

Table 1— Specifying special characters in string.....	12
Table 2— Net types .....	25
Table 3— Truth table for wire and tri nets .....	25
Table 4— Truth table for wand and triand nets .....	26
Table 5— Truth table for wor and trior nets.....	26
Table 6— Truth table for tri0 net.....	30
Table 7— Truth table for tri1 net.....	30
Table 8— Differences between specparams and parameters.....	37
Table 9— Operators in the Verilog HDL .....	40
Table 10—Legal operators for use in real expressions.....	41
Table 11—Operators not allowed for real expressions.....	42
Table 12—Precedence rules for operators .....	42
Table 13—Arithmetic operators defined .....	44
Table 14—Unary operators defined.....	44
Table 15—Examples of modulus operators.....	44
Table 16—Data type interpretation by arithmetic operators .....	45
Table 17—Definitions of the relational operators .....	46
Table 18—Definitions of the equality operators .....	46
Table 19—Bit-wise binary and operator.....	48
Table 20—Bit-wise binary or operator .....	48
Table 21—Bit-wise binary exclusive or operator.....	48
Table 22—Bit-wise binary exclusive nor operator.....	48
Table 23—Bit-wise unary negation operator.....	48
Table 24—Reduction unary and operator.....	49
Table 25—Reduction unary or operator .....	49
Table 26—Reduction unary exclusive or operator .....	49
Table 27—Results of unary reduction operations .....	49

Table 28—Ambiguous condition results for conditional operator .....	51
Table 29—Bit lengths resulting from self-determined expressions .....	60
Table 30—Legal left-hand side forms in assignment statements .....	69
Table 31—Built-in gates and switches .....	77
Table 32—Valid gate types for strength specifications .....	77
Table 33—Truth tables for multiple input logic gates.....	82
Table 34—Truth tables for multiple output logic gates.....	83
Table 35—Truth tables for three-state logic gates.....	84
Table 36—Truth tables for MOS switches .....	85
Table 37—Strength levels for scalar net signal values .....	88
Table 38—Strength reduction rules .....	102
Table 39—Rules for propagation delays .....	103
Table 40—UDP table symbols .....	110
Table 41—Initial statements in UDPs and modules .....	113
Table 42—Mixing of level-sensitive and edge-sensitive entries .....	117
Table 43—Detecting posedge and negedge.....	138
Table 44—Intra-assignment timing control equivalence.....	144
Table 45—Net types resulting from dissimilar port connections .....	192
Table 46—List of valid operators in state dependent path delay expression .....	217
Table 47—Associating path delay expressions with transitions.....	224
Table 48—Calculating delays for x transitions .....	226
Table 49—\$setup arguments .....	242
Table 50—\$hold arguments.....	243
Table 51—\$setphold arguments .....	244
Table 52—\$removal arguments.....	246
Table 53—\$recovery arguments.....	247
Table 54—\$crem arguments.....	248

Table 55—\$skew arguments.....	250
Table 56—\$timeskew arguments .....	251
Table 57—\$fullskew arguments .....	253
Table 58—\$width arguments.....	255
Table 59—\$period arguments.....	257
Table 60—\$nochange arguments .....	258
Table 61—User-defined responses to timing violations.....	260
Table 62—Mapping of SDF delay constructs to Verilog declarations.....	271
Table 63—Mapping of SDF timing check constructs to Verilog.....	272
Table 64—SDF annotation of interconnect delays.....	274
Table 65—SDF to Verilog delay value mapping .....	277
Table 66—Escape sequences for printing special characters .....	280
Table 67—Escape sequences for format specifications.....	280
Table 68—Format specifications for real numbers .....	282
Table 69—Logic value component of strength format.....	284
Table 70—Mnemonics for strength levels.....	284
Table 71—Explanation of strength formats.....	285
Table 72—Types for file descriptors .....	288
Table 73—mtm spec argument.....	298
Table 74—scale type argument .....	298
Table 75—\$timeformat units_number arguments .....	300
Table 76—\$timeformat default value for arguments.....	300
Table 77—Diagnostics for \$finish.....	302
Table 78—PLA modeling system tasks.....	303
Table 79—Types of queues of \$q_type values.....	308
Table 80—Parameter values for \$q_exam system task .....	308
Table 81—Status parameter values .....	309

Table 82—Verilog to C function cross-listing.....	314
Table 83—Rules for left-extending vector values .....	332
Table 84—How the VCD can shorten values.....	332
Table 85—Keyword commands .....	333
Table 86—Arguments of time_precision.....	360
Table 87—(normative) Predefined misc tf reason constants .....	367
Table 88—(informative) Additional misc tf reason constants.....	368
Table 89—List of fetch routines .....	371
Table 90—List of handle routines .....	372
Table 91—How next routines use the target object argument.....	373
Table 92—List of next routines .....	374
Table 93—Values that can be modified.....	375
Table 94—List of modify routines .....	375
Table 95—List of miscellaneous routines .....	376
Table 96—List of VCL routines .....	376
Table 97—Operations on module instances .....	378
Table 98—Operations on module ports.....	378
Table 99—Operations on bits of a port.....	379
Table 100—Operations on module paths and data paths.....	379
Table 101—Operations on intermodule paths .....	380
Table 102—Operations on top-level modules .....	380
Table 103—Operations on primitive instances.....	380
Table 104—Operations on primitive terminals .....	381
Table 105—Operations on nets .....	381
Table 106—Operations on reg types .....	382
Table 107—Operations on integer, real, and time variables .....	382
Table 108—Operations on named events .....	382

Table 109—Operations on parameters and specparams.....	383
Table 110—Operations on timing checks.....	383
Table 111—Operations on timing check terminals .....	383
Table 112—Operations on user-defined system task/function arguments .....	384
Table 113—List of all predefined type and fulltype constants.....	384
Table 114—Exception values returned by ACC routines on errors .....	389
Table 115—Number of possible delays for Verilog HDL objects .....	390
Table 116—Configuration parameters for delay ACC routines .....	391
Table 117—Number of delay arguments in single delay mode .....	391
Table 118—Number of delay elements in min:typ:max delay mode .....	392
Table 119—Configuring accToHiZDelay to determine the toZ delay .....	395
Table 120—Predefined vc_reason constants .....	399
Table 121—Predefined out_value constants .....	400
Table 122—Predefined out_value constants .....	400
Table 123—Pulse control example.....	409
Table 124—How the value of accPathDelayCount affects acc_append_pulse() .....	410
Table 125—accDefaultAttr0 configuration parameter .....	416
Table 126—accDevelopmentVersion configuration parameter .....	416
Table 127—accDisplayErrors configuration parameter .....	416
Table 128—accDisplayWarnings configuration parameter.....	416
Table 129—accEnableArgs configuration parameter.....	417
Table 130—accMapToMipd configuration parameter .....	417
Table 131—accMinTypMaxDelays configuration parameter .....	417
Table 132—accPathDelayCount configuration parameter .....	418
Table 133—accPathDelimStr configuration parameter.....	418
Table 134—accToHiZDelay configuration parameter .....	419
Table 135—Naming conventions for attributes.....	428

Table 136—Example module path names using delimiter strings .....	429
Table 137—Controlling the default value returned by <code>acc_fetch_attribute()</code> .....	430
Table 138—Predefined constants used by <code>acc_fetch_delay_mode()</code> .....	435
Table 139—The operation of <code>acc_fetch_direction()</code> .....	441
Table 140—Edge specifiers constants .....	442
Table 141—Named objects supported by <code>acc_fetch_fullname()</code> .....	444
Table 142—Module path names returned by <code>acc_fetch_fullname()</code> .....	445
Table 143—The difference between the type and the fulltype of an object .....	446
Table 144—Deriving indices.....	449
Table 145—Objects supported by <code>acc_fetch_location()</code> .....	451
Table 146—Named objects supported by <code>acc_fetch_name()</code> .....	453
Table 147—Module path names returned by <code>acc_fetch_name()</code> .....	454
Table 148—Casting <code>acc_fetch_paramval()</code> return values.....	456
Table 149—Polarity types returned by <code>acc_fetch_polarity()</code> .....	458
Table 150—Value returned by <code>acc_fetch_precision()</code> .....	459
Table 151—Pulse control example.....	460
Table 152—How the <code>accPathDelayCount</code> affects <code>acc_fetch_pulsere()</code> .....	461
Table 153—Casting <code>acc_fetch_tfarg()</code> return values.....	466
Table 154—Return values from <code>acc_fetch_timescale_info()</code> .....	469
Table 155—Value returned by <code>acc_fetch_timescale_info()</code> .....	470
Table 156—The difference between the type and the fulltype of an object .....	471
Table 157—How <code>acc_fetch_value()</code> returns values.....	474
Table 158—Format constants for the <code>s_acc_value</code> structure.....	475
Table 159—Encoding of bits in the <code>s_acc_vecval</code> structure.....	476
Table 160—How <code>acc_handle_by_name()</code> works.....	480
Table 161—Named objects supported by <code>acc_handle_by_name()</code> .....	480
Table 162—How <code>acc_handle_modpath()</code> works .....	490

Table 163—How <code>acc_handle_object()</code> works .....	493
Table 164—Named objects.....	493
Table 165—Deriving port indices .....	499
Table 166—How <code>acc_handle_tchk()</code> works .....	505
Table 167—Edge group constants .....	505
Table 168—Edge specific constants.....	506
Table 169—Deriving terminal indices.....	511
Table 170—How <code>acc_handle_tfarg()</code> operates .....	512
Table 171—Type and fulltype constants supported by <code>acc_next()</code> .....	517
Table 172—How <code>acc_next_child()</code> works .....	525
Table 173—How <code>acc_next_port()</code> works.....	539
Table 174—Special object properties.....	551
Table 175—Product types returned by <code>acc_product_type()</code> .....	553
Table 176—Pulse control example.....	561
Table 177—How the value of <code>accPathDelayCount</code> affects <code>acc_replace_pulsere()</code> .....	562
Table 178—Pulse control example.....	566
Table 179—How <code>acc_set_scope()</code> works .....	568
Table 180—Predefined constants for the format field of <code>s_setval_value</code> .....	571
Table 181—Encoding of bits in the <code>s_acc_vecval</code> structure.....	572
Table 182—Predefined delay constants for the model field of <code>s_setval_delay</code> .....	572
Table 183—Predefined time constants for the type field of <code>s_acc_time</code> .....	573
Table 184—Predefined assign/force constants for the model field of <code>s_setval_delay</code> .....	574
Table 185— <code>vcl_flag</code> constants used in <code>acc_vcl_add()</code> .....	576
Table 186—Return values for <code>tf_compare_long()</code> .....	591
Table 187—Predefined constants used with <code>tf_exprinfo()</code> .....	599
Table 188—avalbits/bvalbits encoding.....	600
Table 189—Return values for <code>tf_getnextlongtime()</code> .....	605

Table 190—Code returned by <code>tf_gettimeprecision()</code> and <code>tf_i_gettimeprecision()</code> .....	611
Table 191—Code returned by <code>tf_gettimeunit()</code> and <code>tf_i_gettimeunit()</code> .....	612
Table 192—Predefined constants for <code>node_type</code> .....	622
Table 193—How the <code>node_value</code> union is used .....	623
Table 194— <code>avalbits/bvalbits</code> encoding .....	623
Table 195—Format characters .....	641
Table 196— <code>delaytype</code> codes .....	642
Table 197—Format characters .....	643
Table 198—Format characters .....	645
Table 199— <code>delaytype</code> codes .....	646
Table 200—Format characters .....	647
Table 201— <code>delaytype</code> codes .....	648
Table 202—Predefined <code>tf_typep()</code> constants .....	654
Table 203—VPI routines for simulation related callbacks .....	664
Table 204—VPI routines for system task/function callbacks .....	664
Table 205—VPI routines for traversing Verilog HDL hierarchy .....	665
Table 206—VPI routines for accessing properties of objects .....	665
Table 207—VPI routines for accessing objects from properties .....	665
Table 208—VPI routines for delay processing .....	665
Table 209—VPI routines for logic and strength value processing .....	665
Table 210—VPI routines for simulation time processing .....	665
Table 211—VPI routines for miscellaneous utilities .....	666
Table 212—Return error constants for <code>vpi_chk_error()</code> .....	701
Table 213—Size of the <code>s_vpi_delay-&gt;da</code> array .....	711
Table 214—Return value field of the <code>s_vpi_value</code> structure union .....	719
Table 215—Size of the <code>s_vpi_delay-&gt;da</code> array .....	740
Table 216—Value format field of <code>cb_data_p-&gt;value-&gt;format</code> .....	747

Table 217—cbStmt callbacks ..... 750