

## Symbols

- !
  - compared to ‘==0’ 47
  - logical negation operator 40, 47
- !=
  - logical inequality operator 40, 46
- !==
  - case inequality operator 41, 46
- ""
  - null string 57
- \$ 362, 365, 429
- \$async\$and\$array 303
- \$async\$and\$plane 303
- \$async\$nand\$array 303
- \$async\$nand\$plane 303
- \$async\$nor\$array 303
- \$async\$nor\$plane 303
- \$async\$or\$array 303
- \$async\$or\$plane 303
- \$bitstoreal 190, 311
- \$countdrivers 788–789
- \$display 279–286
  - compared to \$monitor 286, 287
  - compared to \$write 279
  - escape sequences 279
  - format specifications 280–282
  - size of displayed data 282–283
- \$displayb 279
- \$displayh 279
- \$displayo 279
- \$dist\_chi\_square 313
- \$dist\_erlang 313
- \$dist\_exponential 313
- \$dist\_normal 313
- \$dist\_poisson 313
- \$dist\_t 313
- \$dist\_uniform 313
- \$dumpall 328, 337
- \$dumpfile 325
- \$dumpflush 329
- \$dumplimit 328
- \$dumploff 328, 337, 338, 344
- \$dumpon 328
- \$dumpports 340
  - rules to use 341
- \$dumpportsall 342
- \$dumpportsflush 343
- \$dumpportslimit 342
- \$dumpportsoff 341
- \$dumpportson 341
- \$dumpvars 326
- \$fclose 287–290
- \$fdisplay 289–290
- \$fdisplayb 289
- \$fdisplayf 289
- \$fdisplayh 289
- \$ferror 291, 295
- \$fflush 295
- \$fgetc 291
- \$finish 302, 594
- \$fmonitor 289–290
- \$fmonitorb 289
- \$fmonitorf 289
- \$fmonitorh 289
- \$fopen 287–290
- \$fopen() 584
- \$fscanf 291
- \$fseek 291, 295
- \$fstrobe 289–290
- \$fstrobb 289
- \$fstrobbef 289
- \$fstrobeh 289
- \$ftell 295
- \$fullskew 253
- \$fwrite 289–290
- \$fwriteb 289
- \$fwritef 289
- \$fwriteh 289
- \$getpattern 789
- \$hold 242
- \$incsave 792–793
- \$input 790
- \$itor 311
- \$key 790
- \$list 791
- \$log 791
- \$monitor 287
  - compared to \$display 287
- \$monitorb 287
- \$monitorh 287
- \$monitro 287
- \$monitoroff 287

`$monitoron` 287  
`$nochange` 257  
`$nokey` 790  
`$nolog` 791  
`$period` 256  
`$printtimescale` 298–299  
`$q_add` 308  
`$q_exam` 308  
`$q_full` 308  
`$q_initialize` 307  
`$q_remove` 308  
`$random` 312  
`$readmemb` 296–297  
    and loading logic array personality 304  
`$readmemh` 296–297  
    and loading logic array personality 304  
`$realttime` 310  
`$realtobits` 190, 311  
`$recovery` 246  
`$recrem` 247  
`$removal` 245  
`$reset` 791  
`$reset_count` 791  
`$reset_value` 791  
`$restart` 792–793  
`$rewind` 295  
`$rtoi` 311  
`$save` 792–793  
`$scale` 793  
`$scope` 793  
`$sdf_annotate` system task 297  
`$setup` 242  
`$setuphold` 243  
`$sformat` 290  
`$showscopes` 793  
`$showvars` 794  
`$signed` 62  
`$skew` 250  
`$sreadmemb` 794  
`$sreadmemh` 794  
`$sscanf` 291  
`$stime` 310  
`$stop` 302, 595  
`$strobe` 286  
    compared to `$display` 286  
`$strobeb` 286  
`$strobeh` 286  
`$strobo` 286  
`$write` 290  
`$writeb` 290  
`$writeh` 290  
`$writeo` 290  
`$sync$and$array` 303  
`$sync$and$plane` 303  
`$sync$nand$array` 303  
`$sync$nand$plane` 303  
`$sync$nor$array` 303  
`$sync$nor$plane` 303  
`$sync$or$array` 303  
`$sync$or$plane` 303  
`$test$plusargs` 322  
`$time` 32, 309–310  
`$timeformat` 299–302  
`$timeskew` 251  
`$ungetc` 291  
`$unsigned` 62  
`$vcdclose` 343  
`$width` 255–256  
`$write` 279–286  
    compared to `$display` 279  
    escape sequences 279  
    format specifications 280–282  
    size of displayed data 282–283  
`$writeb` 279  
`$writeh` 279  
`$writeo` 279  
`%`  
    in format specifications 279, 283  
    modulus operator 40  
`&`  
    bit-wise AND operator 41  
    reduction AND operator 41  
`&&`  
    logical AND operator 40, 47  
`(?)`  
    in state table 110  
`(01)`  
    in state table 110  
`(0x)`  
    in state table 110  
`(1x)`  
    in state table 110

(vw)	\
in state table 110	backslash character 12
(x1)	for escape sequences in strings 279
in state table 110	\"
*	as " character 12
arithmetic multiplication operator 40	\\ddd
in state table 110	specify character as octal digits 12
** 44	\\t
”	tab character 12
in null expressions 279	^
/	bit-wise exclusive OR operator 41
arithmetic division operator 40	reduction XOR operator 41
<	^~
relational less-than operator 40, 46	bit-wise equivalence operator 41
<<	reduction XNOR operator 41
left shift operator 49	,
logical left shift operator 41	in compiler directives 351
<<<	\\celldefine 351, 522
arithmetic left shift operator 41	\\default_decay_time 795
<=	\\default_nettype 351, 352
relational less-than-or-equal operator 40, 46	\\default_trireg_strength 795
=	\\define 352
in assignment statement 69	and text macro substitutions 354
==	\\delay_mode_distributed 796
logical equality operator 40, 46	\\delay_mode_path 796
===	\\delay_mode_unit 796
case equality operator 41, 46	\\delay_mode_zero 796
>	\\else 354
relational greater-than operator 40, 46	\\elsif 354
>=	\\endcelldefine 351, 522
relational greater-than-or-equal operator 40, 46	\\endif 355
>>	\\ifdef 354
logical right shift operator 41	\\ifndef 354
right shift operator 49	\\include 358
>>>	\\nounconnected_drive 361
arithmetic right shift operator 41	\\resetall 358
?	\\timescale 359, 611, 612
equivalent to z in literal number values 8, 133	\\unconnected_drive 361
in state table 110, 113	\\undef 354
?:	replication operator 40
conditional operator 41	{}
@	concatenation operator 40, 51
for addressing memory 296	
	bit-wise inclusive OR operator 41
	reduction OR operator 41

||  
     logical OR operator 40, 47  
 ~  
     bit-wise negation operator 41  
 ~&  
     reduction NAND operator 41  
 ~^  
     bit-wise equivalence operator 41  
     reduction XNOR operator 41  
 ~|  
     reduction NOR operator 41  
 Numerics  
 0  
     for minimizing bit lengths of expressions  
         283  
     in state table 110  
     logic 0 284  
     logic zero 20  
 01 transition 113  
 1  
     in state table 110  
     logic 1 284  
     logic one 20  
 A  
 acc\_append\_delays() 389, 405, 415  
 acc\_append\_pulsere() 409, 415  
 acc\_close() 370, 401, 411, 415, 515  
 acc\_collect() 412, 479, 525, 548  
 acc\_compare\_handles() 414  
 acc\_configure() 415  
 acc\_count() 424, 525, 548  
 acc\_error\_flag 387, 465, 467, 468  
 acc\_fetch\_argc() 425  
 acc\_fetch\_argv() 426  
 acc\_fetch\_attribute() 415, 428  
 acc\_fetch\_attribute\_int() 415, 432  
 acc\_fetch\_attribute\_str() 415, 433  
 acc\_fetch\_defname() 434  
 acc\_fetch\_delay\_mode() 435  
 acc\_fetch\_delays() 389, 415, 437  
 acc\_fetch\_direction() 441  
 acc\_fetch\_edge() 442  
 acc\_fetch\_fullname() 415, 444  
 acc\_fetch\_fulltype() 446  
 acc\_fetch\_index() 449  
 acc\_fetch\_itfarg() 465  
 acc\_fetch\_itfarg\_int() 467  
 acc\_fetch\_itfarg\_str() 468  
 acc\_fetch\_location() 451  
 acc\_fetch\_name() 415, 453  
 acc\_fetch\_paramtype() 455  
 acc\_fetch\_paramval() 456  
 acc\_fetch\_polarity() 458  
 acc\_fetch\_precision() 459  
 acc\_fetch\_pulsere() 415, 460  
 acc\_fetch\_range() 463  
 acc\_fetch\_size() 464  
 acc\_fetch\_tfarg() 465  
 acc\_fetch\_tfarg\_int() 467  
 acc\_fetch\_tfarg\_str() 468  
 acc\_fetch\_timescale\_info() 469  
 acc\_fetch\_type() 471  
 acc\_fetch\_type\_str() 473  
 acc\_fetch\_value() 474  
 acc\_free() 412, 479  
 acc\_handle\_by\_name() 480  
 acc\_handle\_calling\_mod\_m() 482  
 acc\_handle\_condition() 483  
 acc\_handle\_conn() 484  
 acc\_handle\_datapath() 485  
 acc\_handle\_hiconn() 486  
 acc\_handle\_interactive\_scope() 488  
 acc\_handle\_itfarg() 512  
 acc\_handle\_loconn() 489  
 acc\_handle\_modpath() 415, 490  
 acc\_handle\_notifier() 492  
 acc\_handle\_object() 493  
 acc\_handle\_parent() 495  
 acc\_handle\_path() 496  
 acc\_handle\_pathin() 497  
 acc\_handle\_pathout() 498  
 acc\_handle\_port() 499  
 acc\_handle\_scope() 501  
 acc\_handle\_simulated\_net() 502  
 acc\_handle\_tchk() 415, 504  
 acc\_handle\_tchkarg1() 508  
 acc\_handle\_tchkarg2() 510  
 acc\_handle\_terminal() 511  
 acc\_handle\_tfarg() 512  
 acc\_handle\_tfirst() 514  
 acc\_initialize() 370, 401, 411, 415, 515  
 acc\_next() 516

- acc\_next\_bit() 520
- acc\_next\_cell() 522
- acc\_next\_cell\_load() 523
- acc\_next\_child() 525, 548
- acc\_next\_driver() 526
- acc\_next\_hiconn() 527
- acc\_next\_input() 529, 556
- acc\_next\_load() 523, 531
- acc\_next\_loconn() 533
- acc\_next\_modpath() 534
- acc\_next\_net() 535
- acc\_next\_output() 536, 556
- acc\_next\_parameter() 538
- acc\_next\_port() 539
- acc\_next\_portout() 541
- acc\_next\_primitive() 542
- acc\_next\_scope() 543
- acc\_next\_specparam() 544
- acc\_next\_tchk() 545
- acc\_next\_terminal() 547
- acc\_next\_topmod() 424, 525, 548
- acc\_object\_in\_typelist() 549
- acc\_object\_of\_type() 551
- acc\_product\_type() 553
- acc\_product\_version() 555
- acc\_release\_object() 556
- acc\_replace\_delays() 389, 415, 557
- acc\_replace\_pulsere() 415, 561
- acc\_reset\_buffer() 564
- acc\_set\_interactive\_scope() 565
- acc\_set\_pulsere() 566
- acc\_set\_scope() 415, 568
- acc\_set\_value() 570
- acc\_user.h file 370
- acc\_vcl\_add() 398, 575
- acc\_vcl\_delete() 398, 577
- acc\_version() 578
- accAndGate 386
- accAssignFlag 574
- accBinStrVal 475, 571
- accBitSelectPort 385
- accBufGate 386
- accBufif0Gate 386
- accBufif1Gate 386
- accCellInstance 384
- accCmosGate 386
- accCollapsedNet 551
- accCombPrim 386
- accConcatPort 385
- accConstant 384
- accDataPath 384
- accDeassignFlag 574
- accDecStrVal 475, 571
- accDefaultAttr0 415, 428
- accDelayModeDistrib 435
- accDelayModeMTM 435
- accDelayModeNone 435
- accDelayModePath 435
- accDelayModeUnit 435
- accDelayModeZero 435
- accDisplayErrors 388, 415
- accDisplayWarnings 388, 415
- accEdge01 442, 504
- accEdge0x 442, 504
- accEdge10 442, 504
- accEdge1x 442, 504
- accEdgex0 442, 504
- accEdgex1 442, 504
- accEnableArgs 415, 504, 568
- access routines
  - accessible objects 376
  - error handling 387
  - exception values 389
  - history 362
  - listed by category
    - fetch routines 371
    - handle routines 372
    - miscellaneous routines 375
    - modify routines 375
    - next routines 373
    - VCL routines 376
  - listed by functional groups
    - routines that operate on bits of a port 379
    - routines that operate on inter-module paths 380
    - routines that operate on module instances 378
    - routines that operate on module or data paths 379
    - routines that operate on module ports 378

routines that operate on named events  
     382  
 routines that operate on nets 381  
 routines that operate on parameters  
     383  
 routines that operate on primitive in-  
     stances 380  
 routines that operate on primitive ter-  
     minals 381  
 routines that operate on registers 382  
 routines that operate on task argu-  
     ments 384  
 routines that operate on timing checks  
     383  
 routines that operate on top-level  
     modules 380  
 routines that operate on variables 382  
     warning messages 388  
 accExpandedVector 551  
 accFaultSimulator 553  
 accForceFlag 574  
 accFunction 384  
 accHexStrVal 475, 571  
 accHold 387, 504  
 accInertialDelay 572  
 accInout 441  
 accInoutTerminal 387  
 accInput 441  
 accInputTerminal 387  
 accIntegerParam 385, 386  
 accIntegerVar 384  
 accIntermodPath 387  
 accIntVal 475, 571  
 accMapToMipd 415  
 accMinTypMaxDelays 391, 406, 415, 438,  
     558  
 accMixedIo 441  
 accModPath 384  
 accModPathHasIfnone 551  
 accModPathhasIfnone 483  
 accModule 384  
 accModuleInstance 384  
 accNamedBeginStat 387  
 accNamedEvent 384  
 accNamedForkStat 387  
 accNandGate 386  
     accNegative 458  
     accNedge 442, 504  
     accNet 385  
     accNetBit 385  
     accNmosGate 386  
     accNochange 387, 504  
     accNoDelay 572  
     accNoedge 442, 504  
     accNorGate 386  
     accNotGate 386  
     accNotif0Gate 386  
     accNotif1Gate 386  
     accOctStrVal 475, 571  
     accOperator 385  
     accOrGate 386  
     accOther 553  
     accOutput 441  
     accOutputTerminal 387  
     accParameter 385  
     accPartSelect 385  
     accPartSelectPort 385  
     accPathDelayCount 391, 415, 438, 460, 562  
     accPathDelimStr 415, 445, 454  
     accPathInput 385  
     accPathOutput 385  
     accPathTerminal 385  
     accPeriod 387, 504  
     accPmosGate 386  
     accPort 385  
     accPortBit 385  
     accPosedge 442, 504  
     accPositive 458  
     accPrimitive 386  
     accPulldownGate 386  
     accPullupGate 386  
     accPureTransportDelay 572  
     accRcmosGate 386  
     accRealParam 385, 386  
     accRealTime 573  
     accRealVal 475, 571  
     accRealVar 386  
     accRecovery 387, 504  
     accReg 386  
     accRegBit 386  
     accReleaseFlag 574  
     accRnmosGate 386

- accRpmosGate 386
- accRtranGate 386
- accRtranif0Gate 386
- accRtranif1Gate 386
- accScalar 551
- accScalarPort 385
- accScalarVal 475, 571
- accScope 551
- accSeqPrim 386
- accSetup 387, 504
- accSetuphold 387
- accSimTime 573
- accSimulator 553
- accSkew 387, 504
- accSpecparam 386
- accStatement 387
- accStringParam 385, 386
- accStringVal 475, 571
- accSupply0 385
- accSupply1 385
- accSystemFunction 387
- accSystemRealFunction 387
- accSystemTask 387
- accTask 387
- accTchk 387
- accTchkTerminal 387
- accTerminal 387
- accTime 573
- accTimeVar 387
- accTimingAnalyzer 553
- accToHiZDelay 391, 415
- accTopModule 384
- accTranGate 386
- accTranif0Gate 386
- accTranif1Gate 386
- accTransportDelay 572
- accTri 385
- accTri0 385
- accTri1 385
- accTriand 385
- accTrior 385
- accTrireg 385
- accUnexpandedVector 551
- accUnknown 458
- accurate simulation
  - requirements 262
- accUserFunction 387
- accUserRealFunction 387
- accUserTask 387
- accVector 551
- accVectorPort 385
- accVectorVal 475, 571
- accWand 385
- accWidth 387, 504
- accWire 385
- accWirePath 387
- accWor 385
- accXnorGate 386
- accXorGate 386
- addressing memory 296–297
- always
  - and activity flow 118
- ambiguous strength 90–101
- and gate 81–82
- arguments
  - data 367
  - paramvc 368
  - reason 367
  - system task/function 364, 512
- arithmetic operators 40, 44
  - 44
  - % 44
  - \* 44
  - \*\* 44
  - + 44
  - / 44
  - and unknown logic values 44
- arrays 33
  - element 33
  - format 304
  - index 33
  - word 33
- assign 67, 573
- assign procedural continuous assignment statement 125
- assignment
  - steps for evaluating 63
- assignment ??–73
  - continuous 69–73, 119
  - left hand side 69
  - of delays to module paths 223–225
  - procedural 119–127

- procedural versus continuous 119
  - right hand side 69
  - variable declaration 73
- assignments
  - scheduling implications 66
- associating PLI routines to a name 366
- asynchronous arrays 303–307
- attribute names 428
- attributes 14
- B
- b
  - binary number format 8
  - in state table 110
- backannotation 270
- backslash character 12
- base format
  - binary 8
  - decimal 8
  - hexadecimal 8
  - octal 8
- basic configuration elements 201
- begin-end block statement 128, 146
- behavioral modeling ??–151
- bidirectional pass gate 86
- binary display format 8
  - and high impedance state 283
  - and unknown logic value 283
- Binary operators 6
- binary operators 42
  - { } 51
  - precedence 42
- binding instances 200
- bit-select
  - of vector net or register 52
  - out of bounds 52, 53, 54
  - references of real numbers 32
- bit-wise operators 47–48
  - AND 41
  - equivalence 41
  - exclusive OR 41
  - inclusive OR 41
  - negation 41
- blank port connection 169
- block comment 6
- block statement 146–149
  - fork-join 146
  - naming of 148
  - parallel 146
  - sequential 146
  - start and finish times 148–149
  - timing for embedded blocks 148
- blocking assignment statement 67
  - process 67
- blocking assignments 119
- blocking procedural assignment 119
- buf gate 82–83
- bufif gate 83–84
- C
- calltf routines 366, 755
- capacitive networks 27–30
- capacitive state 26
- case
  - item expressions 132
- case equality operator 41
- case inequality operator 41
- case statement 130–133
  - compared to if-else-if statement 132
  - constant expression 133
  - with don't-care 133
- casex 133
- casez 133
- cbAfterDelay 751
- cbAssign 747
- cbAtStartOfSimTime 751
- cbDeassign 747
- cbDisable 747
- cbEndOfCompile 752
- cbEndOfRestart 752
- cbEndOfSave 752
- cbEndOfSimulation 752
- cbEnterInteractive 752
- cbError 752
- cbExitInteractive 752
- cbForce 747
- cbInteractiveScopeChange 752
- cbNextSimTime 751
- cbPLIError 752
- cbReadOnlySynch 751
- cbReadWriteSynch 751
- cbRelease 747
- cbSignal 752
- cbStartOfRestart 752

- cbStartOfSave 752
- cbStartOfSimulation 752
- cbStmnt 747
- cbTchkViolation 752
- cbUnresolvedSystf 752
- cbValueChange 747
- cell 200, 522, 523
  - multiple 203
- CELL declaration 270
  - DELAY 270
  - LABEL 270
  - TIMINGCHECK 270
- cell load 523
- characters
  - specified as octal digits 12
- charge decay 30, 105
- charge decay process 105
- charge decay time 105
  - delay specification 105
- charge storage
  - strength 24
- charge storage strength 89
- checktf routines 365
- classes of PLI routines
  - calltf 366, 755
  - checktf 365
  - compiletf 755
  - consumer 366
  - misctf 366
  - sizetf 365
- clause
  - cell 205
    - using 209
  - default 204
    - using 208
  - instance 204
    - using 209
  - liblist 205
  - use 206
- cmos 84, 86
- cmos gate 86–87
- collapsed net 502, 551
- combinational UDPs 107, 111
  - compared to level-sensitive sequential 112
  - input and output fields in state table 109
- combined signal strengths 89–101
- combined signal values 89–101
- command line considerations 207
- command line options 586
- comments 6
- compare
  - string operation 56
- Compiler directives 14
- compiletf routines 755
- concatenation
  - and unsized numbers 51
  - of names 194
  - of operands 52
  - operator 40, 51
  - string operation 56
- concurrency
  - of activity flow 118
- condition
  - deterministic 266
  - non-deterministic 266
- conditional compilation 354
- conditional expression 216
- conditional operator 41, 50–51
  - and ambiguous results 50
  - modeling three-state output busses 51
- conditional operator ?: 42
- conditional statement 127–130
- conditioned event 266–267
  - versus unconditioned event 267
- config 200
- configuration parameters 415
  - accDefaultAttr0 415, 428
  - accDisplayErrors 415
  - accDisplayWarnings 415
  - accEnableArgs 415
  - accMapToMipd 415
  - accMinTypMaxDelays 415
  - accPathDelayCount 415
  - accPathDelimStr 415
- configurations 200, 203
  - hierarchical 206
- conflicts 25, 26
- connecting ports
  - by name 189–190
  - by position with ordered list 188
  - rules 191–192

- connection
  - difference between full and parallel 221
  - full 220–222
  - parallel 220–222
- constant expression 40
- constant function 161
- constant numbers 6
- consumer routine 366, 398
- context-determined expression 59
- continuous assignment 67, 69–73
  - and connecting ports 191
  - and driving strength 89, 285
  - and net variables 119
  - and wire nets 25
  - driving strength of 72
  - explicit declaration 70
  - implicit declaration 70
  - versus procedural assignment 73
- control string 292
- conversion 10, 32
- copy
  - string operation 56
- counting number of drivers 789
- D
- d
  - decimal number format 8
- data argument 367
- data path 485, 529, 536
- data types 20–39
- deassign 67, 573
- deassign procedural statement 125
- decimal display format 8
  - and high impedance state 283
  - and unknown logic value 283
  - compatibility with \$monitor 283
- decimal notation 10
- declaring
  - events 138
  - multiple module paths in a single statement 221–222
  - parameters in specify blocks 37–38
- default
  - in case statement 131
  - in if-else-if statements 129
- default statement 201
- defparam 36, 182–183
- delay
  - calculating for high impedance (z) transitions 103
  - calculating for unknown logic value (x) transitions 103
  - control 136, 137
  - default 103
  - distributed 212–227
  - fall 103
  - falling 104
  - for continuous assignment 72
  - gate 103–104
  - minimum:typical:maximum values 104
  - module path 212–227
  - net 103–104
  - propagation 78, 103
  - rise 103, 104
  - rules for delays controlling the assignment 72
  - specify one value 103
  - specify three values 103
  - specify two values 103
  - triereg charge decay 105
  - turn-off 104
- delay selection 226
- delay specification 78
- delays
  - inertial 572, 642, 646, 648, 743
  - pure transport 572, 642, 646, 648, 743
  - transport 572, 642, 646, 648, 743
- delimiter 429
- describing simple module paths 214–215
- design 201
- design statement 204
- determinism in simulation execution 66
- diagnostic messages
  - from \$stop and \$finish 302
- disable
  - named blocks 163
  - tasks 163
  - use of 163
- displaying information 279–286
- displaying library binding information 210
- don't-care bits
  - in case statements 133
- double quote character 12

- drive strength specification 77
- driven state 26
- driving strength 89
  - compared to charge storage strength 285
  - keywords 73
- E
- e\_limit 409, 460, 561, 566
- edge control specifiers 259
- edge transitions 259
- edge-sensitive paths 215–219
- edge-sensitive state-dependent paths 218
- edge-sensitive UDPs 112
  - compared to level-sensitive UDPs 112
- element
  - of array 33
- else 128
- embedding modules 166, 168
- enable 141
- enabling tasks 152–154
- end
  - sequential block 146
- endconfig 200
- endgenerate 170
- endspecify 38, 212
- equality operators 46–47
  - != 46
  - !== 46
  - == 46
  - === 46
  - and ambiguous results 47
  - and operands of different sizes 47
  - precedence 47
- ERR\_ERROR 616
- ERR\_INTERNAL 616
- ERR\_MESSAGE 616
- ERR\_SYSTEM 616
- ERR\_WARNING 616
- escape sequences 279
- escaped identifiers 12
- espresso format 305
- event
  - active 64
  - control 136, 138
  - evaluation 64
  - explicit 136
  - expression 136
  - future 65
  - implicit 136
  - inactive 64
  - level sensitive control 141
  - monitor 65
  - named 64, 138–139
  - non blocking assign update 65
  - OR construct 139
  - queue 64
  - update 64
- event control
  - repeat 142–145
- event or 41
- event queue 64
  - scheduling an event 64
- event simulation 64
- event\_value\_change 399
- exception values 389
- exit simulator 302
- expanded object 24
- expanded vectors 520, 552
- expansion
  - of vector nets 24
- explicit event 136
- explicit zero delay 65
- expression
  - context-determined 59
  - scalar 40
- expressions ??–61
  - bit lengths 59–61
  - constant 40
  - self-determined 59
  - steps for evaluating 62
- F
- f
  - in state table 110
- fall delay 103, 104
- file descriptor 289
- file inclusion 358
- file output 584
- file path resolution 202
- file positioning 295
- finish time
  - in parallel block statements 148
  - in sequential block statements 148
- flushing output 295

- for loop 134
- force 67, 126, 573
  - precedence over assign 126
- forever loop 134
- fork-join block statement 146
- fork-join construct 144
- format specifications 280–282
  - ASCII character 280
  - b or B 280
  - binary 280
  - c or C 280
  - d or D 280
  - decimal 280
  - h or H 280
  - hexadecimal 280
  - hierarchical name 281
  - library binding 280
  - m or M 281
  - net signal strength 280, 284–285
  - o or O 280
  - octal 280
  - s or S 281
  - string 281, 286
  - t or T 281, 282
  - time format 281
  - timescales 282
  - u or U 281
  - v or V 280
  - z or Z 281
- formats
  - array 304
  - of logic array personality 304–307
  - plane 305
- formatting data to a string 290
- frames 687
- full connection 220–221
- fullname 444, 727
- fulltype 384, 446
- function
  - call 160
  - constant
    - calls 161
- functions 68, 157–161
  - and scope 198
  - as structured procedures 149
  - definition 150
  - purpose 152
  - returning a value 159
  - rules 160
- G
- gate level modeling ??–107
- gate type specification 77
- gates
  - and 81–82
  - bidirectional pass 86
    - delay specifications 86
  - buf 82–83
  - bufif 83–84
  - cmos 86–87
    - delay specification 86
  - compared to continuous assignments 75
  - connection list 79
  - delay 103–104
  - MOS 84–85
  - nand 81–82
  - nor 81–82
  - not 82–83
  - notif 83–84
  - notif0 83–84
  - notif1 83–84
  - or 81–82
  - pulldown 87
  - pullup 87
  - rules for instance connections 79
  - terminal list 79
  - xnor 81–82
  - xor 81–82
- generate 170
- generate case construct 178
- generate instantiations 170
- generate-conditional construct 177
- generated identifiers 13
- generate-loop 173
- genvar 173
- glitch control, see pulse control
- H
- H
  - logic 1 or high impedance state in strength format 284
- h
  - hexadecimal number format 8
- handles

- handle data type 369
- vpiHandle data type 662
- hexadecimal display format 8
  - and high impedance state 283
  - and unknown logic value 283
- Hi
  - high impedance in strength format 284
- hiconn definition 486
- hierarchical config
  - using 209
- hierarchical configurations 206
- hierarchical name 444
- hierarchical path name 193
- hierarchy
  - level 193
  - name referencing 193–199, 281
  - of modules 166
  - scope 193
  - scope rules for naming 198–199
  - structures 166–199
  - top level names 193
- high impedance state
  - and numbers 8
  - and trireg nets 26
  - and UDPs 116
  - display formats 283–285
  - effect in different bases 8
  - strength display format 284
  - symbolic representation 20
- highz0 77
- highz1 77
- I
- I/O error status 295
- identifiers 12
  - escaped 12
  - keywords 13
- if-else statement
  - omitting else from nested if 128
  - purpose 127
- If-else-if 128
- if-else-if statement
  - compared to case statement 132
- ifnone 483, 551
- ifnone condition 219
- implicit
  - declarations 25, 351
  - event 136
  - implicit bidirectional connections 68
  - implicit continuous assignment statements 68
  - implicit conversion 10, 32
  - implicit event 138
  - include command 203
  - incremental restart 793
  - incremental save 792
  - index
    - of array 33
    - of memory 33
  - inertial delays 572, 642, 646, 648, 743
  - initial 150
    - and activity flow 118
    - for specifying waveforms 150
  - initial statements
    - in UDPs 113–115
  - initializing access routines 370
  - instance statement 201
  - instantiation
    - of modules 166–170
  - integer constants 7
  - integer\_value\_change 399
  - integers 31
    - division 44
  - interactive scope 488
  - inter-module paths 496
  - intra-assignment timing controls 142–145
  - invocation options 586
  - io\_mcdprintf() 582, 584
  - io\_printf() 582, 585
  - K
  - keywords 13
  - L
  - L
    - logic 0 or high impedance state in strength format 284
  - La
    - large capacitor in strength format 284
  - large 24, 26
  - left-hand index 78
  - level-sensitive
    - event control 141
    - paths 216–220
    - sequential UDPs 112
    - versus combinational UDP 112

- level-sensitive UDPs
  - compared to edge-sensitive UDPs 112
- lexical conventions 6–14
- lexical token
  - comment 6
  - definition of 6
  - number 6
  - operator 6
  - types 6
  - white space 6
- liblist clause 201
- libraries 201
- library map
  - library declaration 201
- library notation 200
- load 523, 531
- loading memory data from a file 296
- loading timing data from an SDF file 297
- localparam 36
- loconn definition 489
- logic array
  - personality declaration and loading 304
- logic array personality 304–307
  - declaration 304
  - formats 304–307
  - loading 304
- logic gates
  - and 81–82
  - bidirectional pass 86
  - buf 82–83
  - bufif 83–84
  - cmos 86–87
  - compared to continuous assignments 75
  - delay 103–104
  - MOS 84–85
  - nand 81–82
  - nor 81–82
  - not 82–83
  - notif 83–84
  - or 81–82
  - pulldown 87
  - pullup 87
  - xnor 81–82
  - xor 81–82
- logic one 20
- logic planes 304
- logic strength modeling 88–102
- logic zero 20
- logic\_value\_change 399
- logical operators 47
  - ! 47
  - && 47
  - || 47
  - AND 40
  - and ambiguous results 47
  - and unknown logic value 47
  - equality 40
  - inequality 40
  - negation 40
  - OR 40
  - precedence 47
- looping statement 134–136
  - for loop 134
  - forever loop 134
  - repeat loop 134
  - while loop 134
- lsb (least significant bit) 23
- M
- mapping source files to libraries 203
- mc\_scan\_plusargs() 586
- Me
  - medium capacitor in strength format 284
- medium 24, 26
- memory 33
  - addressing 54
  - assigning values to 33
  - index 33
- memval structure 624
- minimum:typical:maximum values
  - delay 104
  - for module path delays 224, 225
  - format 57–59
- minus sign(-)
  - arithmetic subtraction operator 40
  - in state table 110
- miscf routines 366
- mixing path and distributed delays 227
- modeling
  - asynchronous clear/preset on an edge-triggered D flip-flop 125
  - logic strength 88–102
- module 166–169

- and user-defined primitives(UDPs) 107
- definition 166–167
- hierarchy 166
- instance parameter value assignment
  - multiple library mapping files 203
  - 183–184
- instance parameter value assignment by
  - ordered list 183
- instantiation 168–170
- overriding parameter values 180–185
- parameter assignment by name 184
- parameter dependencies 185
- port 169
- terminal 169
- top-level 168
- module cell 522, 523
- module parameter 35
  - dependencies 185
  - overriding values 180–185
  - passing to tasks 154–155
- module path 490, 497, 498, 529, 534, 536
  - definition 213
  - delay 223–227
  - destination 212, 214, 221
  - polarity 222–223
  - simple 214
  - source 212, 214, 221
- module path names 429
- module path restrictions 214
- modulus operator 40
  - definition 44
- monitor flag 287
- monitoring
  - continuous 287
  - strobed 286
- MOS gate 84–85
  - nmos 85
  - pmos 85
  - rnmos 85
  - rpmos 85
- MOS strength handling 102
- msb (most significant bit) 23
- mtm\_flag 711, 740
- multi channel descriptor 288
- multi-channel descriptor 289
- multiple drivers
  - at same strength level 99
  - driving the same net 26
  - inside a module 228
  - outside a module 229
- multiple module path delays
  - assigning in one statement 221–222
- multi-way decisions
  - case statement 130
  - if-else-if statement 129
- N
- n
  - in state table 110
- name 365, 453, 727
- name space 38
  - block name space 38
  - definitions 38
  - module name space 38
  - port name space 39
  - specify block name space 39
- name spaces 38
- named blocks
  - and hierarchical names 193
  - and scope 198
  - purpose 148
- named events 64, 138–139
  - used with event expressions 138
- named objects
  - with acc\_fetch\_fullname() 444
  - with acc\_fetch\_name() 453
  - with acc\_handle\_object() 493
- names
  - of hierarchical paths 193–199
- nand gate 81–82
- negative numbers 8
- negedge 138, 215, 259
- net and register bit addressing 54
- net arrays 33
- net delay 72
- net type resolution rule 192
- net type table 192
- net types 25
- nets 20–31
  - delay 103–104
  - initialization 25
  - triereg strength 89
  - types of 25–31

- wired logic 99
- new line character 12, 280
- newline character 12
- nmos 84–85
- node
  - in hierarchical name tree 193
- non blocking assignment statement 67
- non blocking procedural assignment 121–124
  - evaluating assignments 122
  - multiple assignments 123
- non-determinism in simulation execution 66
- nor gate 81–82
- not gate 82–83
- notif gate 83–84
  - notif0 84
  - notif1 84
- notifier 260–262
  - in edge sensitive UDP 260–262
- notifiers
  - user-defined responses to timing violations 260
- null
  - expression 279
- numbers 6
  - base format 8
  - size specification 8
- O
- o
  - octal number format 8
- object
  - full name 444
  - fulltype 446
  - fulltypes, list of all 384
  - name 453
  - type 471
  - types, list of all 384
- objects
  - supported by acc\_next() 516
  - supported by acc\_object\_in\_typelist() 549
  - supported by acc\_object\_of\_type() 551
  - supported by VCL 398, 575
- octal display format 8
- on/off control
  - of monitoring tasks 287
- one-line comment 6
- opening and closing files 287
- operands 52–57
  - bit-select 52
  - concatenation 52
  - definition 40
  - function call 52
  - part-select 52
  - strings 55–57
- operator
  - event OR 52
- operators 40–52
  - 40
  - ! 40, 47
  - != 40, 46
  - !== 41, 46
  - % 40
  - & 41
  - && 40, 47
  - \* 40
  - \*\* 40
  - \*> 214–222
  - + 40
  - / 40
  - < 40, 46
  - << 41, 49
  - <<< 41, 49
  - <= 40, 46
  - = 69
  - == 40, 46
  - === 41, 46
  - => 214–222
  - > 40, 46
  - >= 40, 46
  - >> 41, 49
  - >>> 41, 49
  - ?: 41
  - ^ 41
  - ^~ 41
  - {{}} 40
  - { } 40, 51
  - | 41
  - || 40, 47
  - ~ 41
  - ~& 41
  - ~^ 41

- ~| 41
- and real numbers 32
- arithmetic 40, 44
- binary 6, 42
- bit-wise 47–48
- bit-wise AND 41
- bit-wise equivalence 41
- bit-wise exclusive OR 41
- bit-wise inclusive OR 41
- bit-wise negation 41
- case equality 41
- case inequality 41
- concatenation 40, 51
- conditional 6, 41, 50–51
- definition 6
- equality 46–47
- event or 41
- left shift
  - arithmetic 41
  - logical 41
- logical 47
- logical AND 40
- logical equality 40
- logical inequality 40
- logical negation 40
- logical OR 40
- modulus 40
- reduction 48–49
- reduction AND 41, 48
- reduction NAND 41, 48
- reduction NOR 41, 48
- reduction OR 41, 48
- reduction XNOR 41, 48
- reduction XOR 41, 48
- relational 40, 46
- replication 40
- right shift
  - arithmetic 41
  - logical 41
- shift 49
- unary 6
- unary reduction 48
- or gate 81–82
- output
  - to files 287–290
- overloading system task/function names 363
- overriding module parameter values 180–185
  - assigning values in-line within module instances 183–184
- defparam 182
  - compared to assignmesions 184
- P
- p
  - in state table 110
- parallel block 147
- parallel block statement
  - finish time 148
  - start time 148
- parallel connection 220–221
- parameter attribute name 428
- Parameter Value Change flags, see PVC flags
- parameters 34
- paramvc argument 368
- parentheses
  - and changing operator precedence 43
- part-select
  - of vector net or register 52
  - references of real numbers 32
- path delay, see module path
- path delimiter 429
- path names 445
- PATHPULSE\$ specparam 230
- personality
  - memory 303
  - of logic array 304–307
- PLA devices
  - array logic types 304
  - array types 303–307
  - list of system tasks 303
  - logic array personality declaration 304
  - logic array personality formats 304–307
  - logic array personality loading 304
- plane
  - format 305
  - in programmable logic arrays 304
- PLI history 362
- PLI interface mechanism 363
- PLI memory restrictions 364
- plus sign(+)
  - arithmetic addition operator 40
- pmos 84–85
- polarity 222–223

- negative 223
- positive 222
- unknown 222
- port 185–193
  - connecting
    - by name 189–190
    - by position with ordered list 188
    - rules for 191–192
  - connecting module instance ports by name 189
  - connecting module instance ports by ordered list 188
  - declaration 186
  - definition 185
  - module 169
  - port connections 68
  - port expression 189
  - posedge 138, 215, 259
  - power operator 44
  - power supplies
    - modeled by supply nets 31
  - precedence
    - binary operators 42
    - equality operators 47
    - logical operators 47
    - relational operators 46
  - precompiling using a separate compilation tool 207
  - primitive instance identifier 78
  - printing, see text output
  - probabilistic distribution functions 312–313
    - \$dist\_chi\_square 313
    - \$dist\_erlang 313
    - \$dist\_exponential 313
    - \$dist\_normal 313
    - \$dist\_poisson 313
    - \$dist\_t 313
    - \$dist\_uniform 313
  - procedural assignment 119–127
    - and integers 32
    - and time variables 32
    - blocking 119
    - non blocking 121–124
    - versus continuous assignment 73
  - procedural assignments
    - blocking assignment 119
    - procedural continuous assignment 67, 573
    - procedural continuous assignments 124–127
      - assign 125–126
      - deassign 125–126
      - force 126
      - precedence 126
      - release 126
    - procedural force 573
    - byprocedural statements
      - in behavioral models 118
    - or-procedural timing controls 136–145
      - delay control 137
      - event control 136
      - fork-join block 147
      - intra-assignment timing controls 142–145
  - procedure
    - always construct 149
    - function 149
    - initial construct 149
    - task 149
  - process 64
  - programmable logic arrays
    - list of system tasks 303
    - logic types 304
  - personality
    - declaration 304
    - formats 304–307
    - loading 304
    - types 303–307
  - propagation delay
    - for gates and nets 103
  - Pu
    - pull drive in strength format 284
  - pull 26
  - pull0 77, 361
  - pull1 77, 361
  - pulldown 77
  - pulldown source 87
  - pullup 77
  - pullup source 87
  - pulse
    - negative
      - detection 233
  - pulse control 409, 460, 561, 566, 711, 740
    - detailed capabilities 232

- pulse filtering
  - on-event versus on-detect 232
- pulse limit value 230
  - global control of 231
  - SDF annotation 231
  - specify block control 230
- pulsere\_flag 711, 740
- pure transport delays 572, 642, 646, 648, 743
- PVC flags 592, 607, 619, 652
- Q
- qualified paths 215–219
  - edge-sensitive 215–219
  - level-sensitive 216–222
- queue management 307–309
  - \$q\_add 307, 308
  - \$q\_exam 307, 308
  - \$q\_full 307, 308
  - \$q\_initialize 307
  - \$q\_remove 307, 308
  - status parameters 309
- queueing models 307
- R
- r
  - in state table 110
- race condition 144
- race conditions 66
- random access memory(RAM)
  - modeled by register arrays 33
- random number generators
  - probabilistic distribution functions 312
- range specification 78
- rcmos 84, 86
- reading a character at a time 291
- reading a line at a time 291
- reading binary data 294
- reading formatted data 291
- read-only memory(ROM)
  - modeled by register arrays 33
- real constant numbers 10
- real declarations 32
- real number constants 32
- real numbers 31
  - and operators 32
  - conversion to integers 10, 32
  - format specifications used with 281
  - in port connections 190
  - operators with real number operands 41–42
- real variable data types 32
- real\_value\_change 399
- realtime
  - variables 31
- realtime declarations 32
- reason argument 367
- reason constants 367, 368
- reason\_disable 368
- reason\_endofcompile 367
- reason\_endofreset 368
- reason\_finish 367
- reason\_force 368
- reason\_interactive 368
- reason\_paramdrc 368
- reason\_paramvc 367, 588, 589
- reason\_reactivate 582, 635, 636, 637
- reason\_reactivate 367
- reason\_release 368
- reason\_reset 368
- reason\_restart 368
- reason\_rosynch 367, 582, 605, 632
- reason\_save 368
- reason\_scope 368
- reason\_startofsave 368
- reason\_synch 367, 581, 632, 651
- recursive, see frames 687
- reducing pessimism 132
- reduction operators 48–49
  - & 41
  - ~& 41
  - inclusive OR 41
  - unary AND 41
  - unary NAND 41
  - unary NOR 41
  - XNOR 41
  - XOR 41
- reentrant, see frames 687
- reg arrays 33
- reg declaration 22
- registers
  - and level-sensitive sequential UDPs 112
  - notifier 260
  - used in procedural assignments 73
- regs 31

- reject\_limit 409, 460, 561, 566
- relational operators 40, 46
  - < 46
  - <= 46
  - > 46
  - >= 46
  - precedence 46
- release 67, 126, 573
- repeat event control 142–145
- repeat loop 134
- replication
  - operator 40
- resistive devices
  - modeled with tri0 and tri1 nets 30
- restrictions on data types
  - in continuous assignments 69, 191
  - in procedural assignments 69, 73, 119
  - when connecting ports 191
- right-hand index 78
- rise delay 103, 104
- rnmos 84–85
- rpmos 84–85
- rtran 86
- rtranif0 86
- rtranif1 86
- rules
  - for delays controlling the assignment 72
  - for describing module paths 222
  - for expression bit lengths 59
  - for expression types 62
  - for instance connections 79
  - net type resolution 192
  - to use the \$dumpports 341
- S
- s
  - in string display format 286
- s\_acc\_time structure 573
- s\_acc\_value structure 475
- s\_acc\_vecval structure 476, 571
- s\_location data structure 451
- s\_setval\_delay structure 572
- s\_setval\_value structure 570
- s\_strengths structure 400
- s\_strengthval structure 623
- s\_tfexprinfo structure 599
- s\_tfnodeinfo structure 621
- s\_timescale\_info structure 469
- s\_vc\_record structure 399
- s\_vecval structure 599, 623
- s\_vpi\_delay structure 710
- s\_vpi\_time structure 710
- scalared 24
- scalars
  - compared to vectors 23
  - scalar nets and driving strength of continuous assignment 72
- scheduling semantics 64
- scientific notation 10
- scope 501, 543
  - and hierarchical names 193
  - rules 198–199
- SDF
  - INTERCONNECT construct 274
  - interconnect delay annotation 274
  - multiple annotations 275
  - pulse limit annotation 276
  - to Verilog delay value mapping 277
- SDF annotation
  - down-hierarchy annotation 275
  - hierarchically overlapping annotations 275
  - NETDELAY construct 274
  - of interconnect delays 274
  - of specparams 273
  - PATHPULSE 276
  - PATHPULSEPERCENT 276
  - PORT construct 274
  - up-hierarchy annotations 275
- SDF annotator 270
- SDF constructs
  - mapping to Verilog 270
- SDF delay constructs
  - mapping to Verilog declarations 270
- SDF files
  - backannotation 270
- SDF timing check constructs
  - mapping to Verilog 272
- seed 313
- self-determined expression 59
- sequential block 118
- sequential block statement 146–147
  - finish time 148

- start time 148
- sequential UDP initialization 113–115
- sequential UDPs
  - input and output fields in state table 110
- set of values (0, 1, x, z) 20
- shift operators 41, 49
  - << 49
  - <<< 49
  - >> 49
  - >>> 49
- short-circuiting 43
- showcancelled behavior 233
- signed expressions 62
  - handling 'X' and 'Z' 63
- signed integers 8
- signed value 45
- simple decimal number 8
- simple state-dependent paths 217
- simulated net 502, 551, 552
- simulating module path delays
  - when driving wired logic 228–229
- simulation
  - going back with incremental restart 793
- simulation cycle 65
- simulation reference model 65
- simulation time 64
- single-pass use-model
  - elaboration-time compiling 207
  - precompiling 207
- size constant 8
- size of displayed data 282–283
- sized numbers 8
- size of displayed data 282–283
- sizeof routines 365
- Sm
  - small capacitor in strength format 284
- small 24, 26
- source
  - pulldown 87
  - pullup 87
- specify 38, 212
- specify block ??–237
- specify block system tasks
  - \$hold 242
  - \$period 256
  - \$recovery 246
  - \$setuphold 243
  - \$skew 250
  - \$timeskew 251
  - \$width 255–256
- specify parameters 37–38
  - as run time constant in specify block 213
- specifying the time unit of delays entered interactively 299
- specifying transition delays on module paths 224–225
  - x transitions 225–226
- specparam 37–38, 273
- specparam attribute name 428
- sregister\_value\_change 399
- St
  - strong drive in strength format 284
- standard output 288
- start time
  - in parallel block statements 148
  - in sequential block statements 148
- state dependent path delays 216–222
- stochastic analysis 312–313
  - probabilistic distribution functions 312–313
  - queue management 307–309
- stop 302
- strength 77–78
  - ambiguous 90–101
    - classifications 90
  - and MOS gates 102
  - and scalar net variables 20
  - charge storage 89
  - driving 89
  - gates that accept specifications 77
  - of combined signals 89–101
  - on trireg nets 26
  - range of possible values 91
  - reduction by non-resistive devices 102
  - reduction by resistive devices 102
  - reduction table 102
  - scale of strengths 89
  - specification 88
  - supply net 102
  - tri0 102
  - tri1 102
  - trireg 102
- strength display format 284–285

- high impedance 284
- large capacitor 284
- logic value 0,1,H,L,X,Z 284
- medium capacitor 284
- pull drive 284
- small capacitor 284
- strong drive 284
- supply drive 284
- weak drive 284
- strength\_value\_change 399
- strengths 24
  - of net types 102
- string buffer 564
- string handling 395
- strings 10–12, 55–57
  - definition 10
  - display format 281, 286
  - in vector variables 56
  - manipulation 11
  - operations 56
  - padding 11
  - special characters 11
  - value padding 56–57
  - variable declaration 11
- strobed monitoring 286
- strong 26
- strong0 77
- strong1 77
- structured procedure 149–151
  - always construct 149
  - function 149
  - initial construct 149
  - task 149
- Su
  - supply drive in strength format 284
- supply 26
- supply net strength 102
- supply nets 31
- supply0 77
- supply1 77
- switch processing 67
- switches
  - MOS 84–85
- synchronous arrays 303–307
- system functions 278–313
- system task/function arguments 364
- system task/function name 365
- system task/functions 514
- system tasks 278–313
  - for continuous monitoring 287
  - for displaying information 279–286
  - for interrupting the simulator 302
  - for processing stimulus patterns faster 789
  - for showing number of drivers 789
  - for writing formatted output to files 287–290
  - generating a checkpoint in the value change dump file 328
  - limiting the size of the value change dump file 328
  - reading the value change dump file during a simulation 329
  - resuming the dump into the value change dump file 327–328
  - showing the timescale of a module 298–299
  - specifying how %t reports time information 299–302
  - specifying the name of the value change dump file 325
  - specifying the variables to be dumped in the value change dump file 326
  - stopping the dump into the value change dump file 327–328
- System tasks and functions 13
- system tasks and functions 278–313
- T
- t
  - timescale format 282
- tab character 12
- task enabling statement 154
- task/function arguments 364, 512
- task/function name 365
- task/function routines
  - history 362
- tasks 68, ??–161
  - and hierarchical names 193
  - and scope 198
  - as structured procedures 149
  - definition 150
  - disabling within a nested chain 163

- enabling 152–154
- passing parameters 154–155
- purpose 152
- text macro substitutions 352–354
  - and `define 352
  - definition 352
  - redefinition 354
  - with arguments 352
- text output
  - io\_mcdprintf() 584
  - io\_printf() 585
  - tf\_error() 596
  - tf\_message() 616
  - tf\_text() 653
  - tf\_warning() 657
  - vpi\_mcd\_close() 730
  - vpi\_mcd\_name() 732
  - vpi\_mcd\_open() 733
  - vpi\_mcd\_printf() 734
  - vpi\_printf() 736
- tf\_add\_long() 587
- tf\_asynchoff() 581, 588
- tf\_asynchon() 581, 589, 592, 607, 619, 652
- tf\_clearalldelays() 582, 590
- tf\_compare\_long() 591
- tf\_copypvc\_flag() 581, 589, 592
- tf\_divide\_long() 593
- tf\_dofinish() 582, 594
- tf\_dostop() 582, 595
- tf\_error() 582, 596, 617
- tf\_evaluatep() 579, 597
- tf\_exprinfo() 579, 597, 598, 626
- tf\_getcstringp() 580, 601
- tf\_getinstance() 582, 602
- tf\_getlongp() 579, 603
- tf\_getlongtime() 581, 604
- tf\_getnextlongtime() 605
- tf\_getp() 579, 606
- tf\_getpchange() 581, 589, 607
- tf\_getrealp() 579, 608
- tf\_getrealtime() 609
- tf\_gettime() 581, 610
- tf\_gettimeprecision() 581, 611
- tf\_gettimeunit() 581, 612
- tf\_getworkarea() 582, 613, 638
- tf\_iasynchoff() 588
- tf\_iasynchon() 589, 592, 607, 619, 652
- tf\_iclearalldelays() 590
- tf\_icopypvc\_flag() 592
- tf\_ievaluatep() 597
- tf\_iexprinfo() 597, 598, 626
- tf\_igetcstringp() 601
- tf\_igetlongp() 603
- tf\_igetlongtime() 604
- tf\_igetp() 606
- tf\_igetpchange() 607
- tf\_igetrealp() 608
- tf\_igetrealtime() 609
- tf\_igettime() 610
- tf\_igettimeprecision() 611
- tf\_igettimeunit() 612
- tf\_igetworkarea() 613, 638
- tf\_imipname() 618
- tf\_imovepvc\_flag() 619
- tf\_inodeinfo() 621
- tf\_integer\_node 622
- tf\_inump() 625
- tf\_ipropagatep() 626
- tf\_iputlongp() 627
- tf\_iputp() 628
- tf\_iputrealp() 629
- tf\_irosynchronize() 632
- tf\_isetdelay() 590, 635
- tf\_isetlongdelay() 636
- tf\_isetrealdelay() 637
- tf\_isetworkarea() 613, 638
- tf\_isizep() 639
- tf\_ispname() 640
- tf\_istrdelputp() 641
- tf\_istrgetp() 643
- tf\_istrlongdelputp() 645
- tf\_istrrealdelputp() 647
- tf\_isynchronize() 651
- tf\_itestpvc\_flag() 652
- tf\_itypep() 654
- tf\_long\_to\_real() 614
- tf\_longtime\_tostr() 615
- tf\_memory\_node 622
- tf\_message() 582, 616, 653
- tf\_mipname() 582, 618
- tf\_movepvc\_flag() 581, 589, 607, 619
- tf\_multiply\_long() 620

tf\_netscalar\_node 622  
 tf\_netvector\_node 622  
 tf\_nodeinfo() 580, 621  
 tf\_null\_node 622  
 tf\_nullparam 599, 654  
 tf\_nump() 579, 625  
 tf\_propagatep() 579, 626  
 tf\_putlongp() 579, 627  
 tf\_putp() 579, 628  
 tf\_putrealp() 579, 629  
 tf\_read\_restart() 630  
 tf\_readonly 599, 654  
 tf\_readonlyreal 599, 654  
 tf\_readwrite 599, 654  
 tf\_readwritereal 599, 654  
 tf\_real\_node 622  
 tf\_real\_to\_long() 631  
 tf\_reg\_node 622  
 tf\_rosynchronize() 581, 632  
 tf\_rwbselect 599  
 tf\_rwmselect 599  
 tf\_rwpselect 599  
 tf\_scale\_longdelay() 633  
 tf\_scale\_realdelay() 581, 634  
 tf\_setdelay() 582, 590, 635  
 tf\_setlongdelay() 636  
 tf\_setrealdelay() 637  
 tf\_setworkarea() 582, 613, 638  
 tf\_sizep() 639  
 tf\_spname() 582, 640  
 tf\_strdelputp() 579, 641  
 tf\_strgetp() 579, 643  
 tf\_strgettime() 581, 644  
 tf\_string 599, 654  
 tf\_strlongdelputp() 579, 645  
 tf\_strealdelputp() 579, 647  
 tf\_subtract\_long() 649  
 tf\_synchronize() 65  
 tf\_synchronize() 581, 632, 651  
 tf\_testpvc\_flag() 581, 589, 652  
 tf\_text() 582, 617, 653  
 tf\_time\_node 622  
 tf\_typep() 579, 654  
 tf\_unscale\_longdelay() 581, 655  
 tf\_unscale\_realdelay() 581, 656  
 tf\_warning() 582, 617, 657  
 tf\_write\_save() 630, 658  
 tfargs 364  
 time  
     arithmetic operations performed on time  
         variables 32  
     variables 31  
 time precision 360  
 time unit 360  
 time\_value\_change 399  
 timing checks ??–267, 504, 508, 510  
     \$hold 242  
     \$period 256  
     \$recovery 246  
     \$recrem 247  
     \$removal 245  
     \$setup 242  
     \$setuphold 243  
     \$skew 250  
     \$timeskew 251  
     \$width 255–256  
     negative 268  
         conditions 264  
         notifiers 266  
     using a stability window 241  
     vector signals 267  
 timing checks for clock and control signals  
 249  
 top-level module 168  
 tran 86  
 tranif0 86  
 tranif1 86  
 transistors 86  
 transitions  
     01 113  
     unspecified 112  
 transport delays 572, 642, 646, 648, 743  
 tree structure  
     of hierarchical names 193  
 tri nets 25  
 tri0 30  
     net type 102  
 tri1 30  
     net type 102  
 triand 26  
 trior 26  
 trireg

- and charge storage strength 89
- turn-off delay 104
- type 384, 471
- types of nets
  - supply nets 31
  - tri nets ??–25, 25–??
  - tri0 102
  - tri0 nets 30
  - tri1 102
  - tri1 nets 30
  - triand 26
  - trior 26
  - trireg 102
  - trireg nets 26, 285
  - wire 25
  - wired AND 26
  - wired logic 99
  - wired nets 26
  - wired OR 26
- U
- UDP port declarations 109
- UDPs ??–117
  - in state table 110
  - (?) in state table 110
  - (01) in state table 110
  - (0x) in state table 110
  - (1x) in state table 110
  - (vw) in state table 110
  - (x1) in state table 110
  - \* in state table 110
  - ? in state table 110
  - 0 in state table 110
  - 1 in state table 110
  - b in state table 110
  - combinational UDPs 111
  - definition 107–109
  - edge-sensitive UDPs 112
  - f in state table 110
  - instances 115–116
  - level-sensitive dominance 117
  - level-sensitive sequential UDPs 112
  - mixing level- and edge-sensitive descriptions 116–117
  - n in state table 110
  - p in state table 110
  - ports 109
  - r in state table 110
  - state table 109
  - summary of symbols in state table 110
  - x in state table 110
- unary arithmetic operators 44
- unary operators 6
  - ! 47
  - << 49
  - >> 49
- unconnected port 169
- undescape character 8
- unexpanded object 24
- unexpanded vectors 552
- unknown logic value
  - and numbers 8
  - display formats 283–285
  - effect in different bases 8
  - in state table 110, 113
  - symbolic representation 20
- unsigned integers 8
- unsigned number 8
- unsigned value 45
- unspecified transitions 112
- upwards name referencing 196–199
- User defined system tasks and functions 659
- user-defined primitives (UDPs) 107
- user-defined system task/function name 365
- user-defined system task/function name overloading 363
- user-defined system task/function names 362
- user-defined system task/function types 363
- user-defined system task/functions 514
- using VCL access routines 397
- V
- value change dump file ??–339
  - creating 325–329
  - creating the extended file 340
  - extended VCD node information 346
  - format 330–339
  - formats of variable values 332–333
  - general rules for extended VCD system tasks 344
  - generating a checkpoint 328, 342
  - keyword commands
    - \$comment 333
    - \$date 334

- \$dumpall 337
- \$dumpoff 337, 338, 344
- \$enddefinitions 334
- \$scope 334
- \$timescale 335
- \$upscope 335
- \$var 336
- \$version 335
- limiting the size 328
- limiting the size of the dump file 342
- reading the dump file during simulation 343
- reading the value change dump file during a simulation 329
- resuming the dump 327–328
- rules to conflicts 349
- specifying the dumpfile name and the ports to be dumped 340
- specifying the name 325
- specifying the variables to be dumped 326
- stopping and resuming the dump 341
- stopping the dump 327–328
- value changes 348
- Value Change Link, see VCL
- value set (0, 1, x, z) 20
- values
  - of combined signals 89–101
- variables 22–23
- VCD file
  - extended 344
- VCL 575, 577
- vcl\_verilog 577
- vcl\_verilog\_logic 398, 576
- vcl\_verilog\_strength 398, 576
- vcl0 400
- vcl1 400
- vclHighZ 400
- vclLarge 400
- vclMedium 400
- vclPull 400
- vclSmall 400
- vclStrong 400
- vclSupply 400
- vclWeak 400
- vclX 400
- vclx 400
- vclz 400
- vector\_value\_change 399
- vectored 24
- vectors 23
  - and vector net expansion 24
  - expanded 520, 552
  - unexpanded 552
- vlog\_startup\_routines array 756
- VPI data model diagrams
  - active time format 697
  - assignments 693
  - attributes 698
  - case statement 695
  - continuous assignments 689
  - delay controls 693
  - delay terminals 688
  - event controls 693
  - expressions 691
  - expressions, simple 690
  - for loops 694
  - forever loops 694
  - frames 687
  - function calls 686
  - functions 685
  - if statement 695
  - instance arrays 671
  - inter-module paths 684
  - IO declarations 672
  - iterator 699
  - memories 679
  - module paths 684
  - modules 670
  - named events 680, 692
  - net drivers and loads 688
  - nets 674
  - object range 680
  - parameters 681
  - path term 684
  - ports 673
  - primitives 682
  - procedural assign statement 696
  - procedural blocks 692
  - procedural deassign statement 696
  - procedural disable statement 696
  - procedural force statement 696

- procedural release statement 696
- process 692
- reg drivers and loads 688
- regs 676
- repeat controls 693
- repeat loops 694
- scopes 672
- simple expressions 690
- specparams 681
- statements 692
- task calls 686
- tasks 685
- timing check 685
- UDPs 683
- variables 678
- wait control 694
- while loops 694
- VPI routines
  - callback overview 659
  - error handling 660
  - history 362
  - key to data model diagrams 666
  - lsited by functional groups 664
  - object access overview 660
  - object classifications 661
  - object types 663
  - traversing expressions 660
- vpi\_chk\_error() 701
- vpi\_compare\_objects() 702
- vpi\_control() 703
- vpi\_flush() 704
- vpi\_free\_object() 705
- vpi\_get() 706
- vpi\_get\_cb\_info() 707
- vpi\_get\_data() 708
- vpi\_get\_delays() 710
- vpi\_get\_str() 713
- vpi\_get\_systf\_info() 714
- vpi\_get\_time() 715
- vpi\_get\_userdata() 716
- vpi\_get\_value() 717
- vpi\_get\_vlog\_info() 723
- vpi\_handle() 724
- vpi\_handle\_by\_index() 725
- vpi\_handle\_by\_multi\_index() 726
- vpi\_handle\_by\_name() 727
- vpi\_handle\_multi() 728
- vpi\_iterate() 729
- vpi\_mcd\_close() 730
- vpi\_mcd\_flush() 731
- vpi\_mcd\_name() 732
- vpi\_mcd\_open() 733
- vpi\_mcd\_printf() 734
- vpi\_mcd\_vprintf() 735
- vpi\_printf() 736
- vpi\_put\_data() 737
- vpi\_put\_delays() 739
- vpi\_put\_userdata() 742
- vpi\_put\_value() 743
- vpi\_register\_cb() 65, 746
- vpi\_register\_systf() 754
- vpi\_remove\_cb() 758
- vpi\_scan() 759
- vpi\_vprintf() 760
- vpiCancelEvent 743, 744
- vpiFile 663
- vpiForceFlag 743
- vpiHandle 662
- vpiInertialDelay 743
- vpiInterModPath 728
- vpiIntFunc 755
- vpiIterator 729
- vpiLineNo 663
- vpiMultiArray 726
- vpiNoDelay 743
- vpiPureTransportDelay 743
- vpiRealFunc 755
- vpiReleaseFlag 743
- vpiReturnEvent 743
- vpiScaledRealTime 745
- vpiSchedEvent 743
- vpiScheduled 744
- vpiSizedFunc 755
- vpiSizedSignedFunc 755
- vpiSysFunction 755
- vpiSysTask 755
- vpiTimeFunc 755
- vpiTimeUnit 706
- vpiTransportDelay 743
- vpiType 663
- vregister\_value\_change 399

W  
 wait statement  
     as level-sensitive event control 141  
     to advance simulation time 136  
 wand 26  
 We  
     weak drive in strength format 284  
 weak 26  
 weak0 77  
 weak1 77  
 while loop 134  
 white space 6  
 wired AND configurations 26  
 wired logic nets  
     wand 99  
     wired-AND configurations 26  
     wired-OR configurations 26  
     wor 99  
 wired-OR configurations 26  
 wires 25  
 wor 26  
 word  
     of array 33  
 writing formatted output to files 287–290  
 writing to files 584  
 X  
 X  
     as display format for unknown logic value 283  
     unknown logic value in strength format 284  
 x  
     as display format for unknown logic value 283  
     in state table 110  
     unknown logic value 20  
 xnor gate 81–82  
 xor gate 81–82  
 Z  
 Z  
     as display format for high impedance state 283  
     high impedance state in strength format 284  
 z  
     as display format for high impedance state 283  
     high impedance state 20



















































