATION command computes *W* and *L*. The saved parameters are in the *npar.dat* file.
5. To generate a SPICE file that contains the MOS-computed *W* and *L* values, use the LPESELECT command.

```
*DESCRIPTION  
PARSET PA1 AREA OVPR ANG W L  
.  
.  
*END  
*OPERATION  
.  
.  
ELEMENT MOS[N] NGATE POLY NSDRN PWEEL  
LEXTRACT PA1 NGATE NSDRN BY MOS[N] NPAR &  
EQUATION W=OVPR/2-0.1*ANG/90 &  
EQUATION L=2*AREA/OVPR-0.1*ANG/90  
.  
.  
LPESELECT MOS[N] OUTPUT SPICE  
*END
```

Dracula Reference

Extracting Electrical Parameters (LPE)

To extract *met2* to *bulk* capacitances with the condition of removing the edge blocked by *met1* for fringe capacitances, follow this procedure:

1. Define parameter set (CCCC) to be extracted by the PARSET command.

AREA = Area of layer1
PERI = Perimeter of layer1
OVPR = Perimeter of layer1 overlapping layer2
C = Capacitance value

2. Define PARASITIC CAP[m].
3. Extract parameters in CCCC parameter set for CAP[M] with the LEXTRACT command.
4. To compute C and save parameters in the *mcap.dat* file, use the EQUATION command.
5. To generate a SPICE file that contains the capacitor and its computed values, use the LPESELECT command.

```
*DESCRIPTION
PARSET CCCC AREA PERI OVPR C
.
.
*END
*OPERATION
.
.
CONNECT MET2 MET1 BY CONTACT
NOT MET2 MET1 NMET
PARASITIC CAP[M] NMET NMET BULK
LEXTRACT CCCC NMET MET1 BY CAP[M] MCAP &
EQUATION C=0.00005*AREA+1.0E-4*(PERI-OVPR)
.
.
LPESELECT CAP[M] GT 0 OUTPUT SPICE
*END
```

Equations for Fringe in extractParasitic

In the 4.7 and previous versions, for the same layer or different layer fringe (fringe command), `extractParasitic` can only take a piece-wise linear table such that default Dracula equation $C=K*WIDT/DEPT$ is used. This 4.8 version provides a solution that allows you to let `extractParasitic` take an equation and have PDRACULA translate this equation to flexible LPE (LEXTRACT/EQUATION) commands.

Command Syntax

For extractParasitic

```
extractParasitic(  
  (  
    .....  
    fringe( layer1 layer2 piecewiseList  
            {equation("userEquation1") {equation(  
              "userEquation2") } } {parset(pname)} )  
    .....  
  )  
)
```

For expanded rules of extractParasitic

```
*OPERATION  
; user specified equation  
  LEXTRACT parsetName layer1 BY CAP[subtype] CAPn1 &  
  EQUATION user-specified equation 1 &  
  EQUATION user-specified equation 2
```

Arguments

userEquation

The user specified equation for fringe capacitance calculation, such as "C=0.2*WIDT/DEPT+0.05*WIDT/(DEPT*DEPT)". More than one equation can be specified and should be protected with double quotation marks. At least one pair of piecewise list is required to give the maximum fringe distance. Piecewise list with zero distance is also necessary to extract capacitance when DEPT=0. Equation command can be placed anywhere after piecewise list. Users need to keep the length of the equation less than 200 characters, including blank. Should the equation be longer, the user needs to break it into two equations.

pname

The default parset name used in LEXTRACT when the equation command is used in `extractParasitic()` is "CAPF." If the equations contain parameters other than "C," "WIDT," and "DEPT" users also need to specify parset name in `extractParasitic()`. The parset command can be place anywhere after piecewise list. More over, if no default parset exist to include all parameter in the user equation, a parset command with user-defined parset name and parameters is required in the rule.

Dracula Reference

Extracting Electrical Parameters (LPE)

Example

```
*DESCRIPTION
*END
*OPERATION
.....
extractParasitic(
( layers( poly metal )
  fringe( metal metal equation("CTMP=0.003*WIDT/DEPT")
    equation ("C=(0.0020888/((DEPT+2.4)*(DEPT+2.4)*(DEPT+2.4)) +
      (-0.000514
92)/((DEPT+2.4)*(DEPT+2.4)) + 3.2717e-05/(DEPT+2.4)) *WIDT")
    parset(CAPF)
    (1.0 0.007) (1.5 0.003) )
  fringe( metal poly (1.0 0.009) (1.5 0.005) equation
    ("C=0.005*WIDT/DEPT") )
)
)
```

Supporting Inductors in Layout Designs Using the PARSET and ELEMENT IN? Commands

In version 4.8.1 and subsequent releases, you can use BOXes to simulate inductors and extract some important parameters for the inductors, such as the number of turns, diameter of inner circle, etc. and output them to the SPICE netlist file for supporting inductors in layout designs. You have to use the PARSET command (in the "Description Block Commands" section of your rule file) to specify what parameters will be extracted., followed by the ELEMENT IN? in the "Operation Block Commands" section in your rule file.

The first step is to use PARSET command to specify what parameters will be extracted. There are four geometric primitives regarding inductors provided: ID, IN, IW, and IS and one reserved parameter, LL. See the [PARSET](#) command in the "Description Block Commands" chapter in this manual for more information on these primitives and reserved parameter.

The next step is to use the ELEMENT IN? command. See the section [ELEMENT IN?](#) in the "Operation Block Commands" chapter of this manual for more information on that command.

In addition, the command file needs to include the PAR-COMPARE command in the DESCRIPTION block, or parameters will not be recognized and compared. See PAR-COMPARE in the "Description Block Commands" chapter of this manual for more information.

The netlist must specify the parameters; however, if the parameters are out of expected ranges, the parameters are just ignored. For example, if the netlist is in Microns (No SCALE

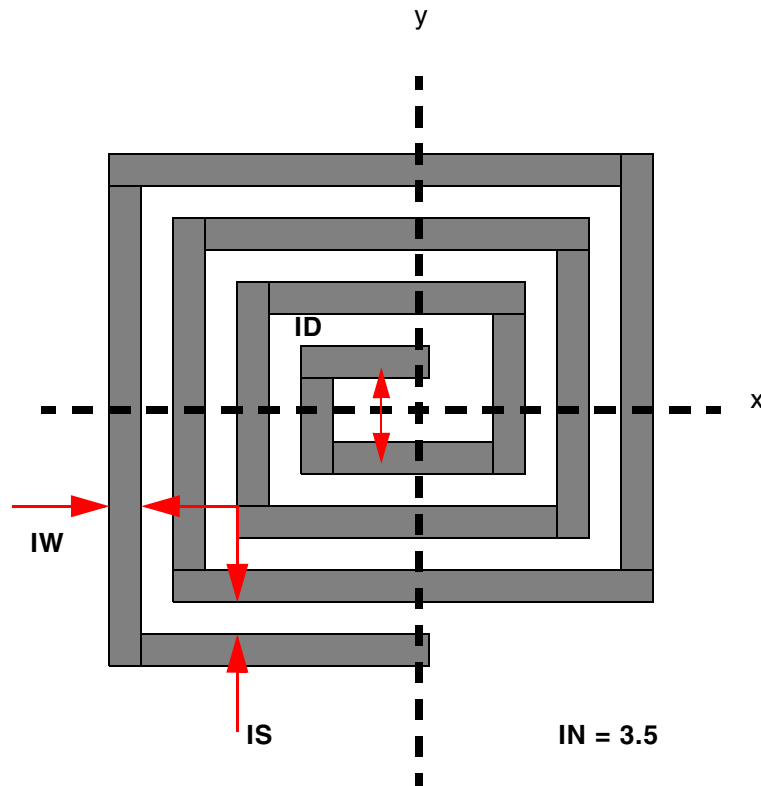
Dracula Reference

Extracting Electrical Parameters (LPE)

METER statement present), then parameters of 1e-8 would mean 1.0e-8 microns and they would just be ignored, since the netlist is already in microns.

Two kinds of shapes of inductors are supported: squares and circles as shown in the following two figures.

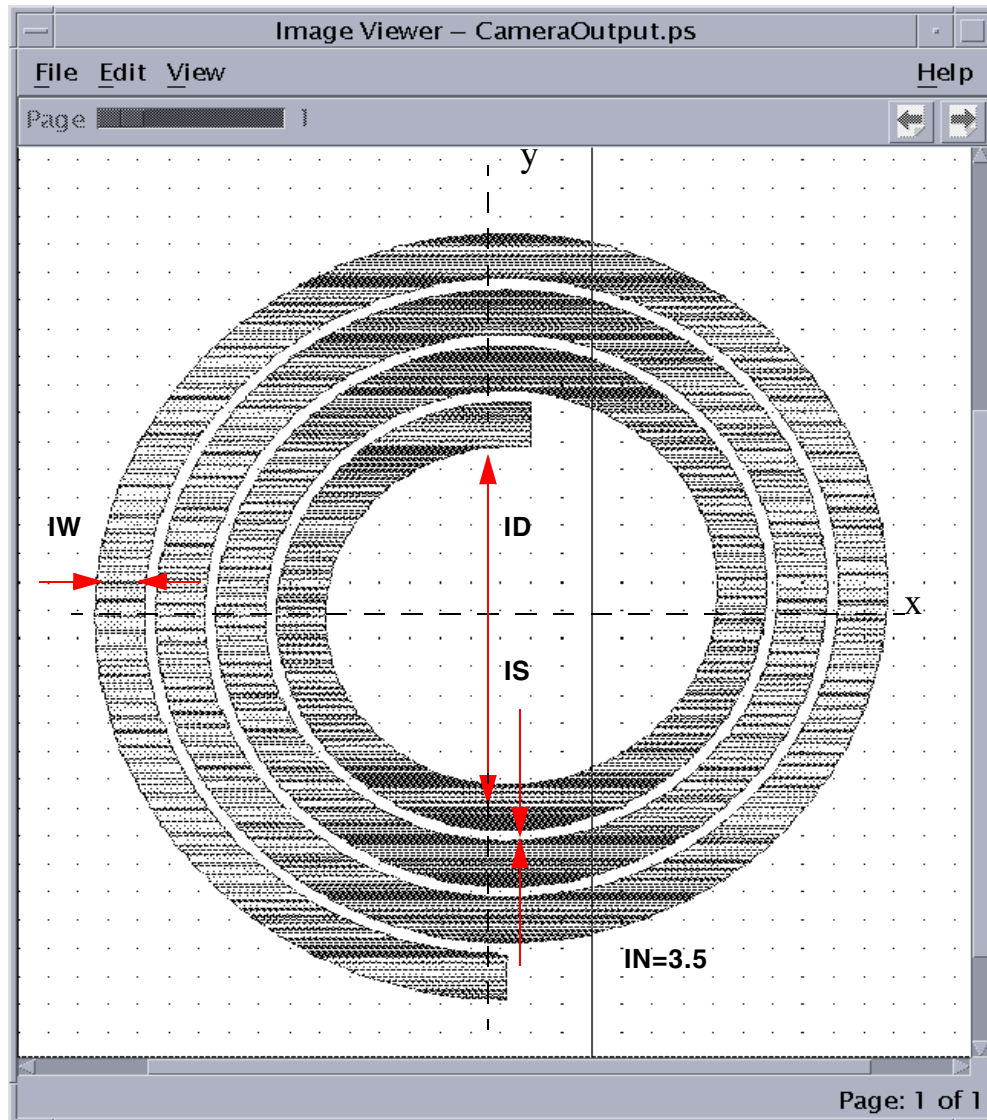
Square Inductors



There are some constraints of the starting and ending points of the shapes.

- They should be either on the X or Y axis. In other words, the count of circles should be integral times of 1/4.
- Depending on the orientation of the circles, both ends of the shapes should be a little bit ahead of the axes to get a more accurate result of the extraction.
- The outside end of the loop of a square inductor should not extend past the outside edge of the inductor body.

Circular Inductors



Supporting More Geometric Parameter Extraction for Box Elements Using the PARSET and LEXTRACT Commands

By default, the Dracula product extracts the `AREA` and `PERI` parameters for box elements. In this version and subsequent versions, you can extract more geometric parameters for a box element. The allowable parameters are `ANG`, `AREA`, `PERI`, `OVPR`, `OVAR`, `ID`, `IN`, `IW`, and `IS`. To extract parameters other than `AREA` and `PERI`, you have to use the `PARSET` command to specify what parameters will be extracted, followed by the `ELEMENT BOX` command. You

define the PARSET command in the “Operation Block Commands” section and you define the ELEMENT BOX command in the “Description Block Commands” section of your rules file. Then, you have to use the Dracula flexible LPE commands LEXTRACT and EQUATION to explicitly specify the parameters extracted for a box element.

Selective LPE

Selective LPE extracts layout parasitics and devices associated with a path. The LPE-QUERY command runs selective LPE. LPE-QUERY is designed to run in both batch and incremental run modes.

In addition to parameter extraction, LPE-QUERY can also output the geometries associated with a node. Dracula performs LPE layout extraction with a node-based scheme. One advantage of this node-based extraction scheme is its efficiency in handling multiple path extraction with overlapped nodes. You need to specify and extract overlapped nodes only once.

A path is defined as a series of nodes or nets. You must define the node set that corresponds to the path you want to analyze. You can specify the nodes using schematic or layout node names or numbers. If you specify schematic node names or numbers, Dracula maps the schematic node names to corresponding layout node numbers to perform extraction. In this case, you must run LVS to establish a cross-reference before you use LPE-QUERY.

Note: Dracula does not check whether a given node set is from a continuous path. You must make sure that the logical path maps correctly to the physical node set.

Use the following commands to run selective LPE:

- **LPE-QUERY = ON/CORE/EXPAND**

Include this command in the Description block to turn on selective LPE.

- **SELECT-MODE**

Include this command in the Description block to indicate whether the node name or number is based on the schematic or layout.

- **NODE-SELECT or NODE-FILTER**

Include one of these commands in the Operation block.

NODE-SELECT specifies a node set.

NODE-FILE specifies a file that lists a node set.

For details about these commands, refer to [Chapter 11, “Description Block Commands.”](#)

Selective LPE Batch Example

The example in this section shows the rules file for a selective LPE run in batch mode.

```
*DESCRIPTION
.
LPE-QUERY = ON
SELECT-MODE = SCH
.
*END
.
*OPERATION
.
;define element devices
.
;define parasitic devices
.
LPECHK
.
;extract first node set
.
NODE-SELECT = NET1 NET2 NET3
;OR
;NODE-FILE = CLOCK
LPESELECT CAP[C] &
LPESELECT CAP[B] OUTPUT SPICE1
.
;extract second node set
.
NODE-SELECT = NET4 NET5 NET6 OUTPUT CRITICAL 63
LPESELECT CAP[A] &
LPESELECT CAP[D] OUTPUT SPICE2
*END
```

Selective LPE Incremental Example

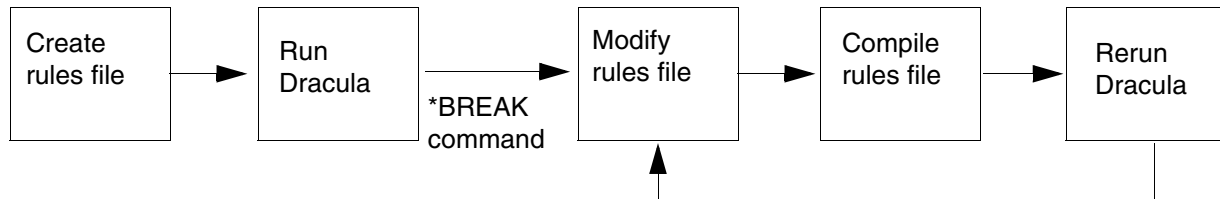
The example in this section shows the rules file for an incremental selective LPE run. If you need to analyze additional paths after finishing the current paths, incremental mode can reduce CPU time.

In incremental mode, the ***BREAK** command interrupts the run so you can modify the rules file to list a new node set, recompile the rules file, and rerun Dracula starting at the command

Dracula Reference

Extracting Electrical Parameters (LPE)

after the *BREAK. Dracula does not repeat steps it has already performed, as shown in the following illustration.



```
*DESCRIPTION
.
LPE-QUERY = ON
SELECT-MODE = SCH
.
*END
.
*OPERATION
.
;define element devices
.
;define parasitic devices
.
LPECHK
.
;extract first node set
.
*BREAK INCRE           ;modify and recompile rules file
NODE-FILE = CLOCK      ;to extract next node set
LPESELECT CAP[C] &
LPESELECT CAP[B] OUTPUT SPICE1
.
;modify rules file to extract second node set
.
NODE-SELECT = NET4 NET5 NET6
LPESELECT CAP[A] &
LPESELECT CAP[D] OUTPUT SPICE2
*END
```

LPE Examples

This section contains a rules file used to extract electrical parameters from an IC layout. LPE extracts the parasitic capacitance of a node, the area and perimeter of a diode, and the gate size of a MOS transistor. LPE creates a netlist that you can use to check the consistency of layout to circuit simulation, circuit and logic-level simulation, and timing verification. The sample problem uses many features of LPE but does not represent all possible combinations. Following this example are sample CMOS and NMOS rules files to use with LPE.

Features of the Example

The example is a circuit processed in silicon-gate CMOS technology. The key features of the problem are as follows:

- *Ptwell* and *ntnsub* pseudo contact layers are created by logicals to make connections between P-well and the P+ diffusion layer and between substrate and the N+ diffusion layer. (If there is no possible connection to the substrate or well, you can use a LINK command to force the substrate or well to connect to a text layout label, such as VCC or GND.)
- The example specifies two kinds of transistors.

MOS[N]: N-channel in the P-well
MOS[P]: P-channel in the *nsub* substrate

- The example specifies two kinds of parasitic diodes.

DIO[N]: N+ diffusion in the P-well with well-ties discarded
DIO[P]: P+ diffusion in *nsub* substrate with substrate-ties discarded

- The example specifies seven kinds of parasitic capacitances.

CAP[A]: Capacitance from *metal* to *poly*
CAP[B]: Capacitance from *metal* to N+ diffusion
CAP[C]: Capacitance from *metal* to P+ diffusion
CAP[D]: Capacitance from *metal* to *P-well*
CAP[E]: Capacitance from *metal* to substrate
CAP[F]: Capacitance from *poly* to *P-well*
CAP[G]: Capacitance from *poly* to substrate

The LPECHK command starts LVS consistency checking and backannotates the extracted SPICE list with the schematic names.

Dracula Reference

Extracting Electrical Parameters (LPE)

The LPESELECT command assigns values in the extracted SPICE list during the LPE run. You can also generate this SPICE list with LPEPRO.

Rules File in a MOS Example

This section describes the construction of an LPE rules file for a silicon-gate CMOS process technology. An LPE rules file is an extension of an ERC or LVS rules file. The file contains three blocks: the Description block, the Input-Layer block, and the Operation block. Samples of these blocks follow, with explanations of their major commands.

The Description Block

Description block commands are similar to DRC, ERC, and LVS commands. The major commands required for LPE include the following:

DIODESEQ

Specifies the parasitic diode parameters A1 and P1 (area and perimeter). The default is A1 and P1.

MODEL

Specifies the MOS and diode SPICE model (type) in the SPICE listing. If you do not specify MODEL, all MOS and diode statements become comments in the SPICE listing.

UNIT

Specifies the parasitic capacitor unit such as *pf*, *picofarad*. The default reports the natural unit on the layout.

SCHEMATIC

Use in conjunction with the LPECHK command. Specify only if an LVS verification check command provides a schematic netlist. The default SPICE listing is based on text layout labels.

```
; *****
; SILICON GATE CMOS LPE RUN
; INPUT RULES FILE
; *****
*DESCRIPTION
OUTDISK = LPEOUT           ; Graphic error cell file name
PRIMARY = ECAD             ; Top cell name
MODE = EXEC NOW            ; Mode execution card
SYSTEM = GDS2              ; Graphic CAD database format
PROGRAM-DIR = QA:          ; Program directory of Cadence modules
SCALE = .001 MICRON        ; Graphic database scale
RESOLUTION = .25 MICRON    ; Database digitized grid unit
PRINTFILE = LPEPRT         ; Print file name of DRACULA run
;
```


Dracula Reference

Extracting Electrical Parameters (LPE)

```
DIODESEQ = A1 P1          ; Spice diode parameter sequence card
MODEL = MOS[N],N MOS[P],P DIO[N],N DIO[P],P ; Spice transistor &
                                   ; diode model specifications
UNIT = CAPACITANCE,PF      ; Spice capacitance unit
                                   ; specification
;
SCHEMATIC = LVSLOGIC        ; Schematic netlist from LOGLVS
                                   ; compiler
*END
```

Input-Layer Block

The major commands for the Input-Layer block include the following:

SUBSTRATE

Generates a new *bulk* layer, which generates the *nsub* layer. The substrate and well make up a terminal of most parasitic capacitors and diodes defined in Dracula. You can use the layer generated by SUBSTRATE to create “reverse field masks” by using logical operations.

If you do not draw and include a *substrate* layer in the layout database, as in the example, you must specify the Input-Layer block SUBSTRATE command to create reverse field masks. Dracula assigns the dimension of this *substrate* layer, but you can assign it yourself if you specify the WINDOW or the WINDOW-CUT=EDGE command. For more information, refer to [Chapter 12, “Input-Layer Block Commands.”](#)

CONNECT-LAYER

Specifies the conductor layers that carry node information, as in ERC or LVS. If you used the CONNECT command in the Operation Block, CONNECT-LAYER is required.

The CONNECT-LAYER command also assigns the text-attaching sequence, which you can redefine with the TEXTSEQUENCE command. For layout texting requirements, refer to the layout texting procedure for running ERC or LVS. For additional information, refer to [Chapter 12, “Input-Layer Block Commands.”](#)

```
*INPUT-LAYER
;
DIFF = 2          ; Thin oxide regions
PWELL = 1         ; P-well regions
POLY = 3          ; Poly silicon
CONT = 5          ; Metal contacts
METAL = 6         ; Metal
VAPOX = 7 TEXT 30 ; Pad opening & text layer 30
PPLUS = 8         ; Layer created for the substrate
SUBSTRATE = BULK 63
;
CONNECT-LAY = NSUB PWELL PSD NSD POLY METAL
; *** The conductor layers of the CMOS process ***
*END
```

Dracula Reference

Extracting Electrical Parameters (LPE)

Operation Block

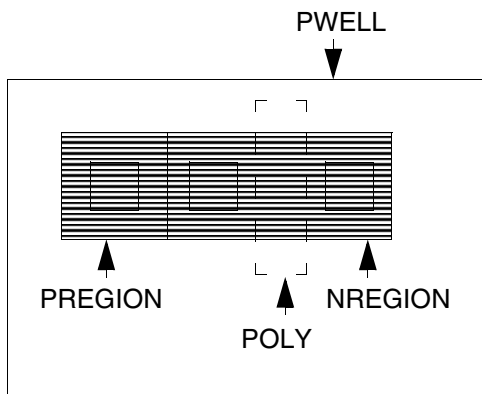
The major commands in the Operation block example are the same as those for running an LVS job with SUBSTRATE type checking. The layers in the CONNECT operations must appear in the CONNECT-LAYER command in the Input-Layer block.

```
; *****
; OPERATION BLOCK
;
; CREATE THE DEVICE LAYERS AND CONDUCTOR LAYERS
; *****

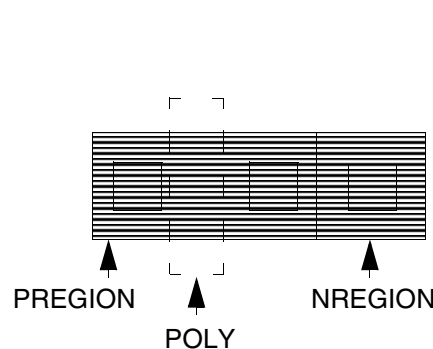
*OPERATION
AND DIFF PPLUS PREGION      ; All active P+ regions
NOT DIFF PREGION NREGION    ; All active N+ regions
AND POLY NREGION NGATE      ; N-Channel devices
NOT NREGION NGATE NSD       ; N+ source/drain & substrate ties
AND POLY PREGION PGATE      ; P-Channel devices
NOT PREGION PGATE PSD       ; P+ source/drain & p-well ties
AND VAPOX METAL BPAD        ; Bonding pad device
NOT BULK PWELL NSUB         ; Created N-substrate layer
AND PSD PWELL PTWELL        ; Pseudo P+ tie for P-well
NOT NSD PWELL NTNSUB        ; Pseudo N+ tie for N-substrate
;
; *****
; CONNECT OPERATION FOR CMOS PROCESS
; *****
;
CONNECT METAL POLY BY CONT
CONNECT METAL NSD BY CONT
CONNECT METAL PSD BY CONT
CONNECT PSD PWELL BY PTWELL
CONNECT NSD NSUB BY NTNSUB
```

The following figure shows the layout of a MOS device in its original form (N-type and P-type).

N-CHANNEL



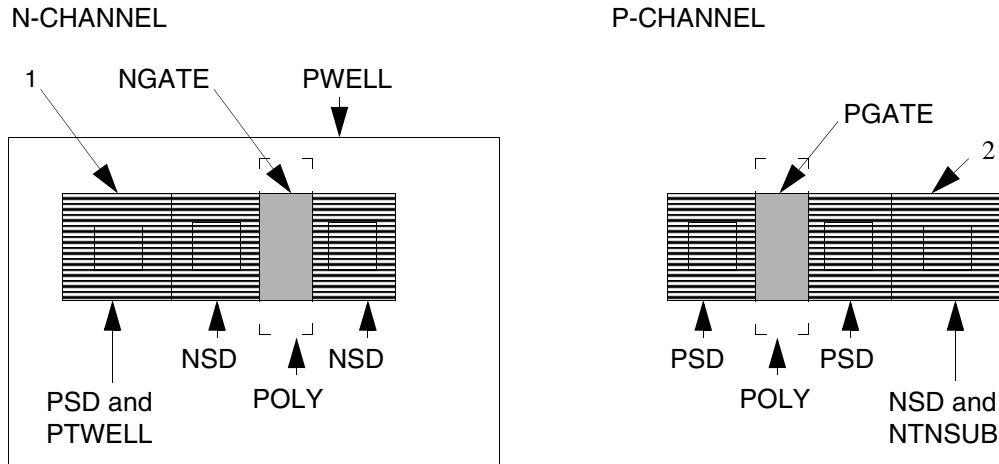
P-CHANNEL



Dracula Reference

Extracting Electrical Parameters (LPE)

The following figure shows the extracted layout after the program performs the logical operations to *substrate*. (1) is a pseudo P+ tie created by the program; (2) is a pseudo N+ tie created by the program.



Connection of the *pwell* is made from node information from *psd* to *pwell* through *ptwell*, the generated contact layer. Connection of *nsub* is made from node information from *nsd* to *nsub* through *ntnsub*, the contact layer. Also, the *nsub* layer (substrate for the P-channel devices) is generated from the *bulk* layer specified by the SUBSTRATE command. This is also known as a “reverse mask operation.”

The major commands include the following:

AND **and** NOT

Carry node information to the output generated layers because their field 1 layers carry node information. These generated layers are used for the device layers of the PARASITIC CAPACITOR and PARASITIC DIODE commands.

The PARASITIC DIO device layer includes only real diodes, not substrate or well ties. The device layer must not include any N+ or P+ ties or guard rings (in the previous figure, “1” and “2” denote a *pwell* tie and *nsub* tie, respectively). The logical operation that generated the *ndio* and *pdio* are N+ diode to the *pwell* and P+ diode to the *nsub*, respectively.

PARASITIC CAP

Specify the parasitic diodes and capacitors.

Dracula Reference

Extracting Electrical Parameters (LPE)

PARASITIC DIO

SPICE listing. For PARASITIC DIO, the PARASITIC CAP device layer must carry node information.

ATTRIBUTE

Assigns the capacitance value for each type of parasitic capacitor.

```
; OPERATION Block (Continued)
;
; *****
; LOGICAL OPERATION DEFINING THE PARASITIC CAPACITORS AND DIODES
; *****
;
AND METAL POLY MPOLY      ; Metal to poly capacitors
NOT METAL POLY METNP      ; Remove poly underneath metal
AND METNP NSD MNSD        ; Metal to N+ source/drain capacitors
AND METNP PSD MPSD        ; Metal to P+ source/drain capacitors
NOT METNP NSD M1
NOT M1 PSD M2              ; Further remove source/drain under metal
AND M2 PWELL MWELL        ; Metal to P-well capacitors
AND M2 NSUB MSUB          ; Metal to N-substrate capacitors
NOT POLY NGATE P1
NOT P1 PGATE P2           ; Remove channel region from poly
AND P2 PWELL POWELL       ; Poly to P-well capacitors
AND P2 NSUB POSUB         ; Poly to N-substrate capacitors
AND NSD PWELL NDIO        ; N+ diode to P-well minus the N+ ties
NOT PSD PWELL PDIO        ; P+ diode to N-substrate minus the P+ ties
;
; *****
; DEFINING TRANSISTOR ELEMENTS AND PARASITIC ELEMENTS
; *****
ELEMENT MOS[N] NGATE POLY NSD PWELL ;N-channel
ELEMENT MOS[P] PGATE POLY PSD NSUB  ;P-channel
ELEMENT PAD BPAD METAL              ;Pad element
PARASITIC DIO[N] NDIO PWELL NDIO    ;N+ diode to P-well
PARASITIC DIO[P] PDIO PDIO NSUB     ;P+ diode to N-substrate
PARASITIC CAP[A] MPOLY METAL POLY   ;Metal to poly capacitor &
ATTRIBUTE CAP[A] 0.00005             ;attribute (pf/sq micron)
PARASITIC CAP[B] MNSD METAL NSD     ;Metal to N+ capacitor &
ATTRIBUTE CAP[B] 0.00005             ;attribute (pf/sq micron)
PARASITIC CAP[C] MPSD METAL PSD     ;Metal to P+ capacitor &
ATTRIBUTE CAP[C] 0.00005             ;attribute (pf/sq micron)
PARASITIC CAP[D] MWELL METAL PWELL  ;Metal to P-well capacitor &
ATTRIBUTE CAP[D] 0.00005             ;attribute (pf/sq micron)
PARASITIC CAP[E] MSUB METAL NSUB    ;Metal to N-substrate
                                           ;capacitor &
ATTRIBUTE CAP[E] 0.00005             ;attribute (pf/sq micron)
PARASITIC CAP[F] POWELL POLY PWELL  ;Poly to P-well capacitor &
ATTRIBUTE CAP[F] 0.00005             ;attribute (pf/sq micron)
PARASITIC CAP[G] POSUB POLY NSUB    ;Poly to N-substrate
                                           ;capacitor &
ATTRIBUTE CAP[G] 0.00005             ;attribute (pf/sq micron)
```

The major commands include the following:

Dracula Reference

Extracting Electrical Parameters (LPE)

LPECHK

Starts LVS. Use LPECHK to backannotate an extracted SPICE listing with schematic names. If no schematic netlist exists, do not specify LPECHK; the extracted CDL listing is based on layout text and system-assigned internal node numbers.

LPeselect

Specifies the *nospi* SPICE list file. The example using the ampersand (&) shows separation of the device in the CDL listing. The [S] option backannotates the CDL listing with schematic names. This option has no meaning without the LPECHK command.

```
;OPERATION Block (Continued)
;
LPECHK                      ;Layout versus schematic
;
LPeselect[S] MOS[P] &      ;SPICE output file for transistors
LPeselect[S] MOS[N] &
LPeselect[S] DIO[P] &
LPeselect[S] DIO[N] &
LPeselect[S] CAP[A] GT 0.0 &
LPeselect[S] CAP[B] GT 0.0 &
LPeselect[S] CAP[C] GT 0.0 &
LPeselect[S] CAP[D] GT 0.0 &
LPeselect[S] CAP[E] GT 0.0 &
LPeselect[S] CAP[F] GT 0.0 &
LPeselect[S] CAP[G] GT 0.0 OUTPUT NOSPI
;
*END
```

Dracula Reference

Extracting Electrical Parameters (LPE)

NMOS LPE Input File

```
;*****
;
;           SILICON GATE NMOS LPE RUN
;           INPUT RULES FILE
;*****
*DESCRIPTION
INDISK = inchip           ; Graphic database file
OUTDISK = lpeout          ; Graphic error database file
PRIMARY = top             ; Top cell name
MODE = exec now           ; Mode execution
SYSTEM = gds2             ; Graphic CAD database format
PROGRAM-DIR = usr1/dracula/bin/
                           ; Program directory of Dracula modules
SCALE = .001 micron       ; Graphic database scale factor
RESOLUTION = .5 micron    ; Graphic digitized grid increments
PRINTFILE = lpeprt        ; Print file name LPE run
;
DIODESEQ = a1 p1          ; Spice diode parameter sequence
MODEL= MOS[E],enh MOS[D],dep DIO[N],ndio ; Spice transistor &
                           ; diode model specifications
UNIT= CAPACITANCE,pf      ; Spice capacitor unit specifier
SCHEMATIC = LVSLOGIC      ; Schematic netlist from LOGLVS
;
*END
;*****
;           INPUT-LAYER BLOCK
;*****
*INPUT-LAYER
diff = 1                  ; Thin oxide region
depl = 2                  ; Depletion VT adjust implant
bcc = 3                   ; Buried contact for poly to diffusion
poly =4                   ; Polysilicon
mcc = 5                   ; Metal contacts
metal =6 TEXT 30          ; Metal & Text layer 30
pad = 7                   ; Pad opening
;
CONNECT-LAYER= srcdrn poly metal
;*** The conductor layers of the NMOS process ***
;
SUBSTRATE psub 63 ; Create p - substrate layer
*END
;*****
;           OPERATION BLOCK
;           CREATE THE DEVICE LAYERS AND CONDUCTOR LAYERS
;*****
*OPERATION
AND diff poly gbc
NOT gbc bcc gate          ; All transistors
NOT diff gate srcdrn      ; N + source/drain layer
AND gate depl dgate       ; Depletion devices
NOT gate dgate egate      ; Enhancement devices
AND pad metal padwin      ; Bonding pad
;*****
;           CONNECT OPERATION FOR NMOS PROCESS
;*****
CONNECT metal poly BY mcc
CONNECT metal srcdrn BY mcc
```

Dracula Reference

Extracting Electrical Parameters (LPE)

```
CONNECT srcdrn poly BY bcc
LINK psub TO vss ; Link the p - substrate to VSS
;*****
LOGICAL OPERATION TO DEFINE THE PARASITIC
; CAPACITORS AND DIODE
;*****
AND metal poly mpoly ; Metal to poly capacitor
NOT metal poly metnp ; Remove poly from metal
AND metnp srcdrn msd ; Metal to source/drain capacitors
NOT metnp srcdrn metsub ; Remove source/drain from metal
;
NOT poly gate polysub ; Poly to substrate capacitors
;*****
; DEFINING TRANSISTOR ELEMENTS
; AND PARASITIC ELEMENT OF NMOS PROCESS
;*****
ELEMENT MOS[D] dgate poly srcdrn psub ; Depletion device
ELEMENT MOS[E] egate poly srcdrn psub ; Enhancement device
;
ELEMENT PAD padwin metal ; Pad element
;
PARASITIC DIO[N] srcdrn psub srcdrn ; N + diode to p-substrate
PARASITIC CAP[A] mpoly metal poly ; Metal to poly capacitor &
ATTRIBUTE CAP[A] 0.0005 ; attribute( pF/micron )
PARASITIC CAP[B] msd metal srcdrn ; Metal to source/drain
ATTRIBUTE CAP[B] 0.0005 ; capacitor & attribute
; ; ( pF/sq micron )
PARASITIC CAP[C] metsub metal psub ; Metal to p-substrate
ATTRIBUTE CAP[C] 0.0002 ; capacitor & attribute
; ; ( pF/sq micron )
PARASITIC CAP[D] polysub poly psub ; Poly to p-substrate
ATTRIBUTE CAP[D] 0.0005 ; capacitor & attribute
; ; ( pF/sq micron )
LPECHK ; Layout versus schematic check
LPESELECT[S] MOS & ; SPICE-like netlist of
; ; transistors, diodes, and
LPESELECT[S] DIO & ; capacitors in file name,
LPESELECT[S] CAP GT 0.0 OUTPUT simon ; SIMON.DAT
*END
```

CMOS LPE Input File

```
;*****
; SILICON GATE CMOS LPE RUN
; INPUT RULES FILE
;*****
*DESCRIPTION
INDISK = inchip ; Graphic database file name
OUTDISK = lpeout ; Graphic error cell file name
PRIMARY = top ; Top cell name
MODE = exec now ; Mode execution
SYSTEM = gds2 ; Graphic CAD database format
PROGRAM-DIR = usr1/dracula/bin/
; Program directory of Dracula modules
SCALE = .001 micron ; Graphic database scale
RESOLUTION = .5 micron ; Database digitized grid unit
PRINTFILE = lpeprt ; Print file name of DRACULA run
;
```

Dracula Reference

Extracting Electrical Parameters (LPE)

```
DIODESEQ = a1 p1          ; diode parameter sequence
MODEL = MOS[N],n MOS[P],p DIO[N],n DIO[P],p
;                          ; transistor & diode model
;                          ; specifications
UNIT = capacitance,pf     ; capacitance unit specification
SCHEMATIC = LVSLOGIC      ; Schematic netlist from LOGLVS compiler
*END

;*****
; INPUT-LAYER BLOCK
;*****
*INPUT-LAYERS
diff = 1                  ; Thin oxide regions
pwell = 2                 ; P-well regions
poly = 3                  ; Polysilicon
nplus = 4                 ; N + implant
pplus = 5                 ; P + implant
cont = 6                  ; Metal contacts
metal = 7                 ; Metal
vapox = 8 TEXT=28         ; Pad opening & Text layer 28
SUBSTRATE = bulk 63       ; Layer created for reverse mask
CONNECT-LAYER = nsub pwell psd nsd poly metal
;*** The conductor layers of the CMOS Process. ***
*END
;*****
; OPERATION BLOCK
; CREATE THE DEVICE LAYERS AND CONDUCTOR LAYERS
;*****
*OPERATION
AND diff nplus nregion    ; All active N + regions
AND poly nregion ngate    ; N-Channel devices
NOT nregion ngate nsd     ; N+ source/drain & taps
;
AND diff pplus pregon     ; All active p + regions
AND poly pregon pgate     ; P-Channel devices
NOT pregon pgate psd      ; P+ source/drain & taps
;
AND vapox metal bpad      ; Bonding pad device
;
NOT bulk pwell nsub       ; Created N-Substrate layer
;
AND psd pwell ptwell      ; Pseudo P+ tap for P-well
;
SELECT nsd OUTSIDE pwell ntsub ; Pseudo N+ tap for N-Substrate
;
;*****
; CONNECT OPERATION FOR CMOS PROCESS
;*****
CONNECT metal poly BY cont
CONNECT metal nsd BY cont
CONNECT metal psd BY cont
CONNECT psd pwell BY ptwell
CONNECT nsd nsub BY ntsub
;*****
; LOGICAL OPERATION DEFINING THE
; PARASITIC CAPACITORS AND DIODES
;*****
AND metal poly mpoly      ; Metal to poly capacitors
NOT metal poly metnp      ; Remove poly underneath metal
AND metnp nsd mnsd        ; Metal to N+source/drain capacitors
```


Dracula Reference

Extracting Electrical Parameters (LPE)

```
AND metnp psd mpsd      ; Metal to P+source/drain capacitors
NOT metnp nsd m1
NOT m1 psd m2           ; Further remove source/drain under metal
AND m2 pwell mwell      ; Metal to p-well capacitors
AND m2 nsub msub        ; Metal to n-substrate capacitors
NOT poly ngate p1
NOT p1 pgate p2         ; Remove channel region from poly
AND p2 pwell powell     ; Poly to p-well capacitors
AND p2 nsub posub       ; Poly to n-substrate capacitors
AND nsd pwell ndio      ; N+diode to p-well minus taps
NOT psd pwell pdio      ; P+diode to n-substrate minus taps

; *****
;           DEFINING TRANSISTOR ELEMENTS AND
;           PARASITIC ELEMENTS
; *****

ELEMENT MOS[N] ngate poly nsd pwell ; N-Channel device
ELEMENT MOS[P] pgate poly psd nsub  ; P-Channel device
ELEMENT PAD bpad metal              ; Pad element
;
;
PARASITIC DIO[N] ndio pwell ndio    ; N+diode to p-well
PARASITIC DIO[P] pdio pdio nsub     ; P+diode to n-substrate
;
PARASITIC CAP[A] mpoly metal poly   ; Metal to poly capacitor &
ATTRIBUTE CAP[A] 0.00005             ; attribute( pF/micron )
PARASITIC CAP[B] mnsd metal nsd     ; Metal to n+capacitor &
ATTRIBUTE CAP[B] 0.00005             ; attribute( pF/sq micron )
PARASITIC CAP[C] mpsd metal psd     ; Metal to p+capacitor &
ATTRIBUTE CAP[C] 0.00005             ; attribute( pF/sq micron )
PARASITIC CAP[D] mwell metal pwell  ; Metal to p-well capacitor &
ATTRIBUTE CAP[D] 0.00005             ; attribute( pF/sq micron )
PARASITIC CAP[E] msub metal nsub     ; Metal to n-substrate
ATTRIBUTE CAP[E] 0.00005             ; capacitor & attribute
;                                     ; ( pF/sq micron )
PARASITIC CAP[F] powell poly pwell  ; Poly to p-well capacitor &
ATTRIBUTE CAP[F] 0.00005             ; attribute( pF/sq micron )
PARASITIC CAP[G] posub poly nsub    ; Poly to n-substrate
ATTRIBUTE CAP[G] 0.00005             ; capacitor & attribute
;                                     ; ( pF/sq micron )
LPECHK                               ; Layout versus schematic
;                                     ; check
;
;
LPESELECT[S] MOS &                  ; SPICE output file for
;                                     ; transistors, diodes,
LPESELECT[S] DIO &                  ; and capacitors in an
LPESELECT[S] CAP GT 0.0 OUTPUT spice ; output file, SPICE.DAT
*END
```

Hierarchical LPE

Hierarchical Layout Parameter Extraction (HLPE) extracts circuit elements and parasitic devices using Hcells. The commands for circuit extractions are the same as those in the LPE flat mode. You can run HLPE in cell and composite modes with the same commands and concepts as in HLVS.

Dracula Reference

Extracting Electrical Parameters (LPE)

HLPE extracts and reports circuit device sizes and parasitic information from the layout. It outputs in SPICE/CDL format a subcircuit listing for each Hcell with its own parasitics and a top subcircuit listing with calls (instantiations) to the hierarchical cells. The output is a description of the circuit in a two-level hierarchy. You can do backannotation to schematic netlist if you input a schematic netlist and run HLVS together with HLPE (just as when LVS is run jointly with LPE in a flat mode run).

To generate a subcircuit listing of Hcells, run cell mode HLPE with all cell pin text. You can then run composite mode HLPE to generate the top-level netlist with subcircuit calls.

To compare the ratio of device area to the layer area driving the devices, you can do an *antenna check*. This check helps ensure that induced capacitance, caused by a fabrication process or a circuit operation, does not cause the drive current or the transition time to become unstable.

To run an antenna check with HLPE

1. Run cell mode antenna check.
2. Run composite mode antenna check on the same directory.

When running composite mode, COMPUTE reads the parameter file generated by the cell mode and adds the data to the composite plane for each cell instance.

See the LEXTRACT, COMPUTE, and CHKPAR commands for more information about antenna checks.

Cell Mode

Cell mode generates the SPICE.CEL file containing subcircuit listings of the hierarchical cells specified in *cell.tab* and schematic names with parasitics and MOS device sizes.

If you want back-annotation to schematics, input a schematic netlist with an Hcell table to LOGLVIS and run the same as a flat mode run.

Composite Mode

Composite mode HLPE generates a SPICE.DAT file containing top level devices with parasitics and subcircuit calls to the hierarchical cells. If you run the job in the same directory that you used for the cell mode run and the SPICE.CEL file exists, Dracula creates SPICE.DAT by appending the composite mode output to SPICE.CEL so that it becomes a complete netlist of the layout circuit.

Dracula Reference

Extracting Electrical Parameters (LPE)

Using Parametric Data (LPEPRO)

 Important

Dracula LPEPRO has been retired, and will be no longer supported as of December 31, 2008. Use the DRAC2CDL executable in order to generate a CDL netlist from specific Dracula binary files. To learn more about DRAC2CDL refer to [Chapter 10, “Using DRAC2CDL Netlist Generator,”](#).

Chapter 8 discusses the following for LPEPRO in Dracula:

- [LPEPRO Overview](#)
- [Terminal Commands](#)
- [Administrative Commands](#)
- [Network and Gate Commands](#)
- [Delay Value Command](#)
- [Path Analysis Commands](#)
- [Report Commands](#)
- [Scanning Layout Parameters](#)
- [SPICE Generator](#)
- [Logic Gate Generator](#)
- [Calculating Delay](#)

LPEPRO Overview

Even on a small chip, a large amount of parametric data is extracted from the layout and the information you need is hidden in it. The Layout Parameter Extraction Postprocessor (LPEPRO) is an interactive program that lets you

Dracula Reference

Using Parametric Data (LPEPRO)

- View parametric information
- Generate logic network information from a layout
- Produce SPICE files for selected paths
- Verify the timing of selected paths based on layout parameters

You use LPEPRO after executing Layout Parameter Extraction (LPE). An LPE run with input schematics provides schematic data so you can annotate node names. LPE creates the parameter database and LPEPRO (the postprocessor) reads and interprets this database. For an explanation of the parameters extracted from a layout, refer to [“LPE Overview”](#) on page 355.

LPEPRO has the following features:

- Layout parameter scanner
- Logic gate generator
- SPICE generator
- Path timing analyzer

Using LPEPRO

To use LPEPRO, the LPE LPEDIR.TAB file, the LPE PARSET.TAB file, and the optional LVSLOGIC.DAT schematic file must be present in the directory in which you ran the LPE job. If you use a logic level schematic netlist to run LPE, you need to issue a SAVE LOGIC command before the CONVERT command when you run LOGLVS. (For a CDL, SPICE, or SIMON netlist, you don't need to issue a SAVE LOGIC command).

If you provide schematic data, LPEPRO uses the node names. If you provide only layout data, the program uses the names in the layout whenever possible; otherwise, it assigns internal node numbers. You can choose whether to use schematic data with the GET LAYOUT, GET SCHEMATIC, or GET SPICE commands.

With the layout parameter scanner, you interactively select the information you want by specifying the range of a value, by specifying a window on the layout, or by providing node names.

If you do not have schematic logic data, LPEPRO generates basic gates from the layout database. The program recognizes NANDs, NORs, inverters, and several composite gates. If LPEPRO does not recognize the circuit, it generates parallel or serial devices whenever possible or leaves the transistors as is. You can then use this netlist in simulation or update it for a future LVS run.

The SPICE generator creates a SPICE file of a small cell, of selected paths, or of the whole chip. You select paths by specifying the starting and ending points. The SPICE file includes all elements along the specified paths. You can select capacitance by its value, and you can do a detailed timing analysis using a circuit simulator.

Selected path timing analysis provides logic schematic data for verifying the timing of the logic network, not the memory cell. LPEPRO calculates the delay of each cell on layout parameters W, L, and C, using the arithmetic equation in the “Extracting Electrical Parameters” chapter. After it identifies the selected paths, you can generate the SPICE file to do detailed timing analysis.

You can direct reports to a terminal, printer, or specified file.

Terminal Commands

You can interactively start all commands in LPEPRO. For some systems, you can group these commands in a file and use a system job control language to submit them as a batch job. LPEPRO commands can be classified into the following categories:

■ Administrative commands

DISK	Specifies a file name for the STORE command (The default file name is <i>IMAGE.LIS</i> .)
INPUT	Reads commands from the specified file
RESET	Resets parameters or the network
SAVE	Saves the current network information in a file
EXIT	Exits from the program

■ Commands to read in the network and generate logic gates

GET	Retrieves network information from a schematic database or layout database
DELAY	Specifies the delay value

■ Command to get delay values

CALCULATE Calculates the delay parameter

■ Commands for path analysis

LPATH	Finds the longest paths
SPATH	Finds the shortest paths
PATH	Finds all elements between starting and ending points
START	Specifies starting points
END	Specifies ending points
MASK	Specifies masked nodes, pins, or feedback loops
UNMASK	Specifies unmasked nodes or pins being masked

■ Commands to generate reports (parameter scanning, SPICE or logic netlist, delay values, path outputs, and so forth)

PRINT	Directs output to a printer
TYPE	Directs output to a terminal
STORE	Directs output to a file

All terminal commands are case-insensitive and you can abbreviate them to their first three letters.

LPEPRO generates a journal file and also lets you input a rules file. The JOURNAL.LPE file contains a list of every command you entered during the session. To use this file in your next LPEPRO session, rename the journal file (for example, from JOURNAL.LPE to *session1.del*) so that LPEPRO does not overwrite it.

Administrative Commands

This section describes in detail the administrative commands for LPEPRO.

DISK

`DISK filename`

Description

Specifies a file name for the STORE command.

Arguments

filename

The name for the file. The default file name for the STORE command is IMAGE.LIS.

Example

In this example, the DISK command closes the current file and opens a file with a new name.

```
DISK PARAM.DAT
```

Dracula Reference

Using Parametric Data (LPEPRO)

INPUT

INPUT *filename*

Description

Reads in commands from a file.

Arguments

filename

The name of the file containing the commands. The file name is limited to 32 characters.

You cannot specify *journal.lpe* because it is being written to during the LPEPRO session.

RESET

```
RESET    ALL
         [START]
         [END]
         [MASK]
         [DELAY]
         [PATH]
```

Description

Resets network and parameter information, such as starting points, ending points, masked nodes, delays, and path information.

Arguments

ALL

Resets all program information and begins the LPEPRO program from the beginning.

START

Resets starting points.

END

Resets ending points.

MASK

Resets masked nodes, either user-specified nodes or nodes found with the MASK .FEEDBACK command.

DELAY

Resets all delay values to zero.

PATH

Resets the path information file and the starting and ending points in the PATH command.

Example

```
RESET PATH
RESET ALL
```

Dracula Reference

Using Parametric Data (LPEPRO)

SAVE

SAVE DATA

Description

Saves the current network information onto the disk.

Arguments

DATA

Saves the current network or timing information on disk in the *network.sav* file. You cannot change this file name.

EXIT

EXIT

X

Description

Exits the LPEPRO program.

Network and Gate Commands

This section describes in detail the network and gate commands for LPEPRO.

GET

```
GET          SCHEMATIC
            LAYOUT  field_1
            SPICE
            DATA
```

Description

Retrieves network information from layout, schematic, or previously saved data.

Note: Do not run LPEPRO on a job that uses the LPECHK[X] command because the input of netlist information requires that reductions of transistors to PUP, SDW, and so on be performed. The GET command generates unusable data if LPECHK[X] outputs are used for LPEPRO.

Arguments

SCHEMATIC

Recovers network information that you saved with the LOGLVS SAVE LOG command. If you did not save the log file in LOGLVS, use the GET SPICE command to recover the network information. If the input schematic is SPICE or CDL only, use the GET SPICE command.

LAYOUT

Reconstructs a logic network from the transistor level netlist extracted from the layout database. If *field_1* is equal to *SCHEMATIC*, use the transistor level netlist generated by LOGLVS from SPICE input.

field_1

Has the value of *SCHEMATIC*. Can only be used with the GET LAYOUT command. The GET LAYOUT SCHEMATIC command reconstructs the logic network from the SPICE data input. You can also use GET SPICE.

SPICE

Reconstructs the logic network from the SPICE data input. Same as GET LAYOUT SCHEMATIC.

DATA

Recovers saved network information whether the data was saved with the LOGLVS SAVE LOG command or the LPEPRO DAVE DATA command.

Dracula Reference

Using Parametric Data (LPEPRO)

Examples

```
GET SCH
GET LAY SCH
GET DAT
GET SPICE
```

DELAY

```
DELAY      .DEFAULT=R1/F1  
           [output/input=R1/F1]  
           [output=R2/F2]
```

Description

Assigns a default rise and fall delay value to all output nodes. Also lets you assign a delay value to an output node.

Arguments

`.DEFAULT`

Updates all output nodes with delay values equal to zero. You must use the equal sign (=) to indicate that a delay specification follows.

`R1/F1`

Indicates rise and fall delay time specification. The delay value must be an integer. If you specify the rise value only, the slash (/) can be omitted. The value after the slash (/) is interpreted as the fall delay.

`output/input=R1/F1`

Updates the delay of one output node of wired-ORed connections. *Output* is the output node of several gates (wire-OR). Only the delay of the gate with the input node *input* is updated.

`output=R2/F2`

Updates the delay of the specified output node. Delays of all gates with this node for output are updated.

Examples

```
DELAY .DEFAULT=200/250  
DELAY OUTPUT1=10/20,OUT2/IN1=10/20
```

Delay Value Command

This section describes in detail the delay value command for LPEPRO.

CALCULATE

```
CALCULATE  PARAMETER = arithmetic equation  
            DELAY/option
```

Description

Specifies the formula to use in the delay calculation or parameter calculation. You can specify more than one option in this command.

Arguments

PARAMETER

Specifies the equation for the parameter calculation. The drain and source node of each transistor are checked separately.

Arithmetic equation

The formula you specified in the delay or parameter calculation. LPEPRO recognizes the following parameters:

W	Width of transistor
L	Length of transistor
C	Nodal capacitance

The operations you can specify include the following:

*	Multiplication
/	Division
+	Addition
-	Subtraction
(	Left parenthesis
)	Right parenthesis

Dracula Reference

Using Parametric Data (LPEPRO)

DELAY/*option*

Calculates the delay according to the formula in the delay calculation. Length and width cannot be used separately; only the ratio of these two values can be used in the equation. The */option* value is the optional delay type specification. It can have the following formats:

/RISE	Calculates and updates only the rise delays of all gates.
/FALL	Calculates and updates only the fall delay.
/ON	Calculates the input to output delay of P-type MOS.
/XON	Calculates the gate to output delay of P-type MOS.
/MOS [T]	Specifies that the delay equation is only applied to T-type MOS. Other types of MOS are not included in the delay calculation.
/RES [T]	Specifies that the delay equation is only applied to T-type MOS with resistor property. The resistor property is defined as the drain or source of a MOS connected to its gate.
/GATE=TYPE1	Delays of gate TYPE1 through TYPEN are calculated.
/OFF	Calculates input to output delay of N-type MOS.
/XOFF	Calculates gate to output delay of N-type MOS.

Examples

You can include a real number, or a number in scientific notation, in the equation.

```
CALCULATE DELAY = (L/W) * C
CAL PARA = 30 * (L/W)
CAL DELAY/ON/OFF = 25 * (L/W + 0.5) * C
CAL DELAY/GATE = BINV, DINV = 10 * (L/W + 10) * C
```

In a parameter calculation, capacitance units are in *pf*. In a delay calculation, capacitance unit is *ff* or whatever value (capacitance per unit area) you specified in the PARASITIC CAP command.

Delay calculations always result in an integer value.

Path Analysis Commands

This section describes in detail the path analysis commands for LPEPRO.

LPATH

LPATH

Description

Performs the longest path analysis. If there are feedback loops, you must mark them with the MASK .FEEDBACK command.

The longest path is defined as a path from any input that has the longest delay value in the network. If there are nodes that cannot be reached because of feedback or other reasons, they are not included.

To cut feedback loops, use the MASK .FEEDBACK command. The LPEPRO program might not be able to cut the feedback loops at the proper places. Review the masks with the TYPE .MASK command after the feedback loops are cut.

To perform path analysis, provide the starting nodes. The starting nodes can be pad type nodes, nodes defined in the schematic as input, or nodes defined by the START command.

To perform path analysis with a schematic netlist that does not come from SPICE or CDL input, you must first issue a SAVE LOG command in LOGLVS. In LOGLVS, you must issue SAVE LOG before the CONVERT command.

SPATH

SPATH

Description

Performs an analysis of the shortest paths.

The shortest path is defined as a path from any input that has the shortest delay value in the network. If nodes cannot be reached because of feedback or other reasons, they are not included.

To cut feedback loops, use the MASK .FEEDBACK command. The program might not be able to cut the feedback loops in the proper places. Review the masks with the TYPE .MASK command after the feedback loops are cut.

To perform path analysis, provide the starting nodes. The starting nodes can be pad type I nodes defined in the schematic as input or nodes defined by the START command.

To perform path analysis with a schematic netlist that does not come from SPICE or CDL input, you must first issue a SAVE LOG command in LOGLVS. In LOGLVS, you must issue SAVE LOG before the CONVERT command.

PATH

PATH

Description

Finds all elements between your specified starting and ending points. You can then print the SPICE file.

An element is included in the path if it appears in both the forward and backward tracing program. Feedback loops result in more elements being included in the path. To avoid this, use the MASK .FEEDBACK command to mask out feedback loops.

To exclude a branch, use the MASK NODE-NAME command. To select a new path, use the RESET PATH command, then redefine the starting and ending nodes before issuing another PATH command.

If the schematic netlist does not come from SPICE or CDL input, a SAVE LOG must be done in LOGLVS or no path analysis can be done with the GET SCHEMATIC command. When running LOGLVS, you must issue SAVE LOG before the CONVERT command.

START

`START   node1,node2,...noden`

Description

Specifies starting points in the PATH, LPATH, and SPATH commands.

Arguments

`node1,...noden`

The names of the nodes. If the network is from the schematic, use the names in the schematic. If the network is from the layout, use the node name or the node number.

Example

`STA   A1,A2,A3`

END

`END node1, node2, ... node n`

Description

Specifies ending points in the PATH, LPATH, and SPATH commands.

Arguments

`node1, node2, ... node n`

The names of the nodes. If the network is from the schematic, use the names in the schematic. If the network comes from the layout, use the node name or the node number.

MASK

```
MASK .FEEDBACK
      arg1...argn
      (or name01/nameI1,nameI:name02,)
      ....nameOn/nameIn
```

Description

Masks out paths. Use this command to mask out feedback loops that prevent the program from finding critical paths.

Arguments

`.FEEDBACK`

Masks out all feedback loops. Initializes the network to find feedback loops. If there are nodes that cannot be initialized, displays the name of the node.

`arg1...argn`

Specifies one or more of the following forms:

<code>name1,...namen</code>	The name of the node. In this case, output nodes are masked.
<code>name01/nameI1,</code>	The name of the output. It is read as gate name 01;
<code>...nameOn/</code>	one of its inputs is <i>nameI1</i> . In this case, if there are
<code>nameI1</code>	two output nodes with the same name, one of them is marked.
<code>nameI2:name02</code>	The name of the input. It is read as the I2 input pin name of the O2 device name. In this case, only the I2 input pin name of the O2 device name is marked.

Examples

```
MASK .FEEDBACK
MASK A1, B1, OUT1/A2
```


UNMASK

`UNMASK arg1...argn`

Description

Selectively removes a masked node. Use this command to unmask nodes that are currently masked. For example, you can use this command to unmask nodes masked with the MASK .FEEDBACK command.

Arguments

`arg1...argn`

Specify one or more of the following forms:

<code>nameO1/nameI1</code>	The name of the <i>WOR</i> output name. Read as gate name O1; one of its input pins is named I1.
<code>nameI2:nameO2</code>	The name of the input. Read as the I2 input pin name of O2 device name.
<code>name3,...namen</code>	The name of the node when nodes are masked.

Example

`UNMASK S1,B1,I1:O2`

Report Commands

This section describes in detail the report commands for LPEPRO.

PRINT, TYPE, and STORE

PRINT *field 1, field 2, field 3*

TYPE

STORE

Description

Directs your output to the device you choose. PRINT directs output to a printer. TYPE directs output to your screen, and STORE directs output to a file.

Field 1 is the subcommand of PRINT, TYPE, or STORE. *Field 2* can be a subcommand of *field 1* if it begins with a period (.). You can use any of the following subcommands and their arguments with the TYPE/PRINT/STORE commands.

SPICE Subcommand Arguments

SPICE

PRINT/TYPE/STORE the path in SPICE format.

/CAP=*n*

Specifies the threshold value of the capacitance in *pf* over which the capacitance is output.

/SCH

Indicates that schematic names are to be used. If no schematic is available, use layout text. To recover the SPICE node number, use the /SCH option.

SPATH

Outputs the shortest path in SPICE format. Use the SPATH subcommand only after the shortest path timing analysis is performed.

LPATH

Outputs the longest path in SPICE format. Use the LPATH subcommand only after the longest path timing analysis is performed.

:*n1, ...nn*

This field can only be used with the LPATH or SPATH subcommands. *:n1,...nn* are the *n*th path specifications. Prints *n1th, ...nnth* longest or shortest paths.

Dracula Reference

Using Parametric Data (LPEPRO)

Nodenam1, ...Nodenamen

The names of the nodes. Can only be used with the LPATH or SPATH subcommands. Indicates that elements along the longest or shortest paths to *node 1,...node n* are included in the SPICE file.

PATH

Outputs the path in SPICE format.

ALL

Outputs all network information including parasitics and LDD devices in SPICE format. This command is usually reserved to generate the SPICE file of a cell.

SPICE Examples

```
PRINT SPICE PATH
STORE SPICE/SCH/CAP=0.01PF LPATH=1
TYPE SPICE SPATH Q1, QBAR1
```

PATH, LPATH, and SPATH Subcommand Arguments

PATH, LPATH, SPATH

Prints all elements along the longest or shortest paths.

:n

Outputs the first *n* longest or shortest paths.

>m

Prints paths with a delay greater than or equal to *m*.

<1

Prints paths with a delay less than or equal to *1*.

name1, name2, ...namen

The longest or shortest path to the listed nodes.

LPATH Examples

```
PRINT LPATH 20
STORE LPATH Q1, QBAR1, Q, QBAR
```

DELAY Subcommand Arguments

DELAY

Prints the gate delay.

R

Indicates rise delay.

F

Indicates fall delay.

R/F and F/R

The ratios of fall/rise delay, respectively.

>*n*

Prints a gate or output node with delay greater than or equal to *n*.

<*m*

Prints a gate or output node with delay less than or equal to *m*.

name1, . . . , namen

Prints the delay value of the gates specified.

DELAY Examples

```
PRINT DELAY A1,A2
```

```
TYPE DELAY/R/F 1
```

CAPACITANCE Subcommand Arguments

CAP

Prints *node1* capacitance.

/detail

Prints the detailed cross-couple capacitance; otherwise, the lumped capacitance is printed.

[T]

Optional type specification; when *T* is absent, all types are printed. You can specify more than one capacitor type.

Dracula Reference

Using Parametric Data (LPEPRO)

>R1

Prints nodes with capacitance value greater than or equal to *R1*. This value is a real number and *pf* is the unit. When a unit is absent, *pf* is assumed. You can use > or .GE for “greater than or equal to.”

<R2

Prints nodes with a capacitance less than or equal to *R2*. This value is a real number in *pf* units. When a unit is absent, *pf* is assumed. You can use < or .LE for “less than or equal to.”

. RANGE=V1 , V2

Prints nodes with a capacitance range of *V1* and *V2*. The values *V1* and *V2* are real numbers in the *pf* unit.

CAPACITANCE Example

```
TYPE CAP[N,P,X] >0.1
```

MOS Subcommand Arguments

MOS

Prints transistor information including areas and perimeters.

[T]

Optional type specification. When *T* is absent, all types are printed.

/W

Indicates that only the transistor width is checked.

/L

Indicates that only length is checked. When both are missing, both width and length are checked.

/W/L

Indicates that the ratio of width to length is checked.

/L/W

Indicates that the ratio of length to width is checked. The specification can only be used with *R1* and *R2* and .RANGE =*V1*,*V2*.

Dracula Reference

Using Parametric Data (LPEPRO)

>R1

Prints transistors whose length or width is greater than or equal to $R1$; or whose width/length or length/width ratio is greater than or equal to $R1$. You can use > or .GE for “greater than or equal to.”

<R2

Prints transistors whose length or width is less than or equal to $R2$, or whose length/width, width/length ratio is less than or equal to $R2$. You can use < or .LE for “less than or equal to.”

node1, . . . node_n

Prints information about transistors connected to these nodes.

.WINDOW= (X1, Y1) / (X2, Y2)

W/L information of transistors inside the (X1,Y1)/(X2,Y2) window specified by $X1, Y1$ and $X2, Y2$ are printed. Parentheses are optional, the slash (/) is required. $X1, Y1, X2$, and $Y2$ use a natural unit.

.DEVICE = n1, n2, . . .

W/L information of transistors $n1, \dots, nn$ are printed; $n1, \dots, nn$ are internal device numbers used in the program.

.RANGE = V1, V2

Prints a node with W/L information in the range $V1, V2$. The values $V1, V2$ have the same units as $R1$ and $R2$.

MOS Examples

```
TYPE MOS[D]/W >0
```

```
PRINT MOS .WIND = 100,100/-100,-100
```

DIODE Subcommand Arguments

DIODE

Outputs diode information.

[T]

Optional type specification.

/option

Value of A1, A2, A3 or P1, P2, P3. Use this option with $R1, R2$, or $.RAN=V1, V2$.

Dracula Reference

Using Parametric Data (LPEPRO)

>R1

Prints diode information if the parameter is greater than or equal to *R1*. You can use > or .GE for “greater than or equal to.”

<R2

Prints diode information if the parameter specified by */option* is less than *R2*. You can use < or .LE for “less than or equal to.”

node1, . . . node_n

Prints diode information from the specified nodes.

.WINDOW= (X1, Y1) / (X2, Y2)

Window specification. Prints diode information for diodes inside window (X1,Y1)/(X2,Y2). Parentheses are optional. Slash (/) is required. *X1,Y1* and *X2,Y2* use a natural unit.

.DEVICE=n1, n2, . . . nn

Outputs diode information with the given device numbers.

.RANGE=V1, V2

Prints nodes with parameters in the range of *V1* and *V2*.

DIODE Examples

```
TYPE DIO[D] A1,B2,IN1
PRINT DIO[D] .WINDOW=(100,200)/(230,300)
```

PARAMETER Subcommand Arguments

PARA

Checks the parameter calculated by the CALC command.

/U, /D, /U/D, /D/U

Prints the ratio parameter value. */U* and */D* represent the pull-up parameter value and the pulldown parameter value, respectively. */U/D* represents the ratio of the pull-up parameter value to the pulldown parameter value and */D/U* is the reverse.

>R1

Prints nodes whose parameters, calculated by the CALC command, are greater than or equal to *R1*. You can use > or .GE for “greater than or equal to.”

Dracula Reference

Using Parametric Data (LPEPRO)

<R2

Prints nodes whose parameters are greater than or equal to *R2*. You can use < or .LE for “less than or equal to.”

node1, . . . node_n

Prints parameters of the specified nodes.

.WINDOW= (X1, Y1) / (X2, Y2)

Window specification. Direct the program to bring (X1,Y1)/(X2,Y2) the parameter information with this window. Parentheses are optional, slash (/) is required. *X1, Y1* and *X2, Y2* use a natural unit.

.DEVICE=*n1, n1, .nn*

Outputs parameter information of MOS *n1, n2, ...nn*.

.RANGE=*V1, V2*

Prints the node whose parameter is within the range *V1* and *V2*. *V1* and *V2* are real numbers.

PARAMETER Examples

```
TYPE PARAM >21
```

```
PRINT PARAM .WIND = 100,100/-100,-100
```

PRINT and TYPE are recognized only after you have specified the parameter equation.

MASK Subcommand

MASK

Prints the names of masked nodes or pins.

START Subcommand

START

Displays all specified starting nodes.

END Subcommand

END

Displays all specified ending points.

PATH Subcommand

PATH

Prints all elements between your specified starting and ending points.

UNREACH Subcommand

UNREACH

Prints any element that cannot be reached by input signals.

Scanning Layout Parameters

LPEPRO lets you review MOS, DIODE, CAPACITANCE, or user-defined parameters. The commands you use to review parameters start with PRINT, TYPE, or STORE.

The DISK command changes the file name in the STORE command. After you issue a PRINT, TYPE, or STORE command, the following become subcommands:

MOS
DIODE
CAPACITANCE
PARAMETER
RES
LDD

To review the above subcommands, use one of the following values:

.GT.GE
.LT.LE
.RANGE

You can also select the parameters using node names or node numbers. For parameters that involve the location of MOS, DIODE, or PARAMETER elements, use the .WINDOW command to select a window.

Identifying Nodes in the Layout

The LPEPRO program uses names, if available, to report results. If names are not available, the scanner uses an internal node number. There are two ways you can specify node names in the layout database:

- Texting the layout

■ Providing the schematic data

If you specify the GET SCHEMATIC command, LPEPRO uses elements or nodes on the schematic data, including node names. Without the GET SCHEMATIC command, LPEPRO assumes that only layout data is provided. In this case, only those nodes with text have node names. LPEPRO assigns an internal node number to all of the other nodes.

The internal node numbers assigned by the LPEPRO program are difficult to identify. To solve this problem, do the following:

1. To zoom into the relevant transistors so you can identify the node numbers, use the .WINDOW command. Based on the known numbers, you can identify the internal node numbers.
2. To display the network connection, use the TYPE MOS command. After the relationship of the connection is established, you can trace a node number.
3. Plot the relevant nodes.
4. Label the significant nodes.
5. Provide the schematic data.

MOS Subcommand

The MOS subcommand lets you check transistors for width, length, width/length ratio and length/width ratio. You select these values by specifying their value range. LPEPRO assumes that width and length are in natural units and ignores the unit you specified in the command. To zoom into an area, use the .WINDOW command.

Example 1

```
:TYPE MOS[D]/W .LT 4
```

Checks any transistor with a width of less than 4.

Arguments

TYPE

Directs output to a terminal.

MOS [D]

Specifies that only D-type MOS are checked.

Dracula Reference

Using Parametric Data (LPEPRO)

/W

Specifies that width is checked. There can be no blank spaces between the bracket (]) and the slash (/).

.LT

Specifies “less than.”

4

Indicates 4 natural units.

Example 2

```
:PRI MOS[E]/L/W .GE 10
```

Checks any transistor with an L/W ratio greater than or equal to 10.

Arguments

L/W

Indicates that the length/width ratio is checked. No blank spaces are permitted between slashes (/).

Example 3

```
:TYPE MOS/W/L .RANG=10,20
```

Checks any transistors whose W/L ratio are between 10 and 20.

Example 4

```
:TYPE MOS .WIND=100,200/200,100
```

Finds the transistors in the X1=100, Y1=200, X2=200, Y2 =100 window. *WIND* is the keyword of the WINDOW command. X1, Y1 and X2, Y2 immediately follow the equal sign (=). A slash (/) is required to separate the two coordinates.

Example 5

```
:TYPE MOS A1,B10,20
```

Dracula Reference

Using Parametric Data (LPEPRO)

Finds all transistors connected to *A1*, *B10*, *20* nodes, where *A1* and *B10* are either schematic names or names put in the layout database. In this example, *A1* is not interpreted as a keyword because it does not have a period (.).

The outputs of all of these examples are similar. Each MOS is printed in the following format:

Mnumber Drain, Gate, Source, Substrate, Model,
L=length, W=width, X=x-coordinate, Y=y-coordinate

For W/L or L/W ratio, the ratio is printed before each MOS statement as a comment. For example,

```
* = 3.86  
M10 GND, 9, 8, GND, NCHNL, L=0.35, W=2.00  
X=-2.99, Y=-25.99
```

The node name is used when it is available.

CAPACITANCE Subcommand

The CAPACITANCE subcommand lets you review nodal capacitance. If the range of the capacitance value is specified, use the *T* option to select the capacitance type. Specify more than one type of capacitance inside brackets ([]). For example, CAP[N,P,X,Y] directs LPEPRO to select *N*, *P*, *X*, or *Y* types of capacitors only. LPEPRO sums up all capacitances without the *T* option or selected capacitances with the *T* option. To obtain detailed cross-couple information, use the /DETAIL option.

Example 1

```
:TYPE CAP .GT 1
```

Displays nodes with capacitances exceeding 1 *pf*.

Example 2

```
:PRINT CAP[N,P] .RANG=1,3
```

Prints node names with capacitances between 1 *pf* and 3 *pf* of *N* and *P* types.

Example 3

```
:STOR/DET CAP A1, B1
```

Detailed information of the nodal capacitance of *A1*, *B1* nodes is stored on disk. The output has the following format:

```
A1/node1=npf :NA1/node2 = mpf :X  
A1/node3=upf :PA1/node4 = lpf :Y  
***TOTAL CAPACITANCE = R pf***
```

In addition to the total capacitance *R pf*, the capacitance between *A1* and *node1*, *A1* and *node2*,... and their corresponding capacitance type are displayed. If no /DETAIL option is specified, only the total capacitance for *A1* and *B1* nodes is stored.

DIODE Subcommand

The DIODE subcommand lets you to review diode information. The options for this subcommand are similar to those for the MOS command. Instead of width and length, there are six parameters: A1, P1, A2, P2, A3, P3. for more information about using this subcommand, refer to the ELEMENT MOS section in Chapter 13.

PARAMETER Subcommand

The PARAMETER subcommand lets you specify the arithmetic equation of W, L, and C parameters extracted from the layout. LPEPRO checks the parameter, calculates according to the equation provided, and displays those elements with parameters that fall within your specification. Specify the arithmetic equation with the CALCULATE PARAMETER command, then use the TYPE PARAMETER command to review the result.

Arithmetic Equations

To specify the equation for the parameter, use the following command:

```
:CAL PAR = arithmetic equation
```

You can use the following symbols in the equation:

W	Width of transistor in a natural unit
L	Length of transistor in a natural unit
C	Nodal capacitance in <i>pf</i>
*	Multiplication
/	Division
+	Addition
-	Subtraction
(	Left parenthesis
)	Right parenthesis

You can include a number, real number, or integer in the equation. For example

```
:CAL PARA = 100*(L/W)*C+10.2
```

Dracula Reference

Using Parametric Data (LPEPRO)

In this case, the program takes the ratio of length to width, multiplies it by 100, multiplies it by the capacitance value, then adds 10.2.

LPEPRO assumes that width and length are in natural units and capacitance is in *pf* when it calculates the parameter. LPEPRO does not assign any unit to the result. You specify the equation and the interpretation. Once you provide the equation, you can review it as you would any other parameter.

Reviewing Parameters

To review parameters, use the PRINT, TYPE, or STORE PARAMETER command. These commands are similar to the MOS subcommand. All of the subcommands are the same, except for the /U, /D, /U/D, and /D/U options. For details about these options, refer to the Terminal Commands section in this chapter.

Example

```
:CAL PAR=(L/W)*C
:TYP PAR/U/D A1
```

This calculates and types the ratio of the pull-up parameter value to the pulldown parameter value for the *A1* node. Parameter values are computed according to your formula.

```
:TYP PAR/U/D <3
```

This types all nodes whose pull-up to pulldown parameter ratio values are less than or equal to 3.

If you use the /U, /D, /U/D, or /D/U options, a PARASITIC CAP command must be specified so that every node in the circuit has a parasitic capacitor calculated for it. Otherwise, not all nodes are reported.

SPICE Generator

LPEPRO can generate a SPICE file for simulation purposes. The SPICE file can be the whole chip, a whole cell, or selected paths. You can specify a capacitance threshold value. The SPICE file includes only capacitances larger than the threshold value.

Whole Chip or Whole Cell

To generate a SPICE file for a whole circuit, use the TYPE command.

```
:TYPE SPICE/CAP=0.5 ALL
```

Dracula Reference

Using Parametric Data (LPEPRO)

In the previous example, LPEPRO displays MOS information. The optional `/CAP=0.5` specification displays only nodes with capacitances over 0.5 *pf*. Diode information follows the capacitance. There are two difficulties with this command:

- The command produces too much information and internal node numbers are used throughout the data.
- It is almost impossible to correspond these node numbers to node names in a schematic.

You can avoid generating too much information by not using this command for an entire chip. Use the `TYPE SPICE ALL` command to generate a SPICE file for a small cell, but do not use it for an entire chip.

If you provide schematic data, you can avoid the second problem by using the following commands:

```
:GET SCH
:TYPE SPICE/SCH/CAP=0.5 ALL
```

In this case, if you use the `/SCHEMATIC` option, node names from the schematic are used. These node names can be names used in your logic network data or in your original SPICE file. This SPICE file can be output as the reference for the simulation.

Use the `DISK` command to store the SPICE deck to a specified file name.

Selected Paths

If you use selected paths, you can select some elements from the chip and generate the SPICE file for them. Use one of the following methods:

- Elements between your specified starting and ending nodes
- Longest paths in the LPEPRO path analysis
- Shortest paths in the LPEPRO path analysis

For a description of longest/shortest path analysis, refer to the LPATH and SPATH commands in this chapter. The SPICE commands are as follows:

```
:TYPE SPICE/CAP=0.5 PATH or  
:TYPE SPICE LPATH :3,5,10 or  
:TYPE SPICE SPATH :10
```

You can use these commands whether you have schematic data, SPICE data, or no schematic data.

Schematic Data

For schematic data, prepare the file as follows:

1. To save the compiled logic network after LOGLVS compiles and before it converts the logic data in the transistor file, issue the following command:

```
:SAVE LOGIC
```

then enter

```
:CONVERT
```

2. To recover the schematic data, use the GET SCHEMATIC command. This command reads the compiled logic network that was saved when you executed LOGLVS.
3. To mask out (or cut) the feedback loops, use the MASK .FEEDBACK command whenever necessary. LPEPRO identifies feedback loops and cuts them at the proper places. If after several iterations there are still nodes that cannot be initialized by an input signal, LPEPRO displays these nodes on your screen. Sometimes LPEPRO cuts the feedback loop in the wrong place.
4. To review nodes that are masked by LPEPRO, use the TYPE.MASK command.
5. To add new nodes, use the MASK command. To change masked nodes, use the UNMASK command.
6. To specify starting nodes and ending nodes, use the START and END commands. For example,

Dracula Reference

Using Parametric Data (LPEPRO)

```
:START A1, D1, A2,...  
:END B1, OUT1, OUT2
```

Where *A1*, *D1*, and *A2* are the starting nodes and *B1*, *OUT1*, and *OUT2* are ending nodes. These node names should be consistent with the names in your logic database.

7. To find the elements along the starting and ending nodes, use the PATH command.
8. To review the result, use the TYPE PATH command. An element is included in the path if it appears on both the forward and backward tracing of the PATH command.
9. Display the SPICE file on your screen as follows:

```
:TYPE SPICE PATH
```

If this produces what you want, use the STORE command to store the results. (You can change the file name with the DISK command.)

10. To screen out unwanted capacitances, use the /CAP=valve option.
11. To select another path or paths, use the RESET PATH command.
12. Repeat steps 5 through 7 for another selection.

No Schematic Data

If there is no schematic data or if the schematic data is SPICE, proceed as follows:

1. To read the transistor level file and reconstruct the logic network from the layout, use the GET LAYOUT command.

For information on how LPEPRO handles this process, refer to the [Logic Gate Generator](#) section in this chapter. For SPICE data and original node names in SPICE, use the following command:

```
:GET LAYOUT SCH or GET SPICE
```

SCH indicates the data are from the schematic. LPEPRO reconstructs the network and recovers node names or numbers you used in the original SPICE deck.

2. To cut the feedback loops, use the MASK .FEEDBACK command.

Because LPEPRO might not be able to form gates from some transistors, many discrete components might exist. LPEPRO might not be able to cut all feedback loops at the proper places.
3. To reselect the network loops to be cut, use the UNMASK and MASK commands.
4. To select the elements you want to include in the SPICE output, follow steps 5 through 8 in the [Schematic Data](#) section of this chapter.

Logic Gate Generator

If you do not have schematic data or if your schematic data is in SPICE format, follow the instructions in this section. The logic gate generator recognizes only predefined primitive gates and handles only MOS technology. A primitive gate should have the following properties:

- Single output
- One stage between inputs and output node

For unrecognizable networks, the program forms parallel pull-up, parallel pulldown, serial pull-up, or serial pulldown devices whenever possible, or it forms an N-type or P-type unidirectional transfer gate. As a last resort, the program uses an MOS device that has bidirectional properties. Once the logic network is reconstructed, selected paths can be formed. In most cases, selected path timing is done.

Using the Logic Gate Generator

You can generate a logic network from a SPICE file or from the layout.

When you generate the logic network from a SPICE file, every node has a number that the system treats as the node name. When the logic network is generated from the layout, only labeled nodes have names and Dracula assigns a number to unlabeled nodes. After the logic network is created, the naming conventions are as follows:

- Program-assigned name: N-number is the internal node number. For example
- User-assigned name: The name in a SPICE file or I/O pad names. For example

N-301 Node 301

101 Node number 101 in SPICE data file.
P1 Name used as label in the layout.

To generate the logic network from the layout, follow these steps:

1. Use the GET LAYOUT command, or if you want to create a logic gate from the SPICE file, use the GET LAYOUT SCHEMATIC command or the GET SPICE command.

Assuming that your SPICE file is the exact image of the layout, the difference between these two commands is that the logic network generated from a SPICE file has your user-assigned node names, while the network generated from layout has program-assigned node names (except for labels).

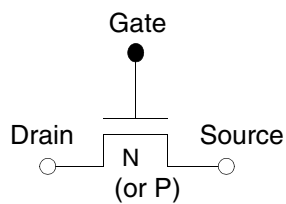
2. To create a network description file, use the STORE command.
3. Change the file name in the STORE command with the DISK command.

4. After the logic network is generated, select path and perform path timing analysis.

Gate Types

All gate types generated by the logic gate generator are listed below.

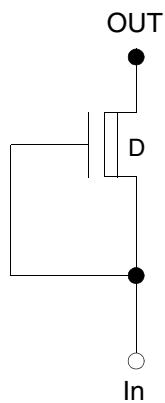
NMOS (or PMOS)



RES: Resistor

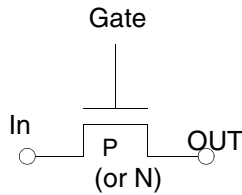
Used for NMOS only. A transistor is converted to a RES if

- Transistor is type D and not type E
- Gate is connected to source



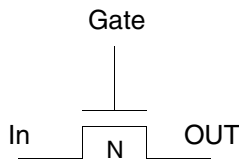
NXFR (or PXFR): N or P Type Transfer Gate

A MOS device is converted to NXFR (or PXFR) if the directionality of the MOS can be determined. Type N in a CMOS circuit or type E in an NMOS circuit is converted to NXFR. Type P in a CMOS circuit is converted to PXFR.



NRXF

Used for NMOS only. Resistive N-type transfer gate. A MOS is converted to NRXF if it is a non-E type in an NMOS circuit.

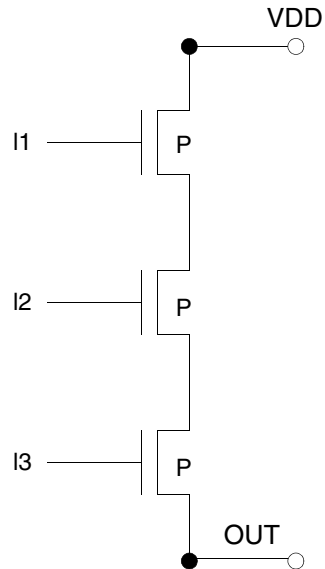


SUP: Serial Pull-Up Device

The cell name is *SUP n* where n is the number of a pull-up MOS gates.

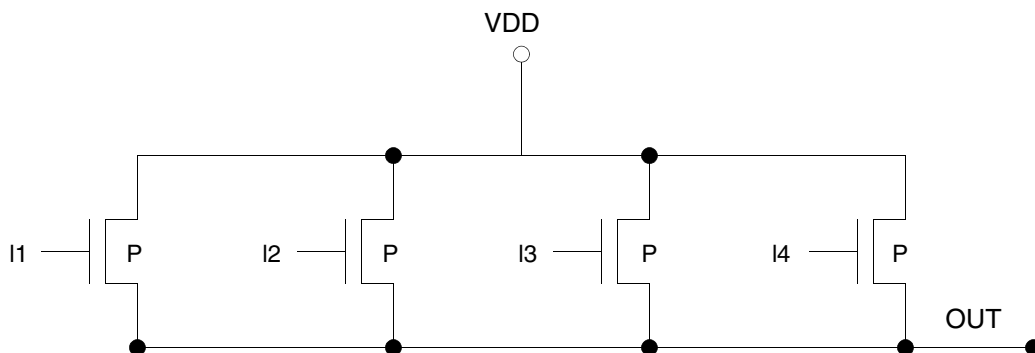
Dracula Reference

Using Parametric Data (LPEPRO)

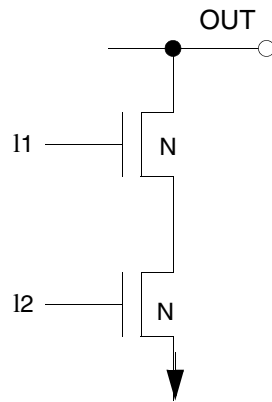


PUP: Parallel Pull-Up Device

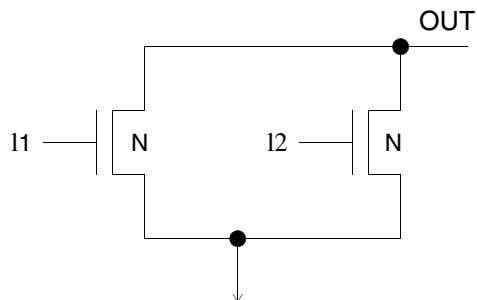
The gate name is PUP_n where n is the number of MOS gates.



SDW: Serial Pull-Down Device



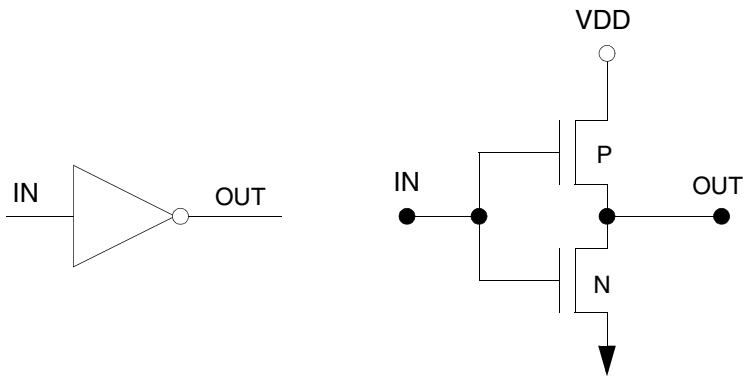
PDW: Parallel Pull-Down Device



Dracula Reference

Using Parametric Data (LPEPRO)

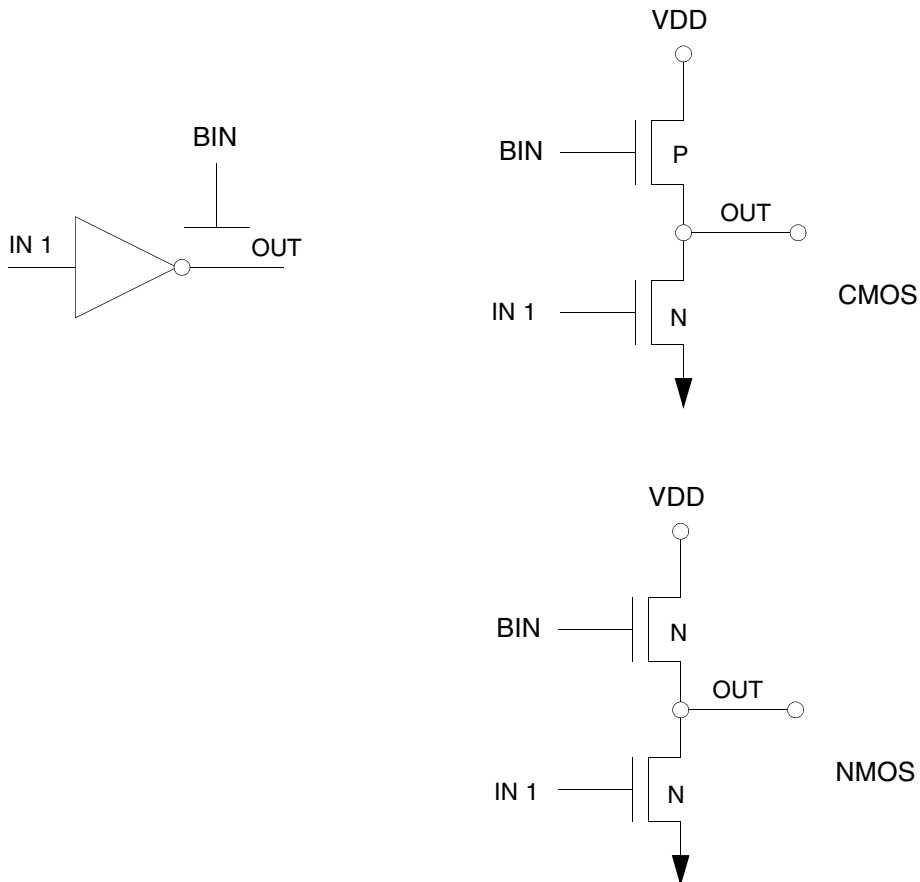
INV: Inverter



Dracula Reference

Using Parametric Data (LPEPRO)

BINV: Buffer-Inverter

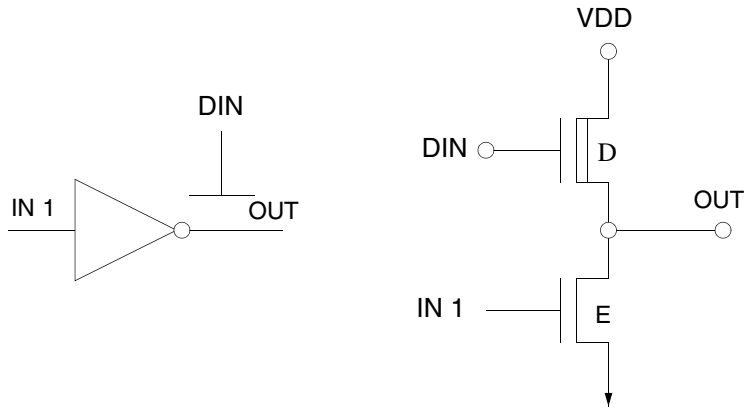


DINV: D-type Buffer-Inverter

Used for NMOS only.

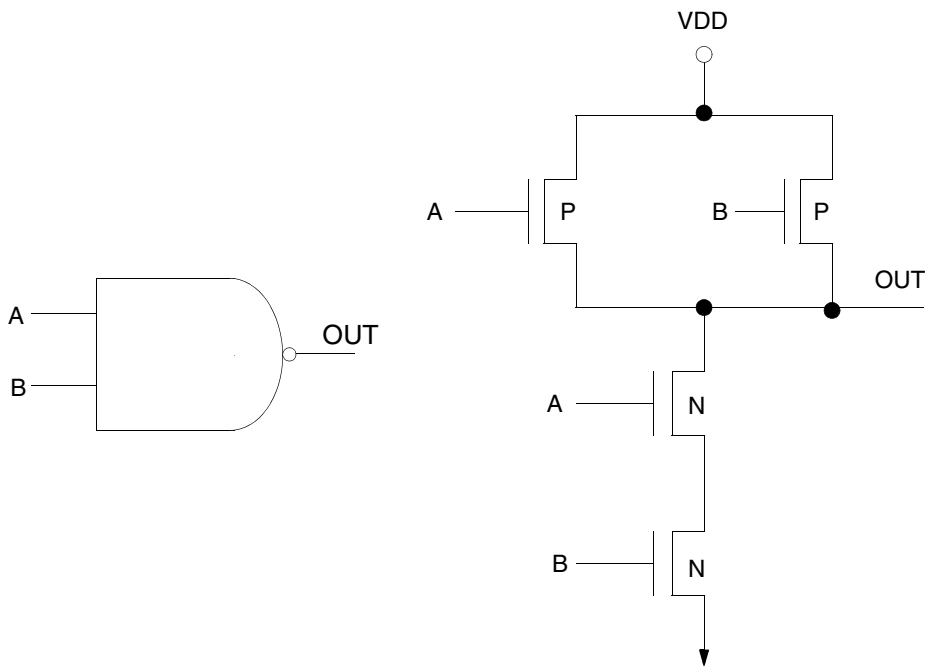
Dracula Reference

Using Parametric Data (LPEPRO)



NAND

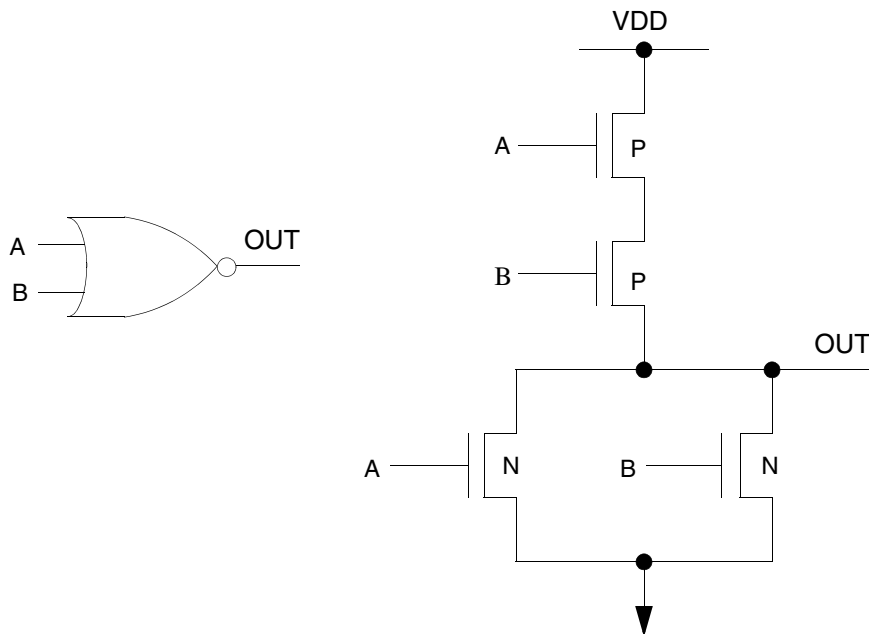
For NMOS, only the depletion mode MOS, type D, is used to form the gate. Other types of pull-ups are not used.



Dracula Reference

Using Parametric Data (LPEPRO)

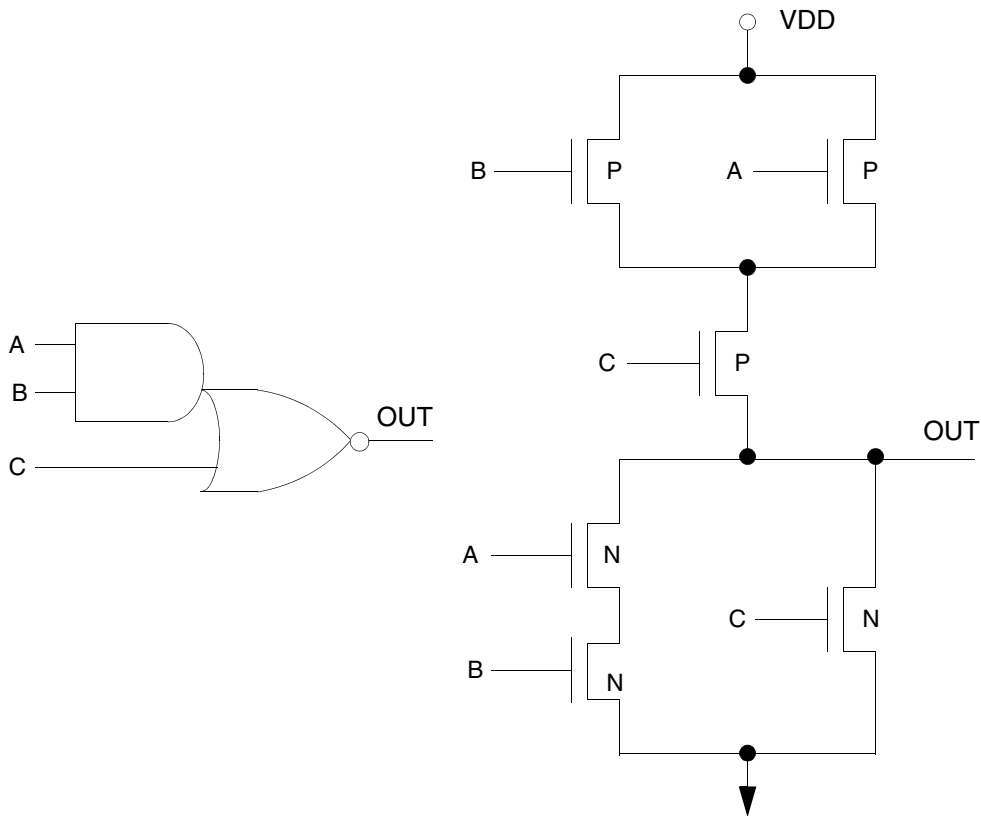
NOR



Dracula Reference

Using Parametric Data (LPEPRO)

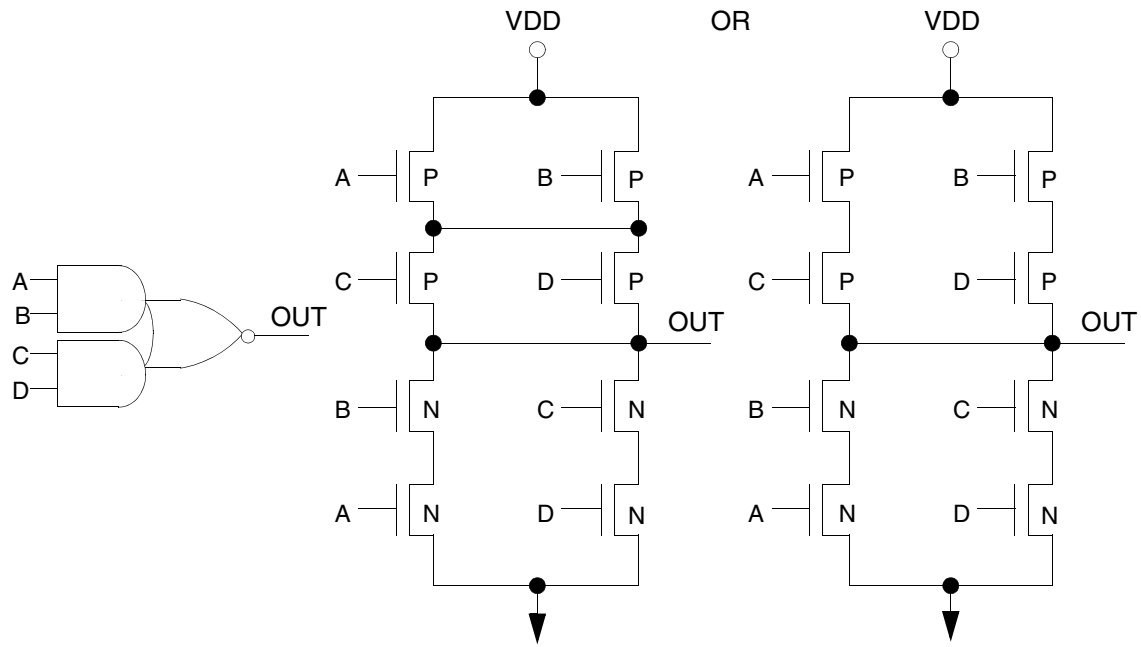
AOI1: AND-OR Inverter Type 1



Dracula Reference

Using Parametric Data (LPEPRO)

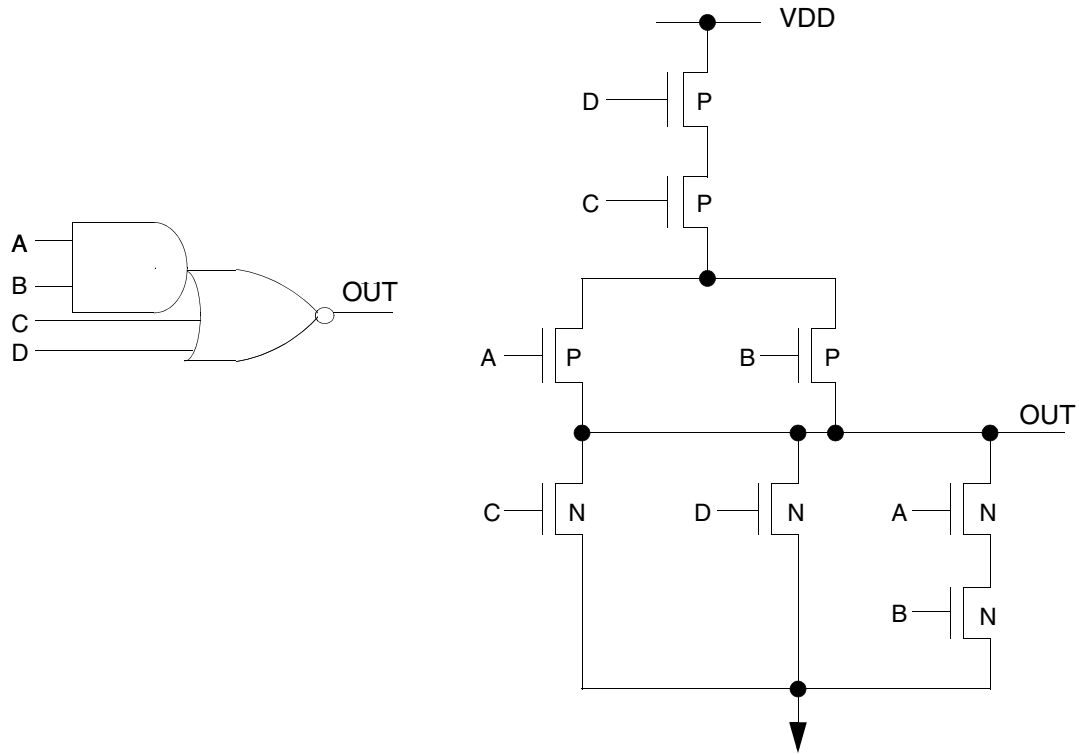
AOI2; AND-OR Inverter Type 2



Dracula Reference

Using Parametric Data (LPEPRO)

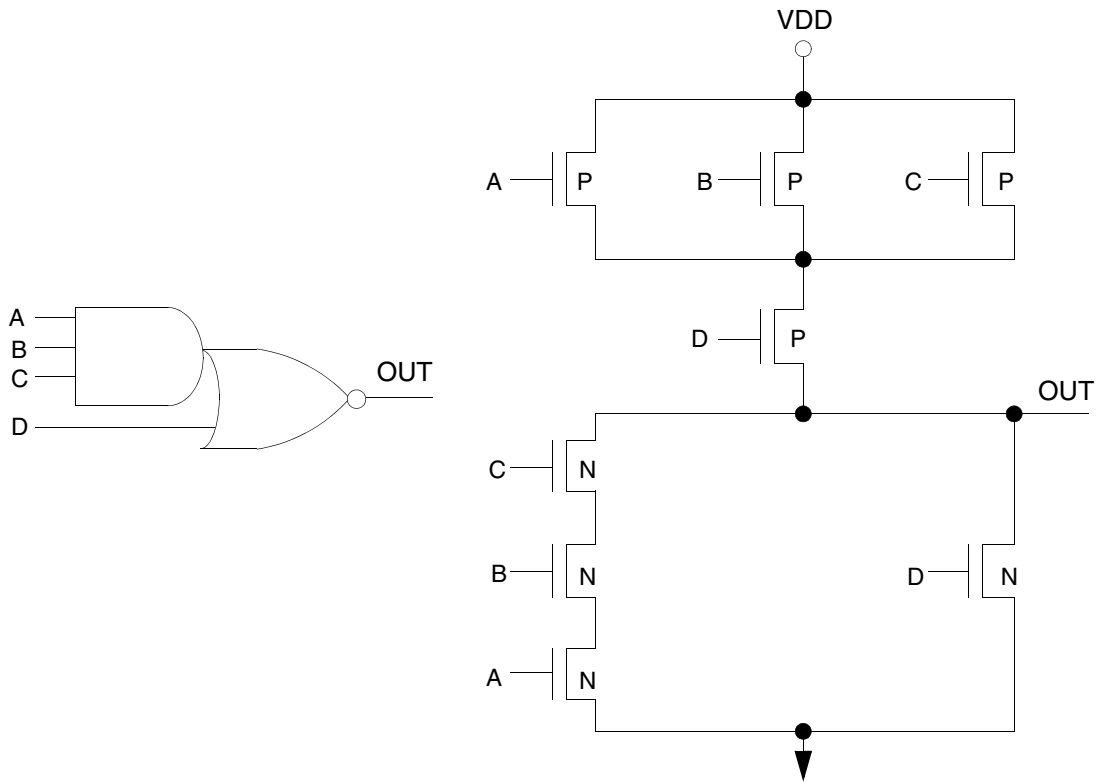
AOI3; AND-OR Inverter Type 3



Dracula Reference

Using Parametric Data (LPEPRO)

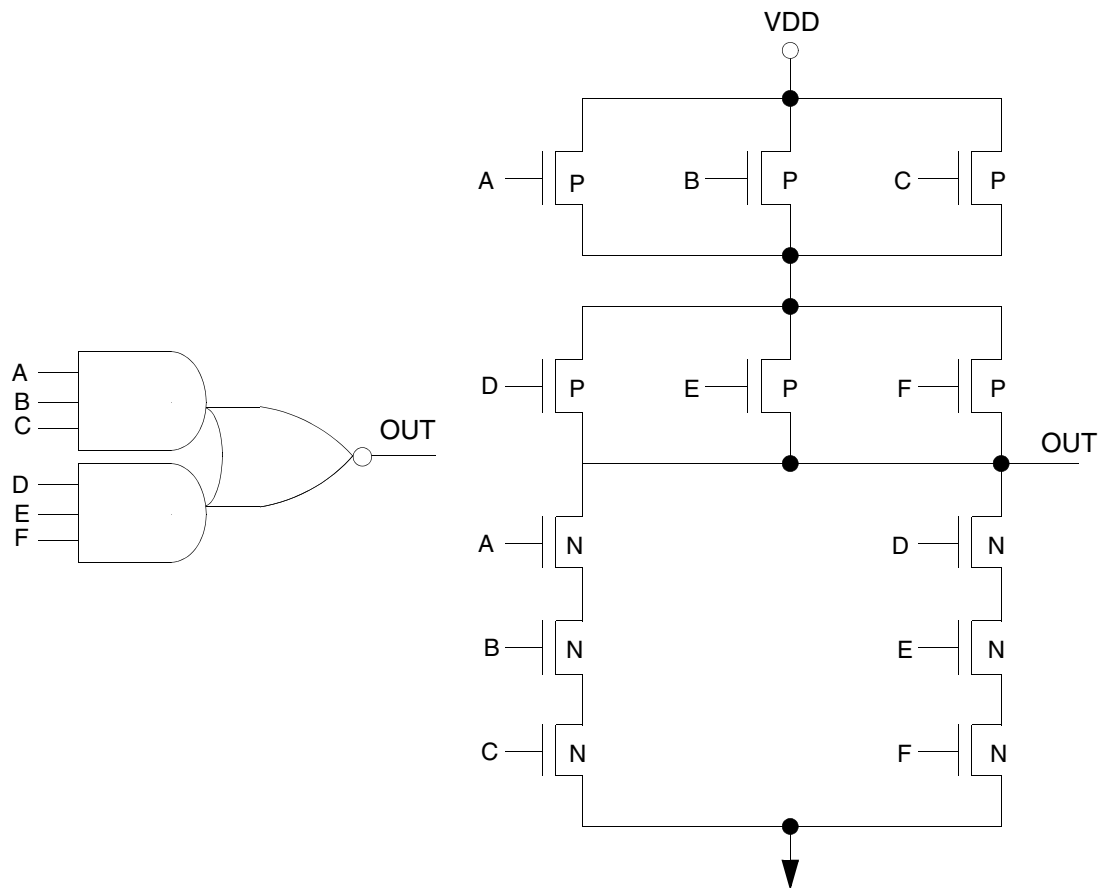
A0I4: AND-OR Inverter Type 4



Dracula Reference

Using Parametric Data (LPEPRO)

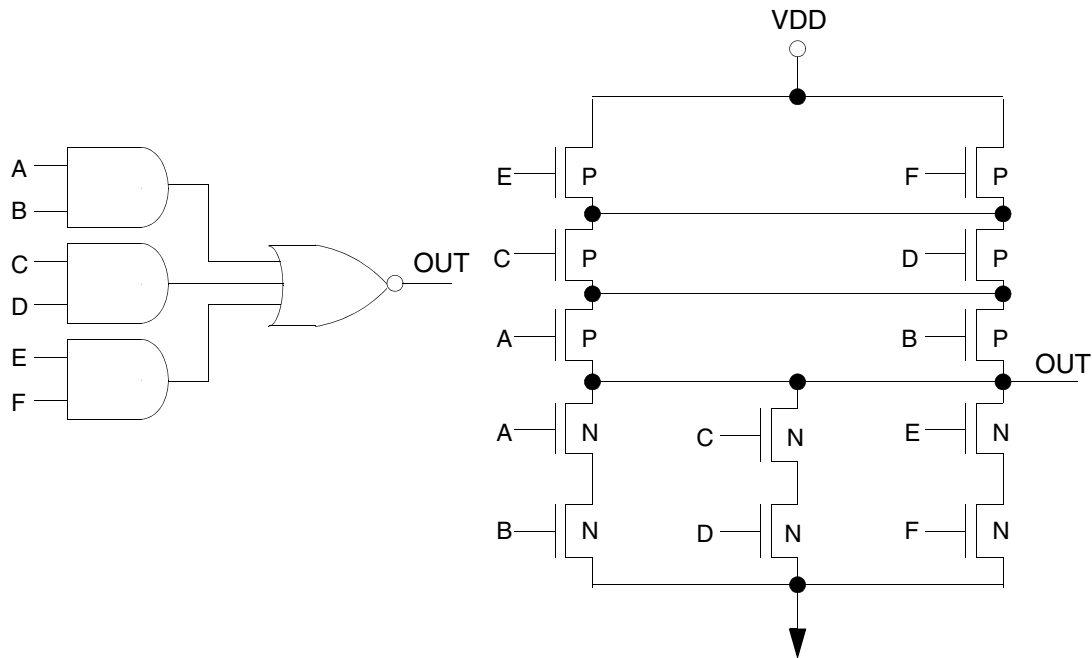
AOI5: AND-OR Inverter Type 5



Dracula Reference

Using Parametric Data (LPEPRO)

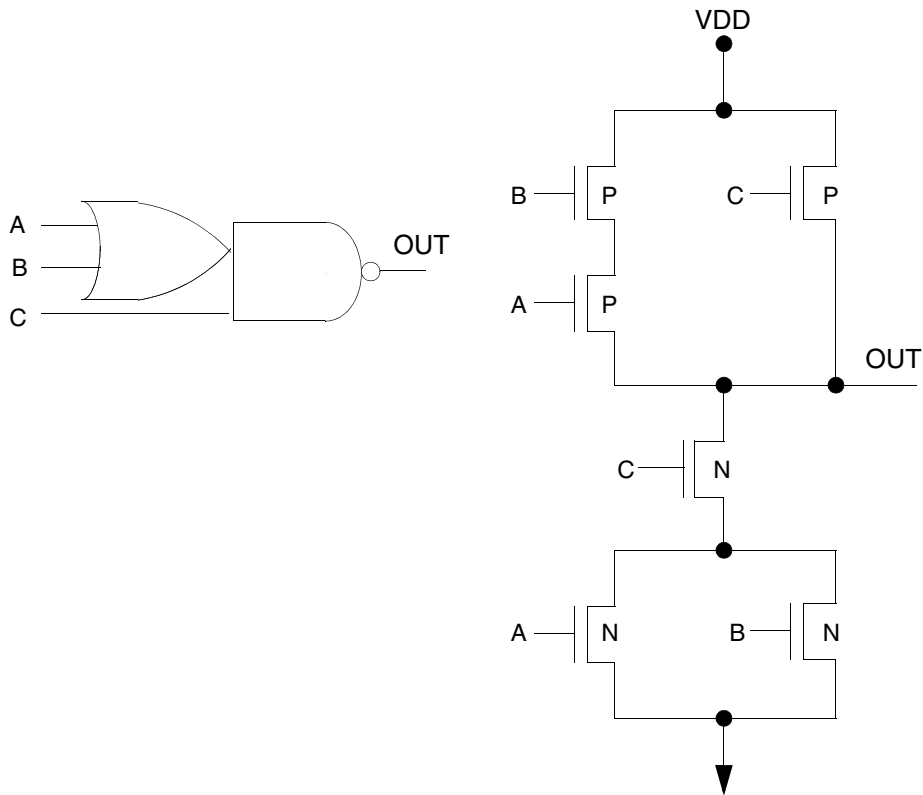
AOI6: AND-OR Inverter Type 6



Dracula Reference

Using Parametric Data (LPEPRO)

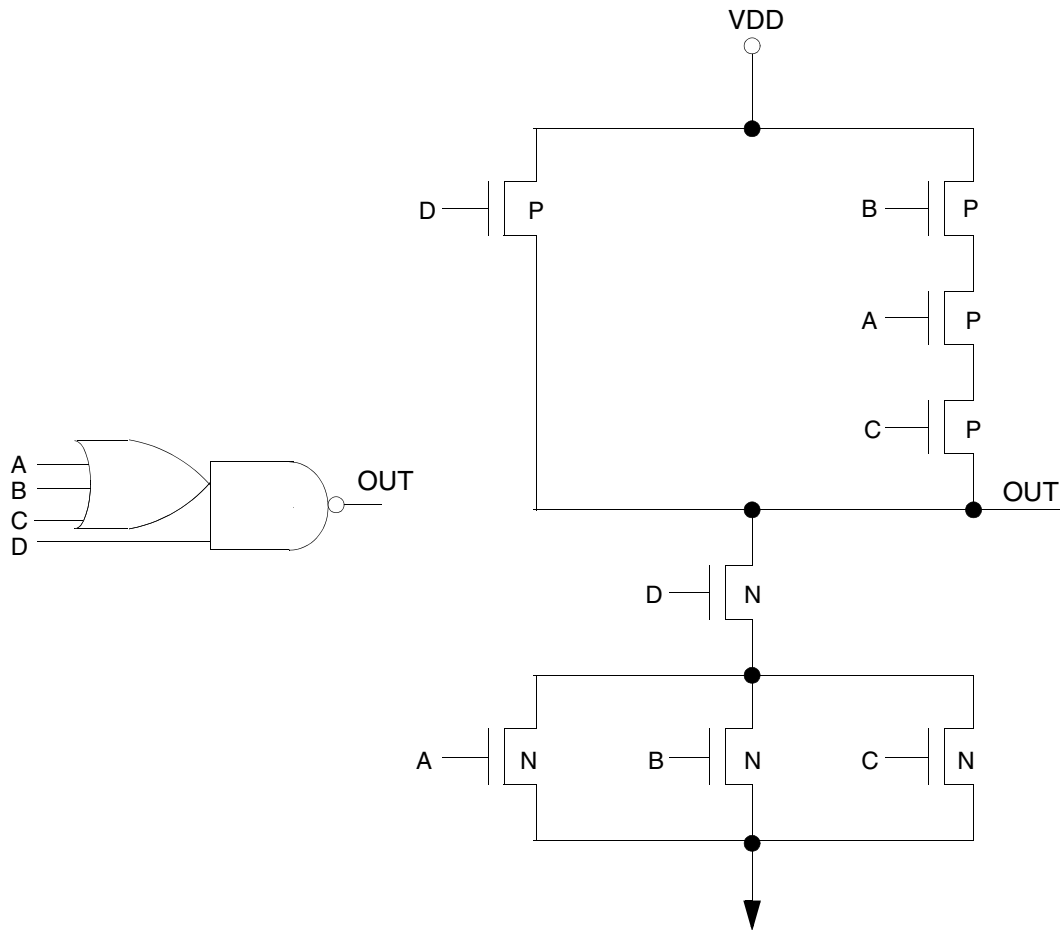
OAI1: OR-AND Inverter Type 1



Dracula Reference

Using Parametric Data (LPEPRO)

OAI2; OR-AND Inverter Type 2

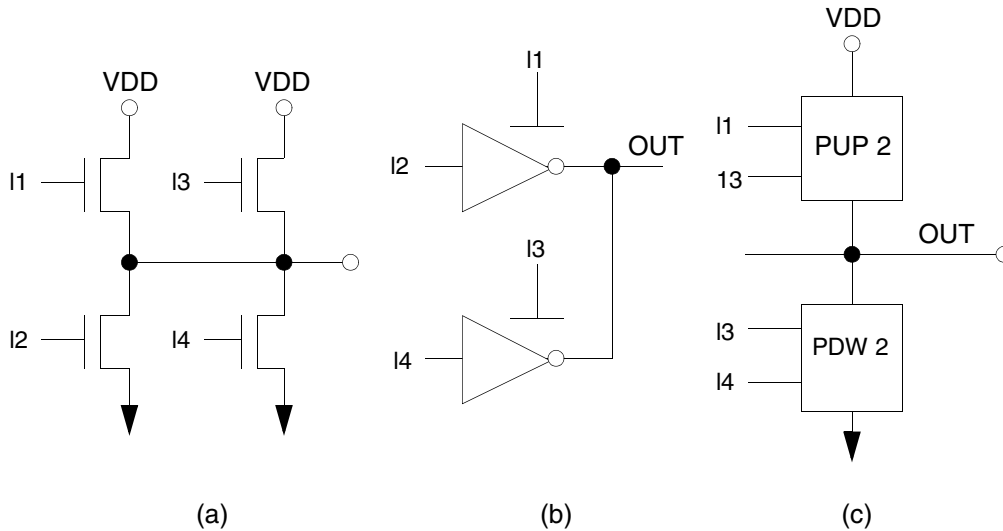


Wired-OR Examples

Wired-OR (or wired-AND) is defined as outputs of two or more connected gates. In the case of wired-ORs and wired-ANDs, the LPEPRO program does not recognize individual gates and creates a parallel up and parallel down structure. If the circuit is too complicated to form a parallel-up/parallel-down structure, transfer gates are used instead.

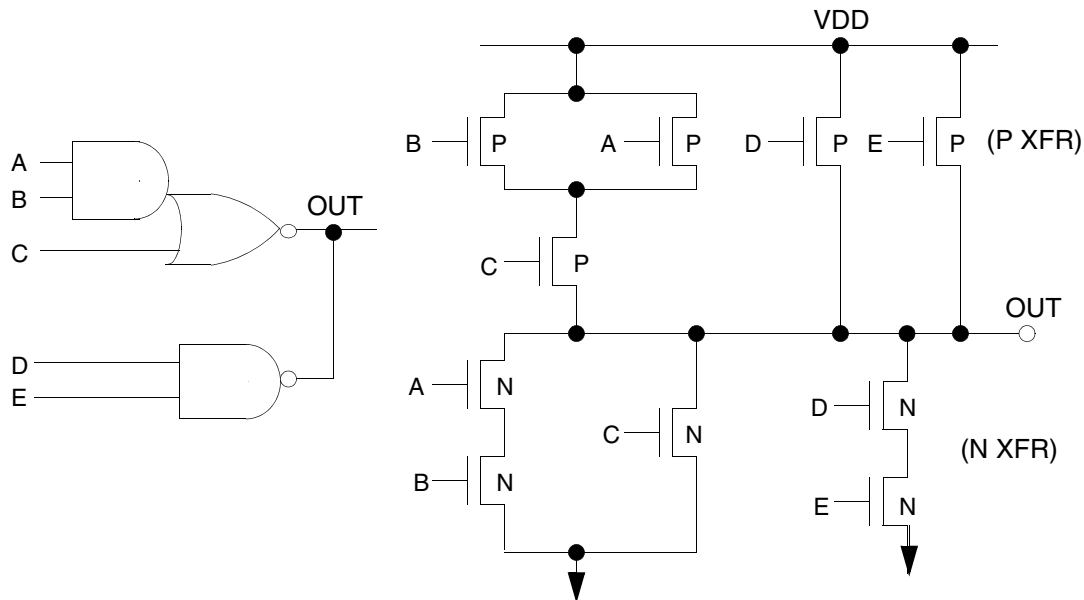
Two Buffer-Inverters Wired-OR

The circuit in (a) is converted to a two-input parallel pull-up and a two-input parallel pulldown (c) instead of two buffer-inverters as in (b).



AOI1 and NAND Gate Wired-OR

When a circuit is more complicated than a simple, parallel/serial structure, the program does not convert transistor circuits into gate equivalents and uses PXFR and NXFR instead.



Calculating Delay

An important function of length (L), width (W), and capacitance (C) of MOS transistor layout parameters is to calculate gate delay. Timing verification and logic simulation are calculated with the first order of delay information.

LPEPRO uses an arithmetic equation based on L/W and C in delay calculation. The effective L/W ratio is used for serial/parallel structures. Capacitance is defined as the total nodal capacitance (measured in picofarads - *pf*) of the output node of the gate.

To conform with most logic simulators, there are four types of delays: rise delay, fall delay, on delay, and off delay. The rise and fall delays are for primitive gates. On and off delays are for transfer gates and MOS only. Delays are shown in integers, and you must define a unit for your application.

In calculating delay, you can selectively update one or two delay types. You can use rise and fall delays to check the pull-up and pulldown ratios. If you provide the logic schematic, LPEPRO puts the delays back into the expanded version of your original netlist and you can check selected paths.

L/W Ratio, Capacitance, and Resistance

The effective L/W ratio of parallel/serial structures is as follows:

■ Serial structure

For each MOS device, the R ratio is defined as

$$R = W/L$$

If there are n MOS device, each with a different W/L ratio, $R1, \dots Rn$, the effective RE ratio is

$$R = 1 / (1/R1 + \dots + 1/Rn)$$

■ Parallel structures

If there are parallel structures, the effective W/L ratio RE is

$$RE = \text{Min} (R1, \dots Rn)$$

■ Parallel/serial structure

If there is a composite structure with both parallel and serial connections, start from the innermost connection. The program iterates to resolve the effective W/L ratio using the criteria described.

Nodal capacitance delays calculated by LPEPRO are only the first order approximation of circuit timing response because the program is not designed to handle detailed timing analysis. Capacitance in the calculation is defined as the total nodal capacitance of the output node of a gate.

Using Delay Equation Options

When you specify a delay equation, use these options to update different types of delays or the delays of selected gates. Rise and fall delays are associated with logic gates, and on/off delays are associated with transistors. The calculation options are as follows:

/RISE

Calculates rise delay

/FALL

Calculates fall delay

/ON

Calculates the turn-on delay of P-type MOS (input to output)

/XON

Calculates the gate-to-output delay of P-type MOS

/OFF

Calculates the turn-off delay of P-type MOS (input to output)

/XOFF

Calculates the gate-to-output delay of N-type MOS

/MOS[T]

Calculates delay of a MOS type T gate

/RES[T]

Calculates delay of a MOS type T resistor /GATE:TYPE1, Calculates delays of Type1 to TypeN gates

...,TYPE N

If you don't specify an option, delays of every gate or transistor are calculated. You can specify more than one option at the same time. The program uses your options and updates the delays.

/RISE and /FALL options calculate rise delay and fall delay respectively. Transistor delays, whether in the format of transfer gate or bidirectional MOS devices, are not calculated.

/ON, /XON, and XOFF options calculate the turn-on and turn-off delay of a transfer gate, CMOS transfer gate, bidirectional MOS, or CMOS bidirectional gate. In most cases, the equation used in on/off delays is different from the equations used in rise/fall delays. With these options, you can distinguish between them for greater accuracy in determining the delays.

/MOS[T] specifies delay equations for different types of MOS. This option is often used in NMOS where there might be different types of depletion modes. Different types of MOS have different timing responses and sometimes it is necessary to distinguish them with this option.

/RES[T] specifies the delay of NMOS gates that have resistive pull-up devices. Any MOS whose gate connects to either drain or source is an MOS with resistive properties.

/GATE:TYPE1,...,TYPE n assigns delay equations to different types of gates. Use this option mainly for schematic data when some of the gates are not primitive and for standard cells and the L/W ratio. The only parameter is capacitance. In this case, different gates should have different constants in the delay equation. For example, specify the following:

Dracula Reference

Using Parametric Data (LPEPRO)

```
DEL/GATE:I1 = 20*C  
DEL/GATE:I2 = 10*C  
DEL/GATE:L1 = 10+100*C
```

Revising Schematic Data

Although you can calculate delays without schematic data, you need to document your schematic data in network description language. LPEPRO can put delays calculated by the program equation into the expanded version of the original logic netlist. Use this new file to run more accurate logic simulations or timing analyses. Observe the following restrictions:

- A gate must be a primitive gate of the logic simulator to allow LPEPRO to put delays back to the expanded version of the original schematic data.
- When there are too many levels in your hierarchy, node names might exceed the limit of your logic simulator. For example, most logic simulators can take only 12-character node names.
- For a nonprimitive gate, provide a macro definition in your logic network whenever possible. The delay calculation provided in LPEPRO is not designed to handle complicated network structures.
- If there are unmatched devices or unmatched nodes, the delay information of those nodes is missing or inaccurate.

To run LPE, follow these steps:

1. Issue the SAVE LOGIC command before the CONVERT command when compiling the network using LOGLVS.

The compiled logic network is saved in the *logic.sav* file and read by LPEPRO following the GET SCHEMATIC command. (This does *not* apply to SPICE data.)

2. To run an LVS check that stamps a schematic node name to each layout node, specify SCHEMATIC= *LVSLOGIC* in your rules file.
3. To recover logic network information after LPE is done and layout consistency is accurate, run LPEPRO and issue the GET SCHEMATIC command.

If there is no logic network, or if you have SPICE data, use the GET LAYOUT, GET LAYOUT SCHEMATIC, or GET SPICE command instead.

4. Specify the delay equation with the CALCULATE DELAY command.

Repeat this command for different delay specifications.

5. To generate a netlist with delay information, use STORE .

Dracula Reference

Using Parametric Data (LPEPRO)

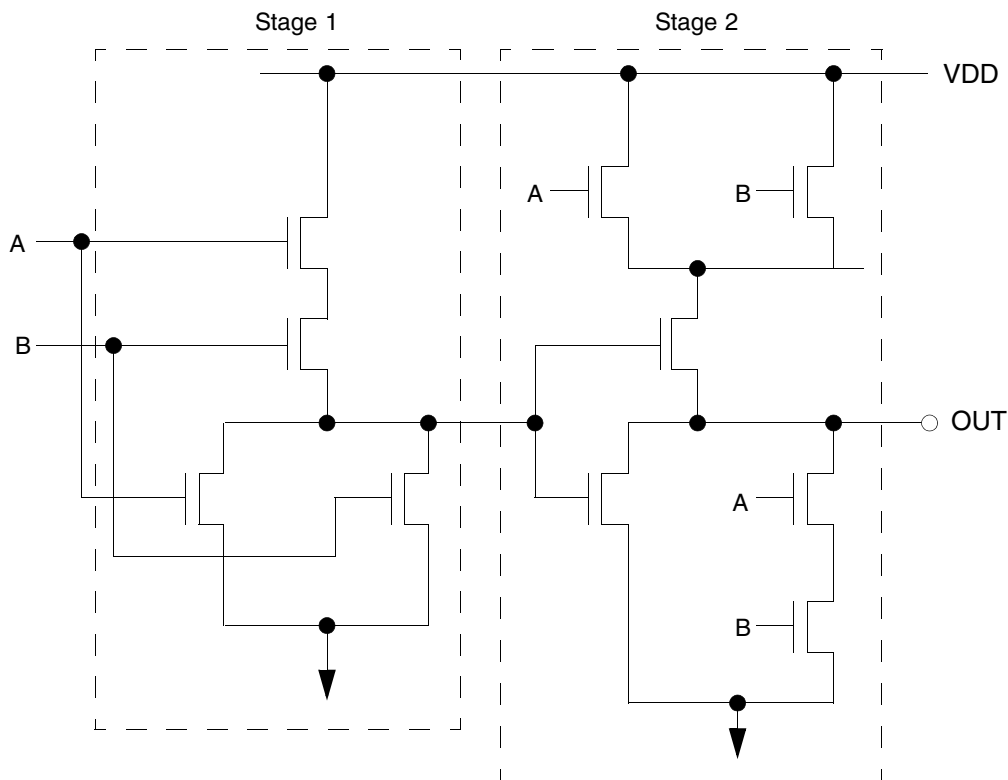
6. To review delay information, use the TYPE DELAY command.

Calculating Delays for Nonprimitive Gates

This section is intended for use with schematic data and nonprimitive gate delays. To use the delay provided by LPEPRO, define the network using primitive logic gates whenever possible.

Multiple Stages Module

As described in the “Logic Gate Generator” section in this chapter, a primitive can have only one stage between input and output node. Under this condition, an XOR gate is not a primitive gate. For example, the following XOR gate has two stages.



The stage 1 rise delay $R1$, and fall delay $F1$, are calculated first. The second stages, $R2$ and $F2$ delays, are added to the first stage delays for total gate delay. For example

$$\begin{aligned}\text{Rise Delay} &= F1 + R2 \\ \text{Fall Delay} &= R1 + F2\end{aligned}$$

When there is more than one stage, the final delays are as follows:

$$\begin{aligned}\text{Rise Delay} &= F1 + R2 + \dots + Fn + Rn + 1 \\ \text{Fall Delay} &= R1 + F2 + R3 + F4 \dots + Rn + Fn + 1\end{aligned}$$

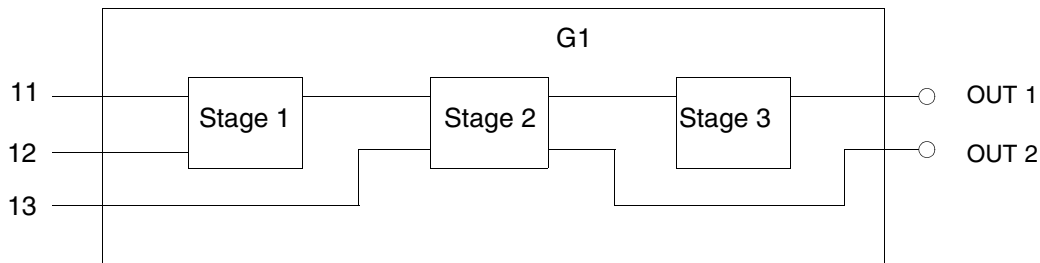
Dracula Reference

Using Parametric Data (LPEPRO)

This rule applies to all multiple stage devices. If there is a transfer gate, the on/off delays are added to both the rise and fall delay.

Multiple Output Devices

Where there is more than one output node, the delays at each output node are different because most logic simulators only let you specify one set of rise and fall delays. LPEPRO selects the worst case delays as gate delays. For example, the delays of gate G1 are the sum of delays in each stage, using the approach in the “Multiple Stages Module” section.



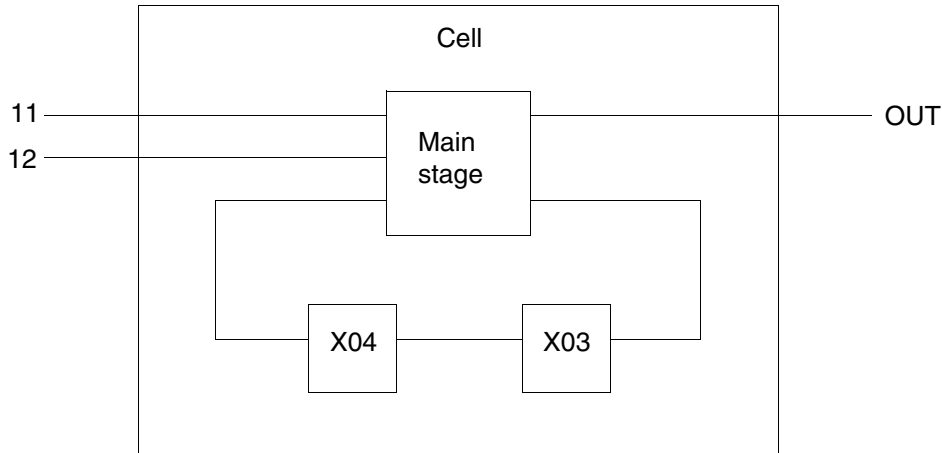
Cells With Internal Feedback Loop or Branch

LPEPRO is not designed to handle complicated cells, so use primitive gates whenever possible. If the cell has an internal feedback loop or branch, delays of elements along the feedback loop or branch are added to the gate delay. For example, the delays of element X03 and X04 are added to the gate delay of this cell, using the formula in 1 above.

Dracula Reference

Using Parametric Data (LPEPRO)

To avoid this, use the */GATE* option to select the cells that have this problem. Because the *W/L* information of this type of element is known, you can calculate delay based on the capacitance of output nodes only, so that the delays are more accurate.



Reviewing Delay Information

Once the delays are calculated, scan them using the same approach you use to scan layout parameters. Use the *TYPE*, *PRINT*, or *STORE* command followed by *DELAY* to display delay information based on range, or the node name you specified. The delay command has the following options:

/RISE

Checks rise delay

/FALL

Checks fall delay

/R/F

Checks the rise delay/fall delay ratio

/F/R

Checks the fall delay/rise delay ratio

The */R/F* option can be used to check pull-up/pulldown ratio.

Dracula Reference

Using Parametric Data (LPEPRO)

Example 1

The following example checks any gate with a delay greater than 50 time units:

```
:TYPE DELAY > 50
```

Example 2

The following example checks pull-up/pulldown ratio and checks the ratio of rise delay to fall delay:

```
:TYPE DELAY/R/F < 1.5
```

In the following example, nodal capacitance is the same for both rise and fall delay. The R/F ratio in this case is the same as the pull-up/pulldown ratio.

```
CAL DELAY = (L/W)*C
```

Example 3

The following example checks delay of gates with output node A10, B10. LPEPRO prints the delay information of these gates only.

```
: TYPE DELAY A10, B10
```

The output listing contains the gate name, gate type, rise delay, fall delay, R/F ratio, and F/R ratio. An example of the output follows.

```
*** DELAY INFORMATION ***
*TYPE DELAY C3.2
C3.2 : CXFR R=76, F=76, R/F=1.00, F/R=1.00
C3.2 : CXFR R=4, F=4, R/F=1.00, F/R=1.00
```

Dracula Reference

Using Parametric Data (LPEPRO)

Extracting Parasitic Resistance (PRE)

This main focus of this chapter is on PRE for Dracula. The following areas are covered:

- [“Overview of Parasitic Resistance Extraction”](#) on page 471
- [“Creating the Resistor and Resistor Terminal Layers”](#) on page 478
- [“Creating the Pad Terminal Layers”](#) on page 479
- [“Calculating Junction Resistance”](#) on page 482
- [“Redistributing Parasitic Capacitances”](#) on page 483
- [“Sample PRE Rules File”](#) on page 485
- [“Considerations for Parasitic Resistance Extraction”](#) on page 490
- [“Sample Data and Results”](#) on page 492
- [“Hierarchical PRE”](#) on page 495

Overview of Parasitic Resistance Extraction

Parasitic Resistance Extraction (PRE) is a Dracula tool that enhances Layout Parameter Extraction (LPE) by extracting parasitic resistance created by any of the conduction layers in the layout. With LPE and PRE, you can extract resistance and capacitance parameters in the interconnect layers of the layout and generate accurate simulations for the most critical designs.

PRE works in both flat and composite modes. For information about running hierarchical PRE in composite mode, refer to the [Hierarchical PRE](#) section in this chapter.

Features of PRE include the following:

- Flexible geometric cutting of resistors, giving you control over the parasitic resistance extraction process
- Automatic generation of resistor body and terminal layers for resistance computation

Dracula Reference

Extracting Parasitic Resistance (PRE)

- Computation of resistor values by square counting (using length and width) and sheet resistance values you provide
- Accurate automatic calculation of junction resistances, including 90-degree corners, T-shaped junctions, and cross-shaped junctions
- Pi (π) and ladder distribution of coupling and fringe capacitances involving parasitic resistor layers
- Resistor shorting and smashing to reduce the R/C network based on thresholds you supply
- Backannotation that extracts names from your schematic netlist and uses them on the extracted SPICE or CDL list
- Output of netlist in SPICE or CDL format

PRE operates in two passes. During the first pass, which is usually performed as part of the Layout Versus Schematic (LVS) check, the circuit is connected normally. During the second pass, the specified conduction layers are geometrically divided into resistor terminals regions and parasitic resistor component regions. During this pass, the circuit is reconnected to include the parasitic components in place. The conduction layer nodes split up into subnodes (the resistor terminal regions) with the parasitic resistors (resistor layer regions) placed across them. Parasitic capacitors are created after this, such that they are distributed across the subnodes. The new circuit, including all parasitic information, is output as a SPICE or CDL file.

You can backannotate the output netlist by deriving node and newly created subnode names from the original schematic netlist used in the LVS check.

The following sections describe in detail the two passes of the extraction procedure.

First PRE Pass

During the first pass, you define and connect the various active components with the CONNECT commands, which is similar to preparing for an LPE or LVS operation. The first pass retrieves the original nodal information for automatic internal layout cross-reference purposes during the second pass. The second pass generates new nodal information because of the introduction of parasitic resistors and new nodes.

During the first pass, PRE does the following:

- Backannotates to the schematic level when an LVS is performed in the first pass
- Verifies there are no LVS errors prior to performing the parasitic resistor extraction

Dracula Reference

Extracting Parasitic Resistance (PRE)

The LVSCCHK command marks the end of the first pass. If you do not need an initial LVS run, follow the CONNECT statements with the commands for the second pass.

If you want a layout-only PRE run, you do not need to include the LVSCCHK command. The resulting SPICE or CDL netlist is based on layout information only.

Second PRE Pass

The second pass uses the nodal information from the first pass to reconnect the circuit. This pass can also use the first pass LVS schematic netlist for backannotation.

In the second pass, PRE does the following:

1. Creates the resistor and resistor terminal layers from the specified resistive conduction layer.
2. Reconnects the circuit using the created terminal layer.
3. Redefines elements.
4. Creates the parasitic resistor and parasitic capacitor devices.
5. Generates the SPICE or CDL output file.

The following sections describe the commands you use for these steps.

Step 1: Creating the Resistor Terminal and Resistor Layers

Split the trapezoids of the conduction terminal regions and create two new layers by using the CUT-TERM command. The CUT-TERM command creates a resistor body layer and a resistor terminal layer from a conduction layer and a contact layer. You can also use a series of geometric operation commands (such as SELECT and SIZE), spacing and logical operations, and the CUT command to achieve similar results. The splits are done in such a way as to isolate the regions where it makes the most sense to create parasitic resistor devices. The regions corresponding to the resistor terminal layer are treated as conduction geometries, whereas the regions corresponding to the resistor layer are treated as parasitic resistor device recognition areas connected at two terminal regions.

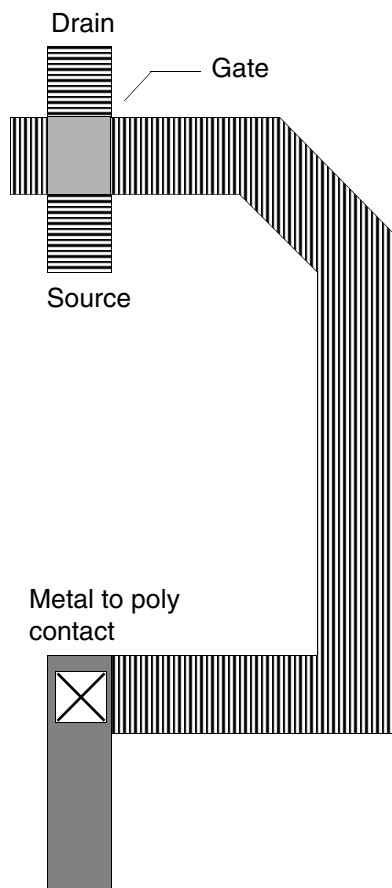
Resistor terminal regions are created over active devices (for example, gate areas of MOS transistors), contacts, corners of conduction lines, nonparallel geometries, wide conduction lines, and places where the width of the conduction line changes.

For more details about splitting the resistance and resistance terminal layers, refer to the section [“Creating the Resistor and Resistor Terminal Layers”](#) on page 478.

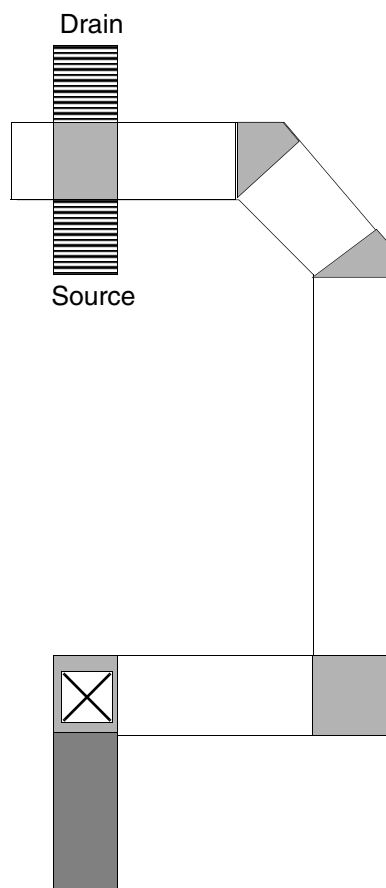
Dracula Reference

Extracting Parasitic Resistance (PRE)

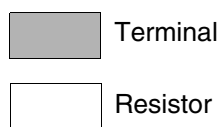
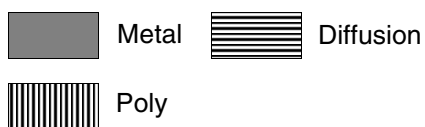
The following example shows how the conduction polysilicon (poly) layer is split into resistor and resistor terminal regions by the various commands.



Layout before splitting poly layer into resistance



Layout after splitting poly layer into resistance and resistance terminal layer



Step 2: Reconnecting the Circuit

After the conduction layer is split, the circuit must be completely reconnected with the resistance terminal layer taking the place of the conduction layer.

Dracula Reference

Extracting Parasitic Resistance (PRE)

Because contacts found in the conduction layer are still found in the resistor terminal regions, the connectivity remains the same with one exception: the parasitic resistors that approximate the resistance of the conduction layer are now in place between the terminal regions.

To reconnect the circuit, use the RCONNECT-LAYER command in the Input-Layer block. The sequence of layers corresponds to the same layers used in the CONNECT-LAYER command, substituting the resistor terminal layers for the corresponding conduction layers. RCONNECT commands in the second pass correspond to the CONNECT commands in the first connection pass, except for the substitution of the resistor terminal layers for the corresponding conduction layers.

All layers on the RCONNECT-LAYER sequence must have nodal information prior to executing the RCONNECT commands. The connection process during the first pass provides some of the information. Use the STAMP command to copy nodal information from the conduction layers that were split into the resistor terminal layers. After reconnection, all layers on the RCONNECT-LAYER sequence get new nodal information, reflecting the subnodes created (the split resistor terminals).

New subnode numbers or names are generated from the original node numbers or from the original node text if you do not request backannotation. The new subnode number or name is generated by concatenating the original number or name with a delimiter and a new number. The second number is chosen to differentiate the new node numbers and names. The sample SPICE output files in this chapter show examples of new node names and numbers.

Step 3:

Creating Parasitic Resistors and Capacitors from the Resistor Layers

You define parasitic resistor devices for a given resistor layer by using the PARASITIC RES command. This command uses the fully created resistor and resistor terminal layers as parameters. You must use an ATTRIBUTE RES command associated with the PARASITIC RES to provide the sheet resistivity. The resistances can then be derived from the geometries of the resistor layers, the sheet resistance value provided, and the resistor terminal pairs from the geometries of the resistance terminal layer.

If you want parasitic capacitors, use the PARASITIC CAP and ATTRIBUTE CAP commands after creating the parasitic resistor devices.

When creating the parasitic capacitors, consider capacitances to both the resistor and the resistor terminal layers, now taking the place of the original conduction layer. Parasitic capacitances associated with the resistance layers are automatically redistributed to the resistor terminals. For more information about parasitic capacitances, refer to the [Redistributing Parasitic Capacitances](#) section in this chapter. If you specify the CUT-TERM

Dracula Reference

Extracting Parasitic Resistance (PRE)

command, Dracula automatically computes junctions of the resistance terminals. Refer to the [Calculating Junction Resistance](#) section in this chapter.

Step 4: Redefining Elements

After replacing a conduction layer with a resistor and resistor terminal layer, you must redefine elements defined with this conduction layer during the first pass, using the new resistor terminal for the second pass. For example, if the poly layer is split into a resistance layer PRES and a terminal layer PTRM, define MOS devices again during the second pass using the PTRM layer in place of the poly layer. The following commands might be used for such a purpose:

```
; Pass one MOS element definition:
. . .
ELEMENT MOS[N] NGATE POLY NSD PWELL
ELEMENT MOS[P] PGATE POLY PSD NSUB
. . .

; Pass two MOS element definition:
. . .
ELEMENT MOS[N] NGATE PTRM NSD PWELL
ELEMENT MOS[P] PGATE PTRM PSD NSUB
. . .
```

Step 5: Generating the SPICE or CDL Output File

Use the LPESELECT command to create the SPICE or CDL netlist file.

To short small resistors created by the cutting of parasitic resistance, specify a threshold with the LPESELECT RES command. The terminals of resistors having less or equal resistance than the threshold are shorted. You can use the ATTRIBUTE RES command to merge resistors that are too small.

Obtain new node names by concatenating a delimiter and a subnode number to the original node name or number. Use the SUBNODE-DELIMITER command to override the default delimiter, a colon (:) that is used in the concatenation. Request backannotation with the S option. With backannotation, node names from the original netlist are used instead of node numbers to create the new node names.

Creating the Resistor and Resistor Terminal Layers

The two basic entities in layout circuit extractions are geometries representing devices and geometries representing conductive terminals or nodes. In cutting parasitic resistors, polygons (or the resistive conduction layers) must be partitioned into the following:

- Rectangular areas (as much as possible) corresponding to resistor devices
- Rectangles or polygons corresponding to resistor terminal regions. Resistor terminal regions act as conduction layer nodes that connect together the two-terminal parasitic resistors and other devices.

You can cut the resistor and resistor terminal layers in two ways: use the CUT-TERM command or use a series of commands. Using the CUT-TERM command is the preferred method for new rules files because it requires fewer commands and produces more accurate results.

Using the CUT-TERM Command

The CUT-TERM command cuts parasitic resistors as follows:

- Cuts contact regions from the resistor layer.
- Cuts junction areas from the resistor layer. It can cut 90-degree corner junctions, T-junctions, and cross-shaped junctions.
- Cuts long geometry strips into smaller segments. A long strip geometry is more accurately represented as a series of R/C lumped circuits.

To learn how to create resistor and resistor terminal layers or for a list of the commands CUT-TERM replaces, refer to the [CUT-TERM](#) section in Chapter 13 of this manual.

Using a Command Sequence

You can use combinations of Dracula spacing and logical commands plus SIZE, SELECT and CUT commands to cut the layers, then save the result as the resistor layer and the resistor terminal layer. The conduction layer polygons can be split in a variety of ways. In general, the splitting process should follow these rules:

- The connection between different layers can only be done through contacts at conduction nodes.

Form resistor terminal regions around contacts involving the specified conduction layer.
- Resistors are two-terminal devices.

Form resistor terminals at all branching corners (corners of T-shaped polygons, Y-shaped polygons, and so forth).
- All resistors must be rectangles.

Cut non-rectangular resistors into rectangles to get better accuracy in the parasitic resistance calculations.
- Terminals for active devices must be connected to nodes.

Include the MOS gate areas made out of a specified resistive conduction layer, such as poly, as resistor terminals.

Creating the Pad Terminal Layers

The basic parts of the layout circuit extractions are geometries representing devices and conductive terminals or nodes. You must do the following when using the PAD-LAYER command:

- Define the pad terminals in the Input-Layer block following the RCONNECT-LAYER command.
- Specify one PAD-LAYER command in the rules file for PRE or HPRE if it applies all cut-term layers. Otherwise, list each pad layer for its effected layers respectively. If you do not specify the affected cut-term layers, PAD-LAYER affects all cut-term layers.
- Input the pad layer from the input database or a derived layer.

PRE or HPRE performs the LOGICAL AND operation on the CUT-TERM input-conduction layer and on the pad layer defined by the PAD-LAYER command in order to form extra terminals automatically.

Dracula Reference

Extracting Parasitic Resistance (PRE)

You can specify multiple PAD-LAYER commands with a set of affected cut-term layers specified for each pad-layer in the same rules file.

Use one of the following methods to create pad layers:

- Generate a ring geometry (that is, on the boundary) as the pad layer (HPRE). This method does not work for hierarchical runs.

```
PAD-LAYER = PPAD
...
SIZE BULK BY -0.1 TBULK
NOT BULK TBULK PPAD
```

- Use the passivation layer as the PAD-LAYER.
- Use the LABEL command. For example, attach text to the METAL layer located near the boundary.

```
PAD-LAYER = PPAD
...
SELECT METAL LABEL[R] ? PPAD
```

- Use the output of the PIN-TEXT-LAYER command as the pad layer.

```
PIN-TEXT-LAYER = pinlay 5
PAD-LAYER = pinlay
```


Examples

The following example shows the location and syntax of the PAD-LAYER command in the Input-Layer block of the rules file.

```
*INPUT-LAYER
;
PWELL = 1                ;PWELL layer
ACTIVE = 2               ;All active diffusion
POLY = 3                 ;Polysilicon
PDIFF = 4                ;P+ implant
CONT = 5                 ;All metal contacts
MET = 6      TEXT=6
SUBSTRATE = BULK 63      ;BULK used for reverse field
;
CONNECT-LAY = NSUB PWELL PSD NSD POLY MET
;
RCONNECT-LAY = NSUB PWELL PSD NSD PTRM MET
;
PAD-LAYER = VAPOX
*END
```

The following example uses multiple PAD-LAYER commands. You can specify a set of affected cut-term layers for each PAD-LAYER command.

```
.
.
CONNECT-LAY = NSUB PWELL PSD NSD POLY MET
RCONNECT-LAY = NSUB PWELL PSD NSD PTRM MTRM
.
.
PAD-LAYER = PAD, POLY
PAD-LAYER = VAPOZ, MET
.
.
CUT-TERM MET CONT MRES MTRM
CUT-TERM POLY CONT PRES PTRM
```

Error Conditions

The following error conditions can occur when you use the PAD-LAYER command.

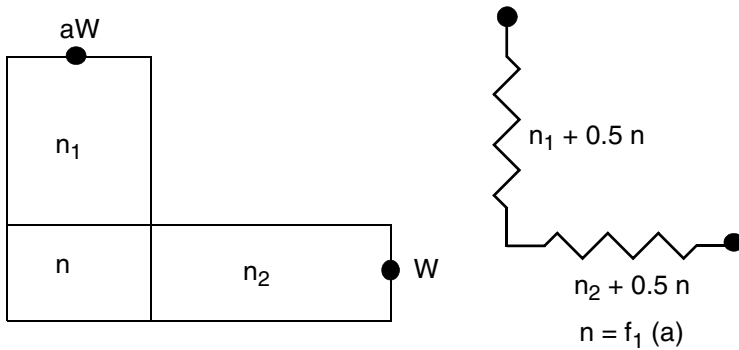
- If you do not specify a pad layer in the rules file for a PRE or HPRE job, PDRACULA reports an error message in the *.log* file.
- If you do not select a pad terminal layer, the UPDPAD module reports a warning message in the *.log* file and randomly picks a subnode terminal for the pad.
- If you select multiple pad terminals for the same node but do not specify the MULTI-PIN-FILE command, PRE shorts the terminals together.

Calculating Junction Resistance

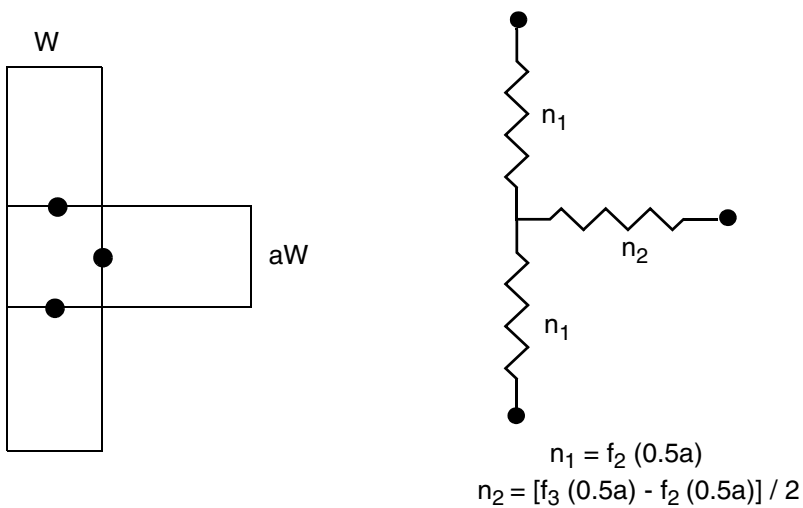
Resistance layer geometries must have two terminals. Single-terminal resistance regions are discarded. Rectangular resistance layer geometries, which provide the most accurate resistance calculation, are preferred.

The CUT-TERM command cuts junction regions from the original resistance layer and treats them as terminals. To improve the accuracy of the resistance extraction results, the resistances associated with junction regions are calculated by the following models for corner junctions, T-shaped junctions, and cross-shaped junctions:

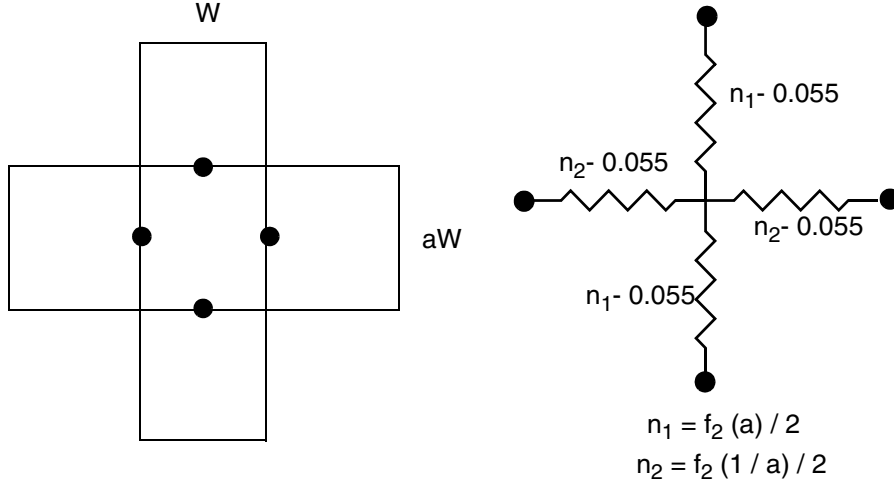
Corner Junction Resistances



T-Shaped Junction Resistances



Cross-Shaped Junction Resistances



The following equations express the functions f_1 , f_2 , and f_3 .

$$f_1(a) = \frac{1}{a} + \frac{a^2 - 1}{\pi a} \arccos \frac{a^2 - 1}{a^2 + 1} - \frac{2}{\pi} \ln \frac{4a}{a^2 + 1} \quad a \geq 1$$

$$f_2(a) = a - \frac{2a}{\pi} \arctan a + \frac{1}{\pi} \ln(a^2 + 1) \quad a > 0$$

$$f_3(a) = \begin{cases} f_1(a) & a \geq 1 \\ f_1\left(\frac{1}{a}\right) & 0 < a < 1 \end{cases}$$

Redistributing Parasitic Capacitances

Define parasitic capacitances only after the introduction of the parasitic resistors so that you take into account the new subnodes.

The parasitic capacitances can be categorized into the following types.

- Node to node
- Resistor to node

Dracula Reference

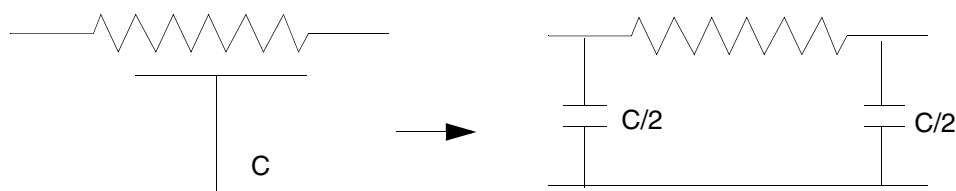
Extracting Parasitic Resistance (PRE)

■ Resistor to resistor

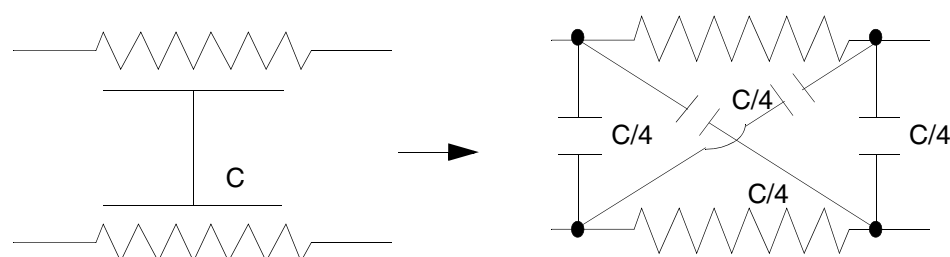
A node corresponds to a connected set of conduction layer geometries, or resistor terminals, or both.

Treat internodal capacitances the same as in LPE. Lump and divide distributed capacitances computed along regions forming resistors as shown in the following two illustrations.

PI Network Arrangement for Resistance-to-node Capacitance



Ladder Network for Resistor-to-resistor Capacitance



Because creating the parasitic resistor elements involves splitting a conduction layer into the resistor elements and the resistor terminals, you need to consider additional parasitic capacitances with the creation of parasitic resistors. The following is an example of a single-metal CMOS process. The layers from the top surface of the die to the bottom are

- Metal
- Poly
- NSD (N - diffusion)
- PSD (P - diffusion)
- P-Well,
- N-Sub (Substrate)

The capacitances for these layers (without creating parasitic resistors) might be the following:

- Metal to NSD
- Metal to PSD
- Metal to P-Well

Dracula Reference

Extracting Parasitic Resistance (PRE)

Metal to Nsub
Poly to Metal
Poly to P-Well
Poly to N-Sub

If the poly layer is used to extract parasitic resistors, then it is split into the resistor layer *PRES* and the resistor terminal layer *Ptrm*. When you write the rules file to generate parasitic capacitances, these two new layers have to be considered instead of the poly layer. The new capacitances to be calculated are the following:

Metal to NSD
Metal to PSD
Metal to P-Well
Metal to Nsub

PRES to Metal
Ptrm to Metal

PRES to P-Well
Ptrm to P-Well

PRES to N-Sub
Ptrm to N-Sub

PRE automatically splits the total resistor-to-node capacitances (PRES to Metal, PRES to P-Well, PRES to N-Sub) into two halves and connects them to the two endpoints of the resistor terminals in the form of a PI network.

Although fringe resistor-to-resistor capacitances are not considered in the previous example, the same consideration of the split layers applies.

Sample PRE Rules File

The following rules file shows a typical command sequence for an LVS check followed by PRE operations that include parasitic resistance extraction. Many of the commands in this file correspond to the commands needed for a normal LVS check.

```
;*****
;*****  SI-GATE CMOS INPUT FILE FOR LPE  *****
;*****  PARASITIC RESISTANCE EXTRACTION  *****
;*****                                *****
;                                sample.com
;
*DESCRIPTION
PRIMARY = STRAIGHT_GEO      ;Top structure name
SYSTEM = GDS2               ;System that data came from
INDISK = STREAM.OUT         ;File on disk where data is stored
OUTDISK = SAMPLE            ;File where output data is stored
SCALE = .001 MICRON        ;1 database unit is equal to .001 micron
```

Dracula Reference

Extracting Parasitic Resistance (PRE)

```

RESOLUTION = .25 MICRON      ;Smallest increment used in DRACULA
MODE = EXEC NO               ;Job is not be submitted automatically
BATCH-QUEUE = FAST           ;Job is entering the FAST queue
PRINTFILE = SAMPLE           ;Specify prefix for output files
PROGRAM-DIR = usr1/dracula/bin
                               ;Path for DRACULA to find executables
;
DIODESEQ = A1 P1              ;Spice parameter sequence
SCHEMATIC = LVSLOGIC          ;Compiled schematic name to be used

;-----
SUBNODE-DEL = ":"             ; Subnode delimiter command
;-----
MODEL = MOS[N],N MOS[P],P DIODE[N],N DIODE[P],P
UNIT = CAPACITANCE,PF AREA,U PERIMETER,M RESISTANCE,K
*END
*INPUT-LAYER
;
PWELL = 1                     ;PWELL layer
ACTIVE = 2                     ;All active diffusion
POLY = 3                       ;Polysilicon
PDIF = 4                       ;P+ implant
CONT = 5                       ;All metal contacts
MET = 6 TEXT=6
SUBSTRATE = BULK 63           ;BULK used for reverse field
;
CONNECT-LAY = NSUB PWELL PSD NSD POLY MET
;
;
;-----
RCONNECT-LAY = NSUB PWELL PSD NSD PTRM MET
;This command establishes the layers that are involved
;when the layout circuit is reconnected by the RCONNECT
;commands given when the parasitic resistance is extracted.
;The layers shown here are the same as the CONNECT-LAY command
;except that the POLY layer has been substituted by the PTRM
;layer, corresponding to the resistor terminal layer.
;
PAD-LAYER = VAPOX
;-----
*END
*OPERATION
;
;*****
;***** DO REGULAR LVS CHECK *****
;*****
;
AND ACTIVE PDIF PDIF          ;All P+ active regions
NOT ACTIVE PDIF NDIF          ;All N+ active regions
AND POLY NDIF NGATE           ;Create all N-channel gates
AND POLY PDIF PGATE           ;Create all P-channel gates
NOT NDIF NGATE NSD            ;Create all N-channel source/drains
NOT PDIF PGATE PSD            ;Create all P-channel source/drains
NOT BULK PWELL NSUB           ;Create N-substrate area
AND PSD PWELL PTAP            ;Create PWELL taps for vss
AND NSD NSUB NTAP             ;Create NSUB taps for vdd
;
CONNECT MET POLY BY CONT       ;Connect the various conduction
CONNECT MET PSD BY CONT        ;layers through the proper

```

Dracula Reference

Extracting Parasitic Resistance (PRE)

```
CONNECT MET NSD BY CONT      ;contacts.
CONNECT PSD PWELL BY PTAP    ;Connect substrate and P well
CONNECT NSD NSUB BY NTAP     ;region through proper taps.
;
ELEMENT MOS[N] NGATE POLY NSD PWELL ;Define N-Channel MOS
;                               ;transistors.
ELEMENT MOS[P] PGATE POLY PSD NSUB  ;Define P-Channel MOS
;                               ;transistors.
;
LVSCHK[C]
;
;
;
;
;
;
;
*****
;Here begins the second pass - which performs the *
;parasitic resistor extraction from the Poly layer*
*****
;
;*****
;*****   CUT RESISTORS ON POLY LAYER   *****
;*****
;
;Cutting Resistor Terminal Areas around Contacts
;
OR   NGATE PGATE GATE      ;GATE gets all gate layer geometries.
CUT-TERM POLY CONT PRES PTRM GATE
;                               ;GATE is T_DEV
;
;PRES now has the geometries corresponding to the Poly parasitic
;resistors and PTRM has the geometries corresponding to the Poly
;resistor terminals.
;*****
;***** RE-CONNECT WITH PARASITIC RESISTANCE CUT *****
;*****
;
STAMP PTRM BY POLY      ;Get original nodal information
;                       ;from the Poly layer stamped
;                       ;into the resistor terminals.
;
;-----
;Establish new connects i.e., reconnect the circuit using
;the Poly resistor terminal layer, PTRM, instead of the
;Poly conduction layer.
RCONNECT MET PTRM BY CONT
RCONNECT MET PSD BY CONT
RCONNECT MET NSD BY CONT
RCONNECT PSD PWELL BY PTAP
RCONNECT NSD NSUB BY NTAP
;-----
;With the Parasitic Resistors created and inserted into
;the circuit, the other parasitic devices are handled.
;
;
;
;*****
;*****   CREATE PARASITIC DIODES   *****
;*****
```

Dracula Reference

Extracting Parasitic Resistance (PRE)

```
;
AND NSD PWELL NDIO ;N+ Diode minus taps
AND PSD NSUB PDIO ;P+ Diode minus taps
;
;
;
;
;*****
;***** CREATE PARASITIC CAPACITANCES *****
;*****
;!!!!!!! Note the splitting of parasitic capacitances !!!!!!!

AND MET PTRM MPOLY ;Metal to Poly Terminal Cap
AND MET PRES MPOLYR ;Metal to Poly Resistor Cap
;
NOT MET POLY METNP ;Create Metal with no Poly layer
AND METNP NSD MNSD ;Metal to N+ Diffusion Cap
AND METNP PSD MPSD ;Metal to P+ Diffusion Cap
;
NOT MET MPOLY M0 ;Prepare for PWELL and NSUB Caps
NOT M0 MPOLYR M1 ;Remove all other Cap values
NOT M1 MNSD M2
NOT M2 MPSD M3
AND M3 PWELL MWELL ;Metal to PWELL cap
AND M3 NSUB MSUB ;Metal to NSUB cap
;
NOT PTRM GATE PTRM1 ;Remove gate from interconnect
AND PTRM1 PWELL POWELL ;POLY interconnect to PWELL cap 1
AND PRES PWELL POWELR ;POLY interconnect to PWELL cap 2
AND PTRM1 NSUB POSUB ;POLY interconnect to NSUB cap 1
AND PRES NSUB POSUBR ;POLY interconnect to NSUB cap 2
;
;*****
;**** DECLARE ELEMENT AND PARASITIC DEVICES ****
;*****
;
ELEMENT MOS[N] NGATE PTRM NSD PWELL ;Redefine elements with poly
ELEMENT MOS[P] PGATE PTRM PSD NSUB ;terminal layer PTRM instead
; ;of POLY layer.
PARASITIC DIO[N] NDIO PWELL NDIO ;N+ Diode to PWELL
PARASITIC DIO[P] PDIO PDIO NSUB ;P+ Diode to SUBSTRATE
;
PARASITIC RES[P] PRES PTRM ;POLY interconnect resistor
ATTRIBUTE RES[P] 0.01 ;Kohms per square
; Note that the above command indicates that the PRES layer
; corresponds to the resistance layer. This tells Dracula
; that any capacitances to these layers should be split
; according to the Pi or ladder network form.
;
;#### Capacitance attributes are in pf per unit area, as
;#### indicated in the UNIT command
;
;!!!! Parasitic Capacitors involving parasitic resistance !!!!!
;!!!! layer and parasitic terminal resistor layer. !!!!!
;!!!! !!!!!
;!!!! Note that capacitances involving parasitic !!!!!
;!!!! poly resistor devices (PRES) are automatically !!!!!
;!!!! split in half and placed across the two ends of the !!!!!
;!!!! resistors. !!!!!
```


Dracula Reference

Extracting Parasitic Resistance (PRE)

```

;
PARASITIC CAP[A1] MPOLY MET PTRM ;Capacitor from METAL layer
ATTRIBUTE CAP[A1] 0.000025 ;to POLY resistor terminal.
PARASITIC CAP[A2] MPOLYR MET PRES ;METAL to POLY resistor
ATTRIBUTE CAP[A2] 0.000025 ;capacitor.
;
PARASITIC CAP[B] MNSD MET NSD ;METAL to N+ capacitor
ATTRIBUTE CAP[B] 0.000025
PARASITIC CAP[C] MPSD MET PSD ;METAL to P+ capacitor
ATTRIBUTE CAP[C] 0.000025
;
PARASITIC CAP[D] MWELL MET PWELL ;METAL to PWELL capacitor
ATTRIBUTE CAP[D] 0.000025
PARASITIC CAP[E] MSUB MET NSUB ;METAL to SUBSTRATE
ATTRIBUTE CAP[E] 0.000025 ;capacitor.
;
PARASITIC CAP[F1] POWELL PTRM1 PWELL;POLY to PWELL capacitor 1
ATTRIBUTE CAP[F1] 0.000075
PARASITIC CAP[F2] POWELR PRES PWELL ;POLY to PWELL capacitor 2
ATTRIBUTE CAP[F2] 0.000075
;
PARASITIC CAP[G1] POSUB PTRM1 NSUB ;POLY to SUBSTRATE cap 1
ATTRIBUTE CAP[G1] 0.000075
PARASITIC CAP[G2] POSUBR PRES NSUB ;POLY to SUBSTRATE cap 2
ATTRIBUTE CAP[G2] 0.00007
;
;
;*****
;***** DEFINE FRINGE CAPACITANCE *****
;*****
PARASITIC CAP[MF] MET MET MET ; MET fringe capacitance
ATTRIBUTE CAP[MF] 5.0 0.00001
;
;!!! Note cut-term conduction layer's fringe capacitance !!!!
;!!! use one FRINGE[R] command to define it !!!
;
FRINGE[R] CAP[PF] PTRM PRE ; POLY fringe capacitance
ATTRIBUTE CAP[PF] 5.0 0.00002
;
;*****
;***** OUTPUT SPICE FILE *****
;*****
;!!!! Note that the [S] option below indicates !!!!
;!!!! back-annotation to the output netlist. !!!!
;
LPESELECT[S] MOS & ;Output MOS transistors
LPESELECT[S] DIO & ;Output diodes
LPESELECT[S] CAP GT 0.0 & ;Output capacitors
;-----
LPESELECT[S] RES GT 0.001 OUTPUT SPICE ;Output resistances
;above .001 Kohms were
;indicated in the units
;command.
;-----
*END

```

Considerations for Parasitic Resistance Extraction

Parasitic resistance extraction (PRE) depends on geometry and method of formation. In defining parasitic resistances, consider the following:

- Because wider poly provides a reduced resistance contribution for a given length, we recommend a value of three to five times the minimum poly width as the upper limit for the MAXWIDTH parameter in the CUT-TERM command. Take into consideration the layout aspects of the resistive layer because the resistance contribution is dependent on sheet resistance and length, as well as the width of the geometries.
- Because parasitic resistance extraction is dependent on the shape of the geometries and the method of extraction, some resistances yield only approximations, rather than accurate values.

The following examples illustrate how the interface between the resistor device geometries and the resistor terminal geometries affects the resistance computation.

The calculation of a trapezoidal resistor is determined by the widths W1 and W2 of the endpoints of the resistor.

Resistance is calculated as

$$R = r * L/W$$

where

$$W = (W1 + W2) / 2$$
$$L = \text{Area} / W$$

When the resistor terminals cleanly touch the resistor element endpoints, as in the following figure, W1 and W2 both are equal to W; that is, the width of the resistor layer and the area are easily determined.



Simple interface between resistor element and terminal

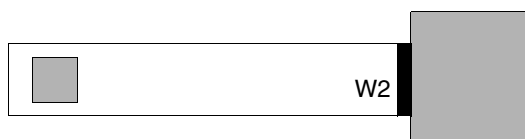
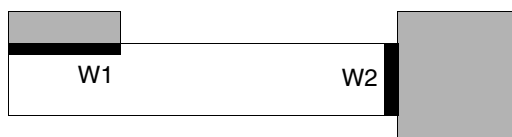
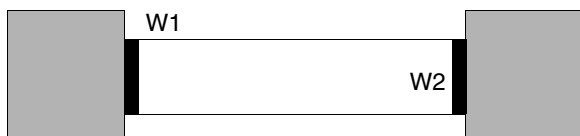
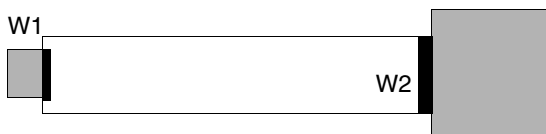
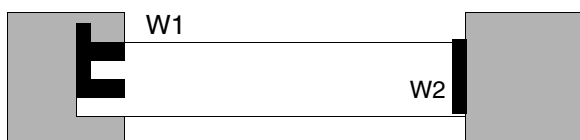


Dracula Reference

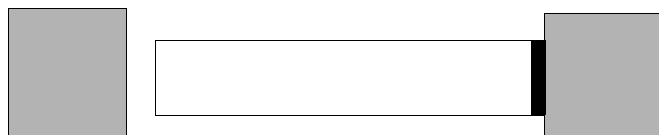
Extracting Parasitic Resistance (PRE)

The following examples indicate the computed values for $W1$ and $W2$ when the resistor terminal geometries abut the resistor elements in special ways. For these cases, the width on a given side is determined by the edge of the resistor geometry along its intersection with the resistor terminal. If the resistor geometry fully encloses the resistor terminal, the width for that terminal is 0. The full area of the resistor regions computes L .

The widths $W1$ and $W2$ are shown with the thicker lines.



$W1=0$ Terminal does not intersect resistor edge

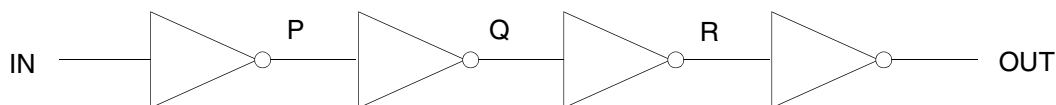


Single terminal resistor is not recognized

 Resistor terminal  Resistor device

Sample Data and Results

The following schematic and netlist represent an example of a four-inverter circuit given as input.



Example Input SPICE/CDL File

```
*.GLOBAL VDD,VSS
*
.SUBCKT INV1 Z / A
* INVERTER
MP1 Z A VDD VDD P WP=75 LP=3
MN1 Z A VSS VSS N WN=25 LN=3
.ENDS INV1
*
.SUBCKT INV2 Z / A
* INVERTER
MP1 Z A VDD VDD P WP=120 LP=3
MN1 Z A VSS VSS N WN=40 LN=3
.ENDS INV2
*
*
.SUBCKT SAMPLE OUT / IN
X1 P IN /INV1
X2 Q P /INV2
X3 R Q /INV1
X4 OUT R /INV2
.ENDS SAMPLE
```

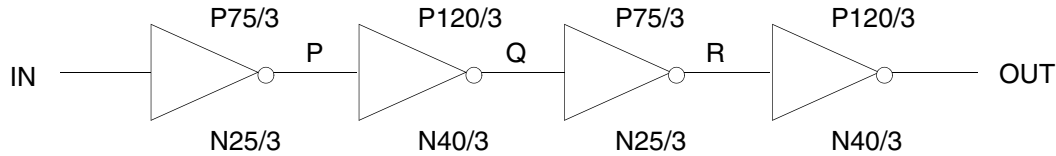
After parasitic resistance extraction, the circuit is shown by the figure and the output SPICE file that follows. Single nodes often translate into many parasitic resistors and capacitors because the number of terminals and parasitic resistors generated along a path relates to the number of bends, contacts, gates, multiway branches, and other such aspects of the layout. The output SPICE file was generated with backannotation enabled so that it reflects the signal names specified in the SPICE input file.

Dracula Reference

Extracting Parasitic Resistance (PRE)

Schematic of Output SPICE File Including Parasitic Resistors

Circuit before extraction



Example Output SPICE File

```
*
* ECAD/LPE SPICE FILE : SPICE
* DATE : 28-AUG-85
*
*****
***** MOS XTOR      PARAMETERS FROM : 7MOSXREF
*****
*
*.GLOBAL  VDD  VSS
*
.SUBCKT SPICE  OUT  IN
*
*****
*****CORNER ADJUSTMENT FACTOR =  0.0000000
*****
*
MP1X1      P IN:3  VDD VDD P   L=3.00M   W=75.00M
MN1X1      P IN:2  VSS VSS N   L=3.00M   W=25.00M
MP1X2      Q P:10  VDD VDD P   L=3.00M   W=120.00M
MN1X2      Q P:9   VSS VSS N   L=3.00M   W=40.00M
MP1X3      R:3 Q:2  VDD VDD P   L=3.00M   W=75.00M
MN1X3      R:3 Q:1  VSS VSS N   L=3.00M   W=25.00M
MP1X4      OUT R:9  VDD VDD P   L=3.00M   W=120.00M
MN1X4      OUT R:8  VSS VSS N   L=3.00M   W=40.00M
*
*----- TOTAL # OF MOS TRANSISTORS FOUND : 8
*----- COMMENTED : 0
*
*****
***** DIODE      PARAMETERS FROM:7DIOXMER
*****
*
D18  P VDD P      364.50U  195.00M
D10  VSS P N      164.50U   95.00M
D20  OUT VDD P    544.50U  285.00M
D12  VSS OUT N    224.50U  125.00M
D22  Q VDD P      544.50U  285.00M
D14  VSS Q N      224.50U  125.00M
D24  R:3 VDD P    364.50U  195.00M
D16  VSS R:3 N    164.50U   95.00M
*
```

Dracula Reference

Extracting Parasitic Resistance (PRE)

```
*----- TOTAL # OF DIODES FOUND :8
*----- COMMENTED :0

*****
***** CAPACITORS PARAMETERS FROM:7CAPXMER
*****
*
C1 P VSS 1.28000E-02PF
C2 P:1 VSS 5.71750E-02PF
C3 P:2 VSS 5.38625E-02PF
C4 P:3 VSS 7.87875E-02PF
C5 R VSS 5.71750E-02PF
C6 R:1 VSS 5.38625E-02PF
C7 VDD VSS 5.64625E-01PF
C8 OUT VSS 3.22750E-02PF
C9 R:2 VSS 3.97625E-02PF
C10 P:4 VSS 3.50750E-02PF
C11 Q VSS 9.36750E-02PF
C12 R:3 VSS 1.28000E-02PF
C13 IN VSS 6.75000E-04PF
C14 N:1 VSS 6.75000E-04PF
C15 R:4 VSS 6.75000E-04PF
C16 R:5 VSS 9.75000E-04PF
C17 R:6 VSS 6.75000E-04PF
C18 R:7 VSS 6.75000E-04PF
C19 R:8 VSS 3.00000E-04PF
C20 R:9 VSS 3.00000E-04PF
C21 R:10 VSS 9.75000E-04PF
C22 P:5 VSS 8.25000E-04PF
C23 P:6 VSS 6.75000E-04PF
C24 P:7 VSS 8.25000E-04PF
C25 R:11 VSS 1.05000E-03PF
C26 R:12 VSS 6.75000E-04PF
C27 R:13 VSS 1.05000E-03PF
C28 P:8 VSS 6.75000E-04PF
C29 P:9 VSS 3.00000E-04PF
C30 P:10 VSS 3.00000E-04PF
C31 R:14 VSS 6.75000E-04PF
*
*----- TOTAL # OF CAPS FOUND : 31
*----- COMMENTED : 0

*****
*****RESISTORS PARAMETERS FROM:7RESPREF
*****

R27 IN IN:1 1.70333E00K
R30 IN:1 IN:2 1.29167E-01K

R31 IN:2 IN:3 1.80833E-01K
R32 P P:1 3.11000E00K
R33 P:1 P:2 3.38333E00K
R35 P:2 P:3 3.03333E00K
R37 R R:4 2.61667E00K
R38 R R:1 3.38333E00K
R40 R:1 R:5 3.13167E00K
R41 R:4 R:7 1.72000E00K
R42 R:5 R:10 2.34750E00K
R44 R:6 R:7 1.27333E00K
R46 R:6 R:8 1.51667E-01K
R47 R:8 R:9 9.00000E-02K
```

Dracula Reference

Extracting Parasitic Resistance (PRE)

```
R49  R:2    R:10  7.50000E-02K
R51  P:3    P:6   1.20000E-01K
R53  P:5    P:6   9.00000E-02K
R54  P:5    P:7   3.47000E00K
R55  P:4    P:7   8.33333E-02K
R58  R:2    R:11  7.83333E-02K
R59  R:11   R:13  1.17250E00K
R60  R:12   R:13  1.11500E00K
R61  R:12   R:14  5.10000E-01K
R63  P:4    P:8   5.03333E-01K
R64  P:8    P:9   1.51667E-01K
R66  P:9    P:10  9.00000E-02K
R67  Q:1    Q:2   1.80833E-01K
R68  R:3    R:14  4.13333E-01K
R69  Q      Q:2   2.06667E-01K
*
*----- TOTAL # OF RESISTORS FOUND :    29
*----- COMMENTED :                0
*
*.ENDS
```

Hierarchical PRE

Composite mode PRE extracts the designed and parasitic devices from the composite plane. Like flat mode PRE, HPRE generates a SPICE file with back annotation to the schematic if you are also running LVS as part of the job. HPRE generates a SPICE file with layout information only if you are not running LVS as part of the job.

The difference between PRE and HPRE is that when you run HPRE in composite mode, the HPRE program includes interconnection between instances or placements. Because of this, the format HPRE generates is not the same as flat SPICE.

Note: There is no cell mode PRE.

For more information on running HPRE, see “[Running Hierarchical Parameter Extractions](#)” in the “Setting up Hierarchical Dracula” chapter of the *Dracula User Guide*.

SPF File Overview

The Standard Parasitic File (SPF) lists extracted electrical data for a design created by Place and Route tools such as Block Ensemble, Cell Ensemble, and Gate Ensemble. This extracted data consists of placed and interconnected cell instances.

The SPF file can have one of two formats: detailed SPF (DSPF) format or reduced SPF (RSPF) format. Embedded in each format are two circuit level models, each with different degrees of complexity and accuracy, that represent the physical design.

DSPF Format

The DSPF format contains complete extracted circuit information, including designed and parasitic devices. Circuit simulators can use DSPF for comprehensive simulation, and timing simulators can use DSPF for delay analysis. DSPF also lists circuit-level debugging information so you can identify parasitic layout components and study their effects.

DSPF has two sections: a net section and an instance section. The net section consists of a series of net description blocks. Each net description block corresponds to a net in the physical design. A net description block begins with a net statement followed by pins, instance pins, subnodes, and parasitic resistor/capacitor (R/C) components that characterize the electrical behavior of the net. The instance section consists of a series of SPICE instance statements. SPICE instance statements begin with an X.

The DSPF format is as generic and as much like SPICE as possible. While native SPICE statements describe the R/C sections, some nonnative SPICE statements complete the net descriptions. These nonnative SPICE statements start with the notation “*|” to differentiate them from native SPICE statements.

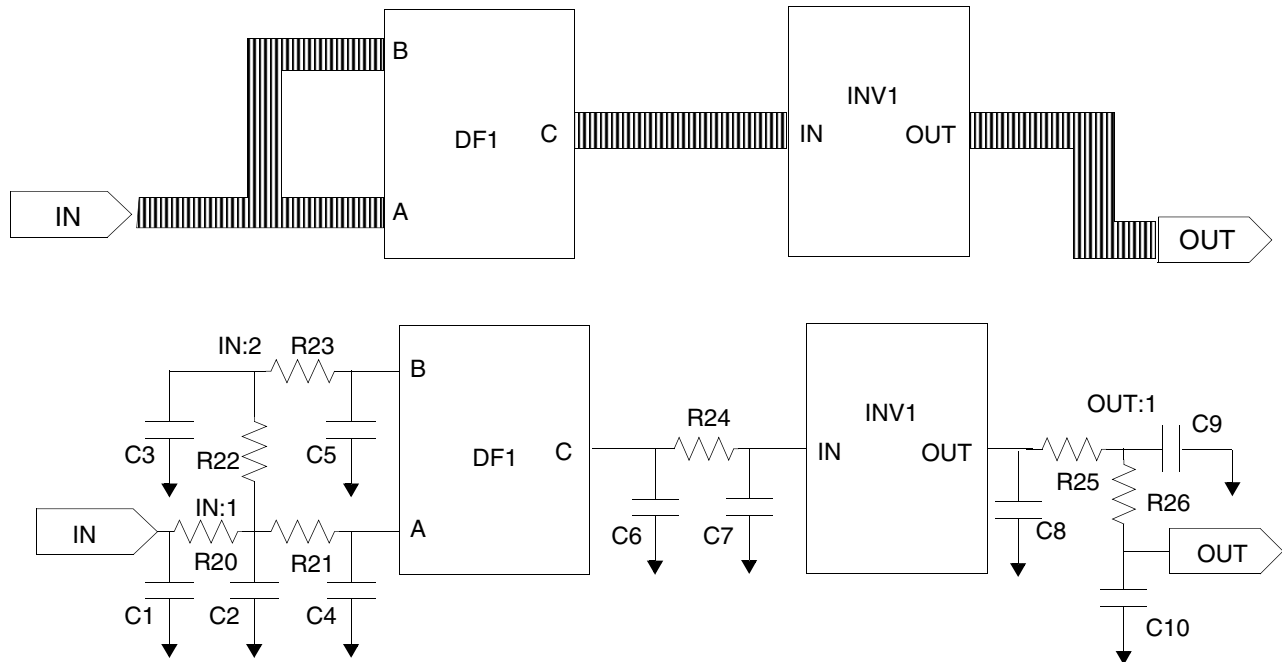
Each file consists of hierarchical cells and interconnects only. When the data also contains designed devices, the DSPF excludes these extracted devices (MOSFETs, bipolar transistors, and so on). These devices are not reported in DSPF.

Dracula Reference

Extracting Parasitic Resistance (PRE)

DSPF Sample

The following sample circuit is shown in DSPF format.



In this example, nonnative SPICE statements start with the notation “*I” to differentiate them from native SPICE statements.

```
.SUBCKT BUFFER OUT IN
*
* Net Section
*
*|GROUND_NET VSS
*
*|NET IN 1.221451PF
*|P (IN I 0.0 0 10)
*|I (DF1:A DF1 A I 0.0 10.0 10.0)
*|I (DF1:B DF1 B I 0.0 10.0 20.0)
*|S (IN:1 5.0 10.0)
*|S (IN:2 5.0 20.0)
C1 IN VSS 1.17763E-01PF
C2 IN:1 VSS 2.76325E-01PF
C3 IN:2 VSS 2.86325E-01PF
C4 DF1:A VSS 2.70519E-01PF
C5 DF1:B VSS 2.70519E-01PF
R20 IN IN:1 1.70333E00
R21 IN:1 DF1:A 1.29167E-01
R22 IN:1 IN:2 1.29167E-01
R23 IN:2 DF1:B 1.70333E00
*
*|NET BF 2.87069E-01PF
*|I (DF1:C DF1 C O 0.0 10.0 10.0)
```

Dracula Reference

Extracting Parasitic Resistance (PRE)

```
*|I (INV1:IN INV1 IN I 0.0 30.0 20.0)
C6 DF1:C VSS 2.08719E-01PF
C7 INV1:IN VSS 7.83500E-02PF
R24 DF1:C INV1:IN 1.80833E-01
*
*|NET OUT 1.48478E-01PF
*|S (OUT:1 45.0 10.0)
*|P (OUT O 0.0 50.0 5.0)
*|I (INV1:OUT INV1 OUT O 0.0 40.0 20.0)
C8 INV1:OUT VSS 1.47069E-01PF
C9 OUT:1 VSS 6.32813E-03PF
C10 OUT VSS 7.76250E-03PF
R25 INV1:OUT OUT:1 3.11000E00
R26 OUT:1 OUT 3.03333E00
*
* Instance Section
*
XDF1 DF1:A DF1:B DF1:C DFF
XINV1 INV1:IN INV1:OUT INV
.ENDS
```

Note: This example does not contain coupling capacitors. Coupling capacitors would normally be listed at the end of the net description block that includes the first node name of the capacitor.

Non-native SPICE syntax and conventions are described below.

NET netName netCap

NET indicates the beginning of a new net description.

netName assigns each net statement a unique net name.

netCap specifies the total capacitance associated with the net. *netCap* includes capacitances to ground and capacitances to nearby wires. It includes a numerical value (which can sometimes be zero) and a unit name, usually PF, with no space between. Because the NET statement includes the total capacitance, you can examine the NET statement to extract the *sim.cap* format from the DSPF format.

Pins, instance pins, subnodes, and all the R/C descriptions that characterize the net follow the NET statement.

P {(pinName pinType pinCap [xCoord yCoord])}

Note: You must specify all pin capacitance values in the timing view library for computation.

P denotes pins in the net. Each pin description is delimited by parentheses and contains the information for its name (*pinName*), its type (*pinType*), its generalized capacitance value (*pinCap*), and its x coordinate location (*xCoord*) and its y coordinate location (*yCoord*). Multiple pin descriptions can be listed on one line.

Dracula Reference

Extracting Parasitic Resistance (PRE)

pinType can be P (power), G (ground), C (clock), I (input), O (output), B (bidirectional), and X (unspecified).

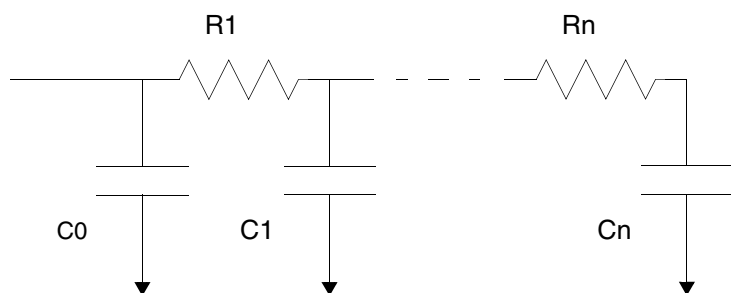
pinCap consists of a capacitance value followed by any number of optional R/C pairs. For example

```
C0 [R1 C1 R2 C2 ...].
```

This format lets you model the pin characteristics by a higher order equivalent R/C ladder circuit than is represented by a single capacitor. The order of the equivalent circuit is increased from the first order (represented by C0) by attaching R/C pairs. The value of each capacitance is determined by a numerical value and a unit specifier such as PF. Use the generalized pin capacitance only if the pin type is I, O, or B. With other pin types, *pinCap* is dropped. In this case, you should use a number (for example, 0) as a place holder.

xCoord and **yCoord** describe a pin location (optional).

Generalized Pin Description



I {(instPinName instName pinName pinType pinCap [xCoord yCoord]..)}

I describes instance pins in the net. Each instance pin description is delimited by parentheses. The *pinName*, *pinType*, *pinCap*, and *xCoord* and *yCoord* follow the same format as described in the P statement. The description of the instance pin location is optional. Multiple instance pin descriptions can be listed on one line. The *xCoord* and *yCoord* can be repeated for feedthrough pins.

instPinName is the instance pin name component obtained by concatenating the *instName* and the *pinName* with a delimiting character specified by the DELIMITER statement.

instName is the instance name component.

pinName is the name of the pin in the master of the instance.

S {(subnodeName [xCoord yCoord])}

Dracula Reference

Extracting Parasitic Resistance (PRE)

S denotes subnodes in the net. Each subnode description is delimited by parentheses and contains the name of the subnode and its location. A subnode name is obtained by concatenating the net name and a subnode number by a colon (*netname:1*). Multiple subnode descriptions can be listed in one line.

GROUND_NET netName

GROUND_NET specifies the ground net for loading capacitors. It must precede the NET statement.

netName is the name of the ground net.

DELIMITER "C"

This statement specifies the character *C* as the delimiter used to construct the instance pin name *instPinName*. If you do not include this statement, the default delimiter character, colon (:), is used.

For native SPICE statements, a continuation line begins with a plus sign (+) in the first column. For other statements, a continuation line begins with an asterisk and a plus sign (*+) in the first and second columns. The maximum number of continuation lines allowed is 1000.

Note: You might see instances of resistors and capacitors starting with the instance names RD and CD, respectively. These are dummy instances created for the CDC, which requires a pie model (capacitor-resistor-capacitor) for the RC mesh. If there are cases that do not fit the pie model, dummy resistors or capacitors with a value of zero are added where needed.

RSPF Format

The DSPF file contains complete circuit information extracted by a parasitic extractor. Although this information is indispensable for debugging purposes, this can be too much data to use for circuit simulation or timing simulation.

One of the major applications of the parasitic circuit extractor is analysis of interconnect nets created by automatic placement and routing tools. In such applications, you do not usually require comprehensive circuit simulation of the parasitics; however, you might be very concerned about the total delays introduced by interconnect because the interconnect delays tend to limit the overall system performance. Traditionally, a delay calculator is invoked to perform the delay analysis. Arbitrary interconnect structures have to be reduced to a simpler form to allow fast delay calculation. The most common simplification is to reduce an interconnect net to a total capacitance, ignoring the effects of resistance and net topology.

The RSPF format is very similar to DSPF format except for the syntax in the *spf_type* and *net_block* sections. In addition to the keywords defined in the DSPF file, two extra keywords are used in the RSPF format: the DRIVER and the LOAD statements. The purpose of these two statements is to identify the driver-load relationship. The following syntax description shows only the syntax specific to RSPF.

*|RSPF {qstring}

qstring represents the version number of the SPF format.

NET netName driver_def {subNode_def} resistor_def2 capacitor_def2 {load_def
{subNode_def} voltage_def resistor_def?}

NET indicates the beginning of a new net description.

driver_block describes the driver pin of the net followed by an optional subNode, then two resistors and two capacitors to form the driver block.

subNode_def is the same as that in DSPF format except that it does not output a subnode's coordinates.

load_block describes the load pin of the net followed by an optional subNode, then a voltage-controlled voltage source, resistor, and capacitor to form the load block. The load block is optional because one pin nets are typically drivers with no load.

driver_def represents the driver pin of the net. It starts with the DRIVER keyword followed either by a top-level pin or an instance pin. For an instance pin, the instance pin name is obtained from the instance name and the pin name by concatenating them with a delimiter. The default delimiter is a colon (:).

Dracula Reference

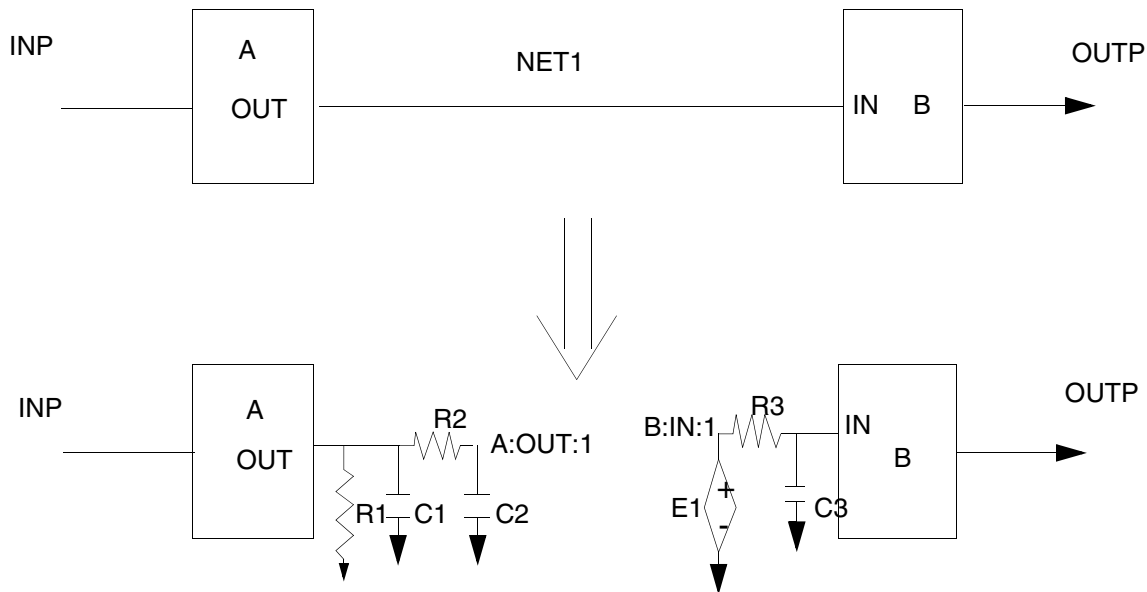
Extracting Parasitic Resistance (PRE)

load_def represents the load pin of the net. It starts with the LOAD keyword followed either by a top-level pin or an instance pin.

voltage_def is a native SPICE statement.

RSPF Example

The following sample circuit is shown in RSPF format.



```
*|RSPF 1.0
*
*|DESIGN "ABC"
*|DATE "September 8, 1995 20:10:00"
*|VENDOR "CADENCE"
*|PROGRAM "PREVIEW"
*|VERSION "PREVIEW_1.0"
*|DIVIDER /
*|DELIMITER :
*
.SUBCKT MODULE1 OUTP INP
*|GROUND_NET VSS
*
* Net Section

*|NET INP 1.7534PF
.
.
.*
*|NET NET1 0.8405578PF
```

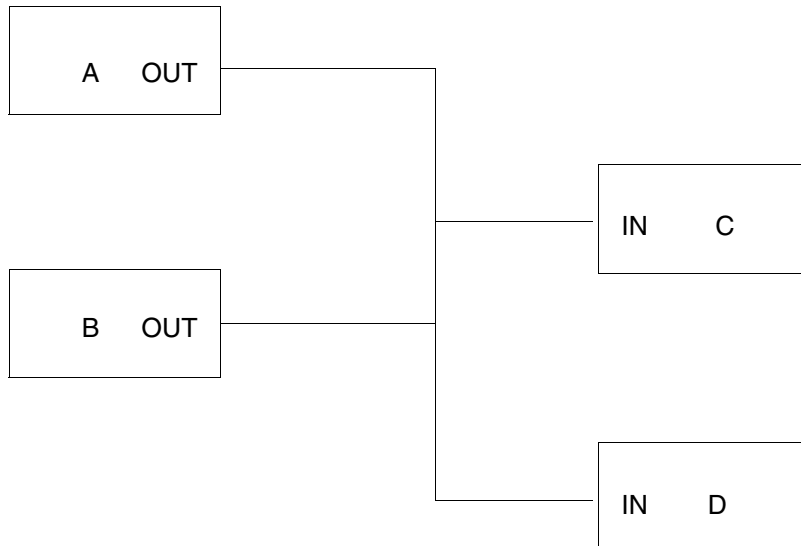
Dracula Reference

Extracting Parasitic Resistance (PRE)

```
*|DRIVER A:OUT A OUT
*|S (A:OUT:1)
R1 A:OUT VSS 2.84626E02
R2 A:OUT A:OUT:1 6.23854E02
C1 A:OUT VSS 0.533474PF
C2 A:OUT:1 VSS 0.392564PF
*|LOAD B:IN B IN
*|S (B:IN:1)
E1 B:IN:1 VSS A:OUT VSS 1.0
R3 B:IN:1 B:IN 1.4835E03
C3 B:IN VSS 1.0PF
*
*|NET OUTP 0.83975PF
...
.
*
*Instance Section
*
XA ...
XB ...
.
.ENDS
```

RSPF Example with Multiple Drivers and Multiple Loads

For a general multiple-driver and multiple-load net, a separate equivalent reduced model must be created for each driver-load pair.



```
*|RSPF "1.0"
*
*|DESIGN "ABC"
*|DATE "December 8, 1995 20:10:00"
*|VENDOR "CADENCE"
*|PROGRAM "PREVIEW"
```

Dracula Reference

Extracting Parasitic Resistance (PRE)

```
*|VERSION "PREVIEW_1.0"
*|DIVIDER "/"
*|DELIMITER ":"
*

.SUBCKT MODULE2 OUTP INP
*|GROUND_NET VSS
*
*Net Section
*
.
.
.
*|NET net 5.4982PF
* 1st driver pin group

*|DRIVER A:OUT A OUT
*|S (A:OUT:1)
R1 ...
R2 ...
C1 ...
C2 ...
*|LOAD C:IN C IN
*|S (C:IN:1)
E1 ...
R3 ...
C3 ...
*|LOAD D:IN D IN
*|S (D:IN:1)
E2 ...
R4 ...
C4 ...
* 2nd driver pin group
*|DRIVER B:OUT B OUT
*|S (B:OUT:1)
R5 ...
R6 ...
C5 ...
C6 ...
*|LOAD C:IN C IN
*|S (C:IN:1)
E3 ...
R7 ...
C7 ...
*|LOAD D:IN D IN
*|S (D:IN:1)
E4 ...
R8 ...
C8 ...
*
*|NET next_net 1.9387PF
.
.
.
*
*Instance Section
*
.
.
.
```


Dracula Reference

Extracting Parasitic Resistance (PRE)

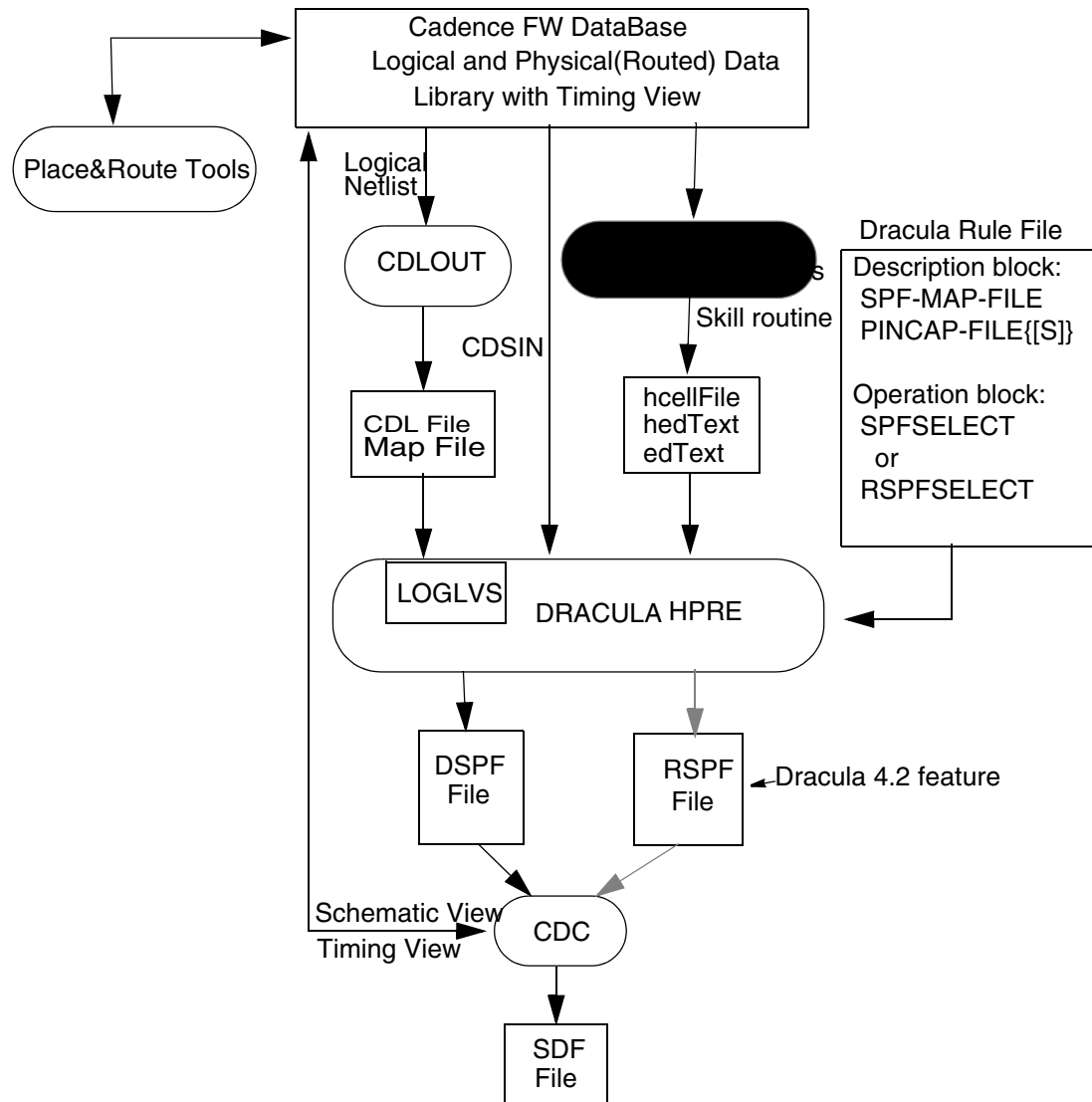
```
XA ...  
XB ...  
XC ...  
XD ....  
.  
.  
.ENDS
```

Dracula Reference

Extracting Parasitic Resistance (PRE)

Dracula, CDC, DSPF and RSPF Flow

The following example illustrates the flow between Dracula, CDC, DSPF, and RSPF.



Using DRAC2CDL Netlist Generator

Overview

DRAC2CDL generates a CDL netlist file from certain Dracula generated binary files. These binary files are the:

- LVSLOGIC.DAT file generated by LOGLVS. The file can be created from processing a simple CDL or Spice netlist or from a combination of a CDL/Spice netlist and a Verilog database generated by the van executable.
- 6NETS.DAT and 6NETL.DAT files created during a Dracula LVS run. These files are the finalized schematic and layout netlists used by Dracula LVS for comparison. When you run DRAC2CDL on either of these two files, the resulting CDL netlist file will reflect the filter and reduction options specified in the Dracula LVS rules file.

Running DRAC2CDL

DRAC2CDL can be run interactively or in batch mode just like PDRACULA or LOGLVS.

Interactive Mode

To run DRAC2CDL interactively, simply execute the following command from the shell:

```
user@host> DRAC2CDL
```

A menu list of commands will appear, followed by a colon (:) prompt, as follows:

```
*****
*/N*  DRAC2CDL (REV. xxxxxxxx.mm-yyyy / LINUX /GENDATE: dd-MMM/yyyy)
      *** ( Copyright yyyy, Cadence ) ***
*/N*  EXEC TIME = hh:mm:ss   DATE = dd-MMM-yyyy   HOSTNAME =   hostname
*****
1)infile <filename> - set name of Dracula netlist file. Default is
"LVSLOGIC.DAT".
2)outfile <filename> - set name of output cdl file. Default is "out.cdl".
```

Dracula Reference

Using DRAC2CDL Netlist Generator

```
3)header <filename> - set name of header include file. Default is "head.cdl".
4)swapbyte {yes|no|auto} - set byte swapping mode for cross-platform run.
Default is auto.
5)r(un) - translate netlist and exit.
6)h(elp) - show this help.
7)exit/x - exit without doing anything.
:
```

The various commands are explained next.

INFILE <filename>

The default input to DRAC2CDL is the `LVSLOGIC.DAT` file. You can omit the `INFILE` command when generating a CDL netlist from default input file.

Use this command to specify the input binary file for translation to a CDL netlist.

OUTFILE <filename>

By default the output CDL netlist is written to a file called `out.cdl`. Use the `OUTFILE` command to redirect the output to a different file.

HEADER <filename>

In order to increase performance, DRAC2CDL writes global signals to a separate header file, which is referenced by the output CDL netlist file via an include statement (`. INCLUDE`). The default header name is `head.cdl`.

Use the `HEADER` command to specify a different CDL header file name.

SWAPBYTE {yes | no | auto}

Potential issues concerning byte-order can occur when running DRAC2CDL using binary data generated from a different Operating System. By default, DRAC2CDL automatically determines whether to swap bytes or not; however, in some cases it may fail. Should this occur, use the `SWAPBYTE` command with one of the following options:

YES

Swaps bytes on the input binary data. This may be needed when running DRAC2CDL using input binary files generated from a different Operating System.

Dracula Reference

Using DRAC2CDL Netlist Generator

NO

Does not to swap bytes on binary input data. It may be useful in rare instances when DRAC2CDL wrongly determines input file-types.

AUTO

Automatically determines whether to swap bytes or not. **This is the default.**

R(UN)

Generates the CDL netlist file based on the input binary file and automatically exits DRAC2CDL. The subcircuit's name in the CDL netlist is “top”.

You can simply type the letter ‘R’ or ‘r’ at the prompt. The command is not case sensitive.

H(ELP)

Redisplays the menu list of commands.

You can type ‘H’ or ‘h’ at the prompt. The command is not case sensitive.

EXIT/X

Exits DRAC2CDL.

Batch Mode

DRAC2CDL can also run in batch mode just like PDRACULA or LOGLVS using standard Unix input redirect:

```
DRAC2CDL << !
infile 6NETS.DAT
outfile my.cdl
r
!
```

Examples

Example 1: This is the simplest example of running DRAC2CDL in interactive mode. It will generate the default head.cdl and out.cdl files from the LVSLOGIC.DAT file:

```
user@host> DRAC2CDL
```

Dracula Reference

Using DRAC2CDL Netlist Generator

```
*****
*/N*  DRAC2CDL (REV. xxxxxxxx.mm-yyyy / LINUX /GENDATE: dd-MMM/yyyy)
      *** ( Copyright yyyy, Cadence ) ***
*/N*  EXEC TIME = hh:mm:ss DATE = dd-MMM-yyyy HOSTNAME = hostname
*****

1)infile <filename> - set name of Dracula netlist file. Default is
"LVSLOGIC.DAT".
2)outfile <filename> - set name of output cdl file. Default is "out.cdl".
3)header <filename> - set name of header include file. Default is "head.cdl".
4)swapbyte {yes|no|auto} - set byte swapping mode for cross-platform run.
Default is auto.
5)r(un) - translate netlist and exit.
6)h(elp) - show this help.
7)exit/x - exit without doing anything.
:r
Translation of LVSLOGIC.DAT finished. Output results are in out.cdl
Bye!
```

If the original CDL netlist processed with LOGLVS to generate the LVSLOGIC.DAT file is as follows:

```
*.BIPOLAR
*.RESVAL
*.EQUATION
*.SCALE METER
*.MEGA
.PARAM
*.GLOBAL VDD! GND!

.SUBCKT inv A Y
MM1 Y A GND! GND! NM W=3.1u L=2.5u M=1
MM0 Y A VDD! VDD! PM W=3.1u L=2.5u M=1
.ENDS

.SUBCKT buffer IN OUT
RR0 net8 net3 1K ${PR}
XI1 net3 OUT / inv
XI0 IN net8 / inv
.ENDS
```

The following will be the output header file (head.cdl):

```
* Header include file
*.bipolar
*.ldd
*.resval
*.capval
.param
* PAD name:VDD! model:P number:1
.GLOBAL 1:P
*.EQUIV VDD!=1
* PAD name:GND! model:G number:2
.GLOBAL 2:G
*.EQUIV GND!=2
```

Dracula Reference

Using DRAC2CDL Netlist Generator

```
* PAD name:IN model:O number:3
.GLOBAL 3:O
*.EQUIV IN=3
* PAD name:OUT model:I number:4
.GLOBAL 4:I
*.EQUIV OUT=4
* The end of include file
```

The following will be the output netlist file (out.cdl):

```
.include head.cdl
.subckt top
* NODE name: VDD! number:1
* NODE name: GND! number:2
* NODE name: IN number:3
* NODE name: OUT number:4
R1 6 7 1000 ${PR} * RR0
M2 4 7 2 2 NM W=3.1 L=2.5 M=1 * MM1XI1
M3 4 7 1 1 PM W=3.1 L=2.5 M=1 * MM0XI1
M4 6 3 2 2 NM W=3.1 L=2.5 M=1 * MM1XI0
M5 6 3 1 1 PM W=3.1 L=2.5 M=1 * MM0XI0
* NODE name:NET8 number: 6
* NODE name:NET3 number: 7
.ends top
* The end
```

Example 2: Generates the default head.cdl file and a CDL netlist file called “layout.cdl” from the finalized 6NETL.DAT layout netlist file in batch mode:

```
DRAC2CDL << !
infile 6NETL.DAT
outfile layout.cdl
r
!
```

Dracula Reference

Using DRAC2CDL Netlist Generator

Description Block Commands

This chapter documents the Description Block Commands for Dracula in alphabetical order.

ABORT-MULT-STAMP

ABORT-MULT-STAMP = YES/NO

Description

Ends checking when a multi-stamped situation occurs. This command is useful when running large jobs that take a long time to complete. When `ABORT-MULT-STAMP = YES`, the relative STAMP module terminates, and the output error cells are still generated in the output GDS file for you to debug. The files that `draculaInteractive` needs will be kept so that you can debug the error layers in the `draculaInteractive` environment, if you have already set `KEEPDATA = YES` or `KEEPDATA = DRACULAINTERACTIVE`.

The Dracula tool creates a new file called `MULSTAMP.DAT`, which is an ASCII file reporting the multi-stamped locations. This file contains the "UPPER" and "LOWER" trapezoids with the location and the node number. This file is automatically generated when there is a multi-stamp or a soft connection violation.

Note: `KEEPDATA = DRACULAINTERACTIVE` works only in Dracula 4.8 and IC 445 version and higher versions. The command `KEEPDATA = INQUERY` will only work in Dracula versions that are lower than Dracula 4.8 and the IC 445 version.

Arguments

YES

Terminates checking when a multi-stamped violation occurs

NO

Is the default

Example

ABORT-MULT-STAMP = YES

The following is the format for `MULSTAMP.DAT`:

```
/*-----*/
LOWER-LAYER: MET1
trapezoid: (10, 10), (20, 10), (10, 20), (20, 20)

UPPER-LAYER: POLY
trapezoid: (15, 7), (18, 7), (15, 12), (18, 12) NODE: 2
trapezoid: (14, 17), (17, 17), (14, 25), (17, 25) NODE: 3
```

Dracula Reference

Description Block Commands

```
/*-----*/  
LOWER-LAYER: MET1  
trapezoid: (30, 30), (40, 30), (30, 40), (40, 40)  
  
UPPER-LAYER: POLY  
trapezoid: (25, 32), (32, 32), (25, 35), (32, 35) NODE: 4  
trapezoid: (35, 27), (37, 27), (35, 33), (37, 33) NODE: 5
```

ABORT-P-G-SHORT

ABORT-P-G-SHORT = YES/NO/ALL/SHORT/OPEN

Description

Specifies whether to abort a job if a power or ground node shorts or opens. If the text you specify as power or ground shorts or opens, the job aborts at the *POSATT* module. No *PRINTFILE.SUM* is generated. You must check your log file for information on the short or open. Power text can be VCC, VDD, or *:P. Ground text can be VSS, GND, GROUND, or *:G.

To find the power or ground short or open, you must remove the ABORT-P-G-SHORT command and add the MULTILAB and SAMELAB commands in the Operation block.

Arguments

YES

Aborts a job if a power or ground node is open or a power is shorted with a ground.

NO

Default.

ALL

Aborts a job if a power or ground node is open or shorted with any other signal.

SHORT

Aborts a job if a power or ground node is shorted with any other signal.

OPEN

Aborts a job if a power or ground node is open.

Examples

ABORT-P-G-SHORT = YES

ABORT-P-G-SHORT = SHORT

ABORT-SOFTCHK

ABORT-SOFTCHK = YES/NO

Description

Aborts checking when a soft connection violation occurs. This command is useful when running large jobs that take a long time to complete. When ABORT-SOFTCHK = YES, an abort alerts you to a major SOFTCHK problem prior to spending CPU time running the complete job.

To examine a SOFTCHK problem, do one of the following:

- Run another job, which runs only the SOFTCHK commands, through completion.
- Re-run the same job with ABORT-SOFTCHK = NO. Take your existing rule file and eliminate all commands not needed for the SOFTCHK rules. Be sure to change ABORT-SOFTCHK = YES to ABORT-SOFTCHK = NO. Then run this second job from beginning to end.

Arguments

YES

Aborts checking when a soft connection violation occurs.

NO

Default.

Example

ABORT-SOFTCHK = YES

ACUTE-CLIP

ACUTE-CLIP = YES/NO

Description

Modifies path geometries that form an acute angle. This command clips the outside corner of the geometry to the width of the path.

Arguments

YES

Modifies path geometries.

NO

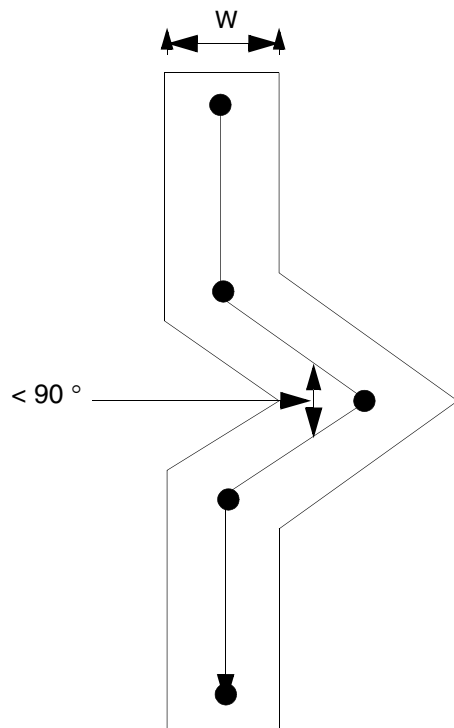
Default.

Dracula Reference

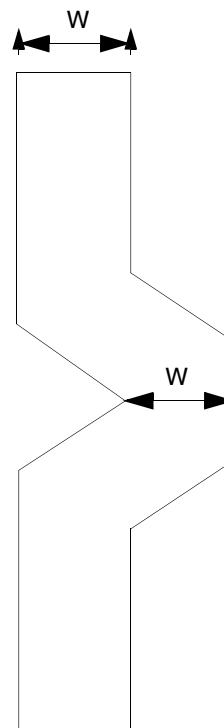
Description Block Commands

Example

ACUTE-CLIP = YES



Digitized path



ACUTE-CLIP = YES

ALLOW-RECONNECT

ALLOW-RECONNECT = YES/NO

Description

Use this command to turn-on incremental connection which allows reconnection up to 16 times. SOFTCHK must be placed right after the connect group when you use this command. See the Example 3 in the “SCONNECT” command section of chapter 13.

Arguments

YES

Allow incremental connection

NO

Is the default

ANGLED-EDGE-TOL

ANGLED-EDGE-TOL [=] {<value> | YES/ON | NO/OFF}

Description

This command enables and defines, or disables a tolerance setting for the DRC checking operation on non-manhattan edges.

See “[ANGLED-EDGE-TOL](#)” on page 876 for a complete description of this command.

ARRAY

ARRAY = location array-name

Description

Flags cells that form an array. This command checks the array and truncates it to its minimal configuration. You can specify special array- handling options to conserve resources. Use ARRAY only for GDSII (maximum of 80).

Arguments

location

Expands the cell into an array and specifies the portion to check.

NONE

Keeps none of the array.

BOUNDARY

Keeps only the boundary.

CORNER

Keeps only the corners.

array-name

Cell name from which to make the array. Expansion dimensions are in the input file.

Examples

ARRAY = BOUNDARY A

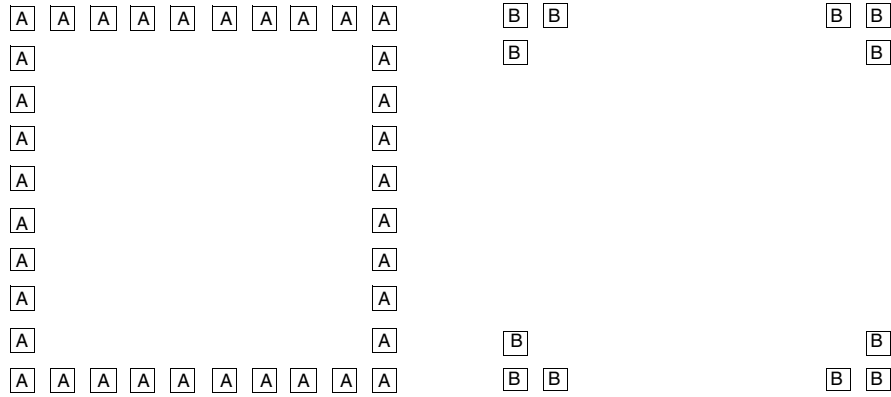
The array is a 10 x 10 expansion of cell A.

ARRAY = CORNER B

Dracula Reference

Description Block Commands

The array is a 10x10 expansion of cell B.



ARRAY-ENABLED

ARRAY-ENABLED = YES/NO

When ARRAY-ENABLED=YES, Dracula searches your layout to find regular array structures. When array structures are found, only the outer ring of array structures is checked against foreign cells. This command should be used to memory designs and ASIC designs with embedded memory.

If you are in an intermediate verification stage and need quick results, you can use ARRAY-ENABLED. Because Dracula does not process the entire array structures, ARRAY-ENABLED reduces the amount of data processed.

Checking Method

modes

Flat

Arguments

YES

Performs array processing.

NO

Default.

Examples

ARRAY-ENABLED = YES

AUTOMATCH

AUTOMATCH = YES/NO/SMART

Description

Directs LVS to use an alternate method that does not require text assigned to the initial correspondence points. LVS usually requires text assigned to the initial correspondence points. AUTOMATCH can find these points when they are not assigned text.

LVS automatically switches to the AUTOMATCH algorithm when it cannot locate initial correspondence points. This automatic switch occurs even if you did not use the AUTOMATCH command in your run file. AUTOMATCH is useful when you check LVS in composite mode.

You can use the AUTOMATCH command even if you placed text on the layout to identify initial correspondence points. Text is essential to match groups of corresponding signals whose interconnections are not identifiable to LVS. For example, an array of data bus signals can have similar interconnection characteristics. LVS can identify these similar signals only when you place unique text on the layout. Also, when LVS reduces transistors to gates or parallel/series structures, LVS requires text on the layout to identify power and ground nodes and ambiguous signals.

Note: Do not use the FILTER options *P* or *Q* when you use AUTOMATCH because AUTOMATCH can locate isolated layout areas in a composite layout.

Arguments

YES

Directs LVS to use an alternate algorithm to locate initial corresponding points.

NO

Turns off AUTOMATCH function.

SMART

When LVS cannot locate initial correspondence points, *SMART* turns on the AUTOMATCH function. Default.

Example

AUTOMATCH = YES

BLACKBOX-FILE

BLACKBOX-FILE = *filename*

Description

Specifies the black box text file name. The black box file defines Hcells to be masked by black boxes. Each line in the file must list the following items:

- The text name
- The x and y coordinates

Dracula centers the pin geometry at these coordinates. The coordinates are relative to Hcell coordinates. The x and y are not case sensitive. Optionally, you can use width (w) and height (h) specifications to define the shape of the pin. The default for both is two database units. The w and h are not case sensitive. The black box file accepts a tab as a delimiter.

- The attached layer

If you do not specify ATTACH, Dracula attaches the text according to the CONNECT-LAYER reference.

- The name of the Hcell

There are three ways to apply black box-like features in Dracula composite mode checking:

- To make the BLACKBOX-FILE command more powerful, modify the database so that black box Hcells are empty. You do this by referencing the Hcell to a dummy empty cell so only small squares around points defined by the BLACKBOX-FILE command exist after Dracula runs the EXPAND module. Because Dracula processes no other geometry, this improves performance.
- When you specify the FRAME BY option for an Hcell in composite mode checking, all geometries inside the cell (except those falling in the ring of the cell) are deleted. The ring size is defined as the FRAME BY value. If you specify the GEN-TEXT-FRAME command, all Hcells are considered FRAME BY Hcells, and the ring size is defined as the GEN-TEXT-FRAME value. You must make sure all texts needed for connections are on the frame of the Hcells. For this feature, the BLACKBOX-FILE command is not mandatory.
- For some designs, you might want to keep the geometries from only some of the connecting layers for some of the Hcells (black box). You can append the stopping connecting layer name to an Hcell so that only the geometries on or above this connecting layer are read. For this feature, the BLACKBOX-FILE command is not mandatory.

Dracula Reference

Description Block Commands

For more information about using black boxes, refer to [“LVS Overview”](#) on page 275.

Checking Method

modes

Composite.

checks

HERC, HLVS, and HLPE.

Example 1

```
BLACKBOX-FILE abc.htx
```

Where *abc.htx* is

IN	x=41	y=90	ATTACH=poly	inv	w=0.6	h=1
OUT	x=89	y=44	ATTACH=metal	inv		
VDD	x=123	y=96	ATTACH=met	nand	w=10	

Example 2

The following is an example for specifying stopping layers:

```
*DESCRIPTION
.
.
HCELL-FILE = blackbox.tab
.
.
*END
*INPUT-LAYER
.
.
CONNECT-LAY = PWELL MET1 MET2
.
.
*END
```

Where *blackbox.tab* is

INV1	INV	MET2
NAND1	NAND	MET1
NA2	NA2	

For *INV1*, only *MET2* geometries are read. For *NAND1*, both *MET1* and *MET2* geometries are read because *MET2* is specified before *MET1*. For *NA2*, data of all the input layers is read.

Dracula Reference

Description Block Commands

Example 3

HCELL-FILE = cell.tab

where *cell.tab* is

```
INV      INV1      MET1
NAND     NAND      MET2
XYZ      XYZ FRAME BY 2.0      MET2
DDD      DDD
ABC      ABC      MET
```

If Hcells have layers appended, only the geometries of the specified layer and layers above are read in.

```
*INPUT-LAYER.
;
PWELL=1
DIFPN=2
POLY =3      ;CTEXT=28 ATTACH=POLY
CONT=5
MET=6 ;CTEXT=29 ATTACH=MET
VAPOX=7 ;TEXT=30
PISL=4
SUBSTRATE= BULK 59
;
MET2=10
VIA=11
;
CELLBNDY = 54      ;BOUNDRY LAYER FOR RECTILINEAR CELLBOXX
;
CONNECT-LAY= NSUB PWELL PSRCDRN NSRCDRN POLY MET MET2
*END
```

The Hcells read in geometries according to the following table:

Cell	Layer
INV	MET2, MET1
NAND	MET2
XYZ	MET2
DDD	All layers
ABC	MET2, MET, POLY

BLOCK-NAME-ONLY

BLOCK-NAME-ONLY = YES/NO

Description

When verifying an Edge database, you can use the layout block name as a cell name for LVS checking. In most circumstances, the rep name is “layout” and the revision name is “current.” For example, the command BLOCK-NAME-ONLY = YES assumes all cell names have the suffix “/layout/current” and chooses only the block name as the cell name to verify.

Arguments

YES

Selects only the block name when encountering a cell name.

NO

Default.

Example

BLOCK-NAME-ONLY = YES

BOUNDING-BOX

BOUNDING-BOX = *cell-name layer-name*

Description

Specifies the bounding box of a cell as an identification layer for special checks. Used for hierarchical and composite modes only.

Arguments

cell-name

Name of the cell.

layer-name

Name of the layer used for the bounding box of the cell. You can use this trapezoid layer in the Operation block for layer operations.

Example

```
*DESCRIPTION
.
.
BOUNDING-BOX = MATX, MATRICE
.
.
*END
.
*INPUT-LAYER
.
ACTIVE = 2
.
*END
*OPERATION
.
AND ACTIVE MATRICE ACTMAT
NOT ACTIVE MATRICE ACT
EXT ACTMAT LT 0.8 DRC1 50
EXT ACT    LT 1.0 DRC2 51
.
.
*END
```

Dracula Reference

Description Block Commands

BOXMAP

```
BOXMAP = layout_box_element_name1/schematic_subckt_name1  
        ,layout_box_element_name2/schematic_subckt_name2  
        ...  
        ,layout_box_element_nameN/schematic_subckt_nameN
```

Description

Specifies the corresponding layout box element the way that a schematic box element should be compared to in an LVS comparison. The command overwrites the *.EQUIV specification in the schematic netlist if there's a conflict.

Arguments

layout_box_element_name

The user-defined device name in the element box command.

schematic_subckt_name

The schematic .SUBCKT name for a schematic box element.

Note: You need to put a comma to separate one set of the

layout_box_element_nameN/schematic_subckt_nameN names from the next set of *layout_box_element_nameN/schematic_subckt_nameN* names.

Example

Schematic netlist

```
*.EQUIV IN1=spiral_rad  
.SUBCKT spiral_rad T1 T2  
.ENDS  
.SUBCKT spiral_turn T1 T2  
.ENDS  
.SUBCKT TEST1 P1 P2 P3 P4  
X2 P1 P2 spiral_rad nr=4.5 rad=6e-05 w=1.5e-05 s=1.5e-06  
X3 P3 P4 spiral_turn nr=3.5 rad=5e-05 w=1.5e-05 s=1.5e-06  
.ENDS
```

Dracula Reference

Description Block Commands

LVS rule

```
*DESCRIPTION
.
.
*DESCRIPTION
.....
PARSET INDP ID IN IW IS LL
BOXMAP = IN2/spiral_rad, IN3/spiral_turn
....
*END
...
*OPERATION
...
ELEMENT IN2 ME5I1 ME4 ME5A
ELEMENT IN3 ME5I2 ME4A ME5B
...
LEXTRACT INDP ME5I2 BY IN2 INDPAR1
EQUATION LL = ID+IN+IW+IS
LEXTRACT INDP ME5I2 BY IN3 INDPAR2&
EQUATION LL = ID+IN+IW+IS
.....
```

In the above example, the BOXMAP command specifies that instances of the schematic box element `spiral_rad` should be compared with instances of the layout box element `IN2` (overwrites the `*.EQUIV` specification). Instances of the schematic box element `spiral_turn` should be compared with instances of the layout box element `IN3`.

BOX-M-COMPARE

BOX-M-COMPARE = YES/NO

Description

Use with BOX-M-FACTOR=YES to smash parallel box, but not take m-factor into account while comparing.

Arguments

YES

All parallel boxes will be smashed and m-factor will be taken into account while LVS comparison. **This is the default.**

NO

Parallel boxes will be smashed but the comparison will not carry about m-factor.

Examples

BOX-M-FACTOR = YES

BOX-M-COMPARE = NO

BOX-M-FACTOR

BOX-M-FACTOR = YES/NO

Description

Allow Dracula to smash parallel boxes.

Arguments

YES

All parallel boxes of the same type will be smashed. If the box instance has a parameter named "M" this parameter is used as the `M-factor` for this box instance. If there is no "M" parameter, the `M-factor` is 1. See ["BOX-M-IGNORE"](#) on page 535 and [BOX-M-COMPARE](#) on page 533.

NO

Parallel boxes will not be smashed. **This is the default.**

Examples

BOX-M-FACTOR = YES

BOX-M-IGNORE

BOX-M-IGNORE = boxtype_1 boxtype_2 boxtype_3

Description

Use with BOX-M-FACTOR = YES to specify boxes that will not be smashed.

Arguments

boxtype_1

Specify the type of boxes to exclude from the smashing operation.

Examples

BOX-M-FACTOR = YES

BOX-M-IGNORE = IND1 VART

CAP-POLARITY

`CAP-POLARITY = subtype1, subtype2, ...`

Description

The 'P' option of LVCHK/LPECHK commands checks the ELEMENT capacitor's polarity regardless of the subtype, which is all right when there is only one type of capacitor. However, with more than one kind of capacitors, Dracula has problems controlling over what types of capacitors should have polarity check.

In the 4.81 and subsequent versions, this new command, CAP-POLARITY lets the 'P' option of the LVCHK and LPECHK commands to only have impact on the specified subtypes of capacitors. That is,

- If the CAP-POLARITY command *is not used* and if the P option is turned on , the polarity of every capacitor will be checked regardless of its subtype.
- If CAP-POLARITY command *is used*, only the capacitors of the subtypes specified in this command will be checked.

Note: Wildcards are not supported in this new command.

Arguments

subtype?

One or two-character CAP subtypes

Note: The maximum types of capacitance subtypes is 10.

Example

`CAP-POLARITY = A, B`

This example specifies that only capacitors of subtypes A and B should have a polarity check.

CARE-SPLIT-ORDER

CARE-SPLIT-ORDER = YES/NO

Description

Checks the input order of split gates and merges them into one series structure. If the input order is not the same, this command does not merge the gates. Use this command for LVS or HLVS jobs.

You must also specify the S option in LVSCCHK or LPECHK. If you use the command without the S option, a warning message appears. Refer to the [FIX-INPUT-ORDER](#) section for cases when Dracula automatically invokes CARE-SPLIT-ORDER.

Note: When you specify FIX-INPUT-ORDER with INSTANCE, EXCEPT-LOGIC, or YES arguments, CARE-SPLIT-ORDER is automatically set to YES.

Arguments

YES

Merges the gates into one series structure only when the input order of the gates is the same.

NO

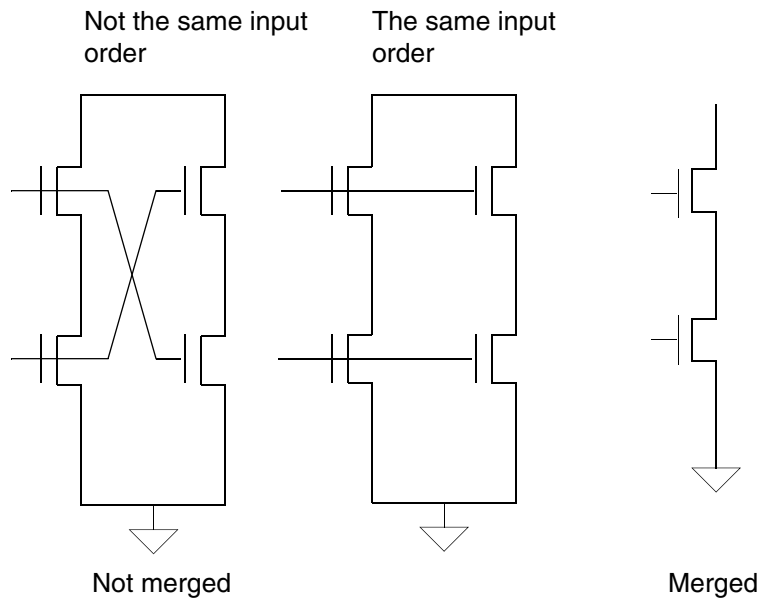
Default.

Dracula Reference

Description Block Commands

Example

CARE-SPLIT-ORDER = YES



CELLBOX-LAYER

CELLBOX-LAYER = *l-num*

Description

Creates cell boundaries, referred to as *cell boxes*, for each Hcell. Dracula places these *cell boxes* into a cell named *OCCELLBOX* under a layer number specified by the *l-num* parameter. The coordinates of cell boxes are relative to *0,0* of the composite plane. This cell also contains textual Hcell information.

Checking Method

modes

Hierarchical and composite.

checks

HDRC, HERC, HLVS, and HLPE.

Arguments

l-num

Specifies the layer name for the cell box. For GDSII format, *l-num* has a range of 0 to 63.

The format of this cell (GDSII) is the same as the original database unless you specify otherwise with a SYSOUT command. Therefore, you can display the Hcell boxes from your original graphics system.

Note that Dracula creates another file named CELLBOX.X.DAT, which Dracula uses internally and is accessible by using Dracula Interactive.

Example

CELLBOX-LAYER = 59

CELL-CHILD-TEXT

CELL-CHILD-TEXT = NO/YES

Description

Specifies whether to place layout text originating from layout cells contained in Hcell child cells in the Hcell plane. By default, Dracula places only text originating in the Hcells.

Checking Method

modes

Cell and composite.

checks

HERC, HLVS and HLPE.

Arguments

NO

Uses only text originating in the Hcells. Default.

YES

Places child cell text in the Hcell plane. Expands text originating in Hcell child cells into the Hcell if it is global to all placements of the individual child cells, such as for VDD and GND. If nonglobal text exists in Hcell child cells, the YES option results in numerous unrelated pins shorting together.

Example

CELL-CHILD-TEXT = YES

CELL-ERROR-REP

CELL-ERROR-REP = ONCE/HIER/ALL/ORIG-ALL/ORIG-DRC/*cell-name*

Description

Specifies the format of cell-based error segments generated by hierarchical and flat mode HDRC checks. Hcell errors are errors in cells and cell environments that are common to all placements of a given cell. For details about checking for these errors, refer to [Chapter 3, “Checking Design Rules \(DRC\).”](#)



CELL-ERR-REP is intended for FLAT or HIER values of CHECK-MODE. Dracula ignores CELL-ERR-REP if CHECK-MODE contains a value other than FLAT or HIER.

CELL-ERR-REP only functions for DRC jobs. Dracula ignores CELL-ERR-REP if any ERC, LVS, or LPE commands are present in the rule file

Arguments

ONCE

Causes the cell error-flag outputs for an Hcell to be instantiated in the OUTPUT error cell at a location corresponding to the first placement of the Hcell in the database starting from the lower left corner. Dracula places the error segments relative to the composite plane within the error cell. Dracula displays Hcell errors only once regardless of how many times you place a particular Hcell. Use in hierarchical mode only. Default.

HIER

Causes the cell error-flag outputs for an Hcell to be instantiated in the OUTPUT error cell at all locations where the Hcell is instantiated in the original database. Dracula reports the error output in a two-level hierarchy. Dracula places Hcell errors in an Hcell error cell and invokes instantiations from the error cell. Hcell errors not common to all the Hcells are stored in the OUTPUT error cell. Use in hierarchical mode only.

ALL

Same as HIER mode except Dracula flattens error-flagged cell instances to the top level and outputs them to error cells. Nothing is output to the Hcell error cell. This form of output is similar to flat Dracula.

Dracula Reference

Description Block Commands

ORIG-ALL

Reports error cells and output cells generated by the layer operations (for example, SIZE and SELECT operations) in the original design hierarchy. Use in hierarchical or flat mode.

ORIG-DRC

Outputs error cells to the original hierarchy. Use in hierarchical or flat mode.

cell-name

Reports error flags from the cell name you specify in that cell only. The cell name you specify in the rules file must be the same as the reported Dracula cell name. Use only in hierarchical or flat mode. You can specify multiple cells.

```
*DESCRIPTION
...
CELL-ERROR-REP = DERR05
CELL-ERROR-REP = DERR10
...
*END

*OPERATION
...
AND POLY DIFF GATE OUTPUT DERR 5
...
EXT POLY LT 6 OUTPUT DERR 10
...
*END
```

Note: The ONCE, HIER, and ALL options report errors in the DRACULA two-level (for example, composite plane or Hcell plane) hierarchy. The ORIG-ALL, ORIG-DRC, and *cell-name* options report errors in the original design hierarchy. With the ORIG-ALL, ORIG-DRC, and *cell-name* options, error segments are reported in the lowest level if the hierarchical cell is nested with other cells.

Note: Runs that include the ORIG-ALL, ORIG-DRC, and *cell-name* arguments take more CPU time because they add postprocessing operations to the end of the run.

Examples

This example instantiates the error flag outputs in the OUTPUT error cell.

```
CELL-ERROR-REP = ONCE
```

This example places the error flag outputs hierarchically.

```
CELL-ERROR-REP = HIER
```

This example displays error flag outputs as in flat Dracula.

```
CELL-ERROR-REP = ALL
```

This example displays error flag outputs in original cell.

Dracula Reference

Description Block Commands

CELL-ERROR-REP = ORIG-DRC

This example reports error data in cell *DERR05*.

CELL-ERROR-REP = DERR05

CELL-LIBRARY

CELL-LIBRARY = YES/NO

Description

Saves a copy of the original database when you size a layout design. This is helpful when changing the technology of a layout.

When you specify CELL-LIBRARY with a SIZE operation, Dracula stores the original database, which preserves the original hierarchy of the cell being sized. Use CELL-LIBRARY with the SIZE operation only. Specify the CELL-LIBRARY command in a hierarchical Dracula rules file.



To store the design, you must specify an OUTDISK command.

Arguments

YES

Stores a copy of the original database.

NO

Default.

CHECK-MODE

CHECK-MODE = FLAT/CELL/HIER/COMP/MULTI

Description

Specifies the mode in which hierarchical Dracula operates. Place this command at the beginning of the rules file. The default is flat mode, which allows all flat Dracula files to run unchanged.

Checking Method

modes

Flat, cell, hierarchical, composite, or multilevel.

checks

All.

Arguments

FLAT

Flattens the layout database as in flat Dracula. Default.

You can specify the output layer (trapezoid format) and the OUTPUT *c-name* the same as when you run flat Dracula.

CELL

Establishes cell mode. Running cell mode is equivalent to running a flat check on each of the cells. Dracula reports discrepancies and errors for each Hcell.

In cell mode HDRC, Dracula automatically selects Hcells according to internal criteria. With all other cell mode checks (HERC, HLVS, and HLPE), you must select the Hcells.

When running cell mode HDRC, send the output from each HDRC check to different layer numbers (*l-num*) to distinguish them in the Hcell error cell.

You can specify the output layer (trapezoid) and the OUTPUT *c-name* in cell mode. The output layer data is kept separately within each cell.

A reserved layer boundary (*boundary.cel*) is always generated in cell mode and contains trapezoidal Hcell box boundaries for each Hcell. Dracula can use these for trimming (AND operation) oversized data that extends outside the boundary of the Hcell.

Dracula Reference

Description Block Commands

HIER

Generates a complete HDRC check of the layout. Dracula chooses Hcells and runs HDRC checks in each Hcell, in the composite, and in the interface from Hcell to Hcell and Hcell to composite plane.

When running hierarchical mode HDRC, send the output from each HDRC check to different layer numbers (*l-num*) to distinguish them in the Hcell error cell.

To speed up the HDRC hierarchical process, specify a maximum environment value as follows;

```
CHECK-MODE = HIER FAST user-defined-max-env-value
```

When you specify this value, PDRACULA creates a data file named ENVLAY.DAT. This file keeps the correct environment values for each LOGICAL operation. You can also specify the variable *setenv FHDRC* in your operation environment to speed up HDRC.

If you specify both the *setenv FHDRC* variable and a maximum environment value in the Description block, the second method has the highest priority. The maximum environment value you specify can be smaller than some of the environment values in ENVLAY.DAT. PDRACULA compares the values in ENVLAY.DAT with the maximum values you define and generates the correct environment value for later use.

When the specified environment value is as follows:

```
CHECK-MODE = HIER OPTIMIZE
```

it will automatically switch from hierarchical mode to flat mode if necessary, and set ENVIRONMENT-MAX to 2.0 microns. It will not skip the violated rules, but still generate the input cards for them and let SPACING do the required optimization.

Dracula automatically switches from hierarchical mode to flat mode if it does not find an Hcell.

COMP

Runs checks on the composite plane, the composite-to-Hcell pins, and the Hcell-to-Hcell pins. For HLVS, you must specify the Hcells for both the schematic subcircuits and layout cell names using the HCELL-FILE command. HLVS composite CHECK-MODE checks for overlapping information and reports it in the *.lvs* file.

Run cell mode before composite mode to be sure Hcells are clean and verified. In some cases, where Hcells are empty and only the pins are provided, you might want to run a composite mode HLVS check without a cell mode check.

In hierarchical and composite modes, if the input layer has not already been flattened, you can specify only the output layer (trapezoid format with two levels of hierarchy). If polygon output is required, the output layer generated by the SIZE command can be flattened and sized by zero with the OUTPUT command. To do the SIZE operation

hierarchically, you must specify the EXCEPTION-ON=[HSP-OUTPUT] command. This results in a two-level output with CELL-ERROR-REP=HIER.

MULTI

Generates a complete HDRC check of the layout in multilevel mode. Dracula chooses Hcells suitable for multilevel hierarchy and creates directories for checking each Hcell and a directory for running flat checks.

During a multilevel HDRC, Dracula checks the selected Hcells in the order they occur in the design hierarchy, from bottom to top. Dracula replaces each checked Hcell with an abstract of the Hcell, where it is placed in an Hcell at the next level of the design hierarchy. Dracula automatically runs large dimension DRC checks and checks with nodal options in the flat directory. The output from each run is kept in its run directory.

When you run multilevel HDRC, Hcells can contain other Hcells. Dracula automatically replaces child Hcells with an abstract of the Hcell. Dracula generates the abstracts using the FRAME BY command.

To determine the frames for the framed Hcells

- Check whether any cell box overlaps framed Hcells. Find the protected area of each framed Hcell that does not have any cell box intrusion.
- While expanding composite data, check if any trapezoid after expansion intrudes the protected area. Redefine the protected area if any intrusion occurs.

INNERBOX.CEL is the protected area reduced by 2 * maximum oversize value + maximum environment size (maximum DRC rule).

For more information about using different run modes, see “[*About Run Modes*](#)” in the “Selecting a Run Mode” chapter in the *Dracula User Guide*.

Examples

```
CHECK-MODE = FLAT      (default)
```

```
CHECK-MODE = CELL
```

```
CHECK-MODE = HIER
```

```
CHECK-MODE = COMP
```

```
CHECK-MODE = MULTI
```

CHECK-PATH

CHECK-PATH = CENTER-LINE/YES/NO

Description

Checks polygons that were created by expanding paths in the EXPAND module.

Arguments

CENTER-LINE

The EXPAND module checks path center-lines for acute or non-45 degree angles.

YES

The EXPAND module checks polygons expanded from paths.

NO

The EXPAND module does not check polygons expanded from paths.
Default.

Example

```
*DESCRIPTION  
CHECK-PATH = CENTER-LINE  
.  
*END
```

CMOS-NTYPE

CMOS-NTYPE = *subtypeN1, subtypeN2, ...*

Description

The LVSCHK or LPECHK commands in the Operation Block of commands will form inverters with N/P MOS pair. However, the inverter will not be formed if the names of the MOS subtypes are not one-character long; that is, if the MOS is not N/P pair.

In Dracula 4.81 and subsequent versions, this CMOS-NTYPE command in the description block of the rule file lets you form inverters with 2-character subtypes.

Note: The default CMOS types are N/P and this command is case sensitive.

Arguments

subtypeN?

Two-character NMOS subtypes

Note: The maximum types of MOS subtypes in each command is 20. The subtypes need not be specified if they are N/P.

Example

CMOS-NTYPE = NC, NN, NE

In this example, it will form inverters if the subtype of NMOS is NC, NN, or NE.

CMOS-PTYPE

CMOS-PTYPE = *subtypeN1, subtypeN2, ...*

Description

The LVSCHK/LPECHK commands in the Operation Block of commands will form inverters with N/P MOS pair. However, the inverter will not be formed if the names of the MOS subtypes are not one-character long; that is, if the MOS is not N/P pair.

In Dracula 4.81 and subsequent versions, this CMOS-PTYPE command in the description block of the rule file lets you form inverters with 2-character subtypes.

Note: The default CMOS types are N/P and this command is case sensitive.

Arguments

subtypeP?

Two-character PMOS subtypes

Note: The maximum types of MOS subtypes in each command is 20. The subtypes need not be specified if they are N/P.

Example

CMOS-PTYPE = PC, PP, PE

In this example, it will form inverters if the subtype of PMOS is PC, PP or PE.

CNAMES-CHK-OFF

CNAMES-CHK-OFF = YES/NO

Description

Overrides default CNAMES-CSEN=NO for GDSII input system

Arguments

YES

Overrides default CNAMES-CSEN=NO

NO

Default

Example

CNAMES-CHK-OFF = YES

CNAMES-CSEN

CNAMES-CSEN = YES/NO

Description

Specifies cell name and text name case sensitivity for all commands that contain cell or text names. These include, but are not limited to PRIMARY, DELCEL, ARRAY, OUTLIB, POWER-NODE, GROUND-NODE, PULL-UP, PULL-DOWN, PROBE, SELECT LABEL, LCONNECT, ECONNECT, and HCELL.

This command affects all cell and text names you specify in the layout and in EDTEXT or HEDTEXT files.

To specify only text name case sensitivity, and not cell name, you must specify the TNames-CSEN command.

Arguments

YES

Passes the exact name and character cases from the commands and text files to the *jxrun.com* file. Default when you define the SYSTEM command as CADENCE, or OPUS.

NO

Converts all cell and text names to uppercase.

If CNAMES-CSEN = YES and TNames-CSEN = NO, Dracula passes the exact name and character cases from the commands to the *jxrun.com* file and converts the text names to uppercase.

If CNAMES-CSEN = YES and TNames-CSEN = YES, Dracula passes the exact name and character cases from the commands and text file to the *jxrun.com* file.

If CNAMES-CSEN = NO and TNames-CSEN = YES, Dracula converts the cell names to uppercase and passes the exact text names to the *jxrun.com* file.

If CNAMES-CSEN = NO and TNames-CSEN = NO, Dracula converts all cell and text names to uppercase.

Note: The default for the CNAMES-CSEN command depends on the SYSTEM command you specify. If you specify the SYSTEM command as CADENCE the default for CNAMES-CSEN is YES. You cannot override the default when you use this system. The default is NO

Dracula Reference

Description Block Commands

for all other SYSTEM commands, such as GDSII. You can override the default when you use any of these systems.

Example

```
CNAMES-CSEN = YES  
POWER-NODE= VDD  
GROUND-NODE = VSS  
OUTLIB = drcout
```

CONN-ALL-WIRE

CONN-ALL-WIRE = YES/NO

Description

Forces node connections among all cell texts with the same text name. In composite mode, this command implies CONN-INTER-WIRE and forces all cell texts with the same name to be connected without looking at the wire numbers, if any.

Checking Method

modes

Composite and cell.

checks

HERC, HLVS, and HLPE.

Arguments

YES

Forces node connections.

NO

No node connections. Default.

Example

CONN-ALL-WIRE = YES

CONN-INTER-WIRE

CONN-INTER-WIRE = YES/NO

Description

Forces node connections among internally wired cell text, even though they are not physically connected. Usually, Dracula checks the node connections among those internally wired cell texts with the same virtual number and reports the warnings only if the nodes are not physically connected. The CONN-INTER-WIRE command suppresses the connection check and forces the connection.

CONN-INTER-WIRE has the same effect on composite nodes connected to internally virtual wired cell pins as the FRAME BY command. The CONN-INTER-WIRE = YES command covers the internal virtual wires in all Hcells, while the FRAME BY command covers only the specified Hcells. In addition, the FRAME BY command takes out all geometries inside the frame, but the CONN-INTER-WIRE command does not.

Checking Method

modes

Composite.

checks

HERC, HLVS, and HLPE.

Arguments

YES

Forces node connections.

NO

No node connections. Default.

Example

CONN-INTER-WIRE = YES

CONN-OPEN-WIRE

CONN-OPEN-WIRE = YES/NO

Description

Forces a connection between composite texts with identical names. The CONN-ALL-WIRE command forces a connection only between identical cell texts. This command works in flat and composite mode.

Arguments

YES

Forces a connection between identical composite texts.

NO

Default.

Example

CONN-OPEN-WIRE = YES

CONVERT-DATABASE

CONVERT-DATABASE = YES/NO

Description

Changes a database from one format to another. Use this command if the SYSTEM and SYSOUT command formats are different. Only the formats change, not the database units. A micron or mil database remains micron or mil after the conversion. The converted database is flat.

You do not need an Operation block when you specify the CONVERT-DATABASE command.

For an example of comparing layers from different database systems, refer to the [Database Comparison Example](#). If you do a database comparison or logical operation, the SCALE and RESOLUTION commands must be identical in the Description blocks of both input rules files.

Database Comparison Example

The following rules file compares the digitized layers of one database to layers of another database. Output is in a graphic format.

```
compare1.com compare2.com
*DESCRIPTION          *DESCRIPTION
INDISK=lay1.db        INDISK=lay2.db
PRIMARY=top1          PRIMARY=top2
.                     PRINTFILE=comprt
.                     OUTDISK=comout
.                     .
*END                  *END
*INPUT-LAYER          *INPUT-LAYER
meta=7                metb=7
EXPORT meta           IMPORT-LAYERS meta
.                     .
*END                  *END
                     *OPERATION
                     NOT metb meta err1 OUTPUT err1 50
                     NOT meta metb err2 OUTPUT err2 50
                     .
                     ;or
                     .
                     XOR meta metb err1 OUTPUT err1 50
                     .
                     *END
```

To submit the comparison job, enter the following:

```
% PDRACULA
                :/GET compare1.com n
                :/NEXT
```

Dracula Reference

Description Block Commands

```
:/GET compare2.com n  
:/FIN
```

This database comparison first runs the *compare1.com* job to read in the first database layers. Then it runs the *compare2.com* job to read in the second database layers and do the graphic comparison. The summary report file is *comprt.sum*, and the graphic output file is *comout.dat*.

CPOINT-FILE

CPOINT-FILE = *filename*

Description

Inputs a file containing initial correspondence points for LVS matching. You can use this command with LVS and LPE.

The correspondence points specified in the CPOINT-FILE are in addition to those specified as text in the layout and input/output/global signals in the schematic. In cases of conflicts or ambiguities, the CPOINT-FILE overrides the text and input/output/global signals.

Specify one correspondence pair per line in the CPOINT-FILE. Each line can be one of the following formats:

- Layout-node-name Schematic-node-name {A}
- Layout-node-name Schematic-node-number
- Layout-node-number Schematic-node-name {A}
- Layout-node-number Schematic-node-number

The character *A* as the third field in a line is optional if you use a pure number for a schematic node name. In cases where the input schematic netlist originates from conventional SPICE format, the node name is always in a pure number. The *A* is required to distinguish these numerical node names from the node numbers generated by Dracula.

Pads and pins listed by CPOINT-FILE are not reported as unmatched, and Dracula does not issue a warning message.

Arguments

filename

Specifies the name of the file in which you specify the initial correspondence points. The maximum number of characters for the file name has been increased from 128 to 10000.

Example

The following example tells CPOINT-FILE to read IPOINT.TXT for additional or replacement initial correspondence points for LVS matching:

```
CPOINT-FILE = IPOINT.TXT
```

An example of the contents of *ipoint.txt* is as follows:

Dracula Reference

Description Block Commands

DUMMY1	X505-X2202-X1234-O887
27	X723-X32-IN1
DUMMY2	67 A
DUMMY3	299

CPOINT-FILE matches the following:

- Text layout node DUMMY1 to an expanded schematic node X505-X2202-X1234-O887
- Layout node with Dracula number 27 to expanded schematic node X723-X32-IN1
- Text layout node DUMMY2 to schematic node with name 67
- Text layout node DUMMY3 to schematic node with the internal Dracula node number 299

LOGLVS and NODAL assign Dracula-generated node numbers to the schematic and layout. For schematics, the IMAGE.LIS file contains a table of node numbers that correspond to expanded node names. For expanded node names, CPOINT-FILE uses the minus sign (-) delimiter to separate names originating from different levels of the hierarchy.

For the layout, the 6PADL.DAT file contains the node numbers that correspond to node names for all of the nodes assigned text. To find node numbers without text for any polygon or trapezoid, use Dracula Interactive.

CSH-F-OPTION

CSH-F-OPTION = YES/NO

Description

Addition of *-f* option to *jxrun.com* prevents job from resourcing *.cshrc*.

Arguments

YES

Default for DD job.

If non-DD job, add *-f* option with *cs*h in *jxrun.com*. The first line in *jxrun.com* is "*#!/bin/csh -f*". With "*-f*" option, neither the *.cshrc* file nor the *.login* file (a login shell) is sourced again when running *jxrun.com*.

NO

Default for non-DD job.

If DD job, take away *-f* option with *cs*h in *jxrun.com*. The first line in *jxrun.com* would be "*#!/bin/csh*". Either of *.cshrc* or *.login* will be sourced again when a DD job is submitted.

Example

CSH-F-OPTION = YES

DATAFORMAT

DATAFORMAT 4.2/ZLIB { *LEVEL* }

Description

Turns on the database compression algorithm. Dracula can take D3, 4.2, and ZLIB input files for the same module. D3 is the default.

Note: You must also specify this command in the Description block for LVS and non-LVS jobs.

Arguments

4.2

Database compression algorithm introduced in Dracula 4.2.

ZLIB

Zlib database compression algorithm. It has about 2 times better data compression ratio compared to 4.2 and up to 6 times better compared to the default compression method, depending on the testcase.

LEVEL

Zlib compression level. Set this value between 1 and 9. 9.9 is the maximum compression ratio and has less performance. 1 is the minimum compression ratio and has more performance. The default value is 6.

Note: The LOGLVS data compression algorithm is defined for LOGLVS output only and has no effect on jxrun.com run.

Example

DATAFORMAT = ZLIB

DD-RSH-COMMAND

DD-RSH-COMMAND = `command`

Description

The DD-RSH-COMMAND command allows you to specify a command to execute jobs on a remote host. This command requires the PARALLEL-FILE command.

Arguments

command

Specifies user-defined command.

Example

DD-RSH-COMMAND = `qsub -x`

DELCEL

```
DELCEL = cell-name1 cell-name2 cell-name3
```

```
DELCEL = cellname?
```

Description

Removes a cell from processing, conserving system resources. You can specify up to 1,024 cells using the `DELCEL` command. You can add a wild card character, that is `?`, at the end of a cell name to refer to a group of cell names. This deletes all the cells beginning with that cell name (including that first cell). You can specify up to 32 cell names containing a wild card.

Arguments

cell-name

Name of the cell to remove from processing

cell-name?

Names beginning with that cell-name (including that first cell) to remove from processing. The wild card character, `?` must appear at the end of the argument. Maximum length of the argument is 32. Maximum number of `DELCEL` commands containing the wild card is 32.

Example

```
DELCEL = iomux
```

or

```
DELCEL = iomux?
```

DEL-VIR-WIRE

DEL-VIR-WIRE = YES/NO

Description

Specifies whether to delete virtual wires from processing. If text input from the layout or EDTEXT file contains the virtual wire connection designator, a trailing colon (:), the virtual wire connection designator is ignored and the DEL-VIR-WIRE command processes all text without virtual connections.

This command is useful during the final stages of IC verification when you expect all nodes throughout the chip to connect as a single node. With this command, you do not need to generate a new text-layer containing text without the virtual wire connection designator.

Arguments

YES

 Ignores virtual wires.

NO

 Default.

Example

DEL-VIR-WIRE = YES

DIODE-P-TO-G

DIODE-P-TO-G = YES/NO

Description

Extracts the diodes connected to power and ground.

Arguments

YES

Extracts diodes connected to power and ground.

NO

Ignores the diodes connected to power and ground.

Example

```
*DESCRIPTION
DIODE-P-TO-G = YES
*END
*OPERATION
PARASITIC      DIO[N]    NSD      PSUB    NSD
PARASITIC      DIO[P]    PSD      PSD     NWELL
PARASITIC      DIO[PD]   NWELL    PSUB    NWELL
LPESELECT[S]   DIO        OUTPUT  SPICE

*END
```

DIODESEQ

`DIODESEQ = A1, P1, {A2}, {P2}, {A3}, {P3}`

Description

Specifies the listing order of LPE diode parameters. You can extract six types of diode parameters: *A1*, *P1*, *A2*, *P2*, *A3*, *P3*. You must specify a correct parameter listing order for your circuit simulator. If you do not specify DIODESEQ, LPE lists only *A1* and *P1*.

Arguments

A1

Specifies the area of diffusion layer (1st layer in PARASITIC DIO) not overlapped by another layer (2nd, 3rd layer in same PARASITIC DIO command).

P1

Specifies the perimeter of diffusion layer not overlapped by other layers.

A2

Specifies the area of diffusion layer overlapped by 2nd layer (in PARASITIC DIO).

P2

Specifies the perimeter of diffusion layer overlapped by 2nd layer.

A3

Specifies the area of diffusion layer overlapped by 3rd layer (in PARASITIC DIO).

P3

Specifies the perimeter of diffusion layer overlapped by 3rd layer.

Examples

`DIODESEQ = A1    P1    A2    P2    A3    P3`

`DIODESEQ = A1    P1    A2    A3`

`DIODESEQ = A1    P1`

DRACBATCH-EMAIL

DRACBATCH-EMAIL = YES/NO

Description

Turns automatic e-mail notification on or off when running `dracq`.

Arguments

YES

Sends an automatic mail message to the submitter when the queued job is finished.

NO

Does not send a mail message when the queued job is finished. NO is the default.

EBOX-SHORT-PIN

EBOX-SHORT-PIN = YES/NO

Description

Restricts the number of pins in element box from going over 4.

Example

```
*DESCRIPTION
```

```
....
```

```
EBOX-SHORT-PIN = YES
```

```
....
```

```
*END
```

```
....
```

```
*OPERATION
```

```
....
```

```
ELEMENT  ILA[LA]  IILA  SPI  ACTSN  BN1  INJ
```

```
ELEMENT  ILB[LB]  IILB  SPI  ACTSN  BN1  INJ
```

```
ELEMENT  ILC[LC]  IILC  SPI  ACTSN  BN1  INJ
```

```
ELEMENT  ILD[LD]  IILD  SPI  ACTSN  BN1  INJ
```

```
.....
```

```
*END
```

The element ILA,ILB,ILC,ILD are element boxes.

EMPTY-BLACKBOX

EMPTY-BLACKBOX = YES/NO

Description

Specifies not to read in blackbox Hcell data. This command works with the BLACKBOX-FILE command. When you specify EMPTY-BLACKBOX =YES, the blackbox Hcells contain only those pin geometries from BLACKBOX-FILE. The original data is dropped.

Arguments

YES

Specifies that blackbox Hcells contain only pin geometries from BLACKBOX-FILE.

NO

GDS2IN generates a rectangle for each pin in the BLACKBOX-FILE and reads in blackbox Hcell data. Default.

Example

In the following example, the blackbox cell *inv* contains only one pin geometry with w=0.6 h=1. The original data in the *inv* cell is dropped.

```
BLACKBOX-FILE = blackBox.fil  
EMPTY-BLACKBOX = YES
```

blackbox.fil:

```
IN      X=41      Y=90      ATTACH=poly inv w=0.6 h=1
```

EMPTY-ENC

EMPTY-ENC = YES/NO

Description

The EMPTY-ENC command causes Dracula to generate empty .ENC files when YES is specified. This mode speeds layer processing for HLVS, HLPE and HPRE composite mode runs. EMPTY-ENC=YES does not ensure reporting correct result of XDEVICE. If XDEVICE command is used in the rule file, EMPTY-ENC will be forced to be NO and PDRACULA will report the warning message if EMPTY-ENC=YES is given in the rule file.

Arguments

YES

Generates empty .ENC files in the data preprocessing stage (GENENV) for non-base layers.

NO

Default.

Example

EMPTY-ENC = YES

END-MACRO

END-MACRO = *macro-name*/"*macro-name arg1 arg2...*"

Description

Executes a macro with the given name after all other commands execute. You can specify only one END-MACRO command. Follow the END-MACRO command with an END statement.

Arguments

macro-name

Specifies the file name of the macro to execute after all other commands. Maximum file name is 128 characters.

arg1 arg2...

Specifies the macro arguments. Enclose the macro and its arguments in quotation marks (" ").

Examples

END-MACRO = *clean.com*

Compiles to *cs**h clean.com*, where the *clean.com* macro file contains the following:

```
cp lvsprt.* /cadence/user
cp lvsout.dat /cadence/user
rm *.dat
```

END-MACRO = "abc b1 b2 b3 b4"

Compiles to *cs**h abc b1 b2 b3 b4*, where *abc* is the macro and *b1-b4* are its arguments.

ENVIRONMENT-MAX

`ENVIRONMENT-MAX = unit-size units`

Description

Specifies an upper limit for the value used to establish the width of the Hcell environment. If the preprocessor finds spacing checks larger than the value specified in this command, it indicates the error so you can correct the problem and recheck the rules file. If you use `CHECK-MODE = HIER OPTIMIZE` with `ENVIRONMENT-MAX` then PDRACULA will not skip the violated rules, but still generate the input cards for them and let SPACING do the required optimization.

You can run HDRC spacing checks greater than the value specified in the `ENVIRONMENT-MAX` command only after Dracula flattens the input layers involved in the checks.

If the dimensional check for hierarchical layers is very large, many composite geometries must be kept and processed in checking Hcell-to-composite and Hcell-to-Hcell relationships. This bookkeeping slows down the checking speed and requires more disk space to store the information. For this reason, keep dimensional checks under 10 microns. If there are only a few dimensional checks whose DRC distances are much greater than the rest, flatten the relevant layers and flat checks for these large dimensional checks.

Note: Assigning a value greater than 15 microns to `ENVIRONMENT-MAX` might cause excessive computation time. The preprocessor warns you if the `ENVIRONMENT-MAX` value is greater than 15 microns.

Checking Method

modes

Hierarchical.

checks

HDRC.

Arguments

`unit-size`

Specifies the maximum dimension for spacing checks on hierarchical layers. Affects the following HDRC spacing checking commands:

`ENC, EXT, INT, WIDTH, LENGTH`

Dracula Reference

Description Block Commands

Note: This command does not restrict dimensions for commands such as PLENGTH and SIZE.

units

Specifies the natural database units in microns or mils.

Examples

```
ENVIRONMENT-MAX = 10 microns  
ENVIRONMENT-MAX = .25 mils
```

ERROR-PATH-WIDTH

`ERROR-PATH-WIDTH = value-size units`

Description

Specifies the width of error flags produced by the Design Rules Checker (DRC). The applicable DRC commands are ENCLOSURE, EXTERNAL, INTERNAL, WIDTH, and LENGTH.

Arguments

value-size

Specifies the DRC error-flag width. Default is 0.25 microns.

units

Specifies the units of *value-size*. Must match units in SCALE and RESOLUTION (microns or mils).

Examples

`ERROR-PATH-WIDTH =.25 micron (default)`

`ERROR-PATH-WIDTH = 1 micron`

`ERROR-PATH-WIDTH = .04 mil`

EXPAND-GATE-DELI

`EXPAND-GATE-DELI = string`

Description

Controls the delimiter for parallel gates in the SPICE output from LPE/PRE. This command lets you select the delimiter to make the gate name unique in the flat output netlist produced by Dracula.

LVS uses “-” as the delimiter when reporting discrepancies in the `.vs` file. This command can control the delimiter in LPE/PRE.

Arguments

string

One to four characters. The string cannot contain a space, a colon (:), a semicolon (;), or a question mark (?). If you specify a string longer than four characters, Dracula reports an error.

EXPAND-INST-DELI

EXPAND-INST-DELI = *string*

Description

Controls the delimiter for instances in the SPICE output from LPE/PRE. If you do not specify EXPAND-INST-DELI, Dracula uses the delimiter you specify in the EXPAND-NODE-DELI command for the instances.

LVS uses “-” as the delimiter when reporting discrepancies in the *.lvs* file. This command can control the delimiter in LPE/PRE.

EXPAND-INST-DELI and EXPAND-NODE-DELI work together in the following ways:

- If you specify EXPAND-INST-DELI = *deli1* and EXPAND-NODE-DELI = *deli2*, Dracula uses *deli1* for the instance name delimiter and *deli2* for the net name delimiter.
- If you specify EXPAND-INST-DELI = *deli1*, but do not specify EXPAND-NODE-DELI, Dracula uses *deli1* as the instance name delimiter and uses “-” as the net name delimiter.
- If you do not specify EXPAND-INST-DELI or EXPAND-NODE-DELI, Dracula uses “-” for both the instance and net name delimiter.
- If you specify EXPAND-NODE-DELI = *deli2*, but do not specify EXPAND-INST-DELI, Dracula uses *deli2* for both the instance and net name delimiter.

Arguments

string

One to four characters. The string cannot contain a space, a colon (:), a semicolon (;), or a question mark (?). If you specify a string longer than four characters, Dracula reports an error

EXPAND-NODE-DELI

EXPAND-NODE-DELI = *string*

Description

Controls the delimiter for both node names and element names in the SPICE output from LPE/PRE. This command lets you select the delimiter to concatenate with the hierarchy instance name in order to make the node name unique in the flat output netlist produced by Dracula.

LVS uses “-” as the delimiter when reporting discrepancies in the *.lvs* file. This command can control the delimiter in LPE/PRE.

EXPAND-NODE-DELI and EXPAND-INST-DELI work together in the following ways:

- If you specify EXPAND-INST-DELI = *deli1* and EXPAND-NODE-DELI = *deli2*, Dracula uses *deli1* for the instance name delimiter and *deli2* for the net name delimiter.
- If you specify EXPAND-INST-DELI = *deli1*, but do not specify EXPAND-NODE-DELI, Dracula uses *deli1* as the instance name delimiter and uses “-” as the net name delimiter.
- If you do not specify EXPAND-NODE-DELI or EXPAND-INST-DELI, Dracula uses “-” for both the instance and net name delimiter.
- If you specify EXPAND-NODE-DELI = *deli2*, but do not specify EXPAND-INST-DELI, Dracula uses *deli2* for both the instance and net name delimiter.

Arguments

string

One to four characters. The string cannot contain a space, a colon (:), a semicolon (;), or a question mark (?). If you specify a string longer than four characters, Dracula reports an error.

EXTERM-NOWARN

EXTERM-NOWARN = YES/NO

Description

When this command is set to YES, the EXTERM module does not list “unconnected pins in cell” warning messages.

Arguments

YES

EXTERM module does not report “unconnected pins in cell” warning messages.

NO

EXTERM module generates “unconnected pins in cell” warning messages. Default.

Example

EXTERM-NOWARN = YES

FASTSIZE

FASTSIZE= YES/NO

Description

Speeds up size performance.

Arguments

YES

Performs fast sizing. Default.

NO

Does not perform fast sizing.

Example

```
*DESCRIPTION  
FASTSIZE = YES  
..  
*END
```

FDELIMITER

`FDELIMITER = string`

Description

Specifies delimiter characters for feedthrough pin names on the layout text. This lets you specify separate feedthrough pin names on the layout and prevents Dracula from shorting all feedthrough pins. Use for flat mode only. You must specify all feedthrough pin names in the CDL netlist file at the .SUBCKT statement. Specify the FDELIMITER command before the input netlist. LOGLVS generates a SCHFPIN.DAT file containing the feedthrough pin information.

When running PRE, the feedthrough pin text must be on the pad layer.

Arguments

`string`

One to four characters. The string cannot contain a space, a colon (:), a semicolon (;), or a question mark (?). If you specify a string longer than four characters, Dracula reports an error.

Example

`FDELIMITER = /`

Specifies the slash (/) as the delimiter character for feedthrough pin names.

FILTER Commands

`FILTER-LAY-OPT = options...`

`FILTER-SCH-OPT = options...`

`FILTER-OPT = options...`

`FILTER-MOSCAP = FLOAT/ALL`

`REMOVE-SCH-OPT=YES/NO`

Description

The LVCHK[FIG] and the LPECHK[FIG] commands initiate an internally set filtering process. These filter options are not appropriate for all technologies. The following commands let you choose different filter options by overriding the LVCHK or LPECHK filter order. You must still specify LVCHK[FIG] or LPECHK[FIG] to activate the filtering process. If you turn on LVCHK[F I G], 3-terminal floating MOS is always filtered no matter what options you select, .

FILTER-LAY-OPT sets the filtering options for layout devices and overrides internally set criteria.

FILTER-SCH-OPT sets the filtering options on the schematic and overrides the default filtering options. YES removes the FILTER stage for the schematic side. The default is NO.

FILTER-OPT sets the same options on both the layout and schematic. Do not use FILTER-OPT with FILTER-LAY-OPT or FILTER-SCH-OPT.

FILTER-MOSCAP reduces the number of filtered devices for options *H* and *I*. FILTER-MOSCAP is activated only when you specify option *H* or *I*. If you specify FILTER-MOSCAP = FLOAT, only MOSCAP that contains a floating gate is filtered. The default is FILTER-MOSCAP = ALL, which means all MOSCAP filtered by options *H* and *I* is filtered even if it is not connected.

REMOVE-SCH-OPT=YES/NO removes FILTER stage from the schematic side after SCHMIN. If you specify YES, Dracula removes FILTER stage from the schematic side and FILTER is not performed. If you specify NO, Dracula will perform the FILTER stage on the schematic side. Default is NO.

Arguments

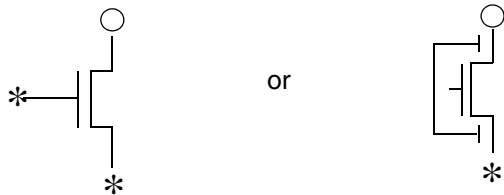
The following are the allowed *options* arguments.

Dracula Reference

Description Block Commands

B

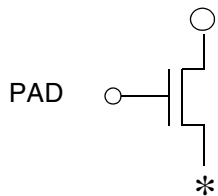
Filters out the MOS device if the gate is floating or does not connect to any pad through a source/drain path and either the source or drain net is floating.



An asterisk (*) denotes a floating node. An open circle (o) denotes a node that is not significant to the filtering criteria.

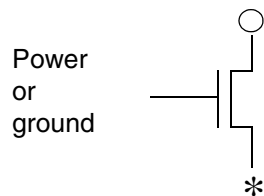
BA

Filters MOS devices if the gate connects to a pad and either the source or drain net is floating.



C

Filters out the MOS device if the gate connects to power or ground and either the source or drain is floating.

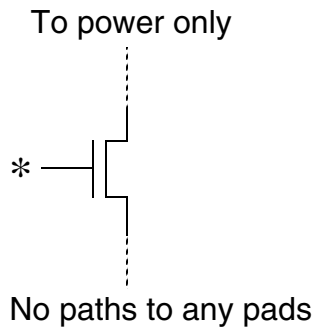


Dracula Reference

Description Block Commands

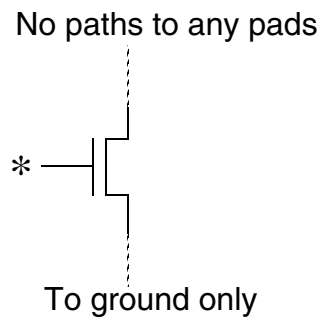
D

Filters out the MOS device if the gate is floating and the source/drain nets have only a power path and no paths to any other pads.



E

Filters out the MOS device if the gate is floating and the source/drain nets have only a ground path and no paths to any other pads.

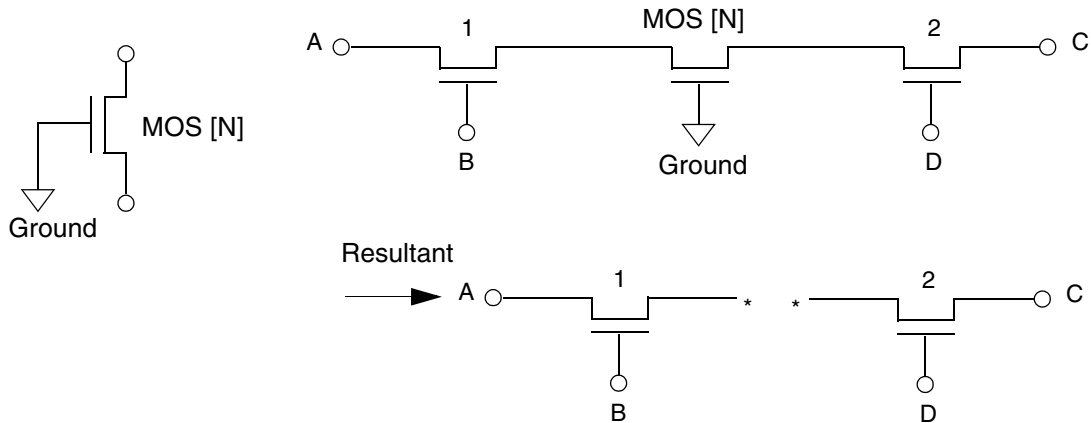


Dracula Reference

Description Block Commands

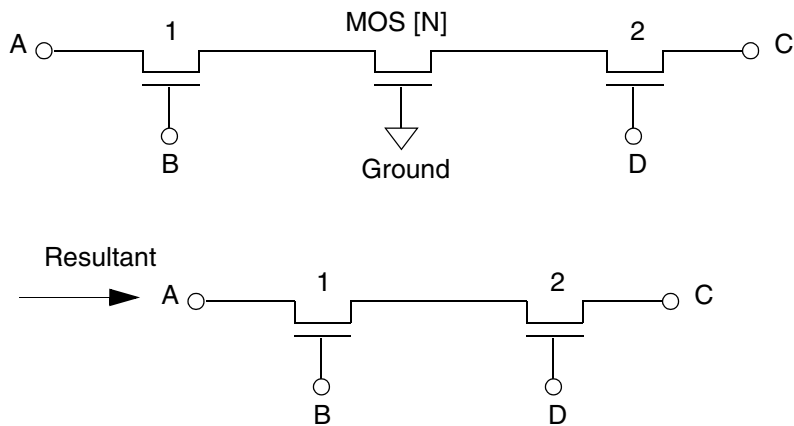
F

Filters MOS[N] devices that have the gate tied to a ground.



FY

Filters MOS[N] devices that have the gate connected to ground, then reconnects the source and drain terminals after the device has been filtered out. If the source and drain connect to two different pads, then no reconnect occurs and a warning message appears.



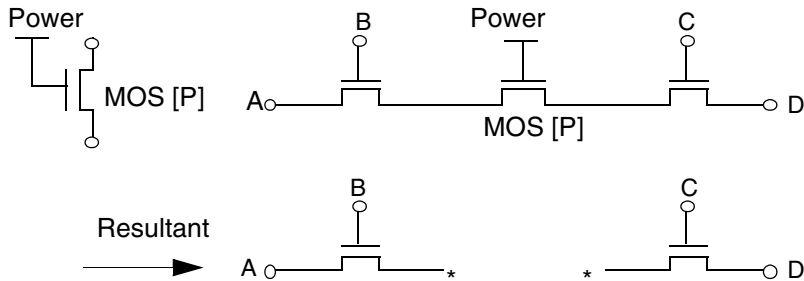
Be careful when using the FY option. It can cause more discrepancies than it reduces.

Dracula Reference

Description Block Commands

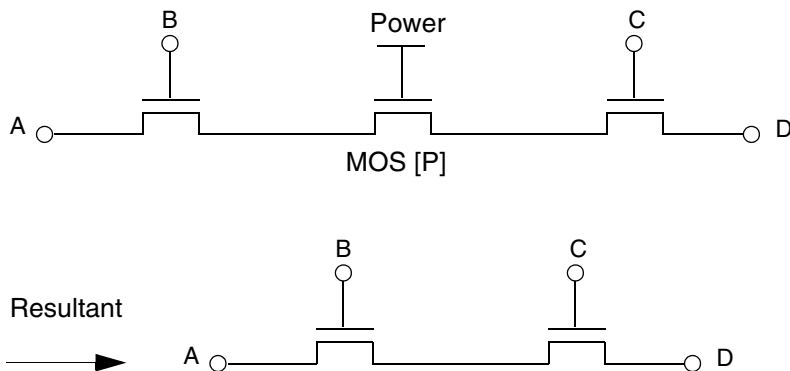
G

Filters MOS[P] devices with the gate tied to power.



GY

Filters MOS[P] devices that connect the gate to power, then reconnects the source and drain terminals after the device has been filtered out.



If the source and drain connect to two different pads, then no reconnect occurs and a warning message appears.



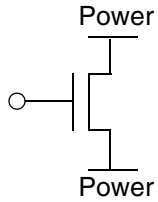
Be careful when using the GY option. It may cause more discrepancies than it reduces.

Dracula Reference

Description Block Commands

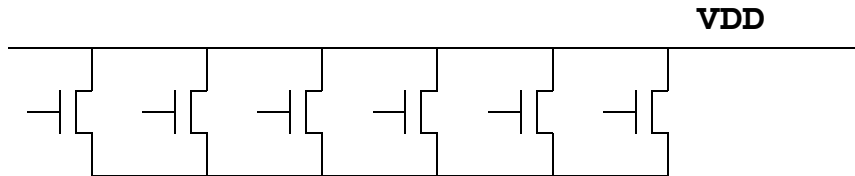
H

Filters series MOS devices when both the source and drain have a path to power.



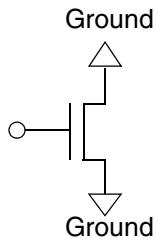
H'

The following structure is filtered out by H option but is kept by H' option.



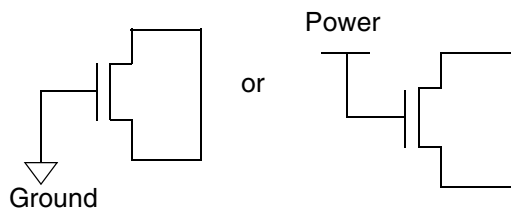
I

Filters series MOS devices when both the source and drain have a path to ground.



J

Filters MOS devices with the gate tied to either power or ground and source and drain tied together.



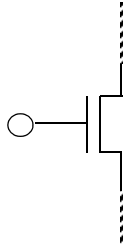
Dracula Reference

Description Block Commands

K

Filters MOS devices with source and drain having no paths to any pad.

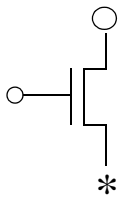
No paths to any pads



No paths to any pads

L

Filters MOS devices if either the source or drain is floating.



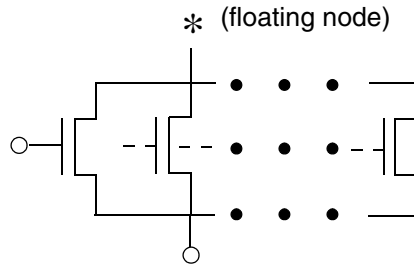
The L option can filter out the MOS device next to a previously filtered-out device and can cause more discrepancies. When you use this option, you should also enable the O option to allow for repeated application until there is no further progress.

Note: Two or more parallel MOSes with a floating source/drain cannot be filtered because Dracula recognizes that there is no longer a floating source/drain since they are connected (see the illustration below). To resolve this problem, use the option RERUN-

Dracula Reference

Description Block Commands

FILTER=YES.



M

Filters MOS[P] devices if the gate connects to ground, then reconnects the source and drain terminals after Dracula filters out the device. You can specify an asterisk (*) after the *P* to filter out MOS devices with any character following the *P* (MOS[P*]).



If the source and drain connect to two different pads, then no reconnect occurs and a warning message appears.

N

Filters MOS[N] devices if the gate connects to power, then reconnects the source and drain terminals after Dracula filters out the device. You can specify an asterisk (*) after the *N* to filter out MOS devices with any character following the *N* (MOS[N*]).



If the source and drain connect to two different pads, then no reconnect occurs and a warning message appears.

O

Repeats the FILTER process until there are no devices to be filtered under the options specified.

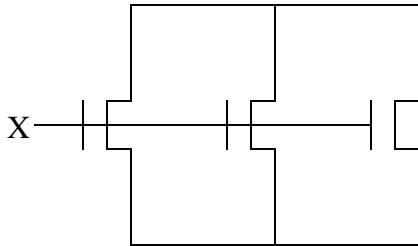
P

Filters out devices with no general paths to any pads. Note that general path is used here to distinguish it from the source-drain paths of MOS devices. A general path can go through any terminal of a device. A source-drain path can only go through the source and

Dracula Reference

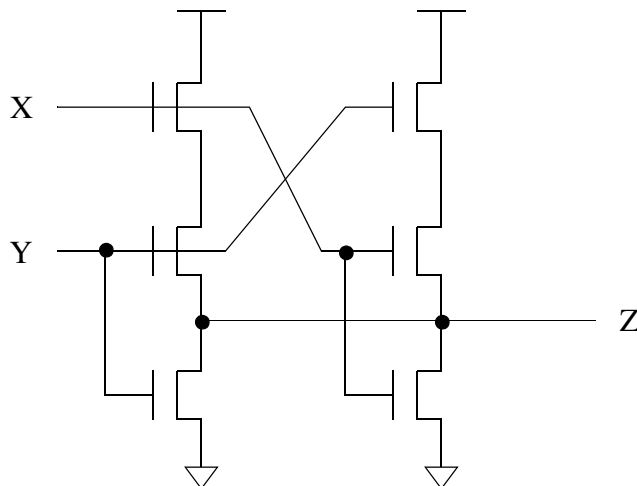
Description Block Commands

drain terminals of a MOS device. The *P* option filters all groups (islands) of devices that are isolated from all pads.



Q

Filters out devices with no general paths to any nonpower/ground pads. (See option *P* for the definition of a general path.) This option filters all groups (islands) of devices that are isolated from all pads, except possibly power/ground pads. By definition, all devices filtered by *P* option are filtered by *Q* option. You cannot use the two options at the same time.

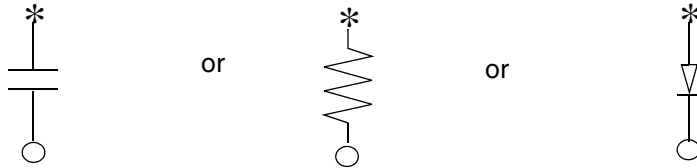


Dracula Reference

Description Block Commands

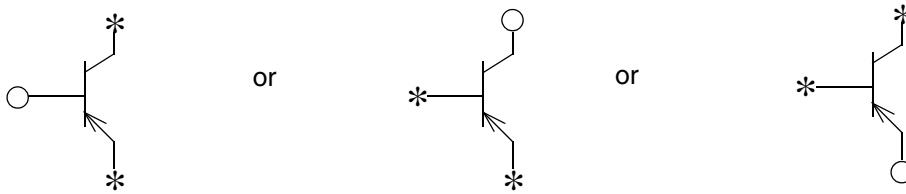
R

Filters out capacitors, resistors, and diodes with at least one floating terminal.



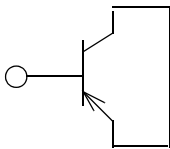
S

Filters out BJT devices with at least two floating terminals.



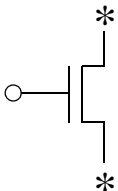
T

Filters out BJT with emitter and collector that connect together and do not connect to other devices.



U

Filters out MOS devices if both the source and drain are floating.



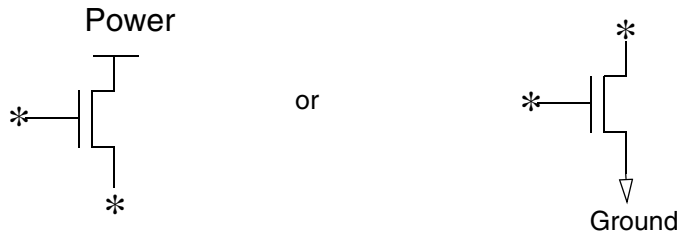
Dracula Reference

Description Block Commands

V

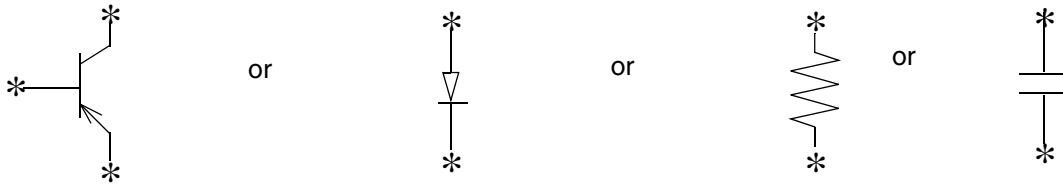
Filters out MOS devices if the following is found:

The gate is floating, and the source or drain is floating, and the non-floating terminal is connected to power or ground.



Z

Filters out floating bipolar transistors, diodes, resistors, and capacitors.



Note: Using the above options, Dracula filters the device in the following order: F, G, B, C, D, E, H, I, J, K, L, M, N, and O.

Example 1

LVSCHK[F] initiates a preset filtering process that is equivalent to the following command sequence.

```
*DESCRIPTION
FILTER-LAY-OPT = BCDEFGHIJKO
FILTER-SCH-OPT = FGHIJKO
*END
```

LVSCHK[G] initiates a preset filtering process that is equivalent to the following command sequence.

```
*DESCRIPTION
FILTER-OPT = BCDEFGHIJKO
*END
```


Dracula Reference

Description Block Commands

Example 2

To set different filter operations, use the FILTER options commands. Remember that you must use LVCHK or LPECHK [F] or [G] to activate the filtering process.

```
*DESCRIPTION
FILTER-LAY-OPT = DEFGHI
FILTER-SCH-OPT = FGHI
*END
*OPERATION
LVCHK[F]
*END
```

FILTER-LDD

`FILTER-LDD = YES/NO`

Description

This command treats the LDD device as a regular MOS device, so all the FILTER options which apply to MOS will apply also to LDD.

Arguments

`YES`

Turn on this function.

`NO`

Turn off this function. `NO` is the default.

Example

`FILTER-LDD = YES`

Dracula Reference

Description Block Commands

FILTER-REPORT

`FILTER-REPORT = ALL/{number}`

Description

The `FILTER-REPORT` command in Dracula generates the information of filtered devices on the layout side in `FILTERL.LIS` and on the schematic side in `FILTERS.LIS`.

Arguments

`ALL`

Report all filtered devices for both layout side and schematic side to `FILTERL.LIS` and `FILTERS.LIS`.

`{number}`

Report the first `{number}` of filtered devices for both the layout side and schematic side to `FILTERL.LIS` and `FILTERS.LIS`.

Example

`FILTER-REPORT = ALL`

`FILTERL.LIS`

DevTyp	DevNum	X	Y	FilOpt	Terminals
MOS	25	19.60	245.55	G	1 17 246
MOS	26	25.50	245.55	G	1 17 246
MOS	27	41.80	245.55	G	1 18 246
MOS	28	47.70	245.55	G	1 18 246
MOS	29	64.00	245.55	G	1 19 246

and so on.

`FILTERS.LIS`

DevTyp	DevNum	----	ElmNam	FilOpt	Terminals
MOS	3	----	MXN72XI71XI3	F	6 6 14
MOS	9	----	MXN82XI71XI3	G	3 15 3
MOS	39	----	MXP131XI658X	G	2 37 2
MOS	46	----	MXP22XI1	G	2 10 17
MOS	47	----	MXP78XI1	G	2 2 21

and so on.

FIX-INPUT-ORDER

FIX-INPUT-ORDER = INSTANCE/EXCEPT-LOGIC/YES/NO

Description

Controls the swapping of logic gates and series/parallel structures. For information about including this information in the netlist file, refer to the *.NONSWAP section in Chapter 5.

The FIX-INPUT-ORDER command might affect backannotation. Although a resistor for an internal node might be assigned a different schematic name, the number of parasitic resistors and their relative locations remain the same.

Note: When you specify FIX-INPUT-ORDER with INSTANCE, EXCEPT-LOGIC, or YES arguments, CARE-SPLIT-ORDER is automatically set to YES.

Arguments

INSTANCE

Instance swapping is controlled by specifying individual or subtype non-swappable MOS devices in the circuit netlist files.

EXCEPT-LOGIC

Swaps only inputs to logic gates.

YES

Specifies that MOS transistors are not swappable.

Note: Parallel device smashing is done if you do not specify LVCHK[K].

NO

Smashes all split gates and forms complex structures. Default.

Examples

FIX-INPUT-ORDER = INSTANCE

FIX-INPUT-ORDER = EXCEPT-LOGIC

FIX-INPUT-ORDER = YES

FIX-INPUT-ORDER = NO

FLAG-ACUTEANGLE

FLAG-ACUTEANGLE = YES/NO

Description

Reports the coordinates of acute-angle polygons. This command works with the RESOLUTION command. RESOLUTION sets the grid step size.

FLAG-ACUTEANGLE reports polygons under “Problem Geometries” in `PRINTF.sum` file. This command processes the polygons and reports the x,y coordinates of the polygon vertices, the layer number, and the associated cell name. It checks the polygon in each cell structure (cell name) before merging the layout data.

Note: You need to specify the number in the PROB-GEOM-LIST statement if the number is over 500.

Arguments

YES

Flags acute-angle polygons.

NO

Default.

Example

FLAG-ACUTEANGLE = YES

FLAG-NON45

FLAG-NON45 = YES/NO

Description

Reports the coordinates of non-orthogonal lines (non 45-degree angles). This command works with the RESOLUTION command. RESOLUTION sets the grid step size.

FLAG-NON45 reports polygons under “Problem Geometries” in the *PRINTF.sum* file. This command processes the polygons and reports the x,y coordinates of the polygon vertices, the layer number, and the associated cell name. It checks the polygon in each cell structure (cell name) before merging the layout data.

Arguments

YES

Flags non-45 degree angle polygons.

NO

Default.

Example

FLAG-NON45 = YES

FLG-EXPTH-OFFGRD

FLG-EXPTH-OFFGRD = YES/NO

Description

Flags the coordinates of a non-orthogonal exploded path that are offgrid. This command works in conjunction with the values specified in the SCALE and RESOLUTION commands.

Arguments

YES

Flags the coordinates of the exploded path that are offgrid.

NO

Does not flag the coordinates of the exploded path that are offgrid. NO is the default.

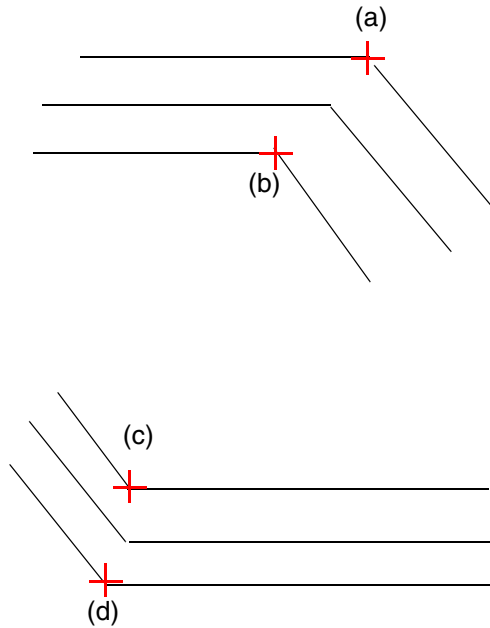
Note: In the user's database, a path is defined with a set of characteristic points of a path and the width of the path. An exploded path means that the path is expanded to a polygon shape with the specified width of the path.

Example

In the following non-orthogonal path, assume that the centerline coordinates are on-grid and the width of the path is an integer multiple of the RESOLUTION value.

Dracula Reference

Description Block Commands



Unless the RESOLUTION value is set equal to the SCALE value, the coordinates (a), (b), (c), and (d) will be technically offgrid when the path is exploded.

```
*DESCRIPTION
...
SCALE = 0.001 MICRON
RESOLUTION = 0.25 MICRON
...
FLG-EXPTH-OFFGRD = YES
...
*END
```

The coordinates (a), (b), (c), and (d) will be flagged as being offgrid in the summary file. The offgrid coordinates will be marked with an asterisk sign (*).

FLAG-OFFGRID

FLAG-OFFGRID = YES/NO {*grid-value*}

Description

Reports the coordinates of cell placements and polygons that have vertices off the grid. Works with the FLAG-PTH-OFFGRID and RESOLUTION commands. The RESOLUTION command sets the grid step size. To use a different grid, specify the *grid-value*.

FLAG-OFFGRID reports cell placements and polygons whose vertices are off the grid and off-grid cell placements under “Problem Geometries” in the *PRINTF.sum* file. This command processes the polygons and reports the x,y coordinates of the polygon vertices, the layer number, and the associated cell name. It checks the polygon in each cell structure (cell name) before merging the layout data.

Arguments

YES

Flags all cell placements and polygons that are off grid.

If FLAG-OFFGRID = YES and FLAG-PTH-OFFGRID = YES, all cell placements, polygons, and paths that are off grid are flagged.

If FLAG-OFFGRID = YES and FLAG-PTH-OFFGRID = NO, only cell placements and polygons that are off grid are flagged.

NO

Default. If FLAG-OFFGRID = NO and FLAG-PTH-OFFGRID = NO, no off-grid checks are done.

If FLAG-OFFGRID = NO and FLAG-PTH-OFFGRID = YES, only paths that are off grid are flagged.

grid-value

Specifies the grid. Value must be a real number greater than 0.0 but, less than 1.0.

Note: In prior versions of the product, the offgrids from individual layers were always taken into account for any FLAG-OFFGRID change. In product version 4.9, the offgrids from individual layers are taken into account for FLAG-OFFGRID changes only if those offgrids are smaller than the initial FLAG-OFFGRID.

Dracula Reference

Description Block Commands

Example

```
FLAG-OFFGRID = YES 0.1  
...  
L1 = 6 OFFGRID = 0.01
```

FLAG-PTH-OFFGRID

FLAG-PTH-OFFGRID = YES/NO

Description

Reports the coordinates of paths that have vertices off the grid. This command works with the FLAG-OFFGRID command.

FLAG-PTH-OFFGRID reports the x,y coordinates and the layer of the paths whose vertices fall off the grid under “Problem Geometries” in the *PRINTF.sum* file.

Arguments

YES

Flags all paths that are off grid.

If FLAG-PTH-OFFGRID = YES and FLAG-OFFGRID = YES, all cell placements, polygons, and paths that are off grid are flagged.

If FLAG-PTH-OFFGRID = YES and FLAG-OFFGRID = NO, only paths that are off grid are flagged.

NO

Default. If FLAG-PTH-OFFGRID = NO and FLAG-OFFGRID = NO, no off-grid checks are done.

Example

FLAG-PTH-OFFGRID = YES

FLAG-SELFINTERS

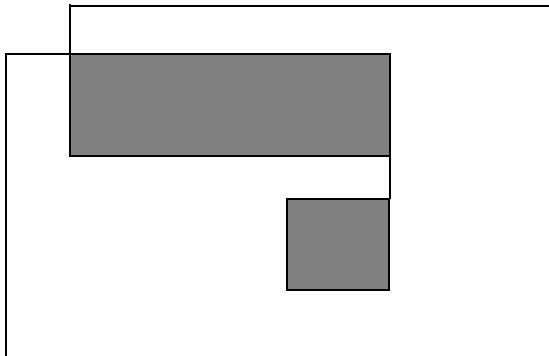
FLAG-SELFINTERS = YES {FULL}/NO

Description

Reports the coordinates of polygons that have self-intersecting vertices. This command works with the RESOLUTION command. The RESOLUTION command sets the grid step size.

FLAG-SELFINTERS reports polygons under “Problem Geometries” in the *PRINTF.sum* file. This command processes the polygons and reports the x,y coordinates of the self-intersecting vertices, the layer number, and the associated cell name and checks the polygon in each cell structure (cell name) before merging the layout data. This command requires moderate CPU resources for the EXPAND module.

If you specify FLAG-SELFINTERS = YES FULL, Dracula flags polygons that twist *and* are reentrant as illegal. For example, the following polygon contains a twist and is reentrant.



If you do not specify the *full* option, Dracula does not report this as an illegal polygon. If you do specify the *full* option, Dracula reports the polygon as illegal and lists its coordinates in the *.sum* file.

Arguments

YES

Flags self-intersecting polygons.

FULL

Flags reentrant polygons.

Dracula Reference

Description Block Commands

NO

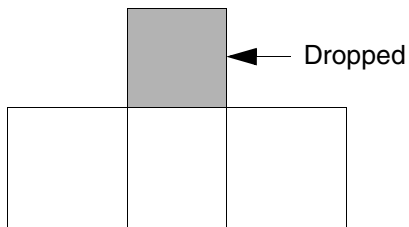
Default.

Example

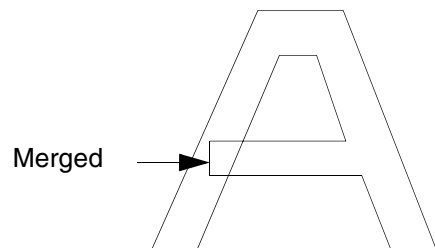
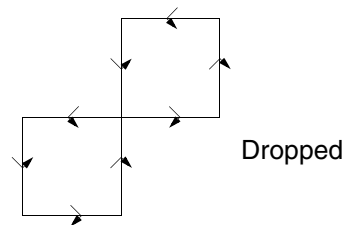
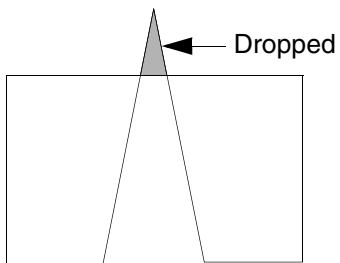
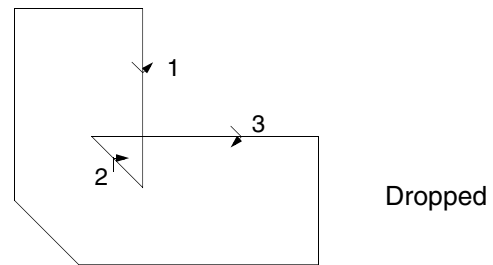
FLAG-SELFINTERS = YES

The following figure shows self-intersecting and reentrant polygons.

Processes self-intersecting
polygons.



Does not process reentrant polygons



FLAG-SELFTOUCH

FLAG-SELFTOUCH = YES/NO

Description

Reports the coordinates of polygons that have touching vertices. This command works with the RESOLUTION command. The RESOLUTION command sets the grid step size.

FLAG-SELFTOUCH reports these polygons under “Problem Geometries” in the *PRINTF.sum* file. This command processes the polygons and reports the x,y coordinates of the touching vertices, the layer number, and the associated cell name and checks the polygon in each cell structure (cell name) before merging the layout data. This command requires moderate CPU resources for the EXPAND module.

Arguments

YES

Flags self-touching polygons.

YES DONUT

Flags self-touching donuts.

NO

Default.

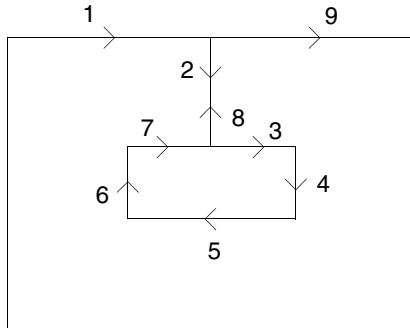
Example

FLAG-SELFTOUCH = YES

Dracula Reference

Description Block Commands

The following figure shows a self-touching polygon. Self-touching polygons are not processed.



FLAT-GDS-SMALL

FLAT-GDS-SMALL=YES

Description

Searches for small cells such as CONTACT or VIA in the given GDS2 stream file, and flattens them one level. This reduces the temporary files of the Dracula EXPAND stage and by reducing the temporary file size, sometimes improves speed.

The FLAT-GDS-SMALL command is best used with large designs where contact or via are defined as a "Cell".

Note: This feature reads the stream file twice and is not recommended for use with a design that is not very big or contains no small cells, as it will cause a little overhead on elapsed time.

Arguments

YES

Expands small cells into polygon format in Interface stage.

NO

Default.

Examples

SYSTEM = GDS2

FLAT-GDS-SMALL = YES

FLATTEN-PWRGND

FLATTEN-PWRGND = NO/YES

Description

Explodes the cell power or ground onto the composite plane to detect shorts formed between cell and composite levels.

If you digitize text directly down to cells without passing through the composite plane, you can cause shorts involving power or ground. This command flattens the cell power/ground text back onto the composite plane to detect these shorts. Be sure that the cell power or ground has the same name as originally used on the composite plane.

Checking Method

modes

Composite.

checks

HERC, HLVS, and HLPE.

Arguments

YES

Explodes power/ground onto the composite plane.

NO

Default.

Example

FLATTEN-PWRGND = YES

FNODE-CONNECT

`FNODE-CONNECT = [SubType1 {,SubType2...}] Power_Pad1 {,[SubTypeN {,SubTypeX...}],Power_Pad2.....}`

Description

When Dracula runs selective LPE with SMART-LPE=SINGLENET, the source/drain of the selected loading devices might be connected to nodes that become floating in the resultant SPICE netlist. Use the FNODE-CONNECT command to connect these nodes to power or ground so that the timing analysis can be done with the SPICE output .

Arguments

SubType1, SubType2

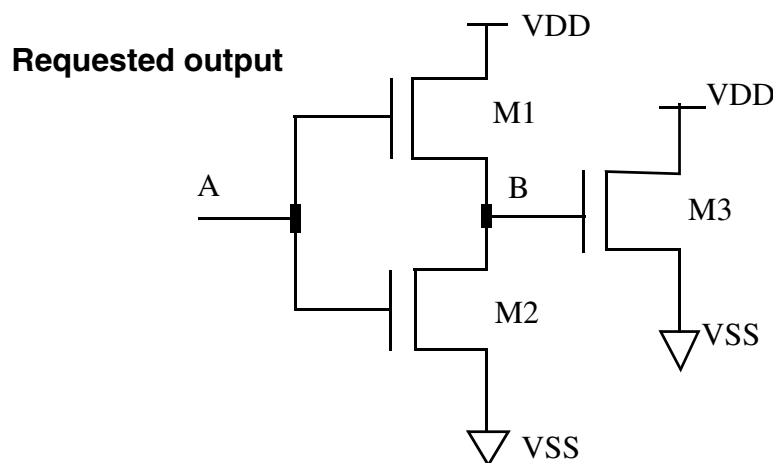
The subtype of loading devices connected to nodes that become floating.

Power_Pad1, Power_Pad2

Power/Ground nodes with which the floating nodes are to be connected.

Examples

`FNODE-CONNECT = [N/NA] VSS, [P,PA] VDD, [NN] VSS1, [PP] VDD`



Dracula Reference

Description Block Commands

When SMART-LPE=SINGLENET and the selected node is A, the netlist is reported as the figure. If M3 is a P type loading device, it connects node C and D to VCC. If M3 is NMOS, it connects node C and D to VSS.

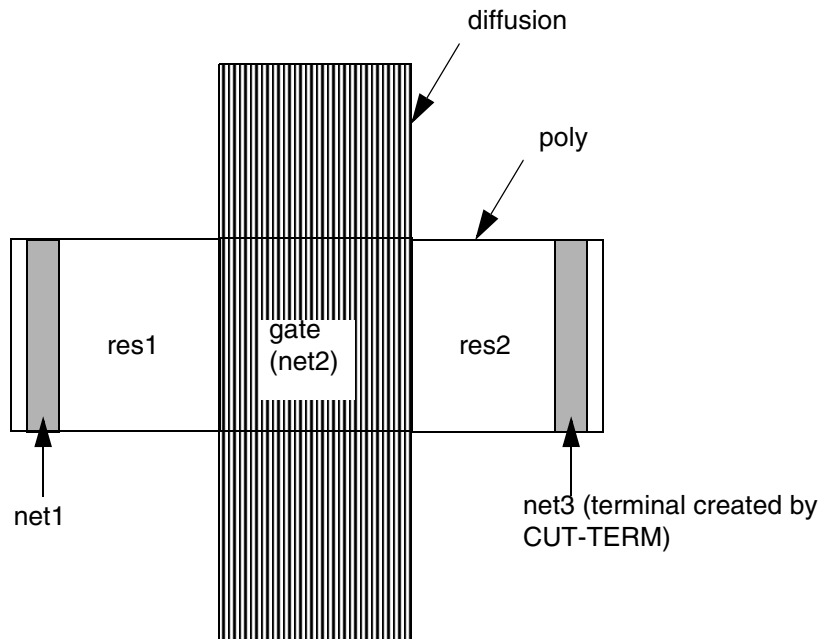
GATE-RES-EXT

`GATE-RES-EXT = [subType1] [subType2]`

Description

Turns on gate resistance extraction. This command explicitly specifies which subtype of MOS extracts gate resistance. You can also use `GATE-RES-EXT = ALL`. This specifies that all subtypes extract gate resistance.

If both sides of the gate are connected, as shown below, the parasitic resistance of the gate is R .



The original netlist looks like the following:

```

net1          net2          net3
O--/\ /\ /\ \-----O-----\ /\ /\ /\ \-----O
      R(res1)          R(res2)

```

After gate resistance compensation, the netlist looks like the following example:

```

net1          net2          net3
O--/\ /\ /\ \-----O-----\ /\ /\ /\ \-----O
      R(res1)+R/2      R(res2)+R/2

```

Parasitic gate resistance is evenly distributed to its adjacent resistors.

Dracula Reference

Description Block Commands

Note: You must specify the gate layer in the CUT-TERM command of the gate connection layer. To extract gate resistance, you must also specify all parameters, including the R parameter in the PARSET command, and use the EQUATION command.

Arguments

[subtype1] [subtype2]

Enumeration list of subtypes that perform gate resistance extraction.

Example

```
...
PARSET USER ANG W1 W2 AREA L W R ; add R parameter
GATE-RES-EXT= [N],[P] ; specify interested
; ; subtype
...
*END
*OPERATION
...
OR PGATE NGATE GATE
OR GATE POLTM PTRM
;
CUT-TERM POLY CNT PYRES PYTRM PTRM ; gate layer must be specified
CUT-TERM ME1 CNTVIA ME1RES ME1TRM
ELEMENT MOS[N] NGATE PYTRM NSD COPWL ;PWELL
ELEMENT MOS[P] PGATE PYTRM PSD CONWL ;NWELL
PARASITIC RES[PY] PYRES PYTRM
ATTRIBUTE RES[PY] 0.01
PARASITIC RES[M1] ME1RES ME1TRM
ATTRIBUTE RES[M1] 0.01
LEXTRACT USER NGATE NSD BY MOS[N] AAA &
EQUATION W= (1-ANG/60*0.02)*(W1+W2)/2 &
EQUATION L= AREA/W &
EQUATION R= 0.723* W/L ; gate resistance is extracted
LEXTRACT USER PGATE PSD BY MOS[P] BBB &
EQUATION W= (1-ANG/60*0.07)*(W1+W2)/2 &
EQUATION L= AREA/W &
EQUATION R= 2.723* W/L ; gate resistance is extracted
```

GEN-TEXT-FILE

GEN-TEXT-FILE = *filename*

Description

Specifies whether Dracula automatically texts the Hcells. At the same time, this command specifies the name of the file that contains the GEN-TEXT generated text. Use this file as the HEDTEXT file in cell or composite mode HLVS or HLPE runs.

After a GEN-TEXT run completes, the named file contains all Hcell text information, including the initial text from the layout (initial correspondence text).

Checking Method

modes

Flat, cell, and composite.

checks

HLVS and HLPE.

Arguments

filename

Name of file that contains the generated text. The file name can include a path name. The maximum size for path name or file name is 128 characters.

Example

The following example generates a text file to be used by the HEDTEXT or EDTEXT command.

```
GEN-TEXT-FILE = cell.htx
```

GEN-TEXT-FLTNODE

GEN-TEXT-FLTNODE = YES/NO

Description

Specifies whether floating nodes or feedthroughs are assigned text by GEN-TEXT. If you have feedthroughs within Hcells, you must specify GEN-TEXT-FLTNODE = YES. Otherwise, Dracula generates many discrepancy errors when running composite mode HLVS or HLPE. HLVS and HLPE handle these text names differently from other text nodes because they do not appear in the netlist version of the Hcell.

Note: In cell mode, the nodal stage does not have information about what layers are SCONNECTed rather than CONNECTed. Therefore, GEN-TEXT-FLTNODE might produce the same results in both flat and cell mode.

Checking Method

modes

Flat, cell

checks

HLVS and HLPE.

Arguments

YES

Specifies that floating nodes be assigned text by GEN-TEXT.

NO

No nodes are assigned text. Default.

You can text feedthroughs with a text name of the form, *FTHRU#*, with # being a number (for example, *FTHRU44*).

Note: If you specify GEN-TEXT-FRAME, the HCELL command *FRAME* option, and GEN-TEXT-FILE when feedthroughs are found, GEN-TEXT texts the layout so that connectivity information is preserved, even though the feedthrough might be split up. For this reason, you must specify GEN-TEXT-WIRE = YES, in addition to GEN-TEXT-FLTNODE = YES, during the Hcell HLVS GEN-TEXT run.

Dracula Reference

Description Block Commands

Dracula issues an error message if connected and soft-connected layers are not separable. (They are mixed in the connecting sequence.)

Example

```
GEN-TEXT-FLTNODE = YES
```


GEN-TEXT-FRAME

`GEN-TEXT-FRAME = value-size units`

Description

Generates a frame inside the Hcell boundary for use with GEN-TEXT. This frame must correspond precisely in size to all frames specified by HCELL commands for the composite HLVS run.

If you specify GEN-TEXT-FRAME and FRAME BY in HCELL commands or in an Hcell file, the values you specify in the FRAME BY option take precedence. This maintains consistency between frame values you use in GEN-TEXT and frame values you use in composite mode LVS. Therefore, if you specify FRAME BY in HCELL commands, you do not need to specify GEN-TEXT-FRAME in cell mode GEN-TEXT.

Checking Method

modes

Flat, cell, and composite.

checks

HLVS and HLPE.

Arguments

value-size

Specifies the dimensions of the frame by giving the distance from the edges of the Hcell box.

units

Natural database units in microns or mils.

If possible, HLVS creates the Hcell text within the frame dimension. If any nodes in the Hcell cannot be assigned text within the frame area, Dracula reports these nodes in the `.lvs` discrepancy report file.

Examples

```
GEN-TEXT-FRAME = 10 microns
HCELL = ALU ALU1 frame by 2
HCELL = PLUG PLUG
```

Dracula Reference

Description Block Commands

For *ALU*, the frame value of 2 is used. For *PLUG*, the frame value of 10 is used.

```
HCELL = ALU ALU1 frame by 2  
HCELL = PLUG PLUG
```

For *ALU*, the frame value of 2 is used; for *PLUG*, no frame is in effect and the whole *PLUG* cell is used for GEN-TEXT.

GEN-TEXT-WIRE

GEN-TEXT-WIRE = YES/NO

Description

Specifies whether GEN-TEXT generates wire type information on text when it finds geometries that need to be marked as an *Internal-Wire* or a *MustJoin* wire type. GEN-TEXT generates wire type information by concatenating a semicolon (;) and a number to the appropriate text.

Checking Method

modes

Flat, cell, and composite.

checks

HLVS and HLPE.

Arguments

YES

Generates wire type information for *Internal-Wire* geometries or *MustJoin* wire types.

NO

Default.



Be aware of potential errors when you don't specify GEN-TEXT-WIRE = YES. In particular, you must always set GEN-TEXT-WIRE if you specify a frame with a GEN-TEXT-FRAME command and with HCELL commands.

Example

GEN-TEXT-WIRE = YES

GEN-TEXT generates wire type information for *Internal-Wire* geometries or *MustJoin* wire types.

GEN-XRF-RPT

GEN-XRF-RPT = YES

Description

Forces Dracula to output device and node cross-reference files `DEVXRF.DAT` and `NETXRF.DAT` on demand.

Note: This command will only support `FLAT` mode. If `CHECK-MODE` is not `FLAT`, `PDRACULA` should output `ERROR` message.

Note: This command will only support `LVS` and `SVS` flow.

Note: If both `DEVXRF.DAT` and `NETXRF.DAT` are not empty, that means old cross-reference files exist and Dracula will do nothing. Only when one or two files are empty or don't exist will Dracula generate those reports again.

Arguments

YES

Will output `DEVXRF.DAT` and `NETXRF.DAT` files.

Example

```
/bin/rm -f FOR008.DAT
/bin/ln -s NETXRF.DAT FOR008.DAT
/bin/rm -f FOR009.DAT
/bin/ln/ -s DEVXRF.DAT FOR009.DAT
```

GROUND-NODE

`GROUND-NODE = name1, name2...`

Description

Specifies ground nodes on text in the layout or in EDTEXT. If you do not properly text a non-unique ground text string in the layout, this command converts the non-unique text string to ground (pad type G). It does not add new text.

You can specify an asterisk (*) as a wildcard character following the ground text string.

Note: In version 4.7.0399, Dracula can support up to 100,000 nodes for the GROUND-NODE command in the Description block.

Arguments

`name1 name2...`

Text to convert to ground (pad type G).

Examples

The following example specifies VSS1, VSS2, and GND as ground nodes:

```
GROUND-NODE = VSS1, VSS2, GND
```

The following example indicates that the VSS text string followed by any character is a ground text string:

```
GROUND-NODE = VSS*
```

HCELL

`HCELL = cell-name {logic-name {FRAME BY value-size}}`

Description

Specifies layout cells to be used as Hcells. An Hcell is a cell structure that is preserved in its cell form because it is used in many instances.

Checking Method

modes

Cell, composite, and hierarchical.

checks

HDRC, HERC, HLVS, and HLPE.

Arguments

cell-name

Name of the cell structure in the layout database that is used as an Hcell.

logic-name

Name of the logic circuit in the schematic that represents the subcircuit description that is equivalent to the layout cell structure (Hcell).

value-size

Specifies the width of the inside border used to frame the cell. The units of *value-size* are specified by the SCALE command (microns or mils).

HDRC uses HCELL commands with its automatic selection algorithm to determine which layout cells to use as Hcells. The other checks use the HCELL command as the only way to determine which Hcells to use.

For HLVS you must give the *logic-name* parameter, which is the name of a cell in the schematic netlist that corresponds to the layout cell chosen as Hcell.

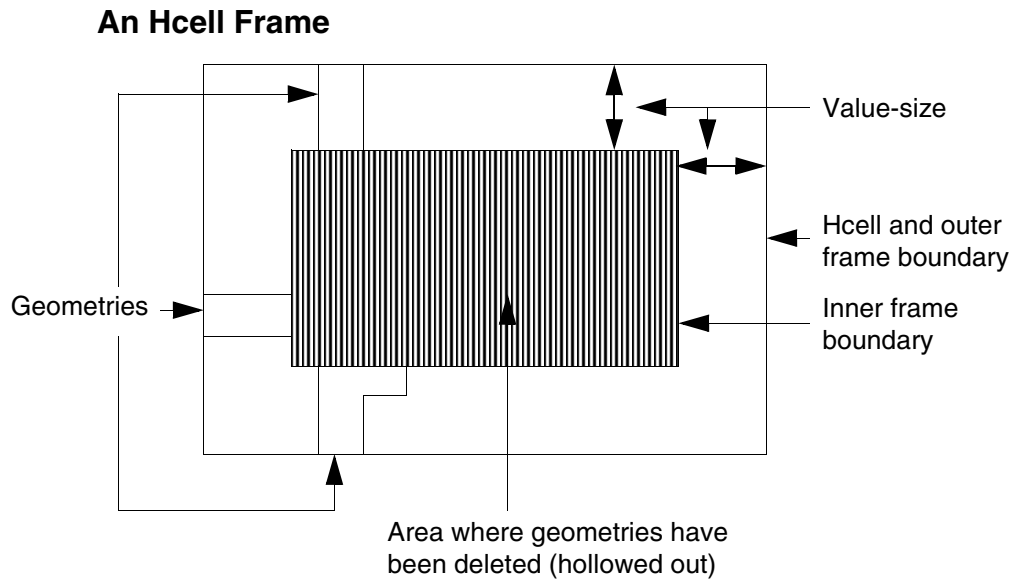
You can use multiple HCELL commands. You can also specify a number of Hcell specifications using the HCELL-FILE command.

You can use the FRAME option only for a composite mode HLVS run. The FRAME option specifies a rectangular boundary within the Hcell, which is “hollowed-out” for the Dracula run.

Dracula Reference

Description Block Commands

The distance from the border of the frame to the border of the Hcell is established with the *value-size* parameter. Refer to the following figure, which is a diagram of an Hcell frame.



Dracula discards all geometries or portions of geometries inside the frame, thereby speeding up the composite HLVS run. The portions of geometries between the frame and the Hcell boundary remain to provide appropriate connectivity for the composite HLVS check.

When using a FRAME option, you must use wire types to text both ends of internal wires that have been disconnected by the frame. The GEN-TEXT operation provides this additional texting.

When using GEN-TEXT in conjunction with frames, you must specify that all frames defined by composite mode HCELL commands have the same dimensions as the frame specified in the cell mode GEN-TEXT-FRAME command. For more information, refer to the GEN-TEXT commands.

Examples

The following example is used for HLVS.

```
HCELL = alu alu1
```

The following example is used for HLVS.

```
HCELL = plug rom frame by 2
```

The following example is used for HDRC.

Dracula Reference

Description Block Commands

HCELL = plug

HCELL-COLUMN-1

HCELL-COLUMN-1 = LAY/SCH

Description

Specifies whether the first column in the Hcell table lists layout or schematic cells. For more information about the Hcell table, refer to the HCELL-FILE command description.

You must include this command in the Description block only if you run a schematic versus schematic (SVS) check with hierarchical Dracula. SVS requires two other commands as well: SVS-LAYOUT and SVS-SCHEMATIC.

Use this command in hierarchical Dracula only.

Checking Methods

modes

Cell and composite.

checks

SVS.

Arguments

LAY/SCH

Type of cell names listed in the first column of the Hcell table. Type *LAY* for layout; type *SCH* for schematic. The default is *LAY*.

Example

To switch the role of the layout and schematic circuits after running LOGLVS, use the HCELL-COLUMN-1 command. For example, on the first LOGLVS run:

```
SVS-SCHEMATIC = LVSLOGIC.DAT
SVS-LAYOUT = ../LVSLOGIC.DAT
```

On the second run, to switch the circuits without rerunning LOGLVS, modify the rules this way:

```
SVS-SCHEMATIC = ../LVSLOGIC.DAT
SVS-LAYOUT = LVSLOGIC.DAT
HCELL-COLUMN-1 = SCH
```

HCELL-FILE

`HCELL-FILE = filename`

Description

Specifies a file containing a set of HCELL command statements. The format of each statement is the same as that of the HCELL command except that the keyword “HCELL =” is left out of each line. The statements in this file serve the same purpose as a group of HCELL commands. Use this command for HLVS runs (cell or composite mode) because LOGLVS, the network compiler, requires the same Hcell data as HLVS.

An Hcell file contains layout cell names, netlist cell names, and FRAME options. For example

```
HCELL = plug rom frame by 2
```

is equivalent to specifying an Hcell file with the following line

```
plug rom frame by 2
```

To connect layers that are not input-layers but that have geometries needed for texting, you can assign level numbers for the appropriate input-layers and use these input-layers for appending to the Hcells.

Checking Method

modes

Cell, hierarchical, and composite.

checks

HDRC, HERC, HLVS, and HLPE.

Arguments

filename

Specifies the name of the file containing the Hcell name descriptions. The file name can also include a path name. The maximum size for a path name or file name is 128 characters.

Note: If you specify the FRAME BY option, it applies for a composite mode HLVS run only.

HCELL-IN-HCELL

HCELL-IN-HCELL = YES/NO

Description

Specifies that the instances of Hcells under parent Hcells flatten into transistors under the parent Hcells so that Dracula allows nested Hcells during hierarchical extraction.

Checking Method

modes

Composite.

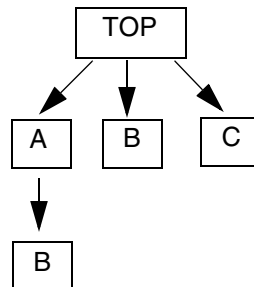
checks

HLVS and LOGLVS.

Arguments

YES

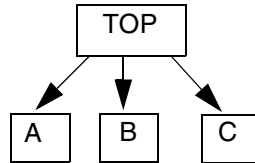
Specifies that if you have the following figure, where you specify A, B, and C are Hcells, and one cell B instance is under cell A,



Dracula Reference

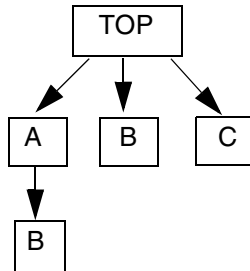
Description Block Commands

both A and B remain Hcells and the instance B under A is exploded, which results in the following:



NO

Default. No nested Hcells are allowed when running Dracula. If nested cells are found, Dracula issues a warning message. For example, in the following figure, if you specify A, B, and C as Hcells, the EXPAND stage aborts.



Dracula issues the following warning messages:

```
*/W* BOTH DESCENDANT AND ANCESTOR ARE SELECTED AS H-CELLS!  
*/E* IT IS NOT ALLOWED IN COMPOSITE MODE, PLEASE CHECK THE  
HCELL-FILE, OR USE <HCELL-IN-HCELL = YES>.
```

HCELL-MAX-PLACEMENTS

`HCELL-MAX-PLACEMENTS = value`

Description

Specifies a parameter for HDRC duplication of instance records. Use this command to avoid duplicating instance records when you select small cells with large numbers of placements as Hcells.

Checking Method

modes

Hierarchical.

checks

HDRC.

Arguments

value

Specifies the maximum number of cell placements a cell can have in order to be an Hcell.

Example

`HCELL-MAX-PLACEMENTS = 600`

HCELL-MAX-SEGMENTS

HCELL-MAX-SEGMENTS= *seg-cnt*

Description

Specifies a parameter for HDRC selection of layout cells as Hcells. Use this command to avoid selecting large cells as Hcells. For a detailed explanation of HDRC Hcell selection, refer to [Chapter 3, “Checking Design Rules \(DRC\).”](#)

Checking Method

modes

Hierarchical.

checks

HDRC.

Arguments

seg-cnt

Specifies the maximum number of line segments required for a cell to be an Hcell candidate. Default is

$\text{MAX}(L, \text{total \# of layers} * 1,000,000)L = N / 256$

Where N is the total number of line segments under the primary cell provided in the EXPAND module.

Example

HCELL-MAX-SEGMENTS = 5,000,000

HCELL-OPTIMIZE-PLACEMENTS

HCELL-OPTIMIZE-PLACEMENTS = value

Description

Specifies a parameter for HDRC duplication of instance records. Using this command avoid many duplicated instance records for single master cell that slow HDRC run.

Checking Method

Modes

Hierarchical

Checks

HDRC

Arguments

value

Specifies the maximum number of cell placements for master cell. Over this number, a new master cell created for other instance records.

Example

HCELL-OPTIMIZE-PLACEMENTS = 50

HCELL-RULE

HCELL-RULE = *place-cnt1, seg-cnt1, {place-cnt2, seg-cnt2} {...}*

Description

Specifies parameters for HDRC selection of layout cells as Hcells. For a detailed explanation of HDRC Hcell selection, refer to Chapter 3, "Checking Design Rules ERC" in this manual.

Checking Method

modes

Hierarchical.

checks

HDRC.

Arguments

place-cnt

Specifies the minimum number of cell placements required for a cell to be an Hcell candidate. Default is 2.

seg-cnt

Specifies the minimum polygon segment count for a cell to be an Hcell candidate. You can specify multiple *place-cntN, seg-cntN* pairs. Default is four times the number of input layers.

Cells placed a number of times greater than or equal to the *place-cnt* parameter and having a number of segments (a rectangular polygon contains four segments) greater than or equal to the *seg-cnt* parameter qualify as Hcell candidates.

Place-cnt and *seg-cnt* act as requirement pairs. When multiple *place-cnt, seg-cnt* requirement pairs are specified, cells meeting any of the requirement pairs qualify as Hcell candidates.

To prevent any cells from being selected automatically, allowing you to direct selection of cells, you can specify a *place-cnt* and a *seg-cnt* of 9999, thus guaranteeing that no cell qualifies as an Hcell candidate.

Dracula Reference

Description Block Commands

Examples

In the following example, cells placed at least four times and having a minimum of 150 polygon edges are considered potential Hcell candidates.

```
HCELL-RULE = 4,150
```

In the following example, cells that either have been placed a minimum of four times and contain at least 150 polygon edges or that have been placed at least two times and contain a minimum of 300 polygon edges are considered potential Hcell candidates.

```
HCELL-RULE = 4,150 2,300
```

HIER-OUTDISK

HIER-OUTDISK = {*outdisk-name*}

Description

Specifies the name of the output file that contains error cells generated in the original design hierarchy. You must specify this command in the Description block when you include the CELL-ERROR-REP command with the ORIG-ALL, ORIG-DRC, or *cell-name* options.

Standard errors Dracula reports in the two-level hierarchy (the cell and composite planes) are still output to the file you specify using the OUTDISK command. If you use CELL-ERROR-REP, you have two output files. The output file specified by OUTDISK contains errors in the two-level Dracula hierarchy. The output file specified by the HIER-OUTDISK command contains the same errors in the original design hierarchy.

If you use CELL-ERROR-REP with the ORIG-ALL, ORIG-DRC, or *cell-name* option, but do not include HIER-OUTDISK, PDRACULA issues a warning and prompts for an output filename or uses the default name, HIEROUT.ODD.

Arguments

outdisk-name

Specifies the output file name for layout error cells in the design hierarchy. Maximum path or file name is 128 characters.

IGNORE-SCH-SUB

IGNORE-SCH-SUB = YES/NO

Description

Ignores the schematic-side substrate in the SCHMIN stage.

Arguments

YES

Sets the MOS substrate automatically in the schematic side to zero. The Dracula tool ignores this substrate whether you specify a substrate node in the netlist or not.

NO

Is the default. This argument accepts the substrate in the schematic side. If you specify a substrate node in the netlist, the substrate takes effect in the SCHMIN stage.

INAME-TOPDOWN

INAME-TOPDOWN = YES/NO

Description

Specify the full path instance names in a flat spice netlist output by Dracula to be in top-down or bottom-up order of the hierarchy.

Arguments

YES

Full path instance names are output in top-down order.

NO

Full path instance names are output in bottom-up order. This is Default. Please notice that by default, full path instances generated by LOGLVIS in file `EXPELE.LIS` is in bottom-up order.

Example

INAME-TOPDOWN = YES

INDISK

`INDISK = {filename} {directory-name}`

Description

Specifies the input file containing your layout. You must specify this command in the Description block.

Arguments

filename

The input file name for a layout database. The maximum path or file name is 128 characters.

directory-name

The directory containing Edge database blocks or a Design Framework II library.

If you assign a logical name and the name matches the file name you specify, the logical name has priority. You can use the logical name to refer to any disk file.

Note: When SYSTEM=CADENCE_44DD and INDISK point to cds.lib, then INDISK is equivalent to the INDISK-FILE command.

Examples

The following example specifies the directory name for Edge or DFII databases:

```
INDISK = /usr/project/data
```

This example specifies the input file name as *myfile.DAT*:

```
INDISK = myfile
```

This example specifies the input file name as *256K.DB*:

```
INDISK = 256K.DB
```

This example specifies the input file name as *256K*. (period (.) extension):

```
INDISK = 256K
```

INDISK-FILE

`INDISK-FILE = filename`

Description

Specifies the file containing the library paths to Design Framework II databases or the block paths for Edge databases. This file is required if you specify multiple Design Framework II library paths or multiple Edge block paths.

There are 2 sections to this command:

■ SYSTEM=CADENCE_44DD or SYSTEM=CADENCE

When `SYSTEM=CADENCE_44DD`, `PDRACULA` will not check the existence of the library paths defined in the `INDISK-FILE`. The `cds.lib` file is used by the `CDSIN44` module to find these paths. In version 4.9, the new `CDSIN44` behaves as follows:

- a. If `INDISK-FILE` is set to a `cds.lib` file, for example

```
INDISK-FILE = /usr/projects/cds.lib
```

then `CDSIN44` will use that file

- b. If `cds.lib` is in the current working directory, then `CDSIN44` will use it.
- c. If neither of the cases given above apply, then `CDSIN44` will search for `cds.lib` in directories defined in the `setup.loc` file in the Cadence hierarchy.
- d. If no `cds.lib` file is found, `CDSIN44` will abort.

Example for `SYSTEM=CADENCE_44DD`:

```
INDISK-FILE = /drac30/test/database/cds.lib
```

where the contents of the `cds.lib` file is

```
DEFINE dracLib /net/cds123/usr1/home/data/dracLib  
DEFINE refLib /net/cds123/usr1/project/refLibs/stdCells
```

This specifies the path for the main library, `dracLib`, as well as any reference library paths that may be used by `dracLib`.

When `SYSTEM=CADENCE_44DD` or `SYSTEM=CADENCE`, the DFII license server program must be running on all workstations from which Dracula accesses DFII databases. Prior to IC 445, the DFII license server program was called `cdsd`; after IC445 it became known as `CLS` (Cadence Locking System).

See the *Cadence Application Infrastructure User Guide* (chapter 11) for more information on `CLS` and `cdsd`.

Dracula Reference

Description Block Commands

For example, if you have DFII data on three workstations that are mounted to your workstation, when you use the `df` command to list your disks you see:

```
workstation1:/usr/libDir1 852847 714817 52746 93% /mount/libDir1
workstation2:/usr/libDir2 852847 714817 52746 93% /mount/libDir2
workstation3:/usr/libDir3 852847 714817 52746 93% /mount/libDir3
```

In your *indisk.file*, you have:

```
./myLibDir
/mount/libDir1
/mount/libDir2
/mount/libDir3
```

The `cdsd` or `CLS` program must be running on workstation1, workstation2, and workstation3 in order for Dracula to access the DFII data.

■ SYSTEM=OA

Similar to the C-level database access (CDBA), the Genesis II database or OpenAccess (OA) database is another Cadence EDA database infrastructure that allows designers to save their electronic designs into digital file format.

The major difference between the OA database and CDBA is that the OpenAccess database is designed to be open to or shared by all users so that it can be shared, accessed and perfected by the users.

In this release and subsequent releases, the Dracula product implements the modules OAIN to read and OAOUT to write in the OA database format. OAIN will read the OA database and translate (interpret) it into the data format that the Dracula product can understand. Similarly OAOUT will convert (translate) the Dracula polygon formatted data into the OA data format. OAIN and OAOUT work exactly the same way as CDSIN44 and CDSOUT.

Once `SYSTEM = OA` is defined in the rule file, PDRACULA interprets the commands in the rule files to invoke OAIN and OAOUT.

Before invoking OpenAccess, you must define these commands in the “Description Block Commands” section of your rule file:

```
SYSTEM=OA
INDISK_FILE = specified location for the library
LIBRARY = name of the library directory
```

For more information on the OpenAccess database, see this website:
<http://www.openeda.org/index.html>.

Dracula Reference

Description Block Commands

Arguments

filename

The name of the file containing the paths to a Design Framework II or Edge database.
Each line is a path to each database.

Examples

Example 1

```
INDISK = /net/cds123/usr1/home/data/dracLib
```

Specifies the path for the library dracLib.

Example 2

```
INDISK-FILE = /drac30/test/database/indisk.file
```

where the `indisk.file` is

```
/net/cds123/usr1/home/data/dracLib  
/cds/lib/cdslib  
/cds/lib/schematic
```

Specifies the path to multiple libraries.

INSTPIN-FILE

INSTPIN-FILE[S] = *instancePin*

Description

Prints out instance pin information in ASCII format in the specified file. Use for composite mode only. The output file follows this structure:

Instance name

TerminalName (X=xcoordinate Y=ycoordinate) SubnodeNet

Instance name

Hcell instance name

Terminal name

Terminal name (pin name) on the Hcell

Subnode net

Signal name on the composite level after parasitic resistors are formed

Checking Method

modes

Composite.

Options

S

Specifies the backannotation naming convention.

Note: If you did not specify LPECHK/LVSCHK in your rule file, you cannot specify the S option.

Arguments

instancePin

Name of the output file.

Example

In the following example, INV is an Hcell and TOP is the Primary block.

```

.SUBCKT    INV    Z    Y
...
.ENDS

.SUBCKT    TOP    O    I
X1  A      I    /INV
X2  O      A    /INV
.ENDS

```

HPRE reports the following information to the output file *instancePin*:

X1	Y	X=2.00	Y=4.00	#I:3
X1	Z	X=4.00	Y=4.00	#A:2
X2	Y	X=6.00	Y=4.00	#A:3
X2	Z	X=8.00	Y=4.00	#O:2

Each line contains an instance pin's information:

- The first field is the cell instance name.
- The second field is the pin name.
- The third and fourth fields are the instance pin's x,y coordinates.
- The last field is the node name of the instance pin connected with the hard-wired prefix '#'.

The convention of the node name is the original net name concatenated with the subnode-delimiter and subnode number.

The resulting SPICE file might look like the following:

```

.SUBCKT TOP O I
R1 I I:2 1.80000E00
R2 I:2 I:3 1.20000E00
R3 A A:2 2.00000E00
R4 A A:3 1.50000E00
R5 O O:2 1.40000E00
X1 A:2 I:3 /INV
X2 O:2 A:3 /INV
.ENDS

```

INTERNAL-NODE

`INTERNAL-NODE = name1, name2...namen`

Description

Specifies the names of internal nodes that are assigned text in the layout or in EDTEXT. If you do not properly text a non-unique internal text string in the layout, this command converts the non-unique text string into an internal node (pad type *N*). It does not add new text.

You can specify an asterisk (*) as a wildcard character following the power text string.

Arguments

namen

Specifies internal nodes (pad type *N*).

Examples

In the following example, INET1, INET2, and INET3 are internal nodes:

```
INTERNAL-NODE = INET1, INET2, INET3
```

In the following example, the INET text string followed by any character is an internal node:

```
INTERNAL-NODE = INET*
```

INTERNODE-PREFIX

INTERNODE-PREFIX = " " or string

Description

INTERNODE-PREFIX lets you specify a prefix string for layout nodes that are not matched by LVS. In some cases, LVS does not match the internal nodes, and so the internal nodes are reported as unmatched. Also, the devices connected to the nodes are commented out due to unmatched terminals. By default, LPE/PRE reports nodes as the node name or the layout node number with the prefix "?" (question mark). INTERNODE-PREFIX reports the unmatched node as the node name or layout node number prefixed with a string that you specify. Connected devices are not commented out.

Note: Be aware of the following risk of using INTERNODE-PREFIX: *Real* unmatched nodes are also reported using the same manner described above.

Arguments

" "

Specifies no prefix.

string

One to four characters. The string cannot contain a space, a colon (:), a semicolon (;), or a question mark (?). If you specify a string longer than four characters, Dracula reports an error.

Example

INTERNODE-PREFIX = " " ; overwrites the prefix to be empty

The SPICE result becomes

```
M18150      N-9541_2      N-9721_10      4625      VDD P L=1.20U W=8.82U
M18151      N-9541_2      N-9720_2       4625      VDD P L=1.20U W=10.50U
M18152      4625          N-9724_6       I-9719-NP1 VDD P L=1.20U W=8.82U
```

Instead of

```
*M18150      N-9541_2      N-9721_10 ?4625      VDD P L=1.20U W=8.82U
*M18151      N-9541_2      N-9720_2       ?4625      VDD P L=1.20U W=10.50U
*M18152      ?4625          N-9724_6       I-9719-NP1 VDD P L=1.20U W=8.82U
```

KCELL-FILE

KCELL-FILE = <kcell_filename>

Description

Specifies the cell name mapping file for the “Keep Cell Text” feature in LVS or SVS or LVL jobs.

KCELL-FILE has the same format as HCELL-FILE except that it only takes the first two columns. The first column specifies the cell name in the schematic side while the second column specifies the cell name in the layout side.

The file should be the same file which you specified in the LOGLVS command KCELL, if you are doing an LVS or SVS job.

“Keep Cell Text” is a feature that allows Dracula in the flat mode to keep the “Cell Text” in a full hierarchical mode which could help LVS or SVS or LVL in two aspects: accuracy and performance. For details about “Keep Cell Text,” see its Application Notes.

Checking Method

CHECK-MODE = FLAT

Arguments

kcell_filename

Specifies the name of the file containing the Kcell name tables.

Example

KCELL-FILE = myKcell.tab

or

KCELL-FILE = /pathname/myKcell.tab

Dracula Reference

Description Block Commands

KEEPDATA

KEEPDATA = YES/NO/SMART/DRACULAINTERACTIVE/INQUERY

Description

Determines whether to save intermediate data files. The `KEEPDATA` command helps to control disk space usage by managing the Dracula binary-generated files that are created during the execution of a job. These files can have one of the following extensions on them: `.CEL`, `.DAT`, `.ENC`, `.ENV`, or `.NDX`. Either the user creates these files via the `OPERATION` block or the software creates these files for its own internal purposes.

You can specify any of these five values with this command:

- `KEEPDATA=YES`
- `KEEPDATA=NO`
- `KEEPDATA=SMART`
- `KEEPDATA=DRACULAINTERACTIVE`
- `KEEPDATA=INQUERY`

Note:

- ❑ Unless you are restarting a Dracula job from the very beginning, it *may not* be possible to restart a Dracula job from any other stage if the `KEEPDATA` command is set to either `NO` or `SMART` or `DRACULAINTERACTIVE` *even if* breakpoints are specified in the Dracula rules file. (Breakpoints you specify for the `BREAK` or `RESTART` commands become meaningless and either command ends the job). The *only* exception to this is when you are restarting a Dracula job from the exact stage where the job aborted due to a lack of computer resources (such as a lack of disk space). This is due to the fact that the input data `.DAT` files needed by the stage where you are restarting the job may have been deleted during the previous run.
- ❑ The `KEEPDATA` command does not delete the input files for the following commands when the run ends: `INDISK`, `OUTDISK`, `SCHEMATIC`, and `EXPORT` regardless of which value you specify for this command.
- ❑ Dracula distributed always runs with `KEEPDATA=YES`, no matter what users specify. For a single CPU job, `KEEPDATA=NO` works correctly. But, for a `MULTICPU` job, specifying `KEEPDATA=NO` will not delete intermediate files such as `.DAT` after you run jobs.
- ❑ See the [Dracula Distributed Processing and the Dracula Graphical User Interface Tool](#) section in Chapter 2 of this manual.

Dracula Reference

Description Block Commands

Arguments

YES

Saves all data files at the end of a run. This option *does not delete* any of the binary-generated files created during the execution of a Dracula job. This is the default value.

NO

Deletes all the binary-generated files at the very end of the Dracula job. *Do not use* this option if you are planning to use the Dracula Interactive tool to display information or errors on your data.

SMART

Deletes a binary-generated file the moment it is no longer needed during the execution of a Dracula job. This option helps to use disk space as efficiently as possible. *Do not use* this option if you are planning to use the Dracula Interactive tool to display information or errors on your data.

DRACULAINTERACTIVE

Saves all the binary-generated files that are needed by Dracula Interactive in order to display information or errors on your data. Like the SMART option, this option also helps to use disk space as efficiently as possible during the execution of a Dracula job.

Note: `KEEPDATA=DRACULAINTERACTIVE` works only for Dracula 4.8 users.
`KEEPDATA=INQUERY` works for Dracula 4.7 (and lower than version 4.7) users.

Example

```
KEEPDATA = SMART
```

KEEP-INST-FPIN

KEEP-INST-FPIN = YES/NO

Description

Keeps floating nets that are connected to cell instances, and also keeps floating Hcell pins in composite jobs. You can use this command if you have a schematic with both connected and unconnected pins and, in your layout, all the pins are not connected at the composite level. The KEEP-INST-FPIN = YES command does not discard these floating pins because the corresponding schematic HCELL pins are designated as floating pins.

See also Chapter 5 of this manual that contains the [KFPIN](#) section, which is the schematic counterpart of this KEEP-INST-FPIN command.

Arguments

YES

Does not filter out floating nets and floating Hcell pins.

NO

Is the default. Filters out floating nets and floating Hcell pins.

Example

KEEP-INST-FPIN = YES

KEEP-SHORT-BJT

KEEP-SHORT-BJT = YES/NO

Description

Causes LVS to keep BJT transistors that have terminals connected to the same node. By default, LVS automatically removes such transistors.

Arguments

YES

Does not remove the transistors.

NO

Removes the transistors. Default.

KEEP-SHORT-CAP

KEEP-SHORT-CAP = YES/NO

Description

Causes LVS to keep capacitors that have terminals connected to the same node. By default, LVS automatically removes such capacitors.

Arguments

YES

Does not remove the capacitors.

NO

Removes the capacitors. Default.

KEEP-SHORT-DIO

KEEP-SHORT-DIO = YES/NO

Description

Causes LVS to keep diodes that have terminals connected to the same node. By default, LVS automatically removes such diodes.

Arguments

YES

Does not remove the diodes.

NO

Removes the diodes. Default.

KEEP-SHORT-MOS

KEEP-SHORT-MOS = YES/NO

Description

Causes LVS to keep transistors that have terminals connected to the same node. By default, LVS automatically removes such transistors.

Arguments

YES

Does not remove the transistors.

NO

Removes the transistors. Default.

KEEP-SHORT-RES

KEEP-SHORT-RES = YES/NO

Description

Causes LVS to keep resistors whose terminals are connected to the same node. By default, LVS removes such resistors automatically.

Arguments

YES

Does not remove the resistors.

NO

Removes the resistors. Default.

KEEP-TEXT

KEEP-TEXT = YES/NO

Description

Output the text to all DRC output files and GDS output files

Arguments

YES

Outputs the text to DRC output files

NO

Is the default

LAYER-FILE

LAYER-FILE = *filepath*

Description

Specifies the path to the directory containing the layer file. This command is mandatory for the Edge database.

Arguments

filepath

The path to the directory containing the layer file.

Example

LAYER-FILE = /drac30/test/layer

LAYER-FILE-TYPE

LAYER-FILE-TYPE = *type*

Description

Specifies the type of layer file. Use this command with the LAYER-FILE command for the Edge database.

LAYER-FILE and LAYER-FILE-TYPE add *current* to the path. The actual layer file is */.../.../layer/display/current* (for example, */drac30/test/layer/display/current*).

Arguments

type

Specifies the actual representation of the layer file. The default type is *display*.

Example

LAYER-FILE-TYPE = display

LEGAL-CHAR-SET

LEGAL-CHAR-SET = *character*

Description

The default characters include A-Z, a-z, 0-9, +, -, *, /, (,), #, ., <, >, _, ', [,], \$, %, &, =, @. If you specify a LEGAL-CHAR-SET command, Dracula also considers those characters defined in that LEGAL-CHAR-SET command as legal.

Example

For Dracula to recognize the global signal name vdd!, gnd!, specify the special character '!' as

```
LEGAL-CHAR-SET = !
```

Dracula Reference

Description Block Commands

LIBRARY

`LIBRARY = library-name`

Description

Specifies the name of the Design Framework II library containing the primary cell in your design. You must include this command in the Description block if you include the command `SYSTEM = CADENCE`.

Arguments

library-name

The name of the Design Framework II library. Do not include the path name.

Example

```
SYSTEM = CADENCE
LIBRARY = ClassLib
```

LIMIT-DRC-ERROR

LIMIT-DRC-ERROR = YES/NO/{OUTPUT} number

Description

Restricts the number of DRC errors reported to a graphic or DRC output file. Use this command to conserve disk space. It works on flat, cell, composite, and hierarchical runs of DRC and HDRC.

Arguments

YES

Outputs the first 100 DRC errors.

NO

Outputs an unlimited number of DRC errors. Default.

OUTPUT

If you turn this option *on*, then it only gives limitation for the OUTPUT polygon files while the trapezoid output files still contain all the errors.

number

If you turn this option *on*, it restricts the number of all DRC errors to be reported and outputs only a specific number of error violations. You cannot specify a number with more than nine digits.

Examples

LIMIT-DRC-ERROR = YES

LIMIT-DRC-ERROR = NO

LIMIT-DRC-ERROR = 10000

LIMIT-RUN-TO

```
LIMIT-RUN-TO = ALL {output-error-list-file} {output-modified-rule file}/error-  
cell-list-file
```

Description

PDRACULA finds all the error cells checked in the rule file and skips all rules when error cell checking did not occur. It writes the cell list to a file for further modification and generates a revised rule file for reference. Default file is ERRCEL.LIS under the current working directory. Default is NEWRULES in the current working directory. Each line represents one cell to be checked. It might be a subset of the file generated by the command option ALL.

Note: This LIMIT-RUN-TO command is *only* valid for a DRC run as it takes into consideration the syntax directly associated with “output” functions when generating the NEWRULES file. You *cannot use* this command for other verification run checks such as an LVS run check because if you use this command in an LVS run, some SELECT commands are commented out that have outputs that you *need* later in the LVS run.

Arguments

ALL

All rules.

output-error-list-file

File containing all error cell names.

output-modified-rule-file

File containing rule deck run.

error-cell-list-file

File containing the error cell list.

Example 1

```
*OPERATION  
AND POLY PDIFF PGATE  
NOT NDIFF NGATE NSD           ; rule(1) : unused  
NOT PDIFF PGATE PSD OUTPUT PSD 34  
AND CONT POLY POLYCON         ; rule(2) : unused  
;  
WIDTH METAL LT 3 OUTPUT XMET 40
```

Dracula Reference

Description Block Commands

```
SIZE[T] METAL BY 1 OUT MET1 1
*END
```

Example 2

```
LIMIT-RUN-TO = ALL
```

To find error cells in the rule file, skip rule (1) and rule (2), write the cell list to `./ERRCEL.LIS`. The content of `ERRCEL.LIS` is

```
PSD34
XMET40
MET101
```

The content of the operation block of modified file `NEWRULES` is

```
*OPERATION
AND POLY PDIFF PGATE
;NOT NDIFF NGATE NSD
NOT PDIFF PGATE PSD OUTPUT PSD 34
;AND CONT POLY POLYCON
WIDTH METAL LT 3 OUTPUT XMET 40
SIZE[T] METAL BY 1 OUT MET1 1
*END
```

Example 3

```
LIMIT-RUN-TO = ALL /tmp/celllist
```

Same as example 2 except the output cell list is `/tmp/celllist`.

Example 4

```
LIMIT-RUN-TO = ALL /tmp/celllist NEW_RULES
```

Same as example 3 and all used rules would be stored in file `NEW_RULES`.

Example 5

```
LIMIT-RUN-TO = ERRCEL.LIS
```

Read cells from file `./ERRCEL.LIS`.

Example 6

```
LIMIT-RUN-TO = tmp/celllist
```

Dracula Reference

Description Block Commands

Read cells from file */tmp/celllist*.

LISTERROR

LISTERROR = *number*/YES/NO [*PERCELL*]

Description

Turns on the extended execution error listing for DRC or ERC runs. For runs on large ICs, this output can become large and unnecessary if your primary error investigation is graphic. The detailed listing of each DRC edge-pair violation and ERC trapezoid error output is in the log file or listing file. This command can also control the number of errors output for the antenna check. For more information, see the [CHKPAR](#) section in Chapter 13 of this manual.

For DRC edge-pair error violations, LISTERROR generates a detailed listing for the ENC, EXT, INT, WIDTH, and LENGTH commands. The listing is in the SPACING module. LISTERROR also generates a *.drc* file that lists the violation pairs.

The x,y coordinates in the log file or *.drc* file are relative to the origin of the cell name/structure name you specify with the PRIMARY command. The x,y coordinates in the *.sum* printfile under “Problem Geometries” are relative to the origin of the cell name/structure name containing the error shapes.

For ERC trapezoid error output, see the ERCSEL module or search for the error cell name (for example, *drc401*, *float60*, or *shorts40*).

Arguments

number

Prints the *number* of error violations you specify. Limit for *number* is four digits.

YES

Prints the first 100 error violations.

NO

No printing. Default.

PERCELL

Specifies the number of errors to be reported per cell. This option is not used in antenna checks.

Examples

LISTERROR = NO

Dracula Reference

Description Block Commands

LISTERROR = YES
LISTERROR = 1000
LISTERROR = 50 PERCELL

LONG-SUM-LEN

LONG-SUM-LEN = YES/NO

Description

Increases the line length of summary file, * .sum.

Arguments

YES

Prints 130 characters per line for .sum file.

NO

Default.

LOOP-TWO-CAP

LOOP-TWO-CAP = SERIES/PARALLEL

Description

Treats a two-device structure with connected terminals as either a series or parallel device after smashing. Dracula treats a loop of two capacitors as series/parallel capacitors. This means

node1 -> dev1 -> node2 -> dev2 - node1

such that

- No other connection is on *node2*.
- *dev1* and *dev2* are the same type of capacitor devices.

To combine a loop of two capacitors to a single capacitor

1. Do not turn on LVCHK [K].
2. Invoke the command LOOP-TWO-CAP = PARALLEL.

Arguments

SERIES

Treats the two-device structure as a series device after smashing. This is the default behavior.

PARALLEL

Treats the two-device structure as a parallel device after smashing.

Example

LOOP-TWO-CAP = SERIES

LOOP-TWO-RES

LOOP-TWO-RES = SERIES/PARALLEL

Description

Treats a two-device structure with connected terminals as either a series or parallel device after smashing. Dracula treats a loop of two resistors as series/parallel resistors. This means

node1 -> dev1 -> node2 -> dev2 - node1

such that

- No other connection is on *node2*.
- *dev1* and *dev2* are the same type of resistor devices.

To combine a loop of two resistors to a single resistor

1. Do not turn on LVCHK [K].
2. Invoke the command LOOP-TWO-RES = PARALLEL.

Arguments

SERIES

Treats the two-device structure as a series device after smashing. This is the default behavior.

PARALLEL

Treats the two-device structure as a parallel device after smashing.

Example

LOOP-TWO-RES = PARALLEL

LPE-FORMAT

LPE-FORMAT = LOGLVS

Description

For LPE/PRE output netlist, Dracula follow the convention of HSPICE as default. But adding this command in the Dracula rule file can switch to the format of LOGLVS(CDL).

Arguments

LOGLVS

Prints the LPE/PRE output netlist in CDL format.

LPE-GROUND

`LPE-GROUND = ground_name`

Description

Lets you specify a unique ground name in multiple-ground cases if you want only one of them as ground node in the output SPICE file. It works on parasitic capacitance only. This LPE-GROUND command overwrites the LPESELECT[G] or SPFSELECT[G] commands. This command is case insensitive.

If you do not specify the preferred ground node name, then Dracula picks up the first ground node in the SPICE file as the lumped capacitance's ground.

Arguments

ground_name

Is a user-specified text.

LPE-PRECISION

LPE-PRECISION = n1

Description

Dracula outputs SPICE file with process precision to decimal point of n1 for device length and width. This command is only valid for layout only LPE job.

Arguments

n1

Precision number

LPE-QUERY

LPE-QUERY = CORE/EXPAND/ON {*filename*}

Description

Runs selective LPE for flat and composite modes.

Include this command in the DESCRIPTION block if you want LPE to extract the layout parasitics and devices associated with a particular path.

To run selective LPE, you must use two other commands:

- SELECT-MODE in the Description block
- Either NODE-SELECT or NODE-FILE in the Operation block.

For details about these commands, refer to the Description and Operation block chapters.

Arguments

CORE/ON

Creates a final selected netlist that includes the user-specified nodes, the devices connected to the user-specified nodes, and the adjacent nodes connected to the terminals of the selected devices.

EXPAND

Creates a final selected netlist that includes both nets and devices reached from the nodes you specify, all nets and devices on traversed paths, and nets connected to selected devices that are not on the traversed paths.

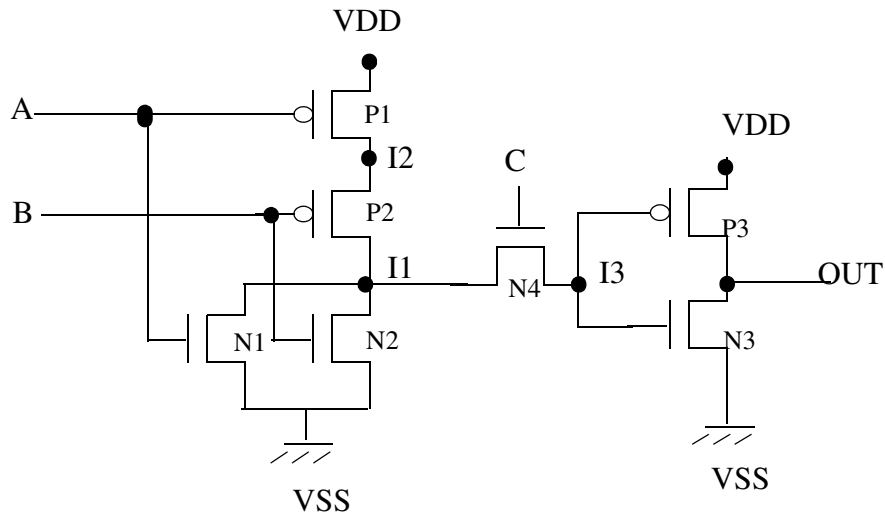
Dracula creates the traversed path by starting from the user-specified nodes and passing the MOS/LDD source-drain connection, the BJT collector-emitter connection, the resistors, capacitors, and diodes. Dracula ends the traversed path on the input/output pins or the power/ground nodes and the gate terminal of MOS or the base terminal of BJT.

filename

Writes selected node numbers and device numbers to filename. Other information is output to the log file. To prevent selected node numbers and device numbers from being output, specify the filename NoDump. If you do not specify a filename, selected node and device information is written to the log file.

Examples

In the following example, the user-specified node set is $\{I1\}$.



When LPE-QUERY = CORE

- Adjacent nodes = $\{I2, A, B, C, I3, VSS\}$
- Selected devices = $\{P2, N1, N2, N4\}$

When LPE-QUERY= EXPAND

- Nodes on the traversed path (expanded from the user-defined nodes)
= $\{I1, I2, I3\}$
- Adjacent nodes = $\{VDD, A, B, VSS, C, OUT\}$
- Selected devices = $\{P1, P2, N1, N2, P3, N3, N4\}$

The rule file for the LPE-QUERY example above might be the following:

```
*DESCRIPTION
LPE-QUERY = EXPAND
SELECT-MODE = SCH
...
*END
*OPERATION
NODE-SELECT = I1
...
*END
```


LUMPCAP

```
LUMPCAP = YES/NO/{subtype1,factor1 {subtype2,factor2} {subtype3,factor3}...}/  
          {ALL,factor}
```

Description

Lets you turn on lumped mode parasitic capacitance extraction for LPE and PRE jobs. By default (LUMPCAP=NO), Dracula stores parasitic capacitance as coupling in its internal binary file. You can use different reporting formats (coupling or lumped) when you output the capacitance in SPICE format with LISTLPE commands.

With the lumped mode option turned on, Dracula sums the capacitance of each node for every subtype and stores them as lumped (node to ground) in its internal binary files. You can use lumped format in LPESELECT commands only.

If you specify lumped mode, the parasitic capacitance binary parameter files occupy less disk space and total CPU time is reduced.

If you specify LUMPCAP=YES, you cannot specify the LPESELECT options *C*, *T*, *P*, and *R*. Dracula issues an error message if you specify these options.

You can also specify different multiplying factors for individual capacitance subtypes when lumping them to ground. This capability of Dracula can help achieve more accurate simulation.

Arguments

YES

Turns on LUMPCAP mode. All subtype multiplying factors are 1.0.

NO

Turns off lumped mode capacitance and uses the regular flow. Default.

subtype1,factor1 {subtype2,factor2}...

Uses different multiplying factors for their corresponding subtypes of capacitance. Uses the default multiple factor 1.0 for the other capacitance.

ALL, factor

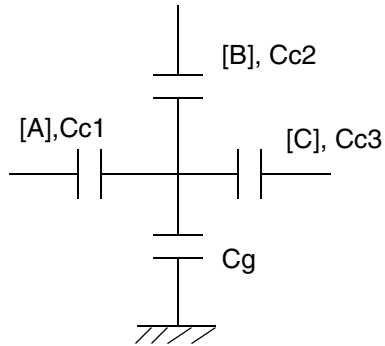
Uses this factor for all capacitances.

Dracula Reference

Description Block Commands

Example 1

LUMPCAP = A,0.2 B,1.0 C,1.5



$$C_t = C_g + 0.2 * C_{c1} + 1.0 * C_{c2} + 1.5 * C_{c3}$$

Dracula multiplies different subtype lump capacitances with their corresponding factors. If you don't specify the factor for some capacitance subtypes, the default factor value is 1.0.

Example 2

LUMPCAP = ALL,2.0

The multiplying factor for all capacitances is 2.0.

Dracula Reference

Description Block Commands

LVL-PART

```
LVL-PART=SCHEMATIC
LVL-PART=GRAPHICS
```

Description

Alerts the preprocessor that you want to perform a layout versus layout (LVL) discrepancy check. PDRACULA processes the reference layout rules file first, then processes the target layout you want to check.

You can run the complete LVL job without breakpoints if you use the /NEXT command to combine the two LVL input rules files. You can also run two separate jobs. In either case, LVL-PART specifies the reference layout and the layout to check.

Examples

<pre>;LVL1.COM: *Description . . INDISK= layout1.db PRINTFILE=LVL1PRT LVL-PART=SCHEMATIC . . *END *INPUT-LAYER . . *END *OPERATION . . LVSCHK *END</pre>	<pre>LVL2.COM: *Description . . INDISK=layout2.db PRINTFILE=LVL2PRT LVL-PART=GRAPHIC . . *END *INPUT-LAYER . . *END *OPERATION . . LVSCHK</pre>

To submit an LVL job, enter the following commands:

```
/GET LVL1.COM
/NEXT
/GET LVL2.COM
/FIN
```

LVL runs the *lv1.com* file, then the *lv2.com* file. The discrepancy report is named *lv2prt.lvl*.

LVS-OPTION

LVS-OPTION = {*options*}

Description

Specifies the LVS error types to report.

Options

You can choose from the following options:

A	Report LVS error type 1.
B	Report LVS error type 2.
C	Report LVS error type 3.
D	Report LVS error type 4.
E	Report LVS error type 5.
F	Report LVS error type 6.
G	Report LVS error type 7.
H	Report LVS error type 8.
I	Report LVS error type 9.
J	Report LVS error type 10.
K	Report LVS error type 11.
L	Report LVS error type 12.
M	Report LVS error type 13.

For an explanation of LVS error types, refer to [Chapter 5, “Compiling Network Descriptions \(LOGLVS\)”](#).

If you do not specify an LVS-OPTION command, Dracula reports all errors. For a shorter LVS report, use LVS-OPTION. If after the initial run you want a different LVS report, use the LVS-OPTION command with the LVSRPT-ONLY command to generate and rerun the job from the LVSCHEM module only.

The PRINTLINE and LISTUNMATCH qualifiers in the LVSCHEM command still control the amount of output.

Dracula Reference

Description Block Commands

Example

The following example reports only matched device to unmatched node, unmatched schematic device, and unmatched layout device errors.

```
LVS-OPTION = BGH
```

LVS RPT-ONLY

LVS RPT-ONLY = YES/NO

Description

Generates a new LVS report from a previous LVS run. This command assumes that you saved files from a previous LVS run in the directory and want a different LVS report. Use this command with the LVS-OPTION command to generate the new report. Only the LV SCHM step is needed in the run.

Arguments

YES

Generates a new LVS report from a previous LVS run.

NO

Default.

Example

The following example reports only subtype mismatch and size mismatch errors. Dracula does not run the matching portion of LVS. If LVS RPT-ONLY is not present, Dracula runs both the matching and reporting portions of LVS, resulting in longer run times.

```
LVS RPT-ONLY = YES
LVS-OPTION = IJ
```

M-Factor Commands

MOS-M-FACTOR = WIDTH-LENGTH/NO

BJT-M-FACTOR = WIDTH-LENGTH/EMITTER-AREA/NO

RES-M-FACTOR = WIDTH-LENGTH/VALUE/NO

DIO-M-FACTOR = AREA/PERI/NO

CAP-M-FACTOR = AREA/VALUE/NO

Description

Evaluates the M-Factor tolerance commands you specified with LVCHK or LPECHK. When you use M-Factor commands, for each group of parallel devices, Dracula checks the tolerances you specified and determines whether to merge the group in the circuit reduction phase.

The M-Factor commands tell Dracula to enable or disable M-Factor comparisons in the circuit matching phase. If you use an LVCHK or LPECHK command with the *K* option ON and the *S* option OFF, Dracula ignores the M-Factor options.

To specify device M-Factors in a SPICE netlist, follow the SPICE/HSPICE conventions for specifying M-Factors. By default, Dracula recognizes schematic and layout devices without M-Factors as M-Factor = 1.

Arguments

WIDTH-LENGTH

Specifies to use device width and length tolerances to evaluate whether to merge the device. Dracula evaluates the tolerances you specify using the M-Factor Tolerance commands and does one of the following:

- Merges parallel devices if the actual tolerances are less than the value you specify
- Does not merge parallel devices if the actual tolerances are greater than the tolerance you specify

EMITTER-AREA

Specifies to use BJT device tolerances for emitter area to evaluate whether to merge a device.

Dracula Reference

Description Block Commands

Dracula evaluates the tolerances you specified using the M-Factor Tolerance commands and does one of the following:

- Merges parallel devices if the actual tolerances are less than the value you specify
- Does not merge parallel devices if the actual tolerances are greater than the tolerance you specify

NO

Default.

AREA

Specifies to use DIO or CAP device tolerances for device area to evaluate whether to merge a device.

PERI

Specifies to use DIO device tolerances for periphery to evaluate whether to merge a device.

VALUE

Specifies to use RES or CAP device tolerances for value to evaluate whether to merge a device.

Examples

```
MOS-M-FACTOR = WIDTH-LENGTH
BJT-M-FACTOR = EMITTER-AREA
```


M-Factor Tolerance Commands

MOS-W-TOLER = $n1$
MOS-L-TOLER = $n2$
BJT-EA-TOLER = $n3$
BJT-W-TOLER = $n4$
BJT-L-TOLER = $n5$
RES-W-TOLER = $n1$
RES-L-TOLER = $n2$
RES-V-TOLER = $n3$
DIO-A-TOLER = $n4$
DIO-P-TOLER = $n5$
CAP-A-TOLER = $n6$
CAP-V-TOLER = $n7$

Description

Indicates the number of separate devices, such as MOS or BJT devices, connected in parallel. You specify this number in a CDL netlist. If a MOS device contains four transistors connected in parallel, the multiplier you specify in your CDL netlist is $M = 4$.

You can use these M-Factor Tolerance commands with an LVCHK or LPECHK command to specify tolerances for Dracula to use to determine whether a group of parallel devices reduces to a single device. If the group of n devices is within the tolerance value you specify using the M-Factor Tolerance commands, Dracula treats these devices as a single device with M-Factor = n . If the group of devices is greater than the tolerances you specify using M-Factor Tolerance commands, Dracula verifies them separately.

When you use the M-Factor Tolerance commands, for every group of parallel devices, Dracula checks the tolerances you specify and decides whether to smash the device group in the circuit reduction phase.

The M-Factor commands tell Dracula to enable or disable M-factor comparisons in the circuit matching phase. If you use an LVCHK or LPECHK command with the K option ON and the S option OFF, Dracula ignores the M-Factor options.

To specify device M-Factors in a SPICE netlist, follow the SPICE/HSPICE conventions for specifying M-Factors. By default, Dracula recognizes schematic and layout devices without M-Factors as M-Factor = 1.

The M-Factor commands are as follows:

Dracula Reference

Description Block Commands

MOS-W-TOLER = *n1*

A MOS width tolerance you specify. This tolerance only works when you specify
MOS-M-FACTOR = WIDTH-LENGTH.

MOS-L-TOLER = *n2*

A MOS length tolerance you specify. This tolerance only works when you specify
MOS-M-FACTOR = WIDTH-LENGTH.

BJT-EA-TOLER = *n3*

A BJT emitter tolerance you specify. You must also specify BJT-M-FACTOR =
EMITTER-AREA.

BJT-W-TOLER = *n4*

A BJT width tolerance you specify. You must also specify BJT-M-FACTOR = WIDTH/
LENGTH.

BJT-L-TOLER = *n5*

A BJT length tolerance you specify. You must also specify BJT-M-FACTOR = WIDTH/
LENGTH.

RES-W-TOLER = *n1*

A RES width tolerance you specify. This tolerance only works when you specify RES-M-
FACTOR = WIDTH-LENGTH.

RES-L-TOLER = *n2*

A RES length tolerance you specify. This tolerance only works when you specify RES-
M-FACTOR = WIDTH-LENGTH.

RES-V-TOLER = *n3*

A RES value tolerance you specify. This tolerance only works when you specify RES-M-
FACTOR = VALUE.

DIO-A-TOLER = *n4*

A DIO area tolerance you specify. This tolerance only works when you specify DIO-M-
FACTOR = AREA.

DIO-P-TOLER = *n5*

A DIO periphery tolerance you specify. This tolerance only works when you specify DIO-
M-FACTOR = PERI.

Dracula Reference

Description Block Commands

CAP-A-TOLER = *n6*

A CAP area tolerance you specify. This tolerance only works when you specify CAP-M-FACTOR = AREA.

CAP-V-TOLER = *n7*

A CAP value tolerance you specify. This tolerance only works when you specify CAP-M-FACTOR = VALUE.

Examples

MOS-W-TOLER= 0.1

MOS-L-TOLER= 0.05

BJT-EA-TOLER= 0.2

BJT-W-TOLER= 0.01

BJT-L-TOLER= 0

Default Tolerance Values

Dracula assigns a default M-Factor tolerance value when you do not specify an M-Factor tolerance in the run file. Two default values, *tight-tolerance* and *loose-tolerance*, are associated with each tolerance. *Tight-tolerance* means the values are required to be almost the same. *Loose-tolerance* means there are no constraints at all.

	Tight tolerance	Loose tolerance
MOS-W-TOLER	0.1 unit	Infinity
MOS-L-TOLER	0.1 unit	Infinity
BJT-EA-TOLER	0.01 square unit	Infinity
BJT-W-TOLER	0.1 unit	Infinity
BJT-L-TOLER	0.1 unit	Infinity

You use the different default values in different situations, depending on whether *MOS-M-Factor* or *BJT-M-Factor* is specified in the run file.

MAG-BEF-GRID

MAG-BEF-GRID = YES/NO

Description

Switches the order of on-grid data placement and cell magnification. This command works with the RESOLUTION command. The default order of inputting layout data is as follows:

1. Place the input geometry on grid.
2. Magnify the input geometry according to the absolute magnification (as defined in the layout).
3. Place magnified geometry on grid.

Arguments

YES

Skips step 1, magnifying the input geometry before placing it on grid (scaled cell).

NO

Default.

Example

This example skips step 1. This process is also known as “scaled cell.”

MAG-BEF-GRID = YES

MAGNIFY-IN

MAGNIFY-IN = *factor*

Description

Scales the data in a GDSII input database by the scale factor you specify. CONVERT-DATABASE must be present and set to YES.

Arguments

factor

Specifies the scale factor. This floating-point number can have up to six digits after the decimal point.

Example

This example magnifies the data in the input GDSII database by 5 percent.

```
CONVERT-DATABASE = YES  
MAGNIFY-IN = 1.05
```

MAGNIFY-OUT

MAGNIFY-OUT = *factor*

Description

Scales the data in a GDSII output database by the scale factor you specify. CONVERT-DATABASE must be present and set to YES.

Arguments

factor

Specifies the scale factor. This floating-point number can have up to six digits after the decimal point.

Example

This example shrinks the data in the output GDSII database by 5 percent.

```
CONVERT-DATABASE = YES  
MAGNIFY-OUT = .95
```

MAP-TEXT-FILE

MAP-TEXT-FILE = text.map

Description

Lets you map multiple text names to one name. For example, if you have multiple names on the same signal, you can use MAP-TEXT-FILE to make all the names equivalent. All names refer to the same signal without causing a “MULTILAB” or requiring re-texting of the layout.

You can map multiple names to one name on one line. For example, instead of

```
VDD1    VDD
VDD2    VDD
VDD3    VDD
```

you can specify

```
VDD1 VDD2 VDD3 = VDD
```

You can also use wildcards in mapping. Only the question mark (?) and asterisk (*) are supported. A question mark (?) means a single occurrence of any character. An asterisk(*) means any number of occurrences of any character. For example,

```
VDD* = VDD
VDD? = VDD
```

Arguments

text.map

Name of text mapping file.

Example

```
*description
MAP-TEXT-FILE= text.map
...
*end
```

text.map contains sets of equivalent texts:

```
-----
text1 text2
text3 text2
text4 text5
...
-----
```

MATCH-P-G-BY-TYP

MATCH-P-G-BY-TYP = YES/NO

Description

Forces Dracula to match power and ground nodes by name. By default, Dracula matches power and ground pads by type regardless of their names. This command turns off the default behavior.

Arguments

YES

Matches power and ground pads by type. Default.

NO

Matches power and ground pads by name.

Example

MATCH-P-G-BY-TYP = YES

MEMORY-CORE

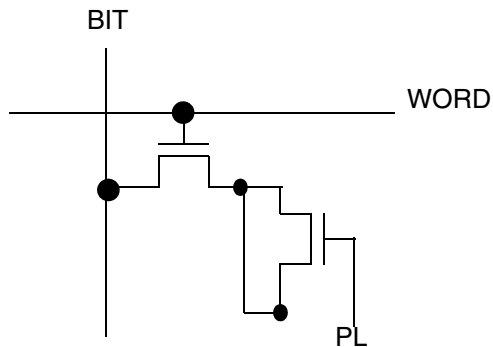
MEMORY-CORE = DRAM/SRAM

Description

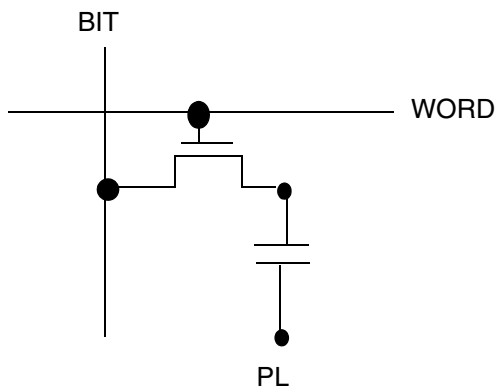
This command enables Dracula to recognize the most popular structure of memory core and can reduce it into a single device. Since the net itself is dramatically reduced, the performance of LVS is also improved.

The recognition of the DRAM or SRAM structures is limited to the following figures:

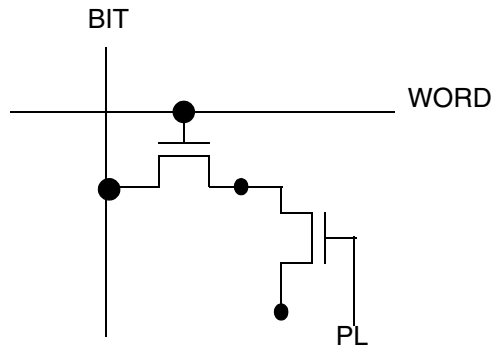
DRAM Cell (A)



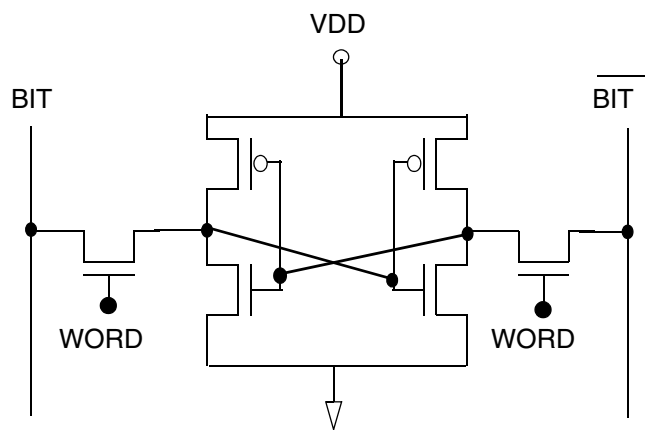
DRAM Cell (B)



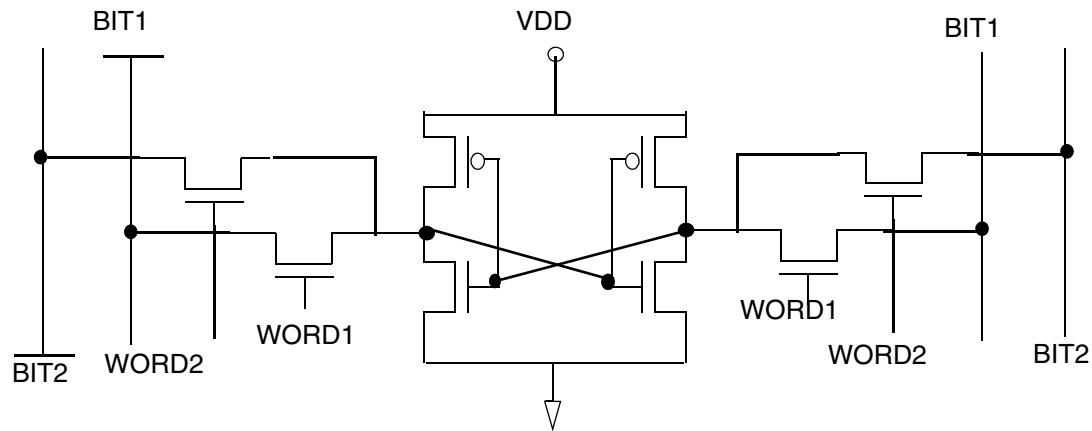
DRAM Cell(C)



SRAM Cell (Six-transistor full CMOSs)



SRAM cell (Dual Port)



Arguments

DRAM

Recognizes DRAM cells

SRAM

Recognizes SRAM cells

Example

```
MEMORY-CORE = SRAM
```

Note: The subtype of MOSs in the SRAM structures must be `P-type` or `N-type`.

Dracula Reference

Description Block Commands

MINI-SUMMARY

```
MINI-SUMMARY = YES {pattern1 pattern2 pattern3 ...}
```

Description

Greps data from *<printfile>.log* and *<printfile>.msm*. If you specify the command MINI-SUMMARY = YES, you get, by default, in the *.log* file:

Symbol	Indicates
/F	Fatal errors
/E	Errors
/W	Warning lines

To find other lines, you must specify them in the mini-summary line. “*/N*” stands for notational output messages. “*/L*” stands for listing output messages.

Example

```
MINI-SUMMARY = YES
MINI-SUMMARY = YES */N* */L*
```

MIRROR

`MIRROR = option`

Description

Flips a layout around the x or y axis.

Note: This command can be used in flat mode only.

Options

x

Mirrors along the horizontal or x axis.

y

Mirrors along the vertical or y axis.

Examples

The following example mirrors the layout database along the y axis.

```
MIRROR = Y
```

You can also use WINDOW and WINDEL commands with MIRROR. The MIRROR command affects the coordinates of the WINDEL and WINDOW commands in the same way.

```
WINDOW = 1000. 500. 2000. 2500.  
WINDEL = 1500. 1500. 1800. 2000.  
ROTATION = 90  
MIRROR = Y
```

MODEL

`MODEL = (layout-element, model-name...)`

Description

Associates the MOS, BJT, and diode circuit elements with their corresponding circuit simulation model names. These model names appear in the SPICE-format listing generated by LPE.

You must specify this command for an LPE run. If you do not specify this command, LPE comments out all device references in the SPICE listing. You can repeat this command as needed.

Arguments

layout-element

Specifies the user-defined layout element; that is, a MOS transistor (MOS), a diode (DIO), or a bipolar transistor (BJT).

model-name

Specifies the model name of the layout element in the circuit simulation (SPICE file).

Examples

```
MODEL=MOS[N],NCHNL...DIODE[N],NDIO...
```

```
MODEL=BJT[pn],pnp
```

MOS-CAP

MOS-CAP = P-G/NODE

Description

Recognizes MOS capacitors as MOSCAP devices. Dracula recognizes a transistor as MOSCAP when it meets one of the following conditions:

- (1) Source and drain are connected to the same power/ground node.
- (2) Source and drain are connect to the same node which is not limited to the power/ground node.

Arguments

P-G

MOS transistors with source and drain tied together to either power or ground.

NODE

MOS transistors with source and drain tied together to the same node.

Example

MOS-CAP = P-G

MOS-M-COMPARE

MOS-M-COMPARE = YES/NO/SCH-M-SET

Description

Enables or disables M-factor comparison during LVSCHM.

Arguments

YES

Default. Will compare the M-factor for all MOS devices, including width and length.

NO

Will not compare the M-factor, but will compare width and length.

SCH-M-SET

Will compare the M-factor, including width and length, only when M-factor is set in the CDL.

Example

MOS-M-COMPARE = SCH-M-SET

MULTICPU

MULTICPU = CPU_NUMBER

Description

Allows you to use Dracula distributed processing in a multi CPU machine. This command is similar to the `?multiCPU` command in the Assura product and you use this command for starting the Dracula distributed processing in a multiple CPU workstation.

Note: The MULTICPU command cannot be used in combination with the PARALLEL-FILE command. The MULTICPU command does not need the parallel file. If it does, PDRACULA issues an error message.

When you specify the MULTICPU command in the Description block section of your rules file, PDRACULA creates the parallel file automatically.

Arguments

CPU_Number

Specifies the number of CPUs. PDRACULA turns on the Dracula distributed processing. The Dracula product uses CPUs of this workstation for Dracula distributed processing.

Note: For using the MULTICPU command with Dracula distributed processing and the Dracula graphical user interface (the `draculaInteractive` tool), see the “Note” under the “KEEPDATA” command section in this chapter and the “Dracula Distributed Processing and the Dracula Graphical User Interface Tool” section in Chapter 2 of this manual.

MULTI-GDS2-IN

MULTI-GDS2-IN = filename CELL/LAYER

Description

Specifies the file containing multiple GDSII input files split by cells or layers. The 4.81 and 4.9 versions have a new feature, the layer shift option where if you need to deal with more than 2 input file layer cases, you should put a shift value in the input filename. This feature allows data to be read from multiple GDSII databases and still be maintained as separate layers even though the original data is on the same GDSII layer.

Arguments

filename

File containing GDSII input file names. Input files can come in any order. You can specify files with absolute or relative paths. Wild card is not supported.

CELL/LAYER

Indicates the GDSII files are split by cells or layers. CNames-CSEN is automatically turned on when the MULTIPLE-GDS2-IN command is used.

Examples

Example 1

MULTI-GDS2-IN = inFile CELL

The contents of inFile could be

```
GDSIIFile1.gds
/data/GDSIIFile2.gds
/usr1/Dracula/data/GDSIIFile3.gds
```

Example 2

MULTI-GDS2-IN = inFile LAYER

You can add a shift value in inFile as follows:

```
inFile:
  gdsin1.gds SHIFT 10
  gdsin2.gds
  gdsin3.gds SHIFT 20
```

Dracula Reference

Description Block Commands

You shift all layers in `gdsin1.gds` by 10, `gdsin3.gds` by 20, and there is no shift for `gdsin2.gds`. Data is merged if there is a layer number collision after the shift. You can provide a rule file for layers you need to manipulate with layer assignment based on new layer numbers.

Example 3

Given below is an example with two GDSII files in the following configuration. This example clearly tells you that Dracula can separately manipulate the layers specified in the `inFile`, if you have properly set the needed layer in the INPUT LAYER of the rule file.

Two GDSII files: `rev1.gds` `rev2.gds`. The original layer assignments for this example are as follows:

```
NWELL = 1
ACTIVE = 5
POLY = 10
M1 = 15
V1 = 20
```

If you want

- a. To edit the *POLY* layer of the design in `rev1.gds` and create `rev2.gds`.
- b. Run PDRACULA (using *POLY* of `rev2.gds` and all other layers of `rev1.gds`).

then use the following syntax:

In the rule file:

```
MULTI-GDS2-IN = inFile LAYER
NWELL = 1
ACTIVE = 5
POLY = 30
M1 = 15
V1 = 20
```

In `inFile`:

```
rev1.gds
rev2.gds shift 20
```

Dracula will get all layers from `rev1.gds` unchanged, such as NWELL, ACTIVE, M1, and V1; and the POLY layer from `rev2.gds`, which involves shifting the layer number by 20.

Then in the rule file, set *POLY* to layer 30 (original (10) + 20) and leave all other layers unchanged.

MULTI-PIN-FILE

MULTI-PIN-FILE = *filename*

Description

Specifies the output file name for the pad coordinates reported in PRE flatten mode. You can choose the option of not shorting multiple pins into one subnode and reporting all multiple pins in the .SUBCKT header of the SPICE file.

PRE generates and reports multiple sub-pin names to the .SUBCKT statement for feedthrough nodes and externally connected nodes. You must define how many pins to create with the PAD-LAYER command.

Note: Keep the following guidelines in mind when using the MULTI-PIN-FILE command:

- This command works only in flatten mode PRE.
- The FDELIMITER command has a higher priority than MULTI-PIN-FILE, which might cause conflicts.
- Because PAD-LAYER defines all pin locations, you must define the pins correctly in the PAD-LAYER command.
- CDL and EDTEXT are unchanged by the MULTI-PIN-FILE command.
- PRE does not shorten pins with the same layout node number. However, it assigns different pin names for these pins. (The original pin name is concatenated with a subnode delimiter and a subnode number.)
- The MULTI-PIN-FILE command generates an ASCII file named *mpfile*, which contains the pin name and its x,y coordinates. The context of this file is similar to that of the EDTEXT file.
- Multiple pin names are reported in the SPICE netlist (.SUBCKT statement), which is different from the original schematic netlist.

Arguments

filename

Output file name for multiple pin names and coordinates

Dracula Reference

Description Block Commands

Example

```
*DESCRIPTION
SYSTEM          = GDS2
PRIMARY         = mychip
INDISK          = mychip.gds
OUTDISK        = PRE.GOUT
SCALE           = .001 MIC
RESOLUTION      = .1 MIC
PRINTFILE       = PRE
SCHEMATIC       = LVSLOGIC
MODE            = EXEC NOW
MODEL           = MOS[N],N MOS[P],P DIO[N],DN DIO[P],DP
DIODESEQ        = A1,P1
UNIT            = CAPACITANCE,PF AREA,P PERIMETER,U    RESISTANCE,K
CNAME-SEN       = YES
PROGRAM-DIR     = /DRACULA/4.0.flhot/bin/
MULTI-PIN-FILE = PINS.FILE
*END
```

MULTIPLE-DISK

MULTIPLE-DISK = YES/NO

Description

Links different physical disks or disk partitions and uses them as an aggregate resource. The disks can be distributed over a network or be NFS mounted.

You create a text file (MDISKIO.TAB) that lists the mounting point and directory paths Dracula uses for linking disk space. One separate MDISKIO.TAB file must be created for each slave node. MDISKIO.TAB.1 is for the master node. MDISKIO.TAB.2 is for slave node 1. MDISKIO.TAB.3 is for slave node 2, and so on. Keep the MDISKIO.TAB file in the same directory as the *jxrun.com* file.

When you write your MDISKIO.TAB file, specify the mounting point for each disk to use in the first field. Specify the directory to hold data in the second field, and specify reserved disk space in the third field.

The first line in your file represents the first disk, the second line represents the second disk, and so on. The default for reserved space is 10 megabytes for the first disk. Designate numbers for the third, or reserve space field, in kilobytes. After the first disk or line, the default for reserve disk space is 0, so you must enter a value in kilobytes.

The MULTIPLE-DISK command has no control over generated text files. Text files are written to the first disk and can include the job log file, summary file, and generated SPICE/CDL/DSPF file. To prevent the job from aborting because text files have filled the file system, reserve a large space in the first disk.

Note: Keep the following guidelines in mind when using the MULTIPLE-DISK command.

- This command works for UNIX systems only.
- PDRACULA does not have a multiple disk capability.
- To enable multiple disk mode in LOGLVS, specify MDISKIO.TAB in the LOGLVS run directory.
- Do not specify the same disk as both a hard and soft mount.
- To add new disks to the MDISKIO.TAB and restart *jxrun.com* from the middle, add the additional mounting points and directory paths to the end of the MDISKIO.TAB. Do not insert additional mounting points and directory paths in the middle of the MDISKIO.TAB file.
- You cannot list the same disk in the MDISKIO.TAB file more than once.

Dracula Reference

Description Block Commands

- When you specify MULTIPLE-DISK = YES, Dracula run speed is slower.
- You must compile the rules file and specify MULTIPLE-DISK in the working directory of the first disk that appears in the MDISKIO.TAB file.

If you specify an absolute or relative path (any path containing a backslash “/”) with the OUTDISK command, MULTIPLE-DISK creates the output file under that path if it exists. If the path you specify does not exist or if you do not specify a path, the MULTIPLE-DISK command determines the location of the output file and shows the location of this file in the *.log* file.

Note: The maximum disk entries have been increased from 10 to 20.

Arguments

YES

Tells Dracula to use the MDISKIO.TAB file to link multiple disk directories. If you issue this command and a MDISKIO.TAB file does not reside in the directory where the *jxrun.com* file resides, PDRACULA issues a warning. Dracula automatically turns off the MULTIPLE-DISK command if the MDISKIO.TAB file is not found.

NO

Tells Dracula to not link multiple disk space. PDRACULA does not warn you if you specify MULTIPLE-DISK = NO in your run file and an MDISKIO.TAB file exists. Default.

Examples

The following example shows the mount point in the first field, followed by the working directory path:

```
/user1      /user1/blk/myname/LVS      ;default disk space (10 MB)
```

The following example shows the mount point, working directory path, and reserve space field. In this example, 5000 kilobytes of disk space is reserved.

```
/user2      /user2/blk/myname/LVS 5000      ;user-defined disk space  
                                           ;default disk space (0 MB)
```

The following example specifies 8 MB of reserved disk space where the default is 0.

```
/user3      /user3/blk/myname/LVS 8000      ;user-defined disk space  
                                           ;default disk space (0 MB)
```

NO-SUB-TERM

NO-SUB-TERM = *subType1* {, *subType2*}{....}

Description

Generates spice netlist without substrate nodes for those MOS devices of specified sub-types in this command. This command only works when the substrate node is not defined in a rule file of a layout only run or LVS run but, without reported schematic names (“S” option is not *on* in LPESELECT).

Arguments

subType

Subtype name of MOS devices as defined in the ELEMENT command.

Example

```
NO-SUB-TERM = N,P
.....
ELEMENT    MOS[P] PDEVICE PCBIAS PDIFDE ; no substrate node
ELEMENT    MOS[N] NDEVICE PCBIAS NDIFC
.....
LPESELECT MOS OUTPUT SPICE
```


NOT-HCELL

```
NOT-HCELL = cell-name1 {cell-name2} {...}
```

Description

Specifies the layout cells that are not to be selected as Hcells. For a detailed explanation of HDRC Hcell selection, refer to Chapter 3, "Checking Design Rules (DRC)."

Checking Method

```
checks  
    HDRC.
```

Arguments

```
cell-name  
    Layout cell name not to be considered an Hcell candidate.
```

Examples

```
NOT-HCELL = old  
NOT-HCELL = old,logo
```

NRX-EXTRACT

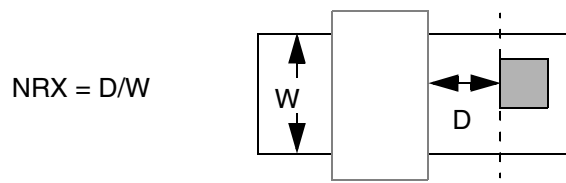
```
NRX-EXTRACT = MOS [subType] [accuracy = HIGH/QUICK/HIGH-PARALLEL]  
               distance {SHADE} {coefficient}
```



Don't use the same `subType` name for LDD NRX-EXTRACT as for MOS.

Description

Outputs the resistance value of MOS source and drain. Dracula extracts the source/drain resistance in terms of SPICE NRX parameters. The extracted resistance is a function of the overlap perimeter of the Source/Drain and Gate layers, and the distance to its closest contact.

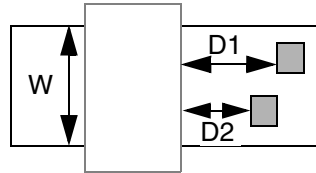


Dracula Reference

Description Block Commands

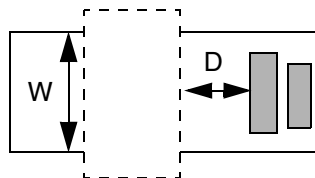
Each gate-edge segment contributes to the final NRX value. The closest contact determines the distance, as shown in the following examples.

$$\text{NRX} = D2/W$$



$$\text{NRX} = D/W$$

shading effect

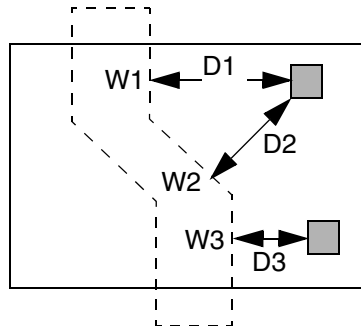


$$\text{NRX1} = D1/W1$$

$$\text{NRX2} = D2/W2$$

$$\text{NRX3} = D3/W3$$

$$\text{NRX} = (\text{NRX1} + \text{NRX2} + \text{NRX3})/3$$



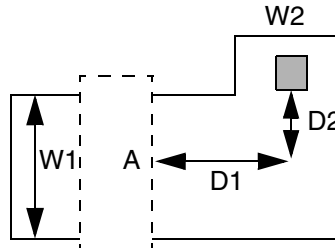
Dracula Reference

Description Block Commands

When you have more complex situations, for example, when contacts are not directly visible, or there is no contact, the distance is calculated as shown in the following examples.

$$NRX = D1/W1 + D2/W2$$

(where A is the center point of W1)

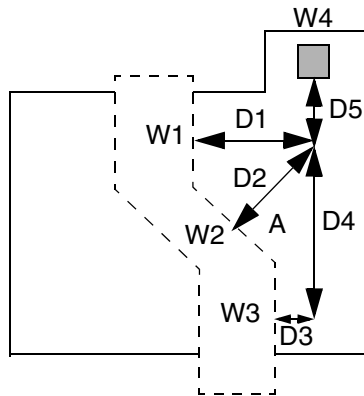


$$SUM1 = D3/W3 + D4/W4$$

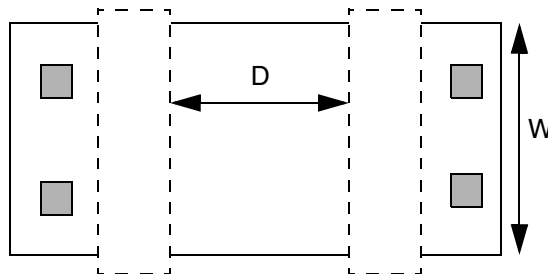
$$SUM2 = (D1/W1 + D2/W2 + SUM1)/3$$

$$NRX = SUM2 + D5/W4$$

(where A is the center point of W2)



$$NRX = (D/W)/2$$



When parallel MOS devices are smashed, the equivalent NRX value is calculated as shown in the following equations:

Dracula Reference

Description Block Commands

For each source or drain, the equation is

$$A = NRX \times (GateEdgeWidth)^2$$

To calculate the final NRX, the equation is

$$\begin{aligned} Asum &= \Sigma(A) \\ Wsum &= \Sigma(GateEdgeWidth) \\ NRX_{final} &= (Asum)/(Wsum)^2 \end{aligned}$$

NRX Extraction With More Accurate Computation

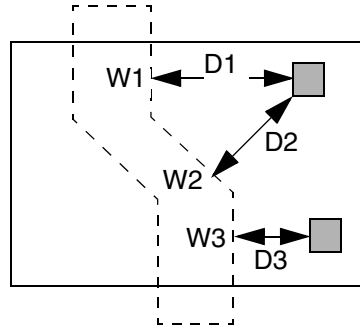
In Dracula 4.81 and subsequent versions, you can get a more accurate computation for the NRX parameter in MOS source/drain resistance extraction (NRX-EXTRACT).

The existing computation of the NRX value in multiple gate edge case uses the average of D_i/W_i as the final NRX value, where W_i is the width of each gate edge segment and D_i is the distance from contact to its corresponding gate edge as shown in the following example. Assume that the coefficient is 1.

Dracula Reference

Description Block Commands

$$\begin{aligned}NRX1 &= D1/W1 \\NRX2 &= D2/W2 \\NRX3 &= D3/W3 \\NRX &= (NRX1+NRX2+NRX3)/3\end{aligned}$$



The three source/drain resistors in this example are actually connected in parallel. The total equivalent resistance value should be $R = 1 / (1/R1 + 1/R2 + 1/R3)$ (assuming that the three resistance values are $R1$, $R2$ and $R3$ respectively). So for the NRX extraction, a more accurate computation for the final NRX value for this example should be

$$NRX = 1 / (1/NRX1 + 1/NRX2 + 1/NRX3)$$

Arguments

subType

MOS element subtype.

accuracy = HIGH

When you specify **HIGH**, the D value is extracted for each gate-edge segment.

accuracy = QUICK

When you specify **QUICK**, the D value is equal to the value you specify.

accuracy = HIGH-PARALLEL

When you specify *accuracy* = **HIGH-PARALLEL**, the D value is of the same as *accuracy* = **HIGH**. If there are multiple gate edge segments, the final NRX value is computed as the following equation where n is the number of gate edge segments. The computation for each $NRXi$ is still the same as the case of *accuracy* = **HIGH**.

Dracula Reference

Description Block Commands

$$NRX = \frac{\text{coefficient}}{n \sum_{i=1} \left(\frac{1}{NRXi} \right)}$$

For example,

```
NRX-EXTRACT      = MOS[HN] HIGH-PARALLEL 0.0 1.0
```

distance

If you specify accuracy of `HIGH`, distance is equal to the distance required to group contacts together. If the distance between the contacts is smaller than the distance value, the contacts are grouped together to make one contact. If the distance value is 0, there is no grouping.

When you specify `accuracy = QUICK`, the value is equal to the contact distance you specify. The value is used as `D` in the NRX calculation.

SHADE

Optional argument you can specify when using `accuracy = HIGH` to take into consideration the shading of contacts.

coefficient

The NRX equation, which is similar to the sheet resistance value. The default is 1 if you do not specify otherwise.

```
NRX = coefficient * Deff / Weff
```

You must follow these rules for calculating NRX:

- Contact distance is defined by the distance between the gate edge and the contact edge.
- If they exist, visible contacts dominate. A contact is *visible* to a gate edge if it overlaps the projection area of the gate edge.
- Only the nearest contact is taken into consideration when contacts are at various distances from the gate edge.

Dracula Reference

Description Block Commands

When MOS devices are smashed, their equivalent NRX value is calculated by the following equations:

$$\begin{aligned} \text{AREAi} &= \text{NRXi} * \text{Wi} * \text{Wi} \\ \text{AREAtotal} &= \text{SUM}(\text{AREAi}) \quad i=1,m \\ \text{Wtotal} &= \text{SUM}(\text{Wi}) \quad i=1m \\ \text{NRXfinal} &= \text{AREAtotal} / \text{Wtotal} / \text{Wtotal} \end{aligned}$$

Where

m	Number of MOS devices in parallel
NRXi	Resultant NRX value for MOS device <i>i</i>
Wi	Total width of gate edges for terminal X of MOS device <i>i</i>
NRXfinal	Final equivalent of NRX for the smashed device

Example

```
*Description
NRX-EXTRACT= MOS[P] QUICK 1.0 2.1
NRX-EXTRACT= MOS[N] HIGH 2.0 1.1
*End
*Operation
element      mos[n] ngate   poly    nsd    pwell
element      mos[p] pgate   poly    psd    nwell
*End
```

The extracted SPICE file might look as follows:

```
***** CORNER ADJUSTMENT FACTOR = 0.0000000*****
M1  VSS  A  VSS  VSS  N TF7 1  L=1.31U  W=16.00U  AD=104.70P  PD=45.09U
+   AS=130.30P  PS=48.29U  NRS=0.381  NRD=0.271
M2  VDD  A  VDD  VDD  P TF7 1  L=1.60U  W=16.00U  AD=102.40P
+   PD=44.80U  AS=128.00P  PS=48.00U  NRS=0.131  NRD=0.131
```


NRX-SAVE

NRX-SAVE = *value*

Description

Control the format and precision of nrs/nrd (the resistance value of MOS source and drain) output, used in conjunction with NRX-EXTRACT command (refer to NRX-EXTRACT).

Arguments

value

Number of decimal places in the output. If value is negative, output is in IEEE format with number of decimal places equal to ABS(value).

Example

```
for nrs=12.34567
if NRX-SAVE=4, the output spice is nrs=12.3456;
if NRX-SAVE=-3, the output is nrs=1.234E+1
```

OAOUTTYPE

OAOUTTYPE = MARKER | POLY

Description

Specifies type of output OA22 data.

Arguments

MARKER

Dracula saves output geometry to OA22 database as error markers. Error markers have no layer or purpose properties. In layoutPlus markers are visible by default.

POLY

Dracula saves output geometry to OA22 database as polygons and paths. This geometry will be visible in layoutPlus if these layer and purpose are defined in OA22 library's technology file.

Example

OAOUTTYPE = MARKER

OAOUTTYPE = POLY

OPTIMIZE-RULE

OPTIMIZE-RULE = YES / NO

Description

Adding this command `OPTIMIZE-RULE = YES / NO` in your Description Blocks section of your rule file will let you optimize original rules and create more optimized rules to speed up your run time. This is a new feature in Dracula 4.81 and 4.9 versions and when you add this command in your Description Blocks section, PDRACULA automatically issues suggestions to you for optimizing your rule.

Arguments

YES

Lets you optimize your original rule and speeds up your run time at least for those commands where you have tried to optimize the run time

NO

Is the default

Example

For examples, click on the following links which takes you to specific examples under their respective commands (SIZE, SELECT, OR, and NOT) in Chapter 13 of this manual for methods to optimize your original rule.

- SIZE: See the section, "[Example 13 - Optimizing the Rule Using the SIZE Command](#)" for more information.
- SELECT: See the section, "[Example 2 - Optimizing the Rule Using the SELECT Command](#)" for more information.
- OR: See the section, "[Example 2 - Optimizing the Rule Using the OR Command](#)" for more information.
- NOT: See the section, "[Example 2 - Optimizing the Rule Using the NOT Command](#)" for more information.

OUTCELL

`OUTCELL = cellname`

Description

Specifies output top cell name in OA22 database. The default cell name is the same as defined by OUTLIB command.

Arguments

cellname

Specifies output cell name

Example

`OUTCELL = result`

`OUTCELL = outtopcell`

OUTDISK

`OUTDISK = filename`

Description

Specifies the output file containing the error cells from the run. If INDISK and OUTDISK are the same, Dracula generates the output file with the same name but with a unique extension.

If you specify the output file with an absolute or relative path (any path containing a backslash “\”), the MULTIPLE-DISK command creates the output file under that path if it exists. If the path you specify does not exist or if you do not specify a path, the MULTIPLE-DISK command determines the location of the output file and shows the location of this file in the *.log* file.

See the MULTIPLE-DISK command in this chapter for more information.

Arguments

filename

Output file name for layout error cells. Maximum path or file name is 128 characters.

Examples

In the following example, Dracula generates an output file named DRCOUT.DAT on disk.

```
OUTDISK = DRCOUT
```

OUTLIB

`OUTLIB = cell-name`

Description

Specifies the names for the output file library and the output primary name of the graphic output error cells for GDSII. The default output primary name is the input primary cell name you specify prefixed with OUT (that is, *OUTcell* name).

Arguments

cell-name

Specifies the cell name for the output file. The maximum name size is 32 characters.

Note: Be aware of special character restrictions for GDSII.

Examples

In the following example, the output library name is *lib2*.

```
OUTLIB = lib2
```

The output library name is *fifo.ERR* in the following example:

```
OUTLIB = fifo.ERR
```

In the following example, the output library name documents the date (February 29th, 1995).

```
OUTLIB = sram_2_29_95
```

In the following example, the output library name documents the circuit (1800 REV Z) and the date (April 1st, 1995).

```
OUTLIB = 1800_REV_Z_4_1_95
```

OUTPUT-ONE-LAYER

OUTPUT-ONE-LAYER= YES/NO

Description

Assigns all geometries associated with an error to one output layer.

Arguments

YES

Assigns all geometries to one output layer.

NO

Default.

Example

In the following example, all layers that make up shorted nodes are output on layer 25 only, instead of on multiple layers, such as *metal* to 25, *poly* to 26, and so on.

```
* DESCRIPTION
.
.
.
OUTPUT-ONE-LAYER = YES
* OPERATION
.
.
.
MULTILAB OUTPUT SHORTS 25
```

OVPR-TOLERANCE

OVPR-TOLERANCE = YES/NO

Description

Specifies a tolerance capability to resolve round-off errors occurring during layout parameter extraction. If you specify YES, the resolution value is automatically set to the scale value.

Arguments

YES

Sets the resolution value equal to the scale value.

NO

Default.

PARALLEL-FILE

PARALLEL-FILE = *configuration-file*

Description

Specifies the node configuration file to use in a distributed Dracula run. You create the node configuration file with a text editor such as *vi* or *emacs*. The node configuration file contains information about the nodes to be used in the run.

Include this command in the Description block if you are running distributed Dracula.

Arguments

configuration-file

The name of the configuration file.

Example

PARALLEL-FILE = machine.com

Node Configuration File

Each line in the node configuration file lists a node to be used in the run and contains the following six fields:

Note: The master node must be the first node in the PARALLEL-FILE command.

Field 1

Name of the workstation to be used in the run.

Field 2

Working directory on this workstation. The working directory must be different for each entry because no two workstations can share the identical disk space.

Field 3

Node type. The possible values are 1, 0, and asterisk (*). For the master node, specify 1. For a slave node, specify 0 or *. The master node must be the first entry.

Field 4

Dracula executable path for this workstation. An asterisk specifies the path listed by the PROGRAM-DIR command in the rules file.

Dracula Reference

Description Block Commands

Field 5

Power rank of this workstation in relation to the other workstations being used in the run. The most powerful workstation has the highest number. An asterisk specifies a rank of one.

Field 6

Mount type, which can be one of the following:

- 0 Soft mount. Appends */net/hostname/* in front of the working directory.
- 1 Hard mount. Appends */dracnet/hostname* in front of the working directory.
- 2 Suppresses automount and keeps the original working directory for each machine.

Sample Node Configuration File

```
iron      /dracula/test  1  *                               4  0
silicon   /dracula/test  0  *                               3  0
nickel    /dracula/test  *  /net/nickel2/dracula/bin 3  0
```

In this example, *nickel* has an executable path that is different from the path specified in the PROGRAM-DIR command.

For more information about the node configuration file and how PDRACULA uses this file, refer to [“Verifying a Design with Dracula”](#) on page 39.

PAR-COMPARE

```
PAR-COMPARE = layout_box_element_name/  
    layout_parameter_name1, schematic_parameter_name1, tolerance1 /  
    layout_parameter_name2, schematic_parameter_name2, tolerance2 / ....  
    layout_parameter_nameN, schematic_parameter_nameN, toleranceN
```

Description

Triggers the LVS parameter comparison for box elements. This command specifies the corresponding schematic parameters to which each layout parameter should be compared to. PAR-COMPARE also defines the comparison tolerance for each pair of parameters.

Arguments

layout_box_element_name

User-defined device name in the ELEMENT BOX command.

layout_parameter_name

Layout parameter name.

schematic_parameter_name

Schematic parameter name defined in the netlist.

tolerance

The percentage tolerance for the comparison of a specified schematic/layout parameter pair.

Example

This example below defines an inductor as a box element on both the schematic and layout; the PAR-COMPARE command triggers the comparison of inductor parameters in LVS.

Schematic netlist

```
*.EQUIV IN1=spiral_rad  
.SUBCKT spiral_rad T1 T2  
.ENDS  
.SUBCKT TEST1 P1 P2 P3 P4  
X2 P1 P2 spiral_rad nr=4.5 rad=6e-05 w=1.5e-05 s=1.5e-06  
X3 P3 P4 spiral_rad nr=3.5 rad=5e-05 w=1.5e-05 s=1.5e-06  
.ENDS
```

Dracula Reference

Description Block Commands

Dracula LVS rule

```
*DESCRIPTION
.....
PARSET INDP ID IN IW IS LL
PAR-COMPARE = IN1/ID,RAD,5/IN,NR,3/IW,W,5/IS,S,6
....
*END
...
*OPERATION
...
ELEMENT IN1 ME5I ME4 ME5A
...
LEXTRACT INDP ME5I BY IN1 INDPAR&
EQUATION LL = ID+IN+IW+IS
....
```

The above example compares instances of the layout box element `IN1` with instances of the schematic box element `spiral_rad` (from the `*.EQUIV` command). The `PAR-COMPARE` command triggers the LVS parameter comparison for instances of the box element `IN1`. The layout parameter `ID` is compared with the schematic parameter `RAD`, which has a 5% tolerance. The layout parameter `IN` is compared with the schematic parameter `NR`, which has a 3% tolerance.

PAROUTNAM

`PAROUTNAM = pname name1=param1 [name2=param2...[namen=paramn]]`

Description

Lets you define the names of parameters in your SPICE output.

Note: PAROUTNAM does not work for PARASITIC capacitance.

Versions of Dracula prior to 4.81 reported the predefined parameters for specific element type only and you had no control on selecting or specifying which parameters are reported in the output SPICE file in Dracula LPE/PRE jobs. This prevented you from changing those parameter names after your Dracula LPE job and before you ran circuit simulation.

In Dracula 4.81 and subsequent versions, with this PAROUTNAM command, Dracula now:

- Lets you select or specify which parameters to be reported in the output SPICE file in Dracula LPE or PRE jobs
- Lets you specify the output names of the parameters

Arguments

pname

Specifies the parameter set name. It must have been defined in a PARSET command, for example,

`PARSET = pname param1 param2..paramn.`

Or, it's a default PARSET, such as MOSD.

name1, name2

The user-defined output parameter name for the specified parameter in the PARSET. It will be output to the SPICE output. The user-defined output parameter name should be one to four alpha-numeric characters with the first character being alphabetic; otherwise, PDRACULA will output an error message:

`**ERROR: ILLEGAL USER-DEFINED OUTPUT PARAMETER NAME .`

param1, param2

The parameter name defined in the corresponding PARSET command,

`PARSET = pname param1 param2..paramn`

Note: When you use PAROUTNAM, only the parameters specified are output to the SPICE output, with the user-defined names for them. Especially, you must specify all of

Dracula Reference

Description Block Commands

the parameters that you want to output to the SPICE file, including the default parameters that you want to output to the SPICE file. When PAROUTNAM is not used, all the default parameters are output to the SPICE file.

You cannot use W;L as parameter names in PARSET for any elements except MOS elements (see *Reserved Parameter Keywords* in “PARSET” on page 731). To set up custom parameter names you must write symbols different from W or L (e.g. WS, LS, WM, LM...). See [Example 3](#) below.

Important

Only 100 PAROUTNAMs are allowed. Each PAROUTNAM can have up to 9 *name=param* definitions.

You can use PAROUTNAM to output user defined parameters, which are not specified for LPESELECT. In this case you should define the types of devices in LPESELECT to be used with PAROUTNAM. [Example 4](#) below shows you how to output capacitance values for the parasitic capacitor “FC” using PARSET and user defined values ‘W’ and ‘L’ for the canonical capacitor “PY” using PAROUTNAM and PARSET.

Examples

Example 1 - Output the Inductor Parameters

```
*DESCRIPTION
...
PARSET = TEST ID IN IW IS LL
PAROUTNAM = TEST ID=ID LSS=LL
...
*END
...
*OPERATION
...
ELEMENT IN1 INDSYM ME4 ME5
LEXTRACT TEST INDSYM BY IN1 PARAM1 &
EQUATION LL=-0.5+4*(ID/100)-0.9*IN
...
LPECHK
LPESELECT IN1 OUTPUT SPICE
...
*END
```

Dracula Reference

Description Block Commands

The output of SPICE netlist:

```
...
*
*****
***** BOX Device PARAMETERS FROM : 7BOXXNET
*****
*
*
XB34      6  5  / IN1  ID=29140.00 LSS=789.00
*
*----- TOTAL # OF BOX Devices FOUND :           1
*----- COMMENTED :                               0
*
...
```

Example 2 - Output the MOS Parameters

```
*DESCRIPTION
...
UNIT = PERIMETER,U
MODEL = MOS[N],N
PAROUTNAM = MOSD LR=L WR=W
...
*END
*OPERATION
...
ELEMENT MOS[N] RBNGAT PLY NSD PSUB
...
LPECHK
LPESELECT MOS OUTPUT SPICE
...
*END
```

The output of SPICE netlist:

```
*
* CADENCE/LPE SPICE FILE : SPICE
...
M1      VSS  A  VSS  VSS  N  LR=1.31U  WR=16.00U
*----- TOTAL # OF MOS TRANSISTORS FOUND :           1
*----- COMMENTED :                               0
```

Dracula Reference

Description Block Commands

*
.ENDS

Example 3 - Output the Resistor Parameters

```
PARSET MRPS AREA W1 W2 WS LS
PAROUTNAM MRPS LR=LS WR=WS
...
...
LEXTRACT MRPS prpip poly B Y RES[PS] RESPS &
EQUATION WS = W1 + W2 &
EQUATION LS = AREA/WS
```

The output of SPICE netlist:

```
*****
***** RESISTORS PARAMETERS FROM: 7RESXNET
*****
*
*
...
R605 7 8 RF_RNRPOS 2.47758E03 $W=0.17U $L=1.40U
R606 1 6 RF_RNS 6.27879E01 $W=0.17U $L=1.40U
R607 8 9 RF_RPRPOS 2.63879E03 $W=0.17U $L=1.40U
R608 6 7 RF_RPS WR=0.33000U LR=0.70000U
R609 9 10 ND 5.00606E02 $W=0.17U $L=1.40U
R610 INDAA 10 PD 1.12848E03 $W=0.17U $L=1.40U
*
*----- TOTAL # OF RESISTORS FOUND :          6
*----- COMMENTED :          0
*
```

Example 4 - Output User Defined Parameters for Capacitor.

```
.....
PARSET = NEWS TPR CLL AREA PERI OVPR C1 C
PARSET = CAPP AREA MIX WID LEN PERI
PAROUTNAM = CAPP L=LEN W=WID
.....
PARASITIC      CAP[FC]      M1S2      BM1      PSUB
ATTRIBUTE      CAP[FC]      1.876E-17    5.856E-17    0.9      3.386E-17
```


Dracula Reference

Description Block Commands

```

.....
ELEMENT          CAP[PY]      CP2CP  POLY  CP2CAP
.....
LEXTRACT NEWS    M1S2  NS2  BY          CAP[FC]  FCT &
EQUATION C1=OVPR &
LEXTRACT NEWS    M1S2  M1S2  BY          CAP[FC]  &
EQUATION C=1.876E-17*AREA+5.856E-17*(PERI-C1-TPR)+CLL
.....
LEXTRACT CAPP CP2CP POLY BY CAP[PY] SCA &
EQUATION MIX = SQRT(PERI*PERI-25*AREA) &
EQUATION LEN = (PERI+MIX)/4 -3*0.04 &
EQUATION WID = (PERI-MIX)/4 -3*0.04
.....
LPESELECT[AG] CAP[FC] &
LPESELECT[AG] CAP[PY] OUTPUT SPIL

```

Example output of SPICE netlist:

```

*****
*****  CAPACITORS  PARAMETERS FROM : 7CAPPFCC
*****
*
*
CC1    GND  GND  6.78957E-15
CC2    A   GND  5.50732E-14
CC3    B   GND  4.81427E-15
CC4    4   GND  9.95367E-16
CC5    6   GND  3.85104E-15
*
*-----  CAPACITANCE TYPE : FC
*-----  TOTAL # OF CAPS FOUND :           5
*-----  COMMENTED :                     0
*
*****
*****  CAPACITORS  PARAMETERS FROM : 7CAPXNET
*****
*
*
C1     B   4   LPE_C    W=7.83000UM  L=28.93000UM
*
*-----  CAPACITANCE TYPE : PY

```

Dracula Reference

Description Block Commands

```
*----- TOTAL # OF CAPS FOUND :          1
*----- COMMENTED :                      0
*
.ENDS
```

PARSET

`PARSET = pname, param1, param2...paramn`

Description

Specifies the names of a parameter set for a group of geometric primitives or computed parameters. To extract these geometric primitives and computed parameters, use the LEXTRACT and EQUATION commands in the Operation block. If you specify LEXTRACT, you must also specify PARSET.

Arguments

pname

Specifies the parameter set name. Maximum length is four alphanumeric. The first characters must be a letter. Do not specify any of the following reserved names: MOSD, DIOD, DIO2, DIO3, CAPD, CAPF, BJTD, or RESD. Default parameter set name is CAPO.

param

Specifies the parameters in a set. The parameter limit is 18. PRE recognizes only the first ten parameter keywords and overwrites the last eight parameters with node and subnode information in order to save the data file size. The reserved parameter keywords used by SPICE must be limited to the first ten parameters. The remaining eight parameters can be used for LPE.

When you need to support inductors in layout designs, you must first use this PARSET command (in the description block commands section of your rule file) to specify what parameters will be extracted, followed by the ELEMENT IN? in the operation block of commands in your rule file. There are four geometric primitives regarding inductors provided: ID, IN, IW, and IS and one reserved parameter, LL.

See the section below for information these 4 primitives, ID, IN, IW, and IS and the parameter, LL.

See the [ELEMENT IN?](#) section in the “Operation Block Commands” chapter of this manual for more information on that command.

Parameters can be one of the following three types:

■ **Geometric Primitives**

Primitives that relate to the layer or layers you specify in LEXTRACT. Extracted values relate to each device layer of *layer-a* in LEXTRACT. You cannot alter these primitives by using an equation.

ANG	Total internal angle
AREA	Area
PERI	Perimeter
W1	Overlapped perimeter to node 1 of second layer
W2	Overlapped perimeter to node 2 of second layer
OVAR	Overlapped area to second layer
OVPR	Overlapped perimeter to second layer
WIDT	Distance that the geometries on the layer run parallel
DEPT	Distance between the parallel geometries
TPR	Perimeter of touching geometries that cause fringe effect
CLL	Sum of fringe effect related to TPR
ID	Diameter of the inner circle. It's possible that the diameter of X-direction is different from that of Y-direction. Whichever is smaller will be selected.
IN	Number of circles
IW	Width of each circle
IS	Spacing between circles

■ **Reserved Parameter Keywords**

Keywords reserved for parameter values that relate to specific devices. You usually specify these keywords for CDL or SPICE simulations. You change these keywords by using an EQUATION command.

W	MOS, BJT, and RES channel width

Dracula Reference

Description Block Commands

L	MOS, BJT, and RES channel length
AD	MOS
PD	MOS
AS	MOS
PS	MOS
A1	DIO area
P1	DIO perimeter
A2	DIO area
P2	DIO perimeter
A3	DIO area
P3	DIO perimeter
C	CAP value
R	RES value
CA	BJT collector area
CP	BJT collector perimeter
BA	BJT base area
BP	BJT base perimeter
EA	BJT emitter area
EP	BJT emitter perimeter
K	Fringe capacitance, attribute 2
LL	Inductance of the inductor and it is calculated depending on the equation specified in the rule file associated with that particular element. See the <u>“Supporting Inductors in Layout Designs Using the PARSET and ELEMENT IN? Commands”</u> section in the “Extracting Electrical Parameters (LPE)” chapter of this manual for information on the different types of inductors.

■ User-Defined Parameters

Names other than reserved parameter keywords. The maximum length is four alphanumeric characters. The first character must be a letter.

Dracula Reference

Description Block Commands

Example

```
PARSET =      TEST  ID  IN  IW  IS LL
```

PATH-W-OFFGRID

PATH-W-OFFGRID = YES/NO {*grid-value*}

Description

Checks whether the width of a path is off grid. Dracula checks whether the width of each path is a multiple of the grid size if paths expand on one or both sides. You can use a different grid by specifying a grid value.

Arguments

YES

Dracula checks each path.

NO

No checks are done. Default.

grid-value

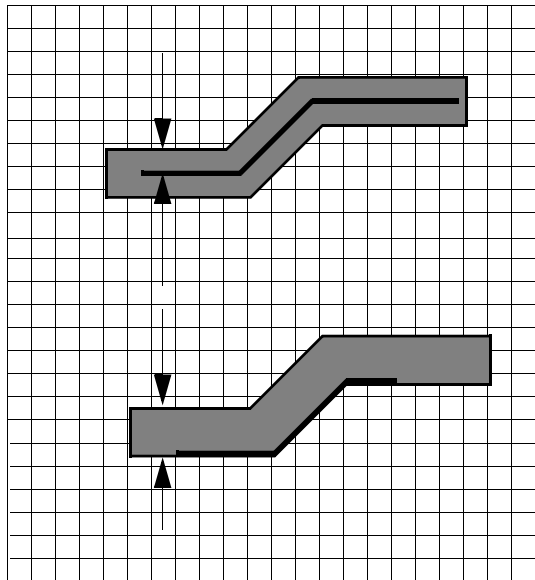
Specifies the grid to use. You must specify a real number greater than 0.0 but less than 1.0

Dracula Reference

Description Block Commands

Example

PATH-W-OFFGRID = YES



Path option 1

Checks if width is a multiple of the grid size

Path option 2

PATH-W-ROUND

PATH-W-ROUND = DOWN/UP/VIRTUOSO

Description

Finishes a non-orthogonal path by rounding the path width either up or down or in a Virtuoso manner.

Arguments

DOWN

Rounds down the path width so that it is smaller than or equal to the true path width.

UP

Rounds up the path width so that it is larger than or equal to the true path width.

VIRTUOSO

Rounds the path width in a Virtuoso manner.

Examples

PATH-W-ROUND = UP

PATH-W-ROUND = DOWN

PATH-W-ROUND = VIRTUOSO

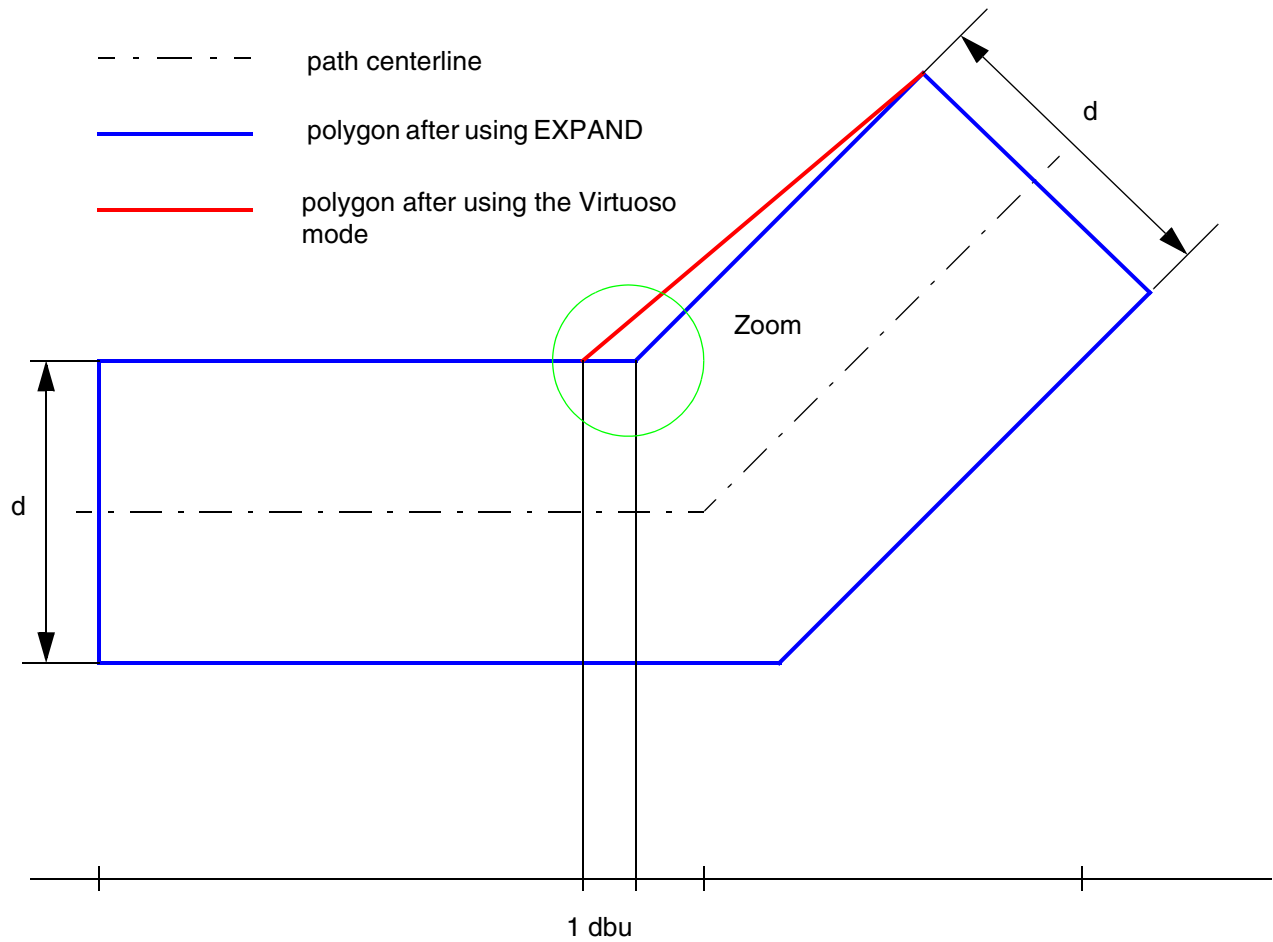
The SIZE or SIZE DOWN-UP commands produces false errors on a 45 degree path after converting it to a polygon. The Virtuoso layout editor and the Dracula tool calculate the coordinates of the path's polygon in the different manner. The difference may be equal to approximately 1 dbu (see the illustration below). The red-colored line represents the line that the Virtuoso layout editor produces.

If you want to see figures that are exactly the same as in the Virtuoso layout editor, you need to add the "convert path to polygon in a Virtuoso style" feature. You add this feature by turning on the Virtuoso mode to the round-off path using the Virtuoso layout editor (even if it's not mathematically correct). The Dracula tool fixes the width of the polygon when building it, instead of saving the parallelism of two opposite edges of the path's polygon in a Virtuoso manner. If the coordinates were floating point type, the result looks the same. But, since the coordinates are integers you get a difference of 1 dbu in some cases.

The Virtuoso-mode allows you to choose the method that is most appropriate for you.

Dracula Reference

Description Block Commands



PINCAP-FILE

PINCAP-FILE = *pincapfile*

Description

Specifies the *pincapfile*, which contains the pin capacitance information in the form:

```
cellName1
pinName1   cap-value1
pinName2   cap-value2

cellName2
pinName1   cap-value1
pinName2   cap-value2
```

For example:

```
INV
IN          3.0
```

If you do not use this command to specify a *pincapfile*, the default capacitance value for all pins is zero.

Arguments

pincapfile
Name of the pin capacitance file.

Example

In the following example, pins in *design* are assigned the capacitance values specified in the *design.pincap* file.

```
PINCAP-FILE = design.pincap
```

POINT-CONNECT

POINT-CONNECT = YES/NO

Description

Tells Dracula to connect point contacts on the same node.

Arguments

YES

Connects all point contacts on the same node.

NO

Does not connect point contacts of the same node.

Note: Potential problems might arise when using this command:

- Dracula might not connect the point contacts when the polygons should be connected (For example, they should be two nodes). This results in a clean LVS run. The polygons, however, might be fused together in manufacturing, resulting in a short.
- Dracula might connect the point contacts when the polygons should be connected. (For example, it should be one single node.) Again, this results in a clean LVS run. However, because the connection is done via a single point, the yield on this chip is very low. The chip might easily be “loosened,” causing the chip to fail.

POWER-NODE

POWER-NODE = *name1, name2...namen*

Description

Specifies the names of power nodes that are assigned text in the layout or in EDTEXT. If you do not properly text a non-unique power text string in the layout, this command converts the non-unique text string to power (pad type P). This command does not add new text.

You can specify an asterisk (*) as a wildcard character following the power text string.

Note: In version 4.7.0399, Dracula can support up to 100,000 nodes for the POWER-NODE command in the Description block.

Arguments

namen

Specifies text to convert to power (pad type P).

Examples

In the following example, VDD1, VDD2, and VPP are power nodes:

```
POWER-NODE = VDD1, VDD2, VPP
```

In the following example, the VDD text string followed by any character is a power text string:

```
POWER-NODE = VDD*
```

PREFIX-PARASITIC

`PREFIX-PARASITIC = (parasitic-element, prefix...)`

Description

Specifies prefixes for parasitic device names generated by LPE in the output SPICE or CDL netlist.

Arguments

parasitic-element

Specifies the user-defined parasitic device in the layout. Can be a diode (DIO), a capacitor (CAP), or a resistor (RES).

prefix

Specifies the prefix to form names for the parasitic element. The prefix can be one to four letters. You can specify only one prefix for each parasitic element type.

The default prefixes are as follows:

Diodes	D
Capacitors	C
Resistors	R

In the SPICE or CDL output file, the prefix is concatenated with a running sequence of unique numbers to form parasitic device names (for example, D99, D100, C1, C2, R158, or R159).

Example

In the following example, the system uses *CC* as the prefix for the parasitic capacitor names, *DID* for the parasitic diodes, and *RAN* for the parasitic resistors.

```
PREFIX-PARASITIC=CAP,CC DIO,DID RES,RAN
```

In the output file, names are of the form CC1, CC2, DID99, DID100, RAN158, or RAN159.

PREINTERACTIVE

PREINTERACTIVE = NO/YES

Description

Specifies whether you plan to use the Cadence Dracula Interactive software to view HLVS error flags.

If you specify PREINTERACTIVE = YES, PDracula adds the module PREINQUERY to the end of the *jxrun.com* file. The PREINQUERY module sorts the files *6LVSCL1.CEL*, *6LVSCL2.CEL*, and *6LVSCL3.CEL* for Dracula Interactive's use. This module adds to Dracula's run time, so include it only if you run HLVS and want to view the error flags with Dracula Interactive.

After running Dracula with PREINTERACTIVE = YES, to rerun Dracula from a break point, the break point must occur before LVSCCHK so Dracula can recreate the *6LVSCL1.CEL*, *6LVSCL2.CEL*, and *6LVSCL3.CEL* files. You cannot use the LVSCMP reserved break point. However, you might be able to use the reserved break points LVSNET and SCHMIN. If only the schematic has changed, you can use SCHMIN. If the layout has changed, you can use LVSNET, but cannot use SCHMIN.

Note: For flat/composite mode LVS error information, Dracula Interactive requires the files *6ICDBG.DAT*, *6LVSCL1.DAT*, *6LVSCL2.DAT*, and *6LVSCL3.DAT*. For cell mode LVS, Dracula Interactive requires the files *6ICDBG.CEL*, *6LVSCL1.CEL*, *6LVSCL2.CEL*, and *6LVSCL3.CEL*.

Arguments

NO

Indicates that you do not want to use Dracula Interactive to view HLVS error flags.
Default.

YES

Specifies you want to use Dracula Interactive to view HLVS error flags.

Example

PREINTERACTIVE = YES

At the end of the run, Dracula sorts the *6LVSCL1.CEL*, *6LVSCL2.CEL*, and *6LVSCL3.CEL* files. PREINTERACTIVE is the last module listed for the run in the *.log* file.

Dracula Reference

Description Block Commands

Note: In versions prior to 4.8, this command was called as PREINQUERY = YES, and this command also still works.

PRIMARY

`PRIMARY = cell-name {view-name}`

Description

Specifies the top-level layout cell name. You must specify this command in the Description block.

Arguments

cell-name

Specifies the top-level layout cell name. Dracula converts the entry to uppercase unless you specify the `SYSTEM = CADENCE_44DD/OA22` or `CNAMES-CSEN = YES` command in the Description block.

view-name

Specifies the cell's view name. This applies only if you specify `SYSTEM = CADENCE_44DD/ OA22` command in the Description block. The default value is `layout` if omitted.

Examples

- The following example specifies the layout cell *iomux* as the primary cell:
`PRIMARY = iomux`
- The following example specifies the DFII *artChip* layout cell's *layoutA* view as the primary cell:
`PRIMARY = artChip layoutA`

Dracula Reference

Description Block Commands

PRINTFILE

`PRINTFILE = filename`

Description

Defines the prefix name of the printed output files. This command generates two files:

- A log file or listing file containing the batch printed output
- A short summary of the violation output cells, the problem geometries listing, and the input command listing

If you use a UNIX system, specify the logfile when you submit *jxrun.com*.

Print Files Per Dracula Run

Type of run	DRC	ERC	LVS	LPE
PRINTFILE =	DRCPRT	ERCPRT	LVSPRT	LPEPRT
INPUT	DRCPRT.INP	ERCPRT.INP	LVSPRT.INP	LPEPRT.INP
LOG	DRCPRT.LOG	ERCPRT.LOG	LVSPRT.LOG	LPEPRT.LOG
ERROR	DRCPRT.ERR	ERCPRT.ERR	LVSPRT.ERR	LPEPRT.ERR
ERC SUMMARY	(NONE)	ERCPRT.ERC	LVSPRT.ERC	LPEPRT.ERC
SUMMARY FILE	DRCPRT.SUM	ERCPRT.SUM	LVSPRT.SUM	LPEPRT.SUM
DISCREPANCY	(NONE)	(NONE)	LVSPRT.LVS	LPEPRT.LVS (only if LPECHK is used)

Arguments

filename

Specifies the file name for printed output. The file name can have up to 48 alphanumeric characters and no special characters. Default file name is PRINTF.

Example

`PRINTFILE = myout`

PROB-GEOM-LIST

PROB-GEOM-LIST = *xxxx/NO*

Description

Specifies the number of problem geometries to be listed in the `.sum` file. The number can be up to 4 digits (*0 - 9999*). If this command is not in the description block, then the default number is 500.

Note: It is recommended that you use this command, if you do not know the limit of data integrity check.

Arguments

xxxx

Specifies the number of problem geometries in the `.sum` file

Example

PROB-GEOM-LIST = 9876

The number of data integrity check specified in the rule file will be 9876.

PROCESS-VAR

```
PROCESS-VAR = CAP, subtype [CAP, subtype] ...  
              RES, subtype [RES, subtype] ...
```

Description

Specifies parasitic devices CAP and RES that are affected by process variation. PDRACULA generates a different LPE/PRE flow.

Instead of doing a geometric extraction for these new given coefficients, you recalculate the R and C values with these new coefficients and the geometric data from the first run.

Arguments

parasitic devices

Specifies the layout parasitic device type. The device type can be parasitic capacitance (CAP) or parasitic resistance (RES).

subtype

Specifies the subtypes of parasitic devices that are affected by process variation. The parasitic devices must be defined in the operation block.

all

Specifies that all subtypes of parasitic devices are affected by process variation.

Example

Process variation supports flexible LPE. Here is an example.

```
*DESCRIPTION  
PROCESS-VAR = CAP [AA], CAP [B0], RES[A], RES[B]  
;PROCESS-VAR = ALL  
LPE-QUERY = CORE  
*END  
*OPERATION  
# rules for 1st set process variation  
*break PROCESS-VAR --> required  
# rules for 2nd set process variation  
*break PROCESS-VAR --> required  
# rules for 3rd set process variation
```

Dracula Reference

Description Block Commands

```
*break PROCESS-VAR --> required  
*END
```

Dracula Reference

Description Block Commands

PROGRAM-DIR

PROGRAM-DIR = *directory-name*

Description

Specifies the directory containing the Dracula program. All reference libraries must be in the run directory or symbolically linked to the run directory.

Arguments

directory-name

Specifies the directory name or an assigned logical name containing the program modules. The maximum path or directory name is 128 characters.

Example

The following example specifies *cadence* as the directory containing Dracula:

PROGRAM-DIR = *cadence*:

Option	Description
UNIX	PROGRAM-DIR = /usr/drac/

PULL-DOWN

`PULL-DOWN = text`

Description

Defines ground text used by the PATHCHK and FILTER commands. If you do not use this command, Dracula uses the default ground text to do path tracing.

Arguments

`text`

Specifies the text you use in addition to default ground text.

Example

This example specifies VSS, ADD1, and ADD2 as pull-down sources in the PATHCHK and FILTER commands. Pull-up sources are not changed.

```
PULL-DOWN = VSS ADD1 ADD2
```

PULL-UP

`PULL-UP = text`

Description

Defines power text used by the PATHCHK and FILTER commands. If you do not use this command, Dracula uses the default power text to do path tracing.

Arguments

`text`

Specifies the text you use in addition to the default power text.

Example

This example specifies VCC, VDD, ADD1, and ADD2 as pull-up sources in the PATHCHK and FILTER commands. Pull-down sources are not changed.

```
PULL-UP = VCC VDD ADD1 ADD2
```


RAM-CELL

RAM-CELL = *cell-name1*, *cell-name2*...

Description

Identifies which cells are RAM cells. Classifying cells this way makes Dracula perform more efficiently.

If Dracula chooses a particular cell as a RAM cell, only the outer ring of RAM cells is checked against foreign cells. In the core of the array, only violations among the same RAM cells are reported in the *drc.rpt* file.

In the EXPAND module, Dracula assigns new cell names to outer ring cells and core cells. Dracula reports DRC errors in the summary file under these new cell names.

Checking Method

modes

Hierarchical.

checks

HDRC.

Arguments

cell-name

Specifies the name of the RAM cell structure arrayed in the layout database. The maximum number of RAM cells is 20.

Example

RAM-CELL = RAM1, RAM2

RAM-CORE

RAM-CORE = *cell-name1*, *cell-name2*...

Description

Identifies which cells are RAM cores. Classifying cells this way makes Dracula perform more efficiently.

If Dracula chooses a particular cell as a RAM core, only the data falling within the DRC distance from the RAM core bounding box is checked against foreign cells. Dracula keeps data that lies across cutting lines unless you specify WINDOW-CUT= EDGE in your run file.

If you are in an intermediate verification stage and need quick results, you can use RAM-CORE. Because Dracula does not process the entire ram-core cell, RAM-CORE reduces the amount of data processed.

Checking Method

modes
 Flat

Arguments

cell-name
 Specifies the name of the RAM core cells. The maximum number of RAM core cells is 20.

Example

RAM-CORE = RAM-CORE1, RAM-CORE2

RCX-MODE

RCX-MODE = YES / LAYOUTONLY / NO

Description

This command allows you to use the data generated through Dracula LVS to perform parasitic extraction using Assura RCX. For additional information see the *Dracula Users Guide*, “Running the DracToRCX Interface”.

Arguments

YES

Enable the DracToRCX flow.

LAYOUTONLY

Enable the DracToRCX flow using only the layout data in cases where the schematic data does not exist.

NO

Disable the DracToRCX flow. This is the default mode.

Example

RCX-MODE = YES

REDUCER

REDUCER = YES {*file_name*} / NO

Description

Enables/disables the interconnect circuit reducer. The LUMPCAP function is automatically turned on when REDUCER is set to YES. REDUCER=NO is the default.

For detailed information about ConclCe, the interconnect reducer, see “[*About ConclCe*](#)” in the *ConclCe Help*.

You cannot use these LPESELECT options and features with ConclCe:

- C, P, R, T, and W
- Grouping capacitor subtypes (LPESELECT *element newsubtype:subtype+subtype+...*)
- Reporting capacitor and resistor subtypes (LPESELECT *element type operator value*)

Checking Method

modes

Composite.

checks

LPE, PRE.

Arguments

file_name

The name of the file containing reduction parameters. You can use this file to run ConclCe iteratively with different parameters. If you do not include a *file_name*, the default parameters are used. Valid parameters are:

- ACCURACY_LEVEL = *value*

The *value* is an integer representing the number of capacitors left between the two end points of a wire segment after reduction.

The higher the accuracy level, the greater the accuracy of the reduction. The default ACCURACY_LEVEL value is 2.

- COUPLING_FACTOR = *value*

Dracula Reference

Description Block Commands

The *value* is a positive floating-point number between 1 and 2 that ConclCe uses as a multiplier to model the effects of coupling capacitance when it is converted to lumped capacitance. The default is 1.

Example

In this example, the following function appears in the Description block of the rules file.

```
REDUCER = YES reduce
```

The file called *reduce* contains the following lines.

```
ACCURACY_LEVEL = 3  
COUPLING_FACTOR = 1.5
```

ConclCe reduces each interconnect segment to three capacitors.

REP-UNM-INST-PG

REP-UNM-INST-PG = YES/NO

Description

Reports all unmatched devices that are connected to matched “big nodes” in the DISCREPANCIES section, as opposed to being reported only in the UNMATCHED DEVICES section of the ./vs report.

Arguments

YES

Reports unmatched devices connected to matched big nodes.

NO

Default.

RERUN-FILTER

RERUN-FILTER = YES/NO

Description

Allows you to re-run the FILTER command after running the REDUCER command. When you add this option line, RERUN-FILTER = YES/NO in your rules file, PDRACULA lets you perform an additional filter of the devices after reduction of those devices having floating diodes.

You use this option particularly while using the R option of the FILTER Commands where you may have a large number of tie-off cells in (H)LVS. These devices may contain some internal nodes which are only connected to the PMOS and NMOS transistors and do not have any connections outside and so these devices with floating diodes need to be filtered off.

Arguments

YES

Does a filter of the devices after a reduction of the devices has been done.

NO

Is the default. It does not allow you to perform an additional filter of the devices after you have run the REDUCER command.

Example

RERUN-FILTER = YES

RESOLUTION

`RESOLUTION = step-size units`

Description

Specifies the minimum step size (grid) for the layout. All processed layout geometries fit on grid points that are integral multiples of the step size you specify in this command. If your layout contains digitized data finer than the step size you specify in RESOLUTION, this command snaps the digitized data vertices to the grid. To see if Dracula has snapped any data vertices, use the FLAG-OFFGRID command.

RESOLUTION relates to the SIZE and RELOCATE commands and the values you specify for them. Because you cannot place data off grid during a SIZE or RELOCATE operation, you must specify values in these two operations that are integral multiples of the step size you specify in RESOLUTION.

If the RESOLUTION size is smaller than SCALE, PDRACULA issues a warning message and changes the RESOLUTION to the same value as SCALE. If the reciprocal value of RESOLUTION is not an integer, PDRACULA issues a warning message and processes the verification job using “real number resolution.”

Arguments

step-size

Specifies the size of the layout’s digitizing increments (internal grid points).

units

Specifies units of *step-size* in microns or mils.

Examples

In this example, the resolution value is set to .125 microns.

```
RESOLUTION = .125 MICRONS
```

In this example, the resolution value is set equal to the SCALE value.

```
SCALE = .001 mic  
RESOLUTION = 0.00005 mic
```

In this example, the resolution value is forced to “real resolution.”

```
REV-SCALE = 8000 mic  
RESOLUTION = 0.333 mic
```


Dracula Reference

Description Block Commands

In this example, the resolution value is forced to “real resolution.”

```
SCALE = .001 mic
RESOLUTION = .08 mic
```

NOTE:

It is strongly recommended that the RESOLUTION value be set to the smallest value possible with respect to the database units specified in the SCALE command. If you set the RESOLUTION value to a larger value, Dracula will snap the polygons and cell placements in your layout data to a grid of that value and may distort the input layout data as well distort derived layers that are created during the course of a Dracula job. This could cause false DRC or LVS errors to be flagged, or allow real DRC or LVS errors to get past Dracula.

For example: If the scale units of your layout is 1000 DBU per micron, you would set the SCALE value to be 0.001 MICRON. You should also set the RESOLUTION value to the same value:

```
*DESCRIPTION
...
SCALE = 0.001 MICRON
RESOLUTION = 0.001 MICRON
...
*END
```

If you want to check that your input layers or cell placements in your layout are on grid, specify the grid value in one of the appropriate FLAG commands. If you want to check if polygons or cell placements are on grid, use the FLAG-OFFGRID command with the desired grid value. For example, if you want to check if your data is on a 0.25um grid, use:

```
DESCRIPTION
...
SCALE = 0.001 MICRON
RESOLUTION = 0.001 MICRON
FLAG-OFFGRID = YES 0.25 ;Check if polygons or cells on a 0.25um grid
...
*END
```

The above example will check if the input data is off grid, but will not snap the data to grid thus keeping the integrity of the drawn shapes as Dracula performs its functions. If there are paths in your input data and you want to check if the centerline vertices are also on grid, in addition to the FLAG-OFFGRID command you also need to use the FLAG-PTH-OFFGRID command:

```
*DESCRIPTION
...
SCALE = 0.001 MICRON
RESOLUTION = 0.001 MICRON
FLAG-OFFGRID = YES 0.25 ;Check if polygons or cells on a 0.25um grid
FLAG-PTH-OFFGRID = YES ;Check if path centerline vertices are on grid
...
*END
```

RESWSMASH

RESWSMASH = YES/NO

Description

This command allows to choose the equation used to calculate the AREA of smashed parallel resistors (see “Calculations on Parallel and Series Resistor Devices width/length for LVS” on page 332).

Arguments

YES

The expression $AREA=1/(1/AREA1+1/AREA2)$ is used for parallel resistors.

NO

Is the default. The expression $AREA=AREA1+AREA2$ is used for parallel resistors.

Example

RESWSMASH = YES

ROTATION

ROTATION = *angle*

Description

Rotates the layout database by the angle you specify.

You can also use the WINDOW and WINDEL commands with the ROTATION command. The ROTATION command affects the coordinates of the WINDOW and WINDEL commands in the same way.

```
WINDOW = 1000.  500.  2000.  2500.  
WINDEL  = 1500.  1500.  1800.  2000.  
ROTATION = 90  
MIRROR  = Y
```

Note: This command can be used in flat mode only.

Arguments

angle

Specifies 90, 180, or 270 degrees of counterclockwise rotation.

Example

This example rotates the entire layout database 90 degrees counterclockwise.

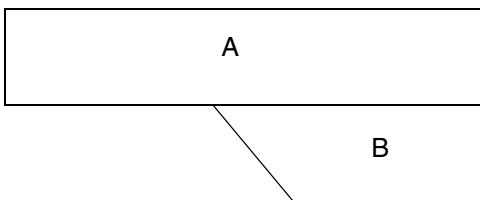
```
ROTATION = 90
```

ROUND-OFF-TOLER

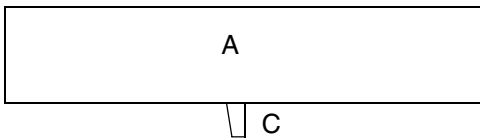
ROUND-OFF-TOLER = *value*

Description

Eliminates slivers caused by rounding off. In the following example, to get shape A only, you use the Boolean operation *NOT* to exclude shape B.



However, because of round-off effects, you might get two shapes, A and C, where C is the result of rounding off.



To prevent slivers from forming as a result of rounding off, specify the ROUND-OFF-TOLER command.

Arguments

value

The value of round-off tolerance allowed in dbu.

SCALE and REV-SCALE

```
SCALE = unit-size units
SCALEIN = unit-size units
REV-SCALE = unit-size units
REV-SCALEIN = unit-size units
```

Description

Specifies the number of internal database units that equals one layout unit. You can specify only one of these commands in the Description block. For GDSII, if you do not specify a SCALE command, the scale value defaults to the STREAM default value.

Arguments

unit-size
Specifies the size of equivalent layout unit.

CAD system	Unit referenced
GDSII	DB Units
Applicon	VLSI Units

The default is the unit size of the original layout database.

units
Specifies units of *unit-size* in microns or mils. The default is the units used in the original layout database.

Examples

```
SCALE = .001 microns
REV-SCALE = 1000 microns
REV-SCALE = 32768 microns
REV-SCALE = 80000 microns
SCALE = .01 microns
SCALE= 0.001 mils
```

SCALEOUT and REV-SCALEOUT

SCALEOUT = *unit-size units*

REV-SCALEOUT = *unit-size units*

Description

Specifies the scale of the output tape (OUTDISK) when you convert a database. These commands work with the SYSOUT command. The default for SCALEOUT is SCALE. The default for REV-SCALEOUT is REV-SCALE.

When you specify different input and output file formats (that is, SYSIN does not equal SYSOUT), the conversion requires a treefile generated by a job that ran with SYSTEM and SCALE specified the same as your SYSOUT and SCALEOUT. For more information about the treefile, refer to the SYSOUT command.

Arguments

unit-size

Specifies the size of equivalent layout unit.

units

Specifies units of *unit-size* in microns or mils.

SCHEMATIC

Description

For details about this command, refer to [“Creating and Compiling an LVS Rules File”](#) on page 279.

SELECT-MODE

SELECT-MODE = LAY/SCH

Description

Indicates whether the node name or number is based on the schematic or layout in a selective LPE run. Include this command in the Description block if you want LPE to extract the layout parasitics and devices associated with a particular path.

You must use two other commands for selective LPE: LPE-QUERY and either NODE-SELECT or NODE-FILE. For details about these commands, refer to [Chapter 13, “Operation Block Commands.”](#)

Arguments

LAY

Specifies that LPE-QUERY use layout node names or numbers.

SCH

Specifies that LPE-QUERY use schematic node names or numbers. If you specify a schematic node name or number, LPE-QUERY reads the input schematic node names and compares them to corresponding layout node numbers to extract the parameters.

Example

SELECT-MODE = SCH

SIZE-MAX-VERTEX

SIZE-MAX-VERTEX = *vertex-value*

Description

Specifies the maximum number of vertices for a polygon during the rebuilding of polygons in the SIZE operation. The default number of vertices is 200, and this default is acceptable to most CAD layout systems.

Some systems can accept only 100 vertices. Others can accept up to 2,000. Use this command to change the number of vertices to obtain the maximum functionality from your system. This command does not apply to the trapezoid output layer.

Arguments

vertex-value

Specifies the maximum number of vertices on polygons after a SIZE operation if you specify a value less than or equal to 200. If you specify a value greater than 200, the maximum value is 2000 for the vertices. Default is 200 vertices.

Example

In this example, when building polygons from layer *metal* to output cell *umetal52*, the maximum number of vertices is 100. No OUTPUT is specified in the oversize of layer *buried* to trapezoid layer *oburied*. Therefore, SIZE-MAX-VERTEX does not apply.

```
*DESCRIPTION
SIZE-MAX-VERTEX = 100
.
.
*END
*OPERATION
SIZE metal BY -1 OUTPUT umetal 52
SIZE buried BY 1 oburied
*END
```

SIZE-MIN-WIDTH

SIZE-MIN-WIDTH = *min-width-value* *unit*

Description

Specifies the minimum width of polygon cutting by the PG handler during the rebuilding of polygons in the SIZE operation. This command does not apply to the trapezoid output-layer. For more information about sizing polygons, refer to the [SIZE](#) command in Chapter 13.

You must specify the correct minimum width value. If the value is too large, there is no place to cut the polygon. If the width of the polygon is not big enough to allow polygon cutting, the command aborts and you get an error message.

Arguments

min-width-value

Specifies the minimum width of polygon cutting during the rebuilding of polygons in the SIZE operation. Value range is 0.005 to 1.0 microns. Default width is 0.4 micron. This command overrides the *W* option in the SIZE command.

unit

Specifies unit of MIN-WIDTH-VALUE. Micron or mil.

Example

In this example, during the building of polygons from layers *cont* and *poly*, the minimum width of polygon cutting is 0.01 microns. Because no OUTPUT is specified in the oversize of layer *sink* to trapezoid layer *ksink*, SIZE-MIN-WIDTH does not apply.

```
*DESCRIPTION
SIZE-MIN-WIDTH = .01 mic
...
*END
*OPERATION
SIZE cont BY 0.5 OUTPUT ocont 50
SIZE poly BY 0.2 OUTPUT opoly 51
SIZE sink BY 1 ksink
*END
```

The SIZE command generates the following output cells:

OCONT50G	Good sized polygons
OCONT50A	Sized acute angle polygons, if any

Dracula Reference

Description Block Commands

OCONT50R	Questionable polygons
OPOLY51G	Good sized polygons
OPOLY51A	Sized acute angle polygons, if any
OPOLY51R	Questionable polygons

Output cells *OCONT50A*, *OCONT50R*, *OPOLY51A*, and *OPOLY51R* are not created if they contain no sized data.

SIZENOT

SIZENOT = *cellnames*

Description

Keeps alignment cells from being sized for fracturing purposes. This command is for flat mode checking only and works with LOGICAL, SIZE/GROW, and RELOCATE commands only. When used with SIZENOT, these commands must follow this sequence in the rules file:

```
*OPERATION
LOGICAL commands / RELOCATE commands
SIZE/GROW/SIZE commands
*END
```

If you use SIZENOT, you cannot use the BULK layer as the input layer for LOGICAL operations.

Arguments

cellnames

Names of the alignment cells that are not to be sized.

Example

```
*DESCRIPTION
SIZENOT = abc def
*END
.
*OPERATION
OR L1 L2 L4
RELOCATE L1 BY -5 0 L6 MAG=5
SHRINK L2 0.5 0.7 L22
SIZE L3 BY -0.75 L33 OUT C 1
*END
```

SKILL-LOADER

`SKILL-LOADER = user-skill-file-path`

Description

If you are using Design Framework II, this command lets you use your own SKILL functions for Pcell data.

Example

`SKILL-LOADER = userSkill.il`

SMART-LPE

SMART-LPE = SINGLENET/PARTIAL/CRITICAL/NO

Description

Runs selective LPE and PRE. If you want to extract the layout parasitics associated with a particular net or nets in a selected window, include this command with the SINGLENET option in the DESCRIPTION block.

When PARTIAL is used, the RC for the selected nets are extracted as SINGLENET option but only extract C for the other nets. The whole result of the whole chip is output. This allows the precise timing analysis for the interested nets and has a coarse concept on the remaining nets.

When CRITICAL is used, the elements of the whole chip and the RC for the selected nets are extracted. The result of the whole chip is output.

It is strongly recommended that the command sequence in the operation block is arranged as follows to avoid unexpected results. For LPE/HLPE

- Layer operations for connection and element definition

- Element definition block

- Layer operations for parasitic definition

- Parasitic definition block

- {lvs/erc check}

- The user-defined extraction statement

- lpeselect/spfselect for output

For PRE/HPRE

- Layer operations for connection and element definition

- Element definition block

- {lvs/erc check}

- Cut-term and reconnect block

- Element definition block (This is necessary for layout-only job, too)

- Layer operations for parasitic definition

Dracula Reference

Description Block Commands

Parasitic definition block

The user-defined extraction statement

Lpeselect/spfselect for output

Something that needs special attention is for a layout-only PRE/HPRE job. The element definition before the CUT-TERM command is necessary now in the new LPEQRY flow.

When Dracula runs selective LPE with SMART-LPE=SINGLENET, the source/drain of the selected loading devices can be connected to nodes that become floating in the resultant SPICE netlist. Use the FNODE-CONNECT command to connect these nodes to power or ground so that the timing analysis can be done with the SPICE output.

Arguments

SINGLENET

Implements the selected window when used with LPE-QUERY command. It applies to LPE and PRE.

PARTIAL

A whole circuit is output but selected nets with RC extraction and others with C extraction only.

CRITICAL

A whole circuit is output, but only the critical nets with RC extraction.

NO

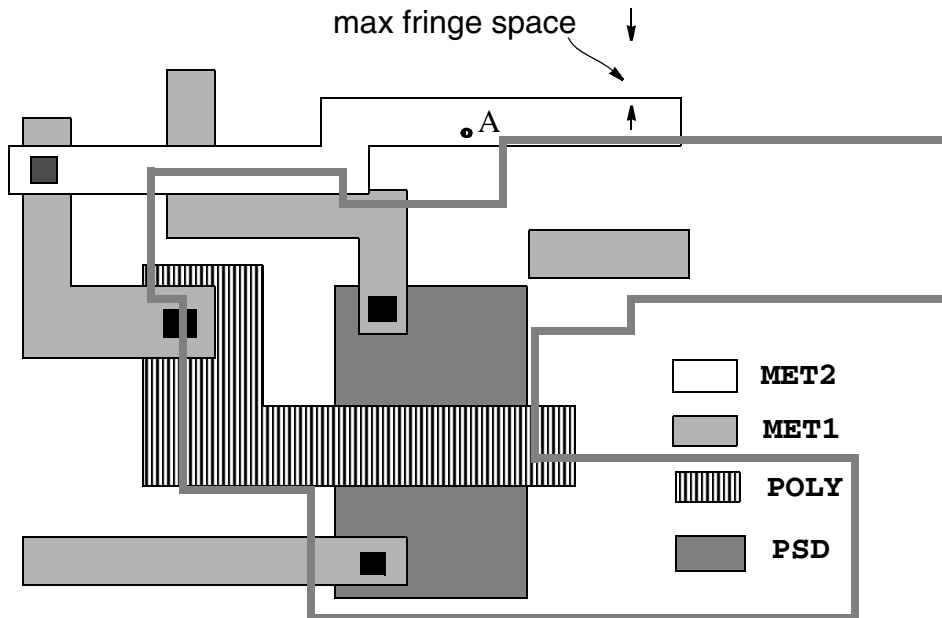
The default. The command is ignored.

Dracula Reference

Description Block Commands

Example 1

For the following layout, if the selected net is A, only the parasitics falling into the selected-window are read in and processed. This convention improves performance.



connect MET2 MET1 by VIA

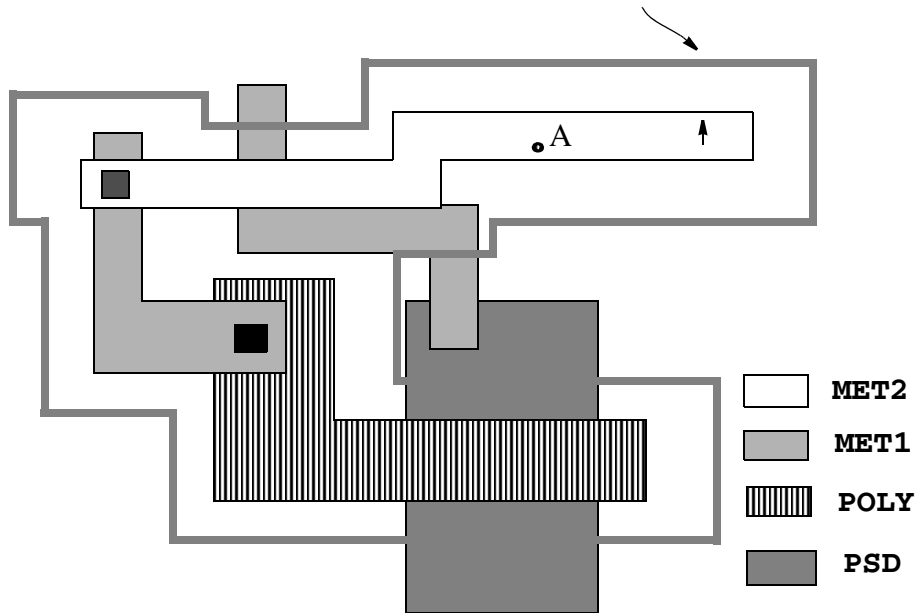
connect MET1 POLY by CONT

connect MET1 PSD by CONT

Dracula Reference

Description Block Commands

From the selected net, a selected window is formed by considering the fringe effect. Only the data in the selected window is used to extract the parasitics.



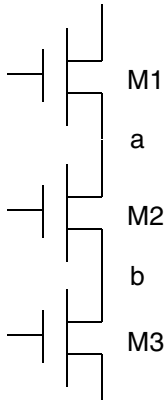
Example 2

```
*Desc
SMART-LPE=PARTIAL
LPE-QUERY= CORE
SELECT-MODE= LAY
*End
*Oper
...
NODE-SELECT= CK
LPESELECT[S] MOS &
LPESELECT RES &
LPESELECT CAP OUTPUT SPI
*End
```

Dracula Reference

Description Block Commands

Note: One thing to be noted is for the source and drain parameter extraction. The extracted result may be not consistent due to incomplete circuit extraction. See the following example.



If the selected node is 'a', the AS/AD/PS/PD for node 'a' of M1 and M2 will be correct. The AS/AD/PS/PD for node 'b' will be incorrect due to the fact that M3 is not considered in the process.

Example 3

```
*DESCRIPTION
SMART-LPE = CRITICAL
LPE-QUERY = ON
SELECT-MODE = SCM
*END
```

SMASH-CAP-TYP

SMASH-CAP-TYP = PARALLEL/SERIES/BOTH/YES/NO

Description

Smashes parallel and/or series capacitors of different subtypes, for example, CAP[N] and CAP[P].

In Dracula 4.81 and subsequent versions, the SMASH-CAP-TYP command offers new arguments that provide users with more control over smashing parallel and/or series capacitors.

Arguments

PARALLEL

Smashes only parallel capacitors of different subtypes.

SERIES

Smashes only series capacitors of different subtypes.

BOTH

Smashes both parallel and series capacitors of different subtypes.

YES

Works identical to the PARALLEL argument. Prior to Dracula 4.81, this is how the PARALLEL argument was implemented and has been retained for backward compatibility.

NO

Does not smash parallel and/or series capacitors of different subtypes. This is the default argument.

Note: If you use the PARALLEL or BOTH argument for this command, you must also use the *A argument in the LVCHK command*. Otherwise, PDRACULA will issue an error message.

Example

SMASH-CAP-TYP = SERIES

SMASH-FLOAT-RES

SMASH-FLOAT-RES = YES/NO

Description

Allows you to keep or smash parallel resistors with one floating terminal.

In versions prior to 4.81, Dracula will not smash these resistors. But, in Dracula 4.81 and subsequent versions, this new command lets you smash these resistors that have one floating terminal.

Arguments

YES

Lets you smash floating resistors

NO

Is the default and does not smash floating resistors

Example

SMASH-FLOAT-RES = YES

SMASH-MOS-TYP

SMASH-MOS-TYP = YES/NO

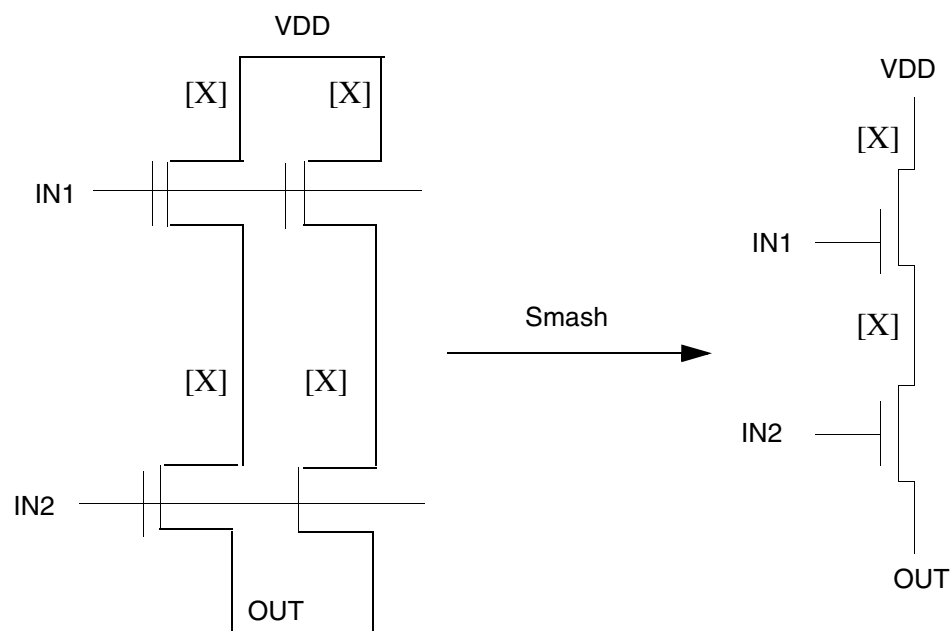
Description

Lets you smash parallel MOS transistors with split gates that use *different* device subtypes in Dracula LVS.

Previous versions of the Dracula product allowed you to only smash the split-gate serial chains of MOS transistors of the *same* subtype using the S option of the LVSCCHK command. But, this new command, SMASH-MOS-TYP now lets you smash parallel MOS transistors with split gates that use *different* device subtypes. This new feature works not only for 2 pairs of transistors, but for any number. The width or length ratio of the transistors with split-gate (which have the same input) should be equal to each other.

SMASH-MOS-TYP has two optional modifiers: CARE-SPLIT-ORDER and FIX-INPUT-ORDER and you need to use SMASH-MOS-TYP with one of these modifiers.

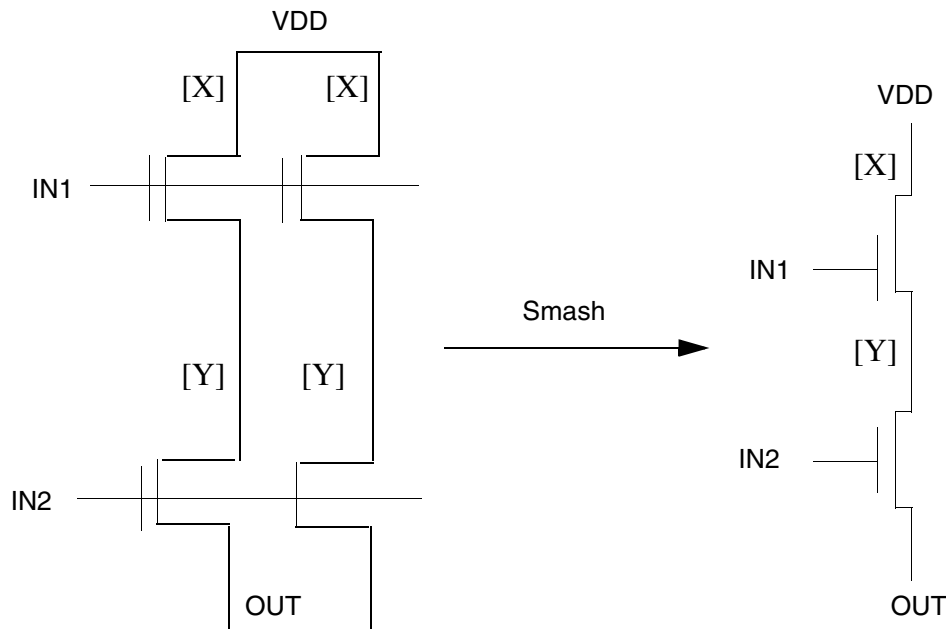
The figure below illustrates this functionality in previous versions of the Dracula product (where only MOS transistors of the same subtype are smashed). In this example below they are all [X] models.



Dracula Reference

Description Block Commands

The figure below illustrates the current enhanced feature where the bottom pair of MOS transistors can use a different model, [Y] as shown here. That is, a MOS device of a different subtype gets smashed.



Commands Combination Matrix Table

Given below is a compatibility table for these commands. This SMASH-MOS-TYP enhancement works *only* with these combinations given below.

Commands	Arguments		
CARE-SPLIT-ORDER	YES	-/NO	YES
FIX-INPUT-ORDER	YES	YES	-/NO
SMASH-MOS-TYP	YES	YES	YES
LVSCHK [S]	YES	YES	YES

PDRACULA translates this new command, SMASH-MOS-TYP for REDUCE to operate. REDUCE smashes the split-gates MOS devices with different subtype the same way as it smashes the split-gates MOS devices with the same subtypes.

Dracula Reference

Description Block Commands

PDRACULA handles the translation in three ways. PDRACULA either does the translation or issues a warning message based on whether you have

- Set the correct combinations in your rule file
- Set incorrect combinations in your rule file
- Set incomplete or conflicting rules in your rule file

Settings in Your Rule File

Description Block Commands

Operation Block Commands

Correct Combinations

■ `SMASH-MOS-TYP = YES`

`{LVSCHK[S]}`

and

■ `CARE-SPLIT-ORDER=YES`

or

■ `FIX-INPUT-ORDER=YES`

PDRACULA does the translation with the above combinations.

Incorrect Combinations

■ `SMASH-MOS-TYP = YES`

(without setting the S option in your LVSCHK command)

PDRACULA then issues the following warning message with the above combinations:

```
SMASH_MOS_TYP description command ignored
without specifying LVSCHK[S]
```

Dracula Reference

Description Block Commands

Settings in Your Rule File

Incomplete or
Conflicting Rules

Description Block Commands

■ SMASH-MOS-TYP = YES

and

■ CARE-SPLIT-ORDER=NO

or

You do not specify the CARE-SPLIT-ORDER command in your rule file

PDRACULA then issues the following warning message with the above combinations:

```
SMASH-MOS-TYP requires CARE-SPLIT-ORDER=YES.  
Previous setting "NO" has been overwritten.
```

Operation Block Commands

```
{LVSCHK[S]}
```

Arguments

YES

Lets you smash parallel MOS transistors with split gates that use *different* device subtypes.

NO

Is the default behavior.

Example

```
*DESCRIPTION  
CARE-SPLIT-ORDER = YES  
SMASH-MOS-TYP=YES  
...  
*END  
*OPERATION  
...  
LVSCHK{ [S] }  
*END
```

Note: Use special MOS element type ELEMENT MOS[K*] (where * is any alphanumeric character): specify the subtype of MOS CDL transistor card of type K* to keep it unsmashed.

SMASH-RES-TYP

SMASH-RES-TYP = PARALLEL/SERIES/BOTH/YES/NO

Description

Smashes parallel and/or series resistors of different subtypes, for example, RES[N] and RES[P].

In the 4.81 and subsequent versions, the SMASH-RES-TYP command provides new arguments that provide users with more control over smashing parallel and/or series resistors.

Arguments

PARALLEL

Smashes only parallel resistors of different subtypes

SERIES

Smashes only series resistors of different subtypes

BOTH

Smashes both parallel and series resistors of different subtypes

YES

Works identical to the SERIES argument. Prior to Dracula 4.81, this is how the SERIES argument was implemented and has been retained for backward compatibility.

NO

Does not smash parallel and/or series resistors of different subtypes. This is the default argument.

Note: If you use the SERIES or BOTH argument for this command, you must also use the *R* argument in the LVSCCHK command. Otherwise, PDRACULA will issue an error message.

Example

SMASH-RES-TYP = BOTH

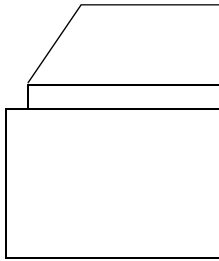
SMOOTH-SIZE

SMOOTH-SIZE = EDGE45 {CIRCLE}/CIRCLE/NO

Description

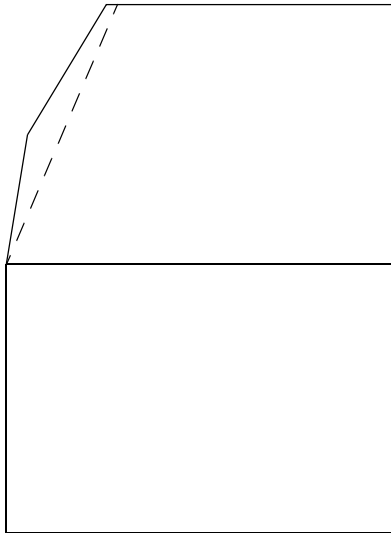
Snaps all non 45-degree edges resulting from the SIZE command back to 45-degree edges. For example, if you have a big polygon in your design and a cut line intersects a 45-degree angle, the angle becomes a non 45-degree angle. To snap the angle back to 45-degrees, specify SMOOTH-SIZE = EDGE45.

For example, the following shape occurs before sizing.



After performing the following SIZE function, a non 45-degree edge results.

```
SIZE LAY1 BY 5 LAY2
```



SMOOTH-SIZE = EDGE 45 snaps the edges to 45 degrees (represented by the dashed line).

Dracula Reference

Description Block Commands

If you specify `SMOOTH-SIZE = CIRCLE`, Dracula snaps all acute angles out to make smoother circles and curves. For example, if you enlarge and shrink a circle by a large value, the angle becomes a sharp angle (less than 10 degrees). `SMOOTH-SIZE = CIRCLE` snaps the angle out. This option is useful for circles or curves.

Arguments

`EDGE45`

Snaps all non 45-degree edges back to 45 degrees.

`EDGE45 CIRCLE`

Snaps all non 45-degree edges back to 45 degrees and smooths acute angles for circles or curves.

`CIRCLE`

Smooths acute angles for circles or curves.

If you specify `SMOOTH-SIZE = CIRCLE`, Dracula does not snap non-45 degree edges back to 45-degree edges.

`NO`

Does not snap angles back to 45 degrees. Does not smooth out acute angles for circles or curves. Default.

Example

The following example snaps non 45-degree edges to 45 degrees.

```
SMOOTH-SIZE = EDGE45
```

The following example smooths acute angles.

```
SMOOTH-SIZE = CIRCLE
```

S-OPTION-CLASS

`S-OPTION-CLASS = PWRGND/NONPWRGND/ALL`

Description

Determines how the LVSCCHK and LPECHK [S] option works. The [S] option can

- Smash MOS SUP1 split gates to single SUP devices and MOS SDWI split gates to single SDW devices
- Smash any split gate devices to single PMID or SMID devices

For descriptions of these device types, refer to [“LVS Overview”](#) on page 275. For descriptions of the LVSCCHK and LPECHK commands and their options, refer to [Chapter 13, “Operation Block Commands.”](#)

Arguments

`PWRGND`

Smashes MOS SUP1 split gates to single SUP devices and MOS SDWI split gates to single SDW devices. Default.

`NONPWRGND`

Smashes split gates that are not connected to power or ground to single PMID or SMID devices. You must also use the *O* option with LVSCCHK or LPECHK to allow gate swapping.

`ALL`

Performs the *PWRGND* and *NONPWRGND* options together. Smashes all split gates, whether they are connected to power, ground, or some other text. You must also use the *O* option with LVSCCHK or LPECHK to allow gate swapping.

Example

This example smashes all split gates not connected to power or ground to single PMID or SMID devices.

```
S-OPTION-CLASS = NONPWRGND
```

SPF-MAP-FILE

SPF-MAP-FILE = *filename* {NET}

Description

Specifies the SPF mapping filename for DSPF output files with the cell instance or net names in the naming convention of DFWII instead of Dracula naming convention.

Checking Methods

mode

Composite.

checks

HPRE.

Arguments

filename

Specifies the map filename output from the Cadence netlister.

NET

If this option is turned on, then you map both instance and net names in *filename* for the DSPF, RSPF file. Otherwise, you only map the instance name.

Example

SPF-MAP-FILE = CDL.run1/map/current

■ DSPF without SPF-MAP-FILE for Dracula naming convention.

```
*|NET flags<0> 8.61E-02PF
*|P (flags<0> I 0.00E00PF 27.11 0.30)
*|I (X_Mod030636:B X_Mod030636 B I 0.00E00PF 42.90 26.10)
*|I (X_Mod051177:A X_Mod051177 A I 0.00E00PF 69.30 22.50)
*|S (flags<0>:2 68.69 6.59)
*|S (flags<0>:3 42.29 7.19)
*|S (flags<0>:6 55.50 7.19)
R54 flags<0>:2 X_Mod051177:A 2.00000E-01
R56 flags<0>:3 X_Mod030636:B 2.00000E-01
```

■ DSPF with SPF-MAP-FILE specified for DFWII naming convention.

```
*|NET flags<0> 8.61088E-02PF
*|P (flags<0> I 0.00E00PF 27.11 0.30)
*|I (/mod030636:B /mod030636 B I 0.00E00PF 42.90 26.10)
```

Dracula Reference

Description Block Commands

```
*|I (/mod051177:A /mod051177 A I 0.00E00PF 69.30 22.50)
*|S (flags<0>:2 68.69 6.59)
*|S (flags<0>:3 42.29 7.19)
*|S (flags<0>:6 55.50 7.19)
R54 flags<0>:2 /mod051177:A 2.000000E-01
R56 flags<0>:3 /mod030636:B 2.000000E-01
```

START-MACRO

```
START-MACRO = macro-name/"macro-name arg1 arg2..."
```

Description

Executes the specified macro before any other commands execute. You can specify only one START-MACRO command.

Arguments

macro-name

The file name of the macro. Maximum file name is 128 characters.

arg1 arg2...

The macro arguments. Pass macro and its arguments with opening and closing quotation marks (" ").

Examples

This example places *csb abc* before BEGIN: in *jxrun.com*.

```
START-MACRO = abc
```

This example places *csb abc a1 a2 a3 a4* in *jxrun.com*, where *abc* is the macro and *a1-a4* are its arguments.

```
START-MACRO = "abc a1 a2 a3 a4"
```

This example compiles *csb move.com /cadence/data/lvslogic.dat* where *move.com* is a macro that contains the UNIX *cp* command and */cadence/data/lvslogic.dat* and "." are its arguments.

```
START-MACRO = "move.com /cadence/data/lvslogic.dat ."
```

STATUS-COMMAND

`STATUS-COMMAND = "command-name"`

Description

Executes a command you specify after each program module in the *jxrun.com* file. You can only specify one STATUS-COMMAND.

Arguments

"command-name"

Specifies the command or rules file name to execute after each program module. You must enclose the command and its arguments in quotation marks (" "). Maximum field is 128 characters.

Examples

These examples are for SUN/UNIX systems.

```
STATUS-COMMAND = "time"
```

```
STATUS-COMMAND = "time;du"
```


SUBCKT-NAME

SUBCKT-NAME = *name*

Description

Specifies the top subcircuit name in the SPICE output file produced by LPE. If you do not specify a subcircuit name, the default name is the file name in the OUTPUT portion of the LPESELECT command.

Arguments

name

Specifies the top subcircuit name. The name can be up to 32 characters.

Note: Because HSPICE only recognizes the first 16 characters, do not specify more than 16 characters if you intend to use the output for HSPICE simulation.

Example

In this example, the program uses WABPROC as the top subcircuit name in the LPE output file SPICE.DAT. If the SUBCKT-NAME command is missing, then SPICE is the default top subcircuit name for this example.

```
*DESCRIPTION
...
SUBCKT-NAME = WABPROC
...
*END
*OPERATION
...
LPECHK
LPESELECT MOS &
LPESELECT DIO &
LPESELECT CAP GT 0 OUTPUT SPICE
*END
```

SUBNODE-DELIM

SUBNODE-DELIM = *name*

Description

Specifies the delimiter in subnode names for PRE. Subnodes are created when a node is split to introduce parasitic resistors. A subnode name is a combination of the node name or node number, the delimiter, and a subnode number. Dracula uses a netlist name if you request backannotation.

The default delimiter is a colon (:).

Checking Methods

modes

Flat and composite.

checks

PRE and HPRE.

Arguments

name

Specifies a delimiter of one to four characters. This delimiter must be acceptable to your version of SPICE or to CDL to run circuit simulation using PRE extraction.

Example

```
SUBNODE-DELIM = "###"
```

Input schematic node names and Dracula-created subnode numbers are concatenated with the delimiter "###". For example, PA1 is represented as PA1###1, PA1###2, PA1###3, and so forth.

SUB-PROGRAM-NAME

`SUB-PROGRAM-NAME = moduleName directory/{hotfixModuleName}`

Description

Runs specific Dracula executable modules instead of the one that is normally executed by the `jxrun.com` file. Use this command to run *hotfix* modules you receive from Cadence between Dracula releases. You can have as many SUB-PROGRAM-NAME commands as needed in your command file.

Dracula also accepts the abbreviation SUB-PROG for this command.

Arguments

moduleName

Specifies the name of the Dracula module that is normally called in the `jxrun.com` file.

directory

Specifies the directory that contains the *hotfix* module.

hotfixModuleName

Specifies the name of the Dracula *hotfix* module. If omitted, the tool defaults to the *moduleName*.

Example 1

If the second argument is just a directory, PDRACULA concatenates the directory with the *moduleName*. For example

```
SUB-PROGRAM-NAME = SELECT /net/nickel/usr2/tmp/
```

executes the SELECT module in the `/net/nickel/usr2/tmp/` directory instead of the SELECT module that normally would be called by the `jxrun.com` file.

Note: Setting the command to

```
SUB-PROGRAM-NAME = SELECT /net/nickel/usr2/tmp/SELECT
```

accomplishes the same thing.

Dracula Reference

Description Block Commands

Example 2

If the second argument is a directory combined with a *hotfixModuleName*, PDRACULA overrides the *moduleName* with the *hotfixModulename* from the specified directory instead of the Dracula module that is normally called by the `jxrun.com` file. The following example executes the module `REDUCE.4.6.HOT` in the `/net/nickel/usr2/tmp` directory instead of the `REDUCE` module that is called by the `jxrun.com` file.

```
SUB-PROGRAM-NAME = REDUCE /net/nickel/usr2/tmp/REDUCE.4.6.HOT
```

Note: PDRACULA recognizes the second argument as being a directory based on the last character. If the last character of the second argument is a / (slash), PDRACULA treats the second argument as a directory. Otherwise, PDRACULA treats the second argument as a path to the *hotfix* module.

Example 3

The following example executes the module called `SIZE` from the directory `/local/cds/IC443/tools/dracula/bin` via the environment variable `path_to_dracula_binaries`. You have to first assign the physical location of the directory to the environment variable using the UNIX command `set` or `setenv`.

For example:

```
set path_to_dracula_binaries = /local/cds/IC443/tools/dracula/bin
```

Or you can use this command:

```
setenv path_to_dracula_binaries /local/cds/IC443/tools/dracula/bin
```

The `SUB-PROGRAM-NAME` command starts the `SIZE` modules using this environment variable starting with the \$ sign:

```
SUB-PROGRAM-NAME = SIZE $path_to_dracula_binaries/SIZE
```

SUBTYPE-CSENS

SUBTYPE-CSENS = YES/NO

Description

Specifies case-sensitive device subtypes for element, parasitic, and attribute definitions. If you specify SUBTYPE-CSEN = YES, devices such as MOS[N] and MOS[n] are treated as different devices. If you specify SUBTYPE-CSEN = NO, PDRACULA converts all subtypes to uppercase. If you do not use this command and you specify devices with the same subtypes with different cases, PDRACULA reports this as an error.

Arguments

YES

Specifies case-sensitive device subtypes.

NO

Converts all device subtypes to uppercase. Default.

Examples

```
*DESCRIPTION
....
SUBTYPE-CSEN = YES
....
*END
.....
*OPERATION
....
ELEMENT MOS[N] NGATE POLY NSD PWELL ; Standard NMOS
ELEMENT MOS[n] NSGATE SPOLY SNSD PWELL ; Special NMOS
....
*END
```

SUMMARY-ONLY

SUMMARY-ONLY = YES/NO

Description

Specifies that the OUTDISK command is not needed, and no OUTDISK is generated from the GDS2OUT stage.

Arguments

YES

Does not create an OUTDISK file.

NO

Generates the OUTDISK file.

Example

SUMMARY-ONLY = YES

SVS-LAYOUT

SVS-LAYOUT = *schematic-file*

Description

Specifies the layout data file generated by LOGLVS.

You must include this command in the Description block if you are running a schematic versus schematic (SVS) check. SVS also requires the SVS-SCHEMATIC command and, if you run hierarchical Dracula, the HCELL-COLUMN-1 command.

Arguments

schematic-file

Specifies the path and file name of the schematic data file generated by LOGLVS. The *schematic-file* directory must be different than the directory specified by the SVS-SCHEMATIC command. The default directory is the working directory.

Example

SVS-LAYOUT = /usr/cds/drac/design1

SVS-SCHEMATIC

SVS-SCHEMATIC = *layout-file*

Description

Specifies the schematic data file generated by LOGLVS.

You must include this command in the Description block if you are running a schematic versus schematic (SVS) check. SVS also requires the SVS-LAYOUT command and, if you run hierarchical Dracula, the HCELL-COLUMN-1 command.

Arguments

layout-file

Specifies the path and file name of the layout data file generated by LOGLVS. The *layout-file* directory must be different than the directory specified by the SVS-LAYOUT command. The directory cannot be the working directory.

Example

SVS-SCHEMATIC = /usr/cds/drac/layout1

SYSOUT

SYSOUT = format {treefile}

Description

Specifies the CAD-system format of the output tape (OUTDISK) if you want a format different from the one you specified in the SYSTEM or SYSIN command. For example, if SYSTEM = GDS2 and SYSOUT = CADENCE, the input is a GDSII database and the output error file is in CADENCE format.

This command requires a treefile generated by a Dracula job that ran with the format you want. This treefile contains information on the scale and physical unit of the format. If none is specified, it will default to using the treefile previously created by the Dracula job.

Arguments

format

Outputs in one of the following formats:

AGDS2

Annotated GDSII

In this format, PDRACULA performs an additional SIZE stage to convert trapezoid files to polygon files. A new module called GENANO outputs net attribute(23) (layout net names for connection layers and contact layers), and device attribute(17) (layout device names in output polygon files). The names must match the names that are output in the HSPIICE file.

GDS2OUT merges all of the output cell data from SIZE and forms a flat database. Except for connectivity, any text attached to the layout database or coming from an external file is output in annotated GDSII as well.

GDS2

GDSII

CADENCE

Cadence Design Framework II SKILL format

CADENCE_44DD

Cadence Design Framework II 4.4DD SKILL format

Dracula Reference

Description Block Commands

`{treefile}`

Specifies the treefile name from a previous run that specified SYSTEM and SCALE the same as you now specify SYSOUT and SCALEOUT (maximum eight characters, with the default extension *.DAT* assumed). If none is specified, it will default to using the treefile created by Dracula job.

OA

Cadence Design Framework II SKILL format.

OA22

OpenAccess 2.2 database.

SYSTEM or SYSIN

SYSTEM/SYSIN = *format*

Description

Specifies your CAD system and the format of your input and output graphic file. You must specify this command in the Description block.

For information about CAD-system database records supported, refer to [Appendix A, “Supported Database Records.”](#)

Arguments

format

Specifies one of the following CAD system formats:

GDS2

GDSII

CADENCE_44DD

Design Framework II database, CDSIN44

CADENCE

Is the same as CADENCE_44DD

Important

Certain limitations apply to the CADENCE database.

- ❑ If you specify CADENCE format, you must use the LIBRARY command described in this chapter.
- ❑ You must run the DFII license server, *cdsd*, on each system from which Dracula accesses the DFII design library.

OA

OpenAccess 2.0 database.

OA22

OpenAccess 2.2 database.

 Important

Certain limitations apply to the OA22 database.

- ❑ If you specify `OA22` format, you must use the `LIBRARY` command described in this chapter.
- ❑ `OUTDISK` command has no effect.
- ❑ You can specify output top cell using `OUTCELL` command.

Examples

```
SYSTEM = GDS2
SYSTEM = OA22
- - - -
SYSIN = GDS2
SYSIN = OA22
- - - -
```

TEXT-HEIGHT

TEXT-HEIGHT = *size {units}*

Description

Specifies height for text in OA22 database. Used only when SYSOUT (or SYSTEM) is OA22.

Arguments

size

Text height in MIL or MIC. If size is not divisible by resolution it is rounded. If size is smaller than 6 DBU, a warning is issued. If TEXT-HEIGHT is not set, default value of size 1.0 used.

units

Specifies the units of *size*. Must match units in SCALE and RESOLUTION (microns or mils).

Examples

TEXT-HEIGHT=100

TEXT-LEVEL

`TEXT-LEVEL = n1:n2 / n`

Description

Specifies the number of levels of hierarchy from which to read text. This command applies to cell and composite texts. Count the depth levels from the primary cell or level 0.

Note: TEXT-LEVEL = 0 has the same effect as the TEXT-PRI-ONLY = YES command.

Arguments

n1:n2

The range of levels of hierarchy from which Dracula reads text.

n

The range from 0 to *n* levels of hierarchy from which Dracula reads text.

Examples

In this example, Dracula reads text from the primary cell.

`TEXT-LEVEL = 0`

In this example, Dracula reads text from level 0 to level 3.

`TEXT-LEVEL = 3`

In this example, Dracula reads text from level 3 to level 5.

`TEXT-LEVEL = 3:5`

In this example, Dracula reads text from level 3 only.

`TEXT-LEVEL = 3:3`

TEXT-PRI-ONLY

TEXT-PRI-ONLY = YES/NO

Description

Processes only the text associated with the top-level (primary) cell in the layout. This command ignores text associated with cells nested below the top-level cell, even if the text is on the same text layer number. If you do not specify TEXT-PRI-ONLY, Dracula places all text at the top level and processes it.

Arguments

YES

Processes text associated with the top-level cell only.

NO

Processes all text from all cells. Default.

Example

This example processes only text associated with the top level and uses the primary cell structure *chip*.

```
TEXT-PRI-ONLY = yes  
PRIMARY = chip
```

TNAMES-CSEN

TNAMES-CSEN = YES/NO

Description

Specifies text name case sensitivity for all commands that contain text names. These include, but are not limited to, POWER-NODE, GROUND-NODE, PULL-UP, PULL-DOWN, PROBE, SELECT LABEL, LCONNECT, and ECONNECT.

This command affects all text names you specify in the layout and in EDTEXT or HEDTEXT files.

To specify both text name and cell name case sensitivity, you must also specify the C NAMES-CSEN command.

Arguments

YES

Passes the exact name and character cases from the text files to the *jxrun.com* file.

NO

Converts all text names to uppercase.

If TNAMES-CSEN = YES and C NAMES-CSEN = NO, Dracula converts cell names to uppercase and passes the exact text names to the *jxrun.com* file.

If TNAMES-CSEN = YES and C NAMES-CSEN = YES, Dracula passes the exact name and character cases from the commands and text file to the *jxrun.com* file.

If TNAMES-CSEN = NO and C NAMES-CSEN = YES, Dracula passes the exact name and character cases from the commands to the *jxrun.com* file and converts the text names to uppercase.

If TNAMES-CSEN = NO and C NAMES-CSEN = NO, Dracula converts all cell and text names to uppercase.

TRANSISTOR

TRANSISTOR = *number*

Description

Pre-allocates more swap space at the start of every Dracula module depending on the expected number of transistors in the layout. This command is the counterpart to the TRANSISTOR command for LOGLVS.

If you are running a very large job and more swap space is needed during execution of a module, that module must stop, request more swap space from the system, and execute from the beginning again. If you allocate enough additional swap space the first time, the above procedure repeats itself until sufficient swap space is allocated to complete the module. Therefore, the TRANSISTOR command saves valuable time.

Arguments

number

Specifies the number of transistors you expect in the layout. Default is 100,000.

Example

TRANSISTOR = 2000000

UNIT

`UNIT = (parameter-name, parameter-unit...)`

Description

Associates natural units to parameters extracted by LPE. These unit names appear in the SPICE-format listing generated by LPE. If you do not specify the unit name, Dracula reports only the parameter. You can repeat this command as needed.

Arguments

parameter-name

Specifies the parameter. Use only the following names:

CAPACITANCE
AREA
PERIMETER
RESISTANCE

parameter-unit

Specifies the unit name of the parameter used in the circuit simulation.

PF = P = 1E-12
F = 1E-15
N = 1E-9
U = 1E-6
M = 1E-3
K = 1E3

Example

`UNIT = CAPACITANCE,PF AREA,P PERIMETER,U RESISTANCE, K`

Note: Dracula appends only parameter units to parameters extracted by LPE. Sage™ or SPICE processes these units.



Caution

Because you can specify the value for the `ATTRIBUTE` command in scientific notation, be careful when you use the `UNIT` command. LPE appends the parameter unit you specify with the `UNIT` command to the capacitors in the SPICE-format listing (generated by the `LPESELECT` command). Some circuit simulators, such as standard SPICE, do not handle mixed numbers, for example, `C1 5 15 2E-1PF`. If you do not want PF in the capacitor command, do not specify a `UNIT` command.

UNSPEC-LAY-PARA

UNSPEC-LAY-PARA = YES/NO/EXCEPT-SUBSTRATE

Description

Reports unspecified layout parameters of matched devices in the LVS report file. Dracula considers the unspecified parameters of matched devices as errors and reports them as a mismatch. LVS does not compare the parameters between the devices if the parameter was not specified in the netlist or layout.

Dracula reports the total number of unspecified parameters in the .lvs summary file. The UNSPEC-LAY-PARA command reports these discrepancies with each device. Use the EXCEPT-SUBSTRATE command if you do not want Dracula to report unspecified substrates.

Arguments

YES

Considers an unspecified parameter as an error and reports the error as UNKNOWN. You can identify the devices that do not have their layout parameters specified.

NO

Default.

EXCEPT-SUBSTRATE

Suppresses unspecified SUBSTRATE warnings.

UNSPEC-PARA

UNSPEC-PARA = YES/NO/EXCEPT-SUBSTRATE

Description

Reports unspecified parameters of matched devices in the LVS report file. Dracula LVS considers the unspecified parameters of matched devices as errors and reports them as a mismatch. LVS does not compare the parameters between the devices if the parameter was not specified in the netlist or layout.

Dracula reports the total number of unspecified parameters in the `./lvs` summary file. The UNSPEC-PARA command reports these discrepancies with each device. Use the EXCEPT-SUBSTRATE command if you do not want Dracula to report unspecified substrates.



Do not use this command simultaneously with the UNSPEC-LAY-PARA or UNSPEC-SCH-PARA commands.

Arguments

YES

Considers an unspecified parameter as an error and reports the error as UNKNOWN. You can identify the devices that do not have layout or schematic parameters specified.

NO

Default.

EXCEPT-SUBSTRATE

Suppresses unspecified SUBSTRATE warnings.

Dracula Reference

Description Block Commands

Sample Report

This sample report shows how the .lvs report appears with *UNSPEC-PARA* warnings.

```
***** DISCREPANCY 1 *****
DEV1      MOS P  ----  M01      :  DEV1      MOS P      X=0      Y=0
105, VCC, 103                    105, VCC, 103
      W = UNKNOWN      L = UNKNOWN      W = 10.40      L = 8.20
SUBSTRATE = UNKNOWN              SUBSTRATE = UNKNOWN

***** DISCREPANCY 2 *****
DEV2      MOS N  ----  M02      :  DEV2      MOS N      X=100     Y=20
106, 104, 103                    106, 104, 103
      W = UNKNOWN      L = UNKNOWN      W = 10.40      L = 8.20
SUBSTRATE = UNKNOWN              SUBSTRATE = UNKNOWN
```

UNSPEC-SCH-PARA

UNSPEC-SCH-PARA = YES/NO/EXCEPT-SUBSTRATE

Description

Reports unspecified schematic parameters of matched devices in the LVS report file. Tells Dracula to consider the unspecified parameters of matched devices as errors and reports them as a mismatch in the LVS report file. LVS does not compare the parameters between the devices if the parameter was not specified in the netlist or schematic.

Dracula reports the total number of unspecified parameters in the `./vs` summary file. The UNSPEC-SCH-PARA command reports these discrepancies with each device. Use the EXCEPT-SUBSTRATE command if you do not want Dracula to report unspecified substrates.

Arguments

YES

Considers an unspecified parameter as an error and reports the error as UNKNOWN. You can identify the devices that do not have their schematic parameters specified.

NO

Default.

EXCEPT-SUBSTRATE

Suppresses unspecified SUBSTRATE warnings.

WINDEL

`WINDEL = xmin ymin xmax ymax`

Description

Deletes an area within a window from processing to conserve system resources. For CHECK-MODE = FLAT or HIER, Dracula keeps data that lies across the WINDEL coordinate lines until it encounters the next trapezoidal cut line inside the WINDEL area unless you also specify WINDOW-CUT=EDGE.

When checking using *CELL* or *COMP*, the layout that lies across WINDEL coordinate lines is cut. Only the portions outside the WINDEL coordinate lines are kept. This separation occurs unless you specify WINDOW-CUT = TRAPEZOID in your run file.

You can specify up to 50 WINDEL areas.

WINDEL with Hierarchical Dracula

The WINDEL command works only on the composite plane in hierarchical or composite mode. WINDEL is ignored in cell mode and on the cell plane of hierarchical or composite mode.

Arguments

xmin

Specifies the minimum x coordinate of window to delete from processing.

ymin

Specifies the minimum y coordinate of window to delete from processing.

xmax

Specifies the maximum x coordinate of window to delete from processing.

ymax

Specifies the maximum y coordinate of window to delete from processing.

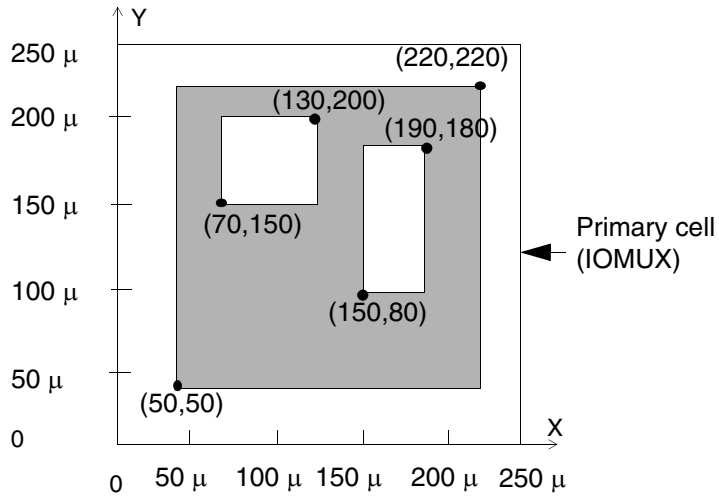
Example

```
PRIMARY = iomux
WINDOW = 50. 50. 220. 220.
```

Dracula Reference

Description Block Commands

WINDEL = 150. 80. 190. 180.
WINDEL = 70. 150. 130. 200.



WINDOW

`WINDOW = xmin ymin xmax ymax`

Description

Specifies a window to process. For CHECK-MODE = FLAT or HIER, Dracula keeps data that lies across WINDOW coordinate lines until it encounters the next trapezoidal outline outside the WINDOW area unless you specify WINDOW-CUT = EDGE in your run file.

When checking using CHECK-MODE = CELL or COMP, the layout that lies across window coordinate lines is cut. Only the portions inside the window coordinates are kept. This separation occurs unless you also specify WINDOW-CUT = TRAPEZOID in your run file.

You can specify multiple window areas (maximum of 50). The default for WINDOW is the entire primary cell. Dracula issues the following fatal error message if you specify a window outside the substrate layer:

```
*/FE*** FATAL ERROR DETECTED: NO DATA HAVE BEEN FOUND
-- PLEASE CHECK DATABASE AND WINDOW SIZES.
```

WINDOW with Hierarchical Dracula

The WINDOW command works on the composite plane in hierarchical or composite mode only. Dracula ignores WINDOW in cell mode and on the cell plane of hierarchical or composite mode.

Arguments

xmin

Specifies the minimum x coordinate of window to process.

ymin

Specifies the minimum y coordinate of window to process.

xmax

Specifies the maximum x coordinate of window to process.

ymax

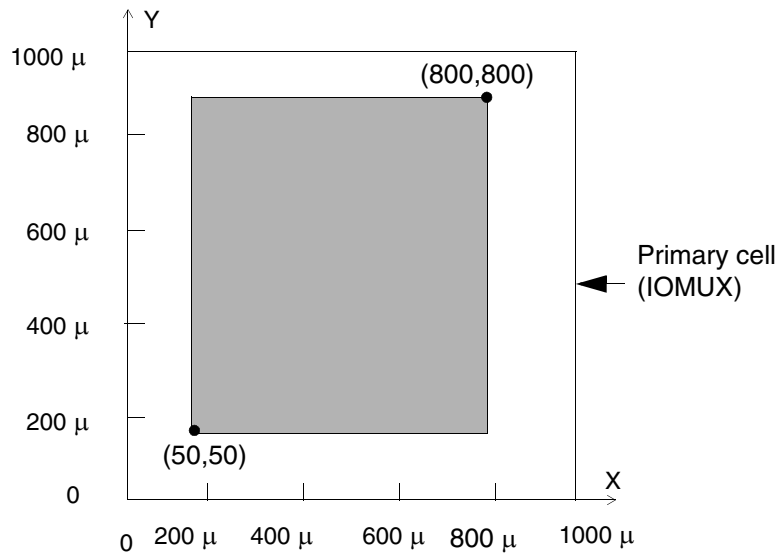
Specifies the maximum y coordinate of window to process.

Dracula Reference

Description Block Commands

Example

```
PRIMARY =iomux  
WINDOW =50. 50. 220. 220.  
WINDOW =220. 220. 400. 400.  
WINDOW =800. 800. 1000. 1000.
```



WINDOW-CUT

WINDOW-CUT = TRAPEZOID/EDGE

Description

Keeps or cuts the portion of a trapezoid that falls across the coordinates specified in a WINDOW, WINDEL, or HCELL command with the FRAME BY option.

Arguments

TRAPEZOID

Keeps the trapezoid if it falls across the window boundary. This is the default for flat and hierarchical mode.

EDGE

Cuts the trapezoid along the window edge and keeps the portion that fulfills the criteria (that is, the portion inside the window if you specified the WINDOW command or the portion outside the window if you specified the WINDEL command, or HCELL with FRAME BY option). Default for cell and composite modes.

WORK-DIR

WORK-DIR = *directory-name*

Description

Specifies the run directory. Dracula places all files generated by the job in this directory or subdirectory. You must enter the full name when you specify this command.

In Dracula 4.81 and subsequent versions, PDRACULA now adds the current directory to the INDISK and/or OUTDISK name if you set the following:

- Specify the WORK-DIR directory
- Leave INDISK and/or OUTDISK as just a file name (without a forward slash (that is, /)

Important

In this release and subsequent releases, you must specify the WORK-DIR command *before* the “Input-Layer Block” section of your Dracula rules file.

If you specify the WORK-DIR command *after* the “Input-Layer Block” section of your Dracula rules file, PDRACULA issues a warning message similar to this example below:

```
** ERROR : WORK-DIR SHOULD BE SPECIFIED BEFORE *INPUT-LAYER SECTION
      THE ERROR MESSAGE LISTED ABOVE IS FOR THE FOLLOWING CARD:
WORK-DIR           = ./work1/
```

Important

You cannot use the WORK-DIR command in Distributed mode.

Arguments

directory-name

Specifies the directory name or an assigned logical name. The default is your current directory. The maximum command line length is 128 characters.

Example

```
WORK-DIR = DBAO:[TEST.DRC]
WORK-DIR = DBA1:[PROD]
```

Dracula Reference

Description Block Commands

You cannot run more than one job at a time from your current or default directory even if you use different work directories. Even though Dracula runs in the working directory, it is run from files in your default directory. Those files can only run one job at a time.

The following files are either required by or generated in the default directory and a working directory. The working directory contains the INDISK, OUTDISK, and EDTEXT files unless you specify path names.

Default directory	WORK-DIR directory
<i>JXRUN.COM</i>	<i>INDISK file</i>
<i>JXPRECHK.COM</i>	<i>OUTDISK file</i>
<i>JXSUB.COM</i>	<i>EDTEXT file</i>
<i>printfile.SUM</i>	<i>SCHEMATIC file</i>
<i>printfile.LVS</i>	<i>JXGOUT.COM (for Dracula Interactive)</i>
<i>printfile.LOG</i>	<i>JXSORT.COM (for Dracula Interactive)</i>

System Info

System name	Modify the command
UNIX	WORK-DIR = /usr/drc1/

ZERO-SPAC-F-EQU

ZERO-SPAC-F-EQU = YES/NO

Description

Used to calculate zero and nonzero spacing capacitance with an equation. The program displays a warning message if "divided by 0" is found in the equation.

Arguments

YES

Turns on the function as described above.

NO

This is the default and will be ignored.

Example 1

Both the zero and nonzero spacing fringe capacitance is calculated with the following equation.

```
*DESCRIPTION
```

```
...
```

```
ZERO-SPAC-F-EQU =YES
```

```
*END
```

```
*OPERATION
```

```
FRINGE CAP[A] ME1 ME2
```

```
ATTRIBUTE CAP[A] 0 1
```

```
ATTRIBUTE CAP[A] .5 1
```

```
Lextract CAPF ME1 ME2 BY CAP[A] LCAPA &
```

```
EQUATION C = WIDT / SQRT((((3.205 + DEPT)**2.861/6.259)**2) + 57.5)
```

```
*END
```

Dracula Reference

Description Block Commands

Example 2

Without this command, Dracula uses $C=K*WIDT$ as the equation for zero spacing fringe capacitance and the following equation

$$C = WIDT / \text{SQRT}(((3.205 + DEPT)**2.861/6.259)**2) + 57.5)$$

for nonzero spacing fringe capacitance.

*OPERATION

FRINGE CAP[A] ME1 ME2

ATTRIBUTE CAP[A] 0 1

ATTRIBUTE CAP[A] .5 1

LEXTRACT CAPF ME1 ME2 BY CAP[A] LCAPA &

EQUATION C = WIDT / SQRT(((3.205 + DEPT)**2.861/6.259)**2) + 57.5)

*END

Dracula Reference

Description Block Commands

Input-Layer Block Commands

Chapter 12 discusses the Input-Layer Block Commands in alphabetical order.

Introduction

This section describes how you enter layer names to correlate with layout layer numbers. You define these layer names in the Input-Layer block. You can create as many Input-Layer blocks as you need as long as you do not use different layer names to represent the same layout layer. The preprocessor concatenates the individual names and checks their syntax.

Important

The INPUT-LAYER block section has a problem with handling more than 51 layer files at a time. When the INPUT-LAYER block section on the Dracula card is set to something as given below, two *LAYER1* layers (shown below) should automatically be merged, and become a single data.

```
LAYER1 = 1 DATATYPE = 0
LAYER2 = 2 DATATYPE = 0
.
.
.
LAYER1 = 9 DATATYPE = 1
```

However, if the *LAYER1* layers are located too far, it will be aborted. The Dracula card can read in only a maximum of 51 layers at once. When this number is exceeded, the layers fail to be merged. This problem can be resolved by reorganizing the order of layer entries such that entries of the same layer name will be handled in the same pass.

Using Text

Dracula requires that you attach text strings to layout geometries during ERC, LVS, LPE, PRE, and consistency checks. Text strings identify circuit components such as the power

Dracula Reference

Input-Layer Block Commands

supply, ground, clock, or pads. Text also identifies nodes and virtual connections and defines circuit elements. Dracula uses text for the following purposes:

- To specify the power supply and ground nodes for ERC, LVS, LPE and PRE

The text string denoting the power supply can be VCC, VDD or XXX:P. The text string denoting the ground node can be VSS, GND, GROUND, or XXX:G. For more information, refer to the [Text Conventions](#) section in this chapter.

- To specify nodes that you probe with the PROBE command

You can probe only text-labeled nodes. For more information, refer to [Chapter 13, “Operation Block Commands.”](#)

- To specify a virtual connection or wire between separate geometries

For example, you can handle two physically separate VCC pads as if they were part of the same node. Thus you can check a partial layout, such as a single cell or a layout excluding the memory arrays. A colon on the end of a text string indicates a virtual connection, for example, VCC: to VCC: or XYZ: to XYZ:.

- To denote correspondence points for LVS

Each point corresponds to one logic schematic node of the same name. For example, VCC to VCC, VSS to VSS, and CLOCK to CLOCK. The greater the number of correspondence points, the more effective the LVS check. As a guideline, you must label all pads. You also need to label critical internal nodes and nodes in highly parallel circuits.

Permissible characters and syntax rules are listed in the [Text Conventions](#) section.

In an Applicon database, text strings are already attached to the geometries. Typically, you define and enter text strings at the polygon level. Dracula recognizes only those text strings at the layers and fields that you specified in the Input-Layer block. In the following example, the *metal* layer (geometry layer 9) has a *text* layer (text layer 60) attached to it.

```
*INPUT-LAYER
metal = 9 TEXT 60
*END
```

In GDSII, you do not directly attach text strings to layout geometries. Text strings are part of a separate layer and do not have pointers back to polygons. Dracula attaches text to its corresponding geometry according to your definition in the Input-Layer block. For example,

```
*INPUT-LAYER
diffus = 2
poly = 8
metal = 9 TEXT 60
text = 20 ATTACH metal
CONNECT-LAYER = diffus poly metal
PAD-LAYER = vapox
*END
```

Dracula Reference

Input-Layer Block Commands

In this example, the text on layer 60 is not specifically attached to one layer, so it attaches to one of the connect layers. The text on layer 20 can attach only to layer *metal*.

Dracula attaches text in the reverse order of the CONNECT-LAYER, where the metal layer has the highest priority and the diffusion layer the lowest. In the previous example, text strings with coordinates within the metal geometries are attached to the metal layer and eliminated from further attachments. All remaining text in the polysilicon is attached. Finally, the text in the diffusion is attached and Dracula discards any text outside the three layers.

To use a text coordinate file, you can enter text directly using the EDTEXT command in the Operation block. For more information about the EDTEXT command, refer to [Chapter 13, “Operation Block Commands.”](#)

Texting Mechanism

For LVS and some DRC, ERC, and LPE checks, you must label some electrical nodes in the database with the texting mechanism. The correct use of text is critical, especially for LVS and ERC checks.

The main way to enter text into the Dracula database is with a CAD graphics system. However, because CAD graphics systems let you enter text that is not meaningful to Dracula (such as cell text), you must establish a screening procedure to put text on reserved text layers or text types.

The second way to enter text into the database is to create a text coordinate file. This file contains text and text origins (x,y coordinates in natural units). Whenever both the CAD graphics system and the text coordinate file have different text at the same location, Dracula discards the text from the CAD graphics system.

Text Conventions

Dracula text can incorporate any of the characters shown in the following table. Dracula converts any other character except the colon (:) to a question mark (?) and treats a question mark as a text delimiter. The text is truncated at that point. There is no limit to the number of characters you can specify on one line.

A to Z	Letters	.	Period
0 to 9	Numbers	=	Equal sign
+	Plus sign	<	Left angle bracket
-	Minus sign	>	Right angle bracket

Dracula Reference

Input-Layer Block Commands

/	Slash sign	_	Underscore
(	Left parenthesis	'	Apostrophe
)	Right parenthesis	\$	Dollar sign
*	Asterisk	?	Wild card
[	Opening bracket	&	Ampersand
]	Closing bracket	%	Percent sign
@	"At" sign		

The following rules apply to text:

- A layer name cannot begin with *temp* or *out*. For example, *temp1* and *outmet* are not allowed. The preprocessor checks for these syntax violations.
- The first character must be a letter and can be in uppercase or lowercase.
- To denote a virtual wire, add a colon (:) after the text.
- To denote a pad or a power or ground, add a colon (:) and one pad character after the text. The pad character can be one of the following:

If the text is VCC or VDD, Dracula assigns the *P* pad type.
If the text is VSS, GND, or GROUND, Dracula assigns the *G* pad type.
If the text is VEE or VBB, Dracula assigns the *S* pad type.
- Use LVS-required pad text for initial correspondence only. You can attach text to the pad, metal, or any other connect layer. For ERC, you must define the pad as an ELEMENT PAD and text is optional, except for power and ground pads.
- Text suffixed with a colon (:), such as ABC:I and ABC:O, is virtually wired even when there is no physical connection. Be cautious when using the virtual wire specifier in ERC, LVS, or LPE.
- Applicon texting requires that you attach the text to its own layer geometries. If the text is on the same layer as metal and you want metal attachment, the input layer and text layer are as follows:


```
metal = 6 TEXT = 6
```
- Be sure you do not label two nodes with the same text string unless you use a colon; otherwise, you create an open circuit.
- You can label a node more than once with the same text string. Be sure not to label the same node with two different text strings. Otherwise, you create a shorted circuit.

Dracula Reference

Input-Layer Block Commands

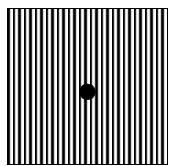
- You can specify a text layer declaration on a different line if the input-layer and text-layer are on different layers.

In the following example, metal is read in from layer 6, and layer 4 is read in as *dummy*. Text from layer 8 is read in and processed.

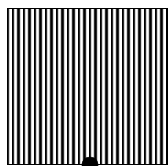
```
metal = 6
dummy = 4
TEXT = 8
*OPERATION
AND metal dummy abc
```

Attaching Text to a Geometry

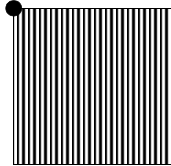
You can attach text to a geometry when the text origin (x,y location) is inside, at the edge, or on the vertex of the geometry, as shown in the following figure.



Inside



At the edge



On the vertex

You can attach text to geometries from one of many layers. The text is attached to geometries on only one layer according to the text-layer sequence until it is finally attached or the text is discarded.

Specify text-layer sequences using the TEXTSEQUENCE command in the Input-Layer block in reverse order of the layer sequence. The layers you specify in TEXTSEQUENCE must be specified in the CONNECT-LAYER command. If you do not specify TEXTSEQUENCE, specify the text-layer sequence with the CONNECT-LAYER command in the reverse order of the layer sequence.

Labeling Electrical Nodes

If you attach text to a geometry that represents part of the layout data for an electrical node, the text becomes the label of the node. A node can carry one unique label, and no other node can carry that label. Therefore, you can identify a node by its label and you can identify a group of nodes by a group of labels. To refer to a group of labels for ERC and DRC checks, you can use a question mark (?) as a wildcard character. For example,

?AB	Refers to all the labels ending with AB

Dracula Reference

Input-Layer Block Commands

XY?	Refers to all the labels starting with XY
?NN?	Refers to all the labels having the substring NN
? : P	Refers to all the power pads
? : ?	Refers to all the pads

Text from a CAD Graphics System

To input text from a CAD graphics system, specify the layer definition in the Input-Layer block on a separate line. You can also specify a text layer by appending the text to a geometry layer definition.

```
name1 = layer1 TEXT = layer2 TEXTTYPE = number ATTACH = name2
```

<i>name1</i>	Specifies the name of the source geometry layer
<i>layer1</i>	Specifies the source geometry layer number in CAD graphics system
<i>layer2</i>	Specifies the text layer number in CAD graphics system
<i>number</i>	Specifies the text type number in CAD graphics system
<i>name2</i>	Specifies the name of geometry layer to which text is attached

TEXTTYPE and ATTACH commands are optional.

If you specify an attached layer name, texts are attached to that layer only.

If you do not specify an attached layer name, texts are attached to one of the layers in accordance with the CONNECT-LAYER or TEXTSEQUENCE command.

Example 1

This example shows how you specify text on one line.

```
METAL = 12 TEXT = 80 ATTACH = METAL1
```

Example 2

This example shows how you specify a text layer definition on a separate line.

```
POLY = 6  
TEXT = 12 TEXTTYPE = 6:12 ATTACH = METAL
```

Dracula Reference

Input-Layer Block Commands

Text for Power and Ground

Dracula recognizes the following power and ground names and converts them to pad types:

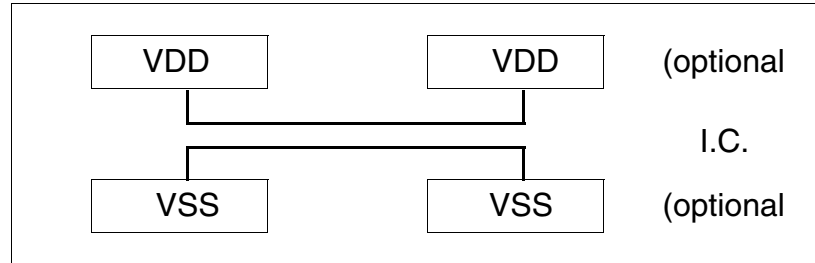
VCC	Converted to VCC:P
VDD	Converted to VDD:P
VSS	Converted to VSS:G
GND	Converted to GND:G
GROUND	Converted to GROUND:G
VEE	Converted to VEE:S
VBB	Converted to VBB:S

Dracula Reference

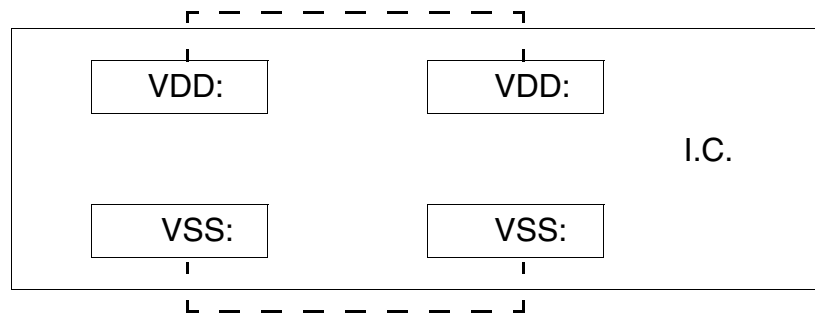
Input-Layer Block Commands

Texting Multiple Power Supply Pads

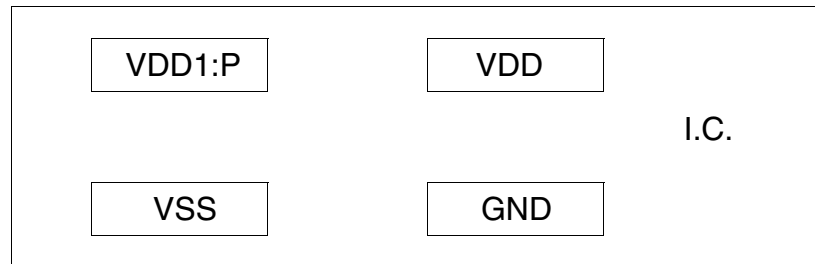
Internally
connected
VDD



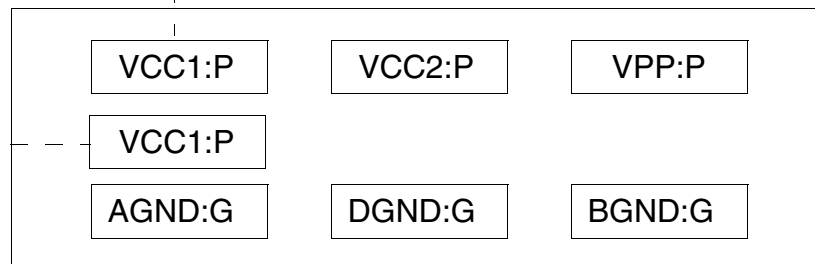
Externally
connected
VDD



Separate
power and



Separate power
and ground pad
names not
recognized by
Dracula. The
colon (:) also



BASE-LAYER

`BASE-LAYER = layer-name1 {layer-name2} {...}`

Description

Specifies layers containing composite plane geometries that must be mapped into Hcells.

Checking Method

`modes`

Cell, composite, Vampire.

`checks`

HLVS, HLPE.

Arguments

layer-name

The name of the layer defined in the Input-Layer block.

You can use this command for gate-array layouts, where composite plane layers (the base-layers) provide a set of symmetric geometries overlaid by Hcells that parameterize the semi-custom circuit with transistors, contacts, and other connections on one or two layers.

Composite mode HLVS cuts out the composite-plane geometry portions that lie within an Hcell boundary. Dracula then maps and merges these geometry portions into the Hcell allowing them to create devices. The corresponding Hcell subcircuits on the netlist must include whatever connectivity the composite-plane geometry portions on the layout side provide.

The BASE-LAYER command handles geometries that are not identical for different placements. Geometries that are the same for each placement move into the cell. Unique portions of the geometries stay at the composite level.

- If the geometries are needed to form devices in the cell, cell mode HLVS reports missing layout devices.
- If the geometries form extra (incorrect) devices in the cell, Dracula finds them in the extra device check and flags them as extra composite-to-cell devices.
- If the geometries are needed to connect nodes in the cell, cell mode HLVS flags these as errors.

Dracula Reference

Input-Layer Block Commands

- If the geometries connect nodes in the cell incorrectly, composite mode flags them either as unmatched composite-to-cell internal node connections or as errors in connectivity for cell pins.

Example

In this example, assume that PWELL and NSUB are in the composite layer. If PWELL and NSUB are not identified as BASE-LAYER, then all CMOS transistors in the Hcells do not have substrates. If PWELL and NSUB are identified as BASE-LAYER, then all MOS devices can have substrates, if the PWELL is under the N devices and the NSUB is under the P devices.

```
BASE-LAYER = PWELL, NSUB
```

CELLBNDY

CELLBNDY = *layer-number* {DATATYPE *data-type*}

CELLBNDY = AUTO [*value*]

Description

Specifies the input layer used to build cell boundaries. For hierarchical cells with boundaries defined in this layer, these boundaries override the bounding boxes generated by the program. If you do not specify CELLBNDY, Dracula derives the bounding boxes from the layers drawn within the cell.

The SUBSTRATE and CELLBNDY commands affect the formation of the substrate layer, which affects the formation of the reverse field layer. For HLVS and HLPE jobs, the connectivity of the reverse field layer can also be affected. For example

```
SUBSTRATE = BULK 63  
NOT BULK PWELL NSUB
```

The NSUB layer of a cell in flat mode and the NSUB layer of the same cell as an Hcell in cell mode might differ because of differences in the BULK layer. If this causes problems in connectivity, you can modify the cell boundary. You can use the CELLBNDY command to make the cell boundary larger than the Hcell default. To prevent creating open circuits, you can add virtual wire texting to connect them.

For more information, refer to the SUBSTRATE command section in this chapter.

When you specify *AUTO*, Dracula automatically generates a rectilinear cell boundary.

Checking Method

modes

Cell and composite.

checks

HDRC, HLVS, HERC, and HLPE.

Arguments

layer-number

Specifies the layer number on the layout database.

Dracula Reference

Input-Layer Block Commands

data-type

Valid for GDSII format or CADENCE. Default is all datatypes. You can also use an inclusive range value (for example, DATATYPE=7:10).

Use this command for cells with non-rectangular cell boundaries and gate arrays. For example, if only *metal* is digitized inside the cell and *diff* and *poly* are digitized at the composite level, the resulting devices are electrically inside the cell but are built from polygons digitized outside the bounding box.

You can specify only one boundary for each cell. The boundary can be any rectilinear polygon. Dracula creates bounding boxes as cell boundaries for Hcells that have no input boundaries specified in the CELLBNDY layer. The input boundaries always supersede the bounding boxes.

Running a job with this command creates the *9CELLBND.DAT*, *6CELLBND.DAT*, *7CELLBND.DAT*, and *CELLBNDY.DAT* files. These are binary files in Dracula formats.

In rectilinear cell applications, under normal circumstances, Dracula makes correct electrical connections and you do not need CELLBNDY. However, when you specify FRAME or GEN-TEXT-FRAME for a rectilinear cell or when you use BASE-LAYER in either cell or composite mode, you must specify CELLBNDY.

AUTO

Dracula automatically generates a rectilinear cell boundary.

value

The amount to oversize or undersize the boundary.

Example 1

This example reads in layer 59 as the cell boundary layer.

```
*INPUT-LAYER
PWELL=1
DIFFPN=2
POLY =3      CTEXT=28 ATTACH=POLY
CONT=5
MET=6        CTEXT=29 ATTACH=MET
VAPOX=7      TEXT=30
PISL=4
CELLBNDY = 59
SUBSTRATE = BULK 63
;
CONNECT-LAY = NSUB PWELL PSRCDRN NSRCDRN POLY MET
*END
```

Example 2

This example automatically creates a rectilinear cell boundary.

```
*INPUT-LAYER
.
.
.
CELLBNDY = auto
*END
```

CONNECT-LAYER

```
CONNECT-LAYER = layer1 layer2...
```

Description

Defines the conductor layers of the IC process from bottom to top. If you use a CONNECT command in the Operation block, you must use the CONNECT-LAYER command in the Input-Layer block. Specify only the layers that appear in the CONNECT commands (excluding the contact layers). You can specify multiple layer names on one line or use multiple lines.

For more information about connecting layers, see “[*Connecting Your Network*](#)” in the “Writing Rules for Dracula” chapter in the *Dracula User Guide*.

Arguments

layer

Specifies the mask order of the layers used in the CONNECT operation (excluding the contact layer). List the bottom layer first and the top layer last.

Layer names must have at least three characters. If you use the layer name in the ELEMENT or PARASITIC commands, the first three characters must be unique.

Text sequence is generated only on layer names that appear in both the CONNECT-LAYER and CONNECT commands.

Example 1

The first three characters of *met2* and *met1* are the same. If you use this example with an ELEMENT CAP command, this is a syntax violation because the first three characters must be unique. The preprocessor checks for this syntax violation.

```
CONNECT-LAYER = nsrdrn poly met1 met2
```

Example 2

```
CONNECT-LAYER = psub nwell nsd psd pl1 pl2 metal
```

or

```
CONNECT-LAYER = psub buried base nplus deepn mt1 mt2
```

or

Dracula Reference

Input-Layer Block Commands

```
CONNECT-LAYER = psub;bottom layer
CONNECT-LAYER = buried
CONNECT-LAYER = base
CONNECT-LAYER = nplus
CONNECT-LAYER = deepn
CONNECT-LAYER = mt1
CONNECT-LAYER = mt2;top layer
```

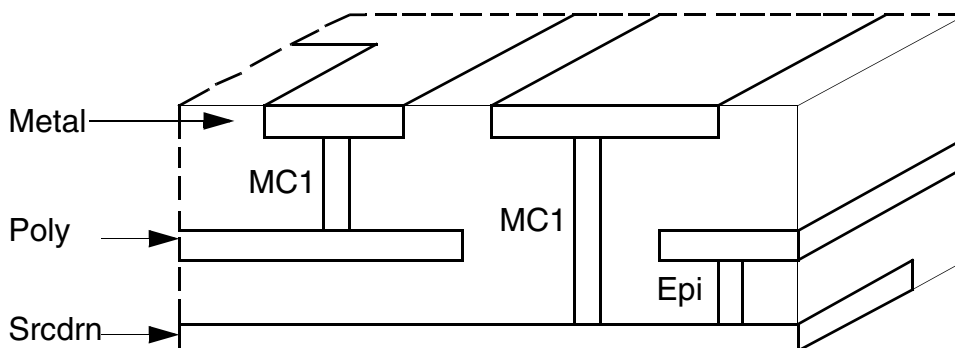
Specify the bottom layer as *layer1* followed by the next layers and end with the top layer, usually *metal*.

Example 3

```
*INPUT-LAYER
.
CONNECT-LAYER = srcdrn poly metal
.
*END
*OPERATION
.
CONNECT metal polyBY mcl
CONNECT metal srcdrnBY mcl
CONNECT poly srcdrnBY epi
```

You must use the CONNECT-LAYER command for ERC, LVS, and LPE because they require text. Dracula processes the text you specify in the Input-Layer block as follows:

- If you do not specify the TEXTSEQUENCE command and text falls inside (or touches) the geometries of the top layer (*metal* in the above example) in the CONNECT-LAYER command, Dracula assigns the text as the node name to the top-layer geometries. Otherwise, Dracula tests to see if it falls inside (or touches) the second topmost layer and assigns it to that layer (*poly*), and so on.
- If you specify the TEXTSEQUENCE command, Dracula uses the layer names and the order of the layers you specify.
- Each contact layer has a master layer. For example



Dracula Reference

Input-Layer Block Commands

The *MC1* contact layer has *metal* as its master layer. The contact layer can only connect from the master layer to other layers. Therefore, *MC1* can connect *metal* to *poly* or *metal* to *scrdn*, but *MC1* cannot connect *poly* to *epi*. The preprocessor checks for this violation.

Example 4

```
CONNECT-LAYER = nsd poly met1 met2
```

In the Operation block:

```
CONNECT met2 met1 BY cont
CONNECT met1 poly BY cont(not allowed)
CONNECT met1 nsd BY cont(not allowed)
```

The master layer is *mt2*. The *cont* contact layer cannot be used to contact *mt1* to *nsd* or *mt1* to *poly*.

CTEXT and ATTACH

CTEXT = text-layer layerPurpose {ATTACH = layer-name}

Description

Specifies the input layer numbers that provide Hcell text. The ATTACH command specifies the input layer to which the Hcell text on the CTEXT layer number is attached.

Without the ATTACH command, Hcell texts on the text-layer fall through the input layers in the order specified by the TEXTSEQUENCE or CONNECT-LAYER commands.

Note: Dracula can read the same layer for both TEXT and CTEXT, but not when using the ATTACH command. You must use different layers for composite and Hcell text.

Checking Method

modes

Cell and composite.

checks

HERC, HLVS and HLPE.

Arguments

text-layer

The number of the text layer from the layout database.

layerPurpose

The purpose for the layer, for example, *drawing*.

layer-name

The name of the layer of one of the connect layers.

The layer definition performs an independent function from the CTEXT command, even though they can be on the same line. It is not necessary to include the CTEXT command exactly on the layer to which it attaches. You can put CTEXT either in the left column or following a layer definition.

Dracula Reference

Input-Layer Block Commands

Example 1

```
*INPUT-LAYER
poly = 5      CTEXT = 27      ATTACH = poly
metal = 4
CTEXT = 19    ATTACH = metal
CTEXT = 18
```

Example 2

```
PDIFF = pdiff          TEXT = text drawing      ATTACH = metal
TEXT = text drawing    ATTACH = metal
NDIFF1 = ndiff1        CTEXT = text1 drawing
CTEXT = text2 drawing
```

EXPORT

EXPORT *layer-name*...

Description

Generates layers from an input database and outputs them as Dracula-compatible data files. You can use this command to combine layers from several databases or to compare layers from different databases. The EXPORT command ensures that Dracula does not delete the layers specified even if you use KEEPDATA=NO or KEEPDATA=SMART in the Description block. You do not need an Operation block when you use the EXPORT command.

You can use the EXPORT command to specify any *layer-name* if KEEPDATA=NO (or SMART) so that the preprocessor does not delete these *layer-names*. Thus, you can specify not to delete files and layer names (other than trapezoid layers). You can run an interactive program such as Dracula Interactive or Layout Parameter Extraction Post-processor (LPEPRO) even if you specified KEEPDATA=NO (or SMART).

See also the [IMPORT-LAYERS](#) section in this chapter.

Argument

layer-name

The user-defined layer name.

Example 1

```
*INPUT-LAYER
Diff      = 1
nplus     = 2
polyA     = 3
mcA       = 4
metalA    = 5
EXPORT polyA mcA metalA
*END
```

Example 2

```
*DESCRIPTION
.
KEEPDATA = smart
.
*END
*INPUT-LAYER
.
EXPORT   = 6nets poly nsd psd pwell nsub contact
```

Dracula Reference

Input-Layer Block Commands

```
EXPORT    = metal 6net1 6pads 6pads 6nxrf 6exrf
EXPORT    = mosxn8g mosxp8g ; for a CMOS process
.
*END
*OPERATION
.
LVSCHK
*END
```

Example 3

```
*DESCRIPTION
PRIMARY = TOPCELL
.
EXPORT = metal2 metall poly psd nsd
EXPORT = BOUNDARY SCRXTXT TREETOPC
```

The first run reads in and exports all the input layers needed. When the second job does not input any new layers and does not run GDS2IN, you should export the BOUNDARY, SCRXTXT, and TREExxxx (where xxxx is the first four characters of your primary name) files. These files are often needed by the second job.

FRAME-EXCLUDE-LAYER

```
FRAME-EXCLUDE-LAYER = layer1 layer2 .....
```

Description

Allows several specified layers not to be clipped by the FRAME BY command.

Note: This command applies only in composite mode.

Arguments

layer

Name of the specified obstruction layer defined in the hcells.

Example

```
*INPUT-LAYER
MT1      = 10  ;METAL ONE LAYER
MT2      = 11  ;METAL ONE LAYER
MT3      = 13  ;METAL ONE LAYER
M10BS    = 75  ;M1 P&R obstruction layers
M20BS    = 76  ;M2 P&R obstruction layers
M30BS    = 77  ;M3 P&R obstruction layers
FRAME-EXCLUDE-LAYER = M10BS M20BS M30BS
*END
*OPERATION
SIZE M10BS BY 0.5 OM10BS
SIZE M20BS BY 0.7 OM20BS
XCELL MT1 OM10BS OUTPUT ERR 1
XCELL MT2 OM20BS OUTPUT ERR 2
*END
```

FRAME-HOLE-LAYER

FRAME-HOLE-LAYER = LAYER

Description

Creates a new layer that is the same shape as the frame, as defined by FRAME BY commands in cell mode.

Arguments

layer
 Layer name.

Example

The following are examples of checking for the illegal presence of layer "CONT4" inside or cutting the frame hole in cell mode.

```
*INPUT-LAYER
CONT4= 4
FRAME-HOLE-LAYER= FL
*END
*OPERATION
SELECT CONT4 OVERLAP FL OUTPUT ERR 1
*END
```

GEN-TEXT-LAYER

```
GEN-TEXT-LAYER = layer-name1 {layer-name2} {...}
```

Description

Specifies the layers to which the Hcell text generated by GEN-TEXT is attached. If you do not specify the GEN-TEXT-LAYER command, GEN-TEXT defaults to the layers specified by the CTEXT and ATTACH commands. The layers specified by the CTEXT and ATTACH commands contain the original cell text from the database. The GEN-TEXT command attaches the text to all layers that qualify.

Checking Method

modes
Cell.

checks
HLVS.

Arguments

layer-name
The layer name of one of the connect layers.

You must ensure that GEN-TEXT, at a minimum, attaches cell text to all layers that can be used as pins (connecting to composite or other Hcells) or as feedthroughs. Otherwise, Dracula reports discrepancy errors when running composite mode HLVS.

Example

```
GEN-TEXT-LAYER = nsub pwell poly metal
```

IDTEXT

layer1 = IDTEXT *layerNumber*

Description

Lets you add text to layers for identification purposes. These layers are independent of connectivity. You can select polygons by *idtext* names from a particular *idtext* layer.

The IDTEXT command is useful only with the SELECT LABEL {[t] | [t']} command.

Arguments

layer1

The IDTEXT layer name in the rules file.

layerNumber

The layer number from which text is extracted for *layer1*.

Example

In this example, two IDTEXT layers, *idtxt1* and *idtxt2*, extract texts from input layers 1 and 2. There is text on layer 2 for connectivity purposes (for example, naming electrical nets).

```
*INPUT-LAYER
metal   = 1
idtxt1  = IDTEXT 1
poly    = 2 TEXT 2
idtxt2  = IDTEXT 2
cont    = 3
*END
*OPERATION
SELECT metal BY idtxt1 LABEL[t] {RES111,RES222,RES333} pad1 OUTPUT mpad 2
SELECT poly  BY idtxt2 LABEL[t'] RES? pad2
*END
```


IMPORT-LAYERS

```
IMPORT-LAYERS  path-name1/layer-name1, path-name2/  
               layer-name2...
```

Description

Combines layers from a previous run with the current job. Use this command to combine or compare layers from different databases.

Note: The resolution in the first job must be the same as the resolution in the second job.

If you do a database comparison or logical operation, the SCALE and RESOLUTION commands must be identical in the Description blocks of both input rules files. For the format required to compare layers from different databases, refer to the database comparison example in the [CONVERT-DATABASE](#) section in Chapter 11.

You can use the SNAP command instead of RESOLUTION to specify grid snapping.

Arguments

path-name

Specifies the layer to import. You can use the directory path name to specify the layer file.

layer-name

A user-defined layer name specified in the Operation block but not in the Input-Layer block. You can specify more than one *layer-name* on a line.

Example

```
*INPUT-LAYER  
IMPORT-LAYERS polyA mcA metalA  
IMPORT-LAYERS polyB  
IMPORT-LAYERS mcB  
.  
.  
polyC = 5  
metalC = 11  
.  
*END
```

KTEXT

```
KTEXT = <gds_layer_number> {ATTACH = < <Dracula_layer_name> >}
```

Description

Specifies the text input source from GDS2 layer for the “Keep Cell Text” feature in an LVS or LVL job. If `ATTACH` is specified, the text will be attached to that specific Dracula layer. The syntax is very similar to the command `TEXT`.

There are two ways to specify the text input source for the “Keep Cell Text” feature. One is the `KTEXT` command in the Input-Layer Block which reads texts from the CAD graphics database, the other is the `KEDTEXT` command in the Operation block which reads texts from a plain ASCII file. The two commands can exist simultaneously.

Checking Method

```
modes  
    Flat
```

Arguments

```
gds_layer_number  
    The number of the assigned layer in the layout CAD system.
```

```
Dracula_layer_name  
    User-defined Dracula layer name.
```

Example

```
METAL1 = 6 KTEXT = 8
```

or

```
METAL1 = 6 KTEXT = 8 ATT = METAL1
```

or

```
KTEXT = 8
```

or

```
KTEXT = 8 ATT = METAL1
```

Dracula Reference

Input-Layer Block Commands

Layer Name Definition

```
layer-name = layer-number {OFFGRID = off-grid} {DATATYPE data-type} {TEXT =  
    text-layer} {TEXTTYPE text-type} {ATTACH layer-name1} {ANGLE = ALL/90/45/  
    NON-45/ACUTE/  
    NON-90/NON-ACUTE}  
  
TEXT = text-layer layerPurpose {TEXTTYPE text-type} {ATTACH  
    layer-name1} {ANGLE = ALL/90/45/NON-45/ACUTE/NON-90  
    /NON-ACUTE}  
  
layer-name = layer-purpose TEXT = PIN  
layer-name = layer PIN  
layer-name = IDTEXT layer
```

Description

The *layer-name* variable assigns names to layout layers. You can name layers from GDSII formats.

Note: Dracula can read the same layer for both TEXT and CTEXT text, but you cannot attach text in the same layer to different layers. In the following example, some of the text attached to MET is lost:

POLY=3	CTEXT=29	ATTACH=POLY
MET=6	CTEXT=29	ATTACH=MET
VAPOX=7	TEXT=29	

Arguments

layer-name

User-defined layer name. The layer-name can have a maximum of seven alphanumeric characters and no special characters. The first character must be a letter. You can use the same layer name on more than one line to group layers from different layer numbers or datatypes. Also, you can use inclusive range values (for example, *metal = 1:3*).

You cannot use the name “TEMP” for any layer. This name is reserved for a temporary Dracula file.

layer-number

The number of the assigned layer in the layout CAD system. You can use inclusive range values. For GDSII, the supported range of *layer-number* is 0-255. For DFII formats, specify the *layer-name*.

off-grid

Checks the off-grid value of each input layer. You can specify different off-grid values for each input layer.

Dracula Reference

Input-Layer Block Commands

data-type

Valid only for GDSII. You can specify a range value (for example, DATATYPE=7:10). The supported range is 0-255. The default is all datatypes.

text-layer

You can designate text as a *layer-name* so that it is associated with a layer number, for example, text = 40 ATTACH metal. You can use a range value, for example, TEXT=11:14. The default is that TEXT is not used.

layer-purpose

Extracts objects from Design Framework II based on layer purpose information you specify in the rule file. A Dracula layer can include EDGE or Design Framework II layer purpose information. The default includes all layer purposes except purpose *pin*.

text-type

Field numbers associated with TEXT *text-layer*. You can use a range value. The default is all text types.

layer-name1

Layer-name for a layer specified in the TEXTSEQUENCE or CONNECT-LAYER commands. The ATTACH command is optional.

ANGLE

You can choose certain shapes of polygons as output from the Dracula job into your layer database. You can only use one ANGLE command per layer number.

ALL

Outputs all polygons. Default.

90

Outputs only rectangles with edges parallel to the x or y axis.

NON-90

Outputs only polygons containing one edge that is not 90 degrees to the x or y axis. These are non-manhattan polygons.

45

Outputs all polygons that have all edges on either 45, 135, 225, or 315 degrees or manhattan.

Dracula Reference

Input-Layer Block Commands

NON-45

Outputs only polygons containing one edge that is not a multiple of 45 degrees to the axis.

ACUTE

Outputs only polygons containing one acute angle.

NON-ACUTE

Outputs only polygons not containing any acute angle.

Note: You cannot use the ANGLE command to process a combination of multiple shapes.

Example 1

GDSII example:

```
diffus      = 2
metal       = 9
poly        = 8
implant     = 4
metall1     = 12
metall      = 22
Diff        = 1
Emitter     = 3:6
base        = 7
metal       = 10:14
TEXT 10 ATTACH metal
TEXT 16 ATTACH scdrn
TEXT = 20 ATTACH poly
```

Dracula reads the *layer-name* commands and processes the data line by line. To split the data onto multiple layers using the DATATYPE or TEXTTYPE commands, you must assign the DATATYPE and TEXTTYPE layer names first.

Example 2

```
emitter = 1 DATATYPE = 2
emitb = 1 DATATYPE = 1:3
nplus = 1
TEXT = 18 TEXTTYPE = 5
```

The *emitter* layer contains layout data from *layer 1* of *DATATYPE 2*. The *emitb* layer contains layout data from *layer 1* of *DATATYPE 1* and *DATATYPE 3*. The *emitb* layer does not contain *DATATYPE 2* because this was previously used for the *emitter* layer. The *nplus* layer contains all data on *layer 1* with *DATATYPE 0* and *DATATYPE 4-63*. This command reads text from *layer 18* of *TEXTTYPE 5*.

Dracula Reference

Input-Layer Block Commands

Note: If you define the *nplus* layer first, it contains all layout data from *layer1*, leaving the *emitter* and *emitb* layers empty.

Example 3

Dracula layers can include EDGE or Design Framework II layer-purpose information.

```
poly      = poly drawing
pwell     = pwell all
metal     = metal
srcdrn    = srcdrn pin
```

If you assign *PIN* to the layer-name variable, Dracula extracts the geometry on the pin purpose layer. If you specify `TEXT = PIN`, Dracula extracts the text from the pin purpose layer.

When you define the pin purpose to create text, Dracula reads in only the pin purpose shapes. When the pins are on a design layer purpose pin, you must define a separate layer for the text, such as

```
ME1 = metall
TEXT=metall pin attach=ME1
TEXT=metall attach=ME1
```

Example 5

IDTEXT layer example:

```
*DESCRIPTION
PRIMARY      = a1
SYSTEM       = GDS2
SCALE        = .001 MICRON
PRINTFILE    = DRC3
RESOLUTION   = .001 MICRON
INDISK       = in.db
OUTDISK      = DRC3OUT
*END
;
*INPUT-LAYER
metal        = 1
idtxt1       = IDTEXT 1
poly         = 2 TEXT 2
idtxt2       = IDTEXT 2
cont         = 3
*END
;
*OPERATION
SELECT metal BY idtxt1 LABEL[t] {RES111,RES222,RES333} pad1
OUTPUT mpad 2
SELECT poly  BY idtxt2 LABEL[t'] RES? pad2
*END
```

OUT-ANNOTATED-LAYER

OUT-ANNOTATED-LAYER = *LAYER1 LAYER2 LAYER3 ...*

Description

With this command, a user can define the names of the output layers to be annotated. Default output layers are layers in CONNECT-LAYER and device layers in the ELEMENT command, for example ELEMENT MOS[P] NGATE POLY NSD PWELL).

Note: OUT-ANNOTATED-LAYER requires the command SYSOUT=AGDS2 in the Description Block. For more information, see [“SYSOUT”](#) on page 801.

Argument

LAYER1 LAYER2 LAYER3 ...

The name of the output layer to be annotated

Example 1

In the following example, the output layers to be annotated are ME1, ME2, ME3, and ME4.

OUT-ANNOTATED-LAYERS = ME1 ME2 ME3 ME4

PAD-LAYER

PAD-LAYER = *pad_layer* [*affected_layer1*] [*affected_layer2...*]

Description

Specifies the location of the pad in the primary cell for PRE or HPRE. Extra terminals are formed automatically in PRE or HPRE jobs by the LOGICAL operation on the CUT-TERM input conduction-layer and by the pad-layer defined in the PAD-LAYER command. Pad terminals are selected from each RCONNECT-LAYER geometry that overlaps the pad-layer and are reported as the pad name/number in the .SUBCKT statement in the SPICE/DSPF file by LPESELECT or SPFSELECT commands. You must specify the PAD-LAYER command in the PRE or HPRE jobs.

You can also specify the affected layers in a PAD-LAYER command so that extra terminals or circuit pads are formed only on the affected layers you specify. You can specify multiple PAD-LAYER commands, with different pad layer names and different affected layers specified for each, in the same rule file.

Note: Dracula does not support this command if the terminal layer from CUT-TERM is not in the RCONNECT-LAYER. In addition, this command is not supported if the affected layers are not CUT-TERM layers.

Checking Method

modes

Composite or flat.

Argument

pad_layer

The name of an input layer from the layout database or a derived layer.

affected_layer

Cut-term layer. This can either be conduction layers or derived layers that are affected by *pad_layer*. If you do not specify any affected layers, the *pad_layer* affects all reconnect layers.

Dracula Reference

Input-Layer Block Commands

Example 1

In the following example, an input layer is defined as a pad-layer, and the pad-layer affects all reconnect-layers.

```
*INPUT-LAYER
METAL      = 10
MET2       = 12 TEXT = 30
VAPOX      = 60
PAD-LAYER  = VAPOX
```

Example 2

In the following example, a derived layer is defined as a pad-layer, and the pad-layer affects all reconnect-layers.

```
*INPUT-LAYER
METAL = 10
PAD-LAYER = PPAD
...
*OPERATION
SELECT METAL LABEL[R] ? PPAD
```

Example 3

In the following example, pad layer *PAD1* affects CUT-TERM layer *METAL1* and *METAL2*. Pad layer *PAD2* affects CUT-TERM layer *METAL3*. Pad layers *PAD1* and *PAD2* do not affect any other RCONNECT layers.

```
CONNECT-LAYER = PSB NWELL PSD NSD POLY METAL1 METAL2 METAL3
RCONNECT-LAYER = PSUB NWELL PWEILL NSD PSD PTRM M1TRM M2TRM M3TRM
PAD-LAYER = PAD1,METAL1,METAL2
PAD-LAYER = PAD2,METAL3
....
CUT-TERM METAL1 CONT M1RES M1TRM
CUT-TERM METAL2 VIA M2RES M2TRM
CUT-TERM METAL3 VIA M3RES M3TRM
....
```

Example 4

In the following example, the *PAD* layer affects the derived layer, *M1BIAS*.

```
RCONNECT-LAYER = ...M1TRM
PAD-LAYER = PAD M1BIAS
...
CUT-TERM M1BIAS CON M1RES M1TRM
```

Dracula does not support the PAD-LAYER command if the terminal layer from CUT-TERM is not in the RCONNECT-LAYER. For example, the following command is not valid:

Dracula Reference

Input-Layer Block Commands

```
RCONNECT-LAYER = ...M1TRM
PAD-LAYER = PAD,M1Z
...
CUT-TERM M1Z CON M1RES OM1TRM
OR OM1TRM OTHER M1TRM
...
RCONNECT M1TRM POLY BY CON
...
```

In addition, Dracula does not support the following command because the affected layers *psd* and *nsd* are not CUT-TERM layers.

```
CONNECT-LAY = PSUB NWELL PWEILL NSD PSD,POLY1 METAL
RCONNECT-LAY = PSUB NWELL PWEILL NSD PSD PTRM MTRM
....
PAD-LAYER = VAPOX,METAL1,PSD, NSD
PAD-LAYER = PAD,POLY1
....
CUT-TERM METAL1 CONT MRES MTRM
CUT-TERM POLY1 CONT PRES PTRM
.....
```

PIN-TEXT-LAYER

PIN-TEXT-LAYER = *pin_layer* [*layer*] [*value*]

Description

Creates an additional physical layer for each text layer. This physical layer creates squares that surround all text on the composite plane only. This layer can be used for logical operations.

If the pad-layer is generated from PIN-TEXT-LAYER, different pin layers can be generated from different texted layers if you specify the texted layer name in PIN-TEXT-LAYER command. You can specify multiple PIN-TEXT-LAYER commands in the same rule file if you specify multiple pad-layer commands. The PIN-TEXT-LAYER command includes text from EDTEXT-FILE.

Checking Method

modes
Composite.

Arguments

pin_layer

The corresponding pin layer generated from *layer*. The layer name can have a maximum of seven alphanumeric characters and no special characters. The first character must be a letter.

layer

A layer with text.

value

Size of the squares that surround the text. The value must be a positive integer, can be less than 1, and can have a maximum of three digits. Default value is 2, which means the square is 2 by 2. The minimum value is one resolution unit.

Example

The following example generates pin-layer *P1* for text on METAL1 and *P2* for text of METAL2. Pin-layer *P1* is defined as a pad-layer that affects METAL1. Pin-layer *P2* is defined as a pad-

Dracula Reference

Input-Layer Block Commands

layer that affects *P2*. If text on METAL1 falls inside an overlapped area of METAL1 and METAL2, only one terminal (on METAL1) is formed.

```
PIN-TEXT-LAYER=P1, METAL1
PIN-TEXT-LAYER=P2, METAL2 3
PAD-LAYER=P1, METAL1
PAD-LAYER=P2, METAL2
```

RCONNECT-LAYER

RCONNECT-LAYER = *layer1 layer2...*

Description

Specifies the mask order in which to reconnect the layout circuit after PRE splits conduction layers to generate parasitic resistors.

This command is required if you include an RCONNECT command in the Operation block.

This command lists the same layers as the CONNECT-LAYER commands, substituting the conduction layers from the RCONNECT commands that have been split into resistor and resistor terminal layers for the corresponding resistor terminal layers. The syntax of this command is the same as the CONNECT-LAYER command. You can specify layer names in a single RCONNECT-LAYER line or in multiple RCONNECT-LAYER commands.

Checking Methods

modes

Flat and composite.

checks

PRE and HPRE.

Arguments

layer1 layer2

The layer names that appear in the RCONNECT operation. Each layer name must be at least three characters long. If the layer names appear in an ELEMENT command, the first three characters must be unique.

Example

In this example, the *POLY* conduction layer is split into a resistor terminal layer, *PTRM*, and a resistor layer. The RCONNECT-LAYER sequence of parameters is the same as the CONNECT-LAYER parameters except for substituting *PTRM* for *POLY*. The RCONNECT command parameters also resemble the CONNECT parameters, with *PTRM* substituting for *POLY*.

```
CONNECT-LAYER = NSD POLY MET1 MET2
RCONNECT-LAYER = NSD PTRM MET1 MET2
```

RCX-LAYERS

RCX-LAYERS = LAYER1 LAYER2 ...

Description

This command defines the layers which are passed into Assura RCX through the `xcn` file created by the `GENXCN` utility. Please refer to the *Assura Physical Verification Developer Guide* for more information on the `xcn` file.

This command works only if `RCX-MODE=YES` or `RCX-MODE=LAYOUTONLY` (see “[RCX-MODE](#)” on page 755). For additional information see the *Dracula Users Guide*, “Running the DracToRCX Interface”.

Argument

LAYER1 LAYER2 ...

Specific layers which will be used by `GENXCN` to create the `xcn` file required by RCX. The specified layer must be defined in the `INPUT-LAYER` section of the rule file.

Note: Layer names must have at least three characters.

Example

```
*DESCRIPTION
...
RCX-MODE=YES
...
*END

*INPUT LAYER
...
*END
...
RCX-LAYERS = mt1, mt2
...
*END
```

SUBSTRATE

`SUBSTRATE = layer-name layer-number`

Description

Generates the substrate layer you can use as a regular layer in an operation. Dracula forms the geometries on the substrate layer in the following ways:

- In flat mode, the substrate layer encloses all the layout data of the processed chip in a rectangle 1.0 micron or 1.0 mil larger. For hierarchical, cell, or composite modes, the substrate layer is slightly different.
- In the cell plane of each Hcell, the substrate layer is the same as the bounding box of the Hcell. You can supersede the “bounding-box” area by providing a rectilinear boundary using the CELLBNDY command. No automatic growing takes place.
- In the composite plane, the substrate region of each cell placement is trimmed from the composite substrate layer. The outer boundary of the composite substrate grows as in flat mode (by 1.0 micron or 1.0 mil), but the interior contains holes as the result of trimming.

The SUBSTRATE and CELLBNDY commands affect the formation of the substrate layer, which in turn affects the formation of the reverse-field layer. The connectivity of the reverse-field layer might affect HLVS and HLPE jobs. The example below illustrates a possible problem.

Dracula issues the following fatal error message if the WINDOW command specifies a window outside the substrate layer:

```
*/FE*** FATAL ERROR DETECTED: NO DATA HAVE BEEN FOUND
-- PLEASE CHECK DATABASE AND WINDOW SIZES.
```

Arguments

layer-name

The name of the generated rectangle. The layer name can have a maximum of seven alphanumeric characters and can have no special characters. The first character must be a letter.

layer-number

A two-character dummy field. You can specify any number from 0 to 99. The number is not significant.

Dracula Reference

Input-Layer Block Commands

Example

```
SUBSTRATE = BULK 63  
NOT BULK PWELL NSUB
```

The *NSUB* layer of a cell in flat mode and the *NSUB* layer of the same cell (as an Hcell) in cell mode might differ because of differences in the *BULK* layer. If this causes connectivity problems, especially in cell mode, use the CELLBNDY command to alter the boundary by making it slightly larger than default. If you have a problem with open circuits, you can provide virtual wire texting to connect the wire.

TEMPORARY-LAYER

TEMPORARY-LAYER = *layer-name1 layer-name2...*

Description

Specifies the layer names you use repeatedly so that the preprocessor does not issue a warning each time it encounters a repeated name. This command does not suppress any real error messages. You can repeat this command as often as necessary, but you need to specify each repeated command only once.

This command is useful when you use the same layer name repeatedly to save disk space or improve readability of your rules.

Note:

- In Distributed Dracula (DD) mode, temporary layers are assigned different names internally. The TEMPORARY-LAYER command should not be used in DD mode. Even if it's used, it has no effects.
- In single mode, TEMPORARY-LAYER command has effects. An original mask layer should not also be defined as a TEMPORARY-LAYER.

Arguments

layer-name

Name of the layer assigned as a temporary-layer.

Example

In this example, in the first line of the Operation block, *abc* is the name assigned as the result of the AND operation. In the second line, *abc* is assigned as the result of the NOT operation. Normally, Dracula issues a WARNING to prevent you from directing the output of multiple commands to *abc*. If you specify the TEMPORARY-LAYER command, the preprocessor does not issue a warning message and allows multiple use of *abc* as an output.

An error message appears for the last command because the program needs *real* in the final output stage to output the *polyg01* error file. The TEMPORARY-LAYER command does not suppress any real error messages. If an error of this type occurs, use a name other than *real* for the last logical operation.

```
*INPUT-LAYER
.
.
TEMPORARY-LAYER = abc, nbc, real
```

Dracula Reference

Input-Layer Block Commands

```
*END
*OPERATION
AND ttt ccc abc
NOT abc ddd abc
SELECT eee cut fff nbc
OR abc nbc nbc
OR abc cbs real OUTPUT polyg01
OR abc obs real <--ERROR!
```

TEXTSEQUENCE

`TEXTSEQUENCE = layer1 layer2...`

Description

Redefines the text sequence generated by the CONNECT-LAYER command. The TEXTSEQUENCE command reorders the interconnect layers for attaching text. Text attaches first to the rightmost argument, then in order from right to left, similar to the CONNECT-LAYER command.

Arguments

layer

The name of the layer in the CONNECT-LAYER command (excluding contact layers).

Example

In this example, TEXTSEQUENCE reorders the text assigned to the interconnect layers *poly* and *srcdrn* and eliminates text attachments to the *metal* layer. Dracula tests whether the text attaches or falls in the *srcdrn* layer and assigns a text name accordingly. For text not assigned to the *srcdrn* layer, Dracula tests for the *poly* layer and assigns the text name accordingly. It disregards text not assigned to either layer.

```
CONNECT-LAYER = srcdrn poly metal
TEXTSEQUENCE = poly srcdrn
```

.

Dracula Reference

Input-Layer Block Commands

Operation Block Commands

Chapter 13 discusses the Operation Block Commands for Dracula in alphabetical order.

Introduction

This chapter describes the commands in the Operation block of your rules file. You can create as many Operation blocks as necessary to define your rules and processes. The preprocessor automatically concatenates the individual definitions in the order you specify.

The Operation block contains the following categories of commands:

- Logical Operations
- Electrical Node Extraction
- Sizing Operations
- Spacing Checks
- Circuit Element Extraction
- Electrical Rule Checking
- Layout Versus Schematic
- Layout Parameter Extraction

Logical Operations

The logical operation commands create new pseudo layers using combinations of original layers, previously created layers, and import layers. The maximum number of logical operation commands you can specify in a single rule deck is 3,000.

Electrical Node Extraction

The electrical node extraction commands define the connectivity between layers where the connection physically derives from the contacts. You define the layer sequence with the CONNECT-LAYER command in the Input-Layer block, and define the contacts that connect the specified layers in the Operation block.

Electrical nodes are formed by the layer processing sequence and the interlayer contacts. Dracula forms and uniquely labels each electrical node formed by the CONNECT-LAYER and CONNECT commands. This process is called stamping. You can carry the stamping information to newly generated layers in two ways:

- Dracula automatically stamps layers produced with the AND and NOT commands and stamps the output layer with the node label of the first layer (*layer-a*).
- The STAMP command stamps the node label from one layer to another. Stamping occurs only for stamped-layer polygons that are overlapped by nodes in the stamping layer.

Sizing Operations

The sizing commands let you alter the geometries in your database.

Spacing Checks

The spacing operations check geometric spacing and sizing on polygons in original and created layers and flag the errors. You can further process these errors using either conjunctive rules, the *R* option, or the *R'* option.

You use conjunctive rules to process the violations produced by prior checking commands. A conjunctive set is a series of commands that generates a single output. You invoke a conjunctive rule by placing an ampersand (&) at the end of a command line. The results of that command are then available to any command in the conjunctive set. Thus you can make a series of dependent checks.

An ampersand (&) generates error flags for the corresponding input layers. Flags for *layer-a* segments are marked on *layer-a*, and flags for *layer-b* segments are marked on *layer-b*. These are conjoined layers. Subsequent checks on a conjoined layer filter these errors until, if any remain, they are finally output. You can use the renamed conjoined layer in later checks, avoiding the need to recompute these error flags.

The ampersand (&) is a reserved character, and Dracula interprets anything following it on the same line as a comment. You cannot specify a command on the same line. Except for

layers you rename, conjoined layers revert back to the original layer after the processing of the conjunctive rule is completed.

You can use the conjunctive rule to filter the error flags through the conjunctive set. The set always ends with an OUTPUT statement.

Note: Renamed conjunctive filenames must begin with an ampersand (&).

The maximum number of spacing commands you can have in a single rule deck is 3,000.

Circuit Element Extraction

The circuit element extraction commands define electrical circuit elements. You use these commands to extract MOS devices, pads, and electrical parameters such as widths and lengths. When you define electrical nodes with CONNECT commands, circuit element extraction commands automatically generate a circuit netlist. You can use this extracted circuit to perform electrical rule checks (ERC), layout parameter extraction (LPE), or layout versus schematic checks (LVS).

Dracula checks the devices defined in an ELEMENT command to verify that the devices are formed correctly. For terminals that are missing or not properly formed, Dracula reports the coordinates of the device in the *.ERC* file.

The first three characters you specify in an ELEMENT or PARASITIC command must be unique. You cannot specify the same layer name more than once in the ELEMENT command. The preprocessor checks this syntax.

Electrical Rule Checking

The electrical rule checking commands check for incorrect devices and gross continuity errors. You can apply some of these rules globally and some locally.

After completing your ERC checks, Dracula flags electrical nodes and elements as potential violations and generates graphic output. For electrical nodes, Dracula traces continuity through the contacts to all areas of layers with the same electrical potential. Dracula uses a definition area to identify elements. For example, the channel area can represent the MOS device to help locate particular violations.

Layout Versus Schematic

The layout versus schematic commands run Layout Versus Schematic (LVS), Layout Versus Layout (LVL), and Schematic Versus Schematic (SVS) comparisons.

Layout Parameter Extraction

The layout parameter extraction commands control parameter extraction and report layout parameters. These commands report layout parameters in SPICE-compatible formats with layout label cross-references. If you provide a schematic, LPE updates the source if it is in SPICE format. LPE reports in SPICE format with I/O and designator cross-references if the schematic netlist is in one of the logic simulation languages that Dracula accepts.

AND

```
AND{[D{n}]} layer-a layer-b trapfile {OUTPUT c-name l-num {d-num}}  
AND[E] layer-a layer-b trapfile
```

Description

Creates a new layer from two other layers. The new layer consists of the region shared by the two layers.

Database partition functions were added for the SELECT, SIZE, and LOGICAL commands. These enhancements resolve capacity limitation issues with Dracula. This allows you to partition input layers of these functions into blocks. When you run Distributed Dracula, each block (pair) can be assigned to one processor. Specify a value between 2-10 inclusive following 'D' to indicate the number of partitions desired. The default value of 'D' is 4 if the value is omitted.

Note: There is a performance gain in the database partition feature for SELECT in the 4.8 version. The same is not true for SIZE and LOGICAL on account of too much overhead from the pre-process (PRESEL for SIZE) and post-process (LOGMRG for LOGICAL) stages during the operations. Database partition scheme is designed specially for resolving Dracula capacity issues and is applied to flat mode only.

Arguments

D

Turns on the database partition feature

n

Number of blocks to be partitioned into

layer-a

Name of the first input layer

layer-b

Name of the second input layer

trapfile

The name of the trapezoid file created by the logical operation. Arguments *layer-a* and *layer-b* are order dependent and can affect the output trapfile if connectivity has been established. Electrical node information is automatically stamped on this output layer from *layer-a* after nodal information is set up on *layer-a* with the CONNECT command.

Dracula Reference

Operation Block Commands

The *trapfile* name can have up to seven characters and no special characters. A-Z and 0-9 are permitted, but the first character must be a letter.

OUTPUT

Sends the results of the operation to an output cell.

c-name l-num

C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

The datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

E

Creates an edge information file for each generated layer. This information is used for sidewall capacitance extraction.

Examples:

```
AND ndiff poly ngate nsd
AND[D7] pdiff poly pgate psd
```

ANDNOT

ANDNOT layer-a layer-b ANDtrapfile NOTtrapfile

ANDNOT[E] layer-a layer-b ANDtrapfile NOTtrapfile

Description

Logical command to do AND/NOT at the same time. This operation simplifies the rule file.

Arguments

layer-a

Name of the first input layer

layer-b

Name of the second input layer

ANDtrapfile

The name of the trapezoid file created by the AND logical operation. Arguments *layer-a* and *layer-b* are order dependent and can affect the output trapfile if connectivity has been established. Electrical node information is automatically stamped on this output layer from *layer-a* after nodal information is set up on *layer-a* with the CONNECT command. The *trapfile* name can have up to seven characters and no special characters. A-Z and 0-9 are permitted, but the first character must be a letter.

NOTtrapfile

The name of the trapezoid file created by the NOT logical operation. Arguments *layer-a* and *layer-b* are order dependent and can affect the output trapfile if connectivity has been established. Electrical node information is automatically stamped on this output layer from *layer-a* after nodal information is set up on *layer-a* with the CONNECT command. The *trapfile* name can have up to seven characters and no special characters. A-Z and 0-9 are permitted, but the first character must be a letter.

E

Creates an edge information file for each generated layer. This information is used for sidewall capacitance extraction.

Example

```
ANDNOT ndiff poly ngate nsd
```

ANGLED-EDGE-TOL

ANGLED-EDGE-TOL [=] {<value> | YES/ON | NO/OFF}

Description

This command enables and defines, or disables a tolerance setting for the DRC checking operation on non-manhattan edges.

For non-manhattan design, round-off issues are a well-known problem. This leads to a large number of "false" minimum DRC violations due to the snapping of vertices to a grid. For example, the user may start with a rectangle drawn at the minimum width of 1 μm . This rectangle is then rotated by 30 or 45 degrees (or any angle that is not multiple of 90°). Due to snapping of the vertices to the grid, the resulting shape may have a width that is slightly less than 1 μm . This can lead to a "false" minimum width violation. `ANGLED-EDGE-TOL` sets the tolerance for the DRC operation in order to eliminate such false violations.

`ANGLED-EDGE-TOL` may be specified multiple times in a ruledeck. Every `ANGLED-EDGE-TOL` affects all DRC commands (`EXT/ENC/INT/WIDTH`) until either the end of the ruledeck or another `ANGLED-EDGE-TOL` command changes the setting. When it is inserted into the `DESCRIPTION` block it behaves the same as if it were the very first command in the `OPERATION` block.

Arguments

=

Optional for convenience.

YES/ON

Turn tolerance on and set value to default 2 dbu.

NO/OFF

Turn tolerance off. This is the default behavior.

value

Turn tolerance on and set tolerance to the specified *value*.

Example

```
*DESCRIPTION
```

```
...
```

```
RESOLUTION = 0.001 MIC
```

Dracula Reference

Operation Block Commands

ANGLED-EDGE-TOL ON

...

*END

*OPERATION

...

; The following rule checks for metal pieces that are thinner than 0.18 um
; but for non-manhattan cases minimal metal width is 0.178 (0.18 - 0.002)
; as defined by the default 2 dbu setting of ANGLED-EDGE-TOL

WIDTH METAL LT 0.18 OUTPUT ERR 1 1

; The following rule changes the <value> of the ANGLED-EDGE-TOL

ANGLED-EDGE-TOL 0.001

; The following rule checks for metal pieces thinner than 0.18 um
; but for non-manhattan cases minimal metal width is 0.179 (0.18 -0.001)

WIDTH METAL LT 0.18 OUTPUT ERR 2 1

; The following rule disables the ANGLED-EDGE-TOL check

ANGLED-EDGE-TOL NO

; The following rule checks for metal pieces thinner than 0.18 um
; with no tolerance allowed for non-manhattan shapes.

WIDTH METAL LT 0.18 OUTPUT ERR 3 1

*END

AREA

```
AREA layer-a {RANGE/NE/EQ/GE/GT/LE/LT} n1 n2 {trapfile} [OUTPUT c-name  
  l-num {d-num}]
```

Description

Performs a minimum area check. This operation checks polygons to determine if their areas are within an area range. It also creates layers that are within specified area boundaries.

Important

The AREA command is intended for flattened layers only and might produce incorrect results if used on hierarchical layers. If using the AREA command on hierarchical layers, results should be verified in FLATTENED mode.

Arguments

layer-a

Name of the input layer.

RANGE *n1 n2*

Flags polygons with areas such that $n1 < \text{AREA} < n2$. (Exclusive).

NE

Reports an error if any value of the layer specified is not equal to the value you specify.

EQ

Reports an error if any value of the layer specified is equal to the value you specify.

GE

Reports an error if any value of the layer specified is greater than or equal to the value you specify.

GT

Reports an error if any value of the layer specified is greater than the value you specify.

LE

Reports an error if any value of the layer specified is lesser than or equal to the value you specify.

Dracula Reference

Operation Block Commands

LT

Reports an error if any value of the layer specified is lesser than the value you specify.

trapfile

Name of the trapezoid file.

OUTPUT

Sends the results of the operation to an output cell.

c-name l-num

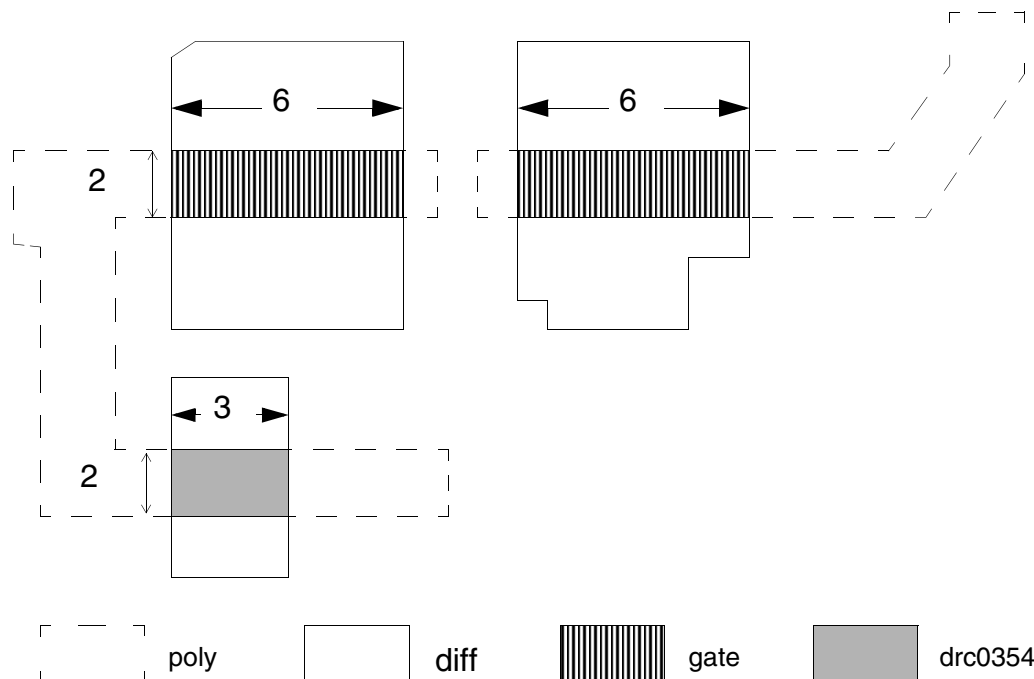
C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

Specifies the datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

Example 1

```
AND poly diff gate
AREA gate RANGE 0 7 OUTPUT drc03 54
```



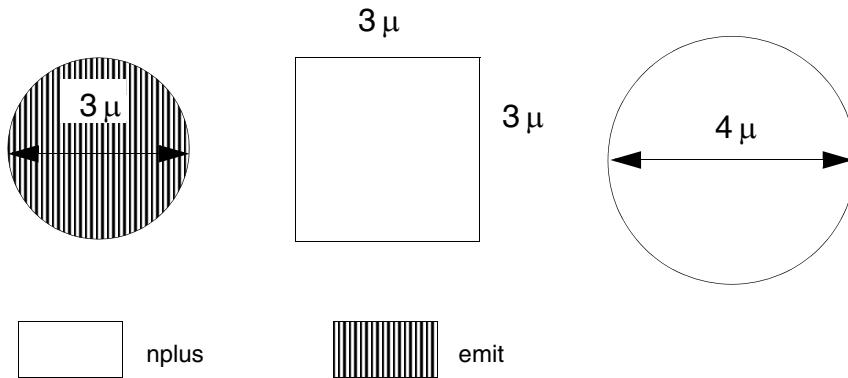
Dracula Reference

Operation Block Commands

Example 2

```
AREA nplus RANGE 7.04 7.08 emit
```

Finds the 3-micron diameter circle emitters in the *nplus* layer. You can use the *emit* layer for further operations.



Note: The area of a circle with a diameter of 3 microns is πr^2 or $3.14159 \times 1.5 \times 1.5$, which equals 7.257. This value is greater than the maximum area specified in the AREA command. This apparent discrepancy occurs because the circle was digitized as straight segments that approximate a circle; therefore, the actual area calculated by Dracula falls within the range of 7.04 to 7.08.

ATTACH

ATTACH device-subtype parameter-file {parset-name} {&}

Description

Joins source and drain parameters to the MOS or LDD devices. When you use this command with the **NODE** option of the **LEXTRACT** command, you can extract the source/drain parameters. Dracula reports the source/drain area in the SPICE output file as having a certain value.

In Dracula 4.81 and previous versions, using the ampersand (&) sign you could group **ATTACH** commands if the commands worked on the *same* device type. In those versions, the **ATTACH** command handled the extraction of common source or drain parameters of different subtype of devices belonging to the same device type using a syntax for example, as given below:

```
ATTACH MOS[A,B]
```

In this current Dracula (version 4.9), using the ampersand (&) sign, you can group **ATTACH** commands if the commands work on either the *same* device type, or *shared* common source or drain. So in this Dracula 4.9 version, along with the previous syntax for the same device type, the **ATTACH** command handles the extraction of common source or drain parameters of different device types grouped, using a syntax for example, as given below

```
ATTACH MOS[A] PDIFFN &  
ATTACH MOS[B] PDIFFN &  
ATTACH LDD[C] PDIFFN
```

Note: Specify an **ATTACH** command after the device definition and **LEXTRACT** commands in your rules file. The **ATTACH** command works only for MOS devices.

Arguments

device-subtype

The device with subtype specified in the rules file. You can use a MOS or LDD device type. Multiple subtypes separated by comma are also allowed in the same **ATTACH** command for source/drain parameters extraction of different subtype of devices which share common source/drain.

parameter-file

The file containing the source/drain parameters. You extract the source/drain parameters by using the **LEXTRACT** command with the **NODE** option.

Dracula Reference

Operation Block Commands

parset-name

The name of a *parset* (refer to the PARSET command) that specifies how the source and drain results are stored in the parameter file. The default is *MOS*.

&

Using the ampersand (&), you can group ATTACH commands if the commands work on the same device type.

Example 1

The following shows how you might use the ATTACH command in your run file:

```
*DESCRIPTION
...
PARSET TEST AREA PERI          ; PARSET FOR S/D
UNIT AREA,P PERIMETER,U
...
*END
...
*OPERATION
...
ELEMENT      MOS[N] NCHNL      METAL    DIFFN    OVPWELL ;N-CHANNEL
ELEMENT      MOS[P] PCHNL      METAL    DIFFP     NSUB     ;P-CHANNEL
...
LEXTRACT TEST DIFFN BY NODE PDIFFN
LEXTRACT TEST DIFFP BY NODE PDIFFP
ATTACH MOS[N] PDIFFN &
ATTACH MOS[P] PDIFFP
...
LPECHK
LPESELECT[S] MOS &
...
```

Example 2

The following shows a SPICE file with ATTACH command results:

```
Mxxx Nd Ng Ns Ns modelName L=4.5 μ W=20.0 μ AD=149.5P PD=59.0u
+ AS=139.9P PS=59.0u
```

Where *AD*, *PD*, *AS*, and *PS* are new keywords representing the areas and perimeters of the drain/source of the MOS transistors. Note that *AS* and *PS* are listed on a second line following the SPICE continuation sign (+) and that their values include units because the UNIT command was specified in the Description block.

Example 3

An example of using the ATTACH command to attach the AS/AD/PS/PD parameters to LDD devices:

Dracula Reference

Operation Block Commands

```
*description block
; ...
PARSET   LDDS   AREA OVPR W L AS PS AD PD
PARSET   TEST   AREA PERI
*end

...
*operation block
...
;                               device gate drain  source sub
;                               ----- ----
ELEMENT   LDD[XV]  xvgate poly hvdiff stdiff psub
...
LEXTRACT TEST  hvdiff BY NODE DRAIN
      LEXTRACT TEST  stdiff BY NODE SOURC
      ATTACH LDD[XV] DRAIN &
      ATTACH LDD[XV] SOURC
...
*end
```

Example 4

The following example shows devices of MOS type N and NH share common diffusion:

```
PARSET SD AREA PERI
.....
ELEMENT MOS[N]  NGATE  POLY NSD PWELL
ELEMENT MOS[NH] NHGATE POLY NSD PWELL
.....
LEXTRACT  TEST   NSD   BY   NODE   PDIFFN
ATTACH    MOS[N,NH] DIFFN
.....
```

Note: In the above example, the ATTACH command is the same as the following two commands in Dracula 4.9 and Dracula 4.81:

```
ATTACH MOS[N] DIFFN &
ATTACH MOS[NH] DIFFN
```

ATTRIBUTE CAP

Case 1:

```
PARASITIC CAP {type} layer-a layer-b layer-c
  ATTRIBUTE CAP {type} value-a value-b
```

Case 2:

```
PARASITIC CAP {type} layer-a layer-a layer-a
  ATTRIBUTE CAP {type} value-a1 value-b1
  ATTRIBUTE CAP {type} value-a2 value-b2
```

Case 3:

```
FRINGE CAP {type} layer-a layer-b
  ATTRIBUTE CAP {type} value-a1 value-b1
  ATTRIBUTE CAP {type} value-a2 value-b2
```

Case 4:

```
PARASITIC CAP {subType} device_layer terminal_layer1 terminal_layer2
  ATTRIBUTE CAP {subType} areaCoeff perimCoeff depthRange
    sidewallCoeff
```

Description

Associates the geometry data of the PARASITIC CAP or FRINGE CAP functions with the corresponding process-dependent electrical properties. The ATTRIBUTE function does not accept units. The scale of the parasitic attribute must match the scale of the layout database. This function specifies four forms of parasitic capacitance.

- Case 1 is for area and perimeter capacitance.
- Case 2 is for same-layer fringe-field capacitance.
- Case 3 is for same-layer or different-layer fringe-field capacitance.
- Case 4 is for overlap capacitance with consideration of the fringe effect of the first terminal layer on the overlap capacitance.

Dracula supports piece-wise coefficients for fringe capacitors.

Arguments

Case 1:

type

The type code specified in the FRINGE CAP [*type*] device.

value-a

The capacitance per unit area between *layer-a* and *layer-b*. The value specified for *value-a* can be floating point (maximum of 7 digits) or a floating-point number followed by an integer exponent (maximum of 2 digits). When floating-point numbers are specified, the leading zeros are not counted in the 7-digit restriction. If the significant digits exceed the 7-digit limitation, the preprocessor rounds the value at the 7th digit. In scientific notation, the limit is 16 characters.

value-b

The capacitance per unit perimeter between *layer-a* and *layer-b*. The value can be a floating point number (maximum of 7 digits) or a floating-point number followed by an integer exponent (maximum of 2 digits). When floating-point numbers are specified, the leading zeros are not counted in the 7-digit restriction. If the significant digits exceed the 7-digit limitation, the preprocessor rounds the value at the 7th digit. In scientific notation, the limit is 16 characters.

Case 2:

type

The type code specified in the FRINGE CAP[*type*] device.

value-a1, value-a2

The maximum distance between two geometries of the same layer opposing each other where the fringe capacitance is analyzed.

value-b1, value-b2

The multiplier of the length of the edges under the maximum value *a2*.

Note: The conductor layers you specify in all three fields must be the same layer.

Case 3:

type

The type code specified in the FRINGE CAP[*type*] device.

value-a1, value-a2

The maximum distance between two geometries of different layers opposing each other where the fringe capacitance is analyzed.

Dracula Reference

Operation Block Commands

value-b1, *value-b2*

The multipliers of the length of the fringe edges under the value *a1*, *a2* respectively.

Note: When dealing with fringe capacitance extraction the maximum distance has to be practical. A large value will result in poor performance.

Case 4:

Use this case to extract the primitive parameters TPR and CLL. You must specify *depthRange* and *sidewallCoeff* at the same time. Dracula evaluates the first terminal layer to calculate the possible fringe effect on the overlap capacitance. Therefore, the sequence of terminal layers is important.

Example 1

```
STAMP metpoly BY metal
PARASITIC CAP[X] metpoly metal poly
ATTRIBUTE CAP[X] 0.0005
```

The PARASITIC capacitor of type *X* forms between *metal* and *poly* conductor layers. The device layer is *metpoly* (*metpoly* = *metal* and *poly*).

The two connecting layers of capacitor terminals are *metal* and *poly*. Capacitance of type *X* is calculated using the following formula:

$$(\text{area}) \times (0.0005)$$

Example 2

```
PARASITIC CAP[F] metal metal metal
ATTRIBUTE CAP[F] 6.0 0.0005
```

The PARASITIC capacitor of type *F* is for the fringe capacitance between metal lines. The device layer is *metal*, and both conductor layers are *metal*. The parasitic capacitance between metal lines is calculated for spacing distances up to 6 units, and the capacitance calculation is calculated by the following formula:

$$(\text{projected metal length}) \times (0.0005) \times (1/\text{distance between metal})$$

Example 3

```
PARASITIC CAP[mp] metpol metal poly
ATTRIBUTE CAP[mp] 1E-18 1E-12
```

or

```
PARASITIC CAP[mp] metpol metal poly
ATTRIBUTE CAP[mp] .0005 .00012
```

Dracula Reference

Operation Block Commands

or

```
PARASITIC CAP[mp] metpol metal poly
ATTRIBUTE CAP[mp] .0001234567E-9 .00007654321E-10
```

Note: The leading zeros do not count in the seven significant digit restriction.

You can use two or more ATTRIBUTE functions to specify the piece-wise coefficients when handling the fringe capacitance. For example

```
PARASITIC CAP[EX] MET12 MET1 MET2
ATTRIBUTE CAP[EX] 0.5 1.2 0.2 0.032 ; PIECE-WISE COEFFICIENT
ATTRIBUTE CAP[EX] 0.5 1.2 0.6 0.019 ; PIECE-WISE COEFFICIENT
```

Note: The first two coefficients remain the same. The two added coefficients identify the capacitance for PDRACULA and specify the piece-wise coefficients for fringe capacitance.

Example 4

In this case, two new reserved parameters, TPR and CLL, are added to the parameter set:

- TPR represents the perimeter where the device layer touches the projected edge of another geometry on the first terminal layer and has fringe effect with the first terminal layer.
- CLL is the sum of the fringe effect related to TPR. CLL is calculated as the current summation of all *sidewallCoeff**TPR.

The default parameter set is as follows:

```
CAPO TPR CLL AREA PERI C
```

CAPO is the default name of the parameter set.

A different fringe coefficient is assigned to the parameter according to the depth between two geometries of the device layer and the first terminal layer, respectively.

The default equation for the capacitance calculation is

$$C = (\text{areaCoeff} * \text{AREA}) + (\text{perimCoeff} * (\text{PERI} - \text{TPR})) + \text{CLL}$$

You can also use the flexible LPE feature to define your own parameter set and equations for the parameter extraction. Your parameter set must include the TPR and CLL parameters. For example:

```
*DESCRIPTION
...
PARSET NEW TPR CLL AREA PERI C ; parameter set for overlap
                                ; cap with fringe
*END
```

Dracula Reference

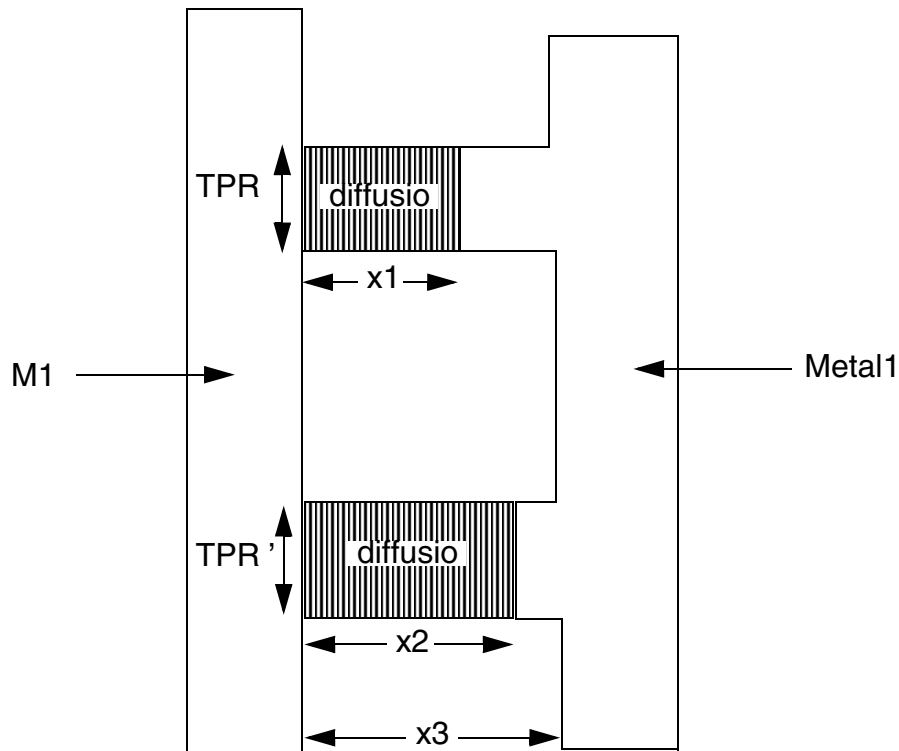
Operation Block Commands

```

*OPERATION
PARASITIC    CAP[F]      M12 MET1 MET2
ATTRIBUTE    CAP[F]      0.5  1.2  0.2 0.019 ; PIECE-WISE COEFFICIENT
ATTRIBUTE    CAP[F]      0.5  1.2  0.6 0.032 ; PIECE-WISE COEFFICIENT
LEXTRACT NEW M12 BY CAP[F] PFILE &          ; FLEXIBLE EQUATION
EQUATION C= 0.5*AREA +1.2*(PERI-TPR) + CLL
...
*END

```

For the following data, the value of TPR is $TPR' + TPR''$, but it is divided into sections when piece-wise fringe capacitance is considered. The value of the *sidewallCoeff* parameter depends on the separation between MET1 and M12.



```

x1, x2, x3 = sep. betn MET1 and M12
If 0<x1<=.2, and .2<x2<=.6 and .6<
TPR = (TPR'+TPR'')
CLL = (0.032*TPR' + 0.019*TPR'')
C = 0.5*AREA +1.2*(PERI-TPR) + CLL

```


Example 5

In Example 4, when *MET1* is cut-terminated to extract parasitic resistance, you must specify the *R* option in the PARASITIC CAP command to ensure the correct extraction of TPR and CLL as shown in the following example:

```
CUT-TERM MET1 CONT M1RES M1TRM
STAMP M1TRM BY MET1

AND M1TRM MET2 M12T
AND M1RES MET2 M12R

PARASITIC[R] CAP[F1] M12T M1TRM MET2
ATTRIBUTE CAP[F1] 0.5 1.2 0.2 0.032
ATTRIBUTE CAP[F1] 0.5 1.2 0.6 0.019

PARASITIC[R] CAP[F2] M12R M1RES MET2
ATTRIBUTE CAP[F2] 0.5 1.2 0.2 0.032
ATTRIBUTE CAP[F2] 0.5 1.2 0.6 0.019
```

Example 6

You can specify 0 depth to give a special coefficient for the co-linear fringe extraction. Currently the equation for the co-linear fringe is hard coded as $C = K * WIDT$.

```
FRINGE CAP[F]          M1  M2
ATTRIBUTE CAP[F]       0   2.5
ATTRIBUTE CAP[F]       1   4.5
ATTRIBUTE CAP[F]       2   1.5
ATTRIBUTE CAP[F]       3   0.5
```

The co-linear fringe will be calculated as $C = 2.5 * WIDT$.

Example 7

Both the zero and nonzero spacing fringe capacitance are calculated with the following equation.

```
*DESCRIPTION
...
ZERO-SPAC-F-EQU =YES

*END

*OPERATION
FRINGE CAP[A] ME1 ME2
ATTRIBUTE CAP[A] 0 1
ATTRIBUTE CAP[A] .5 1

LEXTRACT CAPF ME1 ME2 BY CAP[A] LCAPA &
```

Dracula Reference

Operation Block Commands

```
EQUATION C = WIDT / SQRT((((3.205 + DEPT)**2.861/6.259)**2) + 57.5)
*END
```

Example 8

Without this command, Dracula uses $C=K*WIDT$ as the equation for zero spacing fringe capacitance and the following equation

$$C = WIDT / \text{SQRT}((((3.205 + DEPT)**2.861/6.259)**2) + 57.5)$$

for nonzero spacing fringe capacitance.

```
*OPERATION
FRINGE CAP[A] ME1 ME2
ATTRIBUTE CAP[A] 0 1
ATTRIBUTE CAP[A] .5 1

LEXTRACT CAPF ME1 ME2 BY CAP[A] LCAPA &
EQUATION C = WIDT / SQRT((((3.205 + DEPT)**2.861/6.259)**2) + 57.5)
*END
```

ATTRIBUTE RES

```
ATTRIBUTE RES[type] sheet_res_value  
    {smashResValue maxResValue}
```

Description

Provides the process-dependent sheet resistance used to compute the resistance of the parasitic resistor devices defined by the associated PARASITIC RES command. The ATTRIBUTE RES command must follow a PARASITIC RES command having the same *type* parameter.

Checking Methods

modes

Flat and composite.

checks

PRE and HPRE.

Arguments

type

A two-character code that denotes the *type* of parasitic resistor device. The first character must be A-Z. The second character can be A-Z or 0-9, excluding 8. The second character is optional. Dracula uses this code to match this command with the associated PARASITIC RES command.

sheet_res_value

The sheet resistance value in units per square, where the units correspond to the resistance units specified in the UNIT command.

smashResValue

The resistance value used as a threshold. Series resistors with values below *smashResValue* are smashed.

maxResValue

The maximum value of a resistor that results from smashing serial parasitic resistors. This prevents creating a big resistor from subsequent smashing.

Dracula Reference

Operation Block Commands

Example

In this example, LPE extractions of parasitic resistance values made from *PRES* are computed using the sheet resistance value .01 Kohms/square. Dracula smashes resistors with values below 10 and does not create any resistor with a value greater than 200.

```
*DESCRIPTION
UNIT = CAPACITANCE,PF AREA,U PERIMETER,M RESISTANCE,K
*OPERATION
PARASITIC RES[P] PRES PTRM
ATTRIBUTE RES[P] .01 10 200
```

BBOX

```
BBOX inLayer {CENTER {size}/SELSHAPE/CORNER {size}/RELATIVE ratio/WHOLELAY}
      {PRESERVE} {WARN} {NOT} {WIDTH relation wid1 {wid2}} {LENGTH relation len1
      {len2}} {RATIO relation rat1 {rat2}} trapfile {OUTPUT c-name l-num {d-
      num}}
```

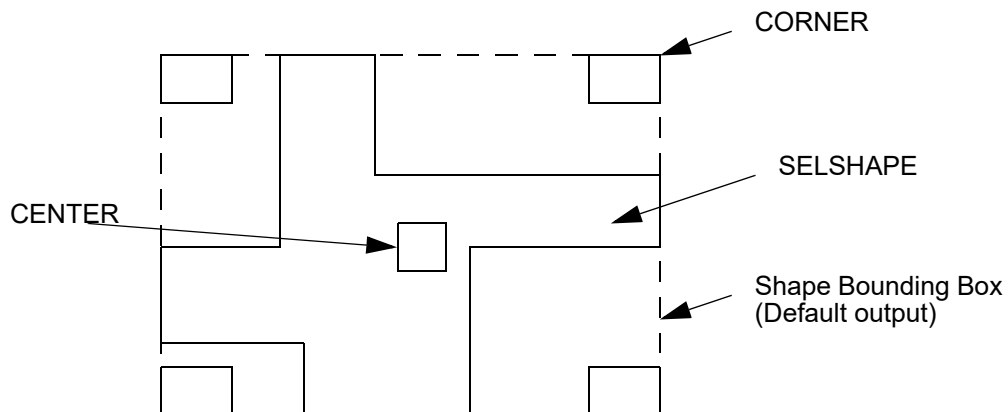
Description

Creates enclosing bounding boxes for shapes on the specified input layer, and outputs one of the following five possible outputs for shapes whose bounding box dimensions fall within the specified length and width ranges:

- Bounding boxes of the original shapes (the default output)
- The input shapes
- Center marker of the bounding box
- Corner markers of the bounding box
- One rectangle which represents the whole layer BBOX

The following figure shows the four possible outputs of the BBOX command:

Figure 13-1 The four possible outputs from the BBOX command



Arguments

<code>inLayer</code>	Input layer name.
<code>CENTER</code>	Optional argument that outputs a marker at the center of the shape's bounding box instead of a bounding box. Figure 13-2 on page 896 illustrates the exact location of the center marker.
<code>SELSHAPE</code>	Optional argument that outputs the original input layer shape to the specified trapezoid file instead of bounding boxes. This is useful to perform the shape filtering based on bounding box size.
<code>CORNER</code>	Optional argument that outputs markers at the corners of the shape's bounding box instead of a bounding box. Note: The <code>CENTER</code> , <code>SELSHAPE</code> , and <code>CORNER</code> arguments can not be specified together in a <code>BBOX</code> command.
<code>size</code>	Optional marker size value that applies to the <code>CENTER</code> and <code>CORNER</code> arguments. Note: If you do not specify a marker size value for the <code>CENTER</code> argument, the default value of 6 DBU (i.e., $6 \times \text{RESOLUTION}$) is used. A marker size value of 5 DBU is used if there is a 0.5 DBU difference between the center marker and the center of the bounding box. If you do not specify a marker size value for the <code>CORNER</code> argument, the default value of 5 DBU (i.e., $5 \times \text{RESOLUTION}$) is used.
<code>RELATIVE</code>	Optional argument which works similar to <code>CENTER</code> , but output marker will have size relative to bounding box size.
<code>ratio</code>	Shows the relative size of markers in comparison to bounding box size.
<code>WHOLELAY</code>	Optional argument that allows to generate one rectangle per layer which represents the whole layer bounding box.
<code>PRESERVE</code>	Optional argument that allows you to preserve the size of center markers regardless of whether the center of a shape is on-grid or not. In the case where the center point is off-grid, the center of the marker will be shifted to the nearest on-grid point. Figure 13-3 on page 897 illustrates how this option works.

Dracula Reference

Operation Block Commands

WARN	Optional argument to output warning messages in case of the marker size value was modified because the shape's center marker is off-grid. If the <code>PRESERVE</code> argument is also specified, a warning is issued if the bounding box's center was shifted. Note: The maximum number of warning messages is ten per <code>BBOX</code> command.
NOT	Optional argument that outputs only bounding boxes that do not satisfy the specified <code>WIDTH</code>, <code>LENGTH</code>, and <code>RATIO</code> constraints.
WIDTH/LENGTH	Optional arguments to specify the width and length constraints to the <code>BBOX</code> command. Selects shapes whose bounding boxes are within specified width and length values. <code>LENGTH</code> value(s) and <code>WIDTH</code> value(s) are compared to the length (major axis) and width (minor axis) of the bounding box.
RATIO	Optional argument to specify BBoxes selection by their <code>L/W</code> ratio. Selects shapes whose <code>L/W</code> ratio is within specified condition. It can be used together with <code>WIDTH/LENGTH</code> keywords.
<i>relation</i>	Relation operators for applying the <code>WIDTH</code> and the <code>LENGTH</code> constraints. The <code>LT</code> (less than), <code>LE</code> (less than or equal), <code>GT</code> (greater than), <code>GE</code> (greater than or equal), and <code>RA</code> (in given range) operators are supported.
<i>wid1, wid2, len1, len2, rat1, rat2</i>	Specifies the <code>WIDTH</code>, <code>LENGTH</code>, and <code>RATIO</code> constraint values. <code>LT</code>, <code>LE</code>, <code>GT</code>, and <code>GE</code> relation operators require one value each. <code>RA</code> relation operator requires two values.
<i>trapfile</i>	Trapezoid file that contains the output of <code>BBOX</code> command.
OUTPUT	Sends the results of the command to the output cell. <code>c-name</code>, <code>l-num</code>, and <code>d-num</code> are standard DRACULA cell name, layer number and optional datatype number.

Figure 13-2 Location of Center Markers

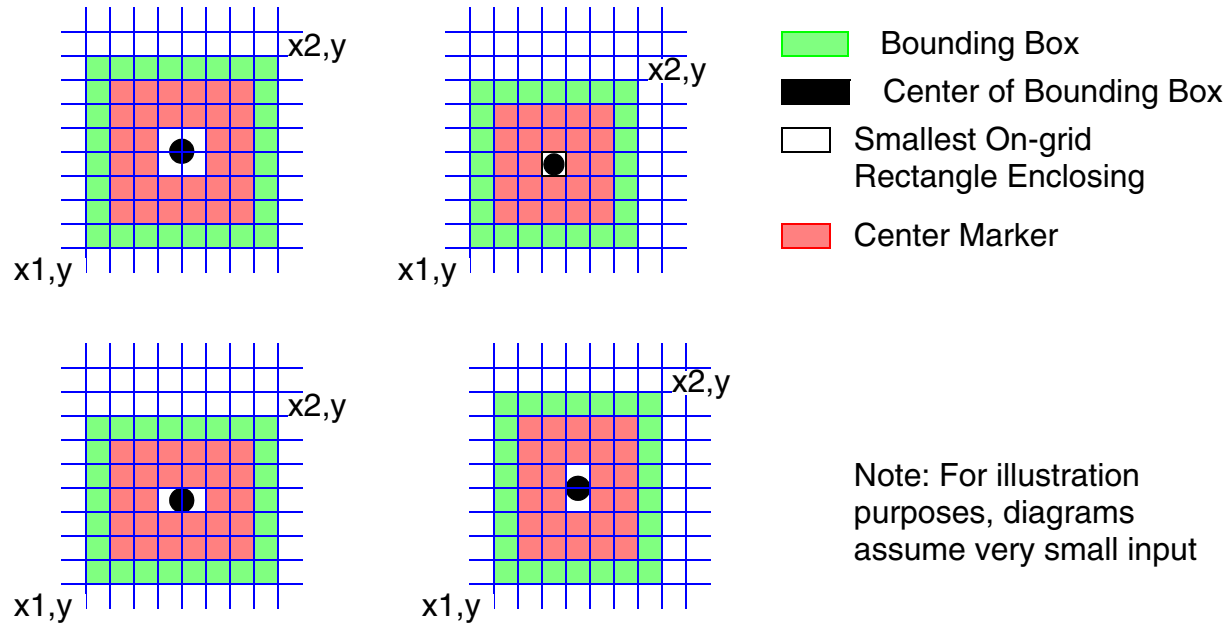
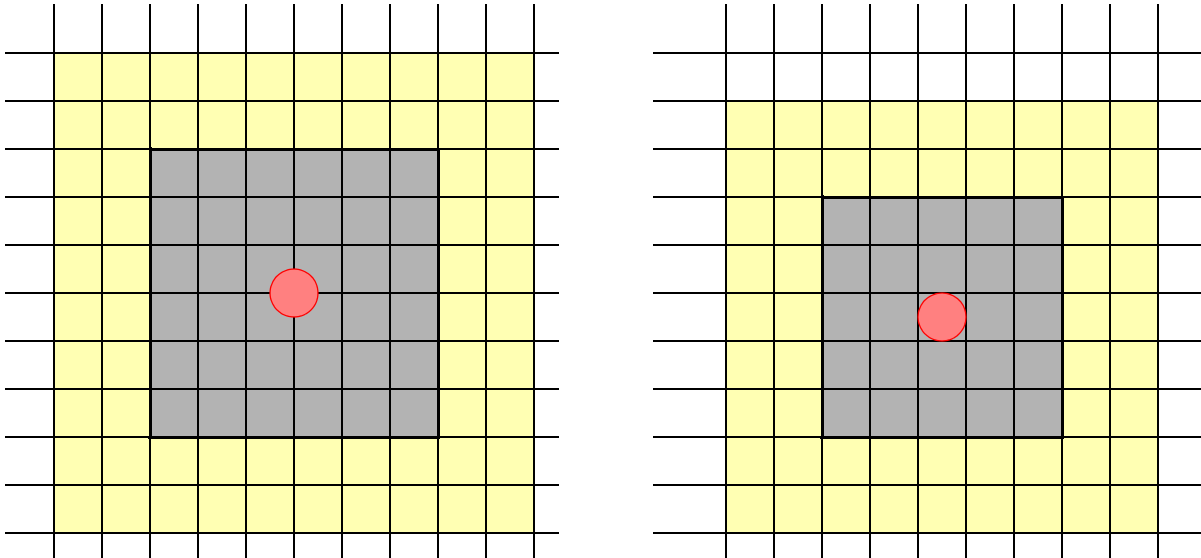
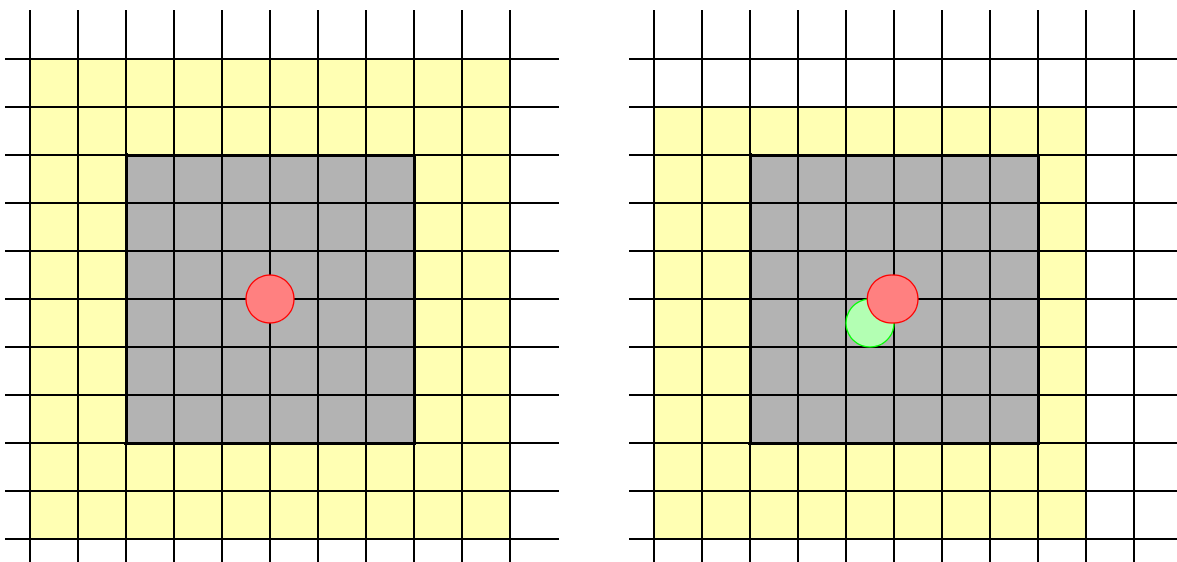


Figure 13-3 How PRESERVE keyword works



A) Default behavior change center marker size.



B) PRESERVE option moves center of marker



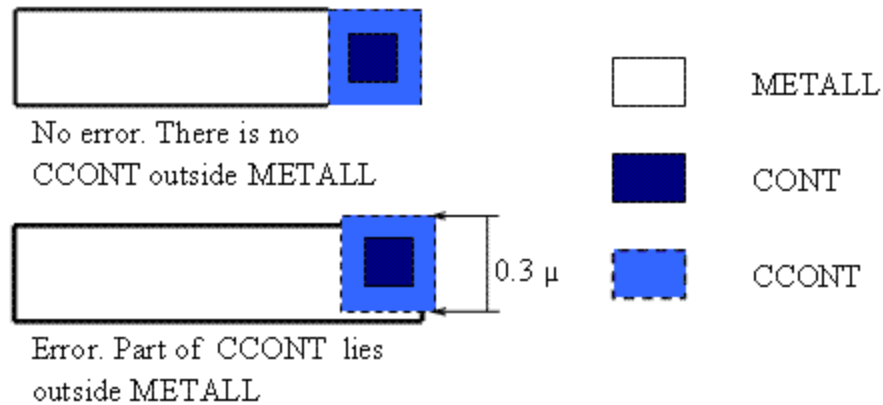
Dracula Reference

Operation Block Commands

Example 1

The following example checks that the contact center should be a maximum distance of 0.15um to the metal edges.

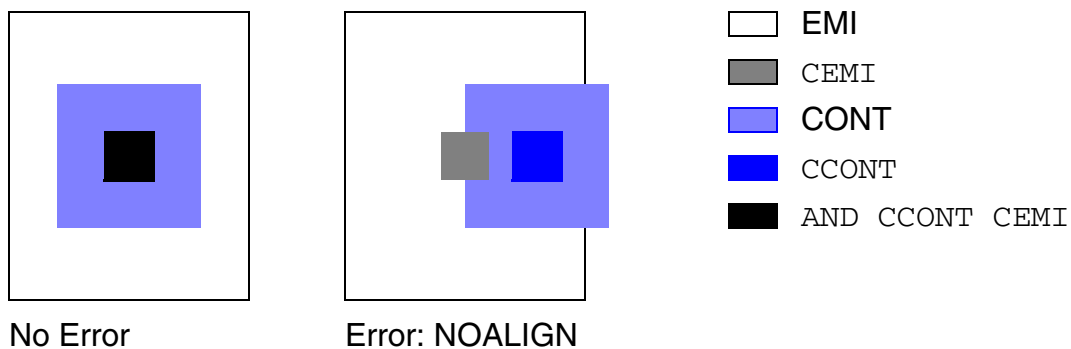
```
BBOX CONT CENTER 0.3 CCONT  
NOT CCONT METALL BADCONT OUT ERR 1
```



Example 2

The following example checks that the contact must lie in the center of the emitter.

```
BBOX CONT CENTER CCONT  
BBOX EMI CENTER CEMI  
XOR CCONT CEMI NOALIGN OUT ERR 2
```



Example 3

The following examples uses METALL1 bounding box to define chip working area and defines all the rest area of chip as test area.

```
BBOX METALL1 WHOLELAY METBBOX  
SIZE METBBOX by 1 CHIPBBOX  
NOT BULK CHIPBBOX TESTAREA
```

Example 4

The following example checks that via layer of M1 to M2 connection covers at least most internal 50% of area M1 and M2 overlaps.

```
AND M1 M2 M1M2OVR  
BBOX M1M2OVR RELATIVE 0.5 TMP  
NOT VIA1_2 TMP BADVIA
```

Example 5

The following example selects all contacts with their length more than width*3.

```
BBOX CONT RATIO GT 3 LONGCNT
```

***BREAK**

**BREAK name*

**BREAK label shell_command { \ | & }*

Description

Specifies the break points for a RESTART operation, or executes a portion of a job. Use this command in the Operation block only. This command is preceded by an asterisk (*).

You cannot insert a *BREAK command between any of the following command groups: DEVTAG, EQUATION, LPECHK, LPESELECT, or LVSCCHK. For example, you cannot insert a *BREAK command between two or more EQUATION commands. PDRACULA issues a warning message if you insert a *BREAK command between any of these commands.

Arguments

name

The name you specify. The name can be of 1-16 alphanumeric characters (A-Z and 0-9). The first character must be a letter.

You cannot use the following reserved names with *BREAK: BEGIN, EXPAND, SORT, MERGE, SYSOUT, DELETE (if KEEPDATA = NO or SMART), END, GENENV, REGION, LVSCCHM, SCHMIN, and LVSNET. These reserved names are internal break points found in the PDRACULA Job Control Language (JCL). You can use these points to restart a job.

When running an LVS job, you can use the following reserved break points: LVSNET, SCHMIN, and LVSCCHM.

label

Either a *start* or *end* point for the *BREAK command. Use this only when specifying shell commands.

shell_command

The names of the shell commands to be executed at this break. You can specify multiple lines by using the ampersand (&) at the end of each line. You can use the backslash (\) to divide a single line into two lines if the length of the line is longer than the limitation.

Dracula Reference

Operation Block Commands

Example 1

This example reruns a short check job. Remember, to restart a job, specify the *KEEPDATA* = *YES* command in your run file.

```
*OPERATION
.
*BREAK SHORTCHK
MULTILAB OUTPUT SHORTS10
*END
```

Example 2

This example specifies start and end points for the *BREAK command.

```
*OPERATION
.
*BREAK START1 if( $mode == 'FLATTEN' || $mode == \
'COMPOSITE' || $mode == 'HIER') then &
echo "RUN MODE = $mode"
NOT AAA BBB CCC
.
*BREAK END1 endif
```

Example 3

This example specifies an 'if then' statement embedded in the shell commands.

```
*BREAK START if( $run == 'now') then &
echo "run_mode = $run" &
endif
```

CALCULATE

CALCULATE RATIOFILE *parExpression*

Description

Calculates the parameter ratio of each node among multiple layers. This command is an extension of the COMPUTE command, which only applies to limited layers with limited operation. To check the “antenna effect,” you must specify this command in addition to the LEXTRACT and CHKPAR commands.

Arguments

RATIOFILE

Output file used as the input file for the CHKPAR command. Contains the ratio of parameter values calculated from the equation specified in the command for each node. Use CHKPAR PAR to check the result.

parExpression

The parameter expression can be one of the following:

- *(parExpression)*
- *+|- parExpression*
- *parExpression arithOperator parExpression*

An arithmetic operator can be +, -, *, or /.

- *function(parFile)*

A function can be any of the following:

EXP SQRT LOG LOG10 SIN
SINH COSH TANH MAX MIN
ASIN ACOS ATAN COS TAN

- *parFile*

A parameter file can be any of the following:

Dracula Reference

Operation Block Commands

- **action.parFile**

Where the action can be SUM, MIN, or MAX. The action is applied to each parameter file before the ratio calculation on each node among different layers is executed. SUM is the default action.

- **parFile.PARAMETERNAME**

Where PARAMETERNAME is defined in the correspondent parameter set used for the LEXTRACT command. AREA is the default parameter name.

- **PARAMETERFILE**

Where PARAMETERFILE contains the parameters extracted on each node of a certain layer or layers.

- **A constant (either integer or real)**

Example 1

```
CALCULATE RATIO1=PNGATE / PGATE
```

is equivalent to

```
CALCULATE RATIO1=SUM.PNGATE.AREA /SUM.PGATE.AREA
```

PNGATE and PGATE are the parameter files from the previous LEXTRACT commands. For each node in the PNGATE file, the area is calculated. The same calculation is done for the PGATE file. Then, the ratio is calculated per node. Nodes that do not exist in all parameter files are set to zero.

```
CALCULATE RES2 = (MIN.PNGATE + c1*MAX.PM1) / PGATE
```

is equivalent to

```
CALCULATE RES2 = (MIN.PNGATE.AREA + c1*MAX.PM1.AREA) / PGATE.AREA
      CALCULATE res3= (MAX.PNGATE.PERI+c1*SUM.PM1.AREA+c2*MAX.PM2.PERI) /
      MAX.PGATE.AREA
```

Example 2

```
PARSET ANT AREA PERI
...
LEXTRACT ANT GATE BY NODE PGATE
LEXTRACT ANT NGATE BY NODE PNGATE
LEXTRACT ANT M1 BY NODE PM1
LEXTRACT ANT M2 By NODE PM2
CALCULATE RATIO1 =
PNGATE / PGATE
```

is equal to

Dracula Reference

Operation Block Commands

```
CALCULATE RATIO1 =  
SUM.PNGATE.AREA /SUM.PGATE.AREA
```

```
CALCULATE res2 =  
MIN(SUM.PNGATE,0.84*SUM.PM1) / PGATE
```

is equal to

```
CALCULATE res2 =  
MIN(SUM.PNGATE.AREA,0.84*SUM.PM1.AREA) /SUM.PGATE.AREA
```

```
CALCULATE res3 =  
(MAX.PNGATE.PERI+0.2*SUM.PM1.AREA+0.7*MAX.PM2.PERI) /MAX .PGATE.AREA
```


CAT

```
CAT lay1 lay2 lay3 ... layN PseudoTrapFile
```

```
CAT[ M ] lay1 lay2 lay3 ... layN TrapFile
```

Description

Lets you quickly merge several layers while conveying nodal information, if present, in all of the input layers.

Note: If only some of the input layers contain nodal information, the produced PseudoTrapFile/TrapFile might be incorrect.

Arguments

```
lay1 lay2 lay3 ... layN
```

Name the input layers.

Note: *N* must be less than or equal to 12.

```
PseudoTrapFile
```

Output layer created by the CAT command.

```
TrapFile
```

Output layer created by the CAT command.

```
M
```

Option that produces a merged trapezoidal file.

The CAT command without the [M] option produces a pseudo trapezoidal file that contains sorted but unmerged trapezoids. This file can be used as an input layer for CAT or OR commands only.

Note: If the resulting layer from the CAT command is used as an input layer to another command, the [M] option is enabled automatically by PDRACULA.

Example 1

This example shows the merging of 4 layers— M1, M2, M3, and M4— into layer RES.

```
CAT[M] M1 M2 M3 M4 RES
```

Example 2

This example shows the merging of 12 layers— M1, M2, M3, M4, M5, M6, M7, M8, M9, M10, M11, and M12— into layer RES.

```
CAT[M] M1 M2 M3 M4 M5 M6 M7 M8 M9 M10 M11 M12 RES
```

Example 3

This example shows the merging of 15 layers— M1, M2, M3, M4, M5, M6, M7, M8, M9, M10, M11, M12, M13, M14, and M15— into layer RES. Use two CAT commands; only the last should contain the [M] option.

```
CAT      M1 M2 M3 M4 TMPL
CAT[M] TMPL M5 M6 M7 M8 M9 M10 M11 M12 M13 M14 M15 RES
```

Example 4

This example shows the merging of 40 layers— M1, M2, M3, M4, M5, M6, M7, M8, M9, M10, M11, M12, M13, M14, M15, M16, M17, M18, M19, M20, M21, M22, M23, M24, M25, M26, M27, M28, M29, M30, M31, M32, M33, M34, M35, M36, M37, M38, M39, and M40— into layer RES. Use four CAT commands; only the last should contain the [M] option.

```
CAT      M1 M2 M3 M4 M5 M6 M7 TMPL1
CAT      M8 M9 M10 M11 M12 M13 M14 M15 M16 M17 M18 M19 TMPL2
CAT      M20 M21 M22 M23 M24 M25 M26 M27 M28 M29 M30 M31 TMPL3
CAT[M] TMPL1 TMPL2 TMPL3 M32 M33 M34 M35 M36 M37 M38 M39 M40 RES
```

CHKPAR

```
CHKPAR [option] infile {layer-a} relation value1 {value2} [{outlayer}] {OUTPUT  
  c-name l-num {data-type}}
```

Description

Selects a group of nodes from the result of a COMPUTE command, and under certain conditions, outputs the layer. This is one of the commands you use for an antenna check. For more information about antenna checks, refer to [Chapter 7, “Extracting Electrical Parameters \(LPE\),”](#) or the LEXTRACT command description in this chapter.

CHKPAR outputs to the log file the node numbers and ratio values for the violated nodes so you can determine the severity of the violation. You set the number of nodes to output by specifying the LISTERROR command in the Description block. If you specify LISTERROR = YES, the first 100 nodes that satisfy the condition are output. This is the default setting for CHKPAR. If you specify LISTERROR = NO, the nodes are not output. You can also specify the number of nodes to output with the LISTERROR command.

Arguments

option

PAR

Checks the computed AREA ratio or the result from the CALCULATE command.

A

Checks the computed AREA value or ratio.

P

Checks the computed perimeter value or ratio.

infile

Name of the output file from a COMPUTE command.

layer-a

Name of the output layer. It must carry nodal information. If this is not specified, all connect layers will be examined.

Dracula Reference

Operation Block Commands

relation

One of the following operations:

GT	Greater than
GE	Greater than or equal to
LT	Less than
LE	Less than or equal to
RA	Range

value1

Value for the *relation* operation.

value2

Upper limit for the *RA* operation.

outlayer

Name of the output layer file.

c-name

Name of the output cell containing geometric results from the operation. This cell name cannot be more than seven characters.

l-num

Layer number of the output cell.

data-type

Datatype associated with the *l-num* of the output cell.

Example

```
*DESCRIPTION
.
.
LISTERROR = YES ; lists the first 100 node violations in
;               ; the log file
.
.
*END
.
.
*OPERATION
.
.
```

Dracula Reference

Operation Block Commands

```
LEXTRACT ANTE FIELD BY NODE ANTEN1
LEXTRACT ANTE GATE BY NODE ANTEN2

COMPUTE MIN ANTEN1 OUT1
COMPUTE MIN ANTEN1 ANTEN2 OUT2

CHKPAR A OUT1 POLY GT 1500 OUTPUT LONG 1
CHKPAR P OUT1 POLY GT 1500 OUTPUT LONG 2

CHKPAR A OUT2 POLY GT 25 OUTPUT WRONG 1
CHKPAR P OUT2 POLY GT 10 OUTPUT WRONG 2
.
*END
```

COEFFICIENT CAP

`COEFFICIENT CAP[type] value-a value-b value-c value-d`

Description

Associates the sidewall-up/down/colinear data coming from the NOT[E], AND[E], and PARASITC CAP commands with the process-dependent capacitance coefficients.

After using at least one of the NOT[E], AND[E] operations to extract capacitance sidewall information of deviceLayer, you can use COEFFICIENT CAP to gather this information for the cap. Without any NOT[E], AND[E] operation rules for the deviceLayer, it is illegal to write COEFFICIENT CAP for that device.

Arguments

type

The type specified in PARASITIC CAP[type] device.

value-a

Capacitance per unit area.

value-b

Capacitance per unit sidewall up perimeter.

value-c

Capacitance per unit sidewall down perimeter.

value-d

Capacitance per unit colinear edge perimeter.

Example

```
AND[E] M1T METAL2 M12
...
PARASITIC CAP[SW] M12 M1T METAL2
COEFFICIENT CAP[SW] 0.3 0.4 0.23 0.44
```

COMPUTE

`COMPUTE action par-file1 [par-file2 {par-file3}] outfile`

Description

For each node, calculates the value and ratio of the geometric parameters that you extract with the LEXTRACT command. When there is only one parameter file, Dracula calculates the value. When there are two or three parameter files, Dracula calculates both the value and the ratio. The AREA and PERI parameters are currently supported.

For more information on using the COMPUTE command for antenna checks, see “[*Extracting Area Ratios using the COMPUTE Function*](#)” section in the “Writing Rules for Dracula” chapter of the *Dracula User Guide*.

Arguments

action

Is one of the following values:

SUM	Total area of separate regions on the same node.
MIN	Minimum area of separate regions on the same node.
MAX	Maximum area of the separate regions on the same node.

mingate

Specifies the total area of all regions on the same node for each *par-file*, then calculates the ratio of the first result over the minimum of the second and third results on the same node.

par-file1

Layer to compare. It must be an output file from the LEXTRACT command.

par-file2

Layer to compare. It must be an output file from the LEXTRACT command.

par-file3

Layer to compare. It must be an output file from the LEXTRACT command. This layer is required only if you specify *mingate*.

Dracula Reference

Operation Block Commands

outfile

Name of the output file for the COMPUTE command. This is the input file for the CHKPAR command. Do not use more than seven characters in this file name.

Example 1

```
DESCRIPTION
PARSET ANTE AREA PERI
*END

LEXTRACT ANTE FIELD BY NODE ANTEN1
LEXTRACT ANTE GATE BY NODE ANTE2

COMPUTE MIN ANTEN1 OUT1
COMPUTE MIN ANTEN1 ANTEN2 OUT2

CHKPAR A OUT1 POLY GT 1500 OUTPUT LONG 1
CHKPAR P OUT1 POLY GT 1500 OUTPUT LONG 2

CHKPAR A OUT2 POLY GT 25 OUTPUT WRONG 1
CHKPAR P OUT2 POLY GT 10 OUTPUT WRONG 2
```

Example 2

```
LEXTRACT ANTE PLY2NG BY NODE ANTEN1
LEXTRACT ANTE NG BY NODE ANTEN2
LEXTRACT ANTE PG BY NODE ANTEN3

COMPUTE MINGATE ANTEN1 ANTEN2 ANTEN3 ANTOUT
CHKPAR A ANTOUT FPOLY2 GT 100.0 OUTPUT D513A 20
```

Note:

```
CHKPAR PAR OUT2 POLY GT 25 OUTPUT WRONG 1
```

is equivalent to

```
CHKPAR A OUT2 POLY GT 25 OUTPUT WRONG 1
```


CONNECT

`CONNECT layer-a layer-b BY cont-layer`

Description

Defines interlayer connectivity; that is, connects two layers by a set of contacts. For more information about connectivity, refer to the CONNECT-LAYER section in Chapter 12.

Arguments

layer-a

The first input layer name.

layer-b

The second input layer name.

cont-layer

The previously defined layer that consists of the contacts that connect *layer-a* to *layer-b*. *Cont-layer* must have at least three characters, and the first three characters must be unique. The fourth character must be a letter, although the name can contain numeric characters (*cont1* is allowed, but *con1* is not). The preprocessor assigns the highest priority layer as the master layer. *Cont-layer* can connect only from the master layer to any other layer.

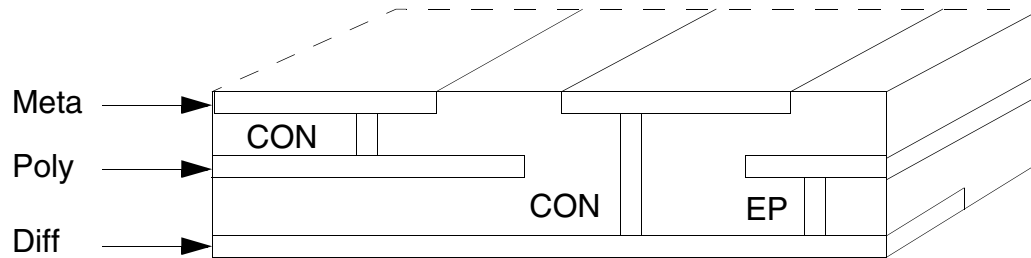
Example

```
CONNECT metal poly BY cont
CONNECT metal diff BY cont
CONNECT poly diff BY epi
CONNECT poly diff BY cont ; (not allowed)
```

Dracula Reference

Operation Block Commands

Note: Master layer for *cont-layer* is metal. *Cont-layer* cannot connect *poly* to *diff*. The preprocessor checks for this condition.



CORNER

```
CORNER {[option]} layer-a [relation-a] [relation-b]
        {{NOT} LT/GT/RANGE angl {ang2} }
        {CORNER-SIZE n} output-layer {OUTPUT c-name l-num {d-num}}
```

Description

Recognizes polygon corners and reports them to the specified output layer for special DRC checking.

Arguments

options

If you do not specify any options, this command works for Manhattan angles.

A

Reports 90 degree angles.

B

Reports 45 degree angles.

C

Reports any angle.

layer-a

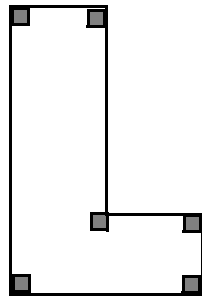
Input layer name.

Dracula Reference

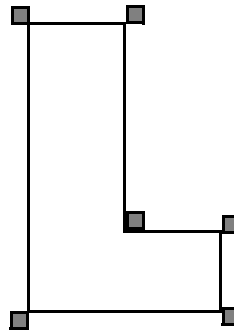
Operation Block Commands

relation-a

INSIDE or OUTSIDE. INSIDE creates a rectangle the size of CORNER-SIZE on the inside of shapes. OUTSIDE creates a rectangle the size of CORNER-SIZE at all corners on the outside of shapes.



CORNER



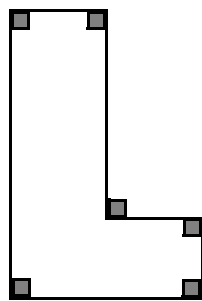
CORNER

Dracula Reference

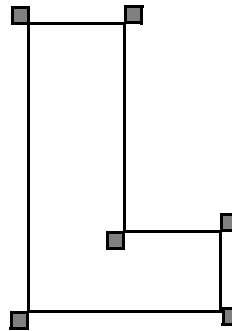
Operation Block Commands

relation-b

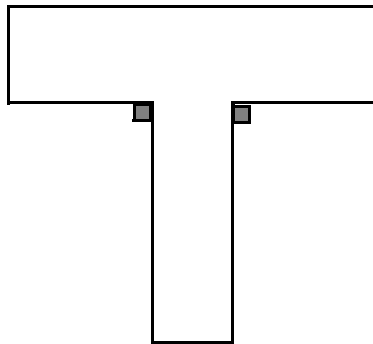
INNER or OUTER. INNER creates a rectangle the size of CORNER-SIZE on the inside edge of all corners. OUTER creates a rectangle the size of CORNER-SIZE on the outside edge of all shapes.



CORNER INNER



CORNER OUTER



CORNER OUTSIDE INNER

{NOT} [LT | GT | RANGE] *ang1* {*ang2*}

This operation flags only corners whose angle matches the given constraint (see Examples 4 and 5 for details). If the keyword RANGE is specified, a second angle (*ang2*) must also be used.

Note: The keyword NOT can be specified to flag corners that do not meet the specified constraint.

CORNER-SIZE n

Corner size in user units (*n*). If you do not specify this option, the default value is 2 times the resolution unit. If you specify a corner size of one resolution unit, round-off might occur at non-Manhattan corners. Dracula issues a warning if you specify a corner size of one resolution unit.

Dracula Reference

Operation Block Commands

output-layer

Output layer name.

OUTPUT

Sends the results of the operation to an output cell.

c-name l-num

C-name is the name of the cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the layer number determined by your CAD system.

d-num

The datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

Example 1

The following example reports Manhattan angles and creates a rectangle of the inside edge of all shapes and corners. The output is sent to *SUB001*.

```
CORNER ME1 INSIDE INNER SUB001
```

Example 2

This example reports Manhattan angles and creates a rectangle of .03 microns at all corners on the outside of shapes. The output is sent to *SUB002*.

```
CORNER ME1 OUTSIDE INNER CORNER-SIZE .03 SUB002
```

Example 3

This example reports 90 degree angles and creates rectangles on the inside edge of all shapes and corners. The output is sent to *SUB004*.

```
CORNER[A] ME1 INSIDE INNER SUB004
```

Example 4

The following checks all acute angles within a tolerance of one degree:

```
CORNER MET LT 89 out acute 1 0
```

Dracula Reference

Operation Block Commands

Example 5

The following gets all 45 and 315-degree corners (but not 135 and 225) and writes them to layer COR1 for further processing:

```
CORNER RA 44.5 45.5 COR1
```

COVERAGE

```
COVERAGE {NOT} input-layer [LT/LE/GT/GE/RA] per1 {per2} windowSize stepSize  
  {CENTRIC} {OUTSQU square-file} {SQSZ size-of-squares}  
  {BOUND boundary-layer} {trapfile}  
  {OUTPUT c-name l-num }
```

```
COVERAGE {NOT} input-layer [LT/GT/RANGE] per1 {per2} [RECT rectangle-layer/  
  BASLAY base-layer] {trapfile} {OUTPUT c-name l-num }
```

```
COVERAGE {NOT} input-layer [LT/GT/RANGE] percentage windowSize stepSize MIN-  
  RATIO minRatio BOUND boundary-layer {trapfile} {OUTPUT c-name l-num  
  {d-num}}
```

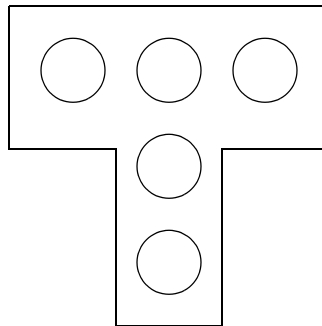
Description

Dracula reports an error if the area surrounding a specified point in the layout is not covered by a specified area percentage.

This command is intended for flat mode only. If you use it in a hierarchical mode, the input layers must be flattened.

In version 4.81 and later versions, Dracula supports region-based density check to implement the process rules in finding the via and bent gate density. In the 4.7 and previous versions of Dracula, you could not implement such rules as only window-based check was supported.

Given below is a T-shaped area over which is an O-shaped area and region-based density check allows you to do a density check of the O-shaped area.



Dracula Reference

Operation Block Commands

A region-based density sums up all the O shape area which is then divided by the T shape area, that is,

$$\text{O shape density over T shape} = \frac{(\text{sum of O shape areas within T shape})}{(\text{T shape area})}$$

Checking Methods

modes

Flat and hierarchical (internally flattened)

Arguments



You must supply only one of these measurements given below.

input-layer

Specifies the input layer name

LT/LE *percentage*

The coverage rate is less than or equal to the *percentage*.

GT/GE *percentage*

The coverage rate is greater than or equal to the *percentage*.

RA *per1 per2*

The coverage rate is between *per1* and *per2*.

windowSize

Specifies the window size in each step of the coverage check. The *windowSize* must be a multiple of *stepSize*.

stepSize

Specifies the step size value to be used.

CENTRIC

A new option added to Dracula 4.81 and subsequent versions. This option defines the central location of squares reported into the *trapfile* and/or output, relative to the violating window. If you don't specify this option, the *trapfile* and/or output will contain the entire merged violating windows. The [“Example 11 - Example for the CENTRIC Option”](#) section provides more information.

Dracula Reference

Operation Block Commands

This enhancement lets you have the possibility to obtain the results of the COVERAGE operation in Dracula verification in an equivalent form to that of Assura verification.

OUTSQU

Specifies the layer created by the SIZE operation. The output-layer can have up to seven characters and must have no special characters. You can use A-Z and 0-9, but the first character must be a letter.

square-file

Writes the step squares in the violating window that has ratio within the percentage as a trapezoid file.

SQSZ

The size of squares in *square-file*.

size-of-square

If you don't specify this parameter, the size of the squares is the same as the window step size.

MIN-RATIO

Keyword used to specify the minimum area occupied ratio of the boundary layer in one check window to trigger the checking while doing density check.

minRatio

Specify the minimum area occupied ratio of the boundary layer in one check-window to trigger the checking. If the area occupied ratio of boundary-layer in the check-window is less than the *minRatio*, the check-window will be ignored and Dracula will report the relative information to the log file; otherwise, Dracula will calculate the density based on the "overlap region" of check-window and the boundary-layer and output the violating check-window. See "[Example 10 - Example for minRatio](#)" in this section for more information.

BOUND

Keyword used to specify boundary layer while doing density check.

boundary-layer

Trapfile with rectangle(s) which define the boundary of the COVERAGE check. The boundary can be larger or smaller than the primary cell boundary. You can specify multiple rectangle in the trapfile. COVERAGE will do the check in each rectangle one by one.

Dracula Reference

Operation Block Commands

trapfile

Output layer with trapezoid file formats. It can be used as input layer by the subsequent commands.

OUTPUT

Sends the results of the operation to an output cell.

c-num

Is the name of the cell and can have six alphanumeric characters or fewer and no special characters.

l-num

Is the layer number determined by your CAD system.

RECT *rectangle-layer*

Region-based density checking within the constraints of a rectangular shape. For each rectangle in the rectangle-layer, Dracula computes this equation:

$$\text{ratio} = (\text{area of shapes inside rectangle}) / (\text{area of rectangle})$$

Supports only the layer with rectangular shapes

BASLAY *base-layer*

For each region in the base layer, COVERAGE computes this equation:

$$\text{density ratio} = (\text{area of shapes inside the region}) / (\text{area of the region})$$

Base-layer supports the layers with any kind of shapes.

{NOT} Reverse

Output ratio not in the range.

Example 1

This example configures the chip as a grid with a step of 10 microns and reports an error if the surrounding 100x100 micron square area of a specific grid point is not covered by an area percentage of 60 percent.

```
COVERAGE POLY LT 0.6 100 10 POLYERR
```

Example 2

```
COVERAGE METAL LT 0.5 100 10 METALC1
```

Dracula Reference

Operation Block Commands

Example 3

```
COVERAGE metal1 LT 0.3 100 10 merr1
```

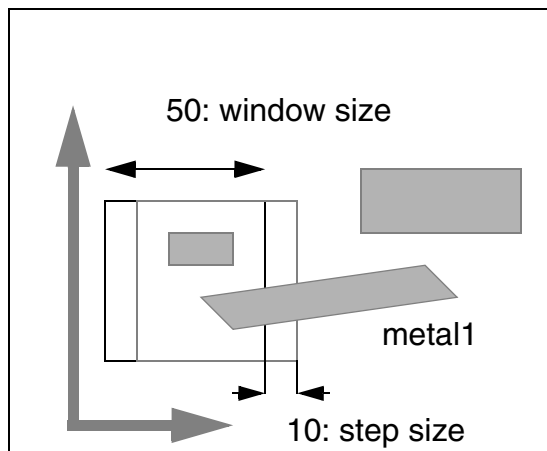
This prints the output:

X	Y	Ratio
100	.23	924.34
102.5	123.52	0.15

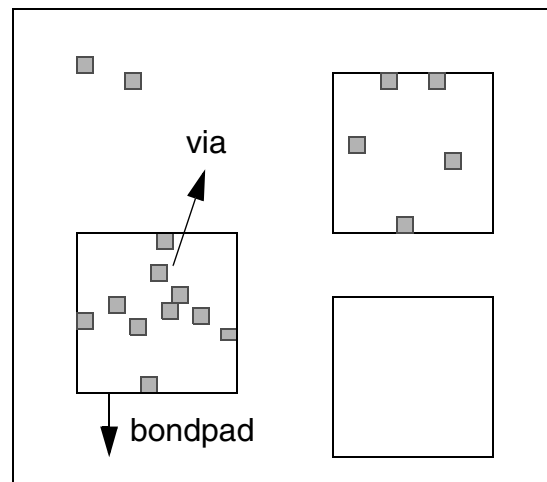
Example 4

```
COVERAGE metal1 LT 0.3 50 10 OUTPUT merr 1
```

Window Step Checking



Rectangular Window Checking



Example 5

```
COVERAGE via LT 0.05 RECT bondpad OUTPUT viabond 1
```

For each bond pad area, computes this equation:

$$\text{ratio} = (\text{area of vias inside bondpad}) / (\text{area of bondpad})$$

If the ratio is less than 0.05 the Dracula software reports it.

Example 6

```
COVERAGE metal LT 0.32 100 20 OUTSQU spa_met SQSZ 18 OUTPUT err 1
```

```
SIZE metal by 4 ttl
```

```
NOT spa_mnet ttl fsqu OUTPUT fillsqu 1
```

Dracula Reference

Operation Block Commands

The above example uses OUTSQU to fill the empty slots of the metal layer. The gap between each fill square is $2u(20u-18u)$. The filling squares are $4u$ away from the original metal.

Example 7

(Checking and generate rectangles for area filling)

```
SIZE BULK by 2.0 mybound
COVERAGE metal LT 0.2 50 10 BOUND mybound ERRWIN
GENRECT errwin XYLEN 2.0 3.0 XYGAP 1.0 1.2
```

(GENRECT generates rectangles that overlap the input layer and would also like to fill the area outside the cell.)

Note: The original BULK is already oversized by $1u$ from cell area. In the above example, the checking boundary is totally oversized from cell area by $3.0u$.

Example 8

```
COVERAGE met2 LT 0.2 50 10 BOUND cframe output err 1
```

Example 9

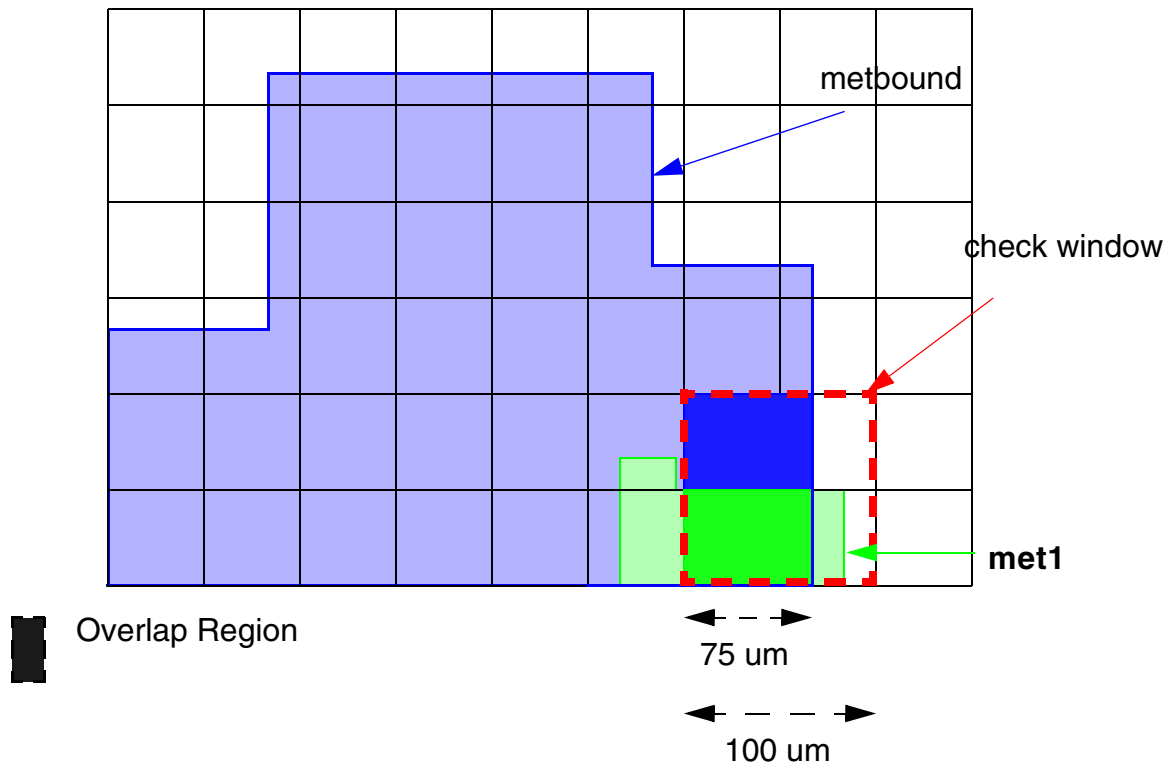
Given below is an example of range-based density check.

```
COVERAGE O-trapfile LT 0.3 BASLAY T-trapfile itrapout OUTPUT merr 1
```

Dracula Reference

Operation Block Commands

Example 10 - Example for minRatio



For the above layout,

```
COVERAGE met1 LT 0.6 100 50 MIN-RATIO 30 BOUND=metbound OUTPUT meterr 0 0
```

Dracula first checks the area occupied ratio of this check-window:

$$\text{area occupied ratio} = (75 \times 100 / 100 \times 100) = 75 \% > \text{minRatio} = 30 \%$$

Thus, it continues to check the density for this check-window:

$$\begin{aligned} \text{density} &= \text{area of input layer inside the boundary-layer} / \text{area of overlap region} \\ &= (75 \times 50 / 75 \times 100) = 50 \% \end{aligned}$$

Dracula finally outputs the violating check-window into the output layer.

Dracula Reference

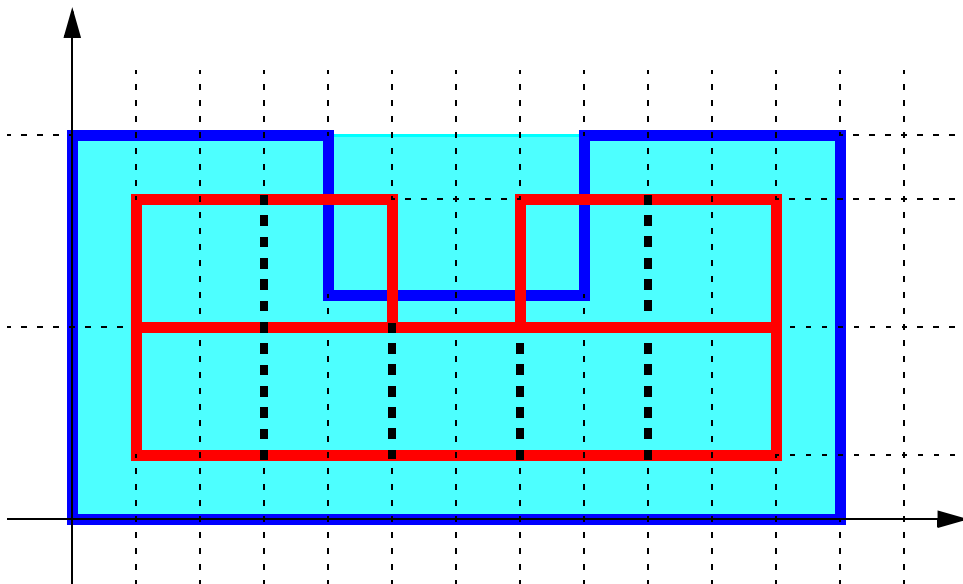
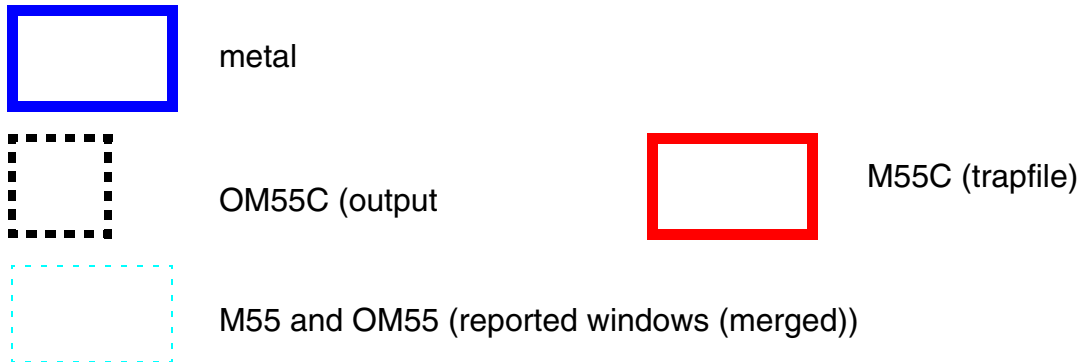
Operation Block Commands

Example 11 - Example for the CENTRIC Option

SIZE BULK BY 4 OM

COVERAGE METAL GE 0.55 4 2 BOUND OM M55 OUTPUT OM55 0 0

COVERAGE METAL GE 0.55 4 2 CENTRIC BOUND OM M55C OUTPUT OM55C 0 0



CUT

`CUT{[STRIP]} layer-a {layer-b} trapfile`

Description

Cuts out geometries and uses them to generate a new layer for PRE. The CUT command does not cut polygons, so you must convert all polygons to trapezoids before using the CUT command.

Note: The CUT-TERM command replaces the CUT command. Use CUT-TERM instead of CUT in new rules files.

You can use the resistor recognition layer as a terminal layer, but you cannot use layers derived from the resistor recognition layer as a terminal layer.

Checking Methods

modes

Flat and composite.

checks

PRE and HPRE.

Arguments

STRIP

Produces a one-resolution-unit wide strip on opposite edges of any rectangular angle on *layer-a*. Dracula forms the strip on the inside of the trapezoid being cut. Use the STRIP option only on rectangular geometries.

layer-a

The layer whose trapezoids are to be cut.

layer-b

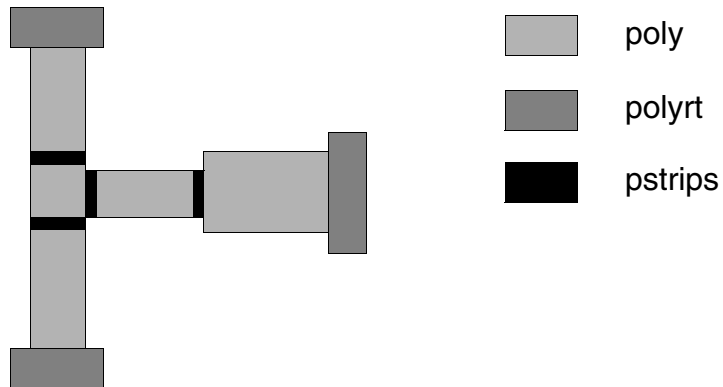
Use with the STRIP option. In the case of parasitic resistors, this layer typically corresponds to the resistor terminal layers generated so far. This layer guides the strip cuts in a direction perpendicular to the flow of current.

Dracula Reference

Operation Block Commands

trapfile

The trapezoid file created by the CUT command. The *trapfile* consists of trapezoids cut from *layer-a* that satisfy the condition.



CUT-TERM

```
CUT-TERM cond-layer cont-layer res-layer term-layer  
        {term-dev} {MAXNS=length} {MAXWIDTH=width}  
        {ALIGN=YES/NO} {45CORNER=compenFactor}  
  
CUT-TERM[P] cond-layer cont-layer res-layer term-layer  
            {term-dev} {MAXNS=length} {MAXWIDTH=width}  
            {ALIGN=YES/NO} {45CORNER=compenFactor} {WLRATIO=value}
```

Description

Generates a resistor body layer and a resistor terminal layer from a conduction layer and a contact layer. PRE uses the output resistor body and terminal layers to compute parasitic resistance and capacitance.

Note: You can use the resistor recognition layer as a terminal layer, but you cannot use layers derived from the resistor recognition layer as a terminal layer.

The CUT-TERM command cuts the following elements:

- Contact terminals
Cuts contact regions from the resistor layer.
- Junction terminals
Cuts junction areas from the resistor layer. A junction is a 90-degree corner, a T-junction, or a cross-shaped junction.
- Long wires
Cuts long geometry strips into smaller segments. A long strip geometry is more accurately represented as a series of RC lumped circuits.

CUT-TERM replaces a series of commands previously required to achieve similar results. CUT-TERM is the preferred method for new rules files because it produces more accurate results. However, you can still use the old commands instead of CUT-TERM. For a sample listing of the commands CUT-TERM replaces, refer to the following “Example” section.

Checking Methods

modes

Flat and composite.

checks

PRE and HPRE.

Arguments

cond-layer

Name of the conduction input layer.

cont-layer

Name of the contact input layer.

res-layer

Name of the resistor body output layer.

term-layer

Name of the resistor terminal output layer.

term-dev

All device terminal layers belonging to *cond-layer*.

MAXNS=*length*

Maximum length of a parasitic resistor device specified as a number of squares. If a resistor exceeds the number of squares you specify (MAXNS), the resistor is partitioned into multiple devices evenly.

Each resistor devices formed is smaller than the MAXNS value, and the values are distributed evenly on the resistor. The maximum difference between two sections is 1. For example, if the resistor body is 91, and you specify a MAXNS value of 20, the sections are 19, 18, 18, 18, 18.

MAXWIDTH=*width*

Value used by the WIDTH command within the command sequence that CUT-TERM executes. The default *width* is 40. For an example of how CUT-TERM uses MAXWIDTH=*value* in the WIDTH command see **Example 1** below.

Dracula Reference

Operation Block Commands

`ALIGN=YES/NO`

If you specify `ALIGN=YES`, `CUT-TERM` forms the contact terminal at the alignment edge of the conduction layer and the contact layer. The edge of the terminal layer aligns with the contact as shown below.

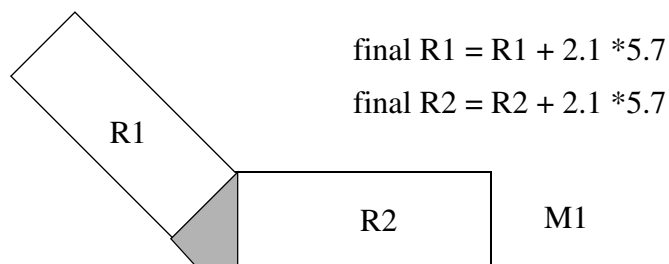


If you specify `ALIGN=NO` (or, if you do not specify the `ALIGN` argument), the terminal layer is formed by extending one-half the width of the conduction layer from the contact as shown below.



`45CORNER=compenFactor`

The number of square to be added as resistance compensation for a 45 degree terminal. The final compensation value will be equal to $(\text{sheetResistance} * \text{compenFactor})$. For example, the following layout, the extracted resistance will be,



$$\text{final R1} = \text{R1} + 2.1 * 5.7$$

$$\text{final R2} = \text{R2} + 2.1 * 5.7$$

`CUT-TERM M1 CONT M1T M1R 45corner=2.1`

...
`PARASITIC RES[P] M1R M1T`
`ATTRIBUTE RES[P] 5.7`

`WLRATIO=value`

This option is only available for use with the `CUT-TERM[P]` option, which enables creating resistor body and terminal layers when the ratio of the width over the length is

Dracula Reference

Operation Block Commands

less than or equal to the specified *value* ($W/L \leq WLRATIO$) . The default value for WLRATIO is 1.

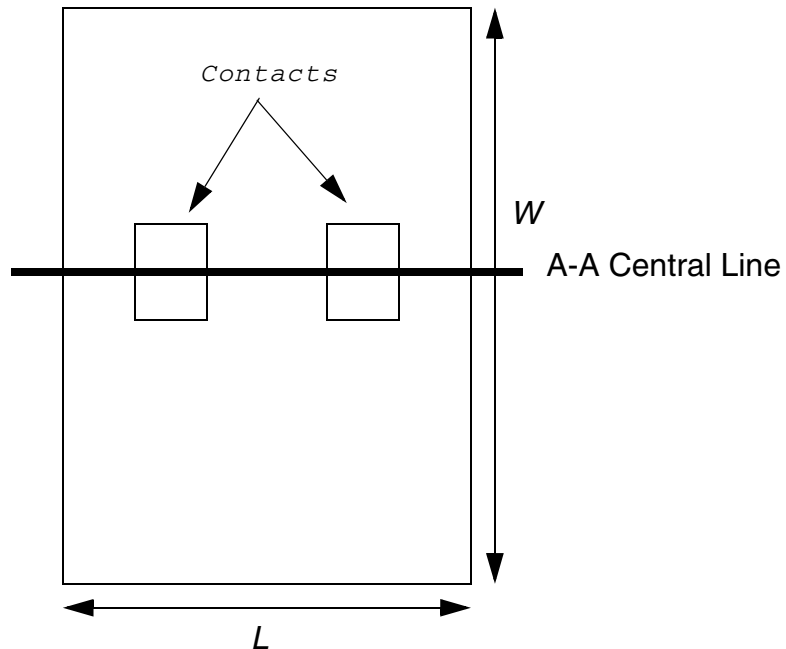
Option [P]

CUT-TERM[P]

The CUT-TERM[P] option supports improved accuracy of parasitic resistor layer cutting for special circumstances. This results in more accurate calculation of the resistor value.

1.) When there is a conducting layer with two contacts and they are placed near the longer side (see below).

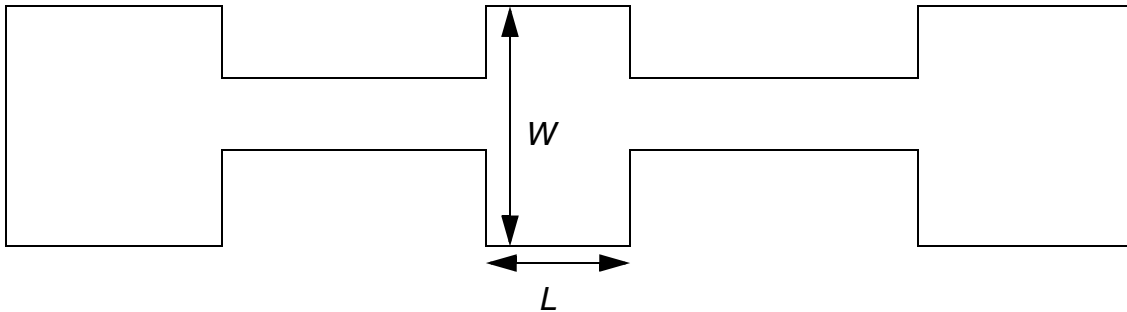
When the width is greater than the length ($W > L$) only the terminal layer is created. Using the CUT-TERM[P] option enables creating both the resistor body layer and a resistor terminal layer in these cases.



Dracula Reference

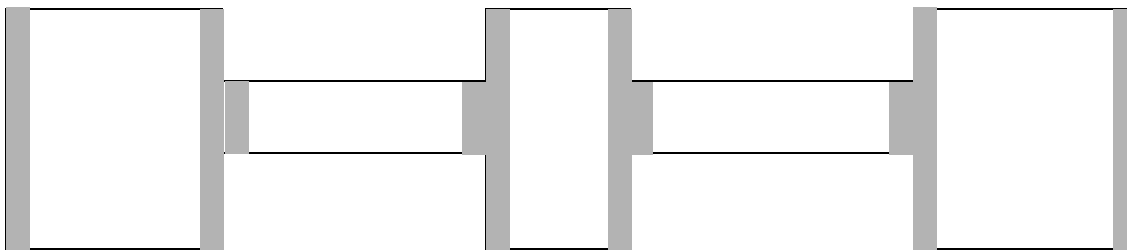
Operation Block Commands

2.) In the following case, the width is greater than the length so only the terminal layer is created for the center piece of the conductive layer below.



Using the `CUT-TERM[P]` option enables creating the resistor body layer and terminal layer for each layer.

Note: In order to provide more accurate extraction there should be two parasitic terminals for each layer (see below):



■ terminal layer with width of 0.001 micron

You should use the `CUT-TERM[P]` option only when there are no 'X' or 'T' connections of metal lines on the layers. In this case wide resistors will be just wide resistors.

Example 1

```
CUT-TERM R_IN C_IN R_OUT T_OUT T_DEV MAXNS=20 MAXWIDTH=50
```

This runs the following command sequence in flat mode. Angle brackets (<>) surround temporary file names. An asterisk (*) indicates internal modules used by CUT-TERM.

```
*PXCONT R_IN C_IN <TX>
OR <TX> <TX> <T1>
NOT R_IN <T1> <R1>
*PXJUNC R_IN <TX>
OR <TX> <TX> <T2>
NOT <R1> <T2> <R2>
AND      R_IN T_DEV <TC>          ;Add AND and NOT only if
NOT <R2> <TC> <R2>                ;you specify T_DEV.
WIDTH[CR] <R2> LE 50 <R3>        ;50 is the MAXWIDTH value.
```

Dracula Reference

Operation Block Commands

```
NOT <R2> <R3> <T3>
OR <T1> <T2> <T4>
OR <T3> <T4> <T5>
OR <T5> <TC> <T5> ;Add only if you specify T_DEV.
CUT STRIP <R3> <T5> <T6>
OR <T5> <T6> <T7>
NOT <R3> <T7> <R4>
*PXLONG <R4> <T8> 20
NOT <R4> <T8> <R5>
OR <T7> <T8> <T9>
OR <T9> <T0> <TA>
SELECT <R5> TOUCH[2:2] <TA> R_OUT
NOT <R5> R_OUT <TB>
OR <TA> <TB> T_OUT
```

The following modules are internal to CUT-TERM. They are included in this listing to clarify how CUT-TERM works. Do not include them in your rules file.

- PXCONT cuts contact regions into a terminal layer.
- PXJUNC cuts corner, cross-shaped, and T-junctions into a terminal layer.
- PXLONG cuts a long strip into shorter segments.

Example 2

```
CUT-TERM[P] R_IN C_IN R_OUT T_OUT T_DEV MAXNS=20 MAXWIDTH=50 WLRATIO = 2.8
```

Example 3

In this example WLRATIO = 1

```
CUT-TERM[P] R_IN C_IN R_OUT T_OUT T_DEV MAXNS=20 MAXWIDTH=50
```

Example 4

In this case PDRACULA writes WARNING messages and the WLRATIO is ignored.

```
CUT-TERM R_IN C_IN R_OUT T_OUT T_DEV MAXNS=20 MAXWIDTH=50 WLRATIO = 2.8
```

DENC

```
DENC[TOEWL] layer1 layer2 {LT/RANGE value {value}} {CORNER-CORNER LT/RANGE value1  
  {value2}} {CORNER-EDGE LT/RANGE value3 {value4}} {CORNER-OICORN LT/RA value5}  
  {CERRGE value6} {OPPGT value} {DEPTHGT value8} {CORRLEN value} {OUTCORR  
  correction-output} {OUTPUT errfile}
```

Description

Deep submicron ENC command. It can do metal line end error correction output, and double side enclosure check. It works for flat mode only. For more details about the ENC command, see the "[ENC](#)" command section in this chapter.

Arguments

layer1

First input layer name containing the enclosed polygons.

layer2

Second input layer name containing the enclosing polygons.

[T]

Touch option. It is filtered by all filters.

[O]

Checks partially outside shapes. It doesn't report shapes fully outside. It is not filtered by any filter.

[E]

Checks external, fully outside shapes. It is not filtered by any filter. Only accepts rectangles in *layer1*.

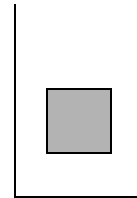
[W]

Similar to `ENC [W]`. Report the error only if the error edge on the metal is the shorter edge (metal line end). [W] option is not filtered by CERRGE. The [W] option only handles simple

Dracula Reference

Operation Block Commands

shapes as follows. When you use both CERRGE and the [W] option you can check most of the via/metal enclosure cases.



[L]

While the [W] option checks only simple shapes, this option was designed to check more complex conditions. The [L] option reports errors only if the error is on a metal line end.

The following is a description, how the end of the metal line is identified by this option:

1.) If both angles at the ends of an edge are less than 180 degrees. In Figure 1 below, Case a) is the end of a metal line (the thick black line indicates the edge of shape, thin line on the gray background indicates the direction to the inside of the shape). Case b) and Case c) are not.

2.) The length of the edge on a metal line should be less than its length in an orthogonal direction. In Figure 2 below, the green shaded area is a metal shape, and the blue rectangle is a via. For a violation between the upper left corner of the via and point C, the length of AB is compared with the length of CD. For a violation between the lower left corner of the via and point E, the length of AB is compared with the length of EF.

Figure 1

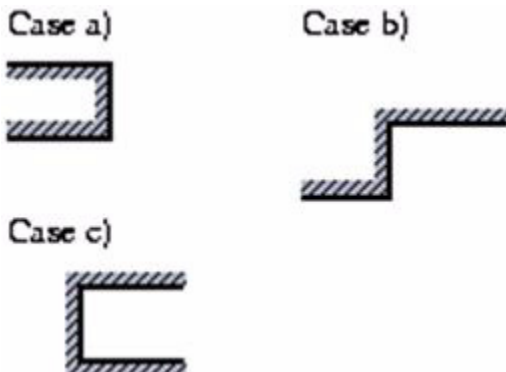
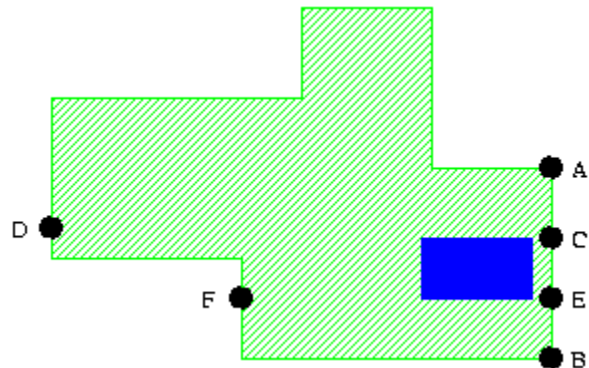


Figure 2

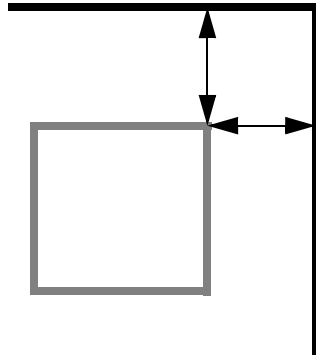


Dracula Reference

Operation Block Commands

D

Report the errors if there are at least double CORNER-EDGE errors at a corner.



LT / RANGE

If you use `LT / RANGE` first, `CORNER-CORNER`, `CORNER-EDGE` and `CORNER-OICOR` will be initialized to the specified value, then you can override them.

`CORNER-EDGE`

Reports the errors on the corner-to-edge distance

`CORNER-CORNER`

Reports the errors on the corner-to-corner distance. Default value is the same as `CORNER-EDGE`

`CORNER-OICORN`

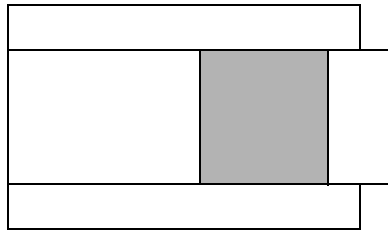
Reports the errors on the corner-to-outside inner corner distance. Default value is the same as `CORNER-EDGE`. This argument only works for manhattan shapes

Dracula Reference

Operation Block Commands

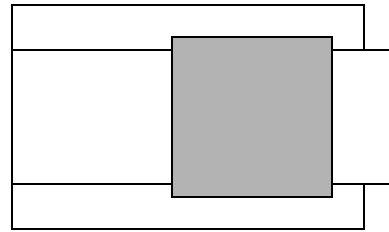
CERRGE

Reports errors only if the `CORNER-CORNER` error count of the shape is greater than or equal to the specified value (This argument only works for rectangles). In the following figure, case A will be filtered out automatically, though it has `CORNER-CORNER` errors.



A

No metal line ending error though it has
corner-corner errors

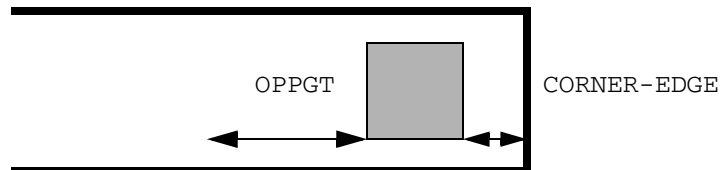


B

True error

OPPGT

If there is an error, reports the error only if the enclosure of the opposite side is greater than the `OPPGT` value. The `OPPGT` must be greater than the `CORNER-EDGE` value. It is not filtered by `CERRGE`.

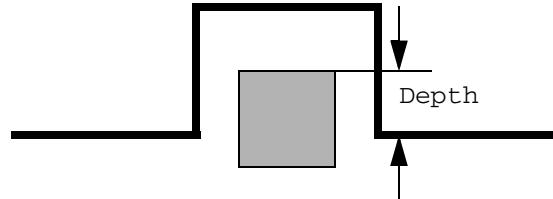


Dracula Reference

Operation Block Commands

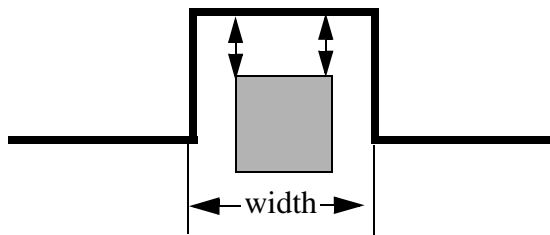
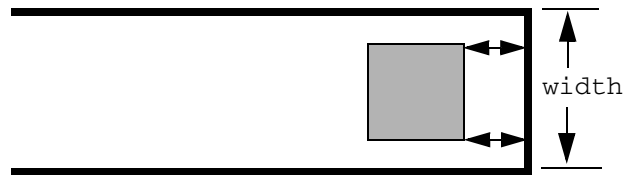
DEPTHGT

If the via is inside the T shape, you can use the depth filter. There must be at least 2 CORNER-CORNER , 4 CORNER-EDGE, and 2 OICORN-EDGE situations . (Only works for rectangles. You must use the CERRGE option).



WIDLT

Reproduce this error only if the width is less than the specified value.



OUTCORR

Outputs the metal line end error correction

Note: The restrictions for this option are as follows:

- a. You must use CERRGE.
- b. The edges of the via must be parallel to the adjacent edges of metal.
- c. The end edge of the metal must not touch the ending edge of the via.

Dracula Reference

Operation Block Commands

CORRLLEN

Single and triple-corner error correction length. Default value is $\text{CORNER} - \text{CORNER} / \sqrt{2}$.

OUTPUT

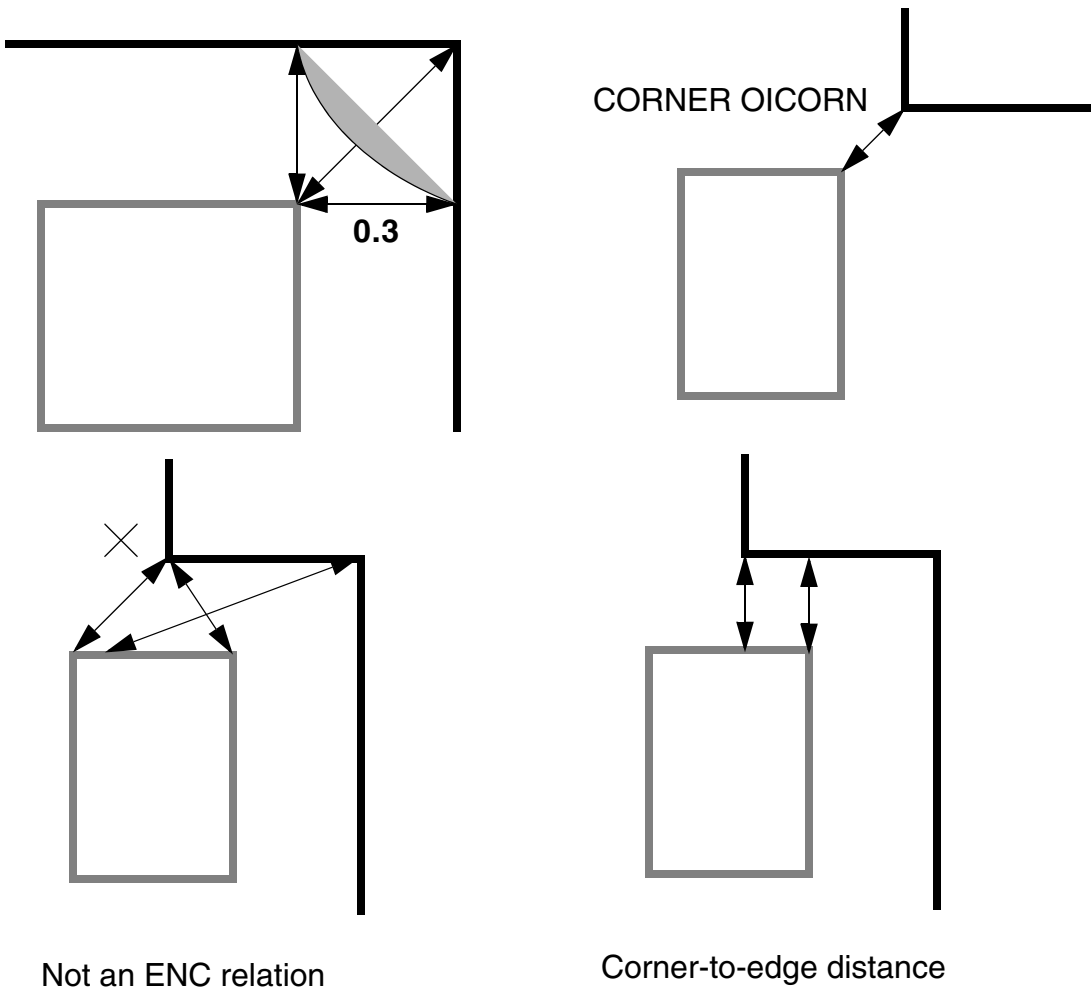
Send the results to an error cell

errfile

Error cell file name

Example 1

```
DENC VIA METAL LT 0.3 OUTPUT vmerr 1
```



Dracula Reference

Operation Block Commands

Example 2

```
DENC VIA METAL CORNER-EDGE LT 0.3 CORNER-CORNER LT 0.4 OUTPUT vmerr 1
```

Example 3

Using CERRGE to ensure there is enough metal around via

Case 1

For example, X, not a real value

```
DENC[TOE] via metal LT 0.1 OUTPUT vmerr 1
```

```
DENC[TL] via metal LT 0.2 CORNER-CORNER LT 0.22 CERRGE 1 OUTPUT vmerr 2  
sqrt(0.1**2 + 0.2**2) = 0.22
```

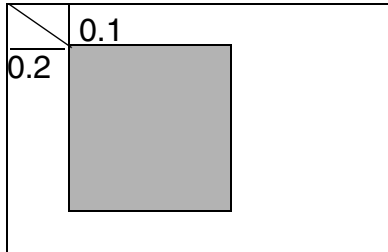
Case 2

For example, Y, not a real value.

Dracula Reference

Operation Block Commands

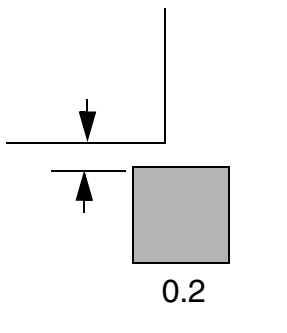
DENC[TOEL] VIA METAL LT 0.2 CERRGE 1 OUTPUT vmerr 2



Case 1



Case 2



Minimal clearance X:

Case 1 : 0.1

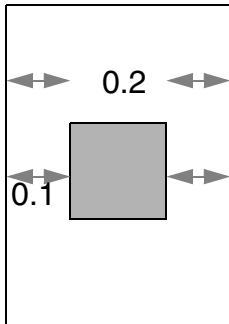
Case 2 : 0

Dracula Reference

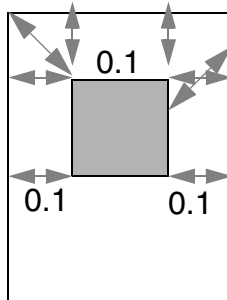
Operation Block Commands

Examples of CERRGE and [L] for both Case 1 and Case 2

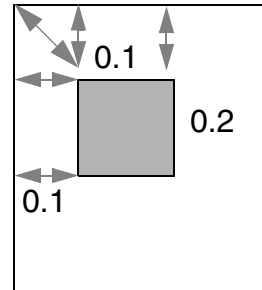
C-C = CORNER-CORNER count for CERRGE



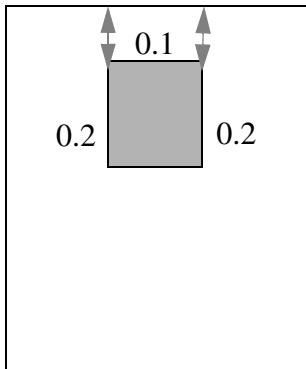
C-C = 0
OK



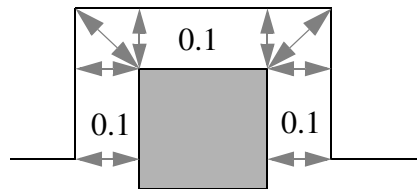
C-C = 2
ERROR



C-C = 1
ERROR



C-C=0
ERROR because of [L]

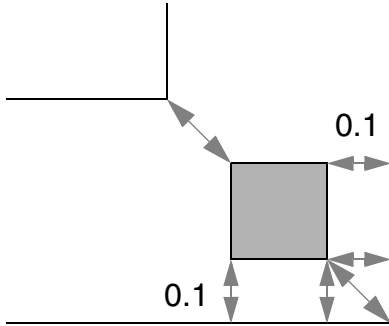


C-C = 2
ERROR (You might want to use the
DEPTHGT option)

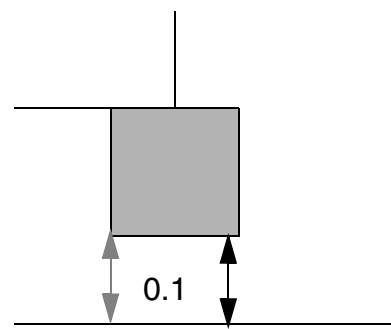
Dracula Reference

Operation Block Commands

Examples of CERRGE and [L] for both Case 1 and Case 2 (*continued*)

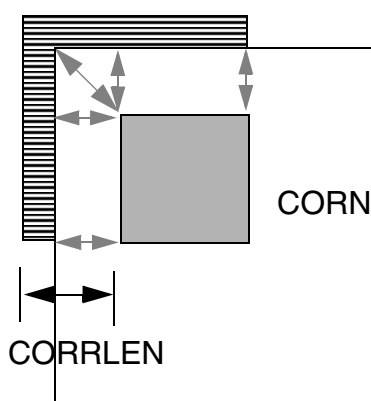


C-C = 1
ERROR

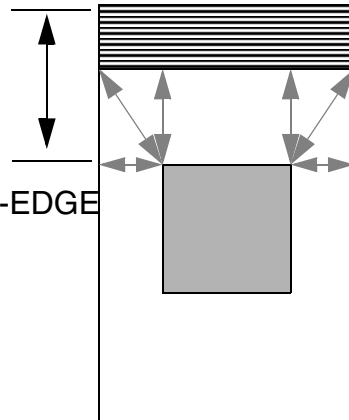


ERROR because of [L], only solid is reported

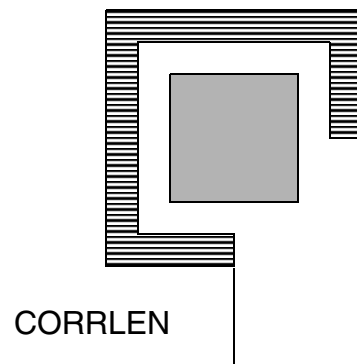
OUTCORR: OUTput error CORRection



1 CORNER-CORNER error
4 CORNER-EDGE errors



2 CORNER-CORNER errors
2x2 CORNER-EDGE errors



3 CORNER-CORNER
errors

DEVTAG

```
DEVTAG element[type] layer-b layer-c  
      (for tagging from a defined ELEMENT BJT device)  
  
DEVTAG[L] layer-a layer-b layer-c  
      (for tagging from an intermediate layer)  
  
DEVTAG[S] BJT[type] layer-d layer-e  
  
DEVTAG[LS] layer-c layer-f layer-g
```

Description

Tags multiple layers that are parts of a device with device numbers assigned by the device layer. You must define the device in the ELEMENT command before using this command. The DEVTAG command links device numbers of the layers that form multiple emitter or collector bipolar devices in the layout database.

The DEVTAG command tags device number information onto the tagged layer (*layer-b*) from the tagging layer (device layer or *layer-a*) when polygons in the tagging layer overlap or touch polygons in the tagged layer. The DEVTAG command outputs tagged polygons to output *layer-c*.

Multiple tagging is allowed in bipolar devices where you can link a polygon to several devices. When the tagged layer (*layer-b*) is not directly overlapped to the device layer, an intermediate layer (such as a SIZE layer) can tag the device number information. You do not have to tag the device layer in the ELEMENT command.

While extracting parameters from a multiple-terminal device, make sure the function or use of all generated layers from which you are extracting parameters represents the exact device (for example, MOS: gate, source-drain, and body; BJT: collector, base, emitter, and so forth). DEVTAG identifies the devices by the layer names you specify in the ELEMENT command. Thus, either the tagged layer you are extracting (*layer-b*) or the intermediate layers in the tagging sequence must correspond to the functional layers in the ELEMENT command.

Note: When you use more than one DEVTAG command in a rules file, they must be grouped together. You cannot use any other command between a group of DEVTAG commands. When running an LVSCCHK or LPECHK job, the DEVTAG command must be grouped with the LEXTRACT, DEVTAG, LVSCCHK, or LPECHK commands.

DEVTAG[S] is used when extracting lateral PNP devices that have multi-collectors with one emitter area. Use this command to distribute the emitter area evenly to all the BJT transistors that share the same emitter in the SPICE output.

When you use the [LS] option with the DEVTAG command, it means that you are

- Tagging from an intermediate layer (*layer-c*), generated from the previous DEVTAG command to the tagged layer (*layer-f*) and putting the tagged result to the output layer (*layer-g*)
- Distributing the parameter values of the tagged result layer (*layer-g*) evenly to all the devices that share the same terminal area

The LS option is mainly used for multi-collectors with one emitter area and the emitter area is not directly overlapped to the device layer. See the section, “Example of Lateral Pnp with Possible Multiple Collector” for more information.

Arguments

element[type]

Type of bipolar device used in the ELEMENT BJT[*type*] command. This ELEMENT[*type*] is the tagging device for tagging directly from the bipolar device (ELEMENT BJT). For example, BJT[NV] for a vertical *nnp* and BJT[LP] for a lateral *pnp*.

layer-a

Tagging (intermediate) layer. This layer must have been previously tagged with a DEVTAG (tagged from a defined ELEMENT BJT device). You must specify the *L* option.

layer-b

Tagged layer. Contains groups of polygons, usually of the multiple collectors or emitters of a bipolar device. The specified layer carries the same device number as the tagging layer that touches or overlaps *layer-b*.

layer-c

Output layer (intermediate layer) containing trapezoids with tagged device number information. Specify this layer for LPE only. This layer is generated from a previous tagged DEVTAG command.

layer-d

Emitter layer of BJT element that you need to distribute.

layer-e

Tagging intermediate layer (see *layer-a* for more information).

layer-f

Tagged layer (see *layer-b* for more information).

layer-g

Tagged result layer (see *layer-c* for more information).

Dracula Reference

Operation Block Commands

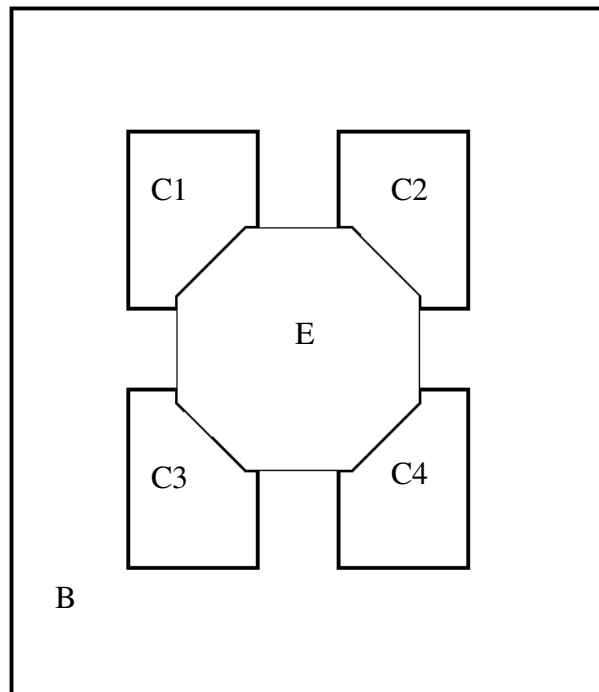
Examples

In this example, the device numbers for BJT[lp] are tagged to *oemit*, then from *oemit* to *emit* and *output* to *emitr*. The *emit1* is an intermediate layer. Through the DEVTAG command sequence, *emitr* can be traced back to *oemit*, which is the emitter layer on the ELEMENT BJT command. For more information, see [Chapter 7, “Extracting Electrical Parameters \(LPE\).”](#)

```
ELEMENT BJT[lp]    coll    coll    base    oemit
DEVTAG  BJT[lp]    oemit    emit1
DEVTAG[L]          emit1    emit    emitr
```

The following is an example of the type of lateral PNP devices. Since the collectors are used as device recognition layers, there are 4 BJT transistors defined.

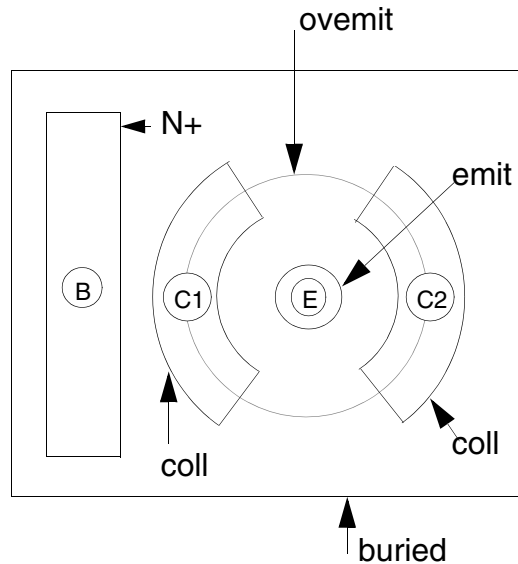
```
PARSET BJT1 AREA  PERI EA EP
ELEMENT BJT[LP]  COLLPN COLLPN BASELPN EMITLPN
DEVTAG[S] BJT[LP] EMITLPN BMIT1
LEXTRACT BJT1 BMIT1 BY BJT[LP] BJTLP
```



C1,C2,C3,C4: COLLPN
B: BASELPN
E: EMITLPN (Area is 100u)

Example of Lateral Pnp with Possible Multiple Collector

The following is an example for extracting the distributed area & perimeter values of the shared emitter for multiple-collectors BJT device where the emitter area is not directly overlapped to the device layer. The device numbers for BJT[LP] are tagged to *ovemit*, then from *ovemit* to *emit1* and *output* to *temit*.



Description

```
PARSET  BJT1  AREA  PERI  EA  EP  CA  CP

MODEL = BJT[LP], mpnp
SCHEMATIC = LVSLOGIC

....
*END
....
*OPERATION
...
ELEMENT      BJT[LP]  coll  coll  buried  ovemit
...
DEVTAG       BJT[LP]  ovemit  emit1
DEVTAG[LS]   emit1    emit    temit

LEXTRACT BJT1  temit  BY  BJT[LP]  BJTLTP  &
```

Dracula Reference

Operation Block Commands

```
LEXTRACT BJT1 coll BY BJT[LP]
```

```
LPECHK
```

```
LPeselect[S] BJT OUTPUT spice
```

The SPICE netlist will look like this:

```
Q3 count1 in3 vcc mpnp $ CA=400 CP=80 EA=200 EP=40
```

```
Q4 count2 in3 vcc mpnp $ CA=400 CP=80 EA=200 EP=40
```

You can see the difference of EA and EP values between the original example (with DEVTAG[L]) and the new example (with DEVTAG[LS])

The original one's SPICE output looks like this:

```
Q3 count1 in3 vcc mpnp $ CA=400 CP=80 EA=400 EP=80
```

```
Q4 count2 in3 vcc mpnp $ CA=400 CP=80 EA=400 EP=80
```

drcAntenna

Note: *Do not* use the `drcAntenna` information given in the 4.7 version of the *Dracula Reference*. Instead, use the information given below.

Syntax

```
drcAntenna (
    gate( (gateLayer1 polyLayer){(gateLayer2 polyLayer)....})
    antenna( antLayer1 {antLayer2 ...})
    diff( diffLayer1 {diffLayer2 ....})
    parset( parsetName parameter1 {parameter2...})
    check((antLayer1 (calc parFile1 parExpr1){(calc parFile2 parExpr2)...}
          (chkpar option parFile1 refLayer
            rel_op val1 {val2} {outLayer}
            {(output c-name l-num {data-type}}))
          {(chkpar option parFile2 ...) ...}
    )
    {(antLayer2 parFile3 parExpr3)...}
)
```

Description

Simplifies the way you write antenna check rules.

The antenna checking summary is written into a `printfile.sum` file. The `printfile` is defined in the DESCRIPTION block by PRINTFILE.

`drcAntenna` assumes that all metal layers connected to the diffusion layer are considered as safe and that the top metal layer connected to the I/O pads will be ignored due to the protection circuit.

`drcAntenna` supports the cumulated charging methods. The gates not connected to the diffusion layer, but having the same nodal information will be calculated separately because the charging effect might be underestimated if calculated together.

Note: Only one `drcAntenna` command is allowed in a rules deck.

Users do not need to distinguish the type of gates because it makes no difference during antenna checking.

`drcAntenna` supports poly and contact checks in the 4.8 version. See Example 4 and Example 5 in the following pages for details.

Arguments

gate

The initial value for `chkpar`. The keyword introducing the layers to be measured for the gate area of the check. It has the following form:

```
gate ( (gateLayer1 polyLayer) {(gateLayer2 polyLayer)...})
```

where *gateLayer1* is derived from *polyLayer*, and *gateLayer2* is derived from *polyLayer*, and so on. Although it is not required, it is better to specify a single *gateLayer* because if more than one *gateLayer* is specified, the layers are automatically merged into one internal *gateLayer*.

antenna

Keyword introducing the routing layers to be measured for the antenna area of the check. It has the following form:

```
antenna( antLayer1 {antLayer2 ...}),
```

where *antLayer1*, *antLayer2*, and so on are routing layers.

diff

Keyword introducing the diffusion source/drain layers. If any shape in the routing layers is connected to the given diffusion layers, the nets of the shapes are assumed to be connected to diodes and excluded from calculating the antenna area. This argument has the following form:

```
diff(diffLayer1 {diffLayer2 ...})
```

Although it is not required, it is better to use a single *diffLayer* layer, because if more than one *diffLayer* layer are specified, the layers are automatically merged into one internal *diffLayer*.

Note: If you define more than one type of MOS transistors in your rules deck (for example, NMOS and PMOS transistors) and you manually combine all the separate source or drain layers into one single *diffLayer* by ORing them all together, you must also specify this single resulting layer in the CONNECT-LAYER sequence in the INPUT-LAYER block and you must use it to establish connectivity via the `CONNECT` or `SCONNECT` commands in the OPERATION block.

parset

This keyword introducing the parameter set to be used for Dracula to record data. It has the following form:

```
parset( parsetName parm1 {parm2 ...} )
```


Dracula Reference

Operation Block Commands

check

Keyword that defines the antenna check to be performed after a particular layer is processed. It has the following form:

```
check( (antLayer1
      (calc parFile1 parExpr1)
      {(calc parFile2 parExpr2).... })
      (chkpar option parFile1 refLayer rel_op val1
       {val2} {outLayer1}
       (output c-name l-num {data-type})) ... )
      (chkpar option parFile2 refLayer rel_op val1 {val2}
       {outLayer2}
       {(output c-name l-num {data-type})) ...}) )
(antLayer2
  (calc parFile3 parExpr3)
  {(calc parFile4 parExpr4).... }
  (chkpar option parFile3 refLayer rel_op val1 {val2}
   {outLayer3}
   {(output c-name l-num {data-type})) ... }
  (chkpar option parFile4 refLayer rel_op val1 {val2}
   {outLayer4}
   {(output c-name l-num {data-type})) ...}) )
)
```

The `calc` command specifies what kinds of parameter to be derived from the nodes among multiple layers. The `parExpr` in the `calc` command could be expressed in terms of the previous parameters derived from a particular `antLayer`, which means a cumulated model will be used for calculating that parameter.

The `chkpar` command specifies the conditions to be checked on the parameters calculated from the `calc` command. You can supply an optional `output` field to direct the violated shapes from the `refLayer` to a particular error cell. You can specify one of the following options in the `check` command.

Important

Given the syntax at the beginning of this `drcAntenna` command, for every antenna layer that you have specified in the `drcAntenna` syntax, you must also use them in the `check` sub-function. PDRACULA will not check for completeness of the layers specified in the `check` sub-function.

Dracula Reference

Operation Block Commands

For example,

```
drcAntenna(  
  gate( GATE POLY )  
  antenna( M1 M2 M3 M4 )  
  diff( PSD )  
  parset( ANTP AREA )  
  check(  
    ( M4 ( calc PARM4=M4.AREA/GATE.AREA )  
      ( chkpar par PARM4 GATE gt 3 ERR4 output ERR4 1))
```

Although PDRACULA will not report any syntax error when it compiles, your Dracula job will not complete and will get aborted because only the M4 antenna layer was specified in the check sub-function. The M1, M2, and M3 layers were not included.

PAR

Check the computed AREA ratio as stored in the *parFile*.

A

Check the computed AREA value or ratio as stored in the *parFile*.

P

Check the computed perimeter value or ratio as stored in the *parFile*.

The antenna parameters are calculated right after the *antLayer* is calculated. These calculations are followed by a *chkpar* on the *parFile* subject to the specified condition.

Note: You cannot use this command with the *extractParasitic()* command.

Diode Area Checking with drcAntenna

If there is layer name *DIODE* in the following lines, you can calculate diode ratios at any check level:

```
DIFF (DIODE)  
ANTENNA (antLayer1 {antLayer2 ...} DIODE)
```

Note: The layer name *DIODE* is only used name to check diode ratios.

In this case you can write an expression in the rule such as *(calc parFile1 parExpr1)*. For example:

```
(calc parFile2_DIODE parExpr2_DIODE)
```

Where, *parExpr2_DIODE* is an expression dependent on *DIODE* layer parameter (such as *DIODE.AREA* or *DIODE.PERI*), for example:

```
(CALC RD1A = DIODE.AREA)  
(CALC RD1P = DIODE. PERI)
```

Dracula Reference

Operation Block Commands

```
(CALC RM1A = ME1.AREA/GATE.AREA)
```

To check antenna ratios for connected and unconnected to diode layers with ratio of corresponding diode value you must write:

```
(chkpar option parFile1 refLayer rel_op1 val1 with parFile2_DIODE rel_op2 val2  
{(output c-name l-num {data-type})})
```

Where, `rel_op2` can be LT; LE or for `val2=0` `rel_op2` is EQ. For example:

```
(CHKPAR RM1A GATE GT 400 with RD1A LT 0.036 ERMD1A OUTPUT EMD1A 45 )
```

This rule means that the layer `ERMD1A` contains gate layer shapes which are connected to `metal1` with an area ratio greater than 400 and this `metal1` is either connected to diode with an area less than 0.036 or not connected to any diode (diode area equals nil).

The `DIODE` must be the latest layer name for `ANTENNA` and it must be used a single `diffLayer` (`DIODE`) in this case (see the **Arguments** for this command). If there is no layer name of `DIODE`, then `ANTENNA` runs without the diode check using the prior rule syntaxes without the “with” option:

```
ANTENNA (antLayer1 {antLayer2 ...})
```

Examples

For antenna checking, form the correct connectivity.

Attached is a four-metal technology file which contains all the information required for an antenna check. First, you calculate `PARME1`, partial antenna ratio (`PAR`) for `ME1` after the placement of `ME1` layer. At that time, `ME2`, `ME3` and `ME4` are not deposited. Next, you find `PARME2`, `PAR` for `ME2`, and `PARME3`, `PAR` for `ME3`, respectively. Finally, after all layers are deposited, you sum up the antenna ratio from `PARME1`, `PARME2` and `PARME3`, which is the sum of the `PARs` of all nodes connected on top of nodes from the `refLayer`, `GATE`. You use the *default(sum)* mode for choosing the gate area in antenna ratio calculation. You don't need to extract `ME4`, because there is an I/O pad with the protection circuit.

Example 1

```
*DESCRIPTION  
...  
*END
```

```
*INPUT-LAYER  
DIFF = DIFF  
POLY = POLY  
CONT = CONTACT  
ME1 = MET1  
ME2 = MET2  
ME3 = MET3  
ME4 = MET4
```

Dracula Reference

Operation Block Commands

```
VA1  = VI1
VA2  = VI2
VA3  = VI3
;CONNECT LAYERS SEQUENCE
CONNECT-LAYER = SDL POLY ME1 ME2 ME3 ME4
*END

*OPERATION
;LAYERS DERIVATION
;
ANDNOT    DIFF    POLY    GATE    SDL
;
;***** CONNECT STATEMENTS *****
;
CONNECT ME4      ME3      BY      VA3
CONNECT ME3      ME2      BY      VA2
CONNECT ME2      ME1      BY      VA1
CONNECT ME1      POLY     BY      CONT
...
;
;ANTENNA PARAMETERS EXTRACTION
;
drcAntenna(
  gate( GATE POLY )
  antenna( ME1 ME2 ME3 )
  diff( SDL )
  parset( ANTP AREA )
  check(( ME1 (calc PARME1= ME1.AREA / GATE.AREA)
          (chkpar par PARME1 GATE GT 100 errme1 output ERR 1))
        ( ME2 (calc PARME2= ME2.AREA / GATE.AREA)
          (chkpar par PARME2 GATE GT 100 errme2 output ERR 2))
        ( ME3 (calc PARME3= ME3.AREA / GATE.AREA)
          (calc CARME=PARM1+PARM2+PARM3)
          (chkpar par PARME3 GATE GT 100 errme2 output ERR 3)
          (chkpar par CARME  GATE GT 100 errmet output ERR 4))
        )
  )
*END
```

Example 2

This example uses the traditional method to do antenna checking.

```
*DESCRIPTION
...
*END

*INPUT-LAYER
NDIFF = NDIFF
PDIFF = PDIFF
POLY  = POLY
CONT  = CONTACT
ME1   = MET1
ME2   = MET2
ME3   = MET3
VA1   = VIA1
VA2   = VIA2
```

Dracula Reference

Operation Block Commands

```
;CONNECT LAYERS SEQUENCE
CONNECT-LAYER = SDL NSD PSD POLY ME1 ME2 ME3
*END

*OPERATION
;LAYERS DERIVATION
ANDNOT    NDIFF    POLY    NGATE    NSD
ANDNOT    PDIFF    POLY    PGATE    PSD
OR         NSD      PSD      SDL      GATE
OR         PGATE    NGATE    GATE
;
;***** CONNECT STATEMENTS *****
;
CONNECT    ME3      ME2      BY      VA2
CONNECT    ME2      ME1      BY      VA1
CONNECT    ME1      POLY     BY      CONT
CONNECT    ME1      PSD      BY      CONT
CONNECT    ME1      NSD      BY      CONT
CONNECT    ME1      SDL      BY      CONT
...
;
;ANTENNA PARAMETERS EXTRACTION
;
drcAntenna(
    gate( GATE POLY )
    antenna( ME1 ME2 ME3 )
    diff( SDL )
    parset( ANTP AREA )
    check(( ME1 (calc PARME1= ME1.AREA/GATE.AREA)
              (chkpar par PARME1 GATE GT 100 errme1 output ERR 1))
          ( ME2 (calc PARME2=(ME1.AREA+ME2.AREA)/GATE.AREA)
              (chkpar par PARME2 GATE GT 100 errme2 output ERR 2))
          ( ME3 (calc PARME3=(ME1.AREA+ME2.AREA+ME3.AREA)/GATE.AREA)
              (chkpar par PARME3 GATE GT 100 errme2 output ERR 3))
        )
    )
;
*END
```

Example 3

```
*DESCRIPTION
...
*END

*INPUT-LAYER
NPLUS = NPLUS
PPLUS = PPLUS
POLY = POLY
CONT = CONTACT
MT1 = MET1
MT2 = MET2
MT3 = MET3
VA1 = VIA1
VA2 = VIA2
;CONNECT LAYERS SEQUENCE
CONNECT-LAYER = SDL NSD PSD POLY MT1 MT2 MT3
```

Dracula Reference

Operation Block Commands

```
*END

*OPERATION
;LAYERS DERIVATION
ANDNOT    NPLUS    POLY    NGATE    NSD
ANDNOT    PPLUS    POLY    PGATE    PSD
OR         NSD      PSD     SDL
OR         PGATE    NGATE    GATE
;
;***** CONNECT STATEMENTS *****
;
CONNECT    MT3      MT2      BY      VA2
CONNECT    MT2      MT1      BY      VA1
CONNECT    MT1      POLY     BY      CONT
CONNECT    MT1      PSD      BY      CONT
CONNECT    MT1      NSD      BY      CONT
CONNECT    MT1      SDL      BY      CONT
;
;ANTENNA PARAMETERS EXTRACTION
;
drcAntenna(
    gate( GATE POLY )
    antenna( MT1 MT2 MT3 )
    diff( SDL )
    parset( ANTP AREA )
    check(( MT1 (calc PARM1 = MT1.AREA/MIN.GATE.AREA )
              (chkpar par PARM1 GATE GT 100 EX1)
              (chkpar par PARM1 MT1 GT 100 EX2))
          ( MT2 (calc PARM2 = (MT1.AREA+MT2.AREA)/MIN.GATE.AREA)
              (chkpar par PARM2 GATE GT 100 EX3)
              (chkpar par PARM2 MT2 GT 100 EX4))
          ( MT3 (calc PARM3 = (MT1.AREA+MT2.AREA+MT3.AREA)/MIN.GATE.AREA)
              (chkpar par PARM3 GATE GT 100 EX5)
              (chkpar par PARM3 MT3 GT 100 EX6))
        )
    )
;
    OR EX1 EX3 ERR1
    OR ERR1 EX5 ERRGATE

*END
```

Example 4

drcAntenna allows users to do the poly layer antenna check.

```
drcAntenna(
    gate(GATE POLY)
    antenna(POLY MET1 MET2)
    diff(DIFF)
    parset(ANTP AREA PERI)
    check(
        (POLY (calc PARM0 = POLY.AREA/GATE.AREA)
              (chkpar PARM0 RA 0 500 OUTPUT ER0 01))
        (MET1 (calc PARM1 = MET1.AREA/GATE.AREA)
              (chkpar PARM1 RA 0 500 OUTPUT ER1 01))
        (MET2 (calc PARM2 = MET2.AREA/GATE.AREA)
              (calc SUM = PARM0+PARM1+PARM2))
    )
)
```

Dracula Reference

Operation Block Commands

```
(chkpar  PARM2 RA 0 500 OUTPUT ER2 01)
      (chkpar SUM RA 0 500 OUTPUT ER3 01))
))
```

The reflater is not specified in `chkpar`, for example,

`chlpar PARM1 RA 0 500 OUTPUT ER1 01` will be considered as:

```
chlpar PARM1 GATE RA 0 500 OUTPUT ER1 01 &
chlpar PARM1 POLY RA 0 500 OUTPUT ER1 02 &
chlpar PARM1 MET1 RA 0 500 OUTPUT ER1 03 &
```

Example 5

`drcAntenna` allows you to do a contact layer antenna check, but it should not be specified in the antenna layer. Here is an example for it.

```
drcAntenna (
  gate(GATE POLY)
  antenna(MET1 MET2)
  diff(DIFF)
  parset(ANTP AREA PERI)
  check((MET1 (calc  PARM1 = MET1.AREA/GATE.AREA)
      (chkpar  PARM1 RA 0 500 OUTPUT ER1 01)
    (calc  PARM2 = CONT.AREA/GATE.AREA)
      (chkpar  PARM2 RA 0 500 OUTPUT ER2 01))
    (MET2 (calc  PARM3 = MET2.AREA/GATE.AREA)
      (chkpar  PARM3 RA 0 500 OUTPUT ER3 01)
    (calc  PARM4 = VAI.AREA/GATE.AREA)
      (chkpar  PARM4 RA 0 500 OUTPUT ER4 01)
      (calc CARM1 = PARM1+PARM2+PARM3+PARM4)
      (chkpar CARM1 RA 0 500 output ER5 01))
  )
)
```

Example 6

Diode Area Checking with `drcAntenna`

```
*DESCRIPTION
MAG-BEF-GRID=YES
INDISK=antenna_0409_3.gds
PRIMARY=ANTENNA1
OUTDISK=test1.out
PRINTFILE=test1
MODE=EXEC NOW
SYSTEM=GDS2
SCALE=0.001 MICRON
RESOLUTION=0.001 MICRON
LISTERROR=YES
KEEPDATA=YES
FLAG-NON45=YES
FLAG-OFFGRID=YES 0.001
FLAG-ACUTEANGLE=YES
FLAG-SELFINTERS=YES
CNAMES-CSEN=YES
*END
```

Dracula Reference

Operation Block Commands

```
*INPUT-LAYER
SUBSTRATE=PSUB 0
NWEL=3
DIFF=1
PO1=41
NPLUS=12
PPLUS=11
SAB=36
CONT=39
ME1=46
VI1=47
ME2=48
VI2=49
ME3=50
VI3=51
ME4=52
VI4=53
ME5=54
VI5=55
ME6=56
PAD=66
CONNECT-LAYER = DIODE PSD NSD PO1 ME1 ME2 ME3 ME4 ME5 ME6
*END

*OPERATION
SEL DIFF OVERLAP NWEL DIFNW
NOT DIFF DIFNW DIFPS

NOT DIFPS PPLUS NDIF
AND DIFNW PPLUS PDIF

AND PO1 DIFF GATE
NOT PDIF PO1 PSD
NOT NDIF PO1 NSD
OR PSD NSD DIODE

CONNECT ME6 ME5 BY VI5
CONNECT ME5 ME4 BY VI4
CONNECT ME4 ME3 BY VI3
CONNECT ME3 ME2 BY VI2
CONNECT ME2 ME1 BY VI1
CONNECT ME1 PO1 BY CONT
CONNECT ME1 PSD BY CONT
CONNECT ME1 NSD BY CONT
CONNECT ME1 DIODE BY CONT

drcAntenna(
GATE( GATE PO1 )
DIFF( DIODE )
ANTENNA( PO1 ME1 ME2 ME3 ME4 ME5 ME6 DIODE )
PARSET( ANT0 AREA PERI )
CHECK(
( PO1 ( CALC RP1A = PO1.AREA/GATE.AREA )
( CALC RDP1A = DIODE.AREA )
( CHKPAR RP1A GATE GT 100 with RDP1A LT 0.005 OUTPUT EP1A 45 ))
( ME1 ( CALC RM1A = ME1.AREA/GATE.AREA )
( CALC RM1CA = RP1A+RM1A )
( CALC RDM1A = DIODE.AREA )
( CHKPAR RM1A GATE GT 400 with RDM1A LT 0.005 ERM1A OUTPUT EM1A 45 )
( CHKPAR RM1CA GATE GT 400 with RDM1A LT 0.005 ERM1CA OUTPUT EM1CA 45 ))
( ME2 ( CALC RM2A = ME2.AREA/GATE.AREA )
( CALC RM2CA = RP1A+RM1A+RM2A )
```


Dracula Reference

Operation Block Commands

```
( CALC RDM2A = DIODE.AREA )
  ( CHKPAR RM2A GATE GT 400 with RDM2A LT 0.005 ERM2A OUTPUT EM2A 45 )
  ( CHKPAR RM2CA GATE GT 400 with RDM2A LT 0.005 ERM2CA OUTPUT EM2CA 45 ))
( ME3 ( CALC RM3A = ME3.AREA/GATE.AREA )
  ( CALC RM3CA = RP1A+RM1A+RM2A+RM3A )
  ( CALC RDM3A = DIODE.AREA )
    ( CHKPAR RM3A GATE GT 400 with RDM3A LT 0.005 ERM3A OUTPUT EM3A 45 )
    ( CHKPAR RM3CA GATE GT 400 with RDM3A LT 0.000 ERM3CA OUTPUT EM3CA 45 ))
( ME4 ( CALC RM4A = ME4.AREA/GATE.AREA )
  ( CALC RM4CA = RP1A+RM1A+RM2A+RM3A+RM4A )
  ( CALC RDM4A = DIODE.AREA )
    ( CHKPAR RM4A GATE GT 400 with RDM4A LT 0.005 ERM4A OUTPUT EM4A 45 )
    ( CHKPAR RM4CA GATE GT 400 with RDM4A LT 0.005 ERM4CA OUTPUT EM4CA 45 ))
( ME5 ( CALC RM5A = ME5.AREA/GATE.AREA )
  ( CALC RM5CA = RP1A+RM1A+RM2A+RM3A+RM4A+RM5A )
  ( CALC RDM5A = DIODE.AREA )
    ( CHKPAR RM5A GATE GT 400 with RDM5A LT 0.005 ERM5A OUTPUT EM5A 45 )
    ( CHKPAR RM5CA GATE GT 400 with RDM5A LT 0.005 ERM5CA OUTPUT EM5CA 45 ))
( ME6 ( CALC RM6A = ME6.AREA/GATE.AREA )
  ( CALC RM6CA = RP1A+RM1A+RM2A+RM3A+RM4A+RM5A+RM6A )
  ( CALC RDM6A = DIODE.AREA )
    ( CHKPAR RM6A GATE GT 400 with RDM6A LT 0.005 ERM6A OUTPUT EM6A 45 )
    ( CHKPAR RM6CA GATE GT 400 with RDM6A LT 0.005 ERM6CA OUTPUT EM6CA 45 ))
))
*END
```

ECONNECT

```
ECONNECT elem{[type]} layer CONN/DISC label {trapfile} OUTPUT c-name l-  
      num {d-num} {&}
```

Description

Outputs elements that are connected (or disconnected) from a labeled node through a specified layer.

Arguments

elem

Type of element being checked. You must define this element using the ELEMENT command.

MOS = MOS devices

PAD = PAD windows

LDD = LDD devices

type

Two-letter (A-Z) code that denotes the type of MOS device. Used to differentiate N-channel from P-channel in CMOS or to differentiate MOS devices with different implants.

layer

Layer through which connections are being checked. A layer name of *ALL* checks all *elem* and *type* conductor layers defined in the ELEMENT command.

CONN/DISC

CONN means a connection to the labeled node is a violation.

DISC means a disconnection from the labeled node is a violation.

label

Text label of the node considered for the check. The label must have text assigned in the layout database and conform to the text rules.

trapfile

Name of the output file to contain composite trapezoids. Before you perform any subsequent operation on a *trapfile*, you need to “OR” it with itself or with the substrate. This merges the multiple layers.

Dracula Reference

Operation Block Commands

OUTPUT

Sends the results of the operation to an output cell.

Note: The output cell you specify cannot be used as the output of another command.

c-name l-num

C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

The datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

&

The ampersand indicates a conjunctive rule. This command can be joined to other ECONNECT and NDCOUNT commands.

Note: ECONNECT outputs the device layer of the element.

Elements that satisfy the ECONNECT command are collected in an output cell. This data in the output cell is the device layer of the element. For silicon-gate MOS, this is the channel region layer.

Examples

The following example checks all MOS P-type elements connected to VSS (the labeled node) through the predefined *srcdrn* layer. The elements that satisfy the condition are output as cell name *cell110* on layer 10.

```
ECONNECT MOS[P] srcdrn CONN vss OUTPUT cell11 10
```

The following example collects all MOS elements connected to VCC through *srcdrn*. The result is passed on to a second check. The elements that satisfy this condition are connected to VSS through *srcdrn*. Thus, output *cell211* on layer 11 has all transistors connected to power and ground simultaneously through the source/drain layers.

Note: This is an electrical rule violation in an MOS circuit.

```
ECONNECT MOS srcdrn CONN vcc &  
ECONNECT MOS srcdrn CONN vss OUTPUT cell2 11
```

The following example collects all BJT[N] elements connected to VCC through the *coll* (collector). The result is passed on to a second check. The elements that satisfy this condition are connected to GND through *emit* (emitter). Thus, the output cell *dever115* on layer 15 has all transistors connected to power and ground simultaneously.

Dracula Reference

Operation Block Commands

```
ECONNECT BJT[N] coll CONN vcc &  
ECONNECT BJT[N] emit CONN gnd OUTPUT devert 15
```

EDGECHK

```
EDGECHK layer {ANGLE[90/-90]} LENGTH GT value1 {trapfile} {OUTPUT ERR layerno  
{datatype}}
```

Description

This command checks the length of 45 non-manhattan edge in GATE.

Arguments

ANGLE[90]

Manhattan edge (parallel with X-Y axis)

ANGLE[-90]

Non-manhattan edge

LENGTH

Allowed operations are LT,RA,GT

trapfile

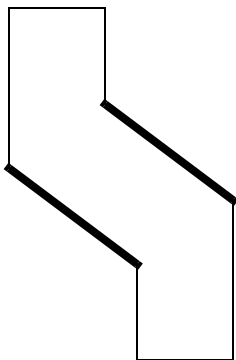
Output as a trapfile

[O]

By default, the output edge is inside the shape. If you use [O], the edge is outside the shape.

Example

```
EDGECHK gate ANGLE[-90] LENGTH GT 1.0 OUTPUT ERR 01
```



EDTEXT

`EDTEXT = text-coordinate-file`

Description

Specifies to use text from a text coordinate file in addition to text in the design. The text coordinate file is a free-format, 80-column text file. Each line specifies one text label, its x,y coordinates, and, optionally, its attached layer name. The x,y coordinates contain significant digits equivalent to the resolution.

Significant digits is the number of digits behind the decimal point. For example, if you set the resolution to .01, the x,y coordinate should have two digits after the decimal point (such as, .33). If the resolution is .001, the x,y coordinate should have three digits after the decimal point (for example, .333).

Text labels not located at two significant digits are rounded up and possibly discarded if they conflict with other text labels at the same coordinate.

The text coordinate file can contain spaces, empty lines, tab characters, or comments (;). The comment must be on its own line. You can put the comment starter (;) in the first column of a line, or you can precede it with blank spaces. You cannot put a comment at the end of a line of text in an EDTEXT file. Dracula ignores all text after the comment starter. The following is an example of a valid file:

```
ABC:I   X=    10   Y=  -200
VCC     X=   120   Y=    40   Attach=Metal
VSS     X=  -120   Y=   -40
IN      X=   -10   Y=   -10
OUT     X=    0.5   Y=    40
```

If you specify a layer name for attaching the text, the text is attached to that layer only. If you do not specify a layer name, the text is attached to one of the layers according to the TEXTSEQUENCE or CONNECT-LAYERS commands.

When both the CAD graphics system database and the text coordinate file have different texts at the same location, the CAD system text is discarded. You can override or delete the graphics system text. To delete the graphics system text, specify an illegal text (for example, 99) in the *EDTEXT* file with the same coordinates as the graphics system text.

Place the EDTEXT command above the CONNECT command in the Operation block.

Example

```
*OPERATION
.
```

Dracula Reference

Operation Block Commands

```
EDTEXT = ram.txt  
.  
.  
CONNECT metal poly BY cont  
.  
.  
*END
```

ELCOUNT

```
ELCOUNT elem {[type]} layer condition n1 {trapfile} OUTPUT c-name l-num  
          {d-num}    {&}
```

Description

Counts the number of elements connected through specific layers in a node. The command outputs the nodes with specified element counts to an output cell.

Note: The output cell you specify cannot be used as the output of another command.

Arguments

elem

Type of element being checked. You must define this element using the ELEMENT command.

MOS = MOS devices
BJT = Bipolar devices
PAD = PAD windows
LDD = LDD devices

type

Two-letter (A-Z) code that denotes the type of device. This can differentiate N-channel from P-channel in CMOS or differentiate MOS devices with different implants.

layer

Layer through which the connections are being checked. A layer name of *ALL* checks all *elem* and *type* conductor layers defined in the ELEMENT command.

condition n1

The tests according to one of the following conditions:

where the number (n1) can be an integer between 0 and 63.

LT <i>n1</i>	For an element count less than (n1)
LE <i>n1</i>	For an element count less than or equal to (n1)
EQ <i>n1</i>	For an element count equal to (n1)
NE <i>n1</i>	For an element count not equal to (n1)

Dracula Reference

Operation Block Commands

GT *n1* For an element count greater than (*n1*)

GE *n1* For an element count greater than or equal to (*n1*)

trapfile

Name of the output file to contain composite trapezoids. Before you perform any subsequent operation on a *trapfile*, you need to “OR” it with itself or with the substrate. This merges the multiple layers.

Note: Dracula issues a warning if you specify the same layer for both the input and output layer.

OUTPUT

Sends the results of the operation to an output cell. The output cell you specify cannot be used as the output of another command.

c-name l-num

C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

&

Indicates a conjunctive rule. This command can be joined to other ELCOUNT and PROBE commands.

Note: When you use ELEMENT PAD, nodes with an ELEMENT PAD attached do not enter into ELCOUNT checks. This also includes all power nodes. ELCOUNT outputs the entire node that is collected.

Examples

This example detects all floating nodes that have no connection with the element through the *srcdrn* layer. The output cell name is *float60*.

```
ELCOUNT MOS srcdrn EQ 0 OUTPUT float 60
```

This example detects nodes that are connected to only one MOS element. The output cell name is *single60*.

```
ELCOUNT MOS all EQ 1 OUTPUT single 60
```

Dracula Reference

Operation Block Commands

In the following example, the first command collects nodes that are connected to more than 60 elements through the *srcdrn* layer. The second command checks the nodes further to see if they are connected to predefined pads. The output cell name is *bignod40*.

```
ELCOUNT MOS srcdrn GT 60 &  
ELCOUNT PAD metal EQ 0 OUTPUT bignod 40
```

The following example detects nodes that are connected to only one bipolar element. The output cell name is *onedev20*.

```
ELCOUNT BJT all EQ 1 OUTPUT onedev 20
```

ELEMENT BJT

```
ELEMENT BJT{[type]} layer-a layer-b layer-c layer-d {layer-s}
```

Description

Defines bipolar devices.

Arguments

type

Two-character code that denotes the type of BJT device. The first character must be any letter from A-Z. The second character is optional and can be any letter from A-Z or any number from 0-9 (except for 8). You can use this code to differentiate BJT devices. For example, vertical *npn* devices are type *NV* and lateral *pnp* devices are type *PL*.

layer-a

Device layer, one region per device. This layer defines the device region that touches or overlaps the bipolar conductor layers. For an *npn* device, this is usually defined by the emitter. For a lateral *pnp* device, this is usually defined by the collector.

layer-b

Terminal conductor layer for the collector terminal. This layer must carry node information.

layer-c

Terminal conductor layer for the base terminal. This layer must carry node information.

layer-d

Terminal conductor layer for the emitter terminal. This layer must carry node information.

layer-s

Substrate terminal layer. If you define this substrate layer, LVS checks for discrepancies or inconsistencies found in the layout or schematic. Discrepancies are reported in the LVS report file. You must also add a CDL definition for substrate.

Note:

- ELEMENT BJT should not have terminal layer names with the 3 first letters BJT.
- In general, the device layer used for a vertical *npn* device must be the emitter region because an *npn* can have multiple emitters. Also, the emitter region overlaps all three terminals of this device. For the lateral *pnp*, the device layer must be the collector region

because a *pnp* can have multiple collectors. The collector layer does not touch or overlap the emitter layer for its terminal connection. To overcome this, the emitter layer is oversized to overlap the collector device layer and then stamped with node information.

 Important

Once a layer has been defined as a device or terminal layer, it must not be changed by any operation (mentioned as an output layer) after the ELEMENT command.

Examples

In the following example, the *enpn* is the emitter region and device layer. Layers *buried*, *base*, and *nplus* are the BJT terminal conductors and they must carry node information.

```
ELEMENT BJT[NV] enpn buried base nplus
```

In this example, *cpnp1* is the collector region and device layer.

```
ELEMENT BJT[PL] cpnp1 base buried oepnp1
```

This example shows the ELEMENT command with a substrate layer defined. The name specified here is *WELL*, but you can specify any name.

```
ELEMENT BJT[A] BJTW COLL BASE EMIT WELL
```

For additional discussion and illustrations of bipolar devices, refer to [Chapter 7, “Extracting Electrical Parameters \(LPE\).”](#)

ELEMENT BOX

`ELEMENT device layer-a layer-b {layer-c} {layer-d} {layer-e}`

Description

Defines devices with a variable number of terminals. This command lets you define special devices (that is, element types other than BJT, CAP, DIO, LDD, MOS, PAD, or RES).

Arguments

device

Three-character name that denotes a user-defined device. The name can have only characters A-Z. However, the first character cannot be B, C, D, L, M, P, or R. This name must match the name of an empty `.SUBCKT` in the netlist corresponding to the ELEMENT BOX.

layer-a

Device layer, one region per device, that touches or overlaps the device conductor layers.

layer-b

Position No. 1 terminal conductor layer with zero to multiple nodes. This is the conductor layer for the position No. 1 terminal and must carry node information.

layer-c

Position No. 2 terminal conductor layer with zero to multiple nodes. This is the conductor layer for the position No. 2 terminal and must carry node information.

layer-d

Position No. 3 terminal conductor layer with zero to multiple nodes. This is the conductor layer for the position No. 3 terminal and must carry node information.

layer-e

Position No. 4 terminal conductor layer with zero to multiple nodes. This is the conductor layer for the position No. 4 terminal and must carry node information.

Note:

- ELEMENT BOX should not have terminal layer names with the 3 first letters BOX.

 Important

Once a layer has been defined as a device or terminal layer, it must not be changed by any operation (mentioned as an output layer) after the ELEMENT command.

Example

This example is for a p+ diffusion resistor element with a buried layer underneath the resistor.

```
ELEMENT ISP pdifres cont buried
```

ELEMENT CAP

```
ELEMENT CAP {[type]} layer-a layer-b layer-c {layer-s}
```

Description

Defines a capacitor device. This command does not define an LPE parasitic element. See also the PARAMETER CAP section in this chapter.

Arguments

type

Two-character code that denotes the type of CAP device. The first character must be a letter from A-Z. A second character is optional and can be any letter from A-Z or a number from 0-9 (except 8). This code differentiates capacitor devices.

layer-a

Device layer, one region per device, that defines the capacitor region that touches or overlaps the capacitor conductor layers.

layer-b

Terminal conductor layer for one side of the capacitor. This layer must carry node information.

layer-c

Terminal conductor layer for the other side of the capacitor. This layer must carry node information.

layer-s

Substrate terminal layer. If you define this substrate layer, LVS checks for discrepancies or inconsistencies found in the layout or schematic. Discrepancies are reported in the LVS report file. You must also add a CDL definition for substrate.

Note:

- ELEMENT CAP should not have terminal layer names with the 3 first letters CAP.

Important

Once a layer has been defined as a device or terminal layer, it must not be changed by any operation (mentioned as an output layer) after the ELEMENT command.

Example 1

```
*INPUT-LAYER
.
.
CONNECT-LAYER=pwell nsd psd pl1 pl2 metal
*END
*OPERATION
AND pl1 pl2 pl2cap; poly1 to poly 2 capacitor
CONNECT pl1 pl2 BY cont2
CONNECT metal pl2 BY cont1
CONNECT metal pl1 BY cont1
.
.
ELEMENT CAP[PP] pl2cap pl2 pl1
ELEMENT CAP[A] metpwe1 metal pwell
```

You can check ELEMENT CAP polarity in LVS or LPE when you specify the *P* option in LVSCCHK or LPECHK. When you specify the *P* option, you cannot swap *layer-b* and *layer-c*.

You can also check the ELEMENT CAP type in LVS or LPE when you specify LVSCCHK or LPECHK. The schematic CDL netlist must specify the type for the capacitors to check. For more information, refer to Chapter 5, "Compiling Network Descriptions (LOGLVS)."

Example 2

```
*OPERATION
.
.
ELEMENT CAP[PP] pl2cap pl2 pl1
ELEMENT CAP[A] metpwe1 metal pwell
.
.
LVSCCHK[P]
*END
```

CDL netlist:

```
C1 sig1 sig2 3pf $ [PP] or $.MODEL=PP
C2 sig3 sig4 1 pf $ [A] or $.MODEL=A
```

Note: LVS checks the capacitor type because the CDL netlist has optional subtypes specified with brackets [] or with the .MODEL command. It also checks the capacitor polarity when you specify the *P* option in LVSCCHK. This *P* option does not allow swapping between ELEMENT CAP terminals and the CDL capacitor terminals (that is, pl2 ↔ sig1, pl1 ↔ sig2, metal ↔ sig3, and pwell ↔ sig4). For more information about capacitor polarity checking, refer to Chapter 5, "Compiling Network Descriptions (LOGLVS)."

ELEMENT DIO

```
ELEMENT DIO {[type]} layer-a layer-b layer-c {layer-s}
```

Description

Defines the P-N junction diode. This command does not define an LPE element.

Arguments

type

Two-character code that denotes the type of DIO device. The first character must be a letter from A-Z. The second character is optional and can be any letter from A-Z or any number from 0-9 (except 8). This code differentiates diode devices. For example, P-diode to N-substrate devices are [P] type, and N-diode to P-well devices are [N] type.

layer-a

Device layer, one region per device. This layer defines the diode region that touches or overlaps the anode and cathode conductor layers.

layer-b

Anode conductor layer for the anode terminal, usually the p+ diffusion or the P-well. This layer must carry node information.

layer-c

Cathode conductor layer for the cathode terminal, usually the n+ diffusion or the N-substrate. This layer must carry node information.

layer-s

Substrate terminal layer. If you define this substrate layer, LVS checks for discrepancies or inconsistencies found in the layout or schematic. It reports discrepancies in the LVS report file. You must also add a CDL definition for substrate.

Note:

- ELEMENT DIO should not have terminal layer names with the 3 first letters DIO.

Important

Once a layer has been defined as a device or terminal layer, it must not be changed by any operation (mentioned as an output layer) after the ELEMENT command.

Dracula Reference

Operation Block Commands

Example

```
*INPUT-LAYER
.
CONNECT-LAYER=nsup pwell nsd psd poly metal
*END
*OPERATION
.
AND psd diode pdio; identify the P+ pdio
CONNECT metal nsd BY cont
CONNECT metal psd BY cont
CONNECT nsd nsup BY ntsub
.
ELEMENT DIO[P] pdio psd nsup
;For a p+ diode element to the n-substrate.
```

Note: There is no `PARAMETER DIO` command. To invoke an LVS diode size check, you must use the `ELEMENT DIO` command and the `LVSCCHK` or `LPECHK` command with *DIOAREA* and *DIOPERI* options. For information about these options, refer to the `LVSCCHK` or `LPECHK` command and Chapter 5, "Compiling Network Descriptions (LOGLVS)."

ELEMENT IN?

`ELEMENT device layer-a layer-b layer-c {layer-s}`

The above is the ELEMENT IN? syntax in the “Operation Block of Commands” of your rule file when the PARSET command’s syntax in the “Description Block of Commands” of your rule file is as follows:

`PARSET =   pname param1 param2 param3,...paramn`

The ? in the ELEMENT IN? command stands for any alphabet, but the first 2 characters of the device name definitely need to start with IN.

Description

Allows you to use BOX devices to simulate inductors and extract some important parameters for the inductors, such as the number of turns, diameter of inner circle, etc. and output them to the SPICE netlist file for supporting inductors in layout designs.

The first step is to use PARSET command to specify what parameters will be extracted. There are four geometric primitives regarding inductors provided: ID, IN, IW, and IS and one reserved parameter, LL. See the PARSET command in “Description Block Commands” chapter of this manual for more information on these four primitives and reserved parameter. See the “Supporting Inductors in Layout Designs Using the PARSET and ELEMENT IN? Commands” section in the “Extracting Electrical Parameters (LPE)” chapter of this manual for information on the different types of inductors.

Important

Once a layer has been defined as a device or terminal layer, it must not be changed by any operation (mentioned as an output layer) after the ELEMENT command.

Arguments

device

User-defined device name with three characters. The first two characters should be IN.

layer-a

Device layer, one region per device.

layer-b

Terminal layer for one side of the inductor.

Dracula Reference

Operation Block Commands

layer-c

Terminal layer for the other side of the inductor.

layer-s

Substrate terminal layer.

Example

Description Block:

```
PARSET = TEST ID IN IW IS LL
```

Operation Block:

```
ELEMENT IND INDSYM ME4 ME5  
LEXTRACT TEST INDSYM BY IND INDPAR1 &  
EQUATION LL=ID+IN+IW+IS  
LPESELECT[NY] IND output SPICE
```

ELEMENT LDD

```
ELEMENT LDD {[type]} layer-a layer-b layer-c layer-d  
            {layer-e}
```

Description

Defines a MOS element whose source and drain cannot be swapped.

Arguments

type

Two-character code used to denote the type of LDD (lightly doped drain MOS) device. The first character must be a letter from A-Z. The second is optional and can be any letter from A-Z or any number from 0-9 (except 8). This code differentiates MOS devices with different implants. For example, CMOS pull-up devices are *P* type and pull-down devices are *N* type. NMOS devices are *E*, *D*, and *N* types. See the following note about the special LDD[X-] type.

layer-a

Device layer, one region per device. This layer defines the device region that touches or overlaps the device conductor layers. For silicon-gate MOS, this is the channel region layer usually defined by the overlapped area of polysilicon and diffusion. For a gallium arsenic transistor, this is the channel region layer usually defined by the overlapped area of schottky metal and lightly doped diffusion. For metal-gate MOS, this is the gate region usually defined by the thin oxide mask.

layer-b

Gate conductor layer. For silicon-gate MOS, this is the polysilicon layer. For metal-gate MOS, this is the metal layer; for gallium arsenic, this is the *schottky* metal layer. This layer must carry node information.

layer-c

Lightly doped drain conductor layer. For silicon-gate MOS, this is the self-aligned lightly doped diffusion layer, usually defined by the lightly doped diffusion layer excluding the channel regions. For metal-gate MOS, this is the lightly doped diffusion layer. This layer must carry node information.

layer-d

Source conductor layer. For silicon-gate MOS, this is the self-aligned diffusion layer, usually defined by the diffusion layer excluding the channel regions. For metal-gate MOS, this is the diffusion layer. This layer must carry node information.

Dracula Reference

Operation Block Commands

layer-e

Substrate conductor layer. This substrate layer is for optional use in running LVS or LPE checking substrate connection mismatch. For MOS: usually N-well, P-well, N-substrate, or P-substrate. If you specify this layer, it must carry node information.

Note:

- ELEMENT LDD should not have terminal layer names with the 3 first letters LDD.
- A special LDD[X-] type stops the generation of pseudo gates when you run an LVS check. To disallow devices from forming pseudo gates, specify an ELEMENT LDD command where the first character type is an X and the second character type is either a letter from A-Z or a number from 0-9 (except 8). This allows LVS to locally stop the formation of pseudo gates in the layout and allows the LVS checking of mixed analog and digital circuits.

Important

Once a layer has been defined as a device or terminal layer, it must not be changed by any operation (mentioned as an output layer) after the ELEMENT command.

The LVSCCHK command governs the reduction of devices to inverters and other LVS options the same way as for ELEMENT MOS.

To perform LVS, you must define LDD devices on the schematic netlist. There is, however, no special element called LDD in SPICE or other standard simulation packages. Using Circuit Description Language (CDL), you can code LDD devices as MOS devices with the designator LDD[*type*] specified as a comment:

```
Mxxxxxx nd ng ns {nb} mname {L=val1} {W=val2} $LDD[type]
```

Example

```
ELEMENT LDD[N] GNDX POLY1 NDDIF1 NDIF1 PWELL1
```

ELEMENT MOS

```
ELEMENT MOS {[type]} layer-a layer-b layer-c {layer-d}
```

Description

Defines metal or silicon-gate MOS devices.

Arguments

type

Two-character code to denote the type of MOS device. The first character must be a letter from A-Z. The second is optional and can be any letter from A-Z or any number from 0-9 (except 8). This code differentiates MOS devices with different implants. For example, CMOS pull-up devices are [P] type and pull-down devices are [N] type. NMOS devices are [E], [D], and [N] types. See the note below on the special ELEMENT MOS[X-] type.

layer-a

Device layer, one region per device. For silicon-gate MOS, this is the channel region layer usually defined by the overlapped area of polysilicon and diffusion. For a gallium arsenic transistor, this is the channel region layer usually defined by the overlapped area of schottky metal and lightly doped diffusion. For metal-gate MOS, this is the gate region usually defined by the thin oxide mask.

layer-b

Gate conductor layer. For silicon-gate MOS, this is the polysilicon layer. For metal-gate MOS, this is the metal layer. For gallium arsenic, this is the schottky metal layer. This layer must carry node information.

layer-c

Source/drain conductor layer. For silicon-gate MOS, this is the self-aligned diffusion layer, usually defined by the diffusion layer excluding the channel regions. For metal-gate MOS, this is the diffusion layer. This layer must carry node information.

layer-d

Substrate conductor layer. This substrate layer is optional when you run LVS or LPE checking substrate connection mismatch. For MOS: usually N-well, P-well, N-substrate, or P-substrate. If you specify this layer, it must carry node information.

Note:

- ELEMENT MOS should not have terminal layer names with the 3 first letters MOS.
- A special MOS[X-] type stops the generation of pseudo gates when you run an LVS check. To disallow devices from forming pseudo gates, specify an ELEMENT MOS command where the first character type is an X and the second character type is either a letter from A-Z or a number from 0-9 (except 8). This allows LVS to locally stop the formation of pseudo gates in the layout and allows the LVS checking of mixed analog and digital circuits.

 Important

Once a layer has been defined as a device or terminal layer, it must not be changed by any operation (mentioned as an output layer) after the ELEMENT command.

Examples

In this example, the MOS[X] transistor forms pseudo gates for LVS. The MOS[XN] transistor is not allowed to form pseudo gates and remain as a transistor in an LVS run.

```
ELEMENT MOS[N]      ngate      poly      nsd      pwell
ELEMENT MOS[D]      depchnl    poly      srcdrn
ELEMENT MOS[N]      nchnl      metal     diff
ELEMENT MOS         chnl       poly      srcdrn
ELEMENT MOS[P]      oxide      metal     diff      nsub
ELEMENT MOS[X]      xgate      poly      nsd      pwell
ELEMENT MOS[XN]     xngate     poly      nsd      pwell
```


ELEMENT PAD

`ELEMENT PAD layer-a layer-b`

Description

Defines the pads of a chip, eliminating false ERC errors related to the pads when you use the ELCOUNT and PATHCHK commands.

Arguments

layer-a

Layer defining the protective oxide opening over the pad area.

layer-b

Conductor layer from pad to circuit. This layer must carry node information.

Note:

- ELEMENT PAD should not have terminal layer names with the 3 first letters PAD.

Important

Once a layer has been defined as a device or terminal layer, it must not be changed by any operation (mentioned as an output layer) after the ELEMENT command.

Examples

```
ELEMENT PAD    vapox    metal
ELEMENT PAD    padwin    metal
ELEMENT PAD    vapox     metal2
```

ELEMENT RES

```
ELEMENT RES {[type]} layer-a layer-b {layer-s}
```

Description

Defines a resistor device (poly, diffusion, thin-film, and so forth). This command does not define an LPE element.

You can check ELEMENT RES type in LVS or LPE when you specify LVSCCHK or LPECHK. The schematic CDL netlist must specify the type for the resistors to check. Refer to Chapter 5, "Compiling Network Descriptions (LOGLVS)."

Arguments

type

Two-character code that denotes the type of RES device. The first character must be A-Z. The second is optional and can be A-Z or 0-9 (excluding the number 8). This code differentiates among resistor devices. For example, P-diffusion devices are [PD] type and poly devices are [PO] type.

layer-a

Device layer, one region per device. This layer defines the resistor region that touches or overlaps the resistor conductor layers.

layer-b

Terminal conductor layer usually defined by a conductor layer excluding the resistor region (*layer-a*). This layer must carry node information.

layer-s

Substrate terminal layer. If you define this substrate layer, LVS checks for discrepancies or inconsistencies found in the layout or schematic. LVS reports discrepancies in the LVS report file. You must also add a CDL definition for substrate.

Note:

- ELEMENT RES should not have terminal layer names with the 3 first letters RES.

Important

Once a layer has been defined as a device or terminal layer, it must not be changed by any operation (mentioned as an output layer) after the ELEMENT command.

Example 1

For a poly resistor element:

```
AND ipoly res pores; digitized resistor mask
NOT ipoly pores poly
ELEMENT RES[PO] pores poly
```

Example 2

For a p+ diffusion resistor element:

```
SELECT pdiff LABEL[R] R? pdifres
STAMP cont BY metal
ELEMENT RES[PD] pdifres cont
```

Example 3

```
ELEMENT RES[PO] pores poly
ELEMENT RES[PD] pdifres cont
.
.
LVSCHK
*END
```

CDL netlist:

```
R1 sig1 sig2 3k $[PO] or $.MODEL=PO
R2 sig3 sig4 1k $[PD] or $.MODEL=PD
```

LVS checks the resistor subtype because the CDL netlist specifies the option subtypes either with the brackets [] or with the .MODEL command.

ENC

```
ENC {[ ]} layer-a {&layer-a1}[ ] layer-b {&layer-b1} LT/LE/EQ/RANGE/SELLT/  
SELLE/SELGT/SELGE/SELEQ/SELNE/SELRA n1 {n2} {CORNER-CORNER n2} {CORNER-EDGE  
n3} {trapfile} {OUTPUT c-name l-num {d-num}} {&}
```

Description

Determines the degree that *layer-b* polygons partially or fully enclose *layer-a* polygons. This command measures between the outer edges of the polygons being enclosed and the inner edges of the enclosing polygons.

Note: The ENC command will not flag violations between obtuse angled edges.

Shell environment variables

■ `setenv SPAC_LISTERR_DIGIT [ 2 | 3 ]`

Use this environment variable to set number of digits after the decimal point for EXT/ENC/WIDTH/INT log report.

Checking Method

modes

Flat, cell, hierarchical, and composite.

checks

HDRC.

Arguments

layer-a

First input layer name containing the enclosed polygons.

layer-b

Second input layer name containing the enclosing polygons.

&*layer-a1*

Renames (conjunctive rename) the layer containing the error flags. The original layer name is then restored to the original layer. Do not put a space between the *layer-a* and &*layer-a1* names; for example, *poly&rpoly*.

Dracula Reference

Operation Block Commands

&layer-b1

Same as *&layer-a1* except that it refers to *layer-b*.

measurement operation

Measures between the outside edge segments of the enclosed polygons (in *layer-a*) and the inside edge segments of the enclosing polygons (in *layer-b*). Compares the measurements against the limits you specify and flags violations.

You must supply only one of these measurements:

LT/LE <i>n1</i>	Flags an error if the two segments are spaced less than <i>n1</i> for <i>LT</i> or less than or equal to <i>n1</i> for <i>LE</i> . Does not flag an error if segments touch or overlap.
EQ <i>n1</i>	Flags an error if the two segments are spaced equal to <i>n1</i> .
RANGE <i>n1 n2</i>	Flags an error if the two segments are spaced less than <i>n2</i> and greater than <i>n1</i> . If you do not want acute angles to be flagged, specify the <i>RANGE</i> measurement.
SELGT/SELGE <i>n1</i>	<i>SELGT</i> outputs in trapezoid format all the <i>layer-a</i> polygons that the program does not flag when you specify an <i>LE</i> measurement with the same options. <i>SELGE</i> has the same function as <i>SELGT</i> except that it works opposite from an <i>LT</i> operation. Do not use the OUTPUT statement with this option. Specify a <i>trapfile</i> name instead. Use this check in flat and cell modes only.
SELLT/SELLE <i>n1</i>	<i>SELLT</i> and <i>SELLE</i> output in trapezoid format the same <i>layer-a</i> polygons associated with the errors that Dracula flags when you specify an <i>LT n1</i> or an <i>LE n1</i> and use the same options. Specify a <i>trapfile</i> name instead of using the OUTPUT statement. Use this check in flat and cell modes only.

Dracula Reference

Operation Block Commands

SELEQ/SELNE
n1 *SELEQ* and *SELNE* output in trapezoid format the same *layer-a* polygons associated with the errors that Dracula flags when you specify an *EQ n1* or an *NE n1* and use the same options. Specify a *trapfile* name instead of using the OUTPUT statement. Use this check in flat and cell modes only.

SELRA n1 n2 *SELRA* outputs in trapezoid format the same *layer-a* polygons associated with the errors that Dracula flags when you specify a *RANGE n1 n2* measurement with the same options. Specify a *trapfile* name instead of using the OUTPUT statement.

trapfile

The trapezoid file name. Use this option only with the *R* or *R'* option or with SELLT, SELLE, SELGT, SELGE, or SELRA operations.

OUTPUT

Sends the results to an output cell.

c-name l-num

C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system.

If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

The datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

&

The ampersand sign indicates a conjunctive rule.

Note: The output cell contains clean polygons in polygon format, resulting in reduced trapezoid construction lines and a more compact cell. The *trapfile* option and OUTPUT command are mutually exclusive.

Command Options

Options that are applicable to hierarchical Dracula are indicated with an  icon in the margin. The plus (+) in an option adds error flags. The minus (-) in an option removes error flags.

B

Flags the actual part of a violation on a segment instead of the entire segment. This option can be applied only with LT or LE modifiers. Neither RA, GT, nor GE modifiers can contain the B option.

Note: Causes round-off side effects on non-45-degree edges.

C

(-) Flags the edge-pair when the edges are parallel.

C'

(-) Inverse of C option. Flags non-parallel edges.

E

(+) Flags polygons from *layer-a* that are totally outside polygons from *layer-b*. Performs an enclosure test in parallel with other functions performed by the ENC command.

F

Outputs error flags in trapezoid format that execute information for edge-sizing SIZE[E] operations.

F'

Outputs error flags in trapezoid format. You must specify only *trapfile* name. It is similar to R option but does not turn on the P option.

N



(-) Does not flag violations on the enclosed and the enclosing polygons if they are part of the same node. Use the CONNECT command to connect nodes before using this option. This option checks nodes on the composite plane and cell plane.

N'



(-) Inverse of N option. Flags violations on the enclosed and the enclosing polygons if they are part of the same node. Use the CONNECT command to establish nodal connectivity before using this option. This option checks nodes on the composite plane and cell plane.

Dracula Reference

Operation Block Commands

O

(+) Flags *layer-a* polygons and *layer-b* polygons that cut/overlap each other. The violation flag covers the edge segments of *layer-b* that intersect *layer-a*. Within a conjunctive rule, only *layer-b* (the enclosing layer) receives error flags.

P

(-) Flags segments of edges that project onto each other. Two edges project if perpendicular lines from a referenced edge intersect the other edge. The referenced edge is the edge most closely aligned to the x or y axis. The intersection portions define the error segments generated.

P'

(-) Inverse of the *P* option. Flags edges that do not project onto each other.

Q

Lets you extend the error marker values in the upper boundary for an upper value of enclosure. You can use this option only in the ENC command along with the [T] option and conjunctive length checking.

By default, Dracula does not do the extension. But, if you need the extension, you need to use this Q option.

R

Outputs in trapezoid format the trapezoids enclosing the projection of violation flag pairs. A projected reference edge is the violation edge aligned closest to the x or y axis that projects orthogonally from that reference edge. Specify the output layer only. Do not specify the OUTPUT error cell. This option turns on the *P* option.

R'

Outputs error flags in trapezoid format so you can reuse error data. You can process the layer created by the error flags with logical operations. The flags are one resolution unit wide as specified in the Description block, and extend to the outside edge of the layers. Do not specify an OUTPUT error cell when you use this option. Do not use this option with the *R* option.

S

(+) Flags violations on the enclosed and enclosing polygons if edges of an enclosing polygon are within a “square boundary” and outside the other polygon. When you use this option, you can also specify the CORNER-CORNER and CORNER-EDGE options. The default values for *n2* and *n3* are *n1*. If you do not specify the CORNER-CORNER option, *n2* is set equal to *n1*. If you do not specify CORNER-EDGE option, *n3* is set equal to *n1*. Use the following syntax for these options:

```
ENC[S] layer1 layer2 LT n1 {CORNER-CORNER n2} {CORNER-EDGE n3}
```


Dracula Reference

Operation Block Commands

Note: The CORNER-CORNER checking value, $n2$, and the CORNER-EDGE value, $n3$, cannot be greater than the normal checking value, $n1$, because $n2$ and $n3$ act like filters and reduce the number of errors that were initially caught by $n1$.

Only 90 degree angles are treated as "CORNER."

T

(+) Flags outside segments of enclosed polygons (*layer-a*) that touch inside segments of enclosing polygons (*layer-b*).

For the ENC[T] functionality, the Q option has been added to the code. Historically, Dracula incorrectly marked error markers in the case of the following sequence of checks:

```
ENC [T] . . . . &  
LENGTH
```

The Q option in the ENC command lets you extend the error marker values in the upper boundary for an upper value of enclosure. Now, by default Dracula does not do the extension. But, if you still need this extension, you can use the [Q] option which puts back this extension.

U



(-) Does not flag violations on the enclosed and the enclosing polygons if they are part of the same node. Use the CONNECT command to connect nodes before using this option. Checks data on the cell plane, on the composite plane, and between the cell and composite planes.

U'



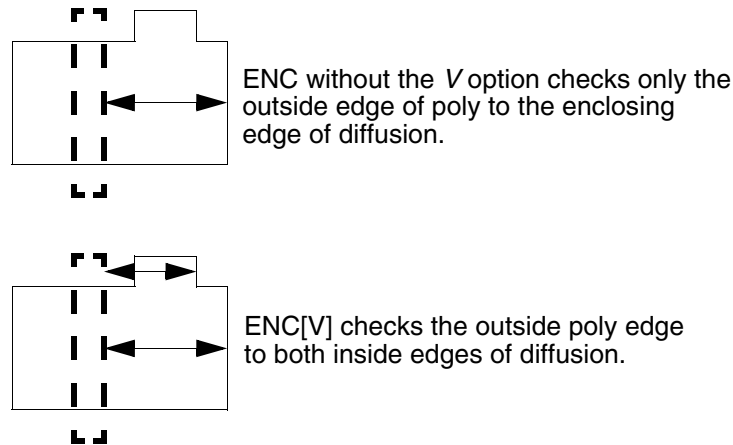
(-) Inverse of U option. Flags violations on the enclosed and the enclosing polygons if they are part of the same node. Use the CONNECT command to connect nodes before using this option. Checks data on the cell plane, on the composite plane, and between the cell and composite planes.

Dracula Reference

Operation Block Commands

V

(+) Checks the outside edge of *layer-a* to the inside edges of all surrounding geometries of *layer-b*.



W

Filters out errors that are not at the end of the metal line. The [W] option turns on the [P] option automatically.

X

Checks the delta value in the x direction between two edges. The edges must project onto each other. Does not check non-Manhattan data.

Y

Checks the delta value in the y direction between two edges. The edges must project onto each other. Does not check non-Manhattan data.

Layer Options

O

Checks the original layer, not the layer generated by the previous operation in the conjunctive rule. Use this option only with a conjunctive rule and on a layer with error flags.

Example 1

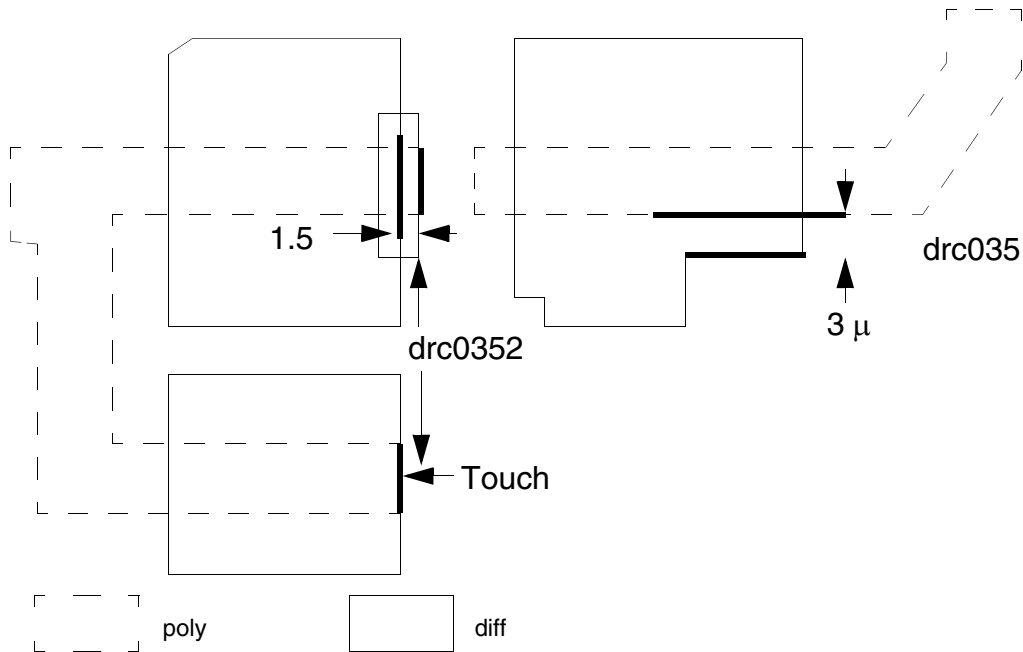
```
ENC layer1 layer2 LT 1 &  
ENC layer1 layer3 LT 1 &  
ENC layer1[0] layer3 LT 2 OUTPUT err1 2 3
```

Dracula Reference

Operation Block Commands

Example 2

```
ENC[T] poly diff LT 4 OUTPUT drc03 51
ENC[T] diff poly LT 2 OUTPUT drc03 52
```



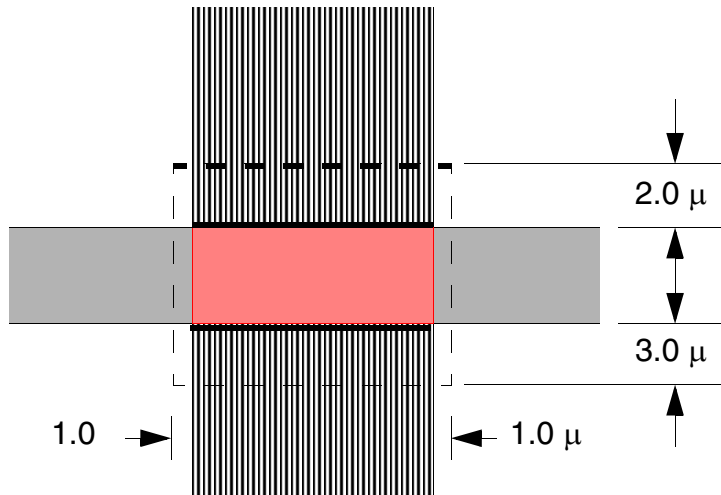
Example 3

This example checks for depletion (*dep*) completely enclosing or touching the transistor gate (*xtor*) in the direction of diffusion only.

Dracula Reference

Operation Block Commands

```
AND      poly diff xtor
ENC[T]   xtor poly LT .001 &
ENC[TO]  xtor depl LT 3 OUTPUT drc03 54
```



Example 4

This example checks for *poly* overlapping or touching *diffusion* (gate extension).

```
ENC[T] diff poly LT 2 OUTPUT drc03 52
```

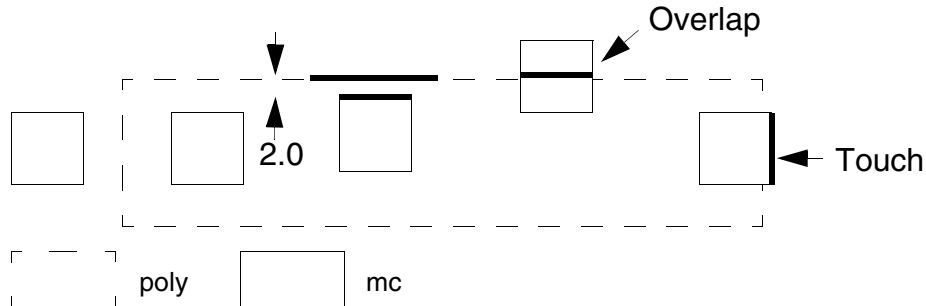
Example 5

This example checks for *poly* that completely overlaps metal contact (*mc*) and for touch. Note in the example that if you include the *E* option, the metal contact outside is also flagged.

Dracula Reference

Operation Block Commands

```
ENC[TO] mc poly LT 3 OUTPUT drc03 53
```



Example 6

Example 3 can be rewritten:

```
ENC[T] xtor&xtorl poly LT .001 &  
ENC[TO] &xtorl depl LT 3 OUTPUT drc03 54
```

Checks for depletion (*depl*) completely enclosing or enclosing and touching the transistor gate (*xtor*) in the direction of diffusion only.

The *xtor* layer is renamed *&xtorl* in the conjunctive rename operation, and now *&xtorl* contains the error flags from the joined operations. You can now use *&xtorl* in a further conjunctive operation:

```
ENC[TO] &xtorl enh LT 3 OUTPUT drc03 55
```

Checks for enhancement (*enh*) completely enclosing or enclosing and touching the transistor gate (*xtor*) in the direction of diffusion only. The conjunctive rename operation is optional but saves CPU resources.

Example 7

The following example outputs *ptie* (*p+* diffusion in *pwell*) polygons enclosed by *pwell* less than 10 microns. The *ptie* polygons that touch the *pwell* inside-edge are also output. The output layer is *badptie* and is in trapezoid format.

```
ENC[T] ptie pwell SELLT 10 badptie
```

Dracula Reference

Operation Block Commands

Example 8

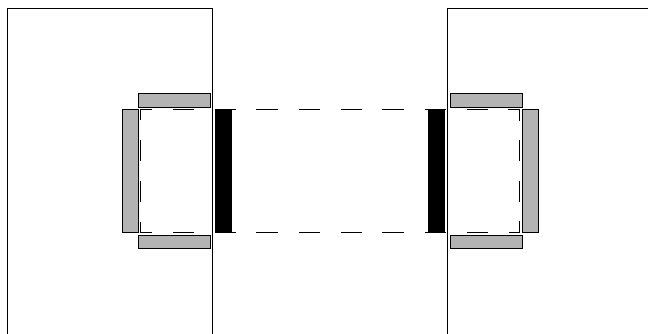
The following example checks for *diffusion* (diff) enclosing metal contact (mc) on nonparallel edges. For example, rectangular contacts inside a diffusion area with its corners removed, having 45-degree edges.

```
ENC[C'] mc diff LT 2.5 OUTPUT drc03 57
```

Example 9

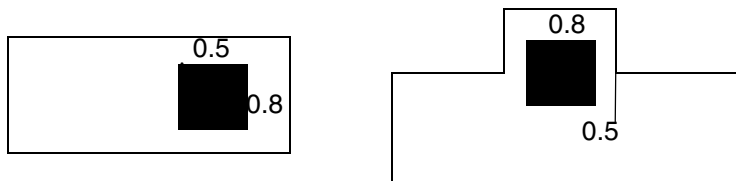
```
ENC[RO] deepn buried LT .001 dbcont
```

The *dbcont* are trapezoids with one resolution width specified in the Description block.



 deepn  buried  dbcont

Example 10

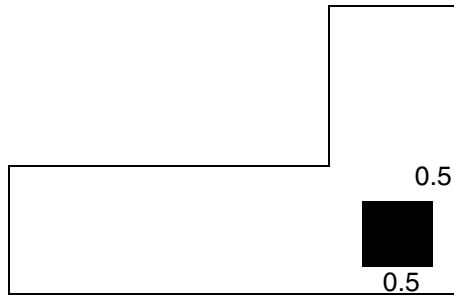


ENC via metal lt 0.5 output err 1

Dracula Reference

Operation Block Commands

ENC[W] via metal lt 0.8 output err 2



Not metal line end

ENCBASEDOVLP

```
ENCBASEDOVLP layer-a layer-b {trapfile} {OUTPUT c-name l-num}
```

```
FIXED-WIDTH width
```

```
TABLE = {  
    X1      Y1      W1  
    X2      Y2      W2  
    .....  
    Xn      Yn      Wn  
}
```

Description

Performs an overlap check based upon enclosure in the Dracula tool. Typically the command rules used to ensure a minimum intersection area of two layers are described in a table format and are lengthy, complicated, and numerous. In order to get complete coverage of the intent of the rules, it is common for a single rule to be expanded into tens or even hundreds of commands.

ENCBASEDOVLP efficiently performs a check of the rules in a table format, eliminates the need to write numerous complicated rules to achieve better performance from the check, and provides an accurate hierarchical check of overlap based upon enclosure.

The following definitions apply to overlap based upon enclosure checks:

- Enclosed shape

A rectangular or polygonal shape that is of fixed width.

- Enclosing shape

The shape that encloses an enclosed shape.

- ENC-BASED-OVLP check

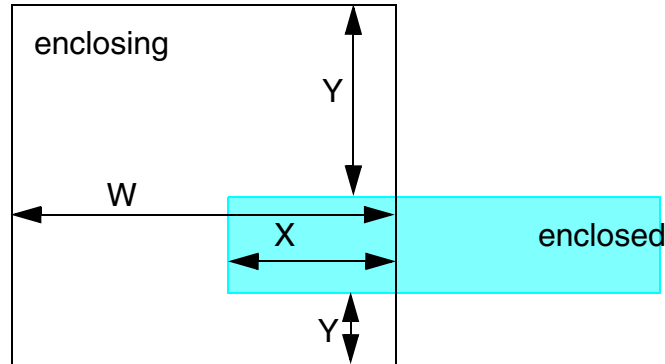
A minimum area intersection check in which a Y value measured from a specific intersection determines the minimums for X and W , where:

- ☐ Y is the minimum enclosure distance of the enclosing shape over the enclosed shape in the direction perpendicular to the center line of the enclosed shape.
- ☐ X is the minimum overlap of the enclosing shape over the enclosed shape in the direction parallel to the center line of the enclosed shape.

Dracula Reference

Operation Block Commands

- W is the minimum width of the enclosing shape measured in the direction parallel to the center line of the enclosed shape where they intersect.



The following principles apply to the enclosing and enclosed shapes.

- The enclosed shapes are rectilinear shapes of fixed width.
- The enclosing shapes consist of edges that are either rectilinear or angled edges of 45 degree only.
- The X , Y , and W values required for this check are shown in the following table:

Table 13-1 XYW Values

RULE	X	Y	W
800h	≥ 1.55	≥ 1.95	≥ 1.75
800a	≥ 1.55	≥ 1.80	≥ 1.85
800b	≥ 1.85	≥ 1.75	≥ 1.90
800c	≥ 1.90	≥ 1.25	≥ 2.35
800d	≥ 1.95	≥ 0.80	≥ 2.75
800e	≥ 2.25	≥ 0.00 or < 0.00	≥ 2.80

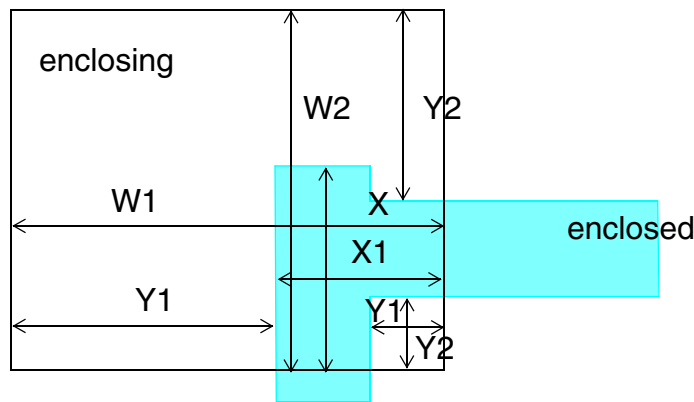
Dracula Reference

Operation Block Commands

where the X and W values are ascending and Y values are descending and all values are greater than or equal to 0.

The following is a geometric evaluation of the overlap area of the enclosing and enclosed shapes.

In the figure below, the overlap area of the enclosing and enclosed shapes has two sets of XYW values, $\{X_1, Y_1, W_1\}$ and $\{X_2, Y_2, W_2\}$. If both sets of XYW values survive the test, the enclosed shape is good. If not, the ENCBASEDOVLP check reports a bad shape.



The following steps explain the algorithm for each set of XYW values.

1. Measure the minimum value Y_1 of a given enclosed rectangular tile. For example, if $Y1$ values are $\{0.19, 0.08\}$, the minimum $Y1$ is 0.08.
2. In Table 13-1 given above, find the largest Y value and proceed down the table through the descending Y values. Find the rule with the Y value that most closely matches this minimum measured Y_1 value. For example, if Y_1 is 0.88, rule 800d is the closest match since $Y \geq 0.80$ is the closest valid value.
3. Measure X_1 and W_1 . If X_1 and W_1 meet the requirements of the rule, this is a good enclosed shape and the evaluation proceeds to the next shape.

Continuing with the previous example in which rule 800d is the closest match, the values of X must be ≥ 1.95 and W must be ≥ 2.75 . If the values of X and W are good, the shape is good and evaluation of the next shape begins.

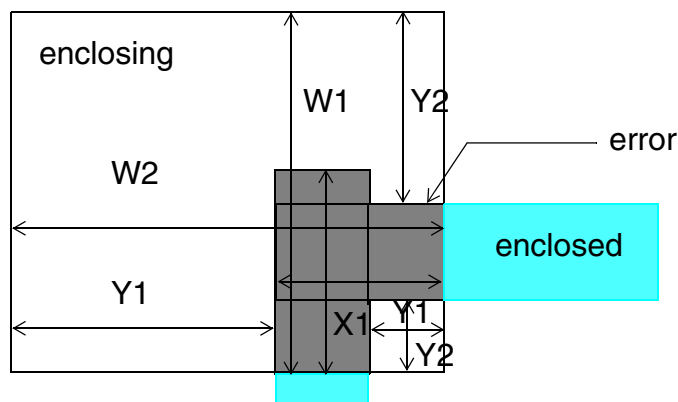
Dracula Reference

Operation Block Commands

If the first three steps fail to find a good overlap, return to step one and continue with the next set of ∇ measurements. If there are no more ∇ measurements, the shape is bad because a good overlap was not found.

The following is an output representation.

The intersection area of the enclosed shape and enclosing shape is the error representation. If the overlap check fails, the gray area shown in the figure below is written to the output layer as the error identification.



Arguments

ENCBASEDOVLP

Modifier used to direct the ENCBASEDOVLP check of the enclosing layer over the enclosed layer.

layer-a

Layer that encloses the enclosed layer.

layer-b

Layer that contains the enclosed shapes.

trapfile

Trapezoid file name. Use this option only with the *R* or *R'* option or with SELLT, SELLE, SELGT, SELGE, or SELRA operations.

Dracula Reference

Operation Block Commands

OUTPUT

Sends the results to an output cell and writes the results of the DRC check to the *outLayer* layer name you specify.

c-name l-num

c-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters and *l-num* is the cell layer number determined by your CAD system.

If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

width

Specifies the fixed width of the enclosed shapes.

x Value

Specifies the minimum overlap of the enclosing shape over the enclosed shape in the directional parallel to the center line of the enclosed shape.

y Value

Specifies the minimum enclosure distance of the enclosing shape over the enclosed shape in the direction perpendicular to the center line of the enclosed shape.

w Value

Specifies the minimum width of the enclosing shape measured in the direction parallel to the center line of the enclosed shape where they intersect.

Examples

Syntax Example

The following example performs the ENCBASEDOVLP check on the enclosed shapes with a fixed width 0.5. The shapes are enclosed by the enclosing shapes according to the table below.

Dracula Reference

Operation Block Commands

Table 13-2

```
ENCBASEDOVLP  MCB RX G190TR OUTPUT G190 62
FIXED-WIDTH    0.5
TABLE = {
                0.85 0.35 0.60
                0.85 0.20 0.65
                0.90 0.15 0.75
                1.05 0.10 0.90
                1.20 0.05 1.20
                1.30 0.00 1.30
                }
```

Geometric Example

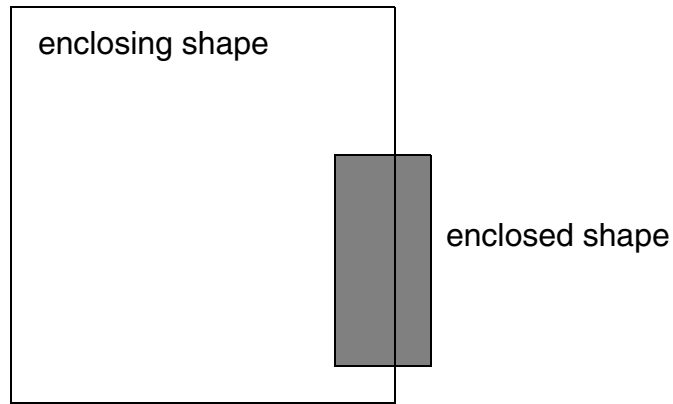
The following figure shows negative enclosure.

If the enclosed shape overlaps the enclosing shape such that a negative Υ enclosure is measured, the ENCBASEDOVLP check follows the standard evaluation algorithm. The minimum Υ is negative and the rule (from Table 13-2 above) with the Υ value that most closely matches this minimum measured Υ value is selected. Since the measured Υ is negative, the last rule in the table is always selected.

A valid R_{neg} value must be placed at the last rule in the table and with the Υ value exactly equal to the minus value of the fixed width of the enclosed shape. For example, if the width is 0.5, then the corresponding R_{neg} must be written as ($x - 0.5 \ w$) and placed at the last row of the table. If the Υ value of the last rule is not -0.5 , then it is not thought as a valid R_{neg} value.

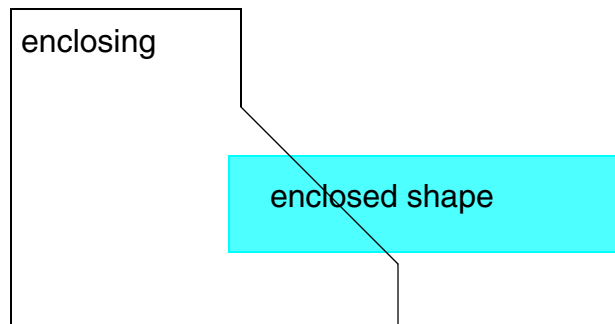
Dracula Reference

Operation Block Commands



The following figure shows angled measurements.

If any measurement encounters an angled edge, the overlap is bad and an error shape is generated.



ENCRECT

```
ENCRECT LAYER layer {NOT} RECT size1 size2 {ANGLE45 0 | 1} {RESULT trapfile}  
      {OUTPUT cellname layernum datatype}
```

Description

This command determines which polygons on the specified *layer* enclose (or NOT enclose) rectangle with sides specified as *size1* and *size2*.

Arguments

LAYER *layer*

Keyword LAYER followed by the input *layer* name.

NOT

Sets checking for *layer* polygons which do not enclose a rectangle of specified size.

RECT *size1 size2*

Specifies a rectangle with sides of *size1* and *size2*. The values are used as both horizontal and vertical sides.

ANGLE45

0 | 1 Checks which layer polygons enclose (or do not enclose) the specified rectangle, with orthogonal sides only ('0'), or also at 45 degree angles ('1'). The default value of ANGLE45 is '0' and does not check for non-orthogonal shapes. Specify "ANGLE45 1" to also check for rectangles at 45-degrees.

RESULT *trapfile*

Specifies the output filename that contains the coordinates of polygons that enclose (or do not enclose) the specified rectangle.

OUTPUT

Sends the results to an output cell of the specified *cellname*.

cellname layernum

cellname is the name of the output cell and can have six or fewer alphanumeric characters and no special characters. *layernum* is the cell layer number.

The *cellname* and *layernum* are always concatenated. If your *cellname* is rule03 and *layernum* is 5, the output cell name is rule0305.

Dracula Reference

Operation Block Commands

datatype

The *datatype* number associated with the layer number (*layernum*) of the output cell. Use *datatype* for GDSII only. Values can range from 0 to 63.

Example

```
ENCRECT LAYER metal RECT 100 200 RESULT layer1
ENCRECT LAYER metal NOT RECT 150 150 ANGLE45 1 OUTPUT res 1 1
```


EQUATION

`EQUATION K parameter = FORTRAN-expression {&}`

Description

Specifies the equations that compute device parameters from extracted layout geometric primitive parameters or other computed parameters. Use this command in conjunction with the LEXTRACT command. Specify the parameter to compute and all EQUATION parameters in the PARSET command in the Description block. If you conjunct equations, EQUATION evaluates them sequentially. All referenced parameters in equations must be computed in previous equations.



Do not use tab characters in the equation line. This can cause LPEXTR to fail.

Arguments

K

Specifies the fringe coefficient, and can apply to the same or different layer fringe extraction and overlap capacitance extraction with fringe consideration. You can use only DEPT and WIDT parameters as arguments. You must specify this equation first.

parameter

Parameter to compute.

FORTRAN-expression

The following expressions are accepted:

Numbers	Integer, real number, scientific notation
Variables	As defined in the PARSET command
Operations	Unary +, unary -, binary +, binary -, *, /, **
Parentheses	(,)
Functions	LOG, EXP, MAX, MIN, SIN, COS, TAN, ASIN, ACOS, ATAN, SINH, COSH, TANH, SQRT, LOG10, INT, and ABS.

Dracula Reference

Operation Block Commands

&

Conjunction of several equations.

Note:

- ❑ A new function, INT now allows you to get the integer value from the EQUATION command. For example, this INT function allows you to output an integer value only of all resistors for contact in order to calculate the CONTACT numbers.

See “[Example 3 - Using the INT Function](#)” in this section for more information.

- ❑ The maximum length of an equation is 80 characters.

To include equations longer than 80 characters, use a continuation as shown in the following example.

- ❑ In version 4.81 and subsequent versions, the ABS function is a new function that has been added to the list of functions used in the parameter expressions that this EQUATION command enables.

Examples

```
EQUATION ca=0.00005*area+1.0E-4*(peri-ovpr)+1.4E-4*ovpr+
0.00005*area+1.0E-12*(2*peri-ovpr)+2.0
EQUATION ca=ca+4.5*(peri-ovpr)
```

Example 1

```
*DESCRIPTION
PARSET cap1 area peri ovpr c
.
.
*END
*OPERATION
AND met1 met2 m12
NOT met2 met1 mt
PARASITIC CAP[A] m12 met1 met2
LEXTRACT cap1 m12 met2 by cap[A] capout &
EQUATION c=0.00005*area+1.0E-4*(peri-ovpr)+1.4E-4*ovpr
```

Example 2

```
PARSET TCAP CLL TPR AREA PERI C
...
PARASITIC CAP[N1] M12 M1 M2
ATTRIBUTE 2.32 1.2 1.4 1.0
...
LEXTRACT TCAP M12 BY CAP[N1] L435BDT &
EQUATION K= 2.875E-5+EXP(DEPT)+DEPT**2 &
EQUATION C= 2.32*AREA+ (PERI-TPR)+ CLL
```

For a discussion of flexible LPE parameter set extraction, refer to [Chapter 7, “Extracting Electrical Parameters \(LPE\).”](#)

Example 3 - Using the INT Function

```
EQUATION CNUM=INT ( (W- (2*M+CW) ) / (CW+CS) +1)
```

where

CNUM is the contact number.

W is the width of resistor body.

M is the margin of the resistor head and contact.

CW is the width of contact.

CS is the space of the contacts.

Example 4 - Using the ABS Function

```
*.GLOBAL
*.BIPOLAR
*.RESVAL
*.SCALE
*.EQUATION
.PARAM
.SUBCKT test MINUS PLUS
Rrmfg1r0 PLUS MINUS
+(12.0+1.1*abs(2.34)) $[R1] $W=2u
.ENDS test
```

EXPLODE

`EXPLODE{ [A] } input-layer output-layer label`

Description

Explodes Hcell shapes on the *input-layer* to the composite plane as *output-layer*. Hcell input-layer shapes on nodes that connect to nodes in the composite plane are assigned the composite plane node ID in the output-layer. Other Hcell shapes on the input-layer (for example, those on nodes internal to the Hcell) are also exploded to the composite plane, but are assigned the node ID of *label* in the output-layer.

Note: This command prevents Dracula from creating parasitic devices between shapes on the same net.



When you use the [A] option, the merge (OR) operation might cause incorrect nodal information. For example, in PRE, the CUT-TERM conduction layers use “no merge” when exploding the cell data.

Arguments

A

Merges the exploded data with the composite data of input-layer to become the output-layer.

If you do not specify this option, Dracula explodes the cell instance data to the composite plane without merging the input composite data.

input-layer

Name of the input layer.

output-layer

Sends the results to an output cell.

label

Name of the composite plane node in which to link to for shapes that form internal Hcell nodes.

Dracula Reference

Operation Block Commands

Example 1

```
EXPLODE[A]  MET1  GMET1  VSS
AND         ME2   GMET1  CAP_M2M1 ;composite ME2 to all MET1
```

Example 2

```
EXPLODE      MET1  XMET1  VSS
AND          ME2   MET1   M2M1      ;comp ME2 to comp MET1
AND          ME2   XMET1  M2CM1     ;comp ME2 to cell MET1
```

Dracula Reference

Operation Block Commands

EXT

```
EXT{[ ]} layer-a{&layer-a1}[ ] {layer-b{&layer-b1}}
    LT/LE/EQ/RANGE/SELLT/SELLE/SELGT/SELGE/SELEQ/SELNE/SELRA n1 {n2}
    {trapfile} {OUTPUT c-name l-num {d-num}} {&}

EXT[S{ }] layer-a{&layer-a1}[ ] {layer-b{&layer-b1}}
    LT/LE/RANGE/SELLT/SELLE/SELGT/SELGE/SELRA n1 {n2} {CORNER-CORNER n2}
    {CORNER-EDGE n3}
    {trapfile} {OUTPUT c-name l-num {d-num}} {&}

EXT[I] layer-a LT/LE n out-layer
```

Description

Checks the external separation between the outside edges of polygons. If you specify two layers, the Dracula product checks the outside edges of both layers against each other. If you specify one layer, the Dracula product checks the outside edges of that layer against each other.

Note: The EXT command will not flag violations between obtuse angled edges, but it will flag right angled intersecting edges when the T option is applied.

Shell environment variables

■ `setenv SPAC_LESS_SHIELD 1`

Use this variable to disable violation checking of shielding, which can take significant time. Using this environment variable may significantly decrease runtime, but can also result in decreased accuracy in designs where there are many violations.

■ `setenv SPAC_LISTERR_DIGIT [ 2 | 3 ]`

Use this environment variable to set number of digits after the decimal point for EXT/ENC/WIDTH/INT log report.

Checking Method

modes

Flat, cell, hierarchical, and composite.

checks

HDRC.

Arguments

layer-a

Is the name of the first input layer.

layer-b

Is the name of the second input layer.

&layer-a1

Renames (conjunctive rename) the layer containing the error flags. The original layer name is then restored to the original layer. Do not put a space between the *layer-a* and *&layer-a1* names, for example, *poly&rpoly*.

&layer-b1

Renames in the same way as *&layer-a1* except that it refers to *layer-b*.

Measurement Operation

Measures between the outside edge segments of polygons in *layer-a* and the outside edges of *layer-b*. Or, measures between the outside edges of *layer-a* if you only specify *layer-a*. Compares the measurements against the limits you specify and flags violations.

You must supply only one of these measurements:

LT/LE *n1*

Flags an error if the two segments are spaced less than *n1* for *LT* or less than or equal to *n1* for *LE*. Does not flag an error if the segments touch.

EQ *n1*

Flags an error if the two segments are spaced equal to *n1*.

RANGE *n1 n2*

Flags an error if the two segments are spaced less than *n2* and greater than *n1*. If you do not want acute-angle error flags, specify the *RANGE* measurement.

Dracula Reference

Operation Block Commands

SELGT/SELGE
n1

SELGT:

Outputs in trapezoid format all *layer-a* polygons that Dracula does not flag when you specify an LE measurement with the same options.

SELGE:

Has the same function as *SELGT* except that it works opposite from an *LT* operation. Do not use the OUTPUT statement with this option. Specify a *trapfile* name instead. Use this check in flat and cell modes only.

SELLT and
SELLE *n1*

Output in trapezoid format the same *layer-a* polygons associated with the errors that the program flags when you specify an *LT n1* or an *LE n1* and use the same options. Specify a *trapfile* name instead of using the OUTPUT statement. Use this check in flat and cell modes only.

SELEQ and
SELNE *n1*

Output in trapezoid format the same *layer-a* polygons associated with the errors that the program flags when you specify an *EQ n1* or an *NE n1* and use the same options. Specify a *trapfile* name instead of using the OUTPUT statement. Use this check in flat and cell modes only.

SELRA *n1 n2*

Outputs in trapezoid format the same *layer-a* polygons associated with the errors Dracula flags when you specify a *RANGE n1 n2* measurement with the same options. Specify a *trapfile* name instead of using the OUTPUT statement.

trapfile

Is the trapezoid file name. Use this option only with the *R* or *R'* option or with the *SELLT*, *SELLE*, *SELGT*, *SELGE*, or *SELRA* operations.

OUTPUT

Sends the results to an output cell.

Dracula Reference

Operation Block Commands

c-name l-num

C-name: Is the name of the output cell and can have six alphanumeric characters or fewer and no special characters.

L-num: Is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

Is the datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

&

Indicates a conjunctive rule.

Note: The output cell contains clean polygons in polygon format, resulting in reduced trapezoid construction lines and a more compact cell. The *trapfile* option and OUTPUT command are mutually exclusive.

Command Options

Options that are applicable to hierarchical Dracula are indicated with an  icon in the margin. The plus (+) in an option adds error flags. The minus (-) in an option removes error flags.

B

Flags the actual part of a violation on a segment instead of the entire segment. This option can be applied only with LT or LE modifiers. Neither RA, GT, nor GE modifiers can contain the B option.

Note: Causes round-off side effects on non-45-degree edges.

C

(-) Flags the edge-pair when the edges are parallel.

C'

(-) Is the inverse of C option. Flags non-parallel edges.

E



(+) Flags a polygon that is totally enclosed by a polygon from another layer. See [Example 14 Using the O, E, and G Options](#) in this section for more information.

Dracula Reference

Operation Block Commands

F

Outputs error flags in trapezoid format that execute information for edge-sizing SIZE[E] operations.

F'

Outputs error flags in trapezoid format. You must specify only *trapfile* name. It is similar to R option but does not turn on the P option.

G

(+) The G option is equivalent to using the O option along with the E option. This G option flags polygons that are fully enclosed by another layer as well as the cut off part of polygons enclosed by another layer. Error flags cover segments within *layer-a* and *layer-b* that outline the overlapping area of the two polygons. See [“Example 14 Using the O, E, and G Options”](#) in this section for more information.

H

(+) Flags the outside edges of notched *layer-a* polygons that fail the spacing check. A notch is a set of non-adjacent facing edges, or adjacent facing edges that create an external angle of less than 90 degrees. Use this option only with a single input layer.

I

Fills up the notches of a single layer automatically.

N



(-) Does not flag violations on polygons that are part of the same node. Use the CONNECT command to connect nodes before using this option. This option checks composite plane and cell plane nodal information, as well as nodal information between the composite plane and cell plane.

N'



(-) Is the inverse of N option. Flags violations on polygons that are part of the same node, but not the same polygon. Use the CONNECT command to connect nodes before using this option. Checks composite plane and cell plane nodal information, as well as nodal information between the composite plane and cell plane.

O

(+) Flags *layer-a* and *layer-b* polygons that cut/overlap each other. Error flags cover segments within *layer-a* and *layer-b* that outline the overlapping area of the two polygons. The segments flagged are the segments of the trapezoids of these two polygons that cut the edges of the polygons. Does not flag a polygon that fully encloses a polygon of the other layer. See [“Example 14 Using the O, E, and G Options”](#) in this section for more information.

Dracula Reference

Operation Block Commands

P

(-) Flags segments of edges that project onto each other. Two edges project if perpendicular lines from a referenced edge intersect the other edge. The referenced edge is the edge most closely aligned to the x or y axis.

P '

(-) Is the inverse of the *P* option. Flags edges that do not project onto each other.

R

Outputs in trapezoid format the trapezoids enclosing the projection of violation flag pairs. A projected reference edge is the violation edge aligned closest to the x or y axis that projects orthogonally from that reference edge. You must specify the output layer only. Do not specify the OUTPUT error cell. Turns on the *P* option.

R '

Outputs error flags in trapezoid format so you can reuse error data. You can process the created layer (from the error flags) with logical operations. The flags are one RESOLUTION unit wide (as specified in the Description block) and extend to the outside edge of the layers. Do not specify an OUTPUT error cell when you use this option. You cannot use this option with the *R* option.

S

(+) Flags violations on two polygons if edges of one polygon are within a “square boundary” and outside of the other polygon. When you use this option, you can also specify the CORNER-CORNER and CORNER-EDGE options. The default values for *n2* and *n3* are *n1*. If you do not specify the CORNER-CORNER option, *n2* is set equal to *n1*. If you do not specify CORNER-EDGE option, *n3* is set equal to *n1*. Use the following syntax for these options:

```
EXT[S] layer1 layer2 LT n1 {CORNER-CORNER n2} {CORNER-EDGE n3}
```

Note: The CORNER-CORNER checking value, *n2*, and the CORNER-EDGE value, *n3*, cannot be greater than the normal checking value, *n1*, because *n2* and *n3* act like filters and reduce the number of errors that were initially caught by *n1*. *n2* must be greater or equal to $n1/\sqrt{2}$.

Only 90 degree angles are treated as "CORNER."

T

(+) Flags outside segments of *layer-a* polygons that touch outside segments of *layer-b* polygons. Reports edges and corners that touch.

Dracula Reference

Operation Block Commands

T'

Flags outside segments of *layer-a* polygons that touch outside segments of *layer-b* polygons at one point. Reports corners that touch.

U



(-) Does not flag violations on polygons that are part of the same node. Use the CONNECT command to connect nodes before using this option. This option checks only data on the composite plane.

U'



(-) Is the inverse of *U* option. Flags violations on polygons that are part of the same node. Use the CONNECT command to establish nodal connectivity before using this option. This option checks only the data on the composite plane.

V

(+) Causes polygons to become visible through any inside surface shielding polygons. Inside surfaces of *layer-a* and *layer-b* are ignored.

EXT[V] = normal EXT + two layer inside shielding off (inside is visible) + single layer shielding off (visible through the layer itself)

V'

(+) Causes polygons to become visible through any inside surface shielding polygons except the polygon itself.

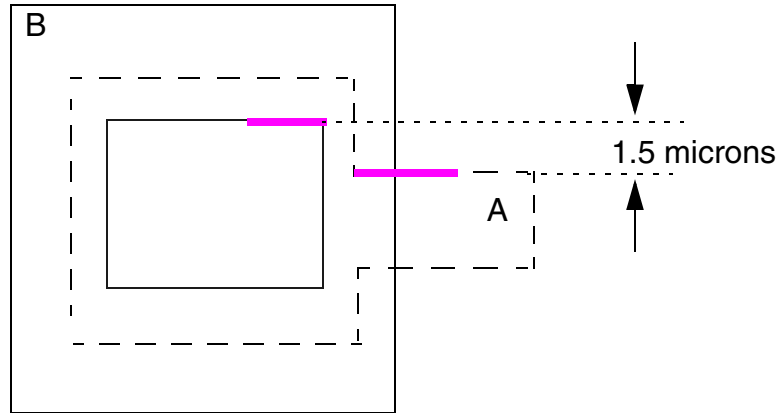
EXT[V]' = normal EXT + two layer inside shielding off (inside is visible)

For example,

```
EXT[V] A B LT 2 OUTPUT drc03 50
```

Dracula Reference

Operation Block Commands



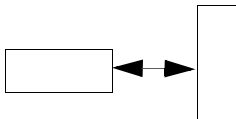
Note: If the option is ∇ , then the errors appear as shown in the above figure where the magenta-colored lines represent the errors. If the option is ∇' , then no errors appear.

X

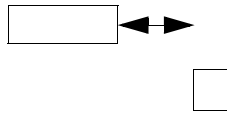
Checks the spacing value in the x direction between two edges. The edges must project onto each other. Does not check non-Manhattan data.

Y

Checks the spacing value in the y direction between two edges. The edges must project onto each other. Does not check non-Manhattan data.



Projecting edges are checked.



Nonprojecting edges are not checked.

Z

Outputs a fictitious region formed from the alignment point. This option must be used with the *R* option. When the error segment output by the *R* option is aligned in the X or Y axis, the region created has a zero projected area, and thus is left out of the final results. The region formed by the *Z* option extends both up and down by 1 dbu if the error segment is aligned along the X axis. The region extends to the left and right if the error segment is aligned along the Y axis.

Dracula Reference

Operation Block Commands

Layer Options

O

Checks the original layer, not the layer generated by the operation in the conjunctive rule. Use this option only with a conjunctive rule and on a layer with error flags.

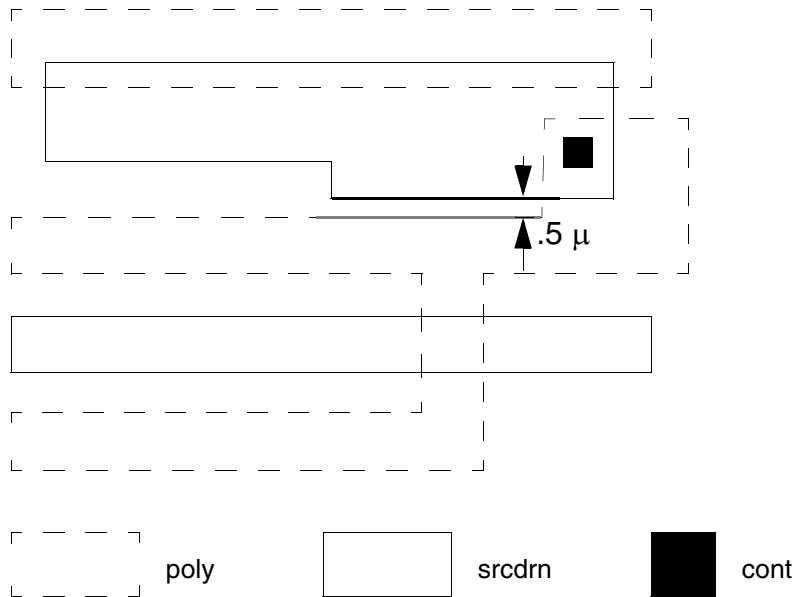
Example 1

```
EXT layer1 layer2 LT 1 &  
EXT layer1 layer3 LT 1 &  
EXT layer1[O] layer3 LT 2 OUTPUT err1 23
```

Example 2

```
CONNECT poly srcdrn BY cont  
EXT poly srcdrn LT 1 OUTPUT drc05 73
```

The error in the figure is a false error. It is not reported as a violation if the node option is flagged (that is, EXT [N] . . .).

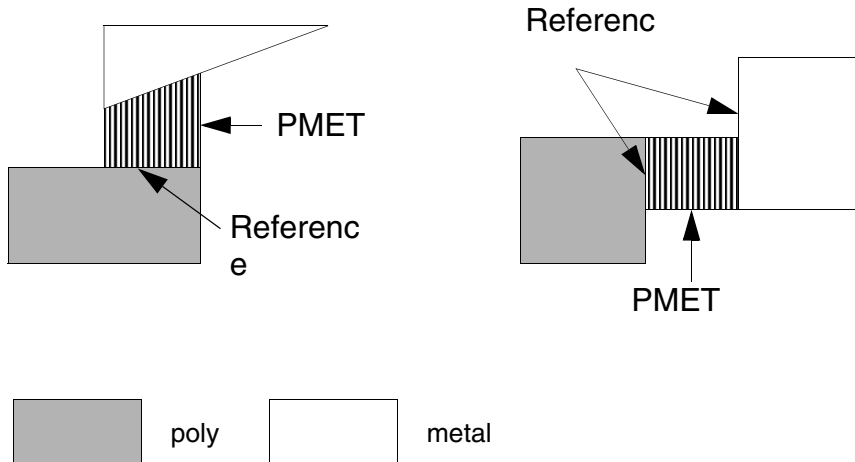


Dracula Reference

Operation Block Commands

Example 3

```
EXT[R] poly metal LT 3 pmet
```



Example 4

This example checks for external spacing of *metal*, but *metal* carrying the same electrical node is not an error.

```
EXT[N] metal LT 5 OUTPUT drc05 70
```

Example 5

This example checks for *gate* external spacing to *epi* in the direction of diffusion only. Touching is also an error.

```
ENC[T] gate poly LT .001 &  
EXT[T] gate epi LT 3 OUTPUT drc03 54
```

Example 6

Example 5 can be rewritten:

```
ENC[T] gate&gatew poly LT .001 &  
EXT[T] &gatew epi LT 3 OUTPUT drc05 55
```

This example also checks for *gate* external spacing to *epi* in the direction of diffusion only. Touch is also an error.

Dracula Reference

Operation Block Commands

The layer *gate* is renamed to *&gatew* in the conjunctive rename operation, and now *&gatew* contains the error flags from the joined operations. You can now use *&gatew* in a further conjunctive operation.

```
EXT[T] &gatew mc LT 2 OUTPUT drc05 56
```

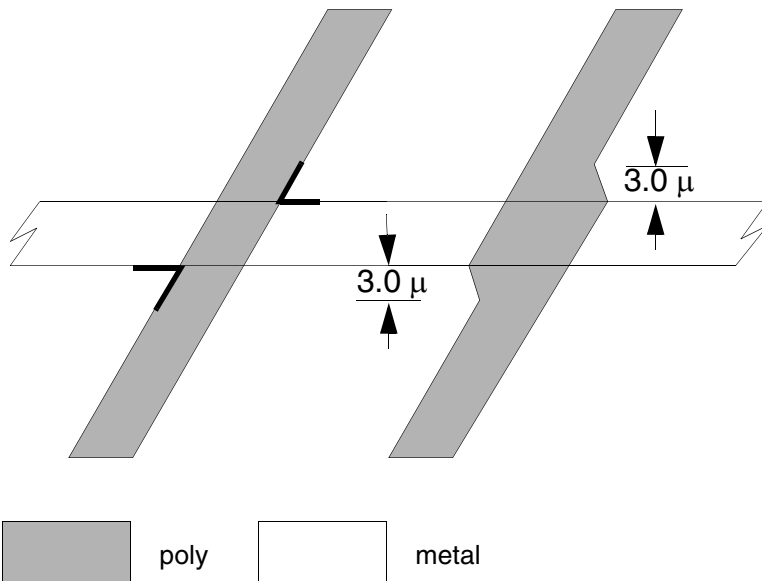
Checks for *gate* external spacing to *metal*/contact (*mc*) in the direction of diffusion only. Touch is also an error.

The use of the conjunctive rename is optional but saves CPU resources.

Example 7

This example checks for *metal* to *poly* spacing and also for *poly* crossing *metal* at an acute angle. The reflection area must be clear by at least 2 microns. Flags the intersecting edges of two layers that form acute angles.

```
EXT metal poly LT 2 OUTPUT drc05 69
```



If you do not want the acute angles flagged, change the command to the following:

```
EXT metal poly RANGE 0 2 OUTPUT drc05 69
```

Example 8

This example checks for *metal* to *poly* spacing and does not flag *poly* crossing *metal* at an acute angle as in Example 5.

Dracula Reference

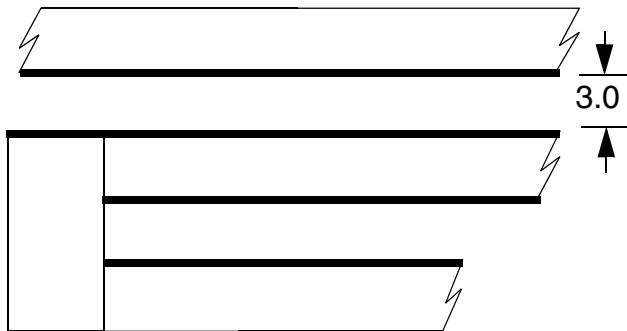
Operation Block Commands

```
EXT metal poly RANGE 0 2 OUTPUT drc05 70
```

Example 9

This example checks for poly shapes spaced 3 microns from each other and for poly notches that are less than 3 microns.

```
EXT[H] poly LT 3 OUTPUT drc05 71
```



Example 10

This example outputs all *psd* (p+ source/drain) polygons that do not have any edges spaced less than or equal to 20 microns to *pwell*. The output layer is *psd20* and is in trapezoid format.

```
EXT psd pwell SELGT 20 psd20
```

Example 11

This example outputs all *ptie* (p+ substrate tie-downs) polygons spaced greater than 40 microns from each other. The output layer, *ptie40*, is in trapezoid format.

```
EXT ptie SELGT 40 ptie40
```

Example 12

The following example checks for metal contact (*mc*) corner-to-corner spacing only. Corners that project onto adjacent corners are not checked.

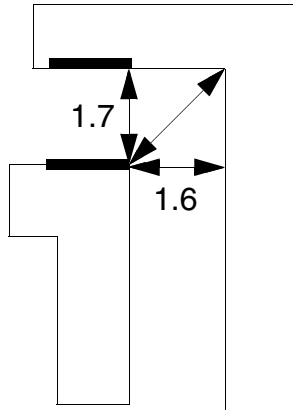
```
EXT[P'] mc LT 2.5 OUTPUT drc05 72
```



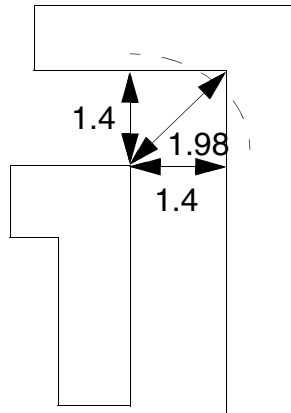
Caution

If you specify a range when using the EXT command, Dracula does not verify geometries that have certain characteristics. For example:

```
EXT MET1 RA 1.61 2.11 OUTPUT DRC40
```



Flagged



Not Flagged

Example 13

The following example outputs errors between *met1* shapes spaced less than 2 microns, including shapes with a corner-edge spacing box or a corner-corner spacing box of less than 2 microns.

```
EXT[S] met1 LT 2.0 OUTPUT DRC40
```

The following illustrates shapes that are in violation.

Example 14 Using the O, E, and G Options

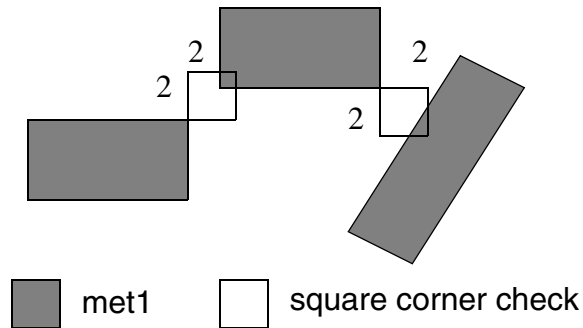
The following example contains 4 figures.

- Figure 1 shows the EXT command by itself.

Dracula Reference

Operation Block Commands

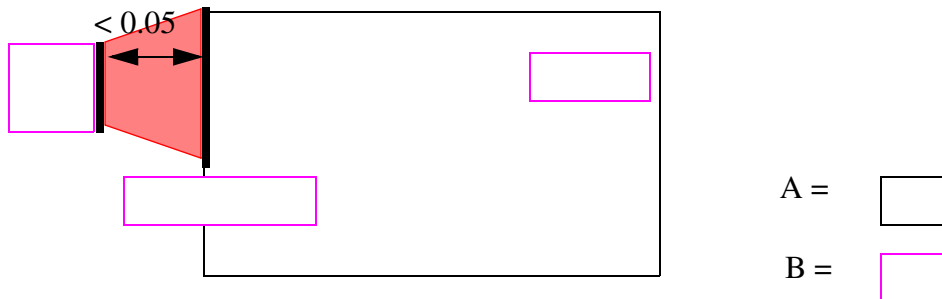
- Figure 2 shows the EXT command with the O option, which does not flag polygons that are fully enclosed by another layer.



- Figure 3 shows the EXT command with the E option, which flags polygons that are fully enclosed by a polygon from another layer.
- Figure 4 shows the EXT command with the G option, which flags polygons that are fully enclosed by a polygon from another layer as well as the cut off part of polygons enclosed by another layer.

Example 14 - Figure 1

EXT A B LT 0.05 OUTPUT ERR 1 1

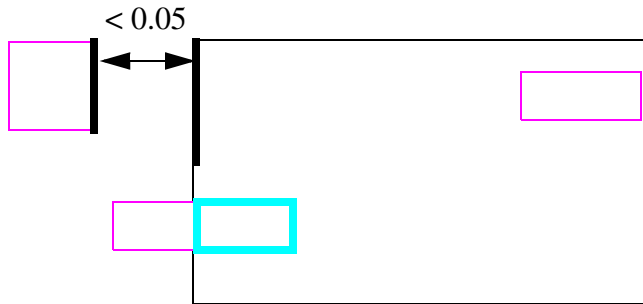


Dracula Reference

Operation Block Commands

Example 14 - Figure 2

EXT[O] A B LT 0.05 OUTPUT ERR 1 1

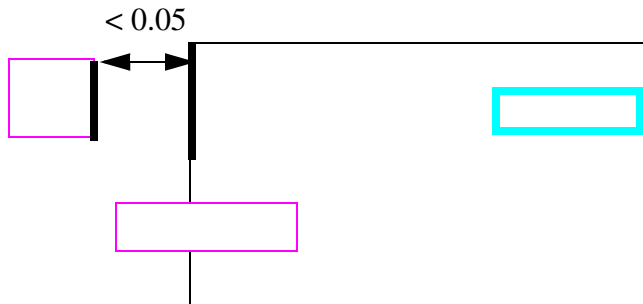


A =

B =

Example 14 - Figure 3

EXT[E] A B LT 0.05 OUTPUT ERR 1 1



A =

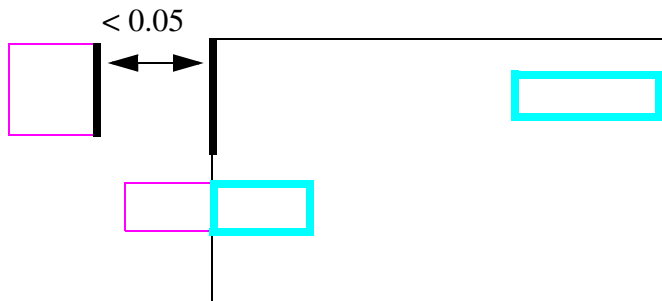
B =

Dracula Reference

Operation Block Commands

Example 14 - Figure 4

EXT[G] A B LT 0.05 OUTPUT ERR 1 1



A =

B =

Dracula Reference

Operation Block Commands

extractParasitic

```
extractParasitic(
    ( layers( layerList )
      cap( layer1 layer2 coefficientList {model(modelType)}
          {lateral(layerN piecewiseList)})
      |fringe( layer1 layer2 piecewiseList {model(modelType)}
          {equation("userEquation1")}
          {equation("userEquation2") ...}}
          {parset(pName)})
      {cap|fringe(...)...}
    )
    ( ( resistor(resLayer sheetResValue {model(modelType)}
        {cont(contactLayerList)}
        {device(deviceLayerList)}
      {multiSheetRes piecewiseList}
      {model(modelType)}
        {maxlength(length)}
        {maxwidth(width)}
        {smash(smashResValue maxResValue)}
        {corner45(compenFactor)}
      {align}
    ) {resistor(...)...}

    ( contact(contactLayer layer1 layer2 contactCoefficient)
      {model(modelType)})
    | gateRes(subtype factor) {gateRes(...) ..}
  )
)
```

Description

Extracts parasitic resistance and capacitance (RC) from specified layers. For examples and a description of the types of parasitic RC you can extract, see the section about [*"Using One Function to Extract RC Parasitics"*](#) in the *Dracula User Guide*.

This command also can support parasitic diode command if you insert `*dracrule` and `*endrule`.

Note: The resistor, device, maxlength, maxwidth, smash, corner45, and align options must be used in the order shown above.

In version 4.7.0399 and later versions, an additional enhancement is that users can specify the model type for each capacitance or resistance command in `extractParasitic()`. PDRACULA will automatically generate the model commands with the correct subtypes.

The Spice files will include model type for resistors and capacitors.

Dracula Reference

Operation Block Commands

Note: For expanded rules of `extractParasitic`, the syntax is as follows:

```
*DESCRIPTION
  MODEL = RES[subType1] modelType1 RES[subType2] modelType2...
  MODEL = CAP[subType3] modelType3 RES[subType4] modelType4...
*END
*OPERATION
.....
```

Arguments

layerList

Layers to be used for capacitor terminal layers. The layer list is a list of layer names or sublists. The maximum number of defined layers is 32. The format for this argument is as follows.

```
layers ( layer1 layer2 ( layer3 layer4...) ... )
```

The order of the list defines the geometric level of the layers. The first entry is considered the lowest level in the layout and the last entry is the highest level. The list is composed of at least two elements. The element can be a layer name or a layer sublist.

A layer sub-list consists of non-overlapping layers on the same geometric level in the layout. A layer sub-list must contain at least two layers.

Any layer in the list must be a connected layer or derived from a connected layer.

layer1, layer2

Layers involved in an extraction operation. For capacitance operations, they must be two different layers defined in the layer list.

coefficientList

Coefficients to use in the capacitance calculation. There are two forms.

```
areaCoeff perimeterCoeff
areaCoeff coCoeff upCoeff downCoeff
```

modelType

The user-specified model type for each capacitor or resistor or contact extraction command. The model option can be placed anywhere after the coefficient list, but before the lateral command for plate capacitor and after piecewise list or resistance coefficient for resistor or contact. PDRACULA will automatically generate model commands with correct subtypes.

areaCoeff

The value to use for overlap area capacitance calculations.

Dracula Reference

Operation Block Commands

perimeterCoeff

The value to use for perimeter capacitance calculations.

coCoeff

The value to use for colinear edge capacitance calculations.

upCoeff

The value to use for sidewall-up capacitance calculations.

downCoeff

The value to use for sidewall-down capacitance calculations.

lateral

Specifies the fringe effect for an overlap capacitor.

layerN

Terminal layer for considering the possible fringe effect on the overlap capacitance.

userEquation

User defined equation for fringe capacitance calculation. The syntax for equation is the same as the syntax for EQUATION command.

Note: *userEquation* is quoted in the `extractParasitic` command.

pName

User defined parset name. The default parset name used in LEXTRACT when equation command is used in `extractParasitic` is CAPF. If the equations contain parameters other than C, WIDTH, DEPT and K, user-defined parset name is required. When parset is specified, PDRACULA uses *pName* in the expanded rule file instead of the default one.

Note: The definition of *pName* must either in the Description block of the same rule file or a pre-defined parset name.

piecewiseList

For capacitance, this is a list of coefficient pairs defining the piece-wise fringe capacitance. The list has the following form:

(separationRange fringeCoeff)

When the separation of two layers is below the *separationRange*, the corresponding *fringeCoeff* is applied as the unit length capacitance value.

For resistance, this is the list of width, sheet resistance value pairs for *multiSheetRes* option. This list has the following form:

Dracula Reference

Operation Block Commands

```
((width1 sheetResValue1) (width2 sheetResValue2)... (widthn  
sheetResValuen))
```

Assume W1 and W2 is the width on the two side of a resistor body, (see Chapter 9, Extracting Parasitic Resistance (PRE)) and $\text{width} = \max(W1, W2)$,

$0 < \text{width} \leq \text{width1}$, use *sheetResValue1* as sheet resistance

$\text{width1} < \text{width} \leq \text{width2}$, use *sheetResValue2* as sheet resistance

...

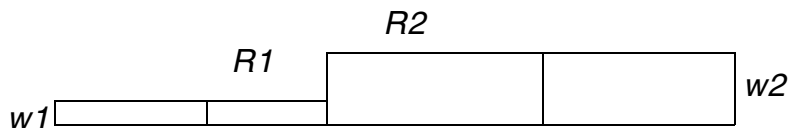
$\text{width}(n-1) < \text{width} \leq \text{widthn}$, use *sheetResValuen* as sheet resistance

Resistance is calculated as,

```
Weffective=(W1+W2)/2 L=AREA/Weffective
```

```
R = sheetResValuen * L/Weffective
```

In the following example, if *multiSheetRes* is used, width for resistor R1 is *w1* and width for resistor R2 is *w2*. The effective width of R1 is *w1* and of R2 is $(w1+w2)/2$.



Note: The *multiSheetRes* option cannot be used with flexible LPE and the field, *sheetResValue*, after *resLayer* must be specified. If the width of a resistor exceeds the maximum width specified, *sheetResValue* of maximum width is used.

resLayer

Name of the parasitic resistor layer. This layer must be defined in a CONNECT-LAYER command.

sheetResValue

Sheet resistance value in units per square, where the units correspond to the resistance units specified in the UNIT command.

contactLayerList

One or more contact layers to use in the contact resistance capacitance calculation. If you do not specify a contact layer, the function searches the contacts of all *resLayers* used in CONNECT commands.

Dracula Reference

Operation Block Commands

deviceLayerList

One or more device terminal layers belonging to *resLayer*. If you do not specify a device layer, the function will search all gate layers used in ELEMENT MOS and ELEMENT LDD commands.

length

Maximum length of a parasitic resistor device in a number of squares. If a resistor device exceeds this maximum, the resistor is partitioned into multiple devices evenly.

Each resistor device that is formed is smaller than the maximum value, and the values are distributed evenly on the resistor. The maximum difference between two sections is one. For example, if the resistor body is 91, and you specify a maximum length of 20, the sections are 19, 18, 18, 18, and 18.

width

Value used by the WIDTH command within the command sequence CUT-TERM runs. For an example of how CUT-TERM uses the WIDTH command and maximum width value, refer to the [“Example” in the CUT-TERM section](#). Default is 40.

smashResValue

Resistance value used as a threshold. Series resistors with values below *smashResValue* are smashed.

maxResValue

Maximum value of a resistor that results from smashing serial parasitic resistors. This prevents the Dracula software from creating a big resistor from subsequent smashing.

compenFactor

Argument used for compensating resistance in 45-degree terminals.

align

The terminal cut aligns with the context edge if you specify this argument for resistor.

contactLayer

Layer to use in the contact resistance calculation.

layer1, layer2

The connection layers of the contact.

contactCoefficient

Value to use for contact resistance calculations. The default equation is

$$R = \text{contactCoefficient} / \text{contactArea}$$

Dracula Reference

Operation Block Commands

subtype

MOS element and subtype in which you want to extract gate resistance.

factor

Gate resistance factor, which is used as

$$R = \text{factor} * (w / L)$$

**dracrul*, **endrul*

For PRE cases, the Dracula rule embedded inside **dracrul* and **endrul* is put behind re-extraction elements in the expand rule file. You can use this function to do parasitic diode extraction composite with the `extractParasitic()` command.

Example 1

```
extractParasitic(
( layers( BULK (NWEELL PWEELL) (PSD NSD POLY) MET MT2)
fringe(MET1 MET1 (1.0 0.02) (1.5 0.015)) ; 1-layer fringe
fringe(MET1 MET2 (1.0 0.03) (1.5 0.01)) ; 2-layer fringe
cap (BULK POLY 0.35 0.12); poly, substrate overlapped cap
cap (BULK MET1 0.4 0.15) ; met1, substrate overlapped cap
cap (POLY MET1 0.2 0.3 ; 2D3B model for overlapped cap
lateral(POLY (1.0 0.03) (1.5 0.02)))
cap (POLY MET2 0.3 0.02 0.1 0.15)) ; sidewall cap
( resistor(MET1 0.1 cont(CONTACT VIA))
resistor(MET2 0.3 cont(VIA) maxlength(30))
resistor(POLY 0.7 cont(CONTACT) device(NGATE PGATE))
contact(VIA MET1 MET2 0.1)))
```

Example 2

```
extractParasitic(
*dracrul
; Define Parasitic Junction Diodes
NOT      SUB      NWZ      PSUB
AND      PDIFC    NWZ      PDIO  ; p+ diode trm to nwell less p+ ties
AND      NDIFC    PSUB     NDIO  ; n+ diode res to psub less n+ ties
AND      RDIFC    PSUB     RDIO  ; n+ diode res to psub less n+ ties
;
PARASITIC DIO{PT} PDIO      PDIO      NWZ      ; pdiff/n-
PARASITIC DIO{NT} NDIO      PSUB      NDIO      ; ndiff/p-
PARASITIC DIO{RT} RDIO      PSUB      RDIO      ; ndiff/p-
;
*endrul
(
layers(SUB NWZ (NDIF PDIF) PCZ MCZ M1BIAS M2BIAS )
fringe(PCZ PCZ (0.32 3.972670e-06) (0.86 2.434180e-06))
fringe(MCZ MCZ (0.26 2.751891e-05) (0.80 2.514589e-05))
fringe(M1BIAS M1BIAS (0.35 2.154846e-05) (0.98 1.084768e-05))
fringe(M2BIAS M2BIAS (0.40 2.712161e-05) (1.21 1.412467e-05))
.
.
cap(SUB NWZ 1 0.9 0.8 0.7)
cap(SUB NDIF 1 0.9 0.8 0.7)
```

Dracula Reference

Operation Block Commands

```

cap(SUB PDIF      1 0.9 0.8 0.7)
cap(SUB PCZ       1 0.9 0.8 0.7)
cap(SUB MCZ       1 0.9 0.8 0.7)
.
.
.
)
(
  resistor(PCZ 7.0E-3 cont(MCDIF MCPC) device(PDEVICE NDEVICE
    RGATER LWGATE XWGATE ZGATE NWCON) smash(0.0E-3 5.0E-3))
  resistor(MCZ 0.24E-3 cont(MCDIF MCPC CNTALL) smash(0.0E-3 0.5E-3))
  resistor(M1BIAS 0.061E-3 cont(CNTALL V1ALL) smash(0.0E-3 0.05E-3))
  resistor(M2BIAS 0.061E-3 cont(V1ALL V2ALL) smash(0.0E-3 0.05E-3))
  .
  .
  contact(MCDIF MCZ NDIF 1      )
  contact(MCDIF MCZ PDIF 1      )
  .
  .
))
.
.
.
;*****
; Parasitic diode extraction formula
;*****
LEXTRACT      MDIF      PDIO      PDEVN BY DIO[PT] PTDIF &
EQUATION      P1=PERI-OVPR &
EQUATION      A1=AREA
;
LEXTRACT      MDIF      NDIO      NDEVN BY DIO{NT} NTDIF &
EQUATION      P1=PERI-OVPR &
EQUATION      A1=AREA
;
LEXTRACT      MOSF      RDIFC      RDEVN BY NODE RMDIF &
EQUATION      PERI=PERI-OVPR
.
.
.

```

Example 3

```

extractParasitic(
( layers( BULK (NWEILL PWEILL) (PSD NSD POLY) MET MT2)
fringe(MET1 MET1 (1.0 0.02) (1.5 0.015)
equation("C=(19.761/(DEPT+4.2))*WIDT*0.001")) ; 1-layer fringe
fringe(MET1 MET2 (1.0 0.03) (1.5 0.01)) ; 2-layer fringe
cap (BULK POLY 0.35 0.12); poly, substrate overlapped cap
cap (BULK MET1 0.4 0.15) ; met1, substrate overlapped cap
cap (POLY MET1 0.2 0.3 ; 2D3B model for overlapped cap
lateral(POLY (1.0 0.03) (1.5 0.02)))
cap (POLY MET2 0.3 0.02 0.1 0.15)) ; sidewall cap
( resistor(MET1 0.1 cont(CONTACT VIA)
multiSheetRes((0.05 0.15) (0.1 0.1)) model(RR1))
resistor(MET2 0.3 cont(VIA) model(RR2) maxlength(30))
resistor(POLY 0.7 cont(CONTACT) device(NGATE PGATE) model(RR3))
contact(VIA MET1 MET2 0.1)))

```

FLATTEN

`FLATTEN layer-a trapfile`

Description

Flattens Hcell data and places it in a flattened layer. In flat Dracula, the FLATTEN command copies one layer to another.

Checking Method

modes

Hierarchical, composite, and flat.

checks

HDRC.

Arguments

layer-a

Name of the input layer or a generated layer in hierarchical Dracula.

trapfile

Name of the trapezoid file. The *trapfile* name can have up to seven alphanumeric characters and no special characters. The first character must be a letter.

In flat Dracula, *trapfile* consists of trapezoid data that is an exact copy of data from *layer-a*.

In hierarchical Dracula, use this command for operations that cannot be done on hierarchical layers, such as HDRC SELECT LABEL, which requires nodal information. You can also run HDRC checking commands with large dimensional checks (for example, 10 microns) without penalizing performance or disk space for the other HDRC checking commands.

Note: *Layer-a* must be a hierarchical (unflattened) layer. If you use the command in a check-mode other than hierarchical mode, Dracula copies only *layer-a* to *trapfile*. Therefore, if you change check-mode from hierarchical to flat, you do not have to edit the rules file.

Example 1

```
FLATTEN cont cont1
```

Dracula Reference

Operation Block Commands

In flat Dracula, layer *cont1* consists of the data from *cont*. You can now use *cont1* in any Operation block command. In hierarchical Dracula, *cont* is a hierarchical layer and *cont1* is the flattened layer.

Example 2

```
FLATTEN PSUB PSUB1
NOT PSUB1 PSUB PSUBX      ;; PSUBX contains exploded cell data
LINK PSUBX TO GND         ;; link cell data to ground

AND MET2 PSUBX M2SUBX     ;; MET2 to cell PSUB overlap
AND MET2 PSUB M2SUB       ;; MET2 to composite PSUB overlap
```

FLOATCHK

`FLOATCHK contact-layer {trapfile} OUTPUT c-name l-num {d-num}`

Description

Finds floating contact geometries on the contact layer you specify. A floating contact is any contact, or part of a contact, not covered by both connect layers defined in the CONNECT command.

Arguments

contact-layer

Name of the contact layer. The *contact-layer* must be a contact used in a CONNECT command.

trapfile

Name you designate as the output file to contain composite trapezoids.

OUTPUT

Sends the results of the operation to an output cell.

c-name l-num

C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system.

If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

Example

```
FLOATCHK CONT OUTPUT CONT1 23
FLOATCHK PCONT OUTPUT PCONT2 23
FLOATCHK NSUB OUTPUT NSUB3 23
```

FRINGE CAP

FRINGE [*options*] CAP [*type*] *layer-a* *layer-b*

Description

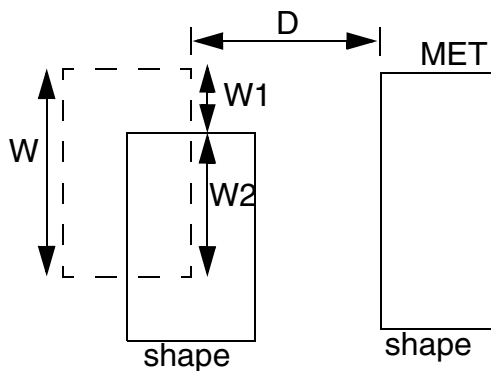
Defines the fringe capacitance of the same or different layers. Use an ATTRIBUTE CAP command after the FRINGE CAP command. ATTRIBUTE CAP associates the geometry data with the process-dependent electrical properties. ATTRIBUTE CAP does not accept units. The scale of the FRINGE CAP attribute must match the scale of the layout database.

Fringe capacitors apply to stray capacitances between two nodes on the same or different conductor layers. This command does not extract capacitances if both terminals are at the same node.

For overlapping fringe capacitance extraction, only the overlapping edges of the two input layers are considered.

The default mode of the FRINGE CAP command uses shielding. For example, in the following graphic, *shape1* of MET1 shields part of POLY from *shape2*. When calculating fringe capacitance, the shielded part is not neglected.

--- =
— POLY



$$W = W1 + W2$$

The fringe capacitance between POLY and *shape2* of MET1 is $C = W1 / D$

because part of poly (*W2*) is shielded by *shape1* of MET1.

FRINGE CAP with PRE

Use the FRINGE CAP command with PRE. Resistor device layers for PARASITIC RES are allowed as terminal layers for FRINGE CAP. In general, these resistor device layers have no nodal information.

If the first terminal layer in a FRINGE CAP command is a PARASITIC RES device layer, the FRINGE CAP device layer does not need nodal information.

Arguments

options

Can be from among the following:

R	Extracts the fringe capacitance of the CUT-TERM layers. Use this option to solve the shielding problem when you extract fringe capacitance between the resistor layer and the terminal layer of a parasitic resistor. When you use this option, either <i>layer-a</i> or <i>layer-b</i> can represent the resistor layer.
S	Specifies that only the overlapping edges of the two input layers are considered when extracting the fringe capacitance. Because the first layer is induced from the second layer, you must specify the correct layer sequence.

type

Type of fringe CAP device. The first character must be a letter from A to Z. The second character is optional and can be any letter from A to Z or any number from 0 to 9 (except 8). This code differentiates between fringe capacitor devices.

layer-a

Terminal conductor layer. This layer is the conductor layer for one side of the capacitor and must carry node information.

layer-b

Terminal conductor layer. This layer is the conductor layer for the other side of the capacitor (can be the same or different from *layer-a*) and must carry node information.

Note: The ELEMENT CAP command represents discrete devices only. The PARASITIC CAP and FRINGE CAP commands represent stray or parasitic capacitance between nodes.

Example 1

```
FRINGE[R] CAP[A] M1RES M1TRM
ATTRIBUTE CAP[A] 5 0.000027
```

Dracula Reference

Operation Block Commands

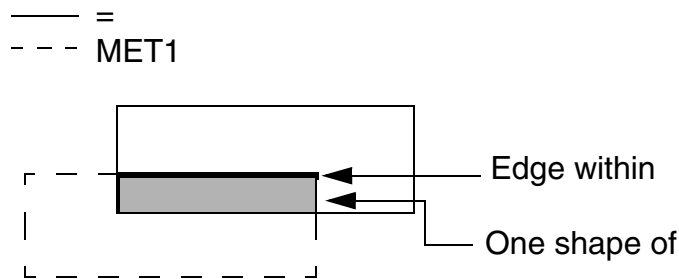
Example 2

```
AND L3 L5 L3L5
FRINGE[S] CAP[L] L3L5 L3
ATTRIBUTE 1.0 0.002
ATTRIBUTE 1.5 0.0003
```

Example 3

In the following example, layer PAOM1 is derived from MET1. POAM1 is also a subset of MET1. To prevent calculating capacitance from false edges of POAM1 (edges of shapes from POAM1 that do not overlap with shapes of MET1, for example, edges within shapes of MET1), you use the [S] option.

```
AND POLY MET1 POAM1
FRINGE[S] CAP POAM1 MET1
```



Example 4

In the following example Dracula supports zero distance FRINGE CAP.

```
FRINGE CAP[A] MET1 MET2
  ATTRIBUTE CAP[A] 0      0.0000005 ; DEPTH = 0
  ATTRUBUTE CAP[A] 0.6    0.0000005 ; 0 < DEPTH <= 0.6
  ATTRUBUTE CAP[A] 1.2    0.0000003 ; 0.6 < DEPTH <= 1.2
  ATTRUBUTE CAP[A] 1.8    0.0000002 ; 1.2 < DEPTH <= 1.8

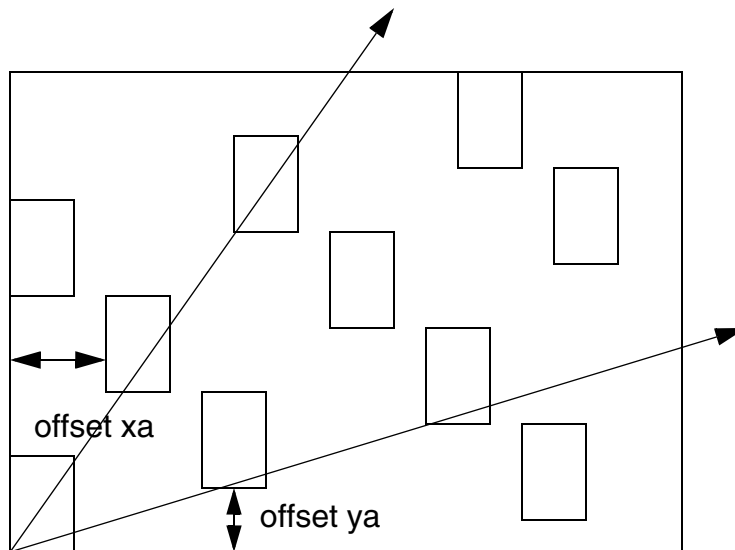
C = K x (WIDTH / DEPTH)    when DEPTH !=0
C = K x WIDTH              when DEPTH  =0
```

GENRECT

```
GENRECT input-layer XYLEN x1 y1 XYGAP xg yg {XYADJ xa ya} trapfile
```

Description

Generate rectangle shapes that overlap input layer, flat mode only. In version 4.81 and later versions, Dracula provides a new option for generating a staggered pattern rectangle array for area fill, as shown in the following figure.



Arguments

`input-layer`

Input layer name

`x1,y1`

Length/width of rectangles

`xg,yg`

Gap between rectangles

Dracula Reference

Operation Block Commands

`xa, ya`

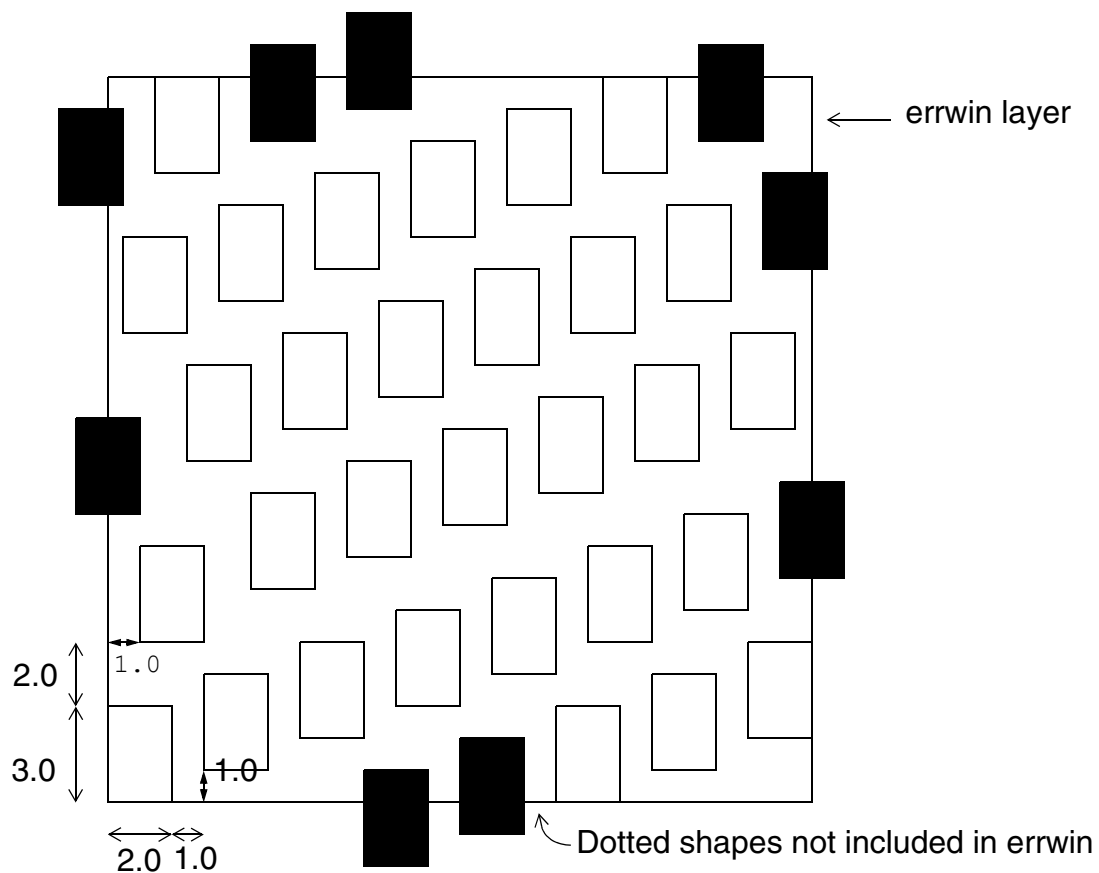
Generate each rectangle with XY-axes offset

`trapfile`

Output layer

Example

```
GENRECT errwin XYLEN 2.0 3.0 XYGAP 1.0 2.0 XYADJ 1.0 1.0 fill
```



GLOBAL-SCONNECT

GLOBAL-SCONNECT = YES/NO

Description

Specifies a global majority rule for SCONNECT. Dracula uses the local majority to determine the node within the local lower region, and it uses the global majority to determine the node within all lower regions. If, after considering these majorities, Dracula cannot decide the winning node, it chooses the node with the smallest node number.

Arguments

YES

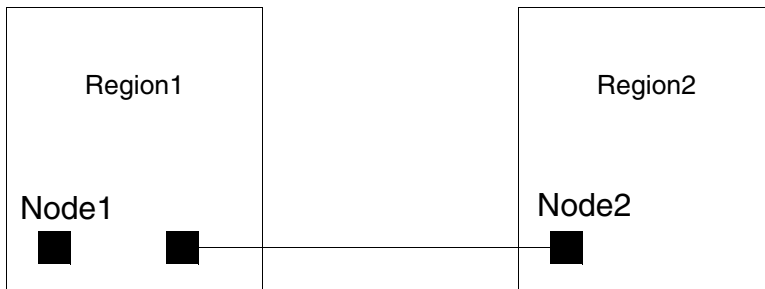
Activates global majority rule for SCONNECT conflict resolution.

NO

Specifies random choice for SCONNECT conflict resolution. Default.

Example

In the following example, Region1 is stamped Node2.



GPATHCHK

```
GPATHCHK  {[options]}  LEVEL  n  {trapfile} OUTPUT  c-name  
          l-name  {d-num}
```

Description

Checks for nodes that should have a path to both power and ground through one or more terminals of transistors, diodes, or resistors. This command assigns an error code to nodes without a proper path to either power or ground.

Before using GPATHCHK, you must use the GPATHDEF command to define the connectivity flow for your devices. For details, refer to the [GPATHDEF](#) section in this chapter.

Arguments

options

Can be from among the following:

<i>P</i>	Specifies that the pads other than power and ground pads are considered to have a path to power.
<i>N</i>	Specifies that the pads other than power and ground pads are considered to have a path to ground.
<i>n</i>	Specifies the error level number (1-3). Nodes that are assigned this error level code are reported and placed in the output cell. The error levels are 1. Nodes with a path to ground but not to power. 2. Nodes with a path to power but not to ground. 3. Nodes without a path to power or ground.

options

Specifies the name of a trapezoid data file to be generated.

Dracula Reference

Operation Block Commands

OUTPUT

Sends the results of the operation to an output cell.

Note: The output cell you specify cannot be used as the output of another command.

c-name l-num

C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

GPATHDEF

```
GPATHDEF elem{[type]} nodetype-1 FROM layer-a TO layer-b  
    {WHEN layer-c nodetype-2}
```

Description

Defines the device power and ground connectivity flow path. Each GPATHDEF command specifies an element type and a rule for the directional flow through the device. After you define the flow rules, use the GPATHCHK command to check the paths.

There are several standard rules that you do not need to specify. For example, MOS and RES elements pass power and ground signals in both directions. If necessary, you can redefine the rules for the MOS or RES elements.

The GPATHDEF command must specify the full device type with subtype if you specify an ELEMENT command. You cannot have generic rules for device types because the layers are different for each device type.

Arguments

elem

Specify one of the following element types:

MOS for a MOS device
BJT for a bipolar device
DIO for diodes

type

The two-letter (A-Z) code that denotes the device type. This code can differentiate N-channel from P-channel in CMOS or differentiate MOS devices with different implants.

nodetype-1

The type of node that flows through the device in this direction. Specify POWER or GROUND.

layer-a

The conducting layer for the terminal into which *nodetype-1* flows.

layer-b

The conducting layer for the terminal from which *nodetype-1* flows.

Dracula Reference

Operation Block Commands

WHEN

If you specify this option, the flow occurs only when the conditions you specify for *layer-c* or *nodetype-2* are true. Otherwise, the flow always occurs.

layer-c

The conducting layer for the terminal that controls the flow through the device.

nodetype-2

The type of node that enables the flow through this device. Specify POWER or GROUND.

Examples

```
GPATHDEF BJT POWER FROM COL TO EMIT WHEN BASE POWER
GPATHDEF BJT GROUND FROM EMIT TO COL WHEN BASE POWER
GPATHDEF BJT GROUND FROM EMIT TO BASE
GPATHDEF BJT POWER FROM EMIT TO EMIT
GPATHDEF BJT POWER FROM EMIT TO EMIT
```

The following example describes the diode rules:

```
GPATHDEF DIO POWER FROM ANODE TO CATHODE
GPATHDEF DIO GROUND FROM CATHODE TO ANODE
```

The following example defines the default MOS and RES rules:

```
GPATHDEF MOS POWER FROM SRCDRN TO SRCDRN
GPATHDEF MOS GROUND FROM SRCDRN TO SRCDRN
GPATHDEF RES POWER FROM RTERM TO RTERM
GPATHDEF RES GROUND FROM RTERM TO RTERM
```

GROW

`GROW input-layer dx dy {dx2 dy2} trapfile {OUTPUT c-name l-num {d-num}}`

Description

Oversizes a single side of a layer. The *dx* and *dy* values determine the direction and amount of growth. You can specify two different directions within a single command.

For more information about sizing geometries, refer to the [SHRINK](#).

Arguments

input-layer

The name of the input layer.

dx dy

The amount you want to size the layer. This value is a real number in microns or mils. This is the unit set by the SCALE command.

If $dx > 0.0$	Right edge grows
If $dy > 0.0$	Top edge grows
If $dx = 0.0$	No growth
If $dy = 0.0$	No growth
If $dx < 0.0$	Left edge grows
If $dy < 0.0$	Bottom edge grows

dx2 dy2

Specifies a second size amount. All values must be positive.

trapfile

The name of the trapezoid file.

c-name

The name of the output cell to which you can optionally output the sized data.

l-num

The layer number of the output cell.

Dracula Reference

Operation Block Commands

d-num

The datatype number associated with the layer number (*l-num*) of the output cell.

Examples

```
*DESCRIPTION
SCALE = 0.001 MIC
RESOLUTION = 0.001 MIC
*END
*OPERATION
...
GROW lay1 -0.5 -0.0 lay2 ;Grow left edge 0.5 microns
...
GROW lay5 0.25 0.25 lay6 ;SIZE MEBES data in
RELOCATE lay6 BY -0.125 -0.125 lay7 ;layf by 0.125,which
*END
```

HEDTEXT

HEDTEXT = *file-name*

Description

Specifies the Hcell text file name. This command is similar to the EDTEXT command, except it is used for texting hierarchical cells. Use this command to attach Hcell pin texts to Hcell pins.

Each statement in the file must have the following items:

- The text name

- The x and y coordinates

These coordinates are relative to Hcell coordinates. The x and y are not case sensitive.

- The name of the Hcell

Optionally, you can use ATTACH parameter to indicate the layer to which to attach the text. If you do not specify ATTACH, the text is attached according to the CONNECT-LAYER reference.

The file accepts a tab as a delimiter. You can use a semicolon (;) to start a comment. The comment must be on its own line. You can put the comment starter (;) in the first column of a line, or you can precede it with blank spaces. You cannot put a comment at the end of a line of text in an HEDTEXT file. The Dracula program ignores all text after the comment starter. You can also use a blank space in place of an equal sign (=).

Checking Method

modes

Cell and composite.

checks

HERC, HLVS, and HLPE.

Example

hedtext abc.htx

For file *abc.htx*:

VDD	x=56	y=89	inv	
VSS	x=6	y=40	ATTACH=met	inv
IN	x=41	y=90	ATTACH=poly	inv

Dracula Reference

Operation Block Commands

OUT	x=89	y=44		inv
VDD	x=123	y=96	ATTACH=met	nand

HIERARCHEN

`HIERARCHEN flayer-name hlayer-name`

Description

Converts a flattened layer to a hierarchical layer.

Checking Method

modes

Hierarchical.

checks

HDRC.

Arguments

flayer-name

The name of a flattened layer generated by a prior FLATTEN command.

hlayer-name

The hierarchical version of *flayer-name*. This layer has the same Hcells as before the FLATTEN command was issued.

If you use this command in a mode other than hierarchical, Dracula makes a copy of only layer *flayer-name* to layer *hlayer-name*. Therefore, if you change check mode from hierarchical to flat, you do not have to edit the rules file.

Example

```
HIERARCHEN fmetal hmetal
```

HOLE

`HOLE {[options]} layer-a {x_boundary y_boundary} trapfile`

Description

The HOLE command extracts holes for the specified layer and forms a new layer for later related DRC checking. This command always selects the holes region by region. Thus the region number assigned by the Dracula tool affects the results of the HOLE command.

Arguments

C

Treats two polygons with only their corners touching as the same region and selects the correct holes.

I

Generates innermost holes that contain no internal holes, but may contain `layer-a` shapes.

I'

Generates innermost empty holes that contain neither internal hole nor other `layer-a` shapes. The output is subset of [I] option.

O

Generates outermost holes that can contain both the internal holes and `layer-a` shapes, but can not be inside of other hole.

O'

Generates a subset of [O] option, which has no `layer-a` shapes inside the hole and is not inside of other hole.

`layer-a`

Specified layer to extract holes.

`x_boundary y_boundary`

Optionally specified filter box boundary.

`trapfile`

Output layer that contains holes of `layer-a`

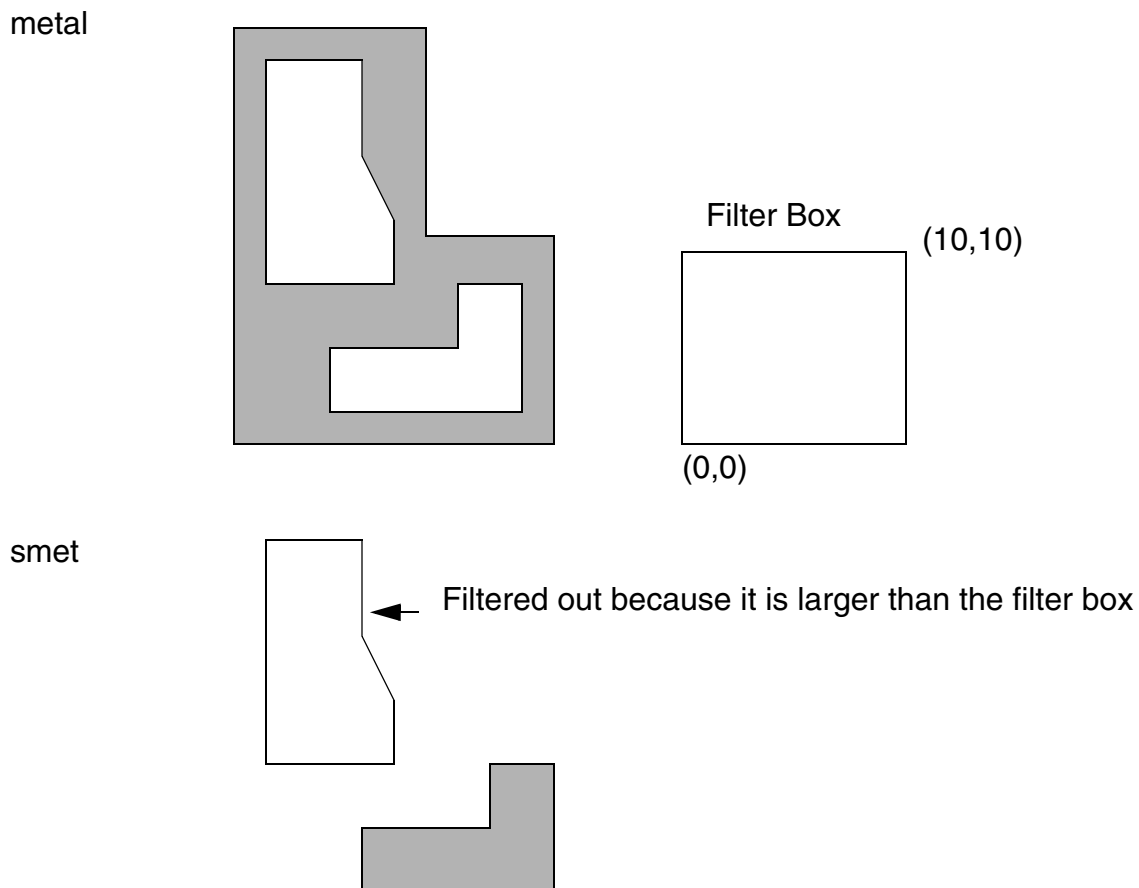
Dracula Reference

Operation Block Commands

Example 1

```
HOLE metal 10 10 smet
```

Layer *smet* contains shapes that extract holes of the input layer *metal* as seen in the following figure.



Example 2

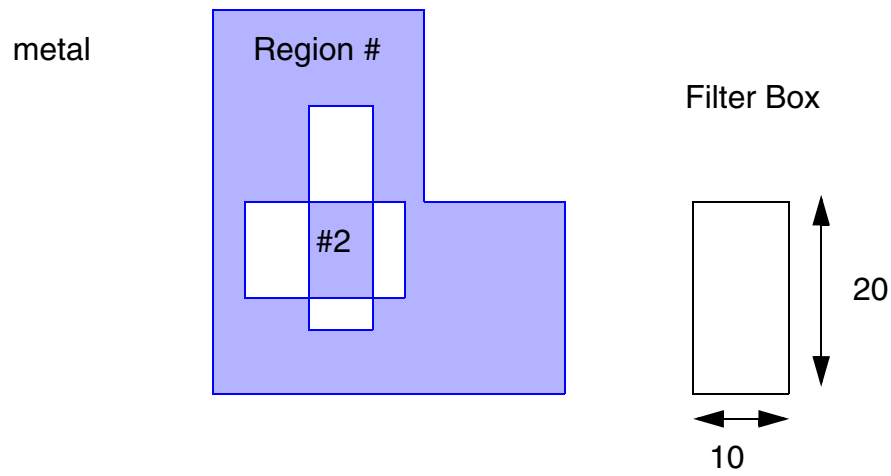
By default, the Dracula tool treats two polygons touched by only the corner points as different regions. In the following figure, the corner-touched polygons are treated as different regions (region #1 and region #2).

Dracula Reference

Operation Block Commands

```
HOLE metal 10 20 smet
```

Using the above rule and layout, layer *smet* is empty.



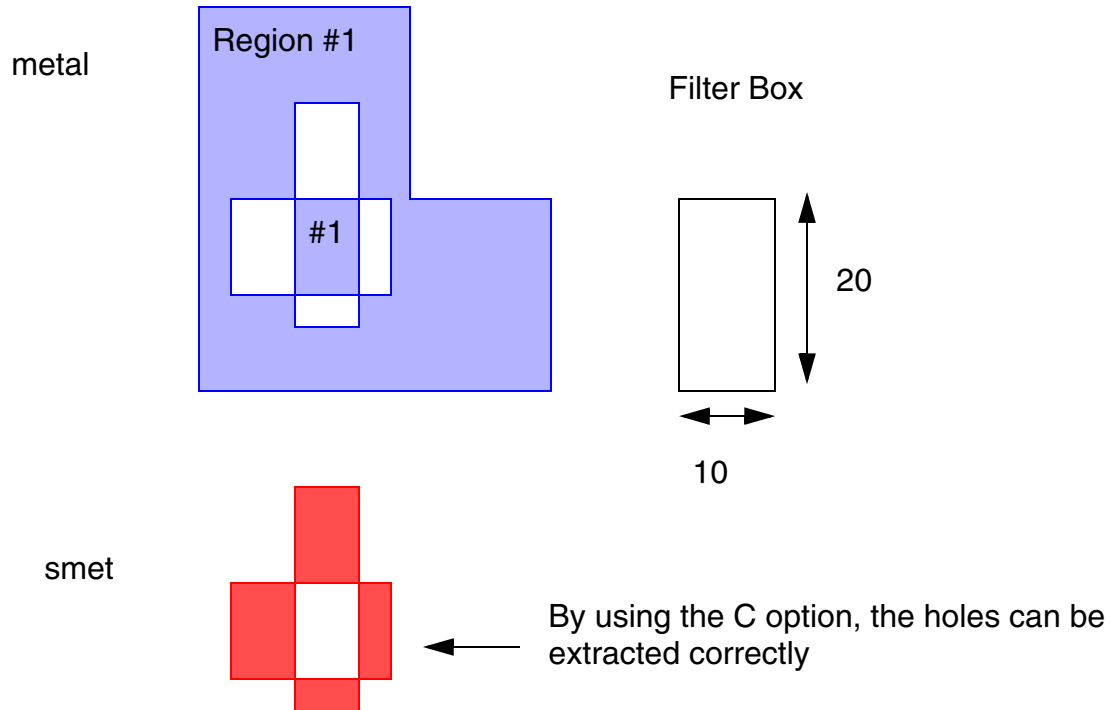
A new option in the Dracula 4.9 release, the option C for HOLE treats two polygons touched by only the corner points as the same region as illustrated in the following figure.

```
HOLE[C] metal 10 20 smet
```

HOLE[C] treats two polygons touched by only corner points as the same region and so can extract correctly the holes inside layer *metal*.

Dracula Reference

Operation Block Commands



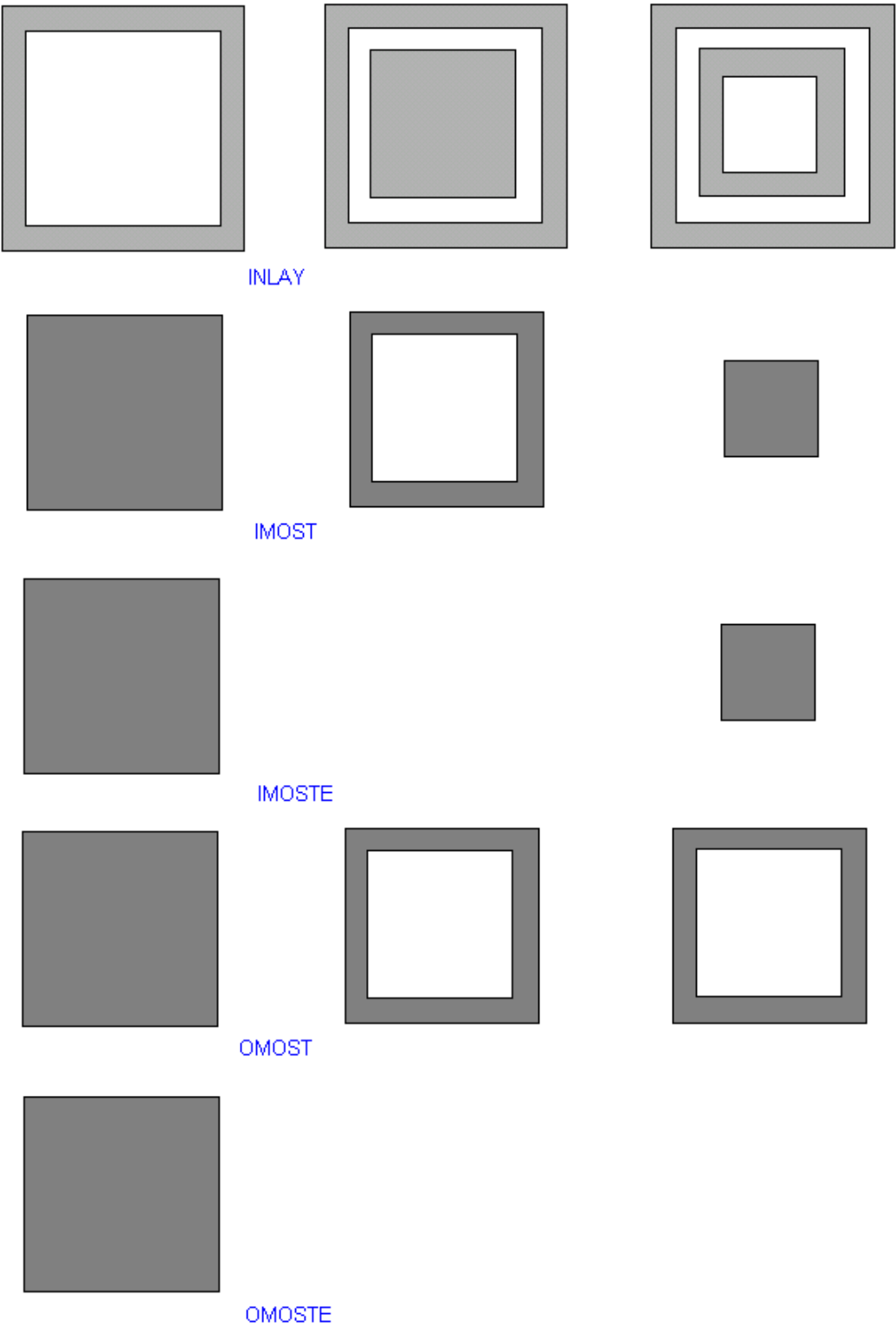
Example 3

The following example illustrates how do $\mathbb{I}$, $\mathbb{I}'$, $\mathbb{O}$ and $\mathbb{O}'$ options work:

```
HOLE[ $\mathbb{I}$ ] INLAY IMOST  
HOLE[ $\mathbb{I}'$ ] INLAY IMOSTE  
HOLE[ $\mathbb{O}$ ] INLAY OMOST  
HOLE[ $\mathbb{O}'$ ] INLAY OMOSTE
```

Figure below shows INLAY and 4 different results of the command above.

Dracula Reference
Operation Block Commands



INT

```
INT{[ ]} layer-a{&layer-a1}[ ] layer-b{&layer-b1}  
LT/LE/EQ/RANGE/SELLT/SELLE/SELGT/SELGE/SELEQ/SELNE/SELRA n1 {n2} {trapfile}  
{OUTPUT c-name l-num {d-num}} {&}
```

Description

Determines the depth to which polygons of two layers overlap. This command measures between the inside edges of polygons.

Note: The INT command will not flag violations between obtuse angled edges.

Shell environment variables

■ `setenv SPAC_LISTERR_DIGIT [ 2 | 3 ]`

Use this environment variable to set number of digits after the decimal point for EXT/ENC/WIDTH/INT log report.

Checking Method

modes

Flat, cell, hierarchical, and composite.

checks

HDRC.

Arguments

layer-a

The name of the first input layer.

layer-b

The name of the second input layer.

&*layer-a1*

Renames (conjunctive rename) the layer containing the error flags. The original layer name is then restored to the original layer. Do not put a space between the *layer-a* and &*layer-a1* names, for example, *poly&rpoly*.

Dracula Reference

Operation Block Commands

&layer-b1

Same as *&layer-a1* except that it refers to *layer-b*.

Measurement Operation

Measures between the polygons of *layer-a* that overlap polygons of *layer-b*. (Note: Overlapping is a commutative relation; that is, if A overlaps B, then B overlaps A.) Measures from inside segments of *layer-a* to inside segments of *layer-b* for segments that do not coincide.

Flags violations after comparing the measurements against the limits you specify, such as:



You must supply only one of these measurements:

LT/LE <i>n1</i>	Flags an error if two segments are spaced less than <i>n1</i> for <i>LT</i> or less than or equal to <i>n1</i> for <i>LE</i> . Does not flag an error if the segments touch.
EQ <i>n1</i>	Flags an error if two segments are spaced equal to <i>n1</i> .
RANGE <i>n1 n2</i>	Flags an error if two segments are spaced less than <i>n2</i> and greater than <i>n1</i> . If you do not want acute-angle error flags, specify the range measurement.
SELGT/SELGE <i>n1</i>	<i>SELGT</i> outputs in trapezoid format the <i>layer-a</i> polygons that the program does not flag if you specify an <i>LE</i> measurement with the same options. <i>SELGE</i> has the same function as <i>SELGT</i> except that it works for an <i>LT</i> operation. Do not use the OUTPUT statement with this option. Specify a <i>trapfile</i> name instead. This check can only be used in flat and cell modes.
SELLT/SELLE <i>n1</i>	<i>SELLT</i> and <i>SELLE</i> output in trapezoid format those <i>layer-a</i> polygons that the program flags if you specify an <i>LT n1</i> or an <i>LE n1</i> and use the same options. Specify a <i>trapfile</i> name instead of using the OUTPUT statement. This check can only be used in flat and cell modes.

Dracula Reference

Operation Block Commands

`SELEQ/SELNE n1` *SELEQ* and *SELNE* output in trapezoid format those *layer-a* polygons that the program flags if you specify an *EQ n1* or an *NE n1* and use the same options. Specify a *trapfile* name instead of using the OUTPUT statement. This check can only be used in flat and cell modes.

`SELRA n1 n2` *SELRA* outputs in trapezoid format those *layer-a* polygons that Dracula flags if you specified a *RANGE n1 n2* measurement, and you used the same options. Specify a *trapfile* name instead of using the OUTPUT statement.

trapfile

The name of the trapezoid file.

OUTPUT

Sends the results to an output cell. The output cell contains clean polygons in polygon format, resulting in reduced trapezoid construction lines and a more compact cell.

Note: The *trapfile* option and OUTPUT command are mutually exclusive. If you use any of the following options, you must also use the *trapfile* option.

R
R'
SELLT
SELLE
SELLGT
SELLGE
SELLRA
SELEQ
SELNE

c-name l-num

C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

The datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

&

Ampersand sign indicates a conjunctive rule.

Command Options

Options applicable to hierarchical Dracula are indicated with an  icon in the margin. A plus (+) in an option adds error flags. A minus (-) in an option removes error flags.

B

Flags the actual part of a violation on a segment instead of the entire segment. This option can be applied only with LT or LE modifiers. Neither RA, GT, nor GE modifiers can contain the B option.

Note: Causes round-off side effects on non-45-degree edges.

C

(-) Flags the edge pair when the edges are parallel.

C'

(-) Inverse of C option. Flags nonparallel edges.

F

Outputs error flags in trapezoid format that execute information for edge-sizing SIZE[E] operations.

F'

Outputs error flags in trapezoid format. You must specify only *trapfile* name. It is similar to R option but does not turn on the P option.

N



(-) Does not flag violations on polygons that are part of the same node. Use the ONNECT command to connect nodes before using this option. This option checks composite plane and cell plane nodal information, as well as nodal information between the composite plane and cell plane.

N'



(-) Inverse of N option. Flags violations on polygons that are part of the same node. Use the CONNECT command to connect nodes before using this option. Checks composite plane and cell plane nodal information, as well as nodal information between the composite plane and cell plane.

P

(-) Flags segments of edges that project onto each other. Two edges project if perpendicular lines from a referenced edge intersect the other edge. The referenced edge is the edge most closely aligned to the x or y axis.

Dracula Reference

Operation Block Commands

P '

(-) Inverse of *P* option. Flags edges that do not project onto each other.

R

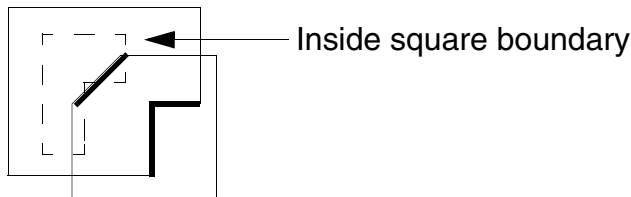
Outputs in trapezoid format the trapezoids enclosing the projection of violation flag pairs. You must specify the only output layer. Turns on the *P* option.

R '

Outputs error flags in trapezoid format so you can reuse error data. You can process the created layer (from the error flags) with logical operations. The flags are one resolution unit wide (as specified in the Description block). You cannot specify an output error cell.

S

(+) Flags violations on the two polygons if the edges of one polygon are within a “square boundary” and inside the other polygon. When you use this option, you can also specify the CORNER-CORNER and CORNER-EDGE options. The default values for *n2* and *n3* are *n1*. If you do not specify the CORNER-CORNER option, *n2* is set equal to *n1*. If you do not specify CORNER-EDGE option, *n3* is set equal to *n1*.



Use the following syntax for these options:

```
INT[S] layer1 layer2 LT n1 {CORNER-CORNER n2} {CORNER-EDGE n3}
```

Note: The CORNER-CORNER checking value, *n2*, and the CORNER-EDGE value, *n3*, cannot be greater than the normal checking value, *n1*, because *n2* and *n3* act like filters and reduce the number of errors that were initially caught by *n1*. *n2* must be greater or equal to $n1/\sqrt{2}$.

Only 90 degree angles are treated as "CORNER".

T

(+) Flags inside segments of polygons of two layers that coincide.

U 

(-) Does not flag violations on polygons that are part of the same node. Use the CONNECT command to establish nodal connectivity before using this option. This option checks only the data on the composite plane.

Dracula Reference

Operation Block Commands

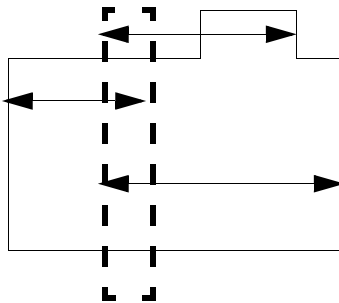
U'



Inverse of the *U* option. Flags violations on polygons that are part of the same node. Use the CONNECT command to connect nodes before using this option. Checks only data on the composite plane.

V

(+) Checks the inside edges of *layer-a* to the inside edges of all surrounding geometries of *layer-b*.



INT[V] checks the inside poly edges to the inside diffusion edges. The polygons do not need to overlap.

X

Checks the delta value in the x direction between two edges. The edges must project onto each other. Does not check non-Manhattan data.

Y

Checks the delta value in the y direction between two edges. The edges must project onto each other. Does not check non-Manhattan data.

Layer Options

O

Checks the original layer, not the layer generated by the previous operation in the conjunctive rule. Use this option only with a conjunctive rule and on a layer with error flags.

Example 1

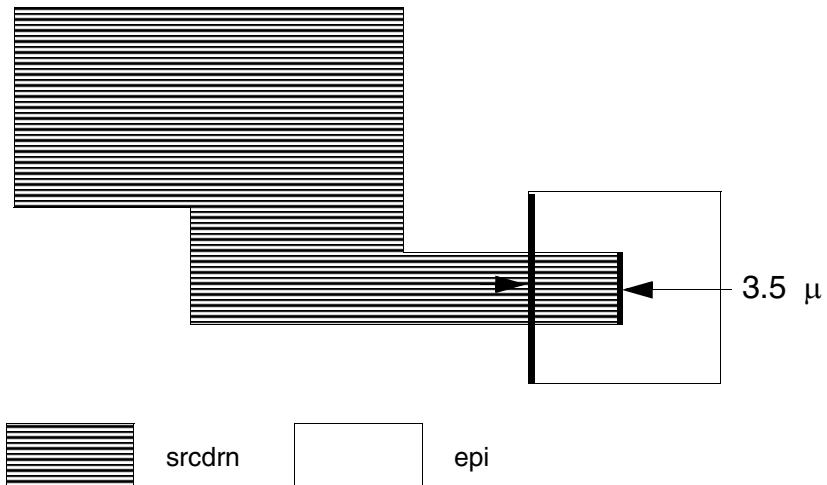
```
ENC layer1 layer2 LT 1 outl1 &  
EXT layer1 layer3 LT 1 outl2 &  
INT layer1[O] layer3 LT 2 OUTPUT err1 23
```

Dracula Reference

Operation Block Commands

Example 2

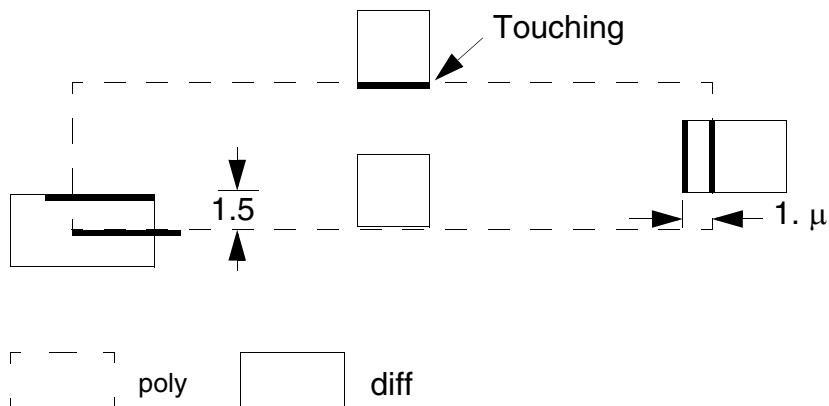
```
INT srcdrn epi LT 4 OUTPUT drc04 31
```



Example 3

This example checks the minimum intersection depth of *poly* and *diffusion* for forming capacitors. Touching is also an error.

```
INT[T] poly diff LT 2 OUTPUT drc04 32
```



Example 4

This example checks for minimum width of *metal* at the region where *metal* overlaps the *poly* by less than 2 microns.

Dracula Reference

Operation Block Commands

```
INT[P] poly metal LT 2 &  
INT[P] metal metal[0] LT 6 OUTPUT drc04 33
```

Example 5

Example 4 can be rewritten:

```
INT[P] poly metal&mpoly LT 2 &  
INT[P] &mpoly metal LT 6 OUTPUT drc04 33
```

This example also checks for minimum width of *metal* at the region where *metal* overlaps the *poly* by less than 2 microns.

Layer *metal* is renamed to *&mpoly* in the conjunctive rename operation and now *&mpoly* contains the error flags from the joined operations. Also layer *metal* reverts to *metal[O]* at this point. The new layer *&mpoly* can be used in a later DRC check. The conjunctive rename is optional but can save CPU resources.

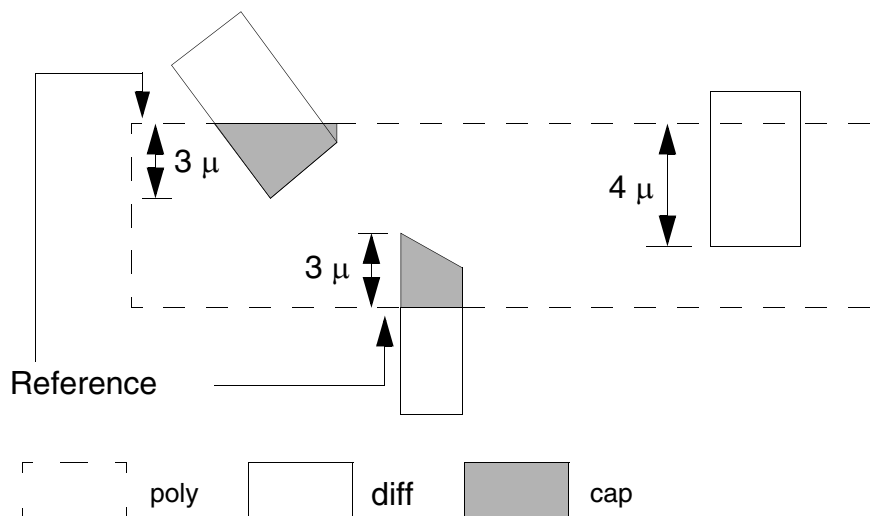
Example 6

This example checks for the minimum intersection depth of *poly* and *metal*, and depths that are less than 2 microns are output as errors. Errors generated by *poly* and *metal* and where *poly* and *metal* carry the same electrical node are not output as an error.

```
INT[N] poly metal LT 2 OUTPUT drc04 34
```

Example 7

```
INT[R] poly diff LT 4 cap
```



KEDTEXT

KEDTEXT = <kedtext_filename>

Description

Specify the text input source from a file for the "Keep Cell Text" feature in an LVS or LVL job. The format for the KEDTEXT file is the same as the HEDTEXT file.

There are two ways to specify text input source for the "Keep Cell Text" feature. One is the KTEXT command in the Input-Layer Block which reads texts from the CAD graphics database, the other is the KEDTEXT command in the Operation block which reads texts from a plain ASCII file. The two commands can exist simultaneously.

Checking Method

mode

Flat

Arguments

kedtext_filename

Specifies the name of the file containing the text input source.

Example

KEDTEXT = myKedtext.txt

or

KEDTEXT = /mypath/myKedtext.txt

LCONNECT

LCONNECT *layer* CONN/DISC *label* {*trapfile*} OUTPUT *c-name* *l-num* {*d-num*}

Description

Determines if areas at specific layers are part of a labeled node. The command outputs the areas you specify to an output cell.

Note: The output cell you specify cannot be used as the output of another command.

Arguments

layer

The layer to check. The layer must be a conductor layer you define in the CONNECT-LAYER and CONNECT commands.

CONN/DISC

CONN means that the areas that are part of the labeled node are reported. *DISC* means that the areas that are not part of the labeled node are reported.

label

Text label of the node considered for the check. The label must be assigned text in the layout database and conform to the text rules.

trapfile

Name you designate as the output file to contain composite trapezoids. Before you do any operation on a *trapfile*, you need to “OR” it.

OUTPUT

Sends the results of the operation to an output cell. The output cell you specify cannot be used as the output of another command.

c-name *l-num*

C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system.

If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

d-num

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

Dracula Reference

Operation Block Commands

Examples

This example checks all polygons on the diffusion layer to see if any are disconnected to the labeled node VCC. The output is on layer 40 of cell *difvcc40*.

```
LCONNECT diff DISC vcc OUTPUT difvcc 40
```

This example checks all polygons on the *pwell* layer to see if any are disconnected to the labeled node VSS. The output is on layer 63 of cell *flowel63*.

```
LCONNECT pwell DISC vss OUTPUT flowel 63
```

LENGTH

```
LENGTH layer-a {&layer-a1} GT/GE/ LT/LE/ RANGE n1 {n2}  
      {OUTPUT c-name l-num {d-num}} {&}
```

Description

Checks length dimensions. Use this check only as part of a conjunctive rule. When the dimensional check fails in the previous check, this command checks the dimension of the failure. It indicates a violation only when both checks fail.

Arguments

layer-a

Name of the input layer.

&*layer-a1*

Renames (conjunctive rename) the layer containing the error flags. The original layer name is then restored to the original layer. No spaces are allowed between the *layer-a* and &*layer-a1* names, for example, *poly&rpoly*.

Measurement Operation

Measures between all nonconsecutive pairs of inside edges of the polygons of the input layer. Flags violations after comparing the measurements against the limits you specify, including:

GT/GE	GT means that error segment length greater than <i>n1</i> layout units causes a violation. GE means that length greater than or equal to <i>n1</i> layout units causes a violation.
LT/LE	LT means that length less than <i>n1</i> layout units causes a violation. LE means that length less than or equal to <i>n1</i> layout units causes a violation.
RANGE <i>n1 n2</i>	RANGE means that length greater than <i>n1</i> and less than <i>n2</i> layout units causes a violation (<i>n1</i> < range < <i>n2</i>).

OUTPUT

Sends the results of the operation to an output cell.

Dracula Reference

Operation Block Commands

c-name l-num

C-name is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

Format of the output is the same as the original database.

d-num

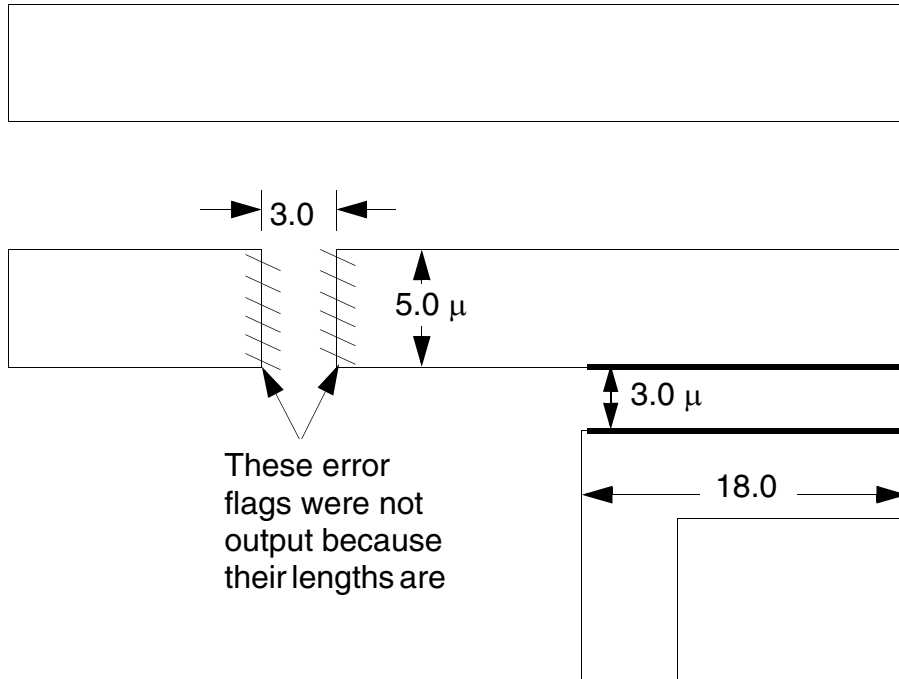
Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

&

Indicates a conjunctive rule.

Example 1

```
EXT[P] metal LT 4 &  
LENGTH metal GT 6 OUTPUT rule10 11
```



Dracula Reference

Operation Block Commands

Example 2

This example checks for metal contact (contact) of internal size 3 by 3 only. Uses the conjunctive rename operation. Layer *&cont3* can be used later in another conjunctive DRC check. Conjunctive rename is optional and can save CPU resources.

```
WIDTH contact LT 3 OUTPUT drc11 15
SIZE contact BY .2 ocont
ENC[P] contact&cont3 ocont RANGE 0.19 0.21 &
LENGTH &cont3 GT 3 OUTPUT drc11 17
```

Example 3

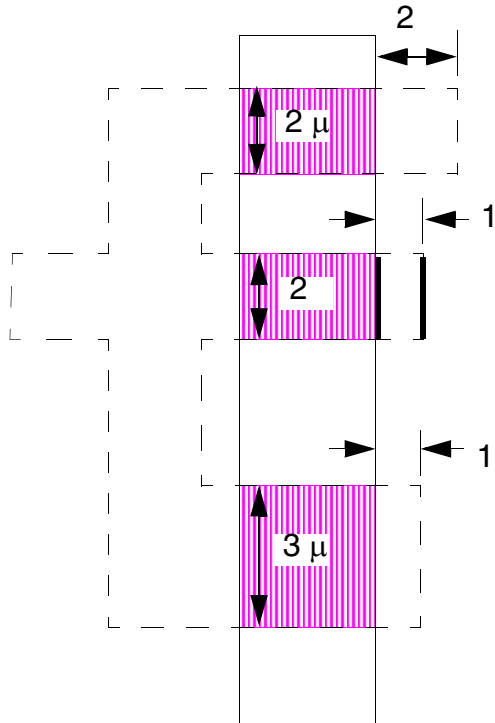
This example checks for poly overlap diffusion (gate extension) only on transistors having gate lengths 2 microns or less. Touching is also an error.

```
AND poly diff gate
ENC[T] gate diff LT .001 &
```

Dracula Reference

Operation Block Commands

```
LENGTH gate LE 2 &  
ENC[T] gate poly LT 2 OUTPUT drc11 12
```



LEXTRACT

```
LEXTRACT pset layer-a {layer-b} BY ELEMENT[type]{pname}{&}
LEXTRACT pset layer-a {layer-b} BY PARASITIC[type]{pname}{&}
LEXTRACT pset layer-a {layer-b} BY NODE {pname} {&}
LEXTRACT[E] RES layer-c layer-d BY RES{[type]} &
```

Description

Extracts geometric parameters from one or two layers of a device (BJT, CAP, DIO, MOS, or RES). Specify parameters to extract in a PARSET command in the Description block. The number of input layers you specify with the LEXTRACT command depends on the primitive parameters you specify with the PARSET command. The first layer you specify must be a device layer or a layer with the device number tagged on by a DEVTAG command. The second layer you specify can augment the parameters you are extracting from the first layer.

When you use the LEXTRACT command, you must also specify a PARSET command in the Description block.

Group parameter extractions for each device type in a conjoined group. Specify only one LEXTRACT conjoined group with the EQUATION command for each device type. For a discussion of flexible LPE parameter set extraction, refer to [Chapter 7, “Extracting Electrical Parameters \(LPE\).”](#)

For bipolar devices, LEXTRACT extracts width and length, as well as the geometric parameters (area and perimeter) from one or more layers. Group all parameter extractions for each device type in a conjoined group. Specify only one conjoined group for each device type (element [type]).

Antenna Check

In certain applications, you might want to compare the ratio of device area to the layer area driving the devices. This check, called an antenna check, is useful when you want to ensure that induced capacitance, caused by a fabrication process or a circuit operation, does not cause the drive current or the transition time to become unstable. Use the NODE argument in the LEXTRACT command as part of an antenna check. Refer also to the CALCULATE, COMPUTE, and CHKPAR commands.

To perform antenna checks, you need to do the following:

- To form the necessary connection layers, use the layer operation.

Dracula Reference

Operation Block Commands

- To extract the parameters per each node for specified layers, use the LEXTRACT BY NODE command.
- To calculate the parameter ration of each node among multiple layers, specify the CALCULATE command.
- To check and output the violation geometries, use the CHKPAR command.

To run an antenna check with HLPE

1. Run cell mode antenna check.
2. Run composite mode antenna check on the same directory.

When running composite mode, COMPUTE reads the parameter file generated by the cell mode and add the data to the composite plane for each cell instance.

See Example 1 for an sample antenna check rule. For additional information about antenna checks, see “[Writing Antenna Checks](#)” in the “Writing Rules for Dracula” chapter of the *Dracula User Guide*.

When you use more than one LEXTRACT command, you need to group them together in the Dracula rules file. You cannot place another command within a group of LEXTRACT commands. In addition, you need to group LEXTRACT, DEVTAG, LVSCCHK, or LPECHK commands together. Do not use any other command within any group of these commands.



If the LEXTRACT commands in your rules file are not grouped together, PDRACULA does not issue a warning.

Arguments

pset

Name of the parameter set PARSET to extract (specified in the Description block). The name can have up to four alphanumeric characters, the first of which must be a letter.

layer-a

The first layer from which parameters are extracted. This layer must be either the device layer or a layer with a device number tagged by the DEVTAG command.

layer-b

The optional second layer from which parameters are extracted. It is used in relation to the first layer to extract geometric parameters such as overlapped areas and perimeters.

Dracula Reference

Operation Block Commands

NODE

Extracts parameters from *layer-a* onto each node. Because of this, *layer-a* must contain nodal information. Use this keyword when checking the area ratio of layers on devices or long nodes (antenna check). For example:

```
LEXTRACT pset layer1 {layer 2} BY NODE outfile {&}
```

type

Type of device specified in the ELEMENT command. For example, BJT[NV] for a vertical *nnp* and BJT[LP] for a lateral *pnp*. Also, the type of device specified in the PARASITIC command. Elements can be BJT, CAP, DIO, MOS, and RES. Parasitics can be CAP, DIO, and RES.

pfname

Output parameter file name. Specify only one file name for each conjoined group.

&

Conjunction of several extractions into one output file.

Note: When you use conjunctive rules in LEXTRACT, only the primitive parameters that appear in the equations are extracted.

Example 1

The following is a sample rules file that checks connectivity up to *metal1* for any antenna violation before processing *metal2* and *metal3*.

```
*DESCRIPTION
PARSET ANT AREA PERI
*END
*INPUT-LAYER
NPLUS=1
NWELL=2
POLY=4
CONT=5
METAL=6 TEXT=20 ATTACH=METAL
PPLUS=7
VAPOX=8
VIA=9
METAL=10 TEXT=21 ATTACH=METAL
CONNECT-LAY= PWELL NWELL PDIFF NDIFF POLY METAL
SUBSTRATE CELLB = 12
*END
*OPERATION
NOT CELLB NWELL PWELL
NOT NPLUS POLY NDIFF
NOT PPLUS POLY PDIFF
OR PPLUS NPLUS PLUS
AND POLY PLUS GATE
```

Dracula Reference

Operation Block Commands

```
AND PDIFF PWELL PPLUG
AND NDIFF NWELL NPLUG
CONNECT NDIFF NWELL BY NPLUG
CONNECT PDIFF PWELL BY PPLUG
CONNECT METAL NDIFF BY CONT
CONNECT METAL PDIFF BY CONT
CONNECT METAL POLY BY CONT
AND NDIFF CONT NCONT
AND PDIFF CONT PCONT

OR NCONT PCONT NPCONT
SELECT[N] METAL CUT NPCONT GMET
NOT METAL GMET M1; the concerned part

AND POLY CONT PLCONT
AND PLCONT GMET PGCONT
SELECT[N] POLY CUT PGCONT GPOL
NOT POLY GPOL NPOL

AND NPOL GATE NGPOL; the concerned part
NOT NPOL GATE NFPOL; the concerned part

LEXTRACT ANT NFPOL BY NODE AFPOLY
LEXTRACT ANT NGPOL BY NODE AGATE
LEXTRACT ANT M1 BY NODE PM1

CALCULATE RATIO= (SUM.AFPOLY+SUM.PM1)/SUM.AGATE
CHKPAR PAR RATIO NGPOL GT 100 LONG1 OUTPUT PLONG 50
CHKPAR PAR RATIO NFPOL GT 100 LONG2 OUTPUT PLONG 51
CHKPAR PAR RATIO M1 GT 100 LONG3 OUTPUT PLONG 52

*END
```

Example 2

Vertical NPN:

```
*DESCRIPTION
PARSET pbip area peri ca cp ba bp ea ep
.
*END
*OPERATION
.
ELEMENT BJT[n]      emit    coll    base emit
DEVTAG BJT[n]      coll    collr
DEVTAG BJT[n]      base    baser
LEXTRACT pbip      emit    BY BJT[n] npar &
LEXTRACT pbip      collr   BY BJT[n] &
LEXTRACT pbip      baser   BY BJT[n]
```

Areas and perimeters of emitter, collector, and base are extracted and stored in the NPAR.DAT file. Because *coll* and *base* layers are not the device layer, they must be tagged with device numbers using the DEVTAG command. The device layer is *emit*. The above example handles *npn* devices with multiple emitters.

Dracula Reference

Operation Block Commands

Example 3

Lateral PNP with multiple collectors:

```
*DESCRIPTION
PARSET plat area peri ca cp ba bp ea ep
.
*END
*OPERATION
.
ELEMENT BJT      [lp]      coll      coll      base      oemit
DEVTAG BJT      [lp]      oemit      emitl
DEVTAG [L]      emit 1      emit      emitr
LEXTRACT      plat      emitr      BY      BJT[lp] bjt lp &
LEXTRACT      plat      coll      BY      BJT[lp]
```

BJT[LP] are lateral *pnp* bipolar transistors with multiple collectors around the same emitter. The *oemit* layer is true size emitter (*emit*) oversized so that it touches or overlaps the device layer (*coll*) to facilitate terminal extractions. Device numbers are tagged to the *emit1* layer and then *emit1* tags the *emit* layer and outputs to *emitr*. Areas and perimeters of *emit* and *coll* are extracted and put in the *BJTLP.DAT* file.

Example 4

Extracting width and length of a BJT:

```
*DESCRIPTION
...
MODEL=BJT[VN],VNPN BJT[LP],LPNP BJT[SP],SPNP DIO[OV],OVD
PARSET BJTS AREA OVPR L W
...
*END
*OPERATION
...
AND      EMIT BURIED      XYZ
NOT      XYZ  TRUBASE      MNO
EXT[PR]  VNPN CONT LT 6 DUMMY
ELEMENT  BJT[VN]  VNPN      BURIED      TRUBASE      METAL1
ELEMENT  BJT[LP]  CLPNP      CLPNP      BURIED      OLPNP
ELEMENT  BJT[SP]  SUBPNP      BULK      EMIT      TRUBASE
...
ELEMENT  RES  RESBODY  TRUBASE
PARAMETER RES 1
...
LEXTRACT BJTS VNPN DUMMY by BJT[VN] VNPARA &
EQUATION W=OVPR &
EQUATION L=AREA/W

LPECHK  BJTWPERCENT = 10.00 BJTLPERCENT = 11.0
LPECHK  RESWPERCENT = 5.00  RESLPERCENT = 8.0

*END
```

Dracula Reference

Operation Block Commands

To extract the length and width of a BJT, you must define a PARSET with a W and L . Then specify an LEXTRACT defining the equations and extraction layers. In this example, an EXT[PR] check is done between the device layer and a contact layer to produce an intermediate layer that can compute the periphery overlap ($OVPR$). The periphery overlap is treated as W while L is computed from $AREA$ divided by W . If no existing layer can be used together with the device layer to produce the overlap periphery you want, you might need to digitize a new layer for that purpose.

In the SPICE file generated by the LPESELECT *BJT* command in conjunction with the LEXTRACT command, the bipolar SPICE file has the following format:

```
CA = the area of the collector BP = the perimeter of the base
CP = the perimeter of the collector EA = the area of the emitter
BA = the area of the base EP = the perimeter of the emitter
```

Example 5

```
Q1 out1 in1 gnd npn $ CA=1050 CP=130 BA=500 BP=90 EA=100 EP=40
```

(Remember that \$ is a SPICE comment character.)

Example 6

```
LEXTRACT ANT GATE BY NODE PGATE
LEXTRACT ANT NGATE BY NODE PNGATE
LEXTRACT ANT M1 BY NODE PM1
LEXTRACT ANT M2 BY NODE PM2
CALCULATE RES1 = PNGATE/PGATE
CALCULATE RES2 = (PNGATE + c1 * PM1)/PGATE
CALCULATE RES3 = (PNGATE + c1 * PM1 + c2 * PM2)/PGATE
CHKPAR PAR RES1 PGATE GT200 LONG1 OUTPUT PLONG 50
CHKPAR PAR RES2 PGATE GT300 LONG2 OUTPUT NLONG 50
CHKPAR PAR RES3 PGATE GT400 LONG3 OUTPUT LONG 50
```

This example is based on one parameter being extracted. For mixed calculations, use the following:

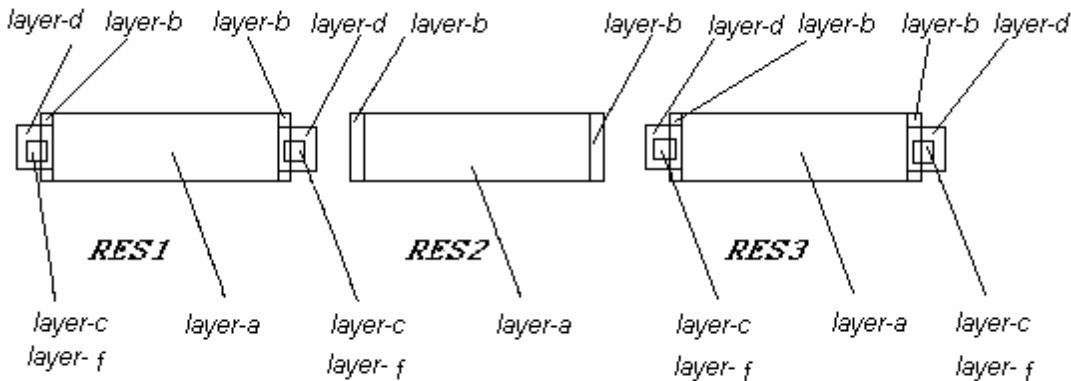
```
CALCULATE RES4 = (PNGATE.AREA + k1 * PM1.AREA + k2 * PM1.PERI)/PGATE.AREA
```

Note: The operands for the CALCULATE command are layers, not variables.

```
LEXTRACT[E] RES layer-c layer-d BY RES{[type]} &
```

Consider the rule fragment for extraction parameters of resistors with contacts (please see [Figure 13-4](#) on page 1081).

Figure 13-4 Test Pattern



Test Pattern

```
;Element Definition
ELEMENT RES{[type]} layer-a layer-b
;Tagging
DEVTAG          RES{[type]} layer-b layer-d
DEVTAG[L]        layer-d layer-f layer-c
; Resistor Body parameter calculation, and ;;; Contact Resistor parameter
calculation
LEXTRACT RES    layer-a    layer-b    BY RES{[type]}    pname &
EQUATION        parameter_1 = FORTRAN-expression &
EQUATION        parameter_2 = FORTRAN-expression &
EQUATION        parameter_3 = FORTRAN-expression depends on
                parameter_1 and parameter_2 &
LEXTRACT RES    layer-c    layer-d    BY RES{[type]}    &
EQUATION        parameter_4 = FORTRAN-expression &
EQUATION        parameter_5 = parameter_3+ parameter_4
```

If *layer-c* is empty, *parameter_4* is not calculated and *parameter_5* is not calculated for this resistor (e.g. **RES2** on [Figure 13-4](#) on page 1081) either. To avoid this issue the [E] option is used. For instance, if you use:

```
LEXTRACT[E] RES layer-c layer-d BY RES{[type]} &
EQUATION      parameter_4 = FORTRAN-expression &
EQUATION      parameter_5 = parameter_3+ parameter_4
```

In this case, *parameter_4* =0 for the empty *layer-c* case and

```
EQUATION      parameter_5 = parameter_3+ parameter_4 (for RES1 and RES3)
```

becomes

```
EQUATION      parameter_5 = parameter_3 (for RES2)
```

and all resistors are extracted.

Dracula Reference

Operation Block Commands

Example 7

```
; Element Definition
ELEMENT      RES[D]    DRES      DRTRM
; Tagging
DEVTAG      RES[D]    DRTRM      TAGCON1
DEVTAG[L]    TAGCON1  DRTRM      DRCON
; Resistor Body parameter calculation, and ;;; Contact Resistor parameter
calculation
LEXTRACT RES      DRES      DRTRM    BY RES[D]    pfname &
EQUATION      W=(W1+W2)/2 &
EQUATION      L=AREA/W &
EQUATION      RB=1000*L/W &
LEXTRACT[E] RES  DRCON      TAGCON1 BY RES[D] &
EQUATION      RC=2*25/(AREA/2) &
EQUATION      R=RB+RC
```

LINK

`LINK layer TO label`

Description

Forces a layer with node information to a label (text). Use the LINK command after establishing the circuit connection. After a layer is linked, the original node information is overwritten. Use the LINK command only to connect the substrate layer when you cannot make a connection with the CONNECT command.

Arguments

layer

Name of the layer of a previously generated layer or a layer name from the Input-Layer block.

label

Text label either from the CAD database text or from EDTEXT.

Example

`LINK sub TO gnd`

Dracula Reference

Operation Block Commands

LPECHK

```
LPECHK  {[options]}  {SUBTYPE-AS-PARAM=YES/NO} {MOS-WIDTH-SUMMARY=YES/NO}
        {WPERCENT=n1} {LPERCENT=n2} {WEFFECT=n3} {CAPAREA=n4} {CAPVAL=n5}
        {DIOAREA=n6} {DIOPERI=n7} {RESSIZE=n8} {RESVAL=n9} {W/L-PERCENT=n10}
        {MULDELW=n11} {LISTUNMATCH=n12} {EAPER=n13} {PRINTLINE=n14}
        {BJTWPERCENT=n15} {BJTLPERCENT=n16} {RESWPERCENT=n17} {RESLPERCENT=n18}
```

Description

Specifies the LPE checking option. If you specify the SCHEMATIC command in the Description block, you can exercise LPE options on layout data and schematic data. The LPECHK and LVS options are similar in function. This command must fit on one line. To use multiple options, specify more than one LPECHK command.

If you specify LPECHK in your rule file, you must define the schematic file. Otherwise, you get an error message. If you do not have a schematic, Dracula interprets this as an error. Dracula ignores the command and generates a *jxrun.com* file. You must determine the seriousness of the error flagged by Dracula and whether you want to submit the job.

Note: When you use more than one LPECHK command, they need to be grouped together in the Dracula rules file. You cannot place another command in a group of LEXTRACT commands. In addition, you need to group LEXTRACT, DEVTAG, LVSCCHK, or LPECHK commands together. Do not use any other command in any group of these commands.

Arguments

options

Can include the following values:

- B Keeps parallel BJT transistors unsmashed, but smashes MOS, LDD, resistor, capacitor, and diode devices.
- C Forms CMOS gates such as NORs, NANDs, AOIs, and OAls, except when you specify the LPECHK *x* option. The subtypes in the ELEMENT MOS command and in the SPICE/CDL netlist must be MOS[N] and MOS[P] and model = N and P, respectively, to form the CMOS gates.

Note: Inverters are formed by default except if the *x* option is turned on.

Dracula Reference

Operation Block Commands

- E Uses the information of body parameters to match parallel devices. LVSCHK uses the percentage tolerances described later (like WPERCENT, LPERCENT, CAPVAL, RESVAL, etc.) to match parallel transistors in the schematic with corresponding parallel transistors in the layout. For 4.8 or higher versions, this is the default. Otherwise, the default is to match pairs at random.
- F Filters unused MOS devices (that is, for gate arrays). The FILTER-OPTIONS command overrides this option.
- G Filters both the schematic and the layout in the same way. The FILTER-OPTIONS command overrides this option.
- K Keeps parallel transistors “unsmashed.” Default is to smash devices. *K* overrides the *B* option.
- L Same as the *C* option except AOI and OAI gates are not formed. The subtype restrictions still apply. *L* option overrides the *C* option.

Dracula Reference

Operation Block Commands

- Forms parallel and series MOS structures even if neither is connected to power or ground. By default, LVS does not form SMID or PMID structures. This option enables LVS to recognize BiCMOS structures and enables the LVS gate level matching capability. You can use “O” option with any other option except the “X” option.

LVS recognizes an SMID or PMID node as the following:

- Any pads assigned text, other than power, ground, or supply nodes
- An input (gate node) of any MOS/LDD transistors
- A conjunction point between MOS elements and LDD elements
- A conjunction point between different subtype MOS/LDD elements

LVS forms parallel/series structures:

- PUP(/SUP): A parallel or series structure with one power end and one terminal end
- PDW(/SDW): A parallel or series structure with one ground end and one terminal end
- PMID(/SMID): A parallel or series structure not connected to power or ground at either terminal

- P Checks the ELEMENT capacitor’s polarity. This option does not swap the terminal order of the ELEMENT CAP and the CDL capacitor command. For addition information, refer to the ELEMENT CAP section in this chapter. See also the Z option in LPESELECT.

- R Smashes series resistors.

Dracula Reference

Operation Block Commands

- S** Smashes MOS split-gates. The S-OPTION-CLASS command in the Description block determines whether this option smashes split gates connected to power or ground, split gates not connected to power or ground, or all split gates. The CARE-SPLIT-ORDER command in the Description block checks the input order of split gates.
- Device sizes (W and L) are combined using the usual formula for the related parallel devices. In cases where some of the input signals to the series structures are swapped, the MOS devices on one series structure are mapped to the MOS devices on the other series structure at random for the purpose of combining device sizes. Intermediate nodes of the series structures are also combined as a single node for the resulting single series structure. Parasitics (diodes, capacitors) for these nodes are lumped together.
- T** Considers substrate terminal connections as regular terminal connections during matching process. The matching process propagates the match information through the substrate connections and the subsequent instances.
- U** Reports in the discrepancy file (.LVS) only the unmatched schematic and unmatched layout devices on matched nodes (Type 4, 5, and 6 LVS errors). Device size, substrate, and subtype checks are not affected.
- X** Compares layout to schematic on the transistor level instead of on the logic-gate level.

SUBTYPE-AS-PARAM=YES/NO

If you specify YES, Dracula matches two devices with different subtypes. Dracula then checks the subtype as if it were checking a parameter.

If you specify NO, and the two devices have different subtypes, Dracula treats them as different devices and never matches them. Default is NO.

MOS-WIDTH-SUMMARY=YES/NO

If you specify YES, Dracula reports the summary of MOS devices by subtype and width in the .lvs file.

If you specify NO, Dracula uses the default report format.

Dracula Reference

Operation Block Commands

WPERCENT=*n1*

Percentage of the transistor width tolerance when compared with the schematic width specification ($0 < n1 < 100$; the default is no check).

LPERCENT=*n2*

Percentage of the transistor length tolerance when compared with the schematic length specification ($0 < n2 < 100$; the default is no check).

WEFFECT=*n3*

A user-specified value, the effective MOS channel width/length calculation. You can specify WEFFECT as a positive or negative value. Each 15-degree turn in a transistor channel causes a 1/6 fraction of WEFFECT specified times length (*L*) to be added or subtracted from a transistor default width (*W*) for a pull down device (default is 0). For a load-device, *W* remains as is and *L* is modified. See the algorithm at the end of this option list.

CAPAREA=*n4*

Percentage tolerance of the capacitor area in the LVS comparing process ($0 < n4 < 100$; the default is no check). See CAPVAL option.

CAPVAL=*n5*

Percentage tolerance of the capacitor value in the LVS comparing process ($0 < n5 < 100$; the default is no check). You can specify only CAPAREA or CAPVAL, but not both; the choice depends on the SPICE or CDL netlist specification.

DIOAREA=*n6*

Percentage tolerance of the diode area in the LVS comparing process ($0 < n6 < 100$; the default is no check).

DIOPERI=*n7*

Percentage tolerance of the diode periphery in the LVS comparing process ($0 < n7 < 100$; the default is no check).

RESSIZE=*n8*

Percentage tolerance of the number of squares in a resistor in the LVS comparing process ($0 < n8 < 100$; the default is no check). See also the RESVAL option.

RESVAL=*n9*

Percentage tolerance of resistor value in the LVS comparing process ($0 < n9 < 100$; the default is no check.). You can specify only RESSIZE or RESVAL, but not both, and the choice depends on the SPICE/CDL netlist specification. For additional discussion and illustrations, refer to [Chapter 7, “Extracting Electrical Parameters \(LPE\).”](#)

Dracula Reference

Operation Block Commands

W/L-PERCENT=*n10*

Percentage tolerance of MOS transistor width/length in the LVS comparing process ($0 < n10 < 100$; the default is no check).

MULDELW=*n11*

User-specified value compensating for multiple parallel finger MOS transistors on effective channel widths. MULDELW is usually a negative value (default is 0). For more information about parallel MOS transistors, refer to [Chapter 6, “Comparing Layouts and Schematics \(LVS\).”](#)

LISTUNMATCH=*n12*

Maximum number of unmatched schematic and layout devices to print out. ($0 \leq n12 \leq 1,000,000$; the default is 100).

EAPER=*n13*

Percentage tolerance of the bipolar emitter area in the LVS comparison ($0 < n13 < 100$; the default is no check). You must input emitter sizes of bipolar transistors from the schematic netlist onto the schematics.

PRINTLINE=*n14*

Maximum unmatched occurrences to print out. ($0 < n14 < 1,000,000$; the default is 1,000).

BJTWPERCENT=*n15*

Percentage tolerance of the BJT transistor width in the LVS comparing process. ($0 < n15 < 100$; the default is no check).

Note: You must specify PARSET and LEXTRACT commands to extract width or length.

BJTLPERCENT=*n16*

Percentage tolerance of the BJT transistor length in the LVS comparing process ($0 < n16 < 100$; the default is no check).

RESWPERCENT=*n17*

Percentage tolerance of the resistor width in the LVS comparing process ($0 < n17 < 100$; the default is no check). Both length and width must be specified if the option for smashing parallel resistors is on. The default is smash parallel resistors.

RESLPERCENT=*n18*

Percentage tolerance of the resistor length in the LVS comparing process ($0 < n18 < 100$; the default is no check). Both length and width must be specified if the option for smashing parallel resistors is on. The default is smash parallel resistors.

Dracula Reference

Operation Block Commands

Note: The value you specify, including 0, (percentage tolerance) for WPERCENT, LPERCENT, CAPAREA, CAPVAL, DIOAREA, DIOPERI, RESSIZE, RESVAL, W/L-PERCENT, and MULDELW, BJTWPERCENT, BJTLPERCENT, RESWPERCENT, RESLPERCENT, and EAPER, comparison accuracy is actually done at + 1/200th of the natural unit for example, 0.005 microns and 0.005 mils.

The following algorithm is used:

For WEFFECT = 0,

$W_{\text{actual}} = W$

$L_{\text{actual}} = A/W$

- For a pulldown device (L is unmodified)

$W_{\{\text{actual}\}} = W_{\{\text{init}\}} - (\text{WEFFECT} * I * L)$; if $W_{\{\text{init}\}}$ is greater than L

where

$W_{\{\text{actual}\}}$ = the actual width of the gate

$W_{\{\text{init}\}}$ = the initial width of the gate. This is always calculated by Dracula.

WEFFECT = a compensation coefficient you supply to Dracula to correctly calculate the actual width/length of a transistor gate.

I = the total of a device's internal angles (in units of 90 degrees)

L = the length of the gate which is equal to the total area of the gate divided by the initial width of the gate (e.g. Area/ $W_{\{\text{init}\}}$)

- For a load device (W is unmodified)

$L_{\{\text{actual}\}} = L_{\{\text{init}\}} - (\text{WEFFECT} * I * L)$; if $L_{\{\text{init}\}} \geq W$

where

$L_{\{\text{actual}\}}$ = the actual length of the gate

$L_{\{\text{init}\}}$ = the length of the gate which is equal to the total area of the gate divided by the width of the gate (e.g. Area/W)

WEFFECT = a compensation coefficient you supply to Dracula to correctly calculate the actual width/length of a transistor gate.

I = the total of a device's internal angles (in units of 90 degrees, with each internal angle rounded to the nearest 15 degrees)

W = the width of the gate. This is always calculated for you by Dracula.

Dracula Reference

Operation Block Commands

For additional discussion and illustrations, refer to [Chapter 7, “Extracting Electrical Parameters \(LPE\).”](#)

Example 1

`LPECHK[X] WPERCENT = 15 LPERCENT = 15`

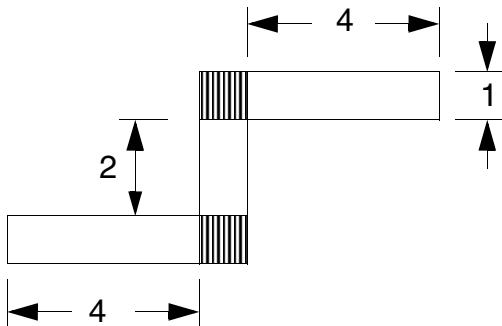
Note: If the SCHEMATIC command is specified in the Description block, a discrepancy file such as *LPEPRT.LVS* is automatically generated. Check this file before further analyzing the LPE run data.

Example 2

$I/90 = 2$ [$90 + 90 = 180$ degrees]

For a pulldown device:
WEFFECT= 0.5, $W = 12$
 $L = 1$, and $W_{\text{actual}} = 11$

For a load device:
WEFFECT= 0.5, $W = 1$
 $L = 12$, and $L_{\text{actual}} = 11$



Example 3

$I/90 = 1$ [$45 + 45 = 90$ degrees]

For a pulldown device:
WEFFECT = 0.0, $W = 16$
 $L = 3.31$, and $W_{\text{actual}} = 16$
where WEFFECT = CORNER.

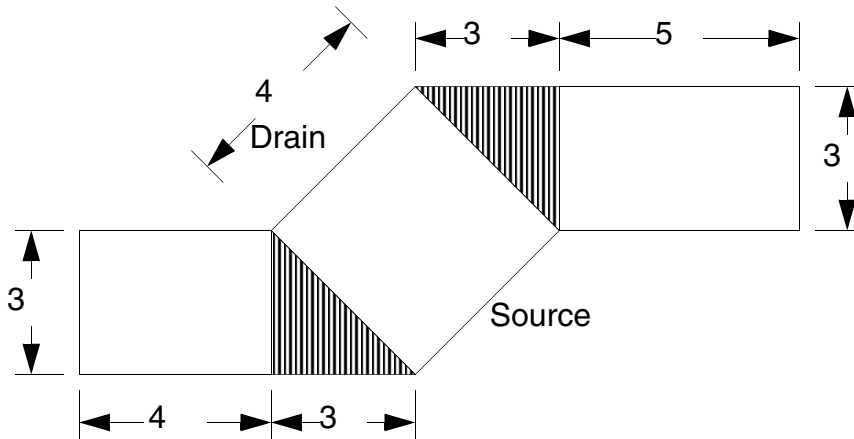
Dracula Reference

Operation Block Commands

For another pulldown device:

WEFFECT = 1.0, $W = 16$

$L = 3.31$, and $W_{\text{actual}} = 12.69$



LPeselect

```
LPeselect {[options]} element {[type]} {GE/GT val-a} {RANGE val-a val-b}  
          {CORNER = n1} {OUTPUT c-name {l-num}} {&}
```

Description

Selects a group of elements to print in CDL or SPICE format. You can use more than one LPeselect command to put more than one group of elements on the same report. You can select only parasitic capacitance when you specify your own thresholds.

LPeselect and PRE

With PRE, the LPeselect RES command indicates that the parasitic resistor elements should be output to a SPICE or CDL netlist for PARASITIC RES. It also short circuits nodes connected to parasitic resistors whose resistance is under a specified threshold. The other LPeselect element options (MOS, DIO, CAP, and BJT) report the shorted nodes as a single node.

The *S* (backannotation) option indicates that the nodes in the output file are labeled with the node names derived from the LVS check netlist used during the first connection pass. Subnodes (split nodes) are labeled with names derived from the schematic netlist, for backannotation, or derived from the node numbers obtained in the first pass, when backannotation is not specified. To get the actual subnode names, the node name or the node number is concatenated with a delimiter and a subnode number. The delimiter is a colon (:) or the delimiter specified in the PRE command SUBNODE-DELIMITER in the Description block.

For layout-only jobs (where there is no backannotation), the prefix of parasitic resistance, capacitance, and diode are RR, CC, and DD, respectively. To specify a different parasitic prefix, see the PREFIX-PARASITIC section in Chapter 11 of this manual.

Arguments

options

Can include the following:

- A Flags capacitances on all nodes, including power and ground nodes. If you do not specify this option, LPeselect does not report capacitances on power and ground nodes.

Dracula Reference

Operation Block Commands

- C Checks cross-coupling capacitance. If you do not specify cross-coupling capacitance, total loading is assumed. Use this option with the *S* option when you want to group or smash different subtypes of capacitors. Option C overwrites option A and filters out power to group parasitic capacitances.
- D When used with the *S* option, specifies that the device number assigned by LOGLVS is reported in the output SPICE file. The corresponding device name and number is stored in EXPELE.LIS in ASCII format, which is generated by LOGLVS.
- E Suppresses the output “/” between the SUBCKT I/O pins and names in the HLPE or HPRE extracted reports.
- In Dracula 4.81 and subsequent versions, you can now apply this E option to box elements too. When this E option is turned on, the “/” between I/O pins and a box element’s boxname in SPICE output is suppressed. See [“Example 6 - Example For The E Option”](#) for more information.
- F Filters options for the layout only.
- G Specifies that original ground node names, rather than the default VSS node name, are used in the output SPICE file. This option should not be used with the *N* option.
- VSS is the default for composite and cell mode.
- In multiple-ground cases, if you want only one of them as ground node in the output SPICE file, the LPE-GROUND command now lets you specify a unique ground name. The LPE-GROUND command overwrites this LPESELECT[G] command.
- H Reports the capacitance, area and perimeter of capacitors. The area and the perimeter are reported as a comment with the prefix \$AREA= and \$PERI=
- K Specifies that parallel transistors, transistors of the same source, gate, and drain (connected to the same nodes) do not smash into one. Instead, LPESELECT reports these transistors individually. For a layout-only job, it can be used to turn off the smash of the BJT elements.
- This option also specifies that parallel boxes are not smashed into one.
- L Disallows pin swapping that might result from backannotating a SPICE file based on the schematic. Use this option with the *S* option when you want to backannotate a SPICE file based on the layout with no pin swapping. For a layout-only job, use this option to retain the terminal order of CAP elements.
- M Smashes MOS split-gates.

Dracula Reference

Operation Block Commands

- N Specifies the pure SPICE format preferred.
- O Specifies to keep the original instance and device names of HLPE/HPRE or CELL mode LPE extracted results instead of renaming the extracted results *Xnnn*.
- P Checks cross-coupling capacitance as a percentage of total loading. If you do not specify cross-coupling capacitance, this command performs a check directly against *val-a*. Options *C*, *P*, and *A* are implemented only to select *CAP* (capacitors).
- R Reports the area and perimeter of parasitic capacitors. Reports the area in place of the capacitance value. Reports the perimeter as a comment with the prefix "*PERI=*."
- S Schematic node names report preferred (from LOGLVS).
- For this *S* option, the schematic ground is used in flat mode. VSS is still used as the default for composite and cell mode. You can use the *G* option or LPE-GROUND to overwrite the default ground VSS.
- Q This option works with the 'S' option. Using this option presents the pin order in the report as it is found in the netlist .SUBCKT statement.
- T Outputs cross-coupling capacitance and does not filter out parasitics between multiple power or ground busses (for example, analog to digital power busses). Parasitic capacitance with both nodes at a power or ground node is not filtered out. Option *T* is a superset of option *C* and cannot be used with *A* or *C*.
- U Does not comment out parasitic CAP and RES whose terminals are connected to power and ground nodes or power and ground subnodes in the SPICE output. This option takes effect whether back annotation is on or off.
- V Specifies the expanded name of cell instances without the prefix *X*.

Dracula Reference

Operation Block Commands

- W** Reports L (length) and W (width) of bipolar transistor (BJT) devices and resistance.
- In the Dracula versions prior to 4.81, this *W* option supported the L/W of only resistors.
- In Dracula 4.81 and subsequent versions, this *W* option in the LPESELECT command supports the L/W of not just resistors, but also the L/W (width) of bipolar transistor (BJT) devices on LVS and prints that W/L into SPICE netlists. For example,
- ```
LPESELECT [W] BJT
```
- Note:** For BJT devices, Dracula needs to specify this L/W using the L<sub>EXTRACT</sub> command in your rule file.
- X** Does not print out commented devices and cell instances. If you do not specify this option, Dracula outputs commented devices and cell instances.
- Y** Reports the x/y location of all element types in the LPESELECT listing file.



## Dracula Reference

### Operation Block Commands

---

**Z** Checks the ELEMENT capacitor polarity. This option does not swap the terminal order of the ELEMENT CAP and the CDL capacitor command.

This is a new option for LPESELECT in Dracula 4.81 and functions the same way as the P option on LPECHK or LVSCHK.

By default, the Dracula tool smashes parallel devices in LVS and LPE operations. But, you can prevent the smashing of capacitors with different polarity by using the [P] option of the LPECHK or LVSCHK commands.

When there is just the LPESELECT command in the rule file (and you are not using LPECHK or LVSCHK commands), you need to prevent the smashing of capacitors with different polarity. In such a case, the Z option of the LPESELECT command helps you check the polarity of capacitors.

If in the rule file, the LPECHK or LVSCHK exists with LPESELECT[Z], PDRACULA produces the following warning messages (depending on the commands that you are using):

■ **LVSCHK[P] + LPESELECT**

```
WARNING: LVSCHK[P] capacitors polarity option has been extended
to LPE operations
```

■ **LVSCHK + LPESELECT[Z]**

```
WARNING: LPESELECT[Z] capacitors polarity option has been
extended to LVS operations
```

■ **LPECHK+LPESELECT[Z]**

```
WARNING: LPESELECT[Z] capacitors polarity option has been
extended to LPE operations.
```

**Note:** The [Z] option takes the CAP element type only. When using with other elements, PDRACULA produces the following WARNING message

```
LPESELECT[Z] option is valid for CAP element only
```

See “Example 7 - First Example With the Z option in LPESELECT” and “Example 8 - Second Example With the Z option in LPESELECT” in this section for more information.

#### *element*

One of the device types listed below. If you do not include *element*, LPESELECT selects all device types.

MOS          MOS devices

DIO          Diode devices

## Dracula Reference

### Operation Block Commands

---

|     |                     |
|-----|---------------------|
| CAP | Capacitor devices   |
| BJT | Bipolar devices     |
| RES | Resistor devices    |
| LDD | Lightly doped drain |

For more information about ELEMENT MOS/BJT/RES, refer to [Chapter 7, “Extracting Electrical Parameters \(LPE\).”](#)

#### *type*

The type code you specify in the *element* device command. If you do not specify a type, this command invokes all element types.

To group or smash different subtypes of capacitors, specify *type* as [*new-subtype:subtype+subtype+...*]. This syntax sums the capacitance values of multiple subtypes into a new subtype. The SPICE output file then contains all parasitic metal caps, all poly capacitors, and so forth, smashed into a new group subtype. The subtypes can have different parset numbers and must be defined by PARASITIC commands.

#### GE/GT

The *GE* and *GT* options work for parasitic devices only.

*GE*      element parameters must be greater than or equal to *val-a*.

*GT*      element parameters must be greater than *val-a*.

If you do not specify this field during a conjunct, the default relational condition is *GE 0*.

**Note:** You can select only parasitic capacitance when you specify your own thresholds.

#### RANGE

Element parameters, which must be greater than or equal to *val-a*, and less than or equal to *val-b*.

#### *value-a/value-b*

Specifies *value-a* or *value-b*, which can be a floating point number (maximum of seven digits) or a floating-point number followed by an integer exponent (maximum two digits). When you specify floating-point numbers, the leading zeros are not counted in the seven-character restriction. The preprocessor rounds the value at the seventh digit. In scientific notation, the limit is 16 characters truncated at the 17th character.

## Dracula Reference

### Operation Block Commands

---

`CORNER = n1`

The effective MOS channel width/length calculation. You can specify *CORNER* as a positive or negative value. For a pulldown device (default = 0), for each 15-degree turn in a transistor channel, a 1/6 fraction of *CORNER* specified times length (*L*) is added or subtracted from a transistor default width (*W*). For a load-device, *W* is not changed and *L* is modified.

*CORNER* uses the same algorithm as *WEFFECT* in LPECHK.

`OUTPUT`

Outputs SPICE listing to a file.

`c-name l-num`

*C-name* and *l-num* are concatenated to make up the file name and the subcircuit name of the SPICE listing. *C-name* can have six alphanumeric characters. If *l-num* is not specified, only *c-name* is used. For example, if *c-name* = *CKTNAM* and *l-num* = 10, the output file name is *CKTNAM10* and the subcircuit name is the same in the SPICE listing.

`&`

Indicates a conjunctive rule. You can join LPESELECT only with other LPESELECT commands.

### Example 1

This example shows how you can obtain layout node number information.

```
LPESELECT mos &
LPESELECT dio &
LPESELECT cap GE 0.01 OUTPUT cktnam 10
```

### Example 2

This example uses the *S* option to obtain schematic node name information. This example assumes a SCHEMATIC is provided and the LPECHK command is invoked.

```
LPECHK
LPESELECT [SY] MOS[N] CORNER = 0.5 &
LPESELECT [SY] MOS[P] CORNER = 0.5 &
LPESELECT[S] DIO[N] &
LPESELECT[S] DIO[P] &
LPESELECT[S] CAP[A] GT 0.0 &
LPESELECT[S] CAP[B] GT 0.0 &
LPESELECT[S] CAP[C] GT 0.0 OUTPUT testck
```

The output file name is *testck.dat*. It contains listings of the following:

## Dracula Reference

### Operation Block Commands

---

- MOS type *N* and type *P* are grouped separately with their effective transistor sizes calculated and their x/y coordinates indicated.
- DIODE type *N* and type *P* are grouped separately.
- Parasitic capacitor types *A*, *B*, and *C* are grouped separately.

#### Example 3

In the following example, parasitic capacitor types *A*, *B*, *C* are grouped as a new group subtype *D*.

```
LPSELECT[CS] CAP[D:A+B+C] GT 0
```

#### Example 4

```
LPSELECT[SO] MOS &
LPSELECT[S] CAP[A] GT 0.0 OUTPUT SPICE
```

#### Example 5

```
LPSELECT[S] RES GE 500 &
LPSELECT[S] MOS &
LPSELECT[S] DIO &
LPSELECT[S] CAP OUTPUT SPICE
```

Any parasitic resistors with values under 500 are shorted. Units are specified in the command under the Description block.

#### Example 6 - Example For The E Option

```
LPSELECT[E] IND OUTPUT SPICE
```

SPICE output:

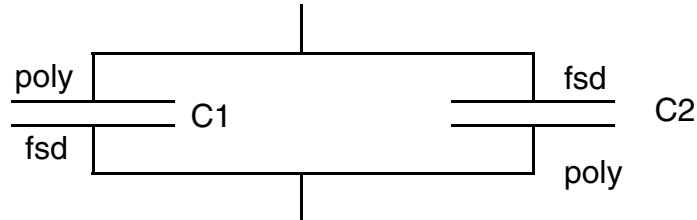
```
XB1009 5 OUT IND NR=2.50 W=15.00U S=1.50U RAD=60.00
XB1010 18 20 IND NR=4.50 W=15.00U S=1.50U RAD=30.00
=====
```

|  
---> pin list

Without the E option, the output would be as follows:

```
XB1009 5 OUT / IND NR=2.50 W=15.00U S=1.50U RAD=60.00
XB1010 18 20 / IND NR=4.50 W=15.00U S=1.50U RAD=30.00
```

### Example 7 - First Example With the Z option in LPESELECT



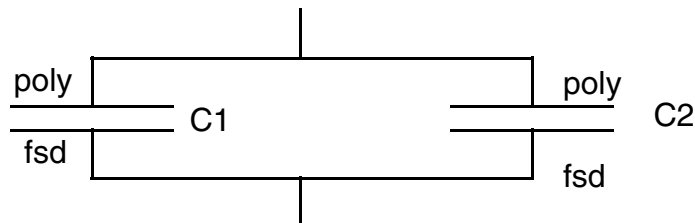
```
LPESELECT CAP OUTPUT SPICE
LPESELECT CAP[A] GT 0.0 OUTPUT SPICE
```

C1 and C2 capacitors are smashed. Dracula produces SPICE parameters of smashed capacitors C1+C2.

```
LPESELECT[Z] CAP OUTPUT SPICE
LPESELECT[Z] CAP[A] GT 0.0 OUTPUT SPICE
LPECHK[P] ... LPESELECT CAP OUTPUT SPICE
LPECHK[P] ... LPESELECT CAP[A] GT 0.0 OUTPUT SPICE
LVSCHK[P] ...LPESELECT CAP OUTPUT SPICE
LVSCHK[P] ...LPESELECT CAP[A] GT 0.0 OUTPUT SPICE
```

C1 and C2 capacitors are not smashed. Dracula produce SPICE parameters of C1 and C2 capacitors separately.

### Example 8 - Second Example With the Z option in LPESELECT



## **Dracula Reference**

### Operation Block Commands

---

C1 and C2 capacitors are smashed. Dracula produces SPICE parameters of smashed capacitors C1+C2

These options, the LPESELECT [Z] option, the LPECHK [P] option, and the LVCHK[P] have no effect on smashing.

## Dracula Reference

### Operation Block Commands

---

## LVSCHK

```
LVSCHK {[options]} {SUBTYPE-AS-PARAM=YES/NO/BCDLMR} {MOS-WIDTH-SUMMARY=YES/
NO} {MOSCAP-AREA = n0} {PRINTLINE = n1} {WPERCENT = n2} {LPERCENT =
n3} {WEFFECT = n4} {CAPAREA=n5} {CAPVAL=n6} {DIOAREA=n7} {DIOPERI=n8}
{RESSIZE=n9} {RESVAL=n10} {W/L-PERCENT=n11} {MULDELW=n12}
{LISTUNMATCH=n13} {EAPER=n14} {BJTWPERCENT=n15} {BJTLPERCENT=n16}
{RESWPERCENT=n17} {RESLPERCENT=n18} {LDDWEFFECT=n19} {SMASH=[CDMRQ]}
```

### Description

Extracts the layout database for a consistency check between layout and schematic. This command must fit on one line. To use multiple options, specify more than one LVSCHK command.

If you specify LVSCHK in your rule file, you must define the schematic file. Otherwise, you get an error message. If you do not have a schematic, Dracula interprets this as an error. Dracula ignores the command and generates a `jxrun.com` file. You must determine the seriousness of the error flagged by Dracula and whether you want to submit the job.

The `DRAC_MAX_AMB_PARA_GRP` variable can be set as the node count limit for ambiguous group matching. The default limit is 1000. A value less than or equal to zero will switch off the algorithm completely. If the variable is set, the following message will appear:

`*/L* USING ENV. VARIABLE DRAC_MAX_AMB_PARA_GRP= n.`

**Note:** When you use more than one LVSCHK command, you must group them together in the Dracula rules file. Also, you cannot place another command in a group of LVSCHK commands. In addition, you need to group LEXTRACT, DEVTAG, LVSCHK, or LPECHK commands. Do not use any other command in any group of these commands.

### Arguments

*options*

Can include the following:

- A        Smashes series capacitors.
- B        Keeps parallel BJT transistors unsmashed, but smashes MOS, LDD, resistor, capacitor, and diode devices.

- C Forms CMOS gates such as NORs, NANDs, AOIs, and OAIs except when you specify the LV<sub>S</sub>CHK *x* option. The subtypes in the ELEMENT MOS command and in the SPICE/CDL netlist must be MOS[N] and MOS[P] and model = N and P, respectively, to form the CMOS gates.

**Note:**

- ❑ The C option forms inverters with one-character subtypes.
- ❑ The CMOS-NTYPE and CMOS-PTYPE commands form inverters with two-character subtypes.

See CMOS-NTYPE and CMOS-PTYPE for more information.

- E and Controls the matching of parallel devices.  
E'

For ambiguous groups of devices, Dracula has to determine which schematic should match layout. E is a scheme to start from the device and E' is scheme to start from the node. It is best to run LV<sub>S</sub>CHK twice, once with E and once with E', (or in reverse sequence) then compare the results. Only real errors will appear in both runs.

**Note:** The E and E' options cannot be used during the same run.

E and E' use the percentage tolerances described later (for example, WPERCENT, LPERCENT, CAPVAL, RESVAL, and so on) to match parallel devices in the schematic with corresponding parallel devices in the layout based on the information of body parameters.

- If it finds percentage tolerances, Dracula 4.8 (or higher) turns on the E option automatically, if it is not already set, and notifies the user with a NOTE message.
- If E' is set, LV<sub>S</sub>CHK uses the E' scheme even if percentage tolerances are present. Dracula does not turn on the E option automatically if E' is set.
- If percentage tolerances are not set, LV<sub>S</sub>CHK matches pairs at random.



## Dracula Reference

### Operation Block Commands

---

- F** Filters the unused MOS devices, for example, gate arrays. Allows the layout and schematic to be filtered differently. The `FILTER-OPTIONS` command overrides this option.
- G** Filters both the schematic and the layout in the same way. The `FILTER-OPTIONS` command overrides this option.
- H** Merges all SUP and SDW serial MOS devices with a common gate and a common substrate into a single MOS device. If `WPERCENT` is specified then all MOS devices must have a same width. The width will be written onto a single merged MOS device. The length value will be equal to the sum of the length of all merged MOS devices. If MOS devices do not have equal width then the devices are not merged.
- Note:** This is used with the `O` option.
- J** Merges all SMID serial MOS devices with a common gate and a common substrate into a single MOS device. If `WPERCENT` is specified then all MOS devices must have a same width. The width will be written onto a single merged MOS device. The length value will be equal to the sum of the length of all merged MOS devices. If MOS devices do not have equal width then the devices are not merged.
- Note:** This is used with the `O` option.
- K** Keeps parallel devices “unsmashed.” Default is to smash devices. The `K` option overrides the `B` option. In Dracula 4.81 and subsequent versions, a qualifier has been added to the `K` option such that it will not smash parallel devices with the exception of those devices you specify in the qualifier.
- When the `S` and `K` options are turned on at the same time, the `S` option will outweigh the `K` option, which means that the `K` option will have no effect and PDRACULA will output warning message to notify users of this conflict situation. However, the `jxrun.com` file will still be generated.
- L** Same as the `C` option except does not form AOI and OAI gates. Subtype restrictions still apply. `L` option overrides the `C` option.

## Dracula Reference

### Operation Block Commands

---

- M** Dracula recognizes inverters when the inverter has been created by shorting the gates of a NAND or a NOR structure together.
- N** Check gate level substrate terminal connections. Without this option, only transistor level substrate connections are considered as regular terminals during the matching process if **T** option is set. Set both **N** and **T** options when you need to form pseudo-gate structures and match the substrate nodes. This option works correctly only if substrate terminal connections are connected to global nodes.
- O** Forms parallel and series MOS structures even if neither is connected to power or ground. By default, LVS does not form SMID or PMID structures. This option enables LVS to recognize BiCMOS structures and enables the LVS gate-level matching capability. You can use the *O* option with any other option except the *X* option.

LVS recognizes an SMID or PMID node as the following:

- Any pads assigned text, other than power, ground, or supply nodes
- An input (gate node) of any MOS/LDD transistors
- A conjunction point between MOS elements and LDD elements
- A conjunction point between different subtype MOS/LDD elements

LVS forms parallel/series structures:

- PUP(/SUP): A parallel or series structure with one power end and one terminal end
- PDW(/SDW): A parallel or series structure with one ground end and one terminal end
- PMID(/SMID): A parallel or series structure not connected to power or ground at either terminal

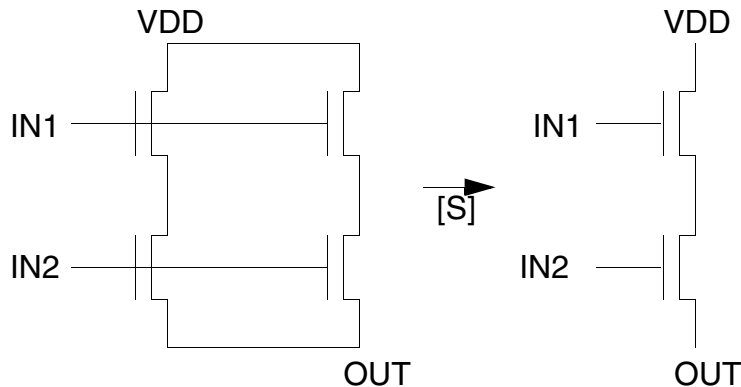
## Dracula Reference

### Operation Block Commands

---

- P Checks the ELEMENT capacitor's polarity. This option does not swap the terminal order of the ELEMENT CAP and the CDL capacitor command. See the [ELEMENT CAP](#) command for additional information. See also the [Z option in LPESELECT](#).
- R Smashes series resistors.
- S Smashes MOS split-gates. The S-OPTION-CLASS command in the Description block determines whether this option smashes split gates connected to power or ground, split gates not connected to power or ground, or all split gates. The CARE-SPLIT-ORDER command in the Description block checks the input order of split gates.

This illustration shows a split gate SUPI device smashed to a single SUP device. This is how the S option works by default.



- T Considers substrate terminal connections as regular terminal connections during the matching process. The matching process propagates the match information through the substrate connections and the subsequent instances. This option cannot trace the substrate nodes of pseudo-gate structures. To match the substrate, do not form a pseudo-gate.

## Dracula Reference

### Operation Block Commands

---

- U Reports in the discrepancy file (.LVS) only the unmatched schematic and unmatched layout devices on matched nodes (Type 4, 5, and 6 LVS errors). Device size, substrate, and subtype checks are not affected.
- X Carries out the comparison at the transistor level (that is, no swap allowed).
- Z Filters out devices with no general paths to any nonpower/ground pad.

`SUBTYPE-AS-PARAM=YES/NO/BCDLMR/X`

Determines whether Dracula matches devices with subtypes and treats the subtypes as parameters.

If you specify YES, Dracula matches all devices with different subtypes. Dracula then checks the subtypes as if it were checking parameters.

If you specify NO, and the MOS, LDD, and BJT devices have different subtypes, Dracula treats them as different devices and does not match them with subtypes. However, Dracula matches all other devices, such as RES, CAP, and DIO, with subtypes and treats the subtypes as parameters, as if `SUBTYPE-AS-PARAM` were set to YES. The default is NO.

If you specify B, C, D, L, M, or R, Dracula treats the subtypes of the specified devices as parameters. The devices that you can specify are the following:

B: BJT

C: capacitor

D: diode

L: lightly doped drain

M: MOS

R: resistor

For example, `SUBTYPE-AS-PARAM=BD` causes Dracula to treat only the subtypes of BJTs and diodes as parameters.

If you specify X, Dracula does not match any device subtype as a parameter.

`MOS-WIDTH-SUMMARY=YES/NO`

If you specify YES, Dracula reports the summary of MOS devices by subtype and width in the .lvs file.

## Dracula Reference

### Operation Block Commands

---

If you specify NO, Dracula uses the default report format.

MOSCAP-AREA=*n0*

Checks the capacitance effect in a MOSCAP device area. Because a MOSCAP device has characteristics similar to a capacitor, this check measures the capacitance effect. By default, Dracula checks only the width and length of a MOSCAP. With this option turned on, if the W/L values do not match, the area is checked for a capacitance effect. If this option is not turned on, Dracula verifies a MOSCAP device as a MOS device. It does not measure the capacitance effect of other types of devices.

PRINTLINE=*n1*

The maximum unmatched occurrences to print out ( $0 < n1 < 1,000,000$ ; default = 1,000).

WPERCENT=*n2*

The percentage tolerance of the MOS transistor width in the LVS comparing process ( $0 < n2 < 100$ ; default = no check).

LPERCENT=*n3*

The percentage tolerance of the MOS transistor length in the LVS comparing process ( $0 < n3 < 100$ ; default = no check).

WEFFECT=*n4*

Effective MOS channel width/length calculation you specify. You can specify *WEFFECT* as a positive or negative value. Each 15 degree turn in a transistor channel causes a 1/6 fraction of *WEFFECT* specified times length (*L*) to be added or subtracted from a transistor default width (*W*) for a pull-down device (default = 0). For a load-device, *W* remains as is and *L* is modified.

The following algorithm is used:

## Dracula Reference

### Operation Block Commands

---

For  $WEFFECT = 0$ ,

$$W_{actual} = W$$

$$L_{actual} = A/W$$

- For a pulldown device (L is unmodified)

$$W_{\{actual\}} = W_{\{init\}} - (WEFFECT * I * L) \quad ; \text{ if } W_{\{init\}} \text{ is greater than } L$$

where

$W_{\{actual\}}$  = the actual width of the gate

$W_{\{init\}}$  = the initial width of the gate. This is always calculated by Dracula.

$WEFFECT$  = a compensation coefficient you supply to Dracula to correctly calculate the actual width/length of a transistor gate.

$I$  = the total of a device's internal angles (in units of 90 degrees)

$L$  = the length of the gate which is equal to the total area of the gate divided by the initial width of the gate (e.g.  $Area/W_{\{init\}}$ )

- For a load device (W is unmodified)

$$L_{\{actual\}} = L_{\{init\}} - (WEFFECT * I * L) \quad ; \text{ if } L_{\{init\}} \geq W$$

where

$L_{\{actual\}}$  = the actual length of the gate

$L_{\{init\}}$  = the length of the gate, which is equal to the total area of the gate divided by the width of the gate (e.g.  $Area/W$ )

$WEFFECT$  = a compensation coefficient you supply to Dracula to correctly calculate the actual width/length of a transistor gate.

$I$  = the total of a device's internal angles (in units of 90 degrees, with each internal angle rounded to the nearest 15 degrees)

$W$  = the width of the gate. This is always calculated for you by Dracula.

For additional discussion and illustrations, see Chapter 7, "Extracting Electrical Parameters (LPE)."

## Dracula Reference

### Operation Block Commands

---

CAPAREA=*n*5

Percentage tolerance of the capacitor area in the LVS comparing process ( $0 < n5 < 100$ ; the default is no check.). See CAPVAL option.

CAPVAL=*n*6

Percentage tolerance of the capacitor value in the LVS comparing process ( $0 < n6 < 100$ ; the default is no check.). Only CAPAREA or CAPVAL can be specified, but not both, and the choice depends on the SPICE/CDL netlist specification.

DIOAREA=*n*7

Percentage tolerance of the diode area in the LVS comparing process ( $0 < n7 < 100$ ; the default is no check.).

DIOPERI=*n*8

Percentage tolerance of the diode periphery in the LVS comparing process ( $0 < n8 < 100$ ; the default is no check.).

RESIZE=*n*9

Percentage tolerance of the number of squares in a resistor in the LVS comparing process ( $0 < n9 < 100$ ; the default is no check.). See RESVAL option. \*.RESIZE doesn't compare the width/length of resistors. It allows these values to be put into the LVSLOGIC.DAT file from the input CDL/Spice netlist.

RESVAL=*n*10

Percentage tolerance of resistor value in the LVS comparing process ( $0 < n10 < 100$ ; the default is no check.). You can specify only RESIZE or RESVAL, but not both, and the choice depends on the SPICE/CDL netlist specification. For additional discussion and illustrations, see Chapter 7, "Extracting Electrical Parameters." \*.RESVAL allows the resistance values to be put into the LVSLOGIC.DAT file from the netlist.

**Note:** When using LOGLVS it is necessary to specify either \*.RESVAL or \*.RESIZE, but not both in the CDL/Spice file. (This is already mentioned in Chapter 5, under the \*.BIPOLAR section. Similarly you can specify either \*.CAPVAL or \*.CAPAREA, but not both in the CDL/Spice file).

If you are running CDLOUT to generate the CDL netlist make sure that you have properly set the values for the Check Resistors and Check Capacitors fields in the form to have the appropriate parameter information sent to the netlist file for resistors and/or capacitors respectively.

In a typical resistor CDL card, you have the following fields:

## Dracula Reference

### Operation Block Commands

---

```
Rxxx term1 term2 {res} {$SUB=substrate} {$[mname] /
$.MODEL=mname} {$W=width} {$L=length}
```

If you want to check the resistor's value, put the value of this resistor in the {res} field. When running LOGLVS make sure you have the \*.RESVAL command in your CDL/SPICE netlist. When running PDRACULA make sure you have the RESVAL option for the LVSCCHK command in your rule file. This will compare the two resistor values (in ohms).

If you want to check the resistor's size (number of squares), put the number of squares of the resistor in the {res} field. When running LOGLVS you should have the \*.RESSIZE command in your CDL/SPICE netlist. When running PDRACULA you should have the RESSIZE option for the LVSCCHK command in your rule file. This will compare the two resistor sizes (in number of squares).

The \$W and \$L have nothing to do with calculating the number of squares. They are strictly used for comparing the tolerances of the width and length of the resistor using the following options on the LVSCCHK command, RESWPERCENT and RESLPERCENT respectively.

W/L-PERCENT=*n11*

Percentage tolerance of the MOS transistor width/length in the LVS comparing process ( $0 < n11 < 100$ ; the default is no check).

MULDELW=*n12*

Value you specify compensating for multiple parallel finger MOS transistors on effective channel widths. *MULDELW* is usually a negative value. The default is 0. For information about parallel MOS transistors, refer to [Chapter 6, "Comparing Layouts and Schematics \(LVS\)."](#)

LISTUNMATCH=*n13*

Maximum number of unmatched schematic and layout devices to print out ( $0 \leq n13 \leq 1,000,000$ ; the default is 100).

EAPER=*n14*

Percentage tolerance of the bipolar emitter area in the LVS comparison ( $0 < n14 < 100$ ; the default is no check). You must input emitter sizes of bipolar transistors from the schematic netlist onto the schematics.

BJTWPERCENT=*n15*

Percentage tolerance of the BJT transistor width in the LVS comparing process ( $0 < n15 < 100$ ; the default is no check).



## Dracula Reference

### Operation Block Commands

---

**Note:** You must specify PARSET and LEXTRACT commands to extract width or length.

BJTLPERCENT=*n16*

Percentage tolerance of the BJT transistor length in the LVS comparing process ( $0 < n16 < 100$ ; the default is no check).

RESWPERCENT=*n17*

Percentage tolerance of the resistor width in the LVS comparing process ( $0 < n17 < 100$ ; the default is no check). Both length and width must be specified if the option for smashing parallel resistors is on. The default is smash parallel resistors. If either length or width is not specified, all width/length of resistors are 0, which means unspecified.

RESLPERCENT=*n18*

Percentage tolerance of the resistor length in the LVS comparing process ( $0 < n18 < 100$ ; the default is no check). Both length and width must be specified if the option for smashing parallel resistors is on. The default is smash parallel resistors.

LDDWEFFECT=*n19*

Add this argument, LDDWEFFECT for effective LDD channel width/length calculation. Same usage as WEFFECT (see the WEFFECT argument above) except that LDDWEFFECT is for LDD calculation whereas WEFFECT is for MOS calculation.

Example:

```
LVSCHK LDDWEFFECT = 1.0
```

**Note:** For any value specified except 0 (percentage tolerance) for WPERCENT, LPERCENT, CAPAREA, CAPVAL, DIOAREA, DIOPERI, RESSIZE, RESVAL, W/LPERCENT, MULDELW, BJTWPERCENT, BJTLPERCENT, RESWPERCENT, RESLPERCENT, and EAPER options, comparison accuracy is actually done at + 1/200th of the natural unit; that is, 0.005 microns or 0.005 mils. For 0 value comparison accuracy is actually done at 0.0001%.

{ SMASH= [ CDMRQ ]

Add this argument to set or clear bytes. When LVSCHK[SK] is used, *S* overrides *K*, and thus all CDMRQ gets cleared. When you use the syntax

```
LVSCHK [K] SMASH= [CDMRQ]
```

where

*C* = capacitors

*D* = diodes

*M* = MOS transistors

*R* = resistors

*Q* = BJT bipolar devices

## Dracula Reference

### Operation Block Commands

---

If you use the *K* option and specify any of the above devices in the SMASH qualifier all devices with the exception of those listed in the qualifier will not be smashed.

See the [“Example 2 - Using the K option and the SMASH Qualifier”](#) in this section for more information.

## Examples

### Example 1

```
LVSCHK[FX] PRINTLINE = 3000
```

Checks a gate array circuit at its transistor level. The printout lists up to 3,000 unmatched occurrences. The device width and length are not checked. *W* and *L* are compared only when *WPERCENT* and *LPERCENT* are mentioned.

```
LVSCHK[S] WPERCENT = 10 LPERCENT = 5 WEFFECT = .5
```

Checks a circuit with pseudo gates being formed. All split-gates in the schematic and layout are smashed to a single SUP or SDW device for LVS checking. MOS transistor width and length are checked to  $\pm 10\%$  and  $\pm 5\%$ , respectively. LVS calculates the device's *W* and *L* values as before and only modifies the device's *W* value according to the formula above for a pull-down device. For every 90-degree turn and every 45-degree turn, *W* is reduced (subtracted) by  $(.5)*(L)$  and  $(.25)*(L)$ , respectively. The *L* value remains the same. If a load device is encountered, *L* is modified and the *W* value remains the same.

```
LVSCHK WPERCENT=5 LPERCENT=2 RESVAL=5 DIOAREA=10
LVSCHK DIOPERI=10 CAPVAL=5
```

This compares layout database device sizes to the netlist device sizes. Comparison of MOS width is at 5%, MOS length at 2%, resistor values at 5%, diode area at 10%, diode perimeter at 10%, and capacitor value at 5%. You can repeat the LVSCHK command to include the desired check options, and their tolerances are +/- . To compare resistor, diode, and capacitor values, the appropriate options must be turned on in the input SPICE/CDL netlists.

```
LVSCHK[P] W/L-PERCENT=1 LISTUNMATCH=500
```

This compares the layout database device size *W/L* ratios to the netlist device size *W/L* ratios. Comparison of MOS *W/L* ratios is at 1%. The capacitor polarity is checked such that ELEMENT CAP terminals are not allowed to swap with the CDL capacitor terminals. The printout lists up to 500 unmatched schematic and layout devices.

**Note:** To use the above device size checks, you must understand the device layer concept for the ELEMENT commands. It is the device layer that carries the device dimension, such as NGATE and PGATE layers for CMOS process.

## Dracula Reference

### Operation Block Commands

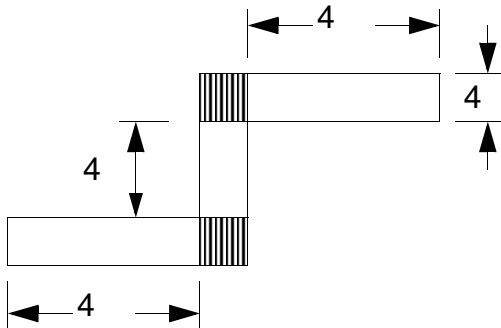
---

To do device size checks of CAP and RES, specify the ELEMENT CAP and ELEMENT RES commands along with the PARAMETER CAP and PARAMETER RES commands. There is no PARAMETER DIO command.

$I/90 = 2$  [ $90 + 90 = 180$  degrees]

For a pulldown device:  
WEFFECT= 0.5,  $W = 12$   
 $L = 1$ , and  $W_{\text{actual}} = 11$

For a load device:  
WEFFECT= 0.5,  $W = 1$   
 $L = 12$ , and  $L_{\text{actual}} = 11$



$I/90 = 1$  [ $45 + 45 = 90$  degrees]

For a pulldown device:  
WEFFECT = 0.0,  $W = 16$   
 $L = 3.31$ , and  $W_{\text{actual}} = 16$   
where WEFFECT = CORNER.

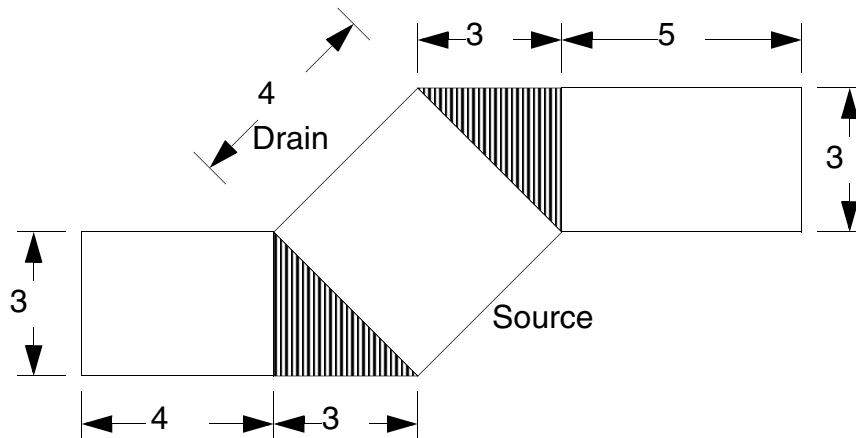
For another pulldown device:  
WEFFECT = 1.0,  $W = 16$

## Dracula Reference

### Operation Block Commands

---

$$L = 3.31, \text{ and } W_{\text{actual}} = 12.69$$



#### Example 2 - Using the *K* option and the *SMASH* Qualifier

LVSCHK[K] SMASH=[RC]

The above command will not smash any parallel devices with the exception of parallel resistors and capacitors. If you use the above command in the OPERATION block and SMASH-RES-TYP = YES and/or SMASH-CAP-TYP=YES command in the DESCRIPTION block of commands, this command will only smash series devices.

#### Example 3 - Merging SUP and SDW Devices into a Single MOS

LVSCHK[H]

This example merges all SUP and SDW serial MOS devices with a common gate into a single MOS device, but it does not merge SMID MOS devices with a common gate:

#### Example 4 - Merging SMID Devices into a Single MOS

LVSCHK[OJ]

This example merges all SMID serial MOS devices with a common gate, but it does not merge SUP and SDW serial MOS devices with a common gate:

#### Example 5 - Merging SDW, SUP, and SMID Devices into a Single MOS

LVSCHK[OHJ]

This last example merges all SUP, SDW, and SMID serial MOS devices with a common gate into a single MOS device:

## MULTILAB

MULTILAB {*trapfile*} OUTPUT *c-name* *l-num* {*d-num*}

### Description

Determines if a single node is labeled with different text strings, which usually occurs when two or more labeled nodes are shorted to each other. For example, if one node is labeled as both VCC and VSS, the power and ground nodes are shorted to each other somewhere in the layout. To locate where these nodes are shorted, MULTILAB traces the closest path between the different labels.

MULTILAB outputs the “pruned” nodes that have more than one label. Text involved with multiple labels is discarded after this check and no longer exists for any other ERC check. Multiple label problems must be cleared of errors before you inspect other ERC checks. MULTILAB and SAMELAB errors lead to false LVS or LPE output, so you should clear their errors before going on to LVS or LPE. For more information about ERC checks, refer to [Chapter 4, “Checking Electrical Rules \(ERC\).”](#)

It is strongly recommended that you include both MULTILAB and SAMELAB commands in an LVS or LPE input rules file.

### Arguments

*trapfile*

Name you designate as the output file to contain composite trapezoids. Before you do any operation on a *trapfile*, you must *OR* it with itself or with the substrate layer.

OUTPUT

Sends the results of the operation to an output cell.

**Note:** The output cell you specify cannot be used as the output of another command.

*c-name* *l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

## Dracula Reference

### Operation Block Commands

---

#### Example

This example checks all nodes that have more than one label. The result is output to the *shorts40* cell on layers beginning with layer 40 and ending at a higher layer number, depending on the number of CONNECT-LAYERS and contact layers.

```
MULTILAB OUTPUT shorts 40
```

## MULTI-SHEETRES

MULTI-SHEETRES RES[*type*] *width sheet\_res\_value*

### Description

Provide different sheet resistance values based on different resistor widths to compute the resistance of the parasitic resistor devices defined by the associated PARASITIC RES command. The MULTI-SHEETRES must follow the corresponding PARASITIC RES and ATTRIBUTE RES commands. This is the expanded rule for multiSheetRes option in extractParasitic command.

### Arguments

*type*

A two-character code that denotes the *type* of parasitic resistor device. The first character must be A-Z. The second character can be A-Z or 0-9, excluding 8. The second character is optional. Dracula uses this code to match this command with the associated PARASITIC RES command.

*width*

For resistors with width less than or equal to this value (larger than the previous width range), use its corresponding sheet\_res\_value to calculate resistance. For resistor width larger than the maximum width specified in MULTI-SHEETRES commands, use *sheet\_res\_value* of ATTRIBUTE for resistance calculation.

*sheet\_res\_value*

The sheet resistance value in units per square, where the units correspond to the resistance units specified in the UNIT command.

### Example

In this example, resistor with width > 0 and <= 0.1, use 0.15 as sheet resistance value. Resistor with width > 0.1 and <= 0.15, use 0.1 as sheet resistance value. for resistor width > 0.15, use 0.15 as sheet resistance value.

```
PARASITIC RES[P] PRES PTRM
ATTRIBUTE RES[P] 0.1
MULTI-SHEETRES[P] 0.1 0.15
MULTI-SHEETRES[P] 0.15 0.1
```

## NDCOUNT

```
NDCOUNT elem {[type]} layer condition n1 {trapfile} OUTPUT c-name l-num
 {d-num} {&}
```

### Description

Counts nodes that connect to an element through specific layers. The command outputs those elements with specified node counts to an output cell.

**Note:** The output cell you specify cannot be used as the output of another command.

### Arguments

*elem*

Type of element to check. You must define this element by an ELEMENT command.

MOS = MOS devices  
BJT = Bipolar devices  
PAD = PAD windows  
LDD = LDD devices

*type*

Two-letter (A-Z) code that denotes the type of MOS device. This can differentiate N-channel from P-channel in CMOS or differentiate MOS devices with different implants.

*layer*

Layer through which the connections are being checked. A layer name of *ALL* can check all of the conductor layers defined in the ELEMENT command for the *elem* and *type*.

*condition n1*

Tests according to one of the following conditions:

|              |                                                             |
|--------------|-------------------------------------------------------------|
| LT <i>n1</i> | For a node count less than the number <i>n1</i>             |
| LE <i>n1</i> | For a node count less than or equal to the number <i>n1</i> |
| EQ <i>n1</i> | For a node count equal to the number <i>n1</i>              |
| NE <i>n1</i> | For a node count not equal to the number <i>n1</i>          |
| GT <i>n1</i> | For a node count greater than the number <i>n1</i>          |



## Dracula Reference

### Operation Block Commands

---

GE *n1*      For a node count greater than or equal to the number  
*n1*

where the number *n1* can be an integer between 0 and 63.

*trapfile*

Name you designate as the output file to contain composite trapezoids. Before you do any operation on a *trapfile*, you must *OR* it with itself or with the substrate layer.

OUTPUT

Sends the results of the operation to an output cell. The output cell you specify cannot be used as the output of another command.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

&

Indicates a conjunctive rule. You can conjoin this command with other NDCOUNT and ECONNECT commands.

**Note:** NDCOUNT outputs the device layers of the element that satisfies the specific condition. ELEMENT PAD has no effect on the NDCOUNT command.

### Examples

In this example, all MOS elements that have only one node connected through the *srcdrn* layer are considered as capacitors. Thus, plotting the output cell *caps40* on layer 40 shows all MOS capacitors.

```
NDCOUNT MOS srcdrn EQ 1 OUTPUT caps 40
```

This example collects MOS elements that have source/drain counts greater than 2 (a three-terminal source/drain MOS device). The output cell name is *twosd40*.

```
NDCOUNT MOS srcdrn GT 2 OUTPUT twosd 40
```

## Dracula Reference

### Operation Block Commands

---

For a previously constructed metal-gate CMOS element, this check detects the elements that have a connection to other than one node through the metal. The output cell name is *ergate40*.

```
NDCOUNT MOS[C] metal NE 1 OUTPUT ergate 40
```

In the following example, all bipolar elements that have their collector, base, and emitter terminals shorted together are output in error cell *mixup50* on layer 50.

```
NDCOUNT BJT all EQ 1 OUTPUT mixup 50
```

## NEIGHBOR

```
NEIGHBOR dense-layer sparse-layer [grow-number] outlayer {OUTPUT c-name l-
num}
```

### Description

Sizes the sparse layer, performs a logical AND with the sized sparse layer and the dense layer, and checks the distance between the sparse layer and the created layer.

This command enhances performance for DRC checks between sparse and dense layers.

### Arguments

*dense-layer*

Input layer on which to perform logical AND with sized *sparse-layer*.

*sparse-layer*

Input layer to size and perform logical AND with *dense-layer*.

*grow-number*

Amount to enlarge the *dense-layer*. The default value is 0.0.

*outlayer*

Output layer for the operation.

OUTPUT

Sends the results of the operation to an output cell.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

### Example

```
NEIGHBOR metal pad 52 padmet
EXT pad padmet LT 50 OUTPUT error 63
```

## NODE-FILE

NODE-FILE [F/N] = *filename* {OUTPUT *cellname layer-number*}

### Description

Identifies a file that lists a node set to be extracted in a selective LPE run. Include this command in the Operation block if you want LPE to extract the layout parasitics and devices associated with a particular path. As an alternative to NODE-FILE, you can use the NODE-SELECT command to list the nodes.

NODE-FILE can optionally output the node set to a cell.

You must use two other commands for selective LPE: LPE-QUERY and SELECT-MODE. For details about these commands, refer to [Chapter 11, “Description Block Commands.”](#)

### Arguments

F

Used with LPE-QUERY=CORE only. Enforces a filtering process so only the devices connected to the user-specified nodes are output. This eliminates some parasitic capacitors that connect only to the adjacent nodes.

N

When you specify node numbers instead of node names, you need to turn this option *on*.

OUTPUT

Outputs the geometries of each conduction layer to a unique file. For example, if you have three conduction layers, the selected geometries are dumped into the following three output files:

```
cellName+layerNumber
cellName+(layerNumber+1)
cellName+(layerNumber+2)
```

*filename*

Name of a file that lists a node set.

*cellname*

Name of an optional output cell. The name can have up to six alphanumeric characters and no special characters.

## Dracula Reference

### Operation Block Commands

---

*layer-number*

Layer number of the output cell. *Layer-number* depends on your CAD system. If your *cellname* is *rule03* and *layer-number* is 5, the output cell name is *rule0305*.

### Example

```
NODE-FILE = nets1
```

## NODE-SELECT

```
NODE-SELECT {[N|F]} = net-name.../ [OUTPUT cellname layer-number]
```

### Description

Lists a node set to be extracted in a selective LPE run. Include this command in the Operation block if you want LPE to extract the layout parasitics and devices associated with a particular path.

As an alternative to NODE-SELECT, you can use the NODE-FILE command to identify a file that lists the nodes. You can use either the NODE-SELECT command or the NODE-FILE command to select the node set, but not both. For details, refer to the NODE-FILE command description. NODE-SELECT can optionally output the node set to a cell.

You must use two other commands for selective LPE: LPE-QUERY and SELECT-MODE. For details about these commands, refer to the “Description Block Commands” chapter.

### Arguments

N

If you specify node numbers instead of node names for *net-name1* and *net-name2*, you must also specify this option.

F

Used with LPE-QUERY=CORE only. Enforces a filtering process so only the devices connected to the user-specified nodes are output. This eliminates some parasitic capacitors that connect only to the adjacent nodes.

*net-name*

A list of the node names or numbers in the node set separated by spaces. You must end the node list with a slash (/).

OUTPUT

Outputs the geometries of each conduction layer to a unique file. For example, if you have three conduction layers, the selected geometries are dumped into the following three output files:

```
cellName+layerNumber
cellName+(layerNumber+1)
cellName+(layerNumber+2)
```

## Dracula Reference

### Operation Block Commands

---

*cellname*

Name of an optional output cell. The name can have up to six alphanumeric characters and no special characters.

*layer-number*

Layer number of the output cell. *Layer-number* depends on your CAD system. For example, if *cellname* is *rule03* and *layer-number* is *5*, the output cell name is *rule0305* for the last layer specified in the *connect-layer* statement. The subsequent cells (*rule0306*, *rule0307*,...) are assigned using a reverse sequence of the *connect-layer* conductor layers.

### Example

```
NODE-SELECT = NET1 NET2 NET3 / OUTPUT NEWCELL 5
```

## Dracula Reference

### Operation Block Commands

---

## NOT

```
NOT {[D{n}]} layer-a layer-b trapfile {OUTPUT c-name l-num {d-num}}
```

### Description

Creates a new layer from two other layers. The trapezoid file created by this operation consists of the first layer minus the region shared by both layers.

Database partition functions were added for the SELECT, SIZE, and LOGICAL commands. These enhancements resolve capacity limitation issues with Dracula. This allows you to partition input layers of these functions into blocks. When you run Distributed Dracula, each block (pair) can be assigned to one processor. Specify a value between 2-10 inclusive following 'D' to indicate the number of partitions desired. The default value of 'D' is 4 if the value is omitted.

### Note:

- There is a performance gain in the database partition feature for SELECT in the 4.8 version. The same is not true for SIZE and LOGICAL on account of too much overhead from the pre-process (PRESEL for SIZE) and post-process (LOGMRG for LOGICAL) stages during the operations. Database partition scheme is designed specially for resolving Dracula capacity issues and is applied to flat mode only.
- In the 4.81 and 4.9 versions, Dracula provides a method to optimize original rules and create more optimized rules to speed up run time. In the description block commands section, you can add the command `optimize-rule=YES/NO` (default is NO) and speed up the run time. In the 4.81 and 4.9 Dracula versions, PDRACULA issues to you suggestions for optimizing your rule. See the section, "[Example 2 - Optimizing the Rule Using the NOT Command](#)" for more information.

### Arguments

D

Turn on the database partition feature

n

Number of blocks to be partitioned into

*layer-a*

Name of the first input layer



## Dracula Reference

### Operation Block Commands

---

*layer-b*

Name of the second input layer

*trapfile*

Layer created by the logical operation. Arguments *layer-a* and *layer-b* are order dependent and can affect the output *trapfile* if connectivity has been established. The electrical node information is automatically stamped on this generated layer from the first layer (*layer-a*) name after nodal information is set up by using the CONNECT command on *layer-a*. The *trapfile* can have up to seven characters and no special characters. A-Z and 0-9 are permitted, but the first character must be a letter.

OUTPUT

Sends the results of the operation to an output cell.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

E

Creates an edge information file for each generated layer. For sidewall capacitance extraction, use this information. This information is used for sidewall capacitance extraction.

### Example 1

```
AND ndiff poly ngate
AND pdiff poly pgate
```

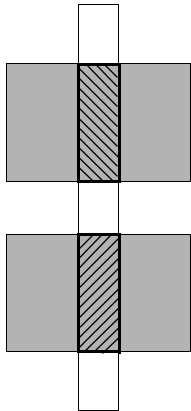
## Dracula Reference

### Operation Block Commands

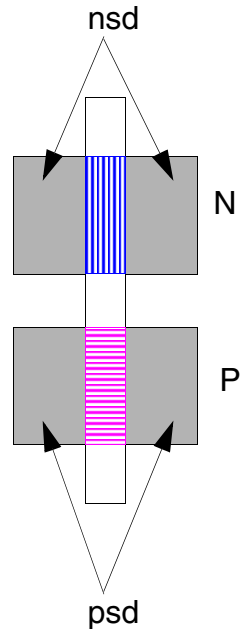
---

```
NOT ndiff ngate nsd
NOT pdiff pgate psd
```

Result of AND



Result of NOT



### Example 2 - Optimizing the Rule Using the NOT Command

In the following rule,

Rule-A:

```
NOT L1 T1 L2
NOT L2 T2 L3
NOT L3 T3 L4
NOT L4 T4 L5
```

In Rule-A (the above rule) L2, L3, and L4 are not used in the other rule. You can optimize the rule as follows:

Rule-B:

```
CAT T1 T2 T3 T4 T5
NOT L1 T5 L5
```

## **Dracula Reference**

### **Operation Block Commands**

---

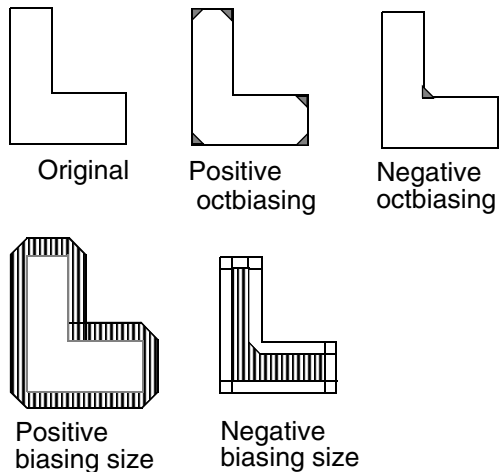
In Rule-A, Dracula requires run time for running four NOT commands, but in Rule-B, Dracula requires only one NOT command's run time.

## OCTBIAS

`OCTBIAS inlayer BY n1 outlayer {OUTPUT c-name l-num}`

### Description

Cuts all Manhattan corners on the input layer into octagonal corners with a radius of *n1* and outputs the results to a new layer.



### Arguments

*inlayer*

Name of the input layer.

*n1*

Radius of the octagonal corners.

*outlayer*

Name of the output layer.

OUTPUT

Sends the results of the operation to an output cell.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD

## Dracula Reference

### Operation Block Commands

---

system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

#### Example

```
*OPERATION
.
OCTBIAS metal BY 5 omet OUTPUT OMET 50
.
SIZE[X] poly BY 10 opoly OUTPUT OPOLY 63
.
*END
```

## Dracula Reference

### Operation Block Commands

---

## OR

```
OR [D{n}]] layer-a layer-b trapfile {OUTPUT c-name l-num {d-num}}
```

### Description

Creates a new layer from two other layers. The trapezoid file created by this operation consists of the total region of both layers.

Database partition functions were added for the SELECT, SIZE, and LOGICAL commands. These enhancements resolve capacity limitation issues with Dracula. This allows you to partition input layers of these functions into blocks. When you run Distributed Dracula, each block (pair) can be assigned to one processor. Specify a value between 2-10 inclusive following 'D' to indicate the number of partitions desired. The default value of 'D' is 4 if the value is omitted.

### Note:

- There is a performance gain in the database partition feature for SELECT in the 4.8 version. The same is not true for SIZE and LOGICAL on account of too much overhead from the pre-process (PRESEL for SIZE) and post-process (LOGMRG for LOGICAL) stages during the operations. Database partition scheme is designed specially for resolving Dracula capacity issues and is applied to flat mode only.
- In the 4.81 and 4.9 versions, Dracula provides a method to optimize original rules and create more optimized rules to speed up run time. In the description block commands section, you can add the command `optimize-rule=YES/NO` (default is NO) and speed up the run time. In the 4.81 and 4.9 Dracula versions, PDRACULA issues to you suggestions for optimizing your rule. See [“Example 2 - Optimizing the Rule Using the OR Command”](#) section for more information.

### Arguments

D

Turns on the database partition feature

n

Number of blocks to be partitioned into

*layer-a*

First input layer name.

## Dracula Reference

### Operation Block Commands

 $layer-b$ 

Second input layer name.

*trapfile*

Layer created by the logical operation. The *trapfile* can have up to seven characters and no special characters. A-Z and 0-9 are permitted, but the first character must be a letter.

## OUTPUT

Sends the results of the operation to an output cell.

*c-name* *l-num*

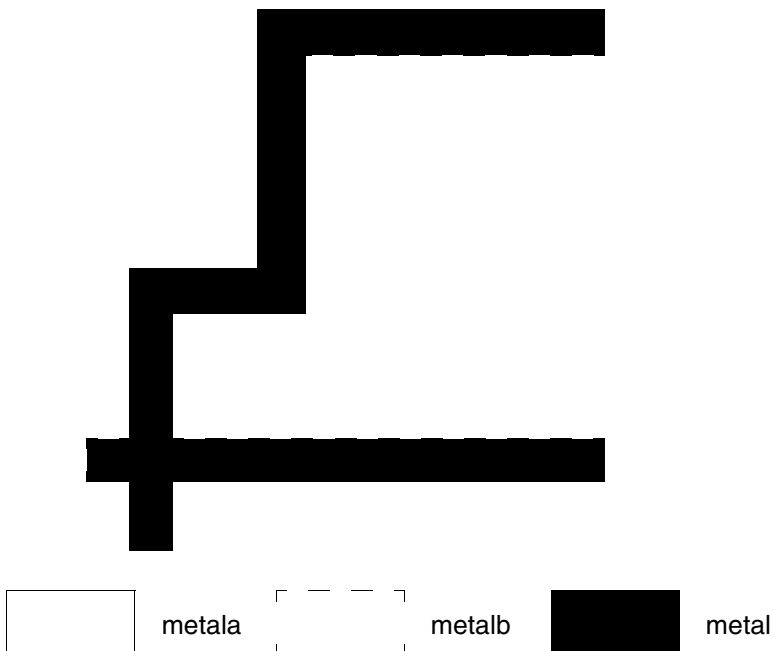
*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

 $d-num$ 

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

### Example 1

OR metala metalb metal



## **Example 2 - Optimizing the Rule Using the OR Command**

You can replace several OR commands by the command, `CAT+one-OR`.

For example,

You can replace these original rules:

```
OR FL FLP FL
OR FL ZFLPFL
OR FL FLN FL
OR FL ZFLNFL
OR FL FLPXFL
OR FL ZFLPXFL
OR FL FLNXFL
OR FL ZFLNXFL
OR FL FLPHFL
OR FL FLNHFL
OR FL FPB FL
OR FL FNB FL
```

by the following:

```
cat ZFL FLP ZFLPFLN ZFLNFLPXZFLNXFLPHFLNHFPBtmp1
or tmp1 flfl
```



## OVERLAP

```
OVERLAP input-layer trapfile {OUTPUT c-name l-num {d-num}}
```

### Description

Creates a layer out of overlapping polygons on the same layer. Use this command only for an input layer from your CAD graphics system.

### Arguments

*input-layer*

Name of the input layer.

*trapfile*

Layer created by this command. Can be up to seven alphanumeric characters and no special characters. The first character must be a letter.

OUTPUT

Outputs edges to the output file in an output cell.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

### Example

This example checks input layer *diff* for digitized polygons that overlap. Use to check overlaps prior to running optical PG.

```
OVERLAP diff difov OUTPUT ov1 50
```

## OVL

```
OVL in-lay1 inlay2 {WITHIN nn} out-lay1 {out-lay2}
 {OUTPUT c-name l-num {d-num} }
```

### Description

Outputs the trapezoids from the first input layer that are overlapped or within a specific distance from those in the second input layer to one or two output layers.

### Arguments

{WITHIN *nn*}

Positive value specifying the growing value of *in-lay2*.

{*out-lay2*}

Outputs non-overlapping shapes from *in-lay1* (while with overlapping shapes in *out-lay1*).

### Example

```
OVL CONTACT METAL WITHIN 2.0 GOODCON BADCON
```

Will output from CONTACT that are within 2 microns from METAL to GOODCON, and the others in BADCON.

## PARAMETER CAP

```
PARAMETER CAP [type] value-a value-b
```

### Description

Inputs the capacitance per unit area (*value-a*) and capacitance per unit perimeter (*value-b*). This LVS command extracts the capacitance values of the capacitors and compares these values to the schematic values. The PARAMETER CAP command works with the ELEMENT CAP command. If you do not specify PARAMETER CAP for a given ELEMENT CAP in an LVS run, the default is zero and LVS does not check capacitor value.

### Arguments

*type*

The type of CAP device specified in the ELEMENT CAP command.

*value-a*

The capacitance per unit area of *layer-a* that exists between *layer-b* and *layer-c* of the ELEMENT CAP command. The value you specify for *value-a* can be a floating point number (maximum 7 digits) or a floating point number followed by an integer exponent (maximum 2 digits). When you specify floating point numbers, the leading zeros are not counted in the 7-character restriction. The preprocessor rounds the value at the 7th digit. In scientific notation, the limit is 16 characters and is truncated at the 17th.

*value-b*

The capacitance per unit perimeter of *layer-a* of the ELEMENT CAP command. The value you specify for *value-b* can be a floating point number (maximum 7 digits) or a floating point number followed by an integer exponent (maximum 2 digits). When you specify floating point numbers, the leading zeros are not counted in the 7-character restriction. The preprocessor rounds the value at the 7th digit. In scientific notation, the limit is 16 characters and is truncated at the 17th.

### Example 1

```
ELEMENT CAP[A] metpol metal poly
PARAMETER CAP[A] 1E-18
```

For LVS capacitance checking, you must use the LVSCCHK or LPECHK option of CAPVAL because it specifies the value of 1E-18 per square unit.

## Example 2

```
ELEMENT CAP[P] pl2 poly buried
PARAMETER CAP[P] 1
```

If you use the LVCHK option of CAPVAL (together with \*.CAPVAL in CDL netlist) and the parameter value of 1 per square unit, as shown in this example, then checking the capacitance value is equivalent to checking capacitor area. The only exception is that the VALUE= *syntax*, instead of the AREA= *syntax*, is reported in the LVS report file.

**Note:** For CAPVAL and CAPAREA specification, refer to the LVCHK or the LPECHK command descriptions and the “Compiling Network Descriptions” chapter.

## PARAMETER RES

PARAMETER RES[*type*] *value*

### Description

Inputs the sheet resistance value of resistors defined by an ELEMENT RES command. This LVS command indicates the resistance values for the resistors extracted from the layout database. If you do not specify PARAMETER RES for a given ELEMENT RES in an LVS run, the default is zero, and LVS does not check the resistor value.

The unit of sheet resistance must agree with the layout unit and the resistance unit. Specify units in the Description block with the UNIT command so the calculated values agree with the schematic information.

### Arguments

*type*

Type of RES device specified in the ELEMENT RES[*type*] command.

*value*

Sheet resistance value. If you specify this value, you must use the LVSCCHK option of RESVAL. The value you specify for *value-a* or *value-b* can be floating point (maximum seven digits) or a floating-point number followed by an integer exponent (maximum two digits). When you specify floating-point numbers, the leading zeros are not counted in the seven-character restriction. However, if the significant digits exceed the seven-digit limit, the preprocessor rounds the value at the seventh digit. In scientific notation, the limit is 16 characters and the number is truncated at the 17th.

If you specify a value of one *ohm* per square, you must specify the LVSCCHK RESIZE (resistor size) option.

### Example 1

In the following example, LVS extracts resistance values for *S* type resistors using a sheet resistance value of 100 per square. When you specify sheet resistance, use the LVSCCHK option of RESVAL.

```
ELEMENT RES[S] PRES PTERM
PARAMETER RES[S] 100
```

#### Example 2

```
ELEMENT RES[P] PRES1 POLY
PARAMETER RES[P] 1
```

If you use the LVCHK option of RESVAL (together with \*.RESVAL in the CDL netlist) and specify the parameter value of 1 ohm per square, as in this example, the resistance value check is equivalent to checking resistor size. The only exception is that Dracula reports the VALUE = *syntax*, instead of the NS = *syntax*, in the LVS report file. Remember that you are checking the resistor size.

If you want LVS to perform resistor size checking, use the LVCHK option of RESIZE. You do not need to specify the parameter value.

For RESVAL and RESIZE specifications, refer to the LVCHK or the LPECHK command descriptions and Chapter 5, "Compiling Network Descriptions (LOGLVS)."

For information about resistance computation, refer to [Chapter 7, "Extracting Electrical Parameters \(LPE\)."](#)

## PARASITIC CAP

PARASITIC CAP[R] [subType] deviceLayer termLayer1 termLayer2

### Description

Defines parasitic capacitor devices resulting from overlaps or fringe effects. Add an ATTRIBUTE CAP command after the PARASITIC CAP command. ATTRIBUTE CAP associates the geometry data with the process-dependent electrical properties. ATTRIBUTE does not take units. The scale of the parasitic attribute needs to match the scale of the layout database.

Parasitic capacitors apply to stray capacitances that are either

- Between two nodes at different conductor layers
- Between two nodes at the same conductor layer

This command does not extract capacitances if both terminals are at the same node. When extracting the overlap capacitance with fringe effect, the first terminal layer determines the possible fringe effect on the overlap capacitance. Thus, the sequence of terminal layers is important. Refer to the [ATTRIBUTE CAP](#) section for capacitance extraction of different types.

**Note:** When specifying the *R* option, note the following conditions:

- When the *poly* layer is extracted for parasitic resistance, you must specify the *gate* layer in the CUT-TERM command.
- The first terminal layer in the PARASITIC CAP command must be derived from the *cut-term* connection layer by using logical operations.

For more information on extracting parasitic capacitance, see “[Extracting Parasitic Capacitance](#)” in the “Extracting RC Parasitics” chapter of the *Dracula User Guide*.

### PARASITIC CAP with PRE

Use the PARASITIC CAP command with PRE. Resistor device layers for PARASITIC RES are allowed as terminal layers for PARASITIC CAP. These resistor device layers, in general, have no nodal information.

If the first terminal layer in a PARASITIC CAP command is a PARASITIC RES device layer, the PARASITIC CAP device layer is not required to have nodal information.

**Note:** Dracula issues a warning if the PARASITIC CAP command is issued before any PARASITIC RES command.

## Arguments

R

Use this option only when extracting TPR and CLL parameters in a PRE job.

*subType*

Two-character code that denotes the type of parasitic CAP device. The first character must be A-Z. The second character is optional and can be A-Z or 0-9 (excluding the number 8). This code differentiates between parasitic capacitor devices.

*deviceLayer*

Device layer, one region per device. This layer defines the area and perimeter of the capacitor. The region of this layer touches or overlaps the capacitor conductor layers. Also, this device layer must carry the same nodal information as the *termLayer1* terminal conductor.

*termLayer1*

Terminal conductor layer. This layer is the conductor layer for one side of the capacitor and must carry node information.

**Note:** When specifying the *R* option, *termLayer1* must be derived from the original *cut-term* layer by logical operations.

*termLayer2*

Terminal conductor layer. This layer is the conductor layer for the other side of the capacitor and must carry node information.

**Note:** The ELEMENT CAP command represents discrete devices only. Use the PARASITIC CAP command for stray or parasitic capacitance.

## Example 1

In the following example, the PARASITIC capacitor of type *X* forms between *mt1* and *poly* conductor layers. The device layer is *mtpoly* (*mtpoly* = metal and poly).

```
STAMP mtpoly BY mt1
PARASITIC CAP[X] mtpoly mt1 poly
ATTRIBUTE CAP[X] 0.0005
```

The two connecting layers of capacitor terminals are *mt1* and *poly*. The device layer, *mtpoly*, must always carry the same electrical node of the first connecting layer, *layer-b*. That is, *mtpoly* and *mt1* must convey the same node in the example. Capacitance of type *X* is calculated using the following formula:

$$(\text{area}) \times (0.0005)$$



## Example 2

In this example, the PARASITIC capacitor of [F] type is for the fringe capacitance between metal lines. The device layer is *metal* and the conductor layers are *metal*. The parasitic capacitance between metal lines is calculated for spacing distances up to 6 units.

```
PARASITIC CAP[F] metal metal metal
ATTRIBUTE CAP[F] 6.0 0.0005
```

The capacitance calculation is calculated using the following formula:

$$(\text{projected metal length}) \times (0.005) \times (1/\text{distance between metal})$$

## Example 3

```
PARASITIC CAP[mp] mtpol mt1 poly
ATTRIBUTE CAP[mp] 1E-18 1E-12
```

or

```
PARASITIC CAP[mp] mtpol mt1 poly
ATTRIBUTE CAP[mp] .0005 .00012
```

or

```
PARASITIC CAP[mp] mtpol mt1 poly
ATTRIBUTE CAP[mp] .0001234567E-9 .00007654321E-10
```

Remember, the leading zeros do not count in the seven significant digit restriction.

Refer to the [ATTRIBUTE CAP](#) section for more examples.

## PARASITIC DIO

```
PARASITIC DIO [type] layer-a layer-b layer-c {layer-d} {layer-e}
```

### Description

Defines the parasitic P-N and N-P junction diodes. You can extract the following six types of parameters from a parasitic junction diode:

#### A1,P1

Area or perimeter of a diffusion geometry not overlapped by another layer.

#### A2,P2

Area or perimeter of a diffusion geometry overlapped by a second layer, such as a metal-contact opening layer.

#### A3,P3

Area or perimeter of a diffusion geometry overlapped by a third layer, such as a buried-contact (a contact without metal) opening layer, an implant layer, or the gate region of a transistor.

### Arguments

*type*

Two-character code that denotes the type of parasitic DIO device. The first character must be A-Z. The second character is optional and can be A-Z or 0-9 (except for the number 8). This code differentiates parasitic diode devices.

*layer-a*

*A1*, *P1* device layer, one region per device. This layer defines the parasitic diode region that touches or overlaps the anode and cathode conductor layers. Using this device layer extracts the diode area and perimeter as *A1* and *P1*, respectively, or

```
A1 <-- A1
```

```
P1 <-- P1
```

This device layer must be the same layer name as the anode or cathode conductor layer, depending whether they are P-N or N-P junction diodes, and must carry nodal information.

## Dracula Reference

### Operation Block Commands

---

#### *layer-b*

Anode conductor layer. This layer is the conductor layer for the anode terminal and must carry node information. For a P-N diode, this layer name must be the same as the name you specified in *layer-a*.

#### *layer-c*

Cathode conductor layer. This layer is the conductor layer for the cathode terminal and must carry node information. For an N-P diode, this layer name must be the same as the name you specified in *layer-a*. The preprocessor checks this syntax.

#### *layer-d*

*A2*, *P2* definition layer. This layer modifies the device layer either by overlapping or touching geometries. The extracted diode area and perimeter are modified to the following:

```
A1 <-- A1-A2
P1 <-- P1-P2
A2 <-- A2
P2 <-- P2
```

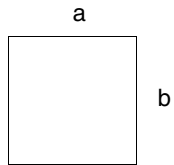
#### *layer-e*

*A3*, *P3* definition layer. This layer further modifies the device layer, after *A2*, *P2* are invoked, either by overlapping or touching geometries. The extracted diode area and perimeter are modified to the following:

```
A1 <-- A1-A2-A3
P1 <-- P1-P2-P3
A2 <-- A2-A3
P2 <-- P2-P3
A3 <-- A3
P3 <-- P3
```

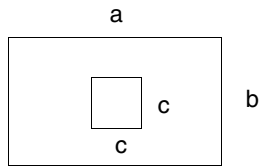
### Perimeter Examples

A1, P1



$$A1 = a \times b$$
$$P1 = 2a +$$

A1, P1, A2, P2



$$A1 = (a \times b) - c \times$$
$$c$$
$$P1 = 2a + 2b$$
$$A2 = c \times c$$

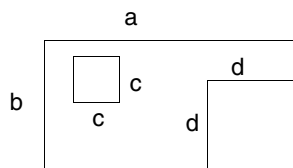
## Dracula Reference

### Operation Block Commands

---

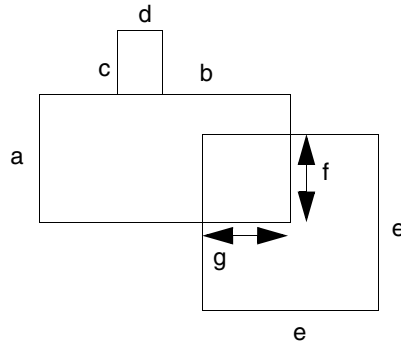
**Note:** The ELEMENT DIO command represents the true discrete device only. Use the PARASITIC DIO command for stray/parasitic diodes.

A1, P1, A2, P2, A3, P3



$$\begin{aligned} A1 &= (a \times b) - (c \times c) - (d \times d) \\ P1 &= 2a + 2b \\ A2 &= c \times c \\ P2 &= 0 \\ A3 &= d \times d \\ P3 &= 2d \end{aligned}$$

A1, P1, A2, P2, A3, P3



$$\begin{aligned} A1 &= (a \times b) - (g \times f) \\ P1 &= 2a + 2b - d - f - g \\ A2 &= 0 \\ P2 &= d \\ A3 &= (f \times g) \\ P3 &= f + g \end{aligned}$$

**LEGEND**  
A1, P1 - a, b  
A2, P2 - c  
A3, P3 - d

## Examples

In the following example, the PARASITIC diode of type *N* forms between *nsr cdrn* and p-substrate of a N-channel MOS process. The device layer is *nsr cdrn*, the anode conductor layer is *p sub*, and the cathode conductor layer is *nsr cdrn*. The extracted diode area and perimeter are calculated from device layer *nsr cdrn*. The *nsr cdrn* and the cathode conduction layer have the same layer name and must carry nodal information.

```
PARASITIC DIO[N] nsr cdrn p sub nsr cdrn
```

In the following example, the PARASITIC diode of type *P* forms between *p dio* and *n well*. The extracted diode area is calculated from device layer *p dio* and the perimeter and from device layer *p dio* minus the side where the *p gate* touches *p dio*. This effectively removes the gate perimeter region of the diode. The anode conductor is *p dio*, and the cathode is *n well*. The device layer *p dio* and the anode conductor layer have the same layer name and must carry nodal information.

```
PARASITIC DIO[P] p dio p dio n well p gate
```

## PARASITIC RES

`PARASITIC RES[type] layer-a layer-b {layer-c}`

### Description

Defines parasitic resistor devices in the layout database for PRE. An ATTRIBUTE RES command must follow the PARASITIC RES command to provide the associated process-dependent sheet resistivity used to compute the resistance values of the devices.

Dracula issues a warning if the PARASITIC CAP command is issued before any PARASITIC RES command.

For more information on extracting parasitic resistance, see “[\*Extracting Parasitic Resistance\*](#)” in the “Extracting RC Parasitics” chapter of the *Dracula User Guide*.

### Checking Methods

modes

Flat and composite.

checks

PRE and HPRE.

### Arguments

*type*

A one- or two-character code that denotes the type of parasitic resistor device. The first character must be A to Z; the second character, which is optional, can be A to Z or 0 to 9, excluding 8. Dracula uses this code to match this command with the associated ATTRIBUTE RES command.

*layer-a*

Device layer, one region per device. This layer defines the resistor region whose parameters are to be extracted. This region must touch or cut the conductor layer.

*layer-b*

Terminal conductor layer.

*layer-c*

Optional second terminal conductor layer.

## Dracula Reference

### Operation Block Commands

---

#### Example

In this example, LPE extractions of parasitic resistance values made from *PRES* are computed using the sheet resistance value .01 Kohms/square.

```
*DESCRIPTION
UNIT = CAPACITANCE, PF AREA, U PERIMETER, M RESISTANCE, K
*OPERATION
PARASITIC RES[P] PRES PTRM
ATTRIBUTE RES[P] .01
```

## Dracula Reference

### Operation Block Commands

---

## PATHCHK

```
PATHCHK {[options]} LEVEL n1 {trapfile} OUTPUT c-name
 l-num {d-num}
```

### Description

Checks MOS circuits where a node is supposed to have a path to both power and ground through one or more sources/drains of the MOS device, resistor device, BJT device, DIO device, and LDD device. This command assigns an error code to nodes without a proper path to either power or ground. The error codes are as follows:

---

|               |                                                                                                                      |
|---------------|----------------------------------------------------------------------------------------------------------------------|
| Error Level 1 | Internal nodes with a path to ground but not to power.                                                               |
| Error Level 2 | Internal nodes with a path to power but not to ground.                                                               |
| Error Level 3 | Internal nodes without a path to either power or ground.                                                             |
| Error Level 4 | Internal nodes without a path to power, ground or pads, where pads are labeled nodes reported in the 6PADL.DAT file. |
| Error Level 5 | Internal nodes with a path to both power and ground.                                                                 |

---

### Arguments

*options*

Can include the following:

- |   |                                                                                                                                                                                                                              |
|---|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| F | Excludes nodes that connect to unused devices (as in a gate array circuit). These nodes are not output. The <i>F</i> option is for error level 4 only.                                                                       |
| X | Excludes nodes that connect to the source/drain of the CROSS-OVER MOS device from the output. This command uses the gate as the crossover and does not use the source/drains. The <i>X</i> option is for error level 4 only. |

*n1*

Error level number (1-4). Nodes assigned this error level code are reported and placed in the output cell.



## Dracula Reference

### Operation Block Commands

---

*trapfile*

Name you designate as the output file to contain composite trapezoids. Before you do any operation on a *trapfile*, you need to “OR” it with itself or the substrate layer.

OUTPUT

Sends the results of the operation to an output cell.

**Note:** The output cell you specify cannot be used as the output of another command.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system.

If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

When you use ELEMENT PAD, input and output nodes with an attached ELEMENT PAD are not checked by the PATHCHK command. Also, the PATHCHK command recognizes a power node texted with :S as a power node. Therefore, error level 4 considers all pads (labeled nodes) to be potential supply nodes. The PATHCHK command level 4 does not trace paths beyond the pads. Path tracing in levels 1, 2, and 3 can be stopped only by power and ground nodes.

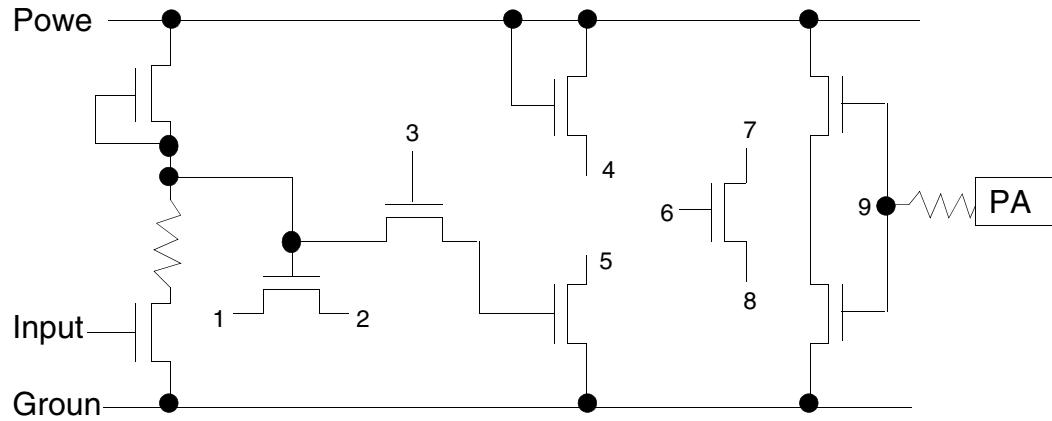
### Example

|             |         |        |           |
|-------------|---------|--------|-----------|
| PATHCHK     | LEVEL 1 | OUTPUT | path1r 50 |
| PATHCHK     | LEVEL 2 | OUTPUT | path2r 50 |
| PATHCHK     | LEVEL 3 | OUTPUT | path3r 50 |
| PATHCHK[FX] | LEVEL 4 | OUTPUT | path4r 50 |

## Dracula Reference

### Operation Block Commands

---



ERROR LEVEL 1:NODE 5  
ERROR LEVEL 2:NODE 4  
ERROR LEVEL 3:NODE 1, 2, 3, 6, 7, 8, 9, Input  
ERROR LEVEL 4 (OPTION F):NODE 1, 2, 3,  
ERROR LEVEL 4 (OPTION X):NODE 3, 6, 7, 8  
ERROR LEVEL 4 (OPTION F X):NODE 3

## PERI

```
PERI layer-a RANGE n1 n2 trapfile {OUTPUT c-name l-num (d-num)}
```

### Description

Performs a polygon perimeter check. This operation checks polygons to determine if their perimeter is within an area range and outputs to the specified output layer.

### Arguments

*layer-a*

Name of the input layer

RANGE *n1 n2*

Flags polygon with perimeter such that  $n1 \leq \text{PERI} \leq n2$

*trapfile*

Name of the output layer

OUTPUT

Sends the results of the operation to an output cell

*c-name l-num c-name*

is the name of the output cell which has six alphanumeric characters or fewer and no special characters, *l-num* is the cell layer number determined by your CAD system. If your *c-name* is `rule03` and *l-num* is 5, the output cell name is `rule0305`.

*d-num*

The datatype number associated with layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

### Example

```
PERI cont RANGE 3.98 4.02 cont1
```

Find the 1 micron length square of the `cont` layer. The perimeter of a square falls within the range of 3.98 to 4.02.

## PLENGTH

```
PLENGTH layer-a {RANGE/NE/EQ} n1 n2 {trapfile} {OUTPUT
 c-name l-num {d-num}}
```

### Description

Checks the length of a series of connected edges generated from other dimensional checks with *T* and *R* options, instead of just checking the length of an edge. PLENGTH has the same format as an AREA command.

### Arguments

*layer-a*

Input layer name. This layer must be generated by an ENCLOSURE, EXTERNAL, or INTERSECT command with *T* and *R* options.

RANGE *n1 n2*

Flags the paths with length such that  $n1 < \text{PLENGTH} < n2$ .

NE

Reports an error if any value of the layer specified is not equal to the value you specify.

EQ

Reports an error if any value of the layer specified is equal to the value you specify.

*trapfile*

Optional trapezoid file name.

OUTPUT

Sends the results of the operation to an output cell.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system.

If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

## Dracula Reference

### Operation Block Commands

---

#### Example 1

This example checks for *gate* length less than 10.0 microns. Option *TR* generates the *gatelen* layer.

```
SELECT gate TOUCH[1:1] epi epgate
ENC[TR] epgate diff LT .001 gatelen
PLENGTH gatelen RANGE 0.0 10.05 output d123 23
```

#### Example 2

This example checks all *gate* widths for less than 4 microns, including bent gates. Option *TR* generates the *gatewd* layer.

```
ENC[TR] gate poly LT .001 gatewd
PLENGTH gatewd RANGE 0 4 OUTPUT gatewd 24
```

## POSTENC

POSTENC *inlayer outlayer*

### Description

Promotes the data to the corresponding Hcell plane if there is any cell-to-cell overlapping. You can use this command in hierarchical mode only. When you run a hierarchical DRC check, the error is reported only on the appropriate cell.

### Arguments

*inlayer*

*Trapfile* from the WIDTH, EXTERNAL, ENCLOSURE, and INTERSECT commands.

*outlayer*

Generated *trapfile* name.

### Examples

POSTENC poly1 polyXX

POSTENC metall metalXX

## PROBE

```
PROBE label {trapfile} {OUTPUT c-name l-num {d-num}} {&}
```

### Description

Probes a node if the node is labeled. You must provide input text in the layout database to label a specified node. Although you can specify labels of up to 12 characters, PROBE recognizes only the first eight, so you need to make sure the first eight characters are unique. You can place the node in an output cell for visual inspection or reference.

### Arguments

*label*

Text label of the node considered for the check. The text must exist in the layout database and conform to the text rules.

*trapfile*

Name you designate as the output file to contain composite trapezoids. Before you do any operation on a *trapfile*, you need to “OR” it with itself or the substrate layer.

OUTPUT

Sends the results of the operation to an output cell.

**Note:** The output cell you specify cannot be used as the output of another command.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

&

indicates a conjunctive rule. This command can be conjoined with other PROBE and ELCOUNT commands.

**Note:** PROBE outputs the entire node that electrically carries the label.

## Dracula Reference

### Operation Block Commands

---

#### Examples

```
PROBE vcc OUTPUT vcc 50
```

```
PROBE clock OUTPUT clock 50
```



## RCONNECT

`RCONNECT layer-a layer-b BY cont-layer`

### Description

Defines the interlayer connectivity after PRE cuts parasitic resistors from the original conduction layers. It reconnects the layers after subnodes have been cut from the original conduction layers.

The syntax of the command is similar to that of the CONNECT command. Layers in the RCONNECT commands must include nodal information extracted from the original conduction layers involved in parasitic extraction.

All PRE runs must include the RCONNECT command because RCONNECT identifies the run as a PRE run to PDRACULA.

### Checking Methods

`modes`

Flat, composite.

`checks`

PRE, HPRE.

### Arguments

`layer-a`

Name of the first input layer. The layer must have nodal information.

`layer-b`

Name of the second input layer. The layer must have nodal information.

`cont-layer`

Specifies a layer that has been previously defined and consists of the contacts that connect *layer-a* and *layer-b*.

Multiple RCONNECT commands with the same `cont-layer` can have only one master layer and can connect from the master layer to any other layers. For more details, refer to the CONNECT command description in this chapter and the CONNECT-LAYER command description in the “Input-Layer Block Commands” chapter.

## Dracula Reference

### Operation Block Commands

---

#### Example

The POLY interconnect has been split into parasitic resistors and resistor terminals. The terminal layer created after the split is *PTRM*. The original connectivity was defined by the following:

```
CONNECT METAL POLY BY MCO
CONNECT METAL DIFF BY MCO
CONNECT POLY DIFF BY EPI
```

To reconnect the layers after the split layers and subnodes have been created, *PTRM* replaces *POLY* in the RCONNECT commands:

```
STAMP PTRM BY POLY
RCONNECT METAL PTRM BY MCO
RCONNECT METAL DIFF BY MCO
RCONNECT PTRM DIFF BY EPI
```

The STAMP command gets back nodal information on the layer *PTRM* from *POLY* if *PTRM* lost it in the extraction process.

## RECTCHK

```
RECTCHK[OPTION] layer {WIDLEN NE/EQ val1 val2} {WIDTH NE/EQ/LT/RA/GT val1 {val2}}
 {LENGTH NE/EQ/LT/RA/GT val1 {val2}} {trapfile} {OUTPUT err lay}
```

### Description

Checks rectangle shape, especially for contact/via layers, in flat mode only. Without any additional options the behavior is as follows:

- If there are no WIDTH/LENGTH/WIDLEN limitations it reports all non-rectangle shapes.
- If there is at least one of WIDTH/LENGTH/WIDLEN limitations, it reports all non-rectangle shapes and all rectangle shapes which satisfy limitation.

### Arguments

*OPTION*

Can include the following:

- |   |                                                                                                                                                                                                                                                                                                                                                                                                |
|---|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| R | Report the error only if the shape is a rectangle. Butted rectangles are not reported.                                                                                                                                                                                                                                                                                                         |
| A | If you use both WIDTH and LENGTH, by default, it is an OR operation. If you use [A], then it's an AND operation.                                                                                                                                                                                                                                                                               |
| T | Checks for butted trapezoids. This option is useful for checking the thickness of rings, and so on. Also, checks every trapezoid in a layer independently from each other.<br><br>The [R] option skips checking if it has butted manhattan rectangles. You should use [T] option only on manhattan shapes. If no WIDTH/LENGTH/WIDLEN limitation given, it outputs only rectangular trapezoids. |
| D | Checks figures that are exact donuts of a particular shape. If no WIDTH/LENGTH/WIDLEN limitation given, it outputs all donut shapes from input layer. See example 5 below to clarify the definition of donut shapes.                                                                                                                                                                           |

**Note:** Any combination of [R ], [ T ], and [D ] options is not allowed.

WIDLEN

Exact width length check

## Dracula Reference

### Operation Block Commands

---

WIDTH

Shorter side of rectangle

LENGTH

Longer side of rectangle

#### Example 1

```
RECTCHK contact WIDLEN NE 3 3 OUTPUT err 1
; Check the size of contact 3x3; Butted contacts are errors.
```

#### Example 2

```
RECTCHK[R] layer WIDTH NE 3 LENGTH GT 6
```

(It's an OR operation, not an AND operation.)

The width must be 3, and length cannot be longer than 6. Don't report it if it is not a rectangle. You no longer need to use WIDTH and AREA to do this kind of check.

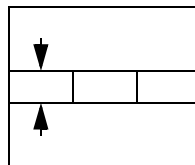
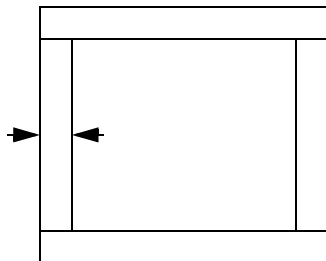
#### Example 3

```
RECTCHK[A] layer WIDTH LT 3 LENGTH GT 6 (it's AND)
```

Select rectangles with WIDTH smaller than 3 and LENGTH greater than 6.

#### Example 4

```
RECTCHK[T] ring WIDTH NE 1.2 OUTPUT err 1
```



False Error if the ring is small and thick

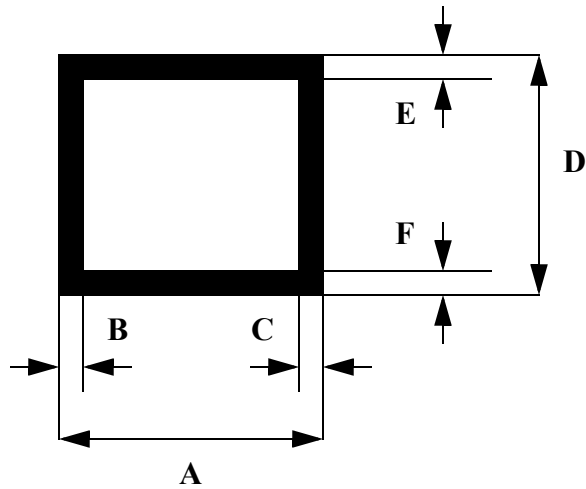
## Dracula Reference

### Operation Block Commands

---

#### Example 5

```
RECT [D] poly1 WIDTH LT ..18 OUTPUT NPE4 46
```



In this example, distance A,D will be checked as a length while distances B,C and E,F will be checked as widths. Any other shape is not considered a donut by [D ] option.

## RELOCATE

```
RELOCATE layer-a BY X Y trapfile {MAG = scale}
```

### Description

A Dracula operation that creates a new layer from another layer. The trapezoid file consists of data from *layer-a* shifted to a new x,y position or data from *layer-a* with each of its vertices multiplied by the specified scale.

### Arguments

*layer-a*

Input layer name.

*X, Y*

New location of *layer-a* data that is shifted by natural units specified for X and Y.

*trapfile*

Trapezoid file name. Consists of trapezoidal data shifted and/or magnified. *Trapfile* can have up to seven characters and no special characters. A-Z and 0-9 are permitted, but the first character must be a letter.

*scale*

Optional magnification factor. Default magnification is equal to unity. Accept only a scale of up to three decimal values.

New data location:

```
new x = (old x).(scale)+ X
new y = (old y).(scale)+Y
```

### Examples

This example shifts the data in the *metal* layer by a negative 5 units in the x direction. Outputs the data to the *met5* layer. Does not magnify.

```
RELOCATE metal BY -5 0 met5
```

This example magnifies the *active* layer by 5 times. Outputs the data to the *active5* layer.

```
RELOCATE active BY 0 0 active5 MAG = 5
```

This example magnifies the *cont* layer by 2 times. Shifts the data by negative 1 unit in the x direction and negative 2 units in the y direction. Outputs the final data to the *newcont* layer.

## **Dracula Reference**

### **Operation Block Commands**

---

RELOCATE cont BY -1 -2 newcont MAG=2

## RLENGTH

```
RLENGTH layer-a {&layer-a1} GT/GE/ LT/LE/ RANGE n1 {n2}
 {OUTPUT c-name l-num {d-num}} {&}
```

### Description

This command is similar to LENGTH, but it is used with EXT while LENGTH is only applied to a single layer. This command checks the length dimension based on the same region. If the error flags are created on the same region, they are merged to form the length of the error flag.

### Arguments

*layer-a*

Name of the input layer.

&*layer-a1*

Renames (conjunctive rename) the layer containing the error flags. The original layer name is then restored to the original layer. No spaces are allowed between the *layer-a* and &*layer-a1* names, for example, *poly&rpoly*.

Measurement Operation

Measures between all nonconsecutive pairs of inside edges of the polygons of the input layer. Compares the measurements against the limits you specify and flags violations.

GT/GE/

GT means that error segment length greater than *n1* layout units causes a violation. GE means that length greater than or equal to *n1* layout units causes a violation.

LT/LE/

LT means that length less than *n1* layout units causes a violation. LE means that length less than or equal to *n1* layout units causes a violation.

RANGE *n1* *n2*

RANGE means that length greater than *n1* and less than *n2* layout units causes a violation ( $n1 < \text{range} < n2$ ).

OUTPUT

Sends the results of the operation to an output cell.



## Dracula Reference

### Operation Block Commands

---

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system.

If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

Format of the output is the same as the original database.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

&

Indicates a conjunctive rule.

### Example

```
EXT METAL1 LT 0.8 &
RLENGTH METAL1 GT 2.0 OUTPUT err 01
```

## RSPFSELECT

RSPFSELECT {[opt]} {INST=YES/NO} *filename*

### Description

Generates a reduced standard parasitic file (RSPF) for HPRE. Use this command in composite mode only.

The default unit of input capacitance is *Farad*, and the default unit of resistance capacitance is *ohm*. The default unit for reduction algorithms is *PF*, and the default for resistance is *ohm*. Dracula converts input resistance capacitance values to the corrected scale during the reduction process. Dracula converts output resistance capacitance values from reduction algorithms back to the specified scale before RSPF is written out. Therefore, you must carefully specify the scale in the UNIT command in order to get meaningful resistance capacitance values.

### Arguments

*opt*

Specifies whether to backannotate (*S* option) or not (no option).

*INST=YES/NO*

The default is YES and the instance section will be output. The user can delete the instance section by using the NO option.

*filename*

Sends the results of the operation to this file. Dracula appends a *.dat* to the file specified.

### Example

This example creates a reduced parasitic file named *RSPF.dat* with backannotation.

```
RSPFSELECT[S] RSPF
```

## SAMELAB

SAMELAB {*trapfile*} OUTPUT *c-name* *l-num* {*d-num*}

### Description

Detects when the same label is assigned to multiple nodes. This situation usually occurs when

- The layout of a node is not complete
- There are missing contacts
- There are gaps or other errors

SAMELAB outputs multiple nodes with the same label in an output cell.

Text involved with SAMELAB is discarded after this check and thus the text no longer exists for any other ERC checks. SAMELAB problems must be cleared of errors before proceeding to inspect other ERC checks. MULTILAB and SAMELAB errors lead to false LVS or LPE output, so you must clear their errors before going on to LVS or LPE. For more information about ERC checks, refer to [Chapter 4, “Checking Electrical Rules \(ERC\).”](#)

You should include both MULTILAB and SAMELAB commands in an LVS or LPE rules file.

### Arguments

*trapfile*

Name you designate as the output file to contain composite trapezoids. Before you do any operation on a *trapfile*, you need to *OR* it.

OUTPUT

Sends the results of the operation to an output cell.

**Note:** The output cell you specify cannot be used as the output of another command.

*c-name* *l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

## Dracula Reference

### Operation Block Commands

---

#### Example

In this example, different nodes with the same label are output to the *opens40* error cell on the layer beginning with layer 40 and ending at a higher layer number depending on the number of CONNECT-LAYERS and contact layers.

```
SAMELAB OUTPUT opens 40
```

## SCONNECT

*SCONNECT upper-layer lower-layer BY contact-layer*  
*{LINK label}*

### Description

Creates a connection between a conductor layer and a soft connect layer. SCONNECT allows unidirectional passing of nodal information according to the CONNECT-LAYER sequence, that is, passing nodal information from the conductor to the soft connect layer, but not the reverse.

If multiple SCONNECT trapezoids are reported in the .ERC file, Dracula reports the following warning in the .ivs file:

```
/W WARNING :: SOFT-CONNECTING MULTIPLE NODES IN UPPER LAYERS TO ONE POLYGON
 REGION IN: PWELL
```

To use the global majority rule for SCONNECT conflict resolution, see GLOBAL-SCONNECT.

### Arguments

*upper-layer*

Upper layer name.

*lower-layer*

Lower layer name. Connectivity passes from *upper-layer* to *lower-layer* (soft connect layer) but not the reverse.

*contact-layer*

Previously defined layer that consists of the contacts that connect *upper-layer* to *lower-layer*. Must have at least three characters.

As an example, you can make a soft connection when you use one *NWELL* region to connect (and pass nodal information) between two separate *NSRCDRN* regions. In the regular nodal passing scheme, Dracula treats the three regions as one electrical node. However, the SCONNECT command assigns different node numbers to the regions.

When the nodes have different numbers, instead of merging all connecting nodes, SCONNECT picks and pushes one node number in *NSRCDRN* onto *NWELL*. Dracula selects the node number that contains the most trapezoids of connected nodes among the soft-connected layers.

## Dracula Reference

### Operation Block Commands

---

You can then use the SOFTCHK command to report the violation geometries in one or both connect layers. You can also use the ABORT-SOFTCHK command to stop the job when soft connection violations exist. If you ignore these violations, the errors might resurface during LVSCCHK. Therefore, it is good practice to stop the job and fix any soft connection violations.

When you use the *U* option in SOFTCHK, this command outputs trapezoids whose node number contains the fewest number of trapezoids among those node numbers that are soft connected by the same region in the lower layer.

LINK *label*

Links floating substrates (*lower-layer*) to *label* (power or ground) unless the substrates are assigned text other than power or ground.

### Example 1

In this example, nodal information passes from *nsd* to *nwell* and not in reverse.

```
*OPERATION
.
.
AND NWELL NSD NPNWELL
AND PSUB PSD PINSUB
;
CONNECT METAL PSD BY CONTS
CONNECT METAL NSD BY CONTS
CONNECT METAL POLY BY CONTS
CONNECT SCMET METAL BY VIA
SCONNECT NSD NWELL BY NPNWELL
SCONNECT PSD PSUB BY PINSUB
;
SOFTCHK NWELL OUTPUT[U] ERR1 50
SOFTCHK PSUB OUTPUT[U] ERR2 50
;
```

### Example 2

This example links *NWELL* to *VDD* unless the *nwell* is assigned another text.

```
*DESCRIPTION
...
GEN-TEXT-FILE = GENTXT.OUT
GEN-TEXT-WIRE = YES
...
*END
*INPUT-LAYER
...
GEN-TEXT-LAYER = NWELL...METAL...
CONNECT-LAYER = NWELL ...METAL...
...
*END
*OPERATION
...
```

## Dracula Reference

### Operation Block Commands

---

```
SCONNECT METAL NWELL BY NWCONT LINK VDD
...
*END
```

### Example 3

This example demonstrates incremental connection. The connection is rebuilt because the AVSS label needs to be excluded from the IPDEF layer.

```
*DESCRIPTION
...
ALLOW-RECONNECT = YES
...
*END
*OPERATION
...
CONNECT ALMI IPF BY PFCON
CONNECT ALMI ADEF BY CWW
CONNECT ADEF IPDEF BY PDCT
CONNECT ADEF INDEF BY NDCT

SELECT IPDEF LABEL AVSS? xIPDEF
NOT IPDEF xIPDEF VSIPDEF
; connection rebuild starting point
CONNECT ALMI IPF BY PFCON
CONNECT ALMI ADEF BY CWW
CONNECT ADEF IPDEF BY PDCT
CONNECT ADEF INDEF BY NDCT
; connection rebuild end point
SCONNECT INDEF INWELL BY SSX
SCONNECT VSIPDEF PWELL BY PPX
SCONNECT VSIPDEF KIBAN BY KKX

SOFTCHK INWELL OUTPUT[U] NST 101 1
SOFTCHK PWELL OUTPUT[U] PST 102 1
SOFTCHK KIBAN OUTPUT[U] KST 103 1
...
*END
```

## SELECT

For the syntax for SELECT, see Option I through Option IX in the sections below.

### Description

Selects a group of polygons from a layer and creates a trapezoid file with the new layer. SELECT is an area-relation command. For example, if a polygon on *layer-a* is related to *layer-b*, the *layer-a* polygon is output. You can use this command to perform selective operations that cannot be done by logical operations, such as sizings, spacing checks, and conjunctive rules.

Nodal information associated with the selected polygons is passed to the trapezoid file. Database partition functions were added for the SELECT, SIZE, and LOGICAL commands. These enhancements resolve capacity limitation issues with Dracula. This allows you to partition input layers of these functions into blocks. When you run Distributed Dracula, each block (pair) can be assigned to one processor. Specify a value between 2-10 inclusive following 'D' to indicate the number of partitions desired. The default value of 'D' is 4 if the value is omitted.

### Note:

- There is a performance gain in the database partition feature for SELECT in the 4.8 version. The same is not true for SIZE and LOGICAL on account of too much overhead from the pre-process (PRESEL for SIZE) and post-process (LOGMRG for LOGICAL) stages during the operations. Database partition scheme is designed especially for resolving Dracula capacity issues and is applied to flat mode only. The performance of the SELECT command in the 4.8 version is equal to the 4.6.0698, 4.6.0998, and 4.6.1298 versions and is 5 to 20 times faster than the 4.7 release.
- In the 4.81 and 4.9 versions, Dracula provides a method to optimize original rules and create more optimized rules to speed up run time. In the description block commands section, you can add the command `optimize-rule=YES/NO` (default is NO) and speed up the run time. In the 4.81 and 4.9 Dracula versions, PDRACULA issues to you suggestions for optimizing your rule. See the section, "[Example 2 - Optimizing the Rule Using the SELECT Command](#)" for more information.

### Option I - Syntax for SELECT

```
SELECT[N] layer-a relation layer-b trapfile {OUTPUT c-name
l-num {d-num}}
```



## Dracula Reference

### Operation Block Commands

---

N

Turns on node-based rather than polygon-based selection. Selection is based on nodal numbers rather than region numbers. Specify this option after the CONNECT command in the operation block of a rule file. PDRACULA issues a warning message if the order in which you specified these commands is incorrect.

Returns all shapes on *layer-a* that

1. Satisfies the relation to *layer-b*
2. Are part of the same node as *layer-a* that was derived in part1.

*layer-a*

Layer to select from

*relation*

Can be one of the following:

INSIDE

Specifies *layer-a* polygon that is totally inside a *layer-b* polygon. Outputs the *layer-a* polygon and inside-touch *layer-b*.

OUTSIDE

Specifies *layer-a* polygon that is totally outside a *layer-b* polygon. Outputs the *layer-a* polygon and touch *layer-b*.

HOLE

Specifies *layer-a* polygon that has its entire outside surface exactly touching the outside surface of *layer-b* (for example, a *layer-a* polygon that exactly fits in the hole of a donut-shaped *layer-b* polygon). The selected *layer-a* polygons are outside and touch.

*layer-b*

Layer to relate to.

*trapfile*

Layer created by the SELECT operation. The *trapfile* can have up to seven characters and no special characters. You can use A-Z and 0-9, but the first character must be a letter.

OUTPUT

Sends the results of the operation to an output cell.

## Dracula Reference

### Operation Block Commands

---

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

### Option II

```
SELECT {[D{n}][N]} layer-a relation {[n1:n2]} layer-b trapfile {OUTPUT c-name
l-num {d-num}}
```

SELECT's D option (SELECT[D]) is now used for the database partition for SELECT. It can apply to all SELECT options except the following:

- SELECT[N].. ("N" option)
- SELECT .. LABEL
- SELECT .. CONN
- SELECT ANGLE

This option selects a range of instances that satisfy the selective condition. This option applies to the CUT, TOUCH, and ENCLOSE relationships.

D

Turns on the database partition feature

n

Number of blocks to be partitioned into

N

Turns on node-based rather than polygon-based selection. Selection is based on nodal numbers rather than region numbers. Specify this option after the CONNECT command in the operation block of a rule file. PDRACULA issues a warning message if the order in which you specified these commands is incorrect.

## Dracula Reference

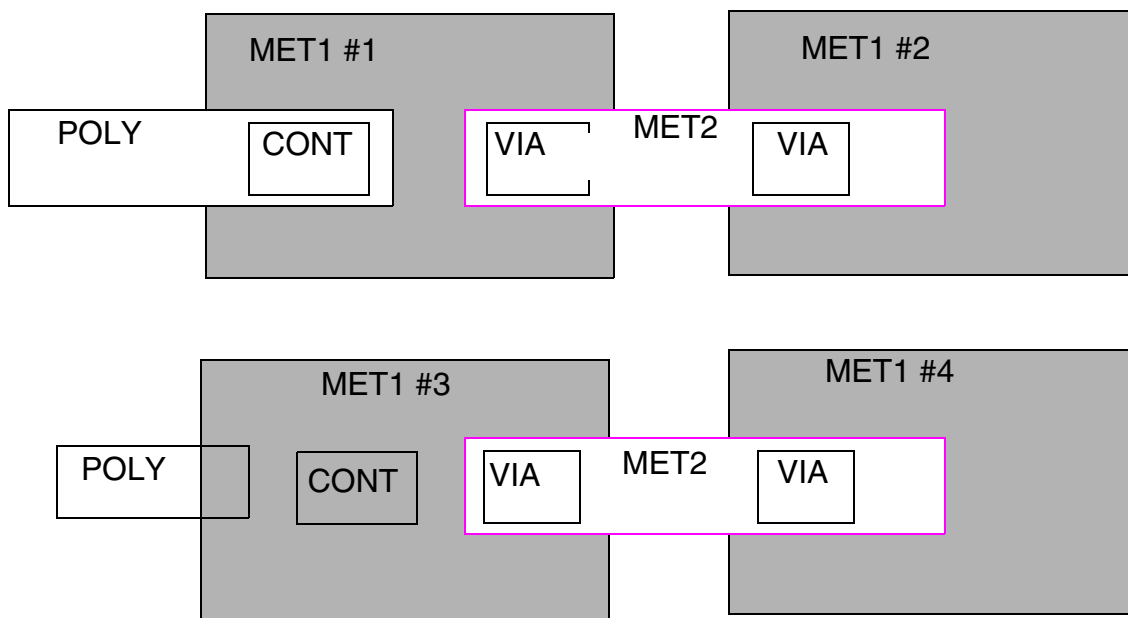
### Operation Block Commands

---

Returns all shapes on *layer-a* that

1. Satisfies the relation to *layer-b*
2. Are part of the same node as *layer-a* that was derived in part1.

*Example* (see the Figure that follows).



INPUT-LAYER

...

CONT = 5

MET1 = 6

VIA = 7

MET2 = 8

...

CONNECT-LAYER = POLY MET1 MET2

\*END

\*OPERATION

...

CONNECT MET2 MET1 BY VIA

CONNECT MET1 POLY BY CONT

...

SELECT[N] MET1 OVERLAP POLY M1NOVC OUTPUT M1NOVC 63

## Dracula Reference

### Operation Block Commands

---

```
SELECT MET1 CONN POLY M1CONC OUTPUT M1CONC 63
...
*END
```

The example illustrates the functionality and differences of SELECT[N] and SELECT .. CONN.

The SELECT[N] command will first include MET1 #1 and MET1 #3 into M1NOVC because they both overlap poly (the fact that MET1 #1 connects to poly with contact is merely coincidental). By virtue of the 'CONNECT MET2 MET1 BY VIA' command, MET1 #1 and MET1 #2 are part of the same net; the same is true for MET1 #3 and MET1 #4. Since MET1 #2 is part of the same node as MET1 #1, MET1 #2 will also be included in M1NOVC. MET1 #4 will also be included in M1NOVC because it is part of the same node as MET1 #3.

The SELECT ... CONN command will first include MET1 #1 into M1CONC because of the "CONNECT MET1 POLY BY CONT" command. MET1 #3 is not connected to POLY and is therefore not considered. By virtue of the "CONNECT MET2 MET1 BY VIA" command, MET1 #1 and MET1 #2 are part of the same node; MET1 #2 will also be included in M1CONC.

[*n1:n2*]

The range of instances. Outputs *layer-a* polygons that fall within the condition range of *layer-b*. The range values (*n1* and *n2*) are inclusive and must be integers. Zero is not allowed.

CUT

Specifies *layer-a* polygon that has portions both inside and outside *layer-b*.

TOUCH

Specifies *layer-a* polygon that touches only from the outside of a *layer-b* polygon. Does not include a *layer-a* polygon that both touches and cuts a polygon of *layer-b*.

ENCLOSE

Specifies *layer-a* polygon that encloses a polygon of *layer-b*. Includes a *layer-a* polygon that is inside and touching *layer-b*.

OVERLAP

Specifies *layer-a* polygon that overlaps (including TOUCH, CUT, ENCLOSE, and INSIDE) a polygon of *layer-b*. The overlap relation includes all relationships except for those of *layer-a* polygons entirely outside of *layer-b* polygons.

## Dracula Reference

### Operation Block Commands

Currently, the SPACING module does not have a database partition feature. However, since `EXT[E]` is translated into `SELECT` by PDRACULA, it is possible to have a data partition for `EXT[E]`. `EXT's D` option must combine with the `E` option (`EXT[DE]`).

#### SELECT Database Partition (Distributed Dracula on 2 Nodes)

|                   | Total                | PRESEL         | POSCAT         | SELECT   |
|-------------------|----------------------|----------------|----------------|----------|
| <b>SELECT</b>     | 00:37:57             | Not Applicable | Not Applicable | 00:37:57 |
|                   | 00:29:27<br>(+1.29x) | 00:06:17       | 00:02:39       | 00:05:56 |
| <b>SELECT[D]</b>  |                      |                |                | 00:08:27 |
|                   |                      |                |                | 00:15:23 |
|                   |                      |                |                | 00:11:06 |
| <b>SELECT[D2]</b> | 00:33:36<br>(+1.13x) | 00:06:07       | 00:02:28       | 00:13:25 |
|                   |                      |                |                | 00:24:54 |
| <b>SELECT[D3]</b> | 00:32:35<br>(+1.16x) | 00:06:13       | 00:02:31       | 00:07:54 |
|                   |                      |                |                | 00:15:35 |
|                   |                      |                |                | 00:15:51 |

**Note:** In version 4.7.0399 and later versions, the original `EXT[TOE]` needs two `SELECT` and one `SPACING`.

`EXT[TOE] = EXT` with `[T]`, `[O]`, and `[E]` options turned to `on`

where *T* represents touch, *O* represents overlap, and *E* represents fully enclosed. `SELECT` is slow. The flow in the 4.7.0399 version uses one `AND` and one `SPACING`. `AND` is faster than `SELECT`.

### Option III

```
SELECT layer-a LABEL{[R]} label-name trapfile {OUTPUT c-name l-num {d-num}}
```

This option selects all *layer-a* polygons that are part of the *label-name* texted net. You must establish nodal connectivity on *layer-a* with the `CONNECT`, `SCONNECT` or `STAMP` command before using this option.

## Dracula Reference

### Operation Block Commands

---

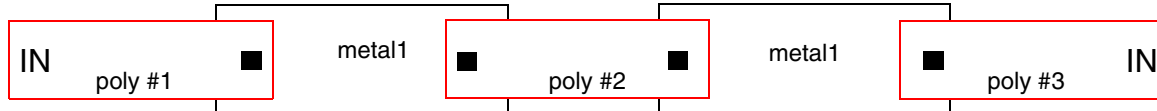
[R]

Selects all *layer-a* polygons that physically attach to the *label-name* text that establishes the net. You must specify the `ATTACH` command for *layer-a* in the INPUT-LAYER block.

*label-name*

The text label of the net to check.

### Example



```
*INPUT-LAYER
...
poly = 5
contact = 6
metall = 7
text = 10 attach = poly
...
CONNECT-LAYER = poly metall
*END
*OPERATION
...
CONNECT poly metall BY contact
...
SELECT poly LABEL IN layerA
SELECT poly LABEL[R] IN layerB
...
*END
```

The `LABEL` operator will include all three poly shapes (poly #1, poly #2 and poly #3) into layerA because they are all part of the texted net `IN`.

The `LABEL[R]` operator will only include the poly #1 and poly #3 shapes into layerB because the `IN` text attaches only to these two shapes.

## Option IV

```
SELECT layer-a VERTEX [n1:n2] trapfile {OUTPUT c-name
 {d-num}}
```

This option selects all *layer-a* polygons that satisfy the vertex count. SELECT counts the vertices after merging the polygon vertices. SELECT counts double digitized vertices only one time.

VERTEX

A *layer-a* polygon that satisfies the vertex count range. A rectangular polygon has four vertices.

[*n1:n2*]

The range of vertex counts. Outputs *layer-a* polygons that fall within the count range. The range values (*n1* and *n2*) are inclusive and must be integers. Zero is not allowed.

## Option V

```
SELECT layer-a ANGLE [n1] output-layer {OUTPUT c-name l-num
 {d-num}}
```

This option lets you separate all-angle geometries from Manhattan geometries.

*n1*

Specifies which angled geometries to separate. You can specify one of the following options:

- |     |                                                                                                                              |
|-----|------------------------------------------------------------------------------------------------------------------------------|
| 90  | For manhattan geometries. (All edges are either vertical or horizontal.)                                                     |
| 45  | For 45-degree geometries. (All edges are either 45 degrees, or edges are manhattan and contain at least one 45 degree edge.) |
| -90 | For non-manhattan geometries. (At least one edge is not vertical or horizontal.)                                             |
| -45 | For non-45 degree geometries. (Angles are other than manhattan and 45 degrees.)                                              |

## Option VI

```
SELECT layer1 BY layer2 LABEL([t] | [t']) (label|label-list) outLayer {OUTPUT
 c-name l-num {d-num}}
```

## Dracula Reference

### Operation Block Commands

---

Use this command when you specify the IDTEXT command in the Input Layer block.

*layer1*

Layer name from which objects are selected.

*layer2*

IDTEXT layer name.

*t*

Selects objects texted by the given labels.

*t'*

Selects objects that are not texted by the given labels.

*label*

Any label, which can include meta characters such as ?.

*label-list*

A list of up to a maximum of 20 labels (*{label...}*).

*outLayer*

Name of the output layer.

### Option VII

```
SELECT {NOT} layer-a relation layer-b trapfile {OUTPUT c-name
l-name {d-name}}
```

This option lets you output the complemented outputs without using another NOT command.

NOT

Use this command to provide complemented output with the single rule.

### Option VIII

```
SELECT layer-a CONN layer-b output-layer {OUTPUT c-name l-name {d-num}}
```

This option lets you select geometric based on layer connectivity.

CONN

Output *layer-a* polygons that are connected to *layer-b*. The output is a subset of *layer-a*.



## Dracula Reference

### Operation Block Commands

---

#### Option IX

```
SELECT layer-a NODE schematic-name output-layer
```

This option selects *layer-a* layer geometries that are cross referenced to specified schematic names. LVCHK is necessary before you apply this command.

```
NODE schematic-name
```

Name of the specified schematic

#### Examples

##### Example 1

This example selects all *nplus* polygons with an annotated text string beginning with the characters *hv*. Outputs these polygons on *hvnplus*.

```
SELECT nplus INSIDE pwell nsrdrn
SELECT nplus OUTSIDE pwell nguard
SELECT nplus LABEL hv? hvnplus
```

This example selects *metal* polygons with annotated pads. Outputs these polygons on *pads*.

```
SELECT metal LABEL ?:? pads
```

This example selects transistors that have one source/drain.

```
SELECT gate TOUCH[1:1] nsrdrn cap
```

This example selects a *poly* polygon with *res* text attached as resistor layer, *polres*.

```
SELECT poly LABEL[R] res? polres
```

In the following example, the *gdcont* layer contains quadrilateral contacts.

```
SELECT contact VERTEX[4:4] gdcont
```

In the following example, the *bad1* and *drc317* layers contain contacts with three or fewer vertices.

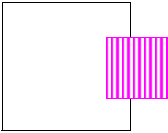
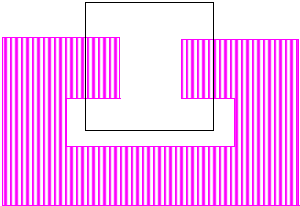
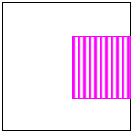
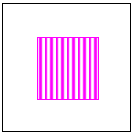
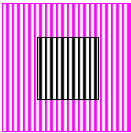
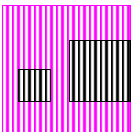
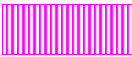
```
SELECT contact VERTEX[1:3] bad1 OUTPUT drc317
```

In the following example, the *bad2* and *drc418* layers contain contacts with more than four vertices.

**Dracula Reference**  
Operation Block Commands

---

SELECT contact VERTEX[5:999] bad2 OUTPUT drc4 18

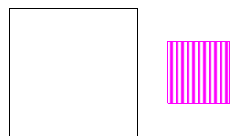
| Case                                                                                     | Relation                                                                                 |
|------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
|         | CUT<br>CUT[1:1]<br>OVERLAP<br>OVERLAP[1:1]                                               |
|         | CUT<br>CUT[1:1]<br>OVERLAP<br>OVERLAP [1:1]                                              |
|         | INSIDE<br>OVERLAP<br>OVERLAP [1:1]                                                       |
|        | INSIDE<br><br>OVERLAP                                                                    |
|       | ENCLOSE<br>ENCLOSE[1:1]<br>OVERLAP<br>OVERLAP [1:1]                                      |
|       | ENCLOSE<br>ENCLOSE[2:2]<br>OVERLAP<br>OVERLAP [2:2]                                      |
|  Lay1 |  Lay2 |

**Dracula Reference**  
Operation Block Commands

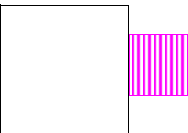
---

Case

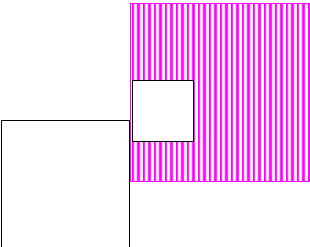
Relation



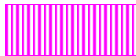
OUTSIDE



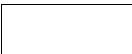
OUTSIDE  
TOUCH  
TOUCH[1:1]  
OVERLAP  
OVERLAP [1:1]



OUTSIDE  
TOUCH  
TOUCH[1:1]  
OVERLAP  
OVERLAP [1:1]



Lay1



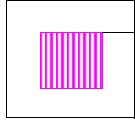
Lay2

## Dracula Reference

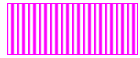
### Operation Block Commands

---

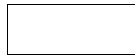
#### Case Relation



HOLE  
OUTSIDE  
TOUCH  
TOUCH[1:1]  
OVERLAP  
OVERLAP [1:1]



Lay1



Lay2

In the following example POLY is selected and cross-referenced to the schematic name VDD\* to output layer POLY1.

```
*OPERATION
...
LVSCHK
...
SELECT POLY NODE VDD* POLY1.....
*END
```

#### ***Example 2 - Optimizing the Rule Using the SELECT Command***

You can optimize the following two rules

```
SELECT layer-a relation layer-b layer-c
NOT layer-a layer-c layer-d
```

(and layer-c is not used for the other rule)

to one rule using the following command

```
SELECT NOT layer-a relation layer-b layer-d
```

The above optimization will save you the run time for one NOT command.

#### **SELECT Command with CONN**

In previous versions of the tool, if you used `SELECT layA CONN layB layC` and `layA` or `layB` did not have nodal information, the SELECT stage terminated.

## Dracula Reference

### Operation Block Commands

---

In the 4.8 version of Dracula, you can set the environment variable *DRAC\_SELCONN\_NOTABORT*. **SELECT CONN** runs without terminating the process. The trapezoid without the nodal information is output to the log file.

## SHRINK

```
SHRINK input-layer dx dy {dx2 dy2} {trapfile} {OUTPUT c-name l-num {d-num}}
```

### Description

Undersizes a single side of a layer. The *dx* and *dy* values determine the direction and amount the data shrinks.

Refer also to the GROW command description in this chapter.

### Arguments

*input-layer*

Input layer name.

*dx dy*

Amount to undersize the layer. This value is a real number in microns or mils. This is the unit set by SCALE.

|                    |                     |
|--------------------|---------------------|
| If <i>dx</i> > 0.0 | Right edge shrinks  |
| If <i>dy</i> > 0.0 | Top edge shrinks    |
| If <i>dx</i> = 0.0 | No sizing           |
| If <i>dy</i> = 0.0 | No sizing           |
| If <i>dx</i> < 0.0 | Left edge shrinks   |
| If <i>dy</i> < 0.0 | Bottom edge shrinks |

*dx2 dy2*

Specifies the second set of direction. All values must be positive.

*trapfile*

Trapezoid file name.

*c-name*

Name of the output cell. You can optionally output the sized data to a cell.

*l-num*

Layer number of the output cell.

## Dracula Reference

### Operation Block Commands

---

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell.

#### Example

```
*DESCRIPTION
SCALE = 0.001 MIC
RESOLUTION = 0.001 MIC
MEBES-UNIT = 0.25 MIC
*END
*OPERATION
...
SHRINK layer3 1 1.5 layer4 ;Shrink right edge 1 micron.
... ;Shrink top edge 1.5 micron.
*END
```

## Dracula Reference

### Operation Block Commands

---

## SIZE

```
SIZE {[options]} input-layer BY n1 {TRUNCATE {trunc_value}} {trapfile} {OUTPUT c-
name l-num {d-num}}
```

```
SIZE {[options]} input-layer BY n1 STEP {step_value/AUTO} {TRUNCATE {trunc_value}}
{output-layer} {trapfile} {OUTPUT c-name l-num {d-num}}
```

```
SIZE {[options]} input-layer-a {WITHIN input-layer-b} BY n1 [STEP step_value]
{TRUNCATE {trunc_value}} {trapfile} {OUTPUT c-name l-num {d-num}}
```

```
SIZE[D{n}] input-layer BY n {STEP step-value} {TRUNCATE {trunc_value}} {trapfile}
{OUTPUT c-name l-num {d-num}}
```

```
SIZE input-layer DOWN-UP BY n1 {TRUNCATE {trunc_value}} {trapfile} {OUTPUT c-name
l-num {d-num}}
```

## Description

Creates a new layer from another layer. This command oversizes or undersizes the new layer by a specified number of layout units and moves the sides of the polygon perpendicularly. The numerical sign determines whether to oversize (+) or undersize (-) the polygon.

Database partition functions were added for the SELECT, SIZE, and LOGICAL commands. These enhancements resolve capacity limitation issues with Dracula. This allows you to partition input layers of these functions into blocks. When you run Distributed Dracula, each block (pair) can be assigned to one processor. Specify a value between 2-10 inclusive following 'D' to indicate the number of partitions desired. The default value of 'D' is 4 if the value is omitted.

## Note:

- There is a performance gain in the database partition feature for SELECT in the 4.8 version. The same is not true for SIZE and LOGICAL because of too much overhead from the pre-process (PRESEL for SIZE) and post-process (LOGMRG for LOGICAL) stages during the operations. Database partition scheme is designed specially for resolving Dracula capacity issues and is applied to flat mode only. Currently, the database partition feature for SIZE does not support other options.

SIZE does not output text information.

When sizing databases with layers digitized with arbitrary angles (circles) or when sizing databases with cell placements not on fixed grids (some auto-routers), set the RESOLUTION command in the Description block at 0.005 microns or less. This setting alerts the SIZE command to handle round-off problems and to effectively "lock" the adjacent line segments.



## Dracula Reference

### Operation Block Commands

---

The `WITHIN` option works only in flat mode. It supports step-wise sizing to avoid significant round-off effects resulting from large sizing and to improve `SIZE` performance for large values.

When using the `SIZE WITHIN` option, a contact can change size and fill the space of any portion of the shape it is within, including ring (donut) shapes, U shapes or L shapes. The contact can "jump over" any gap (notch) that exists for any single area (contacts cannot jump over to adjacent but different areas). If you have ring, U, or L shapes, it is best to use the `STEP x` option with the `WITHIN` option. The value of the `STEP` size `x` should be less than the minimum external notch spacing of the ring or U shape so that each step of the sizing cannot jump across the gap.

- When using the `SIZE` command on a 45-degree angle, oversizing and undersizing might produce a different shape than the original. When you oversize a 45-degree angle, it snaps to the grid, which might modify the shape. If you then undersize this layer, it will snap to the grid again, resulting in a different shape than you started with.
- In the 4.81 and subsequent versions, Dracula provides a method to optimize original rules and create more optimized rules to speed up run time. In the description block commands section, you can add the command `optimize-rule=YES/NO` (default is `NO`) and speed up the run time. In the 4.81 and 4.9 Dracula versions, `PDRACULA` issues to you suggestions for optimizing your rule. See [Example 13 - Optimizing the Rule Using the `SIZE` Command](#) of this section.
- When you size non-orthogonal polygons, set the value of the `RESOLUTION` command to the same value as that of the `SCALE` command; otherwise, Dracula may distort the resulting shape. For example, if `SCALE = 0.001 MICRON`, the `RESOLUTION` command should also be set to `RESOLUTION = 0.001 MICRON`.

If you need to know if any polygons or cell placements are off-grid, add this command:

```
FLAG-OFFGRID = YES value
```

where *value* is the grid value in the Description block.

Here is an example:

```
FLAG-OFFGRID = YES 0.01
```

Do not specify `MICRON` or `MIL` after the grid value, since the scale factor is already implied in the `SCALE` command.

## Arguments

B

Adjusts the width of error flags that result from spacing checks with the *R'* option.

## Dracula Reference

### Operation Block Commands

---

D

Turns on the database partition feature.

n

Represents the number of blocks to be partitioned.

E

Sizing only for edges as projections. This option is used with the SPACING[D] option, which outputs error flags in trapezoid format that contain information for edge sizing later on. Edge size is only supported at flatten mode, and can only produce trapezoid output, not polygon output.

L

Adjusts the length of error flags that result from spacing checks with the *R*' option.

N

Prevents polygon data output from the SIZE command from being included in the OUTDISK file. Use this option only if you want to fracture *sized* polygon data and do not want this data included in the OUTDISK file. Use this option only when you specify an *output* layer.

O

(Oversize only) Sizes the layer but does not merge the sized data. Generates only the sized geometries that overlap. You can specify only the *trapfile* name.

When oversizing a layer, some polygons might extend into each other. SIZE forms one “clean polygon” from the potential overlap. When undersizing a layer, polygons with complex shapes tend to separate. SIZE does not create gaps on polygon cut lines, but a gap can result if part of the polygon width undersizes to 0 or less. This command prevents polygons from turning inside-out.

P

(Undersize only) Sizes the layer as “pre-merged” data (as opposed to “merged” data). Therefore, the abutted polygons generate gaps where previously abutted. Apply the *P* option to the original layer only, not to the derived layer. When specifying the *P* option, input and output layers must be different; otherwise, conflicts might arise in later sizing operations.

R

*R* (Undersize only) Reverses sizing capability. When doing undersize, if the sizing dimension causes the geometry width to go down to zero, this option allows it to reverse direction. This option works for manhattan shapes only. Result layer will contain processed manhattan shapes from input layer. All non-manhattan shapes will generate

## Dracula Reference

### Operation Block Commands

---

new layer which name will combined from original layer name and "-NM". E.g. for "M1" layers non-manhattan shapes will form "M1-NM" layer.

T

Outputs trapezoid shapes to SIZE output file, not a polygon file. If you specify T, you must specify OUTPUT. If you specify T, you cannot specify any other option (A, B, G, and so on).

W

Specifies a minimum polygon cut-line width of 0.5 microns instead of the default of 1 micron. Use this command option when sizing very dense layers such as *diffusion*, *poly*, or *metal* in a MOS memory layout. For more information, refer to the SIZE-MIN-WIDTH command in [Chapter 11, "Description Block Commands."](#)

*input-layer*

Specifies the input layer name.

TRUNCATE

Specifies that the sizing of input layers truncates acute angles. Sizing by positive value means cutting convex angles. Sizing by negative value means cutting concave angles.

*trunc\_value*

Optional value, that determines maximum distance of acute-angles that will not be truncated. The truncation distance =  $n1 * \text{trunc\_value}$ . The default value is  $\text{sqrt}(2)$ . (It means that all angles less than 45 degrees will be truncated)

WITHIN

Specifies that the sizing of input-layer-a is limited to the enclosing input-layer-b. This function efficiently checks latch-up problems.

*n1*

Number of layout units to oversize or undersize the *input-layer*.

STEP

Specifies that the sizing of the *input-layer* is to be step-wise, performing a series of size operations using *step\_value* until the overall distance *n1* is reached. When you specify a SIZE WITHIN operation, a logical AND operation with *input-layer-b* is followed by the size operation.

If you are sizing over 10 microns and you do not use STEP, huge vertex polygons are created, which cause slow performance and undesirable sizing.

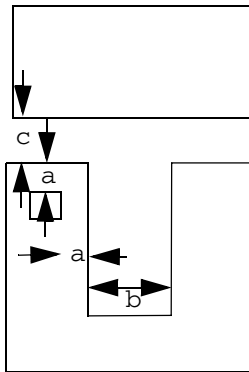
## Dracula Reference

### Operation Block Commands

---

`STEP step_value`

Dimension by which to size in each STEP. Specifying this argument splits the SIZE function into a number of SIZE and logical AND functions. These functions are internally performed by Dracula to increase the accuracy of sizing for U-shaped geometries.



$\text{step\_value} = a + (\text{smaller of } b \text{ or } c)$

`AUTO`

When you specify STEP AUTO, the value of STEP is automatically adjusted according to your rules file. Use AUTO for large value sizing only (sizing over 10 microns).

`trapfile`

Trapezoid file created by the resizing operation.

`OUTPUT`

Sends the results of the operation to an output cell.

`c-name l-num`

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

The output cell (*c-name*) usually generates three cells that contain the total sized data. The data is in PG-compatible polygon format. The three cells are suffixed as follows:

- A Contains the sized data with acute angles. *C-name* generates this cell only if there are sized polygons with acute angles.
- G Contains the sized data created by the first and second passes of the SIZE operation.

## Dracula Reference

### Operation Block Commands

---

**R** Contains the sized data not in the cells suffixed with *G* and *A*. Sized data in this cell are questionable and require inspection. However, the data are good for most PG-compatible systems. Usually your data sizes correctly and there are no *R* cells.

This conservative SIZE algorithm checks the sized data for the following:

- Acute angle
- Re-entrant
- Width too narrow (default is less than 1 micron)

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

DOWN-UP

Keyword implemented to filter the smaller redundant parts of a big polygon. First of all, Dracula cuts the polygon into pieces by both vertical and horizontal directions according to each vertices' coordinations, and then calculates the width of each piece to see if it will remove it or not.

This option helps to extract big geometries that satisfy a specific width condition. Two similar ways to achieve this functionality is to either:

Use the WIDTH check with the [R] option to output the required segments. Refer to the WIDTH[R] command in this chapter for more information.

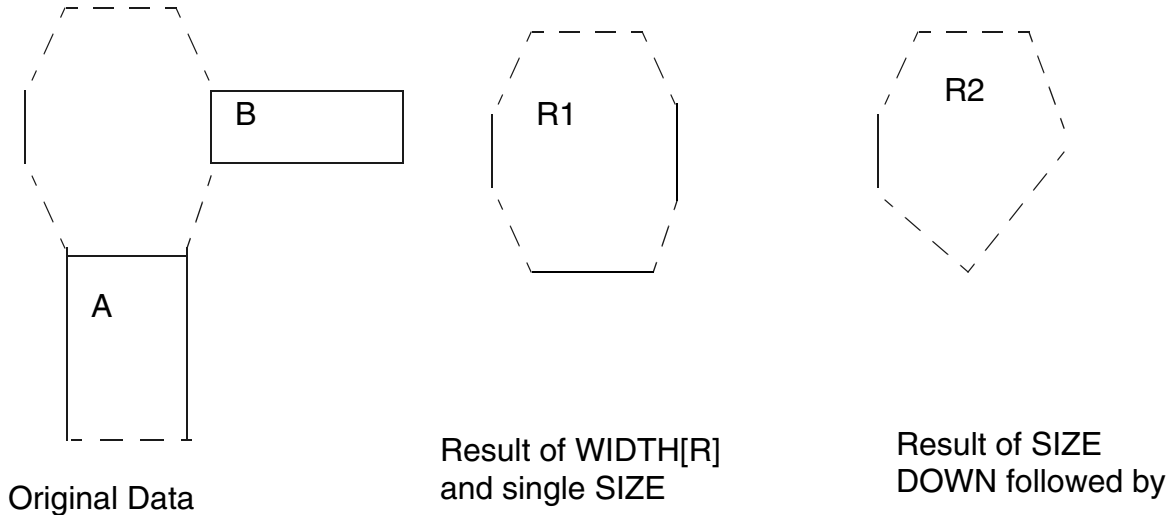
Do a SIZE-DOWN operation, followed by a SIZE-UP operation. This method is more accurate, but introduces performance problems with dense layers like *METAL*. This option is limited to FLAT mode only.

## Dracula Reference

### Operation Block Commands

---

The result of a single SIZE DOWN-UP is not intended to simulate that of a SIZE DOWN followed by a SIZE UP operation. Actually, the output is more like a WIDTH check with the [R] option. For example,



The result of R1 is more preferable than R2 because two consecutive SIZE operations will probably give rise to acute angles and cause false errors in subsequent checks.

For example, the following rules used in one case helped to eliminate false errors caused by acute angles induced by two SIZE DOWN-UP commands:

```

WIDTH[R] MT1 LT 10.0 MT1L10
NOT MT1 MT1L10 MT1G10
SIZE MT1G10 BY -4.95 UM1G10
SIZE UM1G10 BY 4.95 OM1G10
EXT OM1G10 MT1 LT 1.3 OUT D6D 53 ;

```

If MT1 is a very dense layer, it takes longer in the above example to do the WIDTH[R] operation. But, if a single SIZE DOWN-UP command is used it takes hardly 30 seconds and the speed is optimized by 160 times with the result being almost as accurate as MT1G10 without inducing any acute angle.

## Dracula Reference

### Operation Block Commands

#### SIZE Database Partition (Distributed Dracula on 2 Nodes)

|          | Total                | PRESEL         | POSCAT         | SIZE     |
|----------|----------------------|----------------|----------------|----------|
| SIZE     | 00:29:11             | Not Applicable | Not Applicable | 00:29:11 |
| SIZE[D]  | 00:45:09<br>(-1.55x) | 00:28:08       | 00:05:36       | 00:05:57 |
|          |                      |                |                | 00:05:45 |
|          |                      |                |                | 00:05:39 |
|          |                      |                |                | 00:05:30 |
| SIZE[D2] | 00:43:39<br>(-1.50x) | 00:27:19       | 00:05:14       | 00:11:13 |
| SIZE[D3] | 00:44:03<br>(-1.51x) | 00:25:44       | 00:05:24       | 00:10:31 |
|          |                      |                |                | 00:07:18 |
|          |                      |                |                | 00:06:48 |
|          |                      |                |                | 00:06:03 |

#### Example 1

This example sizes all *poly* data, outputs it to the *unpoly* layer, and creates *upoly30G*, *upoly30A* (sometimes), and *upoly30R* (rarely).

```
SIZE poly BY -.1 unpoly OUTPUT upoly 30
```

#### Example 2

In this example, the **T** option specifies that output cell **OMETALIO.DAT** is a polygon file that contains trapezoid shapes instead of a normal polygon file.

```
SIZE[T] METAL1 BY 5 OMETAL OUTPUT OMETAL 10
```

#### Example 3

This example sizes *cont* layer shapes by 25 microns, but keeps the resulting *cont* shapes within their enclosing *met1* shapes.

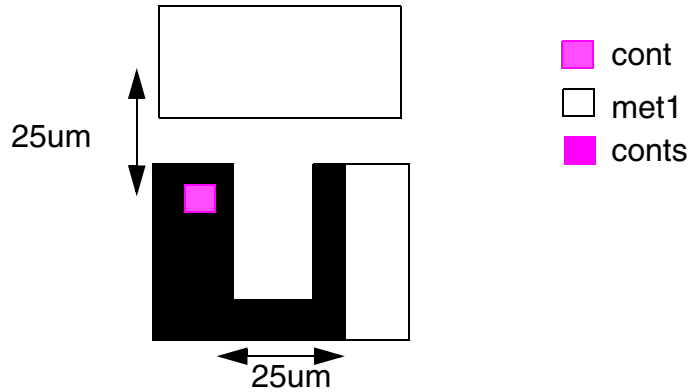
```
SIZE cont WITHIN met1 BY 25.0 conts
```

## Dracula Reference

### Operation Block Commands

---

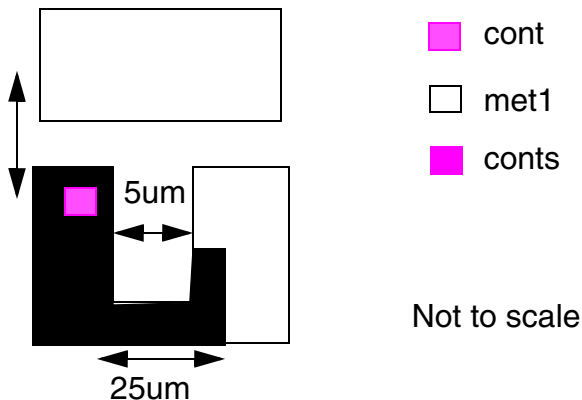
**Note:** This option sizes across a notch in the *met1* layer by the full amount.



#### Example 4

The following example sizes the *cont* layer shapes by 25 microns in steps of 4.9 microns, keeping the resulting shape within the enclosing *met1* shape at each step. This results in growing around the notch (rather than skipping over it), but this takes Dracula longer to execute than the method used in Example 3. The value of 4.9 microns is chosen to be just less than the minimum *met1* notch allowed in the process. Therefore, within any one step, the sized shape cannot skip across the notch.

```
SIZE cont WITHIN met1 BY 25.0 STEP 4.9 conts
```





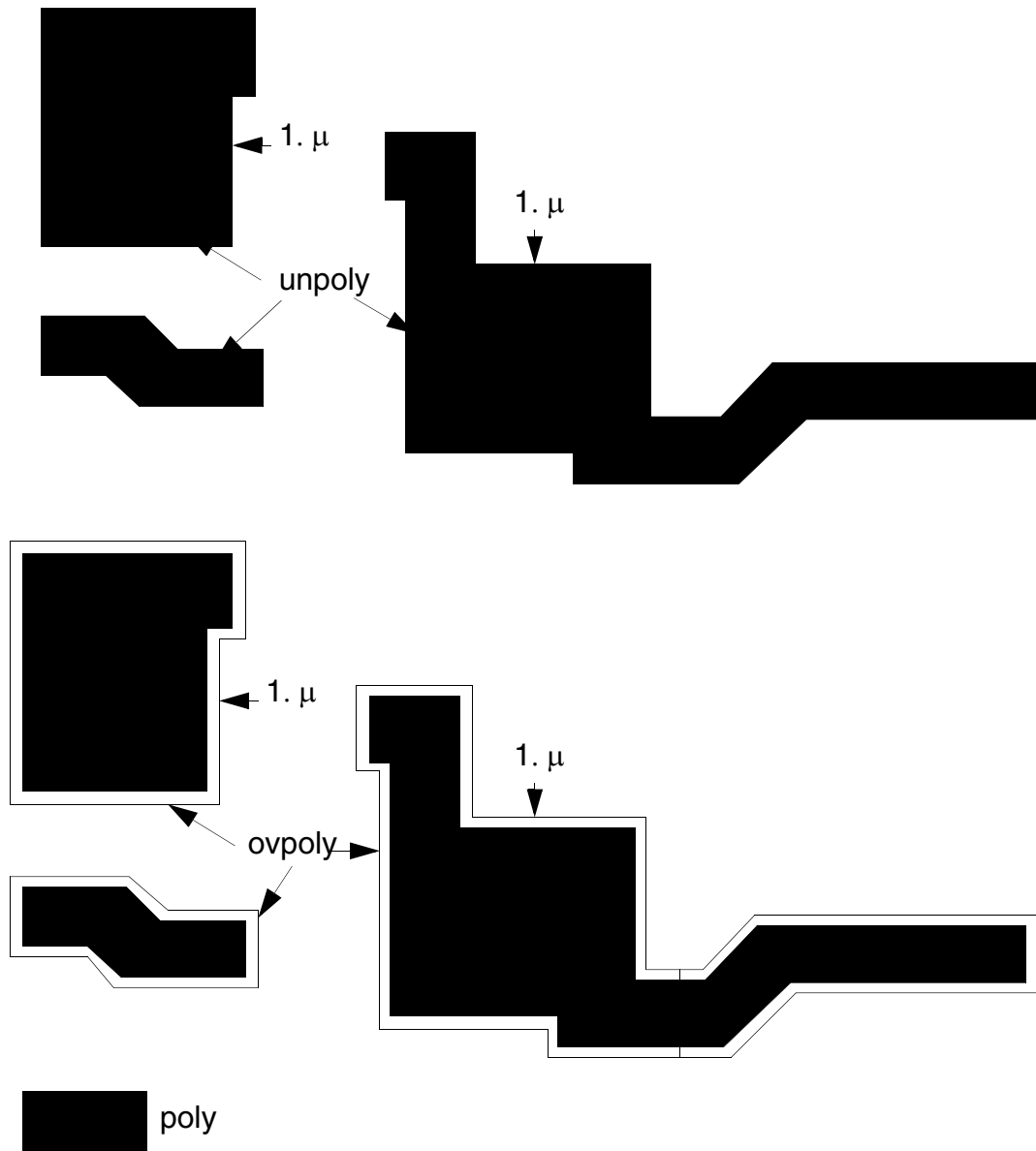
## Dracula Reference

### Operation Block Commands

---

#### Example 5

```
SIZE poly BY -1 unpoly
SIZE poly BY 1 ovpoly
```



#### Example 6

After the logical operation, this example sizes the layer gate by 0.

```
AND dif poly gate
SIZE gate BY 0 OUTPUT gate 16
```

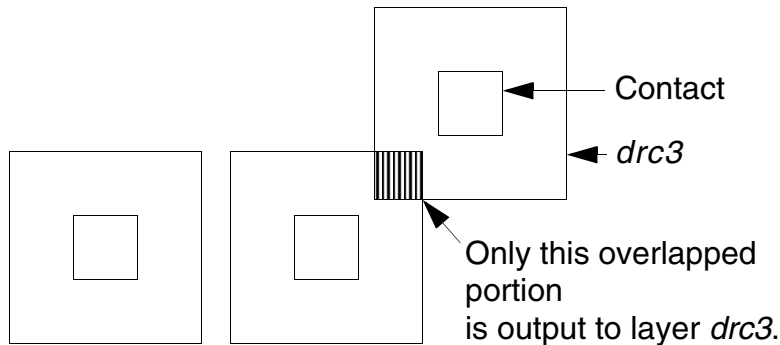
## Dracula Reference

### Operation Block Commands

---

#### Example 7

```
SIZE[0] contact BY 2 drc3
```



#### Example 8

When sizing layers from technologies that contain small geometries (2 microns or less), you can specify the SIZE-MIN-WIDTH command in the Description block. In this example, the SIZE command uses minimum polygon cut-line widths of 0.005 microns.

```
*DESCRIPTION
SIZE-MIN-WIDTH=0.005 mic
.
.
*END
*OPERATION
SIZE metal BY -.2 OUTPUT umetal 53
*END
```

If you want a 10X reticle and you are using an optical system, you must specify a minimum value of 0.4 microns for SIZE-MIN-WIDTH because 4 microns is the minimum flash width.

#### Example 9

The SIZE-MAX-VERTEX default value is 200 vertices. Use the default only with a CAD layout system that accepts PG-compatible polygon formats with less than 200 vertices. In this example, the maximum number of vertices is 50 when building polygons on layer *nbur54*.

```
*DESCRIPTION
SIZE-MAX-VERTEX=50
.
.
*END
*OPERATION
SIZE buried BY .2 OUTPUT nbur 54
*END
```

## Dracula Reference

### Operation Block Commands

---

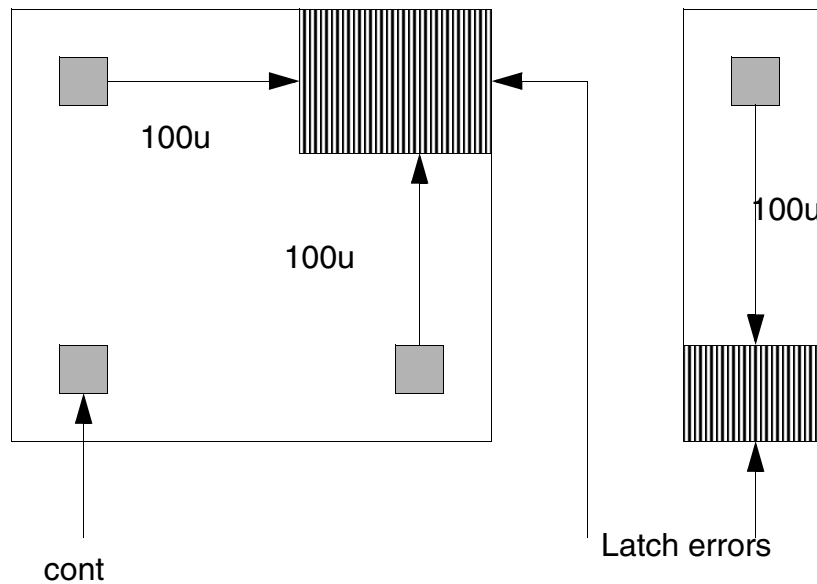
#### Example 10

```
SIZE[WN] poly BY 1 OUTPUT ovpoly 46 ; to output PG compatible
; ; polygon format data to
; ; a CAD system
SIZE met BY .25 omet
```

#### Example 11

This example checks a well region that contains a contact every 100 microns.

```
SIZE cont WITHIN well by 100 cont1
NOT well cont1 latcherr
```



#### Example 12

```
;find metal 1 with width greater than 15 microns
SIZE metall DOWN-UP by 7.5 bigmet15
```

#### Example 13 - Optimizing the Rule Using the SIZE Command

In Dracula 4.81 and higher versions, you can speed up the run time by replacing

- Oversize or undersize sequence to “SIZE up-down” commands
- Undersize or oversize sequence to “SIZE down-up” commands

## Dracula Reference

### Operation Block Commands

---

- If you have set `optimize-rule=yes` in your description block and if your statements in your rule file are as shown below:

```
SIZE L1 by 0.1 T1
SIZE t1 by -0.1 L2
```

then PDRACULA will suggest to you to replace these above 2 statements by the following statement if you want to optimize your rule:

```
SIZE L1 up-down by 0.1 L2
```

- If you have set `optimize-rule=yes` in your description block and if your statements in your rule file are as shown below:

```
SIZE L1 by -0.1 T1
SIZE t1 by 0.1 L2
```

then PDRACULA will suggest to you to replace these above 2 statements by the following statement if you want to optimize your rule:

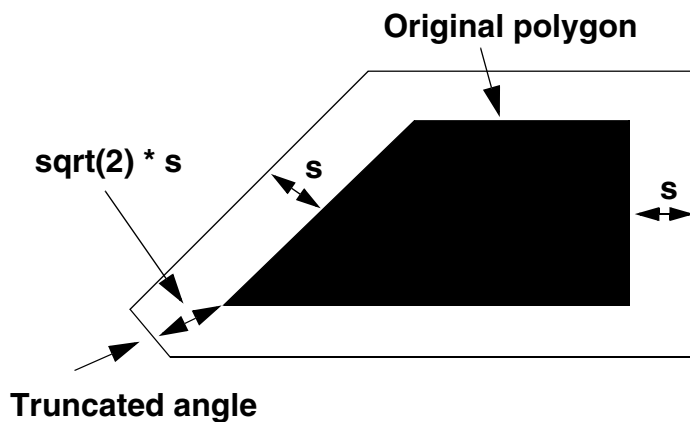
```
SIZE L1 down-up by 0.1 L2
```

#### Example 14 -

```
SIZE L1 by <s>* TRUNCATE 1.41 L2
```

where `<s>` is the sizing distance.

This will size the following figure:



## SNAP

`SNAP{[P]} layer-a TO grid-value {trapfile}`

### Description

Snaps the coordinates of the geometry to a specified grid and outputs the result to another layer.

### Arguments

*P*

Snaps the coordinates of the geometry to a specified grid in a polygon file. The default is to snap the coordinates of the geometry to a specified grid in a trapezoid file.

*input-layer*

Input layer name. If *P* is specified, the *input-layer* must be a polygon file.

*grid-value*

Grid to which to snap the layer. The value must be the multiple of the RESOLUTION specified in the Description block.

*trapfile*

Output layer name. The *trapfile* can have up to seven alphanumeric characters and no special characters. The first character must be a letter. If *P* is not specified, the *trapfile* name must be specified.

All coordinates of the input-layer are snapped to the nearest grid. The middle point is snapped in the positive direction.

### Example

```
*DESCRIPTION
 RESOLUTION = 0.005 mic
*END
*OPERATION
SNAP YYPOLY TO 0.1 ZZPOLY
SIZE ZZPOLY BY 0.0 OUTPUT PPOLY 4
SNAP[P] PPOLY04 TO 0.1
```

## SOFTCHK

SOFTCHK *lower-layer* {*trapfile*} OUTPUT[U] *c-name* *l-num* {*d-num*}

### Description

Reports regions on a specified layer that connect to more than one node.

### Arguments

*lower-layer*

Layer defined in SCONNECT that you are checking and from which you need to generate the soft connection violation outputs.

**Note:** The internal file naming convention is the input layer name prefixed with "S\_". Do not specify a layer name (for example, NWELL) and the S\_ prefixed version (S\_NWELL) as it might result in a layer names conflict in which user data is overwritten by intermediate SOFTCHK results.

*trapfile*

Name you designate as the output file to contain composite trapezoids. Before you do any operation on a *trapfile*, you need to either "OR" the layer with itself or "AND" the layer with the substrate layer.

OUTPUT

Sends the results of the operation to an output cell.

A

Outputs an *upper\_layer* file containing all geometries.

U

Outputs an *upper\_layer* file containing the errors Dracula selects.

*c-name* *l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

## Dracula Reference

### Operation Block Commands

---

If there is more than one connection, the soft connection check region is connected by only the contacts whose node number contains the most number of trapezoids among the node numbers of the multiple contacts. It is best to fix the violations before continuing the job.

All involved layers, except the specified layer, must have nodal information before any soft connection check can be made.

Soft connection violations are reported in the *.ERC* file. In addition, the *U* option generates a graphic output file. This file has the trapezoids on the layers connecting the *upper\_layer* and *lower\_layer*. The file name is as specified in OUTPUT but preceded by *U*.

### Example

```
CONNECT-LAYER = NSUB PWELL PSD NSD POLY METAL MET2
.
.
.
CONNECT MET2 METAL BY VIA
CONNECT METAL POLY BY CONT
CONNECT METAL PSD BY CONT
CONNECT METAL NSD BY CONT
SCONNECT PSD PWELL BY CONT
SCONNECT NSD NSUB BY NCONT
.
.
.
SOFTCHK PWELL OUTPUT[U] DSOFT 50
```

## SPFSELECT

```
SPFSELECT{[opt]} element{[type]}{REL val-a} {RANGE val-a val-b} {CORNER = n1}
 {OUTPUT c-name {l-num}} {&}
```

### Description

Generates a detailed standard parasitic file (DSPF) for HPRE. For more information about the DSPF format, refer to [DSPF Format](#) on page 496

Argument definitions for the SPFSELECT command are the same as for the LPESELECT command except that you cannot specify a *type* if you specify a CAP element in your run file.

To generate a DSPF file according to DFII instance or net naming conventions, specify the SPF-MAP-FILE command in the Description block. This DSPF file can be used as an input to Cadence Central Delay Calculator (CDC) tools for timing analysis.

### Checking Method

*modes*

Composite.

*checks*

HPRE.

### Arguments

*opt*

Options are the following:

- |   |                                                                                                                                               |
|---|-----------------------------------------------------------------------------------------------------------------------------------------------|
| A | Flags capacitances on all nodes, including power and ground nodes; if not specified, capacitances on power and ground nodes are not reported. |
| C | Checks cross-coupling capacitance; if not specified, total loading is assumed. Dummy C's and R's are not output into the SPF file.            |



## Dracula Reference

### Operation Block Commands

---

- G** Specifies that the original ground node names, rather than the default VSS node name, are used in the output SPICE file. This option should not be used with the *N* option.
- VSS is the default for composite and cell mode.
- In multiple-ground cases, if you want only one of them as ground node in the output SPICE file, the LPE-GROUND command now lets you specify a unique ground name. The LPE-GROUND command overwrites this SPFSELECT[G] command.
- K** Does not smash parallel transistors as one; for example, transistors of the same source, gate, drain (connected to the same nodes). Instead, these transistors are reported individually.
- N** Pure SPICE format preferred.
- P** Checks cross-coupling capacitance as a percentage of total loading. If you do not specify this option, a check is performed directly against val-a. Options *C*, *P*, and *A* are implemented to select only capacitors.
- R** Reports the area and perimeter of parasitic capacitors. The area is reported in place of the capacitance value. The perimeter is reported as a comment with the prefix *PERI=*.
- S** Schematic node names report preferred (from LOGLVIS).
- For this *S* option, the schematic ground is used in flat mode. VSS is still used as the default for composite and cell mode. You can use the *G* option or LPE-GROUND to overwrite the default ground VSS.
- u** Does not comment out parasitic CAP and RES with terminals connected to power and ground nodes or power and ground subnodes in the DSPF output. This option takes effect whether you do or do not specify back annotation. This option is used together with *Z* option. Whenever you specify the *Z* option, resistors and capacitors extracted on power and ground nets are output in commented form. You can use the *U* option to uncomment them.
- Y** Reports the transistor *x* and *y* coordinates in the LPESELECT listing file.

## Dracula Reference

### Operation Block Commands

---

**Z** Outputs resistors and capacitors extracted on power and ground nets in the DSPF file. You cannot use this DSPF as input to Cadence CDC. The Z option also outputs the instance name in front of power and group net names in the instance section of the DSPF file.

#### *element*

Type code specified in the ELEMENT device command. If no type is specified, all ELEMENT types are invoked.

|     |                   |
|-----|-------------------|
| MOS | MOS Devices       |
| DIO | Diode Devices     |
| CAP | Capacitor Devices |
| BJT | Bipolar Devices   |
| RES | Resistor Devices  |

#### *type*

Specifies a one- or two-character code to denote the type of the element.

To group or smash different subtypes of capacitors, specify type as [*new-subtype:subtype+subtype+...*]. This syntax sums the capacitance values of multiple subtypes into a new subtype. The SPICE output file then contains all parasitic metal caps, all poly capacitors, and so forth, smashed into a new group subtype. The subtypes can have different parset numbers and must be defined by PARASITIC commands. You cannot use the S option when you group different subtypes.

#### REL

Specifies one of the following:

|    |                                                                    |
|----|--------------------------------------------------------------------|
| GE | Element parameters must be greater than or equal to <i>val-a</i> . |
| GT | Element parameters must be greater than <i>val-a</i> .             |

#### RANGE

Specifies element parameters, which must be greater than or equal to *val-a* and less than or equal to *val-b*.

## Dracula Reference

### Operation Block Commands

---

*val-a/val-b*

Specifies value for *value-a* or *value-b*, which can be floating point (maximum of 7 digits) or a floating-point number followed by an integer exponent (maximum 2 digits). When floating-point numbers are specified, the leading zeros are not counted in the 7-character restriction. The preprocessor rounds the value at the 7th digit. In scientific notation, the limit is 16 characters and is truncated at the 17th.

*CORNER = nl*

Effective MOS channel width/length calculation. You can specify *CORNER* as a positive or negative value. For a pulldown device (default = 0), for each 15-degree turn in a transistor channel, a 1/6 fraction of *CORNER* specified times length (*L*) is added or subtracted from a transistor default width (*W*). For a load-device, *W* is not changed and *L* is modified.

*CORNER* uses the same algorithm as *WEFFECT* in LPECHK.

OUTPUT

Outputs SPICE listing to a file.

*c-name l-num*

*C-name* and *l-num* are concatenated to make up the file name and the subcircuit name of the SPICE listing. *C-name* can have six alphanumeric characters. If *l-num* is not specified, only *c-name* is used. For example, if *c-name* = *CKTNAM* and *l-num* = 10, the output file name is *CKTNAM10* and the subcircuit name is the same in the SPICE listing.

&

Indicates a conjunctive rule. This command can be used only with other LPESELECT commands.

For more information about ELEMENT MOS/DIO/CAP/BJT/RES, see [Chapter 7, “Extracting Electrical Parameters \(LPE\).”](#)

### Example

```
SPFSELECT[S] RES[A] &
SPFSELECT[S] CAP OUTPUT SPF1
```

## STAMP

```
STAMP stamped-layer BY stamping-layer {OUTPUT {[options]}}
 c-name l-num {d-num}}
```

### Description

The layer processing sequence and the interlayer contacts form electrical nodes. Dracula forms and uniquely labels each electrical node that results from the CONNECT-LAYER and CONNECT commands. This process is called *stamping*.

The STAMP command flags generated layers that are part of an electrical node but are not defined in the CONNECT command. Use this command after establishing the circuit connection. STAMP places the node labeling information from the stamping layer on the stamped layer. Stamping occurs only for stamped-layer polygons that overlap polygons in the stamping layer.

The STAMP command is not necessary for layers that result from the AND and NOT commands. Dracula automatically stamps layers produced with the AND and NOT commands. Dracula stamps the output layer with the node labeling of the first layer (*layer-a*).

### Arguments

*stamped-layer*

Name of a previously generated layer to stamp with the electrical node specification.

*stamping-layer*

Name of a layer that contains the electrical node labeling information. This layer can result from a previous CONNECT or STAMP.

OUTPUT

Generates an error file containing multiple-stamped geometries. The error output coordinates are also reported in the .LOG and .ERC files.

*c-name* *l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

## Dracula Reference

### Operation Block Commands

---

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

**Note:** The output cell contains clean polygons in polygon format, resulting in reduced trapezoid construction lines and a more compact cell.

*options*

Can be from among the following:

- |   |                                                                                                                                                                                             |
|---|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| T | Appends text to the output files. This option is applicable to all ERC output files.                                                                                                        |
| U | Generates one more output file with the same file name prefixed by <i>U</i> . This file contains the error geometries on the stamping layer, which caused the multiple stamping violations. |

**Note:** If regions overlap a stamped layer more than once and these regions carry different electrical nodes, the stamped layer becomes an undefined node. In other words, you can stamp only a singular node overlap.

### Example

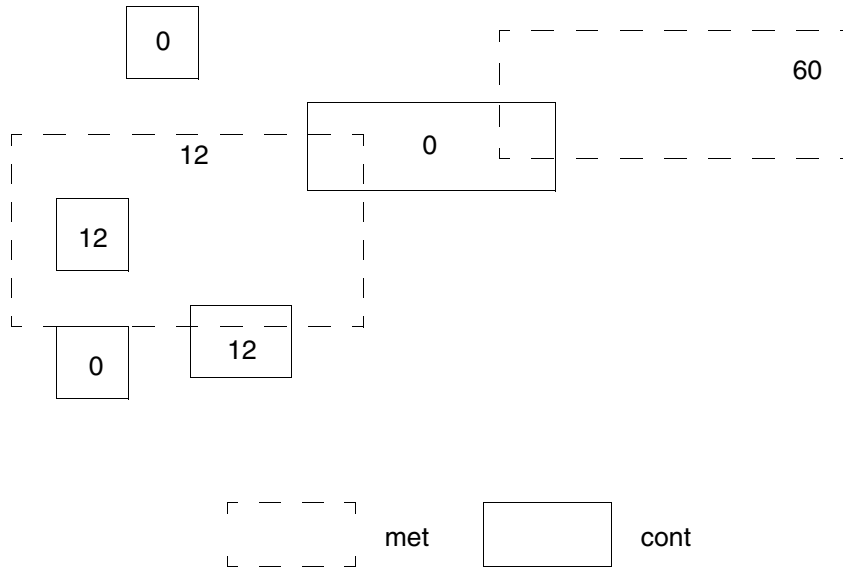
```
CONNECT met poly BY cont
STAMP cont BY met ;met has node information
```

## Dracula Reference

### Operation Block Commands

---

In the figure, numbers 12 and 60 are internal node numbers of layer *met*, and 0 represents either a no *stamped* node number or a conflicting node number.



## TRIANGLE

`TRIANGLE input-layer size output-layer`

### Description

Creates an output file that contains four triangle shapes at four corners of input trapezoids.

### Arguments

*input-layer*

BULK layer defined in the SUBSTRATE command.

*size*

Length (in microns) of the legs of the triangles. If the length or width of the trapezoid is less than the specified size, Dracula does not generate the triangles for that trapezoid.

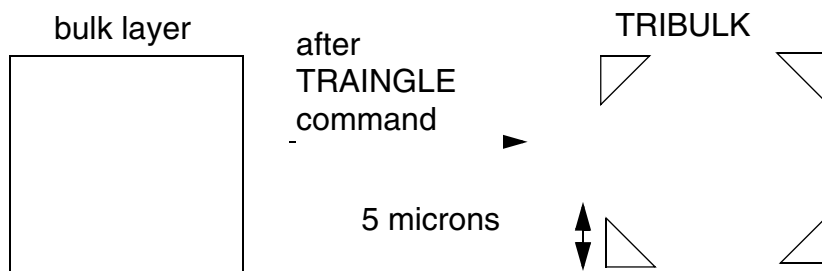
*output-layer*

Output layer for the triangles.

### Example

```
*OPERATION
.
TRIANGLE BULK 5 TRIBULK
*END
```

This example results in the following:



## WIDTH

```
WIDTH{[option1]} layer-a{&layer-a}[option2] LT/LE/RANGE/SELLT/SELLE/ SELGT/
SELGE/SELRA/ GT/GE/EQ/SELEQ/SELNE n1 {n2} ANGLE = angle-opt {trapfile}
{OUTPUT c-name l-num {d-num}} {&}
```

```
WIDTH{[option1]} layer-a RECT = n3 BY n4 {trapfile} {OUTPUT c-name l-num {d-num}}
```

### Description

Checks the dimensions between inside edges of polygons on the input layer.

**Note:** The WIDTH command will not flag violations between obtuse angled edges.

### Shell environment variables

■ `setenv SPAC_LISTERR_DIGIT [ 2 | 3 ]`

Use this environment variable to set number of digits after the decimal point for EXT/ENC/WIDTH/INT log report.

### Checking Method

modes

Flat, cell, hierarchical, composite.

checks

HDRC.

### Arguments

*option1*

Can include the following:

**Note:** The plus sign (+) in an option adds error flags and the minus sign (-) removes error flags.

**B** Flags the actual part of a violation on a segment instead of the entire segment. This option can be applied only with LT or LE modifiers. Neither RA, GT, nor GE modifiers can contain the B option.

**Note:** Causes round-off side effects on non-45-degree edges.



## Dracula Reference

### Operation Block Commands

---

|    |                                                                                                                                                   |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------|
| C  | (-) Flags the edge-pair when the edges are parallel.                                                                                              |
| C' | (-) Inverse of the C option. Flags nonparallel edges.                                                                                             |
| D  | Disables acute-angle checking.                                                                                                                    |
| F  | Outputs error flags in trapezoid format that execute information for edge-sizing SIZE[E] operations.                                              |
| F' | Outputs error flags in trapezoid format. You must specify only <i>trapfile</i> name. It is similar to R option but does not turn on the P option. |
| L  | Length check. Useful for exact contact check.                                                                                                     |



WIDTH[L] mt1 SELEQ 3 err1 ; will output A  
WIDTH[L] mt1 SELNE 3 err2 ; will output B



***This option is deprecated and should not be used. Use RECTCHK command instead.***

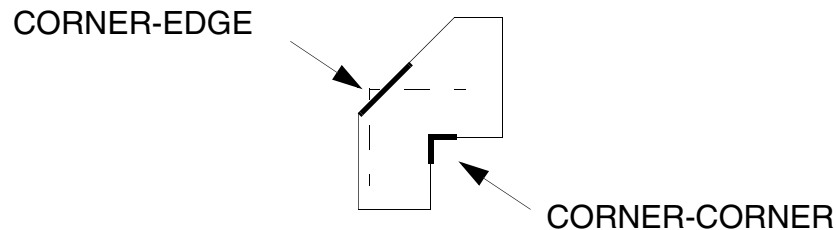
|    |                                                                                                                                                                                                                                                                                 |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| P  | (-) Flags segments of edges that project onto each other. Two edges project if perpendicular lines from a referenced edge intersect the other edge. The referenced edge is the edge most closely aligned to the x or y axis.                                                    |
| P' | (-) Inverse of the P option. Flags edges that do not project onto each other.                                                                                                                                                                                                   |
| R  | Outputs in trapezoid format the trapezoids enclosing the projection of violation flag pairs. You must specify only the <i>trapfile</i> name. This option turns on the P option.                                                                                                 |
| R' | Outputs error flags in trapezoid format so you can reuse error data. You can process the created layer (from the error flags) with logical operations. The flags are one resolution unit wide (as specified in the Description block). You cannot specify an OUTPUT error cell. |

## Dracula Reference

### Operation Block Commands

---

- S (+) Flags violations on the polygon if the edges of the polygon are within a “square boundary” inside of the other polygon edge. When you use this option, you can also specify the *CORNER-CORNER* and *CORNER-EDGE* options. The default values for *n2* and *n3* are *n1*. If you do not specify the *CORNER-CORNER* option, *n2* is set equal to *n1*. If you do not specify *CORNER-EDGE* option, *n3* is set equal to *n1*. The values you specify for *n2* and *n3* must be less than or equal to the default normal checking value, *n1*.



Use the following syntax for these options:

```
WIDTH[S] layer1 LT n1 {CORNER-CORNER
 n2} {CORNER-EDGE n3}
```

*CORNER-CORNER* works only when the two corners are opposite each other. This way, rotation has no effect on the area being checked.

*CORNER-EDGE* is bypassed if any error between two edge pairs has been flagged by a normal check.

The *CORNER-CORNER* checking value, *n2*, and the *CORNER-EDGE* value, *n3*, cannot be greater than the normal checking value, *n1*, because *n2* and *n3* act like filters and reduce the number of errors that were initially caught by *n1*. *n2* must be greater than or equal to  $n1/\sqrt{2}$ .

Only 90 degree angles are treated as "CORNER"

- X Checks the delta value in the x direction between two edges. The edges must project onto each other. Does not check non-Manhattan data.

## Dracula Reference

### Operation Block Commands

---

**y** Checks the delta value in the y direction between two edges. The edges must project onto each other. Does not check non-Manhattan data.

*layer-a*

First input layer name.

*&layer-a1*

Renames (conjunctive-rename) the layer containing the error flags. The original layer name is then restored to the original layer. No spaces are allowed between the *layer-a* and *&layer-a1* names, for example, *poly&rpoly*.

*option2*

Measures between all nonconsecutive pairs of inside edges of the polygons on the input layer.

Compares the measurements against the limits you specify and flags violations.



***You must supply only one of these measurements:***

LT/LE *n1*

Flags an error if the two segments are spaced less than *n1* for LT or less than or equal to *n1* for LE. Does not flag an error if the segments touch.

RANGE *n1 n2*

Flags an error if the two segments are spaced less than *n2* and greater than *n1*. If you do not want acute-angle error flags, you must specify the RANGE measurement.

SELLT/SELLE *n1*

SELLT and SELLE output in trapezoid format those *layer-a* polygons that the program flags if you specify an LT *n1* or an LE *n1* and use the same options. Specify a *trapfile* name instead of using the OUTPUT statement.

SELGT/SELGE *n1*

SELGT outputs in trapezoid format the *layer-a* polygons that the program does not flag if you specify an LE measurement with the same options. SELGE has the same function as SELGT except that it works for an LT operation. You cannot use the OUTPUT statement with this option. Specify a *trapfile* name instead of using the OUTPUT statement.

## Dracula Reference

### Operation Block Commands

---

SELRA *n1 n2*

SELRA outputs in trapezoid format those *layer-a* polygons that the program would flag if you specify a RANGE *n1 n2* measurement and use the same options. Specify a *trapfile* name instead of using the OUTPUT statement.

GT/GE *n1*

Flags an error if the two segments are spaced greater than *n1* for GT or greater than or equal to *n1* for GE. Does not flag an error if the segments touch.

**Note:** "WIDTH GE/GT <value> ..." is interpreted by DRACULA as "WIDTH RANGE <value> 999 ..."

EQ *n1*

Flags an error if the two segments are spaced equal to *n1* for EQ. Does not flag an error if the segments touch.

SELEQ/SELNE *n1*

SELEQ output in trapezoid format those *layer-a* polygons that the program flags if you specify an EQ *n1* and use the same options while SELNE outputs in trapezoid format those *layer-a* polygons that fail the EQ condition. Specify a *trapfile* name instead of using the OUTPUT statement.

*angle-opt*

Specifies the 45, non-45, 90, or non-90 degree angle to be checked. When you use this option, the C option for the ANGLE option is automatically invoked, and you cannot disable it.

*trapfile*

Trapezoid file name. Use this option only with the *R* or *R'* option or with the SELLT, SELLE, SELGT, SELGE, SELRA operations.

OUTPUT

Sends the results of the operation to an output cell.

RECT

Checks rectangle shapes that are exactly this size: *n3xn4*.

**Note:** RECT check will be turned off immediately if it is used with LT/LE/GT/GE/EQ/RANGE/SELxx in WIDTH checks.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD

## Dracula Reference

### Operation Block Commands

---

system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

&

Indicates a conjunctive rule.

**Note:** The output cell contains clean polygons in polygon format, resulting in reduced trapezoid construction lines and a more compact cell. The *trapfile* option and the OUTPUT command are mutually exclusive.

### Layer Options

O

Checks the original layer, not the layer generated by the previous operation in the conjunctive rule. Use this option only with a conjunctive rule and on a layer with error flags.

### Example 1

```
ENC layer1 layer2 LT 1 outl1 &
WIDTH layer1[0] LT 2 OUTPUT err1 23
```

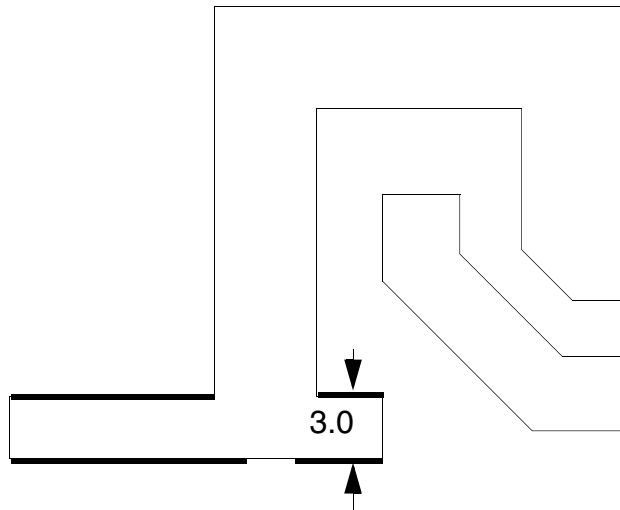
## Dracula Reference

### Operation Block Commands

---

#### Example 2

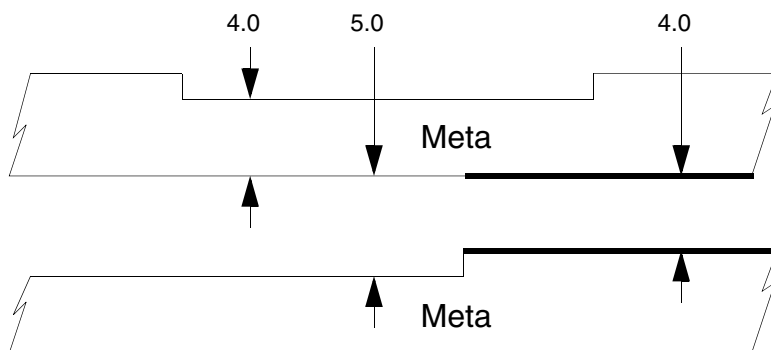
```
WIDTH metal LT 5 OUTPUT drc07 92
```



#### Example 3

This example checks for external metal spacing of 5.0 microns of metal lines that are 4.0 microns wide. These narrow metal lines must be spaced at 5.0 microns and not at 4.0 microns for wider busses.

```
WIDTH[PC] metal LE 4 &
EXT[PC] metal metal[O] LT 5 OUTPUT drc07 93
```



#### Example 4

Example 3 rewritten:

## Dracula Reference

### Operation Block Commands

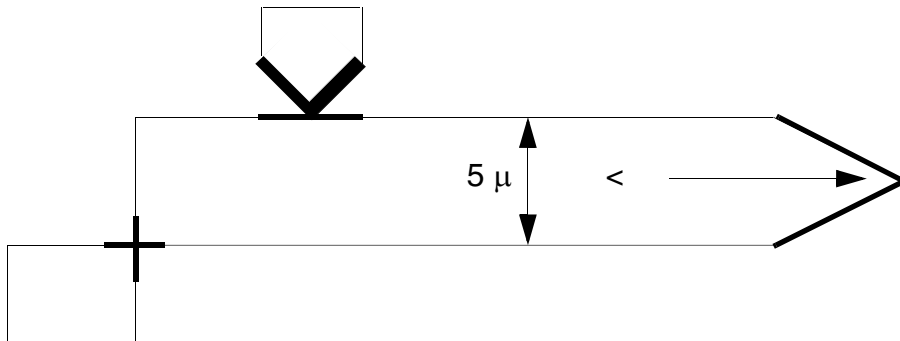
---

```
WIDTH[PC] metal&meta E 4 &
EXT[PC] &meta metal LT 5 OUTPUT drc07 94
```

Also checks for external metal spacing as described in Example 2. Use the conjunctive-rename operation. This allows the *&meta* conjuncted layer to be used later in another DRC conjunctive check. Layer *metal* reverts to *metal[O]* at this point. Conjunctive-rename is optional and can save CPU resources.

#### Example 5

```
WIDTH met LT 3 output drc07 98
```



#### Example 6

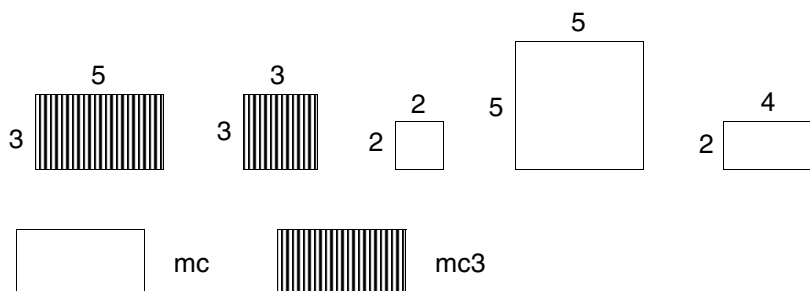
This example checks for the width of circle metal contact (*cmc*) and produces no false errors.

```
WIDTH[C] cmc LT 5 drc07 90
```

#### Example 7

This example checks for metal contacts (*mc*) that are not 3 by *n* microns.

```
WIDTH[R] mc RANGE 2.99 3.01 mc3
NOT mc mc3 badmc OUTPUT drc07 91
```



### Example 8

The first three command lines check for transistor channel length of less than 3 microns. The fourth command checks for gate external spacing to contact (*mc*) in the direction of diffusion only. Touching is also an error.

This example renames the *gate* layer to *&gwidth* in the conjunctive-rename operation. This optional conjunct-rename operation saves a conjunctive command for the DRC check *drc0796*, thereby saving CPU resources.

```
AND poly diff gate
ENC[T] gate&gwidth poly LT .001 &
WIDTH[P] &gwidth LT 3 OUTPUT drc07 92
```

also

```
EXT[T] &gwidth mc LT 4 OUTPUT drc07 96
```

### Example 9

This example outputs metal polygons that have widths greater than 10 microns only. Does not output widths less than or equal to 10 microns. The output layer is *wmetal* and is in trapezoid format.

```
WIDTH metal SELGT 10 wmetal
```

### Example 10

This example outputs metal polygons that are 10 microns in width at some point and are output to layer *metal10* in trapezoid format.

```
WIDTH metal SELRA 9.9 10.1 metal10
```

### Example 11

This example outputs the contact in the 3 X 3 rectangle shape.

```
WIDTH[L] CONTACT SELEQ 3 OUTPUT ERR 01
```



## XBOX

XBOX {[T]} {[F]} *layer-a* OUTPUT *c-name* *l-num* {*d-num*}

### Description

Checks for composite geometries in the specified layer that overlap or touch *cell boxes* (Hcell boundaries). Use the XBOX command to gather information about an HLVS operation.

### Checking Method

modes

Composite.

checks

HLVS.

### Arguments

[T]

Indicates that touching is a violation.

[F]

Uses inner boundaries for framed Hcells, instead of Hcell boundaries, to check overlapping or touching.

*layer-a*

Layer to be checked.

OUTPUT

Sends the results to an output cell.

*c-name* *l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

## Dracula Reference

### Operation Block Commands

---

#### Example

```
XBOX poly OUTPUT xpoly 55
XBOX[T] metal OUTPUT xmet 10
XBOX[TF] poly OUTPUT xpoly 51
```

## XCELL

`XCELL layer-a layer-b OUTPUT c-name l-num {d-num}`

### Description

Checks for geometries from the composite plane of *layer-a* or *layer-b* that overlap geometries in the Hcells from *layer-a* or *layer-b*.

Use this command prior to an HLVS run to check for any device formation caused by overlapping areas of Hcells or by composite geometries that overlap Hcells. These devices are also checked by *XDEVICE*. Devices can be formed only by geometries that are solely from the composite or solely from within an Hcell. (The BASE-LAYER command is an exception to this rule.)

### Checking Method

modes

Composite.

checks

HLVS.

### Arguments

*layer-a*

First layer to be checked.

*layer-b*

Second layer to be checked.

OUTPUT

Outputs the results of the operation in the form of an output cell, where:

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

## Dracula Reference

### Operation Block Commands

---

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

#### Example

```
XCELL poly diff OUTPUT badgat 10
```

## XDEVICE

```
XDEVICE element[subtype] OUTPUT C-name l-num {d-num}
```

### Description

Outputs devices formed by composite-to-cell or cell-to-cell interactions in a composite mode HLVS/HLPE/HERC run. These devices are extra devices that are not present in the schematics. The LVS report lists these devices, but they are not used in LVS matching nor reported in the HLPE output file.

Specify only one XDEVICE command for each ELEMENT declaration. Dracula creates *.ENC* files for every layer. These *.ENC* files store information about the area between the cell plane and composite plane.

### Arguments

*element*[*subtype*]

The element and its subtype.

OUTPUT

Sends the results to an output cell.

*c-name* *l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

### Example

This example outputs the composite-to-cell and cell-to-cell devices for MOS[P] to error cell ILLP10 and those for MOS[N] to ILLN11.

```
*OPERATION
ELEMENT MOS[P] PGATE POLY PSRCDRN NSUB
ELEMENT MOS[N] NGATE POLY NSRCDRN PWELL
XDEVICE MOS[P] OUTPUT ILLP 10
XDEVICE MOS[N] OUTPUT ILLN 11
```

## Dracula Reference

### Operation Block Commands

---

```
LVSCHK[C]
*END
```

Dracula checks, and flags with x and y coordinates, devices formed by cell-to-cell or cell-to-composite interactions in the LVS report file. By the definition of an Hcell, these devices do not exist in the schematics. They are flagged as extra layout devices. The report format is as follows:

```
***** XDEVICE SUMMARY REPORT *****
EXTRA COMPOSITE/CELL OR CELL/CELL DEVICES IN LAYER ILLN
 ILLN X= 5521.50, Y= 4396.50
 ILLN X= 5540.50, Y= 4396.50
```

Dracula lists only the first 100 occurrences in the LVS report. They are listed whether or not you specified an XDEVICE. If you want an output layer for these devices, use the XDEVICE command. The output layer is a flat layer and all devices are output.

## XOR

```
XOR {[D{n}]} layer-a layer-b trapfile {OUTPUT c-name l-num {d-num}}
```

### Description

Creates a new layer from two other layers. The new layer consists of the total region of both layers minus the region shared by both layers.

Database partition functions were added for the SELECT, SIZE, and LOGICAL commands. These enhancements resolve capacity limitation issues with Dracula. This allows you to partition input layers of these functions into blocks. When you run Distributed Dracula, each block (pair) can be assigned to one processor. Specify a value between 2-10 inclusive following 'D' to indicate the number of partitions desired. The default value of 'D' is 4 if the value is omitted.

**Note:** There is a performance gain in the database partition feature for SELECT in the 4.8 version. The same is not true for SIZE and LOGICAL on account of too much overhead from the pre-process (PRESEL for SIZE) and post-process (LOGMRG for LOGICAL) stages during the operations. Database partition scheme is designed specially for resolving Dracula capacity issues and is applied to flat mode only.

### Arguments

D

Turn on the database partition feature

n

Number of blocks to be partitioned into

*layer-a*

First input layer name

*layer-b*

Second input layer name

*trapfile*

Trapezoid file created by the logical operation. The *trapfile* name can have up to seven characters and no special characters. A-Z and 0-9 are permitted, but the first character must be a letter.

## Dracula Reference

### Operation Block Commands

---

#### OUTPUT

Sends the results of the operation to an output cell.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only. Values can range from 0 to 63.

#### Example

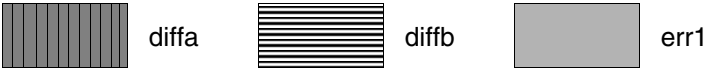
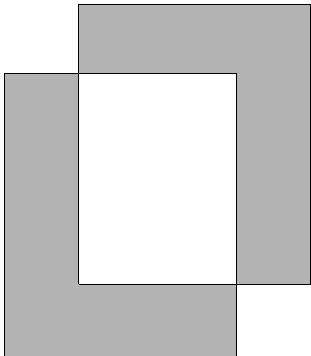
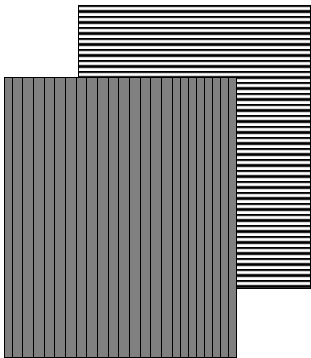
The following example shows a database comparison between layers that are supposed to be identically digitized.



**Dracula Reference**  
Operation Block Commands

---

XOR diffa diffb err1



## **XVIA**

*XVIA layer-a OUTPUT C-name l-num {d-num}*

### **Description**

Checks for a contact layer that is not covered by a related conductor layer in either the composite plane or in the Hcells. You must specify the contact layer and related connect layers in a prior CONNECT command.

### **Checking Method**

modes

Composite.

checks

HLVS.

### **Arguments**

*layer-a*

Layer containing the contacts to be checked.

OUTPUT

Sends the results to an output cell.

*c-name l-num*

*C-name* is the name of the output cell and can have six alphanumeric characters or fewer and no special characters. *L-num* is the cell layer number determined by your CAD system. If your *c-name* is *rule03* and *l-num* is 5, the output cell name is *rule0305*.

*d-num*

Datatype number associated with the layer number (*l-num*) of the output cell. Use *d-num* for GDSII only; values can range from 0 to 63.

Use this command to detect floating contacts that are not found otherwise and that are not allowed in HLVS runs.

## Dracula Reference

### Operation Block Commands

---

#### Example

```
XVIA contact OUTPUT uncont 10
XVIA via OUTPUT unvia 44
```

## **Dracula Reference**

### Operation Block Commands

---

---

## Supported Database Records

---

This appendix discusses Dracula Supported Database Records.

- [“Introduction”](#) on page 1237
- [“GDSII Records”](#) on page 1237

### Introduction

Dracula supports the CAD system formats listed below. The SYSTEM or SYSIN command in the Description block specifies your database format.

- GDSII
- Design Framework II database (Cadence)

### GDSII Records

GDS2IN processes the GDSII database records listed in the following table. It ignores all other records. Dracula Release 4.4 supports GDSII Release 6.0 or lower.

| Record number | Name    |
|---------------|---------|
| 0             | HEADER  |
| 1             | BGNLIB  |
| 2             | LIBNAME |
| 3             | UNITS   |
| 4             | ENDLIB  |
| 5             | BGNSTR  |
| 6             | STRNAME |

**Dracula Reference**  
Supported Database Records

---

| Record number | Name         |
|---------------|--------------|
| 7             | ENDSTR       |
| 8             | BOUNDARY     |
| 9             | PATH         |
| 10            | SREF         |
| 11            | AREF         |
| 12            | TEXT         |
| 13            | LAYER        |
| 14            | DATATYPE     |
| 15            | WIDTH        |
| 16            | XY           |
| 17            | ENDEL        |
| 18            | SNAME        |
| 19            | COLROW       |
| 22            | TEXTTYPE     |
| 23            | PRESENTATION |
| 25            | STRING       |
| 26            | STRANS       |
| 27            | MAG          |
| 28            | ANGLE        |
| 33            | PATHTYPE     |
| 45            | BOX          |
| 48            | BGNEXTN      |
| 49            | ENDEXTN      |

GDS2IN prints an error message for missing database cells, which might occur when inputting graphical data.

---

## Data Structure Translator

---

Appendix B discusses the Data Structure Translator.

- “Running the Translator” on page 1239
- “Examples” on page 1240

### Running the Translator

You can use the Translator utility to convert Dracula III, 4.4 (called *new*) data files into Dracula II (*old*) data files and convert Dracula II to Dracula III.

To run the translator on a UNIX workstation, type the following:

```
% DBCONVER
```

The Translator displays a header similar to the following:

```

*/N DBCONVER (REV. 4.4 / SUN-4 /GENDATE: 28-Jun-95/10)
 *** (Copyright 1995, Cadence) ***
*/N EXEC TIME =20:16:01 DATE = 5-Aug-95

>>If you have any questions, please type 'help' at any time.(x/exit)<<
```

Enter IN & OUT FILE (case sensitive):

Enter file names with at least one space between IN and OUT file names.

```
: FILE-IN.DAT FILE-OUT.DAT
```

The program converts the IN file name to the OUT file name. When the conversion is finished, the program displays the file names and the direction of the conversion (new to old, or old to new).

```
(new->old) : FILE-IN.DAT -> FILE-OUT.DAT
```

To terminate the program, enter

```
% [RETURN]
```

The following restrictions apply to this program:

## Dracula Reference

### Data Structure Translator

---

- The IN file name and OUT file name are both case sensitive. The program also preserves the case you use when entering the names.
- The IN file name must be either a file name or a full path name. The OUT filename can be a directory name, a full path name, or a blank.
- If the IN file name and OUT file name are identical and in the same directory, the program overwrites the IN file name.
- You can create a file containing a list of the IN and OUT file names called *DIRFILE.DAT* and use this file as the IN file name. The format for this file is the same as that given above.
- The program also converts ASCII files to the new file name and/or overwrites the old file.

If you are using PDRACULA II as well as PDRACULA III, you need to document the variable RUNDRAC when switching between Dracula II and Dracula III.

- If you are using PDRACULA II, set RUNDRAC to 2. This tells the software to generate an uncompressed format database. With RUNDRAC set to 2, LOGLVS generates LOGLVS.DAT for the SCHMIN module in an uncompressed format.
- If you are using PDRACULA III, set RUNDRAC to 3 so the software generates a compressed format database. With RUNDRAC set to 3, LOGLVS generates LVSLOGIC.DAT for SCHMIN module in a compressed format.

## Examples

|                    |        |                    |        |           |
|--------------------|--------|--------------------|--------|-----------|
| FN.DAT             | -->    | FN.DAT             | FN.DAT |           |
| fn.dat             | fm.dat | -->                | fn.dat | fm.dat    |
| fn.dat             | ../    | -->                | fn.dat | ../fn.dat |
| ../FN.DAT../fm.dat | -->    | ../FN.DAT../fm.dat |        |           |

The program can do conversions in only one direction during a run. If you are converting old data files (Dracula II) to new (Dracula III), you must exit and invoke DBCONVER again if you want to convert new to old.



---

# Running Dracula

---

Appendix C focuses on Running Dracula.

- [“Running a Dracula DRC/ERC Job”](#) on page 1241
- [“System Requirements”](#) on page 1244
- [“UNIX Case-Sensitivity Enhancements”](#) on page 1245
- [“Batch Job Queuing Utility”](#) on page 1249

## Running a Dracula DRC/ERC Job

To run a DRC or an ERC job, complete the following steps.

1. Change to the directory where the job is to run.
2. Run the preprocessor to generate the *jxrun.com* file.

PDRACULA or PDRACULA2

3. Type

```
: /get filename.com
: /fin
```

**Note:** The *jxrun.com* file is the UNIX version submitted to run the Dracula job. PDRACULA is the preprocessor for DRACULA III, and PDRACULA2 is for DRACULA II.

## Submitting the *jxrun.com* File

- To run a job in the foreground, enter the following command:

```
jxrun.com >& drc2.log
```

The log file is *drc2.log*.

- To run a job in the background, enter one of the following commands:

```
jxrun.com >& drc2.log &
[.] 8432(Job number)
```

## Dracula Reference

### Running Dracula

---

The log file name is *drc2.log*.

or

```
jxsub.com
[.] 8433(Job number)
```

The log file name is *printfile.log*.

- To run a job in the background from a break point, enter the following command:

```
jxrun.com BEGIN EXPAND >& drc2.log &
[.] 8433
```

This entry runs the job in the background from the *BEGIN* to *EXPAND* break point names. The log file name is *drc2.log*.

## Running Other Dracula Programs

You can execute the other interactive Dracula programs by entering the following commands:

`LOGLVS`(LVS Network Compiler)

`LPEPRO`(LPE Postprocessor)

These names are aliases of their all-cap equivalents.

## UNIX Commands

The following UNIX commands might help you use Dracula.

---

|                                       |                                                                                                       |
|---------------------------------------|-------------------------------------------------------------------------------------------------------|
| <code>ps</code>                       | Gives process status                                                                                  |
| <code>kill &lt;number&gt;</code>      | Kills a background job                                                                                |
| <code>df</code>                       | Checks disk space                                                                                     |
| <code>Control-c</code>                | Stops a foreground job                                                                                |
| <code>tail &lt;filename&gt;</code>    | Lists last 10 lines of file name                                                                      |
| <code>tail -f &lt;filename&gt;</code> | Lets you output the file as it is written. Lets you monitor a big job while still storing a log file. |
| <code>cat &lt;filename&gt;</code>     | Displays a file                                                                                       |
| <code>ls</code>                       | Lists a directory                                                                                     |
| <code>ls -l</code>                    | Lists a directory in long format                                                                      |
| <code>cp</code>                       | Copies a file                                                                                         |
| <code>ftp</code>                      | Transfers files to and from a remote network site                                                     |
| <code>mkdir</code>                    | Creates a directory                                                                                   |
| <code>rm</code>                       | Deletes a file                                                                                        |
| <code>cd</code>                       | Changes directory                                                                                     |
| <code>lpr</code>                      | Prints a file                                                                                         |

---

## Dracula Reference

### Running Dracula

---

#### Sun:

---

|                       |                   |
|-----------------------|-------------------|
| <code>pstat -s</code> | Checks swap space |
|-----------------------|-------------------|

---

#### Sun or HP:

---

|                                         |                                                                                                  |
|-----------------------------------------|--------------------------------------------------------------------------------------------------|
| <code>swapon -a &lt;filename&gt;</code> | Adds swap space. <i>filename</i> is a file created with the UNIX <i>mkfile</i> command.          |
| <code>swapon {special}</code>           | Adds swap space. <i>special</i> is the block device made available to the system for allocation. |
| <code>reboot</code>                     | Reboots the system                                                                               |

---

You must have root privileges to execute the above three commands.

## System Requirements

To run Dracula, the recommended system has 32 Mbytes of physical memory and a disk drive with a minimum of 300 Mbytes.

These estimates of needed memory are conservative. Because Release 4.4 uses dynamic memory allocation, it only takes 5 to 10 Mbytes to verify small designs (for example, 2000 transistors).

The recommended “swap space” is 20X memory size for a single Dracula job. If the system is used by more than one user or used to run more than one Dracula job, a larger swap space is recommended.

If you are running large Dracula jobs and encounter the system error message “NOT ENOUGH MEMORY,” increase the swap space.

Because of dynamic memory allocation, Dracula allocates memory as needed. If there is no more memory left in the system to allocate, Dracula aborts with following message:

```
* /F* ERROR: function flloc can't allocate memory
** ABORT AT ROUTINE : falloc STATEMENT # = 1
```

Increase the swap space to make up for the lack of memory.

After you change the system’s swap space, back up and reformat the disks, then reload them with the backed-up data or use swap. The disk swap space must be contiguous. Refer to your system’s installation guide for further information.

**HP:** The system's default swap space for each process is 25 Mbytes. You might need to increase this space for large jobs. Check with your system administrator before making a change because this change involves rebuilding the kernel.

## UNIX Case-Sensitivity Enhancements

All commands and responses to questions are case insensitive. File names, however, are case sensitive. Some commands, as described below, also search for different file name extensions.

The first file found that fits the file name is used. If no file fits any of the possible file names searched for, an error message occurs.

### PDRACULA

All command keywords (such as /GET and /FIN) are case insensitive.

**/g filename** searches for

```
filename (where filename is taken literally, case sensitive)
filename.rul
filename.RUL
filename.com
filename.COM
```

in that order.

**/g /dir-path/filename** searches for

```
/dir-path/filename (where filename and
 dir-path are taken literally)
```

and again for *.rul*, *.RUL*, *.com*, and *.COM* in that order.

The file name can contain periods (for example, *drc.1a*) and can be any combination of upper and lower case. The *dir-path* is also taken literally with any legal UNIX character accepted. Total */dir-path/filename* length cannot exceed 128 characters, including any extension.

**/g ../filename** searches as above one level up the directory.

**/ed /(optional)dir-path/filename** edits exactly

```
/dir-path/filename
```

**/f** creates lowercase *jx* files

## Dracula Reference

### Running Dracula

---

```
jxrun.com
jxsub.com
```

and so forth. Non-*jx* files remain uppercase.

#### **/f filename** creates

```
filename.com (where filename is taken literally)
jxsub.com
```

and so forth.

In the input rules file (commonly referred to as the *RULE*, or *.rul* or *.com* or *rules file*), command keywords are case insensitive (for example, *INDISK=indisk*, *AND=and*, *SELECT=select*).

Where inputs to commands include the dollar sign (\$), the following rule applies: input that is a UNIX path assumes that “\$” refers to a variable set externally by a set command; input that is not a UNIX path accepts “\$” as a valid character, not as an externally set variable.

Examples:

- PROGRAM-DIR=\$abc/ where abc = /usr/drac, is the same as PROGRAM-DIR=/usr/drac/
- SUB-PROG=SIZE \$abc/ is similar to the previous example.
- PRIMARY=Name\$Xyz is interpreted as NAME\$XYZ where \$XYZ is not an external variable.

Note the uppercase.

- If CNAMES-CSEN=YES, the above example is instead *Name\$Xyz* where *\$Xyz* is not an external variable.

Note the mixed case.

**PRINTFILE=filename**

All files affected by the command have lowercase extensions.

```
filename.log (where filename is taken literally, case sensitive)
filename.sum
filename.lvs
filename.mlg
filename.psm
filename.msm
filename.err
```

**INDISK=filename** searches for

## Dracula Reference

### Running Dracula

---

filename            (where *filename* is taken literally)  
filename.db  
filename.DB  
filename.DAT

in that order.

**OUTDISK=filename** creates the literal file name without extension.

**PROGRAM-DIR=/dir-path/** is interpreted literally. Maximum of 128 characters. The trailing / is required.

**SCHEMATIC=filename** searches for

filename            (where *filename* is taken literally)  
filename.DAT

in that order.

You can specify an optional */dir-path/* (for example, *SCHEMATIC=/dir-path/filename*).

**WORK-DIR=/dir-path/** is interpreted literally. Maximum of 128 characters.

## LOGLVS

LOGLVS uses case-sensitive input file names and accepts full directory paths. All command keywords are case insensitive.

**cir filename** searches for

```
filename (literal)
filename.cir
filename.CIR
filename.DAT
```

**lib filename** searches for

```
filename (literal)
filename.teg
filename.TEG
filename.DAT
```

**inp filename** searches for

```
filename (literal)
filename.sil
filename.SIL
filename.ilo
filename.ILO
filename.DAT
```

**cel filename** searches for

```
filename (literal)
filename.tab
filename.TAB
```

**cir \$abc/filename** searches for *filename* with *dir-path* defined by *\$abc*

**cir \$file** searches for *filename* defined by variable *\$file*

**cir /dir-path/filename** (*dir-path* and *filename* are taken literally)

**con SUBCKT-name** searches for

```
.SUBCKT SUBCKT-name (literal)
.SUBCKT SUBCKT-NAME (all uppercase)
.SUBCKT subckt-name (all lowercase)
```

The *LVSLOGIC.DAT* file name is created as the default.



## Batch Job Queuing Utility



***Effective from June 2009, the `dracq`/`dracbatch` utilities are no longer supported. Although, these utilities will continue to be shipped in future Dracula releases. You use them at your own risk. Dracula has been using the FlexLM licencing scheme for quite some time, which already has built-in, automatic queuing capabilities.***

### Commands

`dracq`  
`dracbatch`

### Syntax

```
dracq [-s] job [job ...]
dracq -l
dracq (-l|-r) job# [job# ...]
dracq (-l|-r) <login>
dracbatch [maxJobNumber] &
```

### Options

- s**  
Sets up job requests into the FIFO job queue
- l**  
Lists job requests in the FIFO job queue
- r**  
Removes job requests from the FIFO job queue

### Description

The UNIX batch queuing system consists of two programs, *dracq* and *dracbatch*, which access the same job queue. The *dracq* program (like sender) places job requests containing command lines onto the FIFO queue. The *dracbatch* program (like server) removes a job request from the FIFO queue and executes the commands specified in it.

## Dracula Reference

### Running Dracula

---

Once this job request is done, *dracbatch* removes the next job request in the queue and executes it until there are no more job requests waiting in the queue. Jobs are stored in first-in, first-out (FIFO) order, so jobs are executed in the order they are queued.

To specify a maximum number of jobs in the, specify a maximum job number for the *dracbatch* command. For example

```
$ dracbatch [maxJobNumber] &
```

If you specify a maximum job number of five, and there are three jobs in the queue, the fourth and fifth jobs join the queue. If there is a sixth job, it cannot join the queue and you must resubmit the job later when the active job finishes or a job is dequeued.

The *dracq* program has three functions:

- Puts job requests into the queue
- Lists job requests currently in the queue
- Deletes job requests from the queue

### Usage

```
dracq [-slr] [job ...]
```

### Examples

```
dracq [-s] [path1/]job1 [[path2/]job2 ...]
```

Sets up job1, job2, and so on, into the FIFO job queue. When it is time to execute the jobs, the *dracbatch* daemon removes job1 from the queue, changes to the path1 directory, executes job1, removes job2 from the queue, changes to the path2 directory, executes job2, and so on.

**Note:** The queuing system only takes a job request in the script format of *jxsub.com*, either in its name *jxsub.com* or any other user-defined name.

Following is an example of a job request, *jxjob1*. You can generate the job request file, *jxjob1*, by running PDRACULA to create *jxsub.com*, then renaming it to *jxjob1*. Submit the *jxjob1* to the batch queue by issuing the command *dracq -s jxjob1*. The format of *jxjob1* is shown below:

```
#!/bin/csh
setenv jclf "jxrun.com"
setenv logf "/usr/Dracula/bsdqa/drc/drc4/DRC4.log"
setenv msmf "/usr/Dracula/bsdqa/drc/drc4/DRC4.msm"
csh jxrun.int $jclf $logf $msmf $1 $2 >& \
 /usr/Dracula/bsdqa/drc/drc4/DRC4.mlg&
```

## Dracula Reference

### Running Dracula

---

The following example lists all jobs in the job queue. When there is no entry, a message “Currently no job in the queue” is displayed.

```
dracq -l
```

The following example lists all jobs queued by the user *login*. When there are no such jobs, a message “No job matching request” is displayed.

```
dracq -l <login>
```

The following example lists the requested numbered jobs only.

```
dracq -l <job#> [job# ...]
```

The following example removes all jobs queued by the user *login*. When there are no such jobs, a message “No job matching request” is displayed

```
dracq -r <login>
```

The following example removes the requested numbered jobs only

```
dracq -r <job#> [job# ...]
```

The following example specifies a maximum of three jobs are allowed in the queue. If you submit a forth job, you receive an error message telling you to resubmit the job later.

```
$ dracbatch 3 &
```

## Installation

Before running these programs, verify that *dracq* and *dracbatch* are owned by root and have the proper protections:

```
cd /(path...) #where dracq, dracbatch is accessed by all
chown root dracq dracbatch
chmod 6111 dracq
chmod 100 dracbatch
```

The location of these programs can be in the Dracula hierarchy *if* that hierarchy is on the same machine, or if it is copied elsewhere, for example, */usr/local*.

Output files generated by *dracq -s* are stored in the */usr/spool/batch* spooling directory.

To create this spooling directory, enter the following commands:

```
su root
cd /usr/spool
mkdir batch
chown root batch
chmod 777 batch
chmod g+s batch
vi /etc/rc.local
```

## Dracula Reference

### Running Dracula

---

Add a conditional entry to execute *dracbatch* upon reboot to make sure the queue is always running.

The *dracbatch* program runs when the system is brought to multi-user operation (generally run level 2 for UNIX System V™). This can be done automatically by invoking *dracbatch* in one of the script files under */etc/rc2.d* (UNIX System V.3 and later) or by an entry in the */etc/rc* file (System V.2 or BSD).

The following is a sample shell script for invoking *dracbatch*.

```
#
Invoke the job queue execution daemon
#
Only run this script if the previous run level was single-user mode(to avoid
multiple invocations):
#
if [-x /usr/local/bin/dracbatch]
then
 /usr/local/bin/dracbatch & (echo -n ' dracbatch ') >
 /dev/console
fi
```

Because *dracq* is for anyone who needs to queue a job, it must be executable by every user. It must be owned by *root* and stored in a local *bin* directory that all users have in their path (such as */usr/sbin* or */usr/local/bin*). It must also have 6111 permission (--s--s--x). Make sure that everyone can *only* access the resident *dracq* command on the machine running the batch queue. Because ownership and permissions are critical, do not put *draq* in a directory that might accidentally be accessed by others.

The *dracbatch* daemon is run by *init* at startup and must be owned by *root* with 100 permission mode (--x-----).

The *dracbatch* daemon spawns a child shell to process the jobs in the batch queue. If there is no batch job in the queue, the *dracbatch* daemon spawns a child process and checks the batch queue every 30 seconds.

### Scenario 1

Suppose there are three users (*A*, *B*, and *C*) logged in to the same machine (*M*). At one time, user *A* issues the *dracq -s jxsubA* command before user *B* issues the *dracq -s jxsubB* command; user *C* does not use the batch utility, but instead issues the *jxsub.com* command.

Therefore, on machine *M*, there are two Dracula jobs running (*jxsubA* of user *A* and *jxsub.com* of user *C*), and there is one Dracula job waiting in the queue for execution (*jxsubB* of user *B*). Job *jxsubB* executes when *jxsubA* is finished.

The batch queue utilities do not support break points, so you can only run the job request from stage 1 to the last stage.

## Dracula Reference

### Running Dracula

---

When submitting multiple Dracula jobs at one time, *dracq* treats all the jobs as one batch job. For example, to submit multiple Dracula jobs:

```
% dracq -s drc/jxsub.com erc/ercjxsub.com lvs/lvljxsub.com
```

To list batch jobs:

```
% dracq -l
```

| Rank   | Owner  | Job | Files                                   |
|--------|--------|-----|-----------------------------------------|
| active | mchang | 13  | jxsub.com,ercjxsub.com,<br>lvljxsub.com |

To remove batch jobs:

```
% dracq -r 13
```

*Dracq* removes the 13th batch job; that is, the three Dracula jobs *jxsub.com*, *ercjxsub.com*, and *lvljxsub.com*.

## Scenario 2

If you want to submit multiple Dracula jobs with multiple batch job listings, you can create a UNIX shell script file. Define a name for the file (for example, *dracqmulti*) and put the file in your local search \$PATH. For example:

```
% dracqmulti drc/jxsub.com erc/ercjxsub.com lvs/lvljxsub.com
```

where *dracqmulti* is

```
#!/bin/csh
foreach i ($*)
 dracq -s $i
end
```

To list batch jobs:

```
% dracq -l
```

| Rank   | Owner  | Job | Files        |
|--------|--------|-----|--------------|
| active | mchang | 13  | jxsub.com    |
| 1      | mchang | 14  | ercjxsub.com |
| 2      | mchang | 15  | lvljxsub.com |

To remove batch jobs:

```
% dracq -r 13
```

*Dracq* removes only the 13th batch job (that is, the Dracula job, *jxsub.com*).

## **Dracula Reference**

### Running Dracula

---

---

# Glossary

---

The below glossary defines the terms and concepts that you should understand to use Dracula effectively.

### Backannotation

The process of replacing the net and device name information in the layout netlist with the corresponding net and device name information from the schematic netlist.

### Blackbox

A pin geometry-only entity that represents layout data that does not exist or can be ignored.

### Breakpoint

Labels you can specify in a Dracula rules file via the \*BREAK command. When PDRACULA processes a file containing these labels, it generates a *jxrun.com* script file that allows you to submit a run with a starting as well as an optional stopping point in the script file, based on these labels.

### CDL

An acronym that stands for *Circuit Description Language*, which is a subset of the SPICE format that describes the device and block level connectivity.

### Cell

A schematic or layout entity that contains leaf (device) or block (instances) data as well as interconnect data.

### Cellbox

Boundary generated by Dracula that encompasses the data inside an Hcell.

### Cell Layers

Layout layer data used to create the canonical devices, such as nwell, pplus, nplus, poly, contact, metal1, and so on.

### Cell Mode

Hierarchical Dracula mode that verifies cells declared as HCELLs individually.

**Composite Error**

Errors reported from a Composite Mode Hierarchical Dracula run.

**Composite Mode**

Hierarchical Dracula mode that verifies the rest of the flattened design not designated as HCELLs.

**Concatenate**

The process of combining two inputs together but do not merge them.

**.DAT**

The file extension of the Dracula binary-generated data files.

**Description Block**

Refers to the section of the Dracula rules file that specifies the administrative commands, which the tool uses for run, such as the layout format and names of input files.

**Discrepancy Number**

Number of errors found during a Dracula LVS run.

**Discrepancy Point**

A mismatch found during an LVS run, based on an unmatched net or device.

**DRC**

An acronym that stands for *Design Rules Checker*, which analyzes the layout data and flags any physical violations, such as spacing or enclosure, based on the commands in the DRC rules files.

**Edtext File**

An ASCII text file lists text strings and its (X,Y) coordinates with respect to the flattened layout on each line. Also, you can optionally specify which conductive layer the text attaches to. If you omit this information, Dracula attaches the text to the conductive layer in the layout, defined in the rules file that's underneath the specified coordinates. You can use this file to supplement layout texts or to supersede the existing layout text that exists at the specified coordinates.

**ERC**

An acronym that stands for *Electrical Rules Checker*, which performs electrical connectivity checks on a layout and flags any violations, based on the commands in the ERC rules file.



**Error Cell**

Graphic structure that contains Dracula generated error flags that are used to display a detected violation, which you can overlay on top of the layout to help locate and resolve these errors.

**Error Flag**

Data generated by Dracula representing a violation based on the commands in a Dracula rules file.

**Extraction**

Process of generating a representation of a layout that contains the layout's electrical functions rather than its physical description.

**Flatten**

Expands the hierarchical layout or schematic data to one common plane.

**HCELL**

A cell you either define or let the software automatically define for you based on certain criteria for a hierarchical run. Dracula verifies a cell defined as an HCELL only once during the run. An HCELL may contain devices only, instances of other cells, or a combination of both.

**Hedtext File**

Is similar in structure to the *edtext* file, but used for supplementing or superseding the layout texts in HCELLs.

**Heuristics**

An exploratory problem-solving technique that utilizes the self-educating techniques to improve performance.

**Hierarchical Mode**

Verification that uses the nested structure of a layout database to perform designated operations on Hcells, the composite plane of the layout, as well as the Hcell-to-Hcell and Hcell-to-Composite interactions.

**Initial Correspondence Node Pair**

Corresponding texted nets with same names in the layout, and schematic used by Dracula LVS for device matching during the verification process.

**Input-Layer Block**

Refers to the section of the Dracula rules file that specifies the layout layers, which you want Dracula to read in for the run.

## **Interconnect Wiring**

Electrical net connections between the canonical devices.

## **jxrun.com**

Is the name of the PDRACULA-generated script file listing all modules called during a run, based on the commands in the rules file.

## **Keyword**

Words reserved by a program for special meanings, for example, commands, functions, or options.

## **Layout**

Is a representation of a circuit in a geometric topological form; instead in a logical or an electrical form.

## **Logical Operation**

An operation that analyzes one or more inputs and generates an output based on a set of rules. Boolean (AND, OR, NOT, XOR).

## **LOGLVS**

Compiler that converts network descriptions or ASCII netlists into a binary file, LVSLOGIC.DAT, used as schematic input for LVS.

## **LPE**

An acronym that stands for *Layout Parameter Extraction*, which extracts and computes the canonical and/or parasitic parameter values from the layout, and creates a CDL or SPICE compatible netlist.

## **LVL**

An acronym that stands for *Layout Versus Layout*, which compares two layouts and reports only the electrical differences between the two. Note that LVL does not report graphical differences.

## **LVS**

An acronym that stands for *Layout Versus Schematic*, which compares the devices and connectivity in the layout to the devices and connectivity in the corresponding schematic and reports discrepancies found in the layout.

## **Macro**

A file that contains a series of commands, which the tool can run during the job.

## Dracula Reference

### Glossary

---

#### **Mask**

Glass plate containing photographically produced images corresponding to patterns from layout data during fabrication.

#### **Matched Devices**

Devices that correspond between the layout and schematic.

#### **Micron**

Unit of measurement equal to 1/1000 of a millimeter.

#### **Mil**

Unit of measurement equal to 1/1000 of an inch.

#### **Model**

Representation of a specific device in the schematic or layout.

#### **Net**

A wire in the schematic or shape in the layout that carries an electrical information.

#### **Netlist**

An ASCII file that lists the components and the connections to them.

#### **Notch**

A set of non-adjacent facing edges or adjacent edges that create an external angle of less than 90 degrees.

#### **Open Circuit**

A design that has an incomplete connection.

#### **Operation Block**

Refers to the section of the Dracula rules file that specifies the commands, which drives the tool to perform the desired DRC, ERC, or LVS functionality.

#### **Orthogonal**

Shapes whose edges form internal right angles and are parallel to the X and Y axis, also known as "*Manhattan Geometry*".

#### **Output Cell**

Graphic structure that contains Dracula generated error flags that are used to display a detected violation, which you can overlay on top of the layout to help locate and resolve these errors.

## Dracula Reference

### Glossary

---

#### **Pad**

Refers to a relatively large metallic area on the periphery of a chip to which I/O wires are attached. These are also called as “*bonding pads*”.

#### **Parameter**

A value associated with a device, such as the width and length of a MOS gate.

#### **Parameter Extraction**

The process of measuring and calculating parameter values associated with a device and adding them to an appropriate netlist for comparison during LVS.

#### **Parasitic**

Electrical properties (not elements) that add circuit characteristics to the layout not included in the schematic.

#### **Parasitic Devices**

Phantom layout devices (such as capacitors or diodes) that are not part of the schematic but act like a canonical device, based on the topological relationship of conductive layers in the layout.

#### **PDRACULA**

Dracula preprocessor that compiles and verifies the Dracula rules file, and creates the UNIX script files to perform the requested job.

#### **PDW**

Parallel pull-down structure.

#### **Pin**

Point on a device or cell in the layout or schematic where you can make an electrical connection.

#### **PRE**

An acronym that stands for *Parasitic Resistors Extraction*, which extracts the parasitic resistors formed by conductor layers, such as poly, metal, and so on, and computes their parameter values.

#### **Primary Cell**

Top-level cell in the design’s layout.

#### **Primitive Device**

Canonical device, for example, transistor, resistor, capacitor, diode, and so on in a schematic or layout.

**Pseudo Contact Layer**

Derived layer used as a contact by Dracula, but not digitized as a contact in the layout.

**Pseudo Gate**

See definitions for "PDW", "PUP", "SDW", or "SUP".

**PUP**

Parallel pull-up structure.

**Rules File**

An ASCII file that defines the commands used by Dracula to perform the requested functionality. A Dracula rules file includes three main parts: the Description block, the Input-Layer block, and the Operation block.

**Scale**

Proportion used in determining the relationship of a representation to that which it represents.

**Schematic**

Is the drawing of the logical representation of an electronic design. The drawing includes symbols, pins, connecting lines, and paths to indicate connectivity. The drawing can also contain other graphics or text, such as sheet borders or notes that are not part of the connectivity data.

**SDW**

Series pull-down structure.

**Sheet Resistance**

Is the factor of resistance for one square of a canonical or parasitic resistor device.

**Smash**

Is the process of reducing parallel or series devices to a single device.

**SPICE**

Stands for *Simulation Program with Integrated Circuit Emphasis*; it is a general-purpose simulation netlist format used for IC checking.

**Stamping**

The process of passing electrical information from one conductive layer to another by virtue of the fact that they touch without the use of a contact or via opening.

**Substrate**

Is a material beneath the layout that encompasses all the layer data.

**.sum file**

Default extension of the Dracula summary file.

**SUP**

Series pull-up structure.

**SUPI**

Serial-up incomplete structure.

**SDWI**

Serial-down incomplete structure.

**Terminals**

Conductor layers used as connections to a device.

**Toggle**

A switch that allows you to change between two states, for example, on or off.

**Trapezoid**

A four-sided polygon constructed with two parallel sides.

**Treefile**

Dracula file containing system and hierarchical information from your layout.

**Unmatched**

Any device or I/O, which does not match schematic to layout or vice-versa.

**Virtual Wire**

Simulated net connection in the layout.

**Wire**

See definition for “*Net*”.

**Working Window**

The bounding box coordinates of the layout, which Dracula processes.

# Index

- (minus sign) in rules files [43](#)

## Symbols

, (comma) in rules files [43](#)  
 ; (semicolon) in rules files [43](#)  
 : (colon), use in text [827](#)  
 .DAT, definition [1256](#)  
 .log files [63](#)  
 .mlg files [63](#)  
 .msm files [63](#)  
 .sum files [63](#)  
 [] (brackets) in rules files [43](#)  
 \* (asterisk) in block statements [41](#)  
 \* (asterisk) in rules files [43](#)  
 / (slash) with PDRACULA commands [47](#)  
 & (ampersand) [870](#)  
 & (ampersand) in rules files [43](#)  
 + (plus sign) in rules files [43](#)  
 = (equal sign) in rules files [43](#)

## Numerics

6GPADS.DAT file [239](#)  
 6PADL.DAT file [239](#)

## A

/ABORT command, PDRACULA [48](#)  
 ABORT-MULT-STAMP command [514](#)  
 ABORT-P-G-SHORT command [516](#)  
 ABORT-SOFTCHK command [517](#)  
 absolute magnification [684](#)  
 active input sequence, defined [47](#)  
 ACUTE-CLIP command [518](#)  
 ALL option, CELL-ERROR-REP  
     command [119](#)  
 ALLOW-RECONNECT command [520](#)  
 ampersand (&) [870](#)  
     used to join commands [81](#)  
 ampersand (&) in rules files [43](#)  
 analog elements in netlist [157](#)  
 AND command [873](#)

angle of rotation [763](#)  
 anode conductor layer [977](#), [1147](#)  
 antenna [907](#)  
 antenna check [1075](#)  
 APPLEOUT module [63](#)  
 Applicon  
     Hcell error file format [121](#)  
 AREA command [878](#)  
 ARRAY command [522](#)  
 ARRAY-ENABLED command [524](#)  
 assign geometries [719](#)  
 asterisk (\*)  
     in block statements [41](#)  
     in rules files [43](#)  
 ATTACH command [841](#)  
 attaching text [966](#)  
 ATTRIBUTE CAP command [884](#)  
 ATTRIBUTE command [810](#)  
 ATTRIBUTE RES command, PRE [891](#),  
     [1150](#)  
 AUTOMATCH command [525](#)

## B

backannotation  
     definition [1255](#)  
     HLPE [402](#)  
     LPE [356](#)  
     LPECHK [397](#)  
     LPESELECT/PRE [1093](#)  
     LVS/PRE [472](#)  
     node numbers [476](#)  
     PRE output files [477](#)  
 BASE-LAYER command [833](#)  
     HLVS [344](#)  
 BBOX [893](#)  
 bipolar  
     devices [369](#)  
     counting nodes [1120](#)  
     defining [971](#)  
     extracting parameters [1075](#)  
     tagging [946](#)  
     LPE extraction [369](#)  
     LVS example [333](#)  
     SPICE file [372](#)

## Dracula Reference

transistor parameter extraction [369](#)  
\*.BIPOLAR command [157](#)  
BJT devices, counting [968](#)  
BJT element [1120](#)  
BJT syntax in CDL [210](#)  
BJT-EA-TOLER command [681](#)  
BJTLPERCENT option  
    LPECHK command [1089](#)  
    LVSCHK command [1113](#)  
BJT-L-TOLER command [681](#)  
BJTWPERCENT option  
    LPECHK command [1089](#)  
    LVSCHK command [1112](#)  
BJT-W-TOLER command [681](#)  
black boxes, defining [343](#)  
blackbox, definition [1255](#)  
BLACKBOX-FILE command [526](#)  
blanks in rules files [43](#)  
BLOCK-NAME-ONLY command [529](#)  
blocks in rules files [41](#)  
boundaries [1225](#)  
bounding boxes [835](#)  
BOUNDING-BOX command [530](#)  
BOXMAP command [531](#)  
brackets ([]) in rules files [43](#)  
\*BREAK command [900](#)  
commands  
    \*BREAK [900](#)  
breakpoint, definition [1255](#)  
bulk layer [393](#)  
\*.BUSDELIMITER command [158](#)

## C

CALCULATE command [902](#)  
CALCULATE command (LPEPRO) [417](#)  
\*.CAPA command [159](#)  
capacitance  
    ATTRIBUTE CAP command [885](#)  
    capacitor type [976](#)  
    ELEMENT CAP command [975](#), [1139](#)  
    fringe [1030](#), [1040](#)  
    parasitic [885](#), [1030](#), [1144](#)  
    pins [739](#)  
    redistributing [484](#)  
    region [975](#)  
    types [391](#)  
CAPACITANCE subcommand  
    (LPEPRO) [428](#), [437](#)  
capacitors

describing in CDL [206](#)  
excluding from LVS [159](#)  
\*.CAPAREA command [161](#)  
CAPAREA option  
    LPECHK command [1088](#)  
    LVSCHK command [1111](#)  
CAP-POLARITY [536](#)  
\*.CAPVAL command [162](#)  
CAPVAL option  
    LPECHK command [1088](#)  
    LVSCHK command [1111](#)  
CARE-SPLIT-ORDER command [537](#)  
CASE command [223](#)  
case sensitivity [121](#)  
cell and text name [552](#)  
CNames-CSEN [552](#)  
TNames-CSEN [808](#)  
CDL  
    control commands [156](#)  
    defining elements [206](#) to [213](#)  
    definition [1255](#)  
    flexible LPE [372](#)  
    Hcell example [270](#)  
    LPESELECT command [477](#)  
    macro expansion [213](#)  
    netlists [268](#)  
    sample file (LOGLVS) [215](#)  
    special characters in netlists [156](#)  
    subcircuit definitions [151](#)  
cell boxes [539](#), [1225](#)  
CELL command [224](#)  
cell layers, definition [1255](#)  
cell library, saving [544](#)  
cell mode  
    definition [1255](#)  
    HDRC [114](#)  
    HERC [144](#)  
    running LOGLVS with SPICE [267](#)  
    specifying [545](#)  
cell names, primary [718](#), [807](#)  
cell processing deletion [564](#)  
cell.tab file [403](#)  
CELLBNDY command [835](#)  
cellbox, definition [1255](#)  
CELLBOX-LAYER command [539](#)  
CELLBOXX layers [112](#)  
CELLBOXX.DAT file [112](#)  
CELL-CHILD-TEXT command [540](#)  
CELL-ERROR-REP command [541](#)  
    options described [119](#)  
CELL-LIBRARY command [544](#)



### cells

- definition [1255](#)
- internally wired [555](#)
- output [963](#)
- paths, tracing [146](#)
- placement [632](#)
  - off-grid [601](#)
- primary [658](#)
- structure [622](#), [753](#), [754](#)
- texts [554](#)

.CET files [144](#)

### character limit

- command names [792](#)
- directory names [750](#)
- layer names
  - CONNECT-LAYER command [838](#)
  - SUBSTRATE command [863](#)
- macro names [791](#)
- subcircuit names [793](#)

check modes (HERC) [144](#)

CHECK-MODE command [545](#)

CHECK-PATH command [548](#)

CHKPAR command [907](#)

CIRCUIT command [226](#)

Circuit Description Language. See CDL

circuit element extraction [130](#)

circuit extractions [401](#)

circuit simulators, supported [33](#)

### circuits

- extracting parameters [871](#)
- short or open, checking for [31](#)
- simulation model [694](#)
- simulator restriction [810](#)

/CLEAR command, PDRACULA [49](#)

CMOS AOI device recognition [322](#)

CMOS inverter device recognition [309](#)

CMOS NAND device recognition [321](#)

CMOS NOR device recognition [321](#)

CMOS technology, silicon-gate [391](#)

CMOS-NTYPE [549](#)

CMOS-PTYPE command [550](#)

CNAMES-CHK-OFF command [551](#)

CNAMES-CSEN command [552](#)

colon (:), use in text [827](#)

comma (,) in rules files [43](#)

command file. See rules file [40](#)

### Commands

- INDISK-FILE [638](#)

### commands

- ABORT-MULT-STAMP [514](#)

- ABORT-P-G-SHORT [516](#)

- ABORT-SOFTCHK [517](#)

- ACUTE-CLIP [518](#)

- ALLOW-RECONNECT [520](#)

- AND [873](#)

- ANDNOT [875](#)

- AREA [878](#)

- ARRAY [522](#)

- ARRAY-ENABLED [524](#)

- ATTACH [841](#), [881](#)

- ATTRIBUTE [810](#)

- ATTRIBUTE CAP [884](#)

- ATTRIBUTE RES [891](#)

- AUTOMATCH [525](#)

- BASE-LAYER [833](#)

- BJT-EA-TOLER [681](#)

- BJT-L-TOLER [681](#)

- BJT-W-TOLER [681](#)

- BLACKBOX-FILE [526](#)

- BLOCK-NAME-ONLY [529](#)

- BOUNDING-BOX [530](#)

- BOXMAP [531](#)

- CALCULATE [902](#)

- CALCULATE (LPEPRO) [417](#)

- CAP-POLARITY [536](#)

- CARE-SPLIT-ORDER [537](#)

### CDL

- \*.BIPOLAR [157](#)

- \*.BUSDELIMITER [158](#)

- \*.CAPA [159](#)

- \*.CAPAREA [161](#)

- \*.CAPVAL [162](#)

- \*.DEFAULT [163](#)

- \*.DIOAREA [164](#)

- \*.DIODE [166](#)

- \*.DIOPERI [165](#)

- \*.EQUATION [167](#)

- \*.EQUIV [168](#), [235](#)

- \*.GNONSWAP [172](#)

- \*.LDD [173](#)

- \*.MEGA [174](#)

- \*.NONSWAP [175](#)

- \*.NOPIN [177](#)

- \*.NOSUB\_M [178](#)

- \*.PIN [179](#)

- \*.PININFO [180](#)

- \*.RESI [182](#)

- \*.RESSIZE [184](#)

- \*.RESVAL [185](#)

- \*.REVERSE [186](#)

- \*.SCALE [187](#)

- \*.SPICE [189](#)

## Dracula Reference

---

|                            |           |                     |      |
|----------------------------|-----------|---------------------|------|
| *.UNSPEC                   | 190       | ELEMENT DIO         | 977  |
| *.UNSPEC-MOS               | 192       | ELEMENT IN?         | 979  |
| CELLBNDY                   | 835       | ELEMENT LDD         | 981  |
| CELLBOX-LAYER              | 539       | ELEMENT MOS         | 983  |
| CELL-CHILD-TEXT            | 540       | ELEMENT PAD         | 985  |
| CELL-ERROR-REP             | 541       | ELEMENT RES         | 986  |
| CELL-LIBRARY               | 544       | EMPTY-BLACKBOX      | 570  |
| CHECK-MODE                 | 545       | EMPTY-ENC           | 571  |
| CHECK-PATH                 | 548       | ENC                 | 988  |
| CHKPAR                     | 907       | ENCBASEDOVLP        | 1000 |
| CMOS-NTYPE                 | 549       | END                 | 423  |
| CMOS-PTYPE                 | 550       | END-MACRO           | 572  |
| CNAMES-CHK-OFF             | 551       | ENVIRONMENT-MAX     | 573  |
| CNAMES-CSEN                | 552       | EQUATION            | 1009 |
| COMPUTE                    | 911       | EQUIV               | 235  |
| CONN-ALL-WIRE              | 554       | ERROR-PATH-WIDTH    | 575  |
| CONNECT                    | 913, 1212 | EXIT                | 413  |
| CONNECT-LAYER              | 838, 1212 | EXPAND-GATE-DELI    | 576  |
| CONN-INTER-WIRE            | 555       | EXPAND-INST-DELI    | 577  |
| CONN-OPEN-WIRE             | 556       | EXPAND-NODE-DELI    | 578  |
| CONVERT-DATABASE           | 557       | EXPORT              | 843  |
| CORNER                     | 915       | EXT                 | 1014 |
| COVERAGE                   | 920       | EXTERM-NOWARN       | 579  |
| CPOINT-FILE                | 559       | FASTSIZE            | 580  |
| CSH-F-OPTION               | 561       | FDELIMITER          | 581  |
| CTEXT                      | 841       | FILTER              | 582  |
| CUT                        | 928       | FILTER-LAY-OPT      | 582  |
| CUT-TERM, PRE              | 930       | FILTER-MOSCAP       | 582  |
| DATAFORMAT                 | 562       | FILTER-OPT          | 582  |
| DD-RSH-COMMAND             | 563       | FILTER-SCH-OPT      | 582  |
| DELAY                      | 416       | FILTER-LDD          | 594  |
| DELCEL (description block) | 564       | FILTER-REPORT       | 595  |
| DEL-VIR-WIRE               | 565       | FIX-INPUT-ORDER     | 596  |
| DENC                       | 936       | FLAG-ACUTEANGLE     | 597  |
| DEVMAP                     | 231       | FLAG-NON45          | 598  |
| DEVTAG                     | 946       | FLAG-OFFGRID        | 601  |
| DIODE-P-TO-G               | 566       | FLAG-PTH-OFFGRID    | 603  |
| DIODESEQ                   | 567       | FLAG-SELFINTERS     | 604  |
| DISK                       | 409       | FLAG-SELFTOUCH      | 606  |
| dracbatch                  | 1249      | FLAT-GDS-SMALL      | 608  |
| DRACBATCH-EMAIL            | 568       | FLATTEN             | 1037 |
| dracq                      | 1249      | FLATTEN-PWRGND      | 609  |
| DrcAntenna                 | 951       | FLG-EXPTH-OFFGRD    | 599  |
| EBOX-SHORT-PIN             | 569       | FLOATCHK            | 1039 |
| ECONNECT                   | 962       | FNODE-CONNECT       | 610  |
| EDTEXT                     | 966       | FRAME-EXCLUDE-LAYER | 845  |
| ELCOUNT                    | 968       | FRAME-HOLE-LAYER    | 846  |
| ELEMENT                    | 946       | FRINGE CAP          | 1040 |
| ELEMENT BJT                | 971       | GATE-RES-EXT        | 612  |
| ELEMENT BOX                | 973       | GENRECT             | 1043 |
| ELEMENT CAP                | 975       | GEN-TEXT            | 623  |

## Dracula Reference

---

GEN-TEXT-FILE [614](#)  
GEN-TEXT-FLTNODE [615](#)  
GEN-TEXT-FRAME [617](#)  
GEN-TEXT-LAYER [847](#)  
GEN-TEXT-WIRE [619](#)  
GEN-XRF-RPT [620](#)  
GET [414](#)  
GPATHCHK [1046](#)  
GPATHDEF [1048](#)  
GROUND-NODE [621](#)  
GROW [1050](#)  
HCELL [622](#)  
HCELL-COLUMN-1 [625](#)  
HCELL-FILE [626](#)  
HCELL-IN-HCELL [627](#)  
HCELL-MAX-PLACEMENTS [629](#)  
HCELL-MAX-SEGMENTS [630](#)  
HCELL-OPTIMIZE-  
    PLACEMENTS [631](#)  
HCELL-RULE [632](#)  
HEDTEXT [1052](#)  
HIERARCHEN [1054](#)  
HIER-OUTDISK [634](#)  
IDTEXT [848](#)  
IGNORE-SCH-SUB [635](#)  
IMPORT-LAYERS [849](#)  
INAME-TOPDOWN [636](#)  
INDISK [637](#)  
INDISK-FILE [638](#)  
INPUT [410](#)  
INSTPIN-FILE [641](#)  
INT [1060](#)  
INTERNAL NODE [643](#)  
INTERNODE-PREFIX [644](#)  
joining in DRC [81](#)  
KCELL-FILE [645](#)  
KEEPDATA [646](#)  
KEEP-INST-FPIN [648](#)  
KEEP-SHORT-BJT [649](#)  
KEEP-SHORT-CAP [650](#)  
KEEP-SHORT-DIO [651](#)  
KEEP-SHORT-MOS [652](#)  
KEEP-SHORT-RES [653](#)  
KEEP-TEXT [654](#)  
LAYER-FILE [655](#)  
LAYER-FILE-TYPE [656](#)  
LCONNECT [1069](#)  
LEGAL-CHAR-SET [657](#)  
LENGTH [1071](#)  
LEXTRACT [1075](#)  
LIBRARY [658](#)  
LIMIT-DRC-ERROR [659](#)  
LIMIT-RUN-TO [660](#)  
LINK [1083](#)  
LISTERROR [663](#)  
LOGLVS  
    CASE [223](#)  
    CELL [224](#)  
    CIRCUIT [226](#)  
    CONVERT [227](#)  
    DELCKT [229](#)  
    DELECEL [230](#)  
    DEVMAP [231](#)  
    DXF [233](#)  
    ENV [234](#)  
    EXIT [236](#)  
    FDELIMITER [237](#)  
    FPIN [238](#)  
    GENPAD [239](#)  
    GET [240](#)  
    GLOBAL [241](#)  
    HELP [242](#)  
    HTV/DRE [243](#)  
    KFPIN [245](#)  
    LINK [246](#)  
    MOS\_EXTENDED\_REP [247](#), [265](#)  
    NO\_WARNING [248](#)  
    NVER [249](#)  
    PRECISION [250](#)  
    PRINT [251](#)  
    RESET [252](#)  
    RESISTOR [253](#)  
    SAVE [254](#)  
    SET FANIN [255](#)  
    SET FANOUT [256](#)  
    STORE [258](#)  
    SUMMARY [259](#)  
    TNAME [260](#)  
    TRANSISTOR [261](#)  
    TYPE [262](#)  
LONG-SUM-LEN [665](#)  
LOOP-TWO-CAP [666](#)  
LOOP-TWO-RES [667](#)  
LPATH [419](#)  
LPECHK [1084](#)  
LPE-FORMAT [668](#)  
LPE-GROUND [669](#)  
LPE-PRECISION [670](#)  
LPE-QUERY [671](#)  
LPESELECT [1093](#)  
LUMPCAP [673](#)  
LVL-PART [675](#)

## Dracula Reference

---

LVSCHK 1103  
LVSCHK and the SMASH  
    qualifier 1113  
LVS-OPTION 676  
LVS-RPT-ONLY 678  
MAG-BEF-GRID 684  
MAGNIFY-IN 685  
MAGNIFY-OUT 686  
MAP-TEXT-FILE 687  
MASK 424  
MATCH-P-G-BY-TYP 688  
MEMORY-CORE 689  
M-Factor Tolerance 681  
MINI-SUMMARY 692  
MIRROR 693  
MODEL 694  
MOS-CAP 695  
MOS-L-TOLER 681  
MOS-W-TOLER 681  
MULTICPU 697  
MULTI-GDS2-IN 698  
MULTILAB 1117  
MULTI-PIN-FILE 700  
MULTIPLE-DISK 702  
NDCOUNT 1120  
NEIGHBOR 1123  
NODE-FILE 1124  
NODE-SELECT 1126  
NO-SUB-TERM 704  
NOT 1128  
NOT-HCELL 705  
NRX-EXTRACT 706  
NRX-SAVE 713  
OCTBIAS 1132  
OPTIMIZE-RULE 715  
OR 1134  
OUTDISK 717  
OUTLIB 718  
OUTPUT-ONE-LAYER 719  
OVERLAP 1137  
OVPR-TOLERANCE 720  
PAD-LAYER 856  
PARALLEL-FILE 721  
PARAMETER CAP 1139  
PARAMETER RES 1141  
PARASITIC DIO 1146  
PAR-COMPARE 723  
PAROUTNAM 725  
PARSET 731  
PATH 421  
PATHCHK 1152

PATH-W-OFFGRID 735  
PATH-W-ROUND 737  
PDRACULA  
    /ABORT 48  
    /CLEAR 49  
    /DCL 50  
    /FINISH 51  
    /GET 53  
    /HELP 55  
    /LIST 57  
    /NEXT 58  
    /PRINT 59  
    /RESTART 60  
    /STATUS 61  
    /STORE 62  
PINCAP-FILE 739  
PIN-TEXT-LAYER 859  
PLENGTH 1156  
POINT-CONNECT 740  
POSTENC 1158  
POWER-NODE 741  
PRE  
    ATTRIBUTE RES 1150  
    PARASITIC CAP 1143  
    PARASITIC RES 1150  
    RCONNECT 1161  
PREFIX-PARASITIC 742  
PREINTERACTIVE 743  
PRIMARY 745  
PRINT, LPEPRO 426  
PRINTFILE 746  
PROBE 1159  
PROB-GEOM-LIST 747  
PROGRAM-DIR 750  
PULL-DOWN 751  
PULL-UP 752  
RAM-CELL 753  
RAM-CORE 754  
RCONNECT-LAYER 861  
RECTCHK 1163  
REDUCER 756  
RELOCATE 1166  
REMOVE-SCH-OPT 582  
REP-UNM-INST-PG 758  
RERUN-FILTER 759  
RESET 411  
RESOLUTION 760  
REV-SCALE 765  
REV-SCALEIN 765  
REV-SCALEOUT 766  
RLENGTH 1168

## Dracula Reference

---

ROTATION [763](#)  
ROUND-OFF-TOLER [764](#)  
RSPFSELECT [1170](#)  
SAMELAB [1171](#)  
SAVE [412](#)  
SCALE [765](#)  
SCALEIN [765](#)  
SCALEOUT [766](#)  
SCONNECT [1173](#)  
SELECT [1176](#)  
SELECT-MODE [768](#)  
SHRINK [1190](#)  
SIZE-MAX-VERTEX [769](#)  
SIZE-MIN-WIDTH [770](#)  
SIZENOT [772](#)  
SKILL-LOADER [773](#)  
SMASH-CAP-TYP [779](#)  
SMASH-FLOAT-RES [780](#)  
SMASH-MOS-TYP [781](#)  
SMASH-RES-TYP [785](#)  
SMOOTH-SIZE [786](#)  
SNAP [1205](#)  
SOFTCHK [1206](#)  
S-OPTION-CLASS [788](#)  
SPATH [420](#)  
SPF-MAP-FILE [789](#)  
SPFSELECT [1208](#)  
SPICE  
    .ENDS [194](#)  
    .EOM [194](#)  
    .GLOBAL [195](#)  
    .INCLUDE [196](#)  
    .MACRO [200](#)  
    .PARAM [197](#)  
    .SUBCKT [200](#)  
    .SWAP [202](#)  
    subcircuit calls [204](#)  
STAMP [1212](#)  
START [422](#)  
START-MACRO [791](#)  
STATUS-COMMAND [792](#)  
STORE [426](#)  
SUBCKT-NAME [793](#)  
SUBNODE-DELIM, PRE [794](#)  
SUBSTRATE [863](#)  
SUBTYPE-CSENS [797](#)  
SUMMARY-ONLY [798](#)  
SVS-LAYOUT [799](#)  
SVS-SCHEMATIC [800](#)  
SYSIN [803](#)  
SYSOUT [801](#)  
SYSTEM [803](#)  
TEMPORARY-LAYER [865](#)  
TEXT-LEVEL [806](#)  
TEXT-PRI-ONLY [807](#)  
TEXTSEQUENCE [867](#)  
TNAMES-CSEN [808](#)  
TRIANGLE [1215](#)  
TYPE [426](#)  
UNIT [810](#), [1141](#)  
UNMASK [425](#)  
UNSPEC- SCH -PARA [814](#)  
UNSPEC-LAY-PARA [811](#)  
UNSPEC-PARA [812](#)  
WIDTH [1216](#)  
WINDEL [815](#)  
WINDOW [817](#)  
WINDOW-CUT [819](#)  
WORK-DIR [820](#)  
XBOX [1225](#)  
XCELL [1227](#)  
XDEVICE [1229](#)  
XOR [1231](#)  
XVIA [1234](#)  
ZERO-SPAC-F-EQU [822](#)  
comments in rules files [43](#)  
compilation  
    SPICE netlists [274](#)  
components  
    connecting [472](#)  
    parasitic resistor regions [472](#)  
composite error, definition [1256](#)  
composite mode  
    applying black box features [526](#)  
    HDRC [114](#)  
    HERC [144](#)  
    hierarchical checking, description [37](#)  
    running LOGLVS with SPICE [268](#)  
    specifying [546](#)  
composite mode, definition [1256](#)  
composite plane  
    description [35](#)  
    geometries [833](#)  
    HDRC [112](#)  
COMPUTE command [911](#)  
concatenate, definition [1256](#)  
conduction layers [861](#)  
    creating parasitic resistors [484](#)  
    reconnecting [475](#)  
    resistance terminal layers [475](#)  
    splitting [479](#)  
conduction nodes, creating resistors [479](#)

## Dracula Reference

---

conductive terminals [478](#), [479](#)  
conductor layers [1234](#)  
conjoined group [1075](#)  
conjunctive  
    rename [988](#), [1015](#)  
    rules [870](#)  
        using ampersand (&) [43](#)  
    set [870](#)  
conjunctive rules [81](#)  
CONN option, ECONNECT [962](#)  
CONN option, LCONNECT  
    command [1069](#)  
CONN-ALL-WIRE command [554](#)  
CONNECT command [913](#), [1212](#)  
connect layers [1234](#)  
connected node listing conventions, LVS  
    report [295](#)  
connectivity  
    interlayer [913](#)  
CONNECT-LAYER command [838](#), [1212](#)  
CONN-INTER-WIRE command [555](#)  
CONN-OPEN-WIRE command [556](#)  
contact regions [478](#)  
contacts  
    floating [1039](#)  
    layers [1173](#), [1234](#)  
continuation line in DSPF format [500](#)  
control commands, LOGLVS [153](#)  
control statements, syntax [43](#)  
conversion  
    logic network data [227](#)  
    schematic netlists [147](#)  
CONVERT command [227](#)  
CONVERT-DATABASE command [557](#)  
converting databases [557](#)  
CORNER [915](#)  
corner junction resistances [482](#)  
CORNER option, LPESELECT  
    command [1099](#)  
CORNER or WEFECT calculations [360](#),  
    [1090](#)  
correspondence nodes [149](#), [150](#), [277](#)  
correspondence points [826](#)  
COVERAGE command [920](#)  
CPOINT-FILE command [559](#)  
critical internal nodes [826](#)  
CROSREF.LIS file [263](#)  
cross reference file, LOGLVS [263](#)  
cross-shaped junction resistance [483](#)  
CSH-F-OPTION command [561](#)  
CTEXT command [841](#)

CUT command, PRE [928](#)  
CUT option, SELECT command [1180](#)  
CUT-TERM command, PRE [930](#)

## D

data files  
    specifying directories for  
        SVS [799](#) to [800](#)  
database comparison [557](#), [843](#), [849](#)  
databases  
    changing formats [557](#)  
    compatibility between Dracula  
        versions [35](#)  
DATAFORMAT command [562](#)  
/DCL command, PDRACULA [50](#)  
DD-RSH-COMMAND command [563](#)  
debugging LVS errors [324](#)  
\*.DEFAULT command [163](#)  
definition layer, PARASITIC DIO [1147](#)  
delay calculation [417](#)  
DELAY command [416](#)  
delay equation options [462](#)  
DELAY subcommand (LPEPRO) [428](#)  
delay values  
    calculating [408](#)  
    default rise and fall [416](#)  
    resetting [411](#)  
    reviewing [468](#)  
DELCEL command (description block) [564](#)  
DELCKT command (LOGLVS) [229](#)  
DELECEL command (LOGLVS) [230](#)  
delimiters in DSPF format [500](#)  
delimiters, using in rules files [43](#)  
DEL-VIR-WIRE command [565](#)  
DENC command [936](#)  
Description block  
    creating a rules file [41](#)  
    DRC example [81](#)  
    ERC example [135](#)  
    LPE example [390](#)  
Description block, definition [1256](#)  
design rule, example [95](#)  
Design Rules Checker, definition [1256](#)  
Design Rules Checker. See DRC  
device filtering [363](#)  
device recognition [306](#) to [324](#)  
devices  
    bipolar [946](#), [971](#), [1075](#)  
    extra [1229](#)

## Dracula Reference

---

- matched
  - definition [1259](#)
- MOS [1120](#)
- numbers, linking [946](#)
- special [973](#)
- user-defined [973](#)
- DEVMAP command [231](#)
- DEVTAG command [946](#)
- \*.DIOAREA command [164](#)
- DIOAREA option
  - LPECHK command [1088](#)
  - LVSCHK command [1111](#)
- \*.DIODE command [166](#)
- DIODE subcommand (LPEPRO) [430](#), [438](#)
- DIODE-P-TO-G command [566](#)
- diodes
  - describing in CDL [207](#)
  - example [391](#)
  - excluding from LVS [166](#), [230](#)
  - extracting parameters [362](#)
  - information [438](#)
  - parameters, listing order [567](#)
  - parasitic [742](#)
  - type [977](#)
- DIODESEQ command [567](#)
- \*.DIOPERI command [165](#)
- DIOPERI option
  - LPECHK command [1088](#)
  - LVSCHK command [1111](#)
- directory name [750](#)
- DISC option, LCONNECT command [1069](#)
- discrepancy number, definition [1256](#)
- discrepancy point, definition [1256](#)
- discrepancy points [277](#), [295](#)
- discrepancy report
  - conventions [293](#)
  - LVL [675](#)
- DISK command [409](#)
- disk file name [637](#)
- disk usage [703](#)
- distributed Dracula
  - control scripts [70](#)
  - DRAC-FILE.TAB file [71](#)
  - DRAC-SYNC.TAB file [74](#)
  - DRAC-TASK.TAB file [72](#)
  - PARALLEL-FILE command [721](#)
  - prerequisites [66](#)
  - running [70](#)
  - workstations [65](#)
- dracbatch command [1249](#)
- DRACBATCH-EMAIL command [568](#)

- DRAC-FILE.TAB file [71](#)
- DRACHELP4 environment variable [55](#),  
[242](#), [264](#)
- dracq command [1249](#)
- DRAC-SYNC.TAB file [74](#)
- DRAC-TASK.TAB file [72](#)
- drain conductor layer [981](#)
- DRC
  - commands, described [79](#)
  - database comparison, example [109](#)
  - definition [31](#)
  - description block example [82](#)
  - executing from design rule library [91](#)
  - hierarchical [110](#)
  - Input-layer block example [82](#)
  - Operation block example [83](#)
  - results, viewing [96](#)
  - run, example [81](#)
  - texting [827](#)
- drc checks
  - ENCBASEDOVLP (enclosed based overlap) [1000](#)
- DRC errors, restricting number of [659](#)
- dropped data [605](#)
- DSPF
  - DELIMITER “C” statement [500](#)
  - Dracula and CDC flow [506](#)
  - format [496](#)
  - GROUND\_NET statement [500](#)
  - instance pin descriptions [499](#)
  - NET statement [498](#)
  - pin descriptions [498](#)
  - reducing file size [1170](#)
  - specifying pin capacitance [498](#)
  - subnode descriptions [499](#)
- DXF command [233](#)

## E

- EAPER option
  - LPECHK command [1089](#)
  - LVSCHK command [1112](#)
- EBOX-SHORT-PIN command [569](#)
- ECONNECT command [962](#)
- edge-pair error [663](#)
- edges
  - nonparallel [991](#), [1017](#)
  - parallel [991](#), [1017](#)
  - projecting [992](#), [1019](#)
  - separation [1014](#)

## Dracula Reference

---

- violation [992, 1019](#)
- EDTEXT command [966](#)
- edtext file, definition [1256](#)
- ELCOUNT command [968](#)
- electrical circuit elements [871](#)
- electrical node extraction [130](#)
- electrical nodes [827, 871](#)
- electrical properties [884](#)
- Electrical Rules Checker, definition [1256](#)
- Electrical Rules Checker. See ERC
- ELEMENT BJT command [971](#)
- ELEMENT BOX command [973](#)
- element boxes [200](#)
- ELEMENT CAP command [975](#)
- element counting [145](#)
- ELEMENT DIO command [977](#)
- element extraction commands [871](#)
- ELEMENT IN? [979](#)
- ELEMENT LDD command [981](#)
- ELEMENT MOS command [983](#)
- element names, specifying delimiter [577, 578](#)
- ELEMENT PAD command [985](#)
- ELEMENT RES command [986](#)
- element resistance
  - computation
    - diffusion resistor [366](#)
    - multiple terminal-drawn resistors [366](#)
    - poly-silicon resistor [365](#)
    - problem resistors [367](#)
  - parameter extraction [364](#)
- element subtypes, equating to CDL model names [168, 235](#)
- elements
  - connected [962](#)
  - counting [968](#)
  - redefining [477](#)
- emitter, definition [1256](#)
- EMPTY-BLACKBOX command [570](#)
- EMPTY-ENC command [571](#)
- ENC command [988](#)
- ENCBASEDOVLP (enclosed based overlap) check [1000](#)
- ENCLOSE option, SELECT command [1180](#)
- END command [423](#)
- END-MACRO command [572](#)
- .ENDS command [194](#)
- ENV command [234](#)
- Environment Variables [264](#)
- CDL\_BUSPORT\_DESCEND [264](#)
- DRAC\_EQUIVS [264](#)
- DRACHelp4 [55, 242, 264](#)
- FNET\_WARN\_LIMIT [264](#)
- MOS\_EXTENDED\_REP [247, 265](#)
- ENVIRONMENT-MAX command [573](#)
- .EOM command [194](#)
- equal sign (=) in rules files [43](#)
- EQUATION command [1009](#)
- \*.EQUATION command [167](#)
- EQUIV command [235](#)
- \*.EQUIV command [168, 235](#)
- ERC
  - commands, description [130](#)
  - definition [31](#)
  - Description block example [136](#)
  - errors checked [31](#)
  - errors detected [129](#)
  - Input-Layer example [136](#)
  - layer number maximum [132](#)
  - Operation block example [137](#)
  - sample CMOS rules file [141](#)
  - sample NMOS rules file [138](#)
  - summary file, example [133](#)
  - texting [827](#)
- .ERC files [1207](#)
- .ERC output file [133](#)
- error cell
  - definition [1257](#)
- error cells
  - composite Hcells [118](#)
  - file names (HDRC) [121](#)
- error flags [991, 1017](#)
  - definition [1257](#)
- error messages
  - fatal system [64](#)
  - PDRACULA [45](#)
  - temporary layers [865](#)
- error summary files (HDRC) [122](#)
- error-flag width [575](#)
- ERROR-PATH-WIDTH command [575](#)
- errors
  - extended listing [663](#)
  - HDRC segments [541](#)
  - listing [663](#)
  - LVS [295, 676](#)
  - output cell [717](#)
  - specifying HDRC format [541](#)
- examples
  - drc ENCBASEDOVLP (enclosure based overlap) check [1004](#)



## Dracula Reference

---

EXIT command [236](#), [413](#)  
expand commands, LOGLVS [155](#)  
EXPAND-GATE-DELI command [576](#)  
EXPAND-INST-DELI command [577](#)  
EXPAND-NODE-DELI command [578](#)  
EXPORT command [843](#)  
EXT command [1014](#)  
EXTERM-NOWARN command [579](#)  
extracting parasitics [472](#), [1030](#)  
extraction, definition [1257](#)

## F

fall delay time [416](#)  
FASTSIZE command [580](#)  
fatal system errors [64](#)  
FDELIMITER command [237](#), [581](#)  
feedback loops, mask out [441](#)  
feedthroughs (FTHRU) in Hcells [347](#)  
files  
    6GPADS.DAT [239](#)  
    6PADL.DAT [239](#)  
    compiling [53](#)  
    HEDTEXT [614](#)  
    jxrun.com [552](#), [792](#)  
    log [516](#), [746](#)  
    printfile.sum [516](#), [597](#)  
    run.log [746](#)  
    summary [663](#)  
filter options [525](#), [582](#)  
FILTER summary, LVS report [287](#)  
filtering  
    floating bipolar transistors [592](#)  
    floating source/drain [583](#)  
    gate to ground [585](#), [589](#)  
    gate to power [589](#)  
    gate to power/ground [587](#)  
    ground only path [584](#)  
    N-channel to ground [585](#)  
    no path to power/ground [588](#)  
    P-channel to power [586](#)  
    power only path [584](#)  
    power/ground connection [583](#)  
    pre-set [592](#)  
    source/drain to floating [588](#)  
    source/drain to power [587](#)  
FILTER-LDD command [594](#)  
FILTER-REPORT command [595](#)  
/FINISH command, PDRACULA [51](#)  
FIX-INPUT-ORDER command [596](#)

FLAG-ACUTEANGLE command [597](#)  
FLAG-NON45 command [598](#)  
FLAG-OFFGRID command [601](#)  
FLAG-PTH-OFFGRID command [603](#)  
FLAG-SELFINTERS command [604](#)  
FLAG-SELFTOUCH command [606](#)  
flat mode  
    description [37](#)  
    HDRC [114](#)  
    HERC [144](#)  
    specifying [545](#)  
FLAT-GDS-SMALL command [608](#)  
FLATTEN command [1037](#)  
flatten commands, LOGLVS [155](#)  
flatten, definition [1257](#)  
flattened layers [1054](#)  
FLATTEN-PWRGND command [609](#)  
flexible LPE [372](#)  
FLG-EXPTH-OFFGRD [599](#)  
FLOATCHK command [1039](#)  
floating contacts [1039](#)  
floating nodes [238](#), [245](#)  
    checking for [31](#)  
    texting [615](#)  
flow path, connectivity [1048](#)  
FORTRAN expression [1009](#)  
FPIN command [238](#)  
FRINGE CAP command [1040](#)  
fringe capacitors [1040](#)

## G

gate conductor layer [981](#), [983](#)  
gate resistance, extracting [612](#)  
gate types  
    AND-OR INVERTER [452](#)  
    AOI (AND-OR Inverter) [452](#)  
    BINV [449](#)  
    DINV (D-Type Buffer-Inverter) [449](#)  
    INV [448](#)  
    NAND [450](#)  
    NMOS [444](#)  
    NOR [451](#)  
    NRXF [445](#)  
    NXFR [445](#)  
    OAI (OR-AND Inverter) [458](#)  
    PDW [447](#)  
    PMOS [444](#)  
    PUP [446](#)  
    PXFR [445](#)

## Dracula Reference

---

- RES [444](#)
- SDW [447](#)
- SUP [445](#)
- gate-arrays, base layers [833](#)
- GATE-RES-EXT command [612](#)
- gates
  - specifying delimiter [576](#)
  - swapping [596](#)
- GDS2OUT module [63](#)
- GDSII
  - Hcell error files [121](#)
  - layer name example [853](#)
  - layer names [851](#)
  - SYSTEM command [803](#)
- GENPAD command [239](#)
- GENRECT command [1043](#)
- GEN-TEXT
  - Description block commands [348](#)
  - Input-Layer block commands [349](#)
  - using frame options [349](#)
- GEN-TEXT command [623](#)
- HLVS [348](#)
- GEN-TEXT-FILE command [614](#)
- GEN-TEXT-FLTNODE command [615](#)
- GEN-TEXT-FRAME command [617](#)
- GEN-TEXT-LAYER command [847](#)
- GEN-TEXT-WIRE command [619](#)
- GEN-XRF-RPT command [620](#)
- geometric primitives [373](#)
- geometries
  - attaching text to [829](#)
  - composite plane [833](#)
  - distance between [885](#)
  - problem listing [746](#)
- GET command [240, 414](#)
- /GET command, PDRACULA [53](#)
- GLOBAL command [241](#)
- .GLOBAL command [195](#)
- global node names [195](#)
- global signals [559](#)
- \*.GNONSWAP command [172](#)
- GPATHCHK command [1046](#)
- GPATHDEF command [1048](#)
- graphic output (ERC) [131](#)
- graphic output file [1207](#)
- graphics databases, supported [33](#)
- graphics system text [966](#)
- grid [1205](#)
  - step-size [597](#)
- ground names in SPICE files [217](#)
- ground net in DSPF format [500](#)

- ground nodes, texting [826](#)
- ground pad [832](#)
- GROUND-NODE command [621](#)
- GROW command [1050](#)

## H

- HCELL command [622](#)
- Hcell plane [35](#)
- HCELL table
  - column definition for SVS [625](#)
- HCELL table, column definition for SVS [625](#)
- HCELL-COLUMN-1 command [625](#)
- HCELL-FILE command [626](#)
- HCELL-IN-HCELL command [627](#)
- HCELL-MAX-PLACEMENTS command [629](#)
- HCELL-MAX-SEGMENTS command [630](#)
- HCELL-RULE command [632](#)
- Hcells
  - automatic selection [340](#)
  - automatic selection for HDRC [622](#)
  - BASE-LAYER command [344](#)
  - boundaries [833, 1225](#)
  - cell boxes [539](#)
  - composite error [119](#)
  - definition [1257](#)
  - elements on boundary [145](#)
  - environment [112, 573](#)
  - error cells [120](#)
  - errors [122](#)
  - errors, displayed [541](#)
  - excluding [705](#)
  - external connections [340](#)
  - feedthroughs [347](#)
  - frames [346](#)
  - generating frame inside [617](#)
  - GEN-TEXT commands [348](#)
  - Hcell plane [35](#)
  - HDRC [111](#)
  - layout text [540](#)
  - mapping layers [833](#)
  - multilevel [54](#)
  - multilevel DRC [116](#)
  - number of elements [402](#)
  - overlapping geometries [339, 1227](#)
  - parameters [629, 630, 632](#)
  - pins [340](#)
  - plane [113](#)

## Dracula Reference

---

- resolution [340](#)
  - restrictions [339](#)
  - selecting HDRC Hcells [111](#)
  - selection [339](#)
  - text [847](#)
  - text file name [1052](#)
  - texting [341](#), [348](#)
  - texting automatically [614](#)
  - HDRC
    - error reports [118](#)
    - error segments [541](#)
    - error summary [125](#)
    - HCELL command [622](#)
    - Hcell selection [111](#), [630](#)
    - input command file [127](#)
    - modes [114](#)
    - overview [110](#)
    - prerequisites [110](#)
    - spacing checks [573](#)
  - HEDTEXT command [1052](#)
  - HEDTEXT file [614](#)
  - hedtext, definition [1257](#)
  - HELP command [242](#)
  - /HELP command, PDRACULA [55](#)
  - HERC
    - check modes [144](#)
    - commands [145](#)
    - limitations [144](#)
  - heuristics, definition [1257](#)
  - HIER option, CELL-ERROR-REP command [119](#)
  - HIERARCHEN command [1054](#)
  - Hierarchical Design Rules Checking. See HDRC
  - hierarchical Dracula, functions [37](#)
  - hierarchical Dracula, hierarchy illustrated [36](#)
  - Hierarchical Electrical Rules Checking. See HERC
  - hierarchical layers [1037](#), [1054](#)
  - Hierarchical Layout Parameter Extraction. See HLPE
  - Hierarchical Layout Versus Schematic. See HLVS
  - hierarchical mode
    - definition [1257](#)
    - HDRC [114](#)
    - specifying [546](#)
  - hierarchical mode, description [37](#)
  - Hierarchical Parasitic Resistance Extraction. See HPRE
  - HIER-OUTDISK command [634](#)
  - HLPE [401](#)
    - antenna check [402](#)
  - HLVS
    - BASE-LAYER command [344](#)
    - black boxes [343](#)
    - error types [350](#)
    - feedthroughs [347](#)
    - GEN-TEXT commands [348](#)
    - HCELL command [622](#)
    - Hcells
      - constraints [339](#)
      - frames [346](#)
      - overlapping [344](#)
      - pin names [341](#)
      - pins [340](#)
    - pin text [342](#)
    - requirements [338](#)
    - sample input command file [350](#)
    - selecting Hcells [339](#)
    - texting requirements [341](#)
    - wire types [346](#)
  - HOLE option, SELECT command [1177](#)
  - HTV/DRE command [243](#)
- I**
- IDLAB module [133](#)
  - IDTEXT command [848](#)
  - IGNORE-SCH-SUB command [635](#)
  - illegal characters in node names [217](#)
  - IMAGE.LIS files [409](#), [560](#)
  - IMPORT-LAYERS command [849](#)
  - INAME-TOPDOWN command [636](#)
  - .INCLUDE command [196](#)
  - INDISK command [637](#)
  - INDISK-FILE command [638](#)
  - Inductors
    - PARSET and ELEMENT IN? commands [385](#)
    - square and circular inductors [385](#)
  - initial correspondence nodes
    - definition [1257](#)
    - LVS [149](#), [277](#)
    - LVS pad text [828](#)
    - node names [150](#)
  - initial correspondence points
    - locating [525](#)
    - specifying files [559](#)
  - INPUT command [410](#)

- input command listing [746](#)
- input database [843](#)
- input file expansion dimensions [522](#), [524](#)
- input file name [637](#)
- input layer [828](#), [1128](#)
- input signals [559](#)
- Input-Layer block [849](#)
  - creating a rules file [41](#)
  - DRC example [82](#)
  - ERC sample [136](#)
  - LPE example [393](#)
- InQuery, viewing HLVS errors [743](#)
- INSIDE option, SELECT command [1177](#)
- INSTPIN-FILE command [641](#)
- INT command [1060](#)
- interchangeable inputs [327](#)
- interconnect wiring, definition [1258](#)
- interconnects (feedthroughs) [347](#)
- interfaces [33](#)
- interlayer connectivity [913](#), [1161](#)
- internal node numbers [443](#)
- INTERNAL-NODE command [643](#)
- Internal-Wire
  - text [619](#)
  - wire types [346](#)
- INTERNODE-PREFIX command [644](#)
- ipoint.txt file [559](#)
- isolating LVS errors [324](#)

## J

- JFETs, describing in CDL [211](#)
- JK flip-flops
  - CMOS example [271](#)
- JOURNAL.LPE files [408](#)
- junction resistance
  - calculating [482](#)
  - corner [482](#)
  - cross-shaped [483](#)
  - CUT-TERM command [478](#)
  - T-shaped [482](#)
- jxclean.com control script [70](#)
- jxexe.com control script [70](#)
- jxpar.com control script [70](#)
- jxrun.com files [45](#)
  - case sensitivity [552](#), [808](#)
  - rules file [45](#)
  - STATUS-COMMAND command [792](#)
- jxrun.com, definition [1258](#)
- jxsub.com command

- running LVS [280](#)

## K

- KCELL-FILE command [645](#)
- KEDTEXT command [1068](#)
- KEEPDATA command [646](#)
- KEEP-INST-FPIN command [648](#)
- KEEP-SHORT-BJT command [649](#)
- KEEP-SHORT-CAP command [650](#)
- KEEP-SHORT-DIO command [651](#)
- KEEP-SHORT-MOS command [652](#)
- KEEP-SHORT-RES command [653](#)
- KEEP-TEXT command [654](#)
- keywords [733](#)
  - definition [1258](#)
- KFPIN command [245](#)

## L

- labels [962](#), [1117](#), [1159](#)
- layer names
  - duplicate [825](#)
  - multiple [838](#)
  - repeated [865](#)
  - restrictions [828](#)
  - specifying [851](#)
  - specifying substrate [863](#)
  - specifying text sequence [867](#)
- layer numbers
  - defining [851](#)
- layer numbers, defining [132](#)
- LAYER-FILE command [655](#)
- LAYER-FILE-TYPE command [656](#)
- layers
  - bottom [839](#)
  - cathode conductor [977](#)
  - combining [843](#), [849](#)
  - comparing [843](#)
  - composite plane [833](#)
  - conductor [393](#), [838](#), [1173](#)
  - creating with AND [873](#)
  - creating with AREA [878](#)
  - creating with CUT-TERM [930](#)
  - drain conductor [981](#)
  - gate conductor [981](#)
  - Hcell text [847](#)
  - mapping layout to Dracula [41](#)
  - multiple [946](#)

## Dracula Reference

---

- naming [851](#)
- numbers [863](#)
- parasitic capacitance [1030](#)
- pseudo contact [391](#)
- sizing [789](#)
- soft connect [1173](#)
- source conductor [981](#)
- substrate [863](#)
- substrate conductor [982](#)
- tagging [946](#)
- top [839](#)
- trapezoid [769](#)
- layout circuit extraction [478](#), [479](#)
- layout database [1103](#)
- layout element [694](#)
- Layout Parameter Extraction Postprocessor.  
See LPEPRO
- Layout Parameter Extraction,  
definition [1258](#)
- Layout Parameter Extraction. See LPE
- layout parameters
  - database [356](#)
  - scanning [406](#)
- layout parasitics, extracting with selective  
LPE [671](#)
- layout section, LVS report [293](#)
- layout text [540](#)
- layout to layout, comparing [281](#)
- layout to schematic, comparing [275](#)
- layout verification process [39](#)
- Layout Versus Layout, definition [1258](#)
- Layout Versus Layout. See LVL
- Layout Versus Schematic, definition [1258](#)
- Layout Versus Schematic. See LVS
- layout, definition [1258](#)
- LCONNECT command [1069](#)
- \*.LDD command [173](#)
- LEGAL-CHAR-SET command [657](#)
- LENGTH command [1071](#)
- LEXTRACT command [1075](#)
- LIBRARY command
  - Description block [658](#)
- lightly doped drain MOS [981](#)
- LIMIT-DRC-ERROR command [659](#)
- LIMIT-RUN-TO command [660](#)
- line continuation in DSPF format [500](#)
- LINK command
  - LOGLVS [246](#)
  - Operation block [1083](#)
- linking device numbers [946](#)
- /LIST command, PDRACULA [57](#)
- LISTERROR command [663](#)
- listing file [746](#)
- LISTUNMATCH option
  - LPECHK command [1089](#)
  - LVSCCHK command [1112](#)
- log files
  - cell origins [663](#)
  - opens and shorts [516](#)
  - PAD-LAYER errors [481](#)
  - PRINTFILE output [746](#)
- logic circuit [622](#)
- logic gates
  - formation [356](#)
  - generator [443](#)
- logic networks
  - information [406](#)
  - logic gate generator [443](#)
  - retrieving information [414](#)
  - saving current information [412](#)
  - saving information [412](#)
- logic simulations [355](#)
- logic simulators, supported [33](#)
- logical equivalence [327](#)
- logical name [820](#)
- logical operation commands, described [79](#)
- logical operations
  - commands [869](#)
  - comparing layers [557](#), [849](#)
  - definition [1258](#)
  - text layers [859](#)
- LOGLVS
  - CDL example [155](#)
  - command sequence [153](#)
  - commands [221](#) to ??
  - cross reference file [263](#)
  - examples [271](#)
  - hierarchical run example [267](#)
  - JKFF example [271](#)
  - netlists, preparing [151](#)
  - SPICE file example [274](#)
  - transistor-level example [273](#)
  - Verilog examples [156](#)
- LOGLVS, definition [1258](#)
- LONG-SUM-LEN command [665](#)
- LOOP-TWO-CAP commands [666](#)
- LOOP-TWO-RES command [667](#)
- LPATH command [419](#)
- LPATH subcommand (LPEPRO) [427](#)
- LPE
  - bipolar parameters [369](#)
  - defaults

## Dracula Reference

---

- ELEMENT BJT [381](#)
- ELEMENT CAP [378](#)
- ELEMENT DIO [376](#), [377](#)
- ELEMENT LDD [376](#)
- ELEMENT MOS [375](#)
- ELEMENT RES [381](#)
- PARASITIC CAP [378](#), [379](#)
- PARASITIC DIO [377](#)
- definition [32](#)
- Description block command file
  - DIODESEQ [392](#)
  - MODEL [392](#)
  - SCHEMATIC [392](#)
  - UNIT [392](#)
- diode listing [567](#)
- flexible [372](#)
- flow diagram [357](#)
- Input-Layer block command file
  - CONNECT-LAYER [393](#)
  - SUBSTRATE [393](#)
- keywords [375](#)
- Operation block command file
  - AND [395](#)
  - LPECHK [397](#)
  - LPESELECT [397](#)
  - NOT [395](#)
  - PARASITIC CAP [395](#)
  - PARASITIC DIO [396](#)
- overview [355](#)
- parameter units [810](#)
- PARSET keywords [375](#)
- primitives [373](#)
- reports [356](#)
- running LPE/LPEPRO [464](#)
- selective [671](#)
- selective, running [388](#)
- texting [827](#)
- user-defined parameters [374](#)
- LPECHK command [1084](#)
- LPE-FORMAT command [668](#)
- LPE-GROUND [669](#)
- LPE-PRECISION command [670](#)
- LPEPRO
  - arithmetic symbols [438](#)
  - calculating delay values [408](#), [461](#)
  - commands [407](#)
  - delay equation options [462](#)
  - ending the program [413](#)
  - generating a SPICE file [439](#)
  - generating logic gates [407](#)
  - generating reports [408](#)
  - interactive functions [356](#)
  - internal feedback loop or branch [467](#)
  - L/W ratio [462](#)
  - logic simulation [461](#)
  - LOGLVS reference file [254](#)
  - multiple output devices [467](#)
  - multiple stages module [466](#)
  - path analysis [408](#)
  - relationship to LPE, illustrated [357](#)
  - revising schematic data [464](#)
  - running LPE/LPEPRO [464](#)
  - scanning parameters [433](#)
  - schematic data [441](#), [442](#)
  - selected paths [441](#)
  - start and end points, resetting [411](#)
  - timing verification [461](#)
  - wired-OR examples [460](#)
- LPEPRO subcommands
  - CAPACITANCE [428](#), [437](#)
  - DELAY [428](#)
  - DIODE [430](#), [438](#)
  - LPATH [427](#)
  - MOS [429](#), [434](#)
  - PARAMETER [431](#), [438](#)
  - PATH [427](#)
  - SPATH [427](#)
  - UNREACH [433](#)
- LPE-QUERY command [671](#)
- LPERCENT option
  - LPECHK command [1088](#)
  - LVSCHK command [1109](#)
- LPESELECT command [1093](#)
- LT/LE option, WIDTH command [1219](#), [1220](#)
- LUMPCAP command [673](#)
- LVL
  - commands to run [285](#)
  - definition [31](#), [32](#)
  - overview [281](#)
  - prerequisites [282](#)
  - rules files [283](#)
- LVL-PART command [675](#)
- LVS
  - correspondence nodes [277](#)
  - definition [31](#)
  - device recognition [306](#) to [324](#)
  - discrepancies [325](#)
  - discrepancy report [293](#)
  - error summary file, sample [325](#)
  - errors
    - debugging [325](#)

## Dracula Reference

---

- device subtype mismatch [301](#)
- filtered out layout MOS devices [306](#)
- inconsistently matched device [298](#)
- isolating [325](#)
- matched device to unmatched node [298](#)
- matched node to extra layout devices [299](#)
- matched node to extra schematic devices [299](#)
- matched node to no device [297](#)
- matched node to unmatched layout and schematic devices [300](#)
- MOS device size (W, L) mismatch [301](#)
- MOS power connection mismatch [304](#)
- MOS reversibility error [303](#)
- MOS substrate connection mismatch [303](#)
- reduced layout parallel MOS devices [304](#)
- types in report [295](#)
- unmatched layout devices [300](#)
- unmatched schematic devices [300](#)
- flow [278](#)
- flow diagram [149](#)
- layout text guidelines [279](#)
- overview [275](#)
- report [676](#), [678](#)
- report format [287](#)
- rules file [279](#)
- sample report [289](#)
- sections [287](#)
- short report [676](#)
- texting [827](#)
- W/L calculations [328](#)
- within PRE [472](#)
- LVSCCHK command [1103](#)
- SMASH qualifier [1113](#)
- LVSCMP reserved break point [900](#), [912](#)
- LVSNET reserved break point [900](#), [912](#)
- LVSNET summary, LVS report [287](#)
- LVS-OPTION command [676](#)
- LVS RPT-ONLY command [678](#)
- M**
- .MACRO command [200](#)
- macro execution [572](#)
- macro, definition [1258](#)
- MAG-BEF-GRID command [684](#)
- magnification
  - absolute [684](#)
  - RELOCATE command [1166](#)
- MAGNIFY-IN command [685](#)
- MAGNIFY-OUT command [686](#)
- MAP-TEXT-FILE command [687](#)
- MASK command [424](#)
- mask order [838](#), [861](#)
- mask out feedback loops [441](#)
- mask, definition [1259](#)
- masked nodes, resetting [411](#)
- matched devices
  - definition [1259](#)
- matched devices, LVS report [293](#)
- matched node identification, LVS report [293](#)
- MATCH-P-G-BY-TYP command [688](#)
- \*.MEGA command [174](#)
- MEMORY-CORE command [689](#)
- merged data [605](#)
- metal reflection design rules [89](#)
- metal reflection, causes illustrated [88](#)
- M-Factor tolerance commands [681](#)
- microns
  - converting databases [557](#)
  - definition [1259](#)
  - resolution [760](#)
  - specifying units [766](#)
- mils
  - converting databases [557](#)
  - definition [1259](#)
  - resolution [760](#)
- minimum step-size (grid) [760](#)
- MINI-SUMMARY command [692](#)
- minus sign in rules files [43](#)
- MIRROR command [693](#)
- MODEL command [694](#)
- model, definition [1259](#)
- modes, specifying [37](#), [545](#)
- module.exe file [792](#)
- modules
  - APPLEOUT [63](#)
  - GDS2OUT [63](#)
  - PREINQUERY [63](#)
- MOS [143](#)
  - CDL transistor card [328](#)
  - counting elements [968](#)
  - defining devices [983](#)
  - devices [1120](#)

- elements [962](#)
- extracting devices [871](#)
- MOS subcommand (LPEPRO) [429](#), [434](#)
- MOS\_EXTENDED\_REP command [247](#), [265](#)
- MOS-CAP command [695](#)
- MOSCAP, defined [287](#)
- MOSCAP-AREA option
  - LVSCHK command [1087](#), [1109](#)
- MOSFETs, describing in CDL [212](#)
- MOS-L-TOLER command [681](#)
- MOS-W-TOLER command [681](#)
- mount point [703](#)
- MULDELW calculations [329](#)
- MULDELW option
  - LPECHK command [1089](#)
  - LVSCHK command [1112](#)
- MULTICEL.TAB files [54](#), [116](#)
- MULTICPU command [697](#)
- MULTI-GDS2-IN command [698](#)
- MULTILAB command [1117](#)
- MULTILAB error output [145](#)
- multilevel hierarchical mode DRC [116](#)
  - error reports [118](#)
  - example [116](#)
- multilevel mode, specifying [547](#)
- multiple output devices [467](#)
- multiple stages module [466](#)
- MULTIPLE-DISK command [702](#)
- MustJoin
  - text [619](#)
  - wire types [346](#)

## N

- NDCOUNT command [1120](#)
- NEIGHBOR command [1123](#)
- NET in DSPF format [498](#)
- net, definition [1259](#)
- netlists
  - CDL [156](#)
  - definition [1259](#)
  - Hcell pin names [340](#)
  - include files in SPICE [196](#)
  - LVS files [338](#)
  - preparing for LOGLVS [151](#)
  - transistor-level [151](#)
- network, retrieving information [414](#)
- network.sav files [412](#)
- /NEXT command, PDRACULA [58](#)

- NMOS inverter device recognition [306](#)
- NMOS NAND gate device recognition [308](#)
- NMOS NOR gate device recognition [307](#)
- NO\_WARNING command [248](#)
- node names [839](#)
  - delimiter, specifying [577](#), [578](#)
  - equating to node numbers [168](#), [235](#)
  - expanding SPICE element names [150](#)
  - global [195](#)
  - ground [217](#)
  - illegal characters [217](#)
  - in CDL macros [213](#)
  - LPEPRO output [433](#)
  - schematic [559](#)
- node numbers
  - finding [560](#)
  - in CDL macros [213](#)
  - in SPICE files [189](#), [217](#)
  - internal [443](#)
  - LPEPRO output [434](#)
  - schematic [559](#)
- NODE-FILE command [1124](#)
- nodes
  - conductive terminals [478](#), [479](#)
  - connections [554](#), [555](#)
  - counting [1120](#)
  - extraction commands [870](#)
  - floating, checking for [31](#)
  - labeling information [1212](#)
  - multiple connections [1206](#)
  - nodal capacitance [437](#)
  - nodal information, RCONNECT-LAYER sequence [476](#)
  - same label [1171](#)
  - shorted [1117](#)
  - specifying sets for selective LPE [1124](#), [1126](#)
  - subnodes (split resistor terminals) [476](#)
  - texting power/ground [826](#)
- NODE-SELECT command [1126](#)
- non-orthogonal lines [598](#)
- \*.NONSWAP command [175](#)
- \*.NOPIN command [177](#)
- \*.NOSUB\_M command [178](#)
- NO-SUB-TERM command [704](#)
- NOT command [1128](#)
- notch [1018](#)
  - definition [1259](#)
- NOT-HCELL command [705](#)
- NRX-EXTRACT command [706](#)
- NRX-SAVE command [713](#)



numbers in rules files [43](#)  
NVER command [249](#)

## O

OCTBIAS command [1132](#)  
ONCE option, CELL-ERROR-REP  
    command [119](#)  
open abort [516](#)  
open circuits  
    definition [1259](#)  
open circuits, checking for [31](#)  
opens in .ERC file [133](#)  
Operation block  
    creating a rules file [42](#)  
    DRC example [83](#)  
    ERC sample [137](#)  
    LPE example [394](#)  
operation block, definition [1259](#)  
OPTIMIZE-RULE command [715](#)  
options, syntax [43](#)  
OR command [1134](#)  
ORIG-ALL option, CELL-ERROR-REP  
    command [119](#)  
ORIG-DRC option, CELL-ERROR-REP  
    command [119](#)  
orthogonal, definition [1259](#)  
OUTDISK command [717](#)  
OUTLIB command [718](#)  
output  
    database [843](#)  
    error cells [718](#)  
    files [63](#)  
    graphic file [1207](#)  
    layer [769](#)  
    signals [559](#)  
    tape [766, 801](#)  
output cell, definition [1259](#)  
OUTPUT-ONE-LAYER command [719](#)  
OUTSIDE option, SELECT  
    command [1177](#)  
outside polygons [991](#)  
outside segments [1019](#)  
OVERLAP command [1137](#)  
oversize [770](#)  
OVPR-TOLERANCE command [720](#)

## P

PAD element [1120](#)  
pad layers  
    creating [480](#)  
    error conditions [481](#)  
PAD windows  
    connecting [962](#)  
    counting [968](#)  
PAD-LAYER command [856](#)  
pads  
    assigning global names [195, 241](#)  
    character [828](#)  
    definition [1260](#)  
    texting [828](#)  
    types [831](#)  
parallel devices, calculations for LVS [328](#)  
PARALLEL-FILE command [721](#)  
.PARAM command [197](#)  
PARAMETER CAP command [1139](#)  
parameter extraction, definition [1260](#)  
PARAMETER RES command [1141](#)  
PARAMETER subcommand  
    (LPEPRO) [431, 438](#)  
parameter, definition [1260](#)  
parameters  
    calculating [417](#)  
    diode [362](#)  
    extraction [359](#)  
    filename [1077](#)  
    HSPIICE support in LOGLVS [197](#)  
    parametric information [406](#)  
    probing circuit [356](#)  
    resetting network information [411](#)  
    scanning [433](#)  
PARASITIC CAP command [1143](#)  
parasitic devices, definition [1260](#)  
PARASITIC DIO command [1146](#)  
PARASITIC RES command [1150](#)  
Parasitic Resistance Extraction. See PRE  
parasitics  
    capacitance [884](#)  
    capacitance type [1144](#)  
    capacitor names [742](#)  
    creating capacitors [476](#)  
    creating devices [474](#)  
    definition [1260](#)  
    device names [742](#)  
    DIO type [1146](#)  
    diode parameters [362](#)

## Dracula Reference

---

- diodes [742](#)
  - element [742](#)
  - extraction procedure [472](#)
  - information [402](#)
  - junction diode [1146](#)
  - PRES and PTRM [485](#)
  - redistributing capacitance [483](#)
  - resistance extraction [490](#)
    - example [485](#), [492](#)
    - overview [471](#)
  - resistance of conduction layers [476](#)
  - resistors [182](#)
    - component regions [472](#)
    - conduction layers [484](#)
    - CUT-TERM command [478](#)
    - devices [476](#), [1150](#)
    - prefixes [742](#)
    - SPICE [1093](#)
    - terminal regions [472](#)
  - RSPF file [1170](#)
  - PAR-COMPARE command [723](#)
  - PAROUTNAM command [725](#)
  - parse commands, LOGLVS [155](#)
  - PARSET [731](#)
  - PARSET command [731](#)
  - PARSET keywords [375](#)
  - passivation layers [480](#)
  - path check [146](#), [1046](#)
  - PATH command [421](#)
  - PATH subcommand (LPEPRO) [427](#)
  - PATHCHK command [1152](#)
  - paths [407](#)
    - acute angles [518](#)
    - analysis [408](#)
    - modifying geometries [518](#)
    - reporting off-grid coordinates [603](#)
    - selective LPE [671](#)
  - PATH-W-OFFGRID command [735](#)
  - PATH-W-ROUND command [737](#)
  - PDRACULA
    - commands [45](#) to [62](#)
    - relationship to Dracula applications [32](#)
    - relationship to Dracula applications, illustrated [33](#)
    - running [47](#)
  - PDRACULA, definition [1260](#)
  - PDW device recognition [310](#)
  - pdw, definition [1260](#)
  - PDWI device recognition [312](#)
  - PG/E II
    - commands
    - RELOCATE [1166](#)
  - \*.PIN command [179](#)
  - pincapfile [739](#)
  - PINCAP-FILE command [739](#)
  - \*.PININFO command [180](#)
  - pins
    - capacitance, specifying [739](#)
    - definition [1260](#)
    - delimiter for names [581](#)
  - PIN-TEXT-LAYER command [859](#)
  - PLENGTH command [1156](#)
  - plus sign in rules files [43](#)
  - P-N junction diode [977](#)
  - POINT-CONNECT command [740](#)
  - polygons
    - coinciding [1064](#)
    - coordinates, reporting
      - acute angles [597](#)
      - nonorthogonal lines [598](#)
    - off-grid [601](#)
    - self-intersecting [604](#)
    - touching [606](#)
  - enclosed [1017](#)
  - enclosing [991](#)
  - from expanding paths, checking [548](#)
  - overlapping [992](#), [1060](#), [1137](#), [1227](#)
  - oversizing [581](#)
  - reentrant [605](#)
  - self-intersecting [605](#)
  - undersizing [581](#)
- POSATT module [133](#)
- POSTENC command [1158](#)
- power name in SPICE files [217](#)
- power/ground nodes
  - GPATHCHK command [146](#)
  - ground text [621](#)
  - path check [1046](#)
  - power text [741](#)
  - shorts [609](#)
  - texting [826](#)
- POWER-NODE command [643](#), [741](#)
- PRE
  - composite mode [495](#)
  - definition [32](#)
  - LVS check [472](#)
  - operation flow [472](#)
  - sample command file [485](#)
- PRE, definition [1260](#)
- PRECISION command [250](#)
- PREFIX-PARASITIC command [742](#)
- PREINQUERY module [63](#)

## Dracula Reference

---

PREINTERACTIVE command [743](#)  
preprocessor. See PDRACULA  
pre-set filtering [592](#)  
primary cell  
    definition [1260](#)  
    library [658](#)  
    processing text [807](#)  
PRIMARY command [745](#)  
primitive gates [443](#)  
primitive, definition [1260](#)  
PRINT command [251](#)  
    LOGLVS [251](#)  
    LPEPRO [426](#)  
/PRINT command, PDRACULA [59](#)  
PRINTF file [746](#)  
PRINTF.LVS files [337](#)  
PRINTF.SUM files [118](#), [122](#), [125](#)  
PRINTFILE command [746](#)  
printfile.lvs files [280](#)  
printfile.sum file [516](#), [597](#)  
PRINTLINE option  
    LPECHK command [1089](#)  
    LVSCHK command [1109](#)  
PROBE command [1159](#)  
PROB-GEOM-LIST command [747](#)  
problem geometries [746](#)  
process geometries, illustrated [89](#)  
process technologies [31](#)  
program directory [750](#)  
PROGRAM-DIR command [750](#)  
pseudo contact layer, definition [1261](#)  
pseudo gates [327](#)  
    definition [1261](#)  
PULL-DOWN command [751](#)  
PULL-UP command [752](#)  
PUP device recognition [309](#)  
pup, definition [1261](#)  
PUPI and SDWI device recognition [313](#),  
    [316](#)  
PUPI device recognition [312](#)

## Q

question mark (?), use in text [827](#)  
queuing jobs [1250](#)

## R

R option, SELECT command [1182](#)

RAM-CELL command [753](#)  
RAM-CORE command [754](#)  
RANGE option  
    AREA command [878](#)  
    PLENGTH command [1156](#)  
    WIDTH command [1219](#)  
RCONNECT command, PRE [1161](#)  
RCONNECT-LAYER command [861](#)  
RECTCHK command [1163](#)  
REDUCE modules [325](#)  
REDUCE summary, LVS report [287](#)  
REDUCER command [756](#)  
RELOCATE command [1166](#)  
REMOVE-SCH-OPT [582](#)  
repeat summary, LVS report [288](#)  
reports, generating LPEPRO [408](#)  
REP-UNM-INST-PG command [758](#)  
RERUN-FILTER [759](#)  
reserved break points [900](#), [912](#)  
reserved parameter keywords [373](#)  
reserved parameter values [732](#)  
RESET command  
    LOGLVS [252](#)  
    LPEPRO [411](#)  
\*.RESI command [182](#)  
RESISTOR command [253](#)  
resistors  
    body layer [474](#)  
    calculating values [364](#)  
    contact [1030](#)  
    describing in CDL [208](#)  
    devices [986](#)  
    extracting parasitics [485](#), [1030](#)  
    parameters [1141](#)  
    parasitic [742](#), [1030](#)  
    poly-silicon [365](#)  
    problem [367](#)  
    resistance layer junction regions [482](#)  
    shorting [477](#)  
    simulating parasitic [182](#)  
    smashing [779](#), [785](#), [1030](#)  
    specifying width and length [1113](#)  
    terminal layers [474](#), [476](#), [861](#)  
    terminal regions [472](#), [474](#), [479](#)  
    terminal-drawn [366](#)  
    type [986](#), [1141](#)  
    undefined device [366](#)  
RESLPERCENT option  
    LPECHK command [1089](#)  
    LVSCHK command [1113](#)  
RESOLUTION command [760](#)

## Dracula Reference

---

\*.RESSIZE command [184](#)  
RESSIZE option  
    LPECHK command [1088](#)  
    LVSCHK command [1111](#)  
/RESTART command, PDRACULA [60](#)  
\*.RESVAL command [185](#)  
RESVAL option  
    LPECHK command [1088](#)  
    LVSCHK command [1111](#)  
RESWPERCENT option  
    LPECHK command [1089](#)  
    LVSCHK command [1113](#)  
\*.REVERSE command [186](#)  
reverse mask operation [395](#)  
REV-SCALE command [765](#)  
REV-SCALEIN command [765](#)  
REV-SCALEOUT command [766](#)  
rise delay time [416](#)  
RLENGTH command [1168](#)  
ROTATION command [763](#)  
ROUND-OFF-TOLER command [764](#)  
RSPF  
    Dracula and CDC flow [506](#)  
    example circuit [502](#)  
    example with multiple drivers and loads [503](#)  
    format [501](#)  
    specifying pin capacitance [498](#)  
RSPFSELECT command [1170](#)  
rules files  
    blocks [41](#)  
    command line syntax [43](#)  
    compiling [45](#)  
    definition [1261](#)  
    example [44](#)  
    special characters [43](#)

## S

SAMELAB command [1171](#)  
SAVE command  
    LOGLVS [254](#)  
    LPEPRO [412](#)  
SCALE command [765](#)  
scale factors [685](#)  
scale, definition [1261](#)  
scaled cell [684](#)  
SCALEIN command [765](#)  
SCALEOUT command [766](#)  
schematic to schematic, comparing [285](#)

Schematic Versus Schematic. See SVS, LVS  
schematic, definition [1261](#)  
schematics  
    checking [355](#)  
    LVS report [293](#)  
    LVSCHK command [1103](#)  
    netlist conversion [147](#)  
    revising data [464](#)  
SCHMIN reserved break point [900](#), [912](#)  
SCONNECT command [1173](#)  
SDW device recognition [311](#)  
sdw, definition [1261](#)  
SDWI device recognition [312](#)  
sdwi, definition [1262](#)  
segment count [632](#)  
SELECT command [1176](#)  
selective LPE  
    LPE-QUERY command [671](#)  
    NODE-FILE command [1124](#)  
    NODE-SELECT command [1126](#)  
    SELECT-MODE command [768](#)  
SELECT-MODE command [768](#)  
SELGT/SELGE option, WIDTH command [1219](#)  
SELLT/SELLE option, WIDTH command [1219](#)  
SELRA option, WIDTH command [1220](#)  
semicolon (;) in rules files [43](#)  
SET FANIN command [255](#)  
SET FANOUT command [256](#)  
sheet resistance  
    definition [1261](#)  
    parameters [891](#), [1030](#)  
    values [1141](#)  
short circuits, checking for [31](#)  
SHORTBOX.DAT files [145](#)  
shorts  
    aborting [516](#)  
    detecting [609](#)  
SHRINK command [1190](#)  
silicon-gate CMOS technology [391](#)  
simple parallel pulldown (PDW)  
    recognition [310](#)  
simple parallel pull-up (PUP) device  
    recognition [309](#)  
simple series pulldown (SDW) device  
    recognition [311](#)  
simple series pull-up (SUP) device  
    recognition [310](#)  
simulators, supported [33](#)

## Dracula Reference

- SIZE command [1192](#)
- size mismatch [678](#)
- SIZE-MAX-VERTEX command [769](#)
- SIZE-MIN-WIDTH command [770](#)
- SIZENOT command [772](#)
- sizing commands [870](#)
- sizing layers [789](#)
- sizing operation commands, described [79](#)
- sizing, fast [580](#)
- SKILL-LOADER command [773](#)
- slash (/), use with PDRACULA commands [47](#)
- smash, definition [1261](#)
- SMASH-CAP-TYP command [779](#)
- SMASH-FLOAT-RES command [780](#)
- smashing resistors [779](#), [785](#)
- SMASH-MOS-TYP command [781](#)
- SMASH-RES-TYP command [785](#)
- SMOOTH-SIZE command [786](#)
- SNAP command [1205](#)
- soft connection [517](#)
- SOFTCHK command [1206](#)
- SOFTCHK problems [517](#)
- S-OPTION-CLASS command [788](#)
- source/drain
  - conductor layer [983](#)
  - GPATHCHK command [146](#), [1046](#)
  - parameters [881](#)
- spacing checks [115](#)
- spacing commands, described [80](#)
- spacing operations [870](#)
- SPATH command [420](#)
- SPATH subcommand (LPEPRO) [427](#)
- special devices [973](#)
- special elements, LOGLVS [202](#)
- SPF
  - DSPF [496](#)
  - overview [495](#)
  - RSPF [501](#)
- SPF-MAP-FILE command [789](#)
- SPFSELECT command [1208](#)
- SPICE
  - compiling netlists [274](#)
  - creating files [406](#)
  - displaying file [442](#)
  - extracting parameters [358](#)
  - file example [217](#)
  - file generated by HPRE [495](#)
  - flexible LPE [372](#)
  - generating a logic network [443](#)
  - generator [407](#), [439](#)
  - initial correspondence node names [150](#)
  - LPE batch mode [356](#)
  - LPE examples [392](#)
  - LPE listing [694](#)
  - LPESELECT command [477](#)
  - model names [694](#)
  - netlists [268](#)
  - output file format [793](#)
  - preparing files [216](#)
  - running hierarchical LOGLVS [267](#)
  - sample input file (PRE) [492](#)
  - sample output file with parasitic resistors [493](#)
  - subcircuit definitions [151](#)
  - unit processing [810](#)
- \*.SPICE command [189](#)
- SPICE, definition [1261](#)
- SPICE.CEL file [403](#)
- SPICE.DAT file [403](#)
- split conduction layers [479](#)
- split gates, ordering and merging [537](#)
- STAMP command [1212](#)
- Stamping, definition [1261](#)
- stamping, definition [870](#)
- Standard Parasitic File. See SPF
- START command [422](#)
- START-MACRO command [791](#)
- /STATUS command, PDRACULA [61](#)
- STATUS-COMMAND command [792](#)
- step-size [760](#)
- STORE command
  - LOGLVS [258](#)
  - LPEPRO [426](#)
- /STORE command, PDRACULA [62](#)
- STRIP option, CUT command [915](#), [921](#), [928](#)
- subcircuits
  - beginning definitions [200](#)
  - defining [204](#)
  - description data file [147](#)
  - missing subcircuit name [793](#)
- .SUBCKT command [200](#)
- SUBCKT-NAME command [793](#)
- SUBNODE-DELIM command, PRE [794](#)
- subnodes [794](#), [1093](#)
- SUBSTRATE command [863](#)
- substrate, definition [1262](#)
- subtype mismatch [678](#)
- SUBTYPE-CSENS command [797](#)
- SUMMARY command [259](#)
- summary file [663](#)

## Dracula Reference

---

SUMMARY-ONLY command [798](#)  
SUP device recognition [310](#)  
sup, definition [1262](#)  
SUPI and PDWI device recognition [315](#)  
SUPI and SDWI device recognition [319](#)  
SUPI device recognition [312](#)  
supi, definition [1262](#)  
SVS  
    definition [31](#)  
    HCELL-COLUMN-1 command [625](#)  
    hierarchical mode [286](#)  
    overview [285](#)  
    rules files [286](#)  
    SVS-LAYOUT command [799](#)  
    SVS-SCHEMATIC command [800](#)  
SVS-LAYOUT command [799](#)  
SVS-SCHEMATIC command [800](#)  
.SWAP command [202](#)  
swap space, LOGLVS [253](#), [261](#)  
syntax  
    ELEMENT CAP violation [838](#)  
    errors in rules file [45](#)  
SYSIN command [803](#)  
SYSOUT command [801](#)  
SYSTEM command [803](#)

## T

tagging layer [946](#)  
TEMPORARY-LAYER command [865](#)  
terminal layers  
    ELEMENT BJT [971](#)  
    ELEMENT BOX [973](#)  
    ELEMENT CAP [975](#)  
    ELEMENT RES [986](#)  
    FRINGE CAP [1041](#)  
    PARASITIC CAP [1144](#)  
    resistors [474](#)  
terminal regions, creating [479](#)  
terminal, definition [1262](#)  
terminal-drawn resistors [366](#)  
text  
    adding to planes [342](#)  
    attaching [829](#), [966](#)  
        example [826](#)  
    characters to use [827](#)  
    coordinate file [827](#)  
    delimiter [827](#)  
    discarded [1171](#)  
    guidelines [279](#)

Hcells [348](#)  
initial correspondence nodes [277](#)  
input from graphics system [830](#)  
label [1159](#)  
labeled nodes [826](#)  
layers [828](#), [848](#), [859](#)  
LPE [356](#)  
name [867](#)  
power and ground [831](#)  
redefining sequence [867](#)  
rules for applying text [828](#)  
truncated [827](#)  
type [827](#), [852](#)  
text editors, UNIX [40](#)  
TEXT-LEVEL command [806](#)  
TEXT-PRI-ONLY command [807](#)  
TEXTSEQUENCE command [867](#)  
timing analysis [407](#)  
timing verification [355](#)  
TNAME command [260](#)  
TNAMES-CSEN command [808](#)  
toggle, definition [1262](#)  
TOUCH option, SELECT command [1180](#)  
trace operation [277](#)  
TRANSISTOR command [261](#)  
transistor-level network data [148](#)  
transistors  
    checking width and length [434](#)  
    computing width and length [328](#)  
    presetting numbers [261](#)  
    types [391](#)  
    width and length [358](#)  
translator utility [1239](#)  
trapezoids  
    definition [1262](#)  
    ENC command [992](#)  
    EXT command [1019](#)  
    INT command [1064](#)  
treefiles [801](#)  
    definition [1262](#)  
TRIANGLE command [1215](#)  
T-shaped junction resistances [482](#)  
TYPE command  
    LOGLVS [262](#)  
    LPEPRO [426](#)

## U

“unconnected pins in cell” warning [579](#)  
undefined elements, LVS [324](#)

## Dracula Reference

---

UNIT command [810](#), [1141](#)  
units, step-size [760](#)  
UNIX commands for installation [1243](#)  
UNIX text editors [40](#)  
UNMASK command [425](#)  
unmatched devices, LVS report [293](#)  
unmatched, definition [1262](#)  
UNREACH subcommand (LPEPRO) [433](#)  
\*.UNSPEC command [190](#)  
UNSPEC-LAY-PARA command [811](#)  
\*.UNSPEC-MOS command [192](#)  
UNSPEC-PARA command [812](#)  
UNSPEC-SCH-PARA command [814](#)  
upper\_layer option, SOFTCHK  
command [1206](#)  
user-defined devices [973](#)

## V

value size units [575](#)  
verification process [39](#)  
verifying designs, approaches  
described [30](#)  
Verilog  
initial correspondence node names [150](#)  
preparing for LOGLVS [219](#)  
using the LINK command [246](#)  
VERTEX option, SELECT command [1183](#)  
vertices  
maximum [769](#)  
offgrid [601](#), [603](#)  
self-intersecting [604](#)  
touching [606](#)  
virtual connections [826](#)  
Virtual wire, definition [1262](#)  
virtual wires [346](#)  
connecting [555](#)  
deleting [565](#)

## W

W/L calculation, parallel devices [328](#)  
W/L-PERCENT option  
LPECHK command [1089](#)  
LVSCHK command [1112](#)  
warning messages, PDRACULA [45](#), [64](#)  
warning messages, suppressing in  
LOGLVS [248](#)  
Weff MULDELW calculation [329](#)

WEFFECT option  
LPECHK command [1088](#)  
LVSCHK command [1109](#)  
WEFFECT or CORNER calculations [360](#),  
[1090](#)  
WIDTH command [1216](#)  
wildcard characters [829](#)  
WINDEL command [815](#)  
WINDOW command [817](#)  
WINDOW-CUT command [819](#)  
windows, coordinates [817](#)  
wired-OR delay values [416](#)  
wires  
definition [1262](#)  
specifying virtual connections [826](#)  
types [346](#)  
virtual [828](#), [864](#)  
WORK-DIR command [820](#)  
working directory [821](#)  
working window, definition [1262](#)  
WPERCENT option  
LVSCHK command [1109](#)

## X

XBOX command [1225](#)  
XCELL command [1227](#)  
XDEVICE command [1229](#)  
XOR command [1231](#)  
XVIA command [1234](#)