

INTERNATIONAL STANDARD

**IEC
61691-4**

First edition
2004-10

IEEE 1364™

Behavioural languages –

Part 4:

Verilog® hardware description language

Itch Standards
(<https://standards.itech.ai>)
Document Preview

[IEC 61691-4:2004](https://standards.itech.ai/iec/61691-4:2004)

<https://standards.itech.ai/catalog/standards/iec/2a16ba1a-9b1a-43a7-9605-bf8b29930eed/iec-61691-4-2004>



IEEE

Reference number
IEC 61691-4(E):2004
IEEE Std. 1364(E):2001

Publication numbering

As from 1 January 1997 all IEC publications are issued with a designation in the 60000 series. For example, IEC 34-1 is now referred to as IEC 60034-1.

Consolidated editions

The IEC is now publishing consolidated versions of its publications. For example, edition numbers 1.0, 1.1 and 1.2 refer, respectively, to the base publication, the base publication incorporating amendment 1 and the base publication incorporating amendments 1 and 2.

Further information on IEC publications

The technical content of IEC publications is kept under constant review by the IEC, thus ensuring that the content reflects current technology. Information relating to this publication, including its validity, is available in the IEC Catalogue of publications (see below) in addition to new editions, amendments and corrigenda. Information on the subjects under consideration and work in progress undertaken by the technical committee which has prepared this publication, as well as the list of publications issued, is also available from the following:

- **IEC Web Site** (www.iec.ch)

- **Catalogue of IEC publications**

The on-line catalogue on the IEC web site (www.iec.ch/searchpub) enables you to search by a variety of criteria including text searches, technical committees and date of publication. On-line information is also available on recently issued publications, withdrawn and replaced publications, as well as corrigenda.

- **IEC Just Published**

This summary of recently issued publications (www.iec.ch/online_news/justpub) is also available by email. Please contact the Customer Service Centre (see below) for further information.

- **Customer Service Centre**

If you have any questions regarding this publication or need further assistance, please contact the Customer Service Centre:

Email: custserv@iec.ch
Tel: +41 22 919 02 11
Fax: +41 22 919 03 00

INTERNATIONAL STANDARD

**IEC
61691-4**

First edition
2004-10

IEEE 1364™

Behavioural languages –

Part 4: Verilog® hardware description language

Withdrawing
It is a registered trademark in the U.S. Patent & Trademark Office, owned by the Institute of Electrical and Electronics Engineers, Inc.
(<https://standards.iteh.ai>)
Document Preview

<https://standards.iteh.ai/standards/iec/2-16ba1a-9b1a-43a7-9605-bf8b29930eed/iec-61691-4-2004>

<https://standards.iteh.ai/standards/iec/2-16ba1a-9b1a-43a7-9605-bf8b29930eed/iec-61691-4-2004>

Copyright © IEEE 2004 — All rights reserved

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by the Institute of Electrical and Electronics Engineers, Inc.

No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

International Electrotechnical Commission, 3, rue de Varembé, PO Box 131, CH-1211 Geneva 20, Switzerland
Telephone: +41 22 919 02 11 Telefax: +41 22 919 03 00 E-mail: inmail@iec.ch Web: www.iec.ch

The Institute of Electrical and Electronics Engineers, Inc, 3 Park Avenue, New York, NY 10016-5997, USA
Telephone: +1 732 562 3800 Telefax: +1 732 562 1571 E-mail: stds-info@ieee.org Web: www.standards.ieee.org



Commission Electrotechnique Internationale
International Electrotechnical Commission
Международная Электротехническая Комиссия



CONTENTS

FOREWORD	19
IEEE Introduction.....	23
1. Overview.....	25
1.1 Objectives of this standard.....	25
1.2 Conventions used in this standard.....	25
1.3 Syntactic description.....	26
1.4 Contents of this standard.....	26
1.5 Header file listings	28
1.6 Examples.....	29
1.7 Prerequisites.....	29
2. Lexical conventions	30
2.1 Lexical tokens	30
2.2 White space.....	30
2.3 Comments	30
2.4 Operators.....	30
2.5 Numbers.....	30
2.5.1 Integer constants	31
2.5.2 Real constants	34
2.5.3 Conversion	34
2.6 Strings.....	34
2.6.1 String variable declaration.....	35
2.6.2 String manipulation.....	35
2.6.3 Special characters in strings.....	35
2.7 Identifiers, keywords, and system names	36
2.7.1 Escaped identifiers.....	36
2.7.2 Generated identifiers.....	37
2.7.3 Keywords.....	37
2.7.4 System tasks and functions.....	37
2.7.5 Compiler directives.....	38
2.8 Attributes.....	38
2.8.1 Examples.....	39
2.8.2 Syntax.....	40
3. Data types.....	44
3.1 Value set.....	44
3.2 Nets and variables	44
3.2.1 Net declarations	44
3.2.2 Variable declarations	46
3.3 Vectors.....	47
3.3.1 Specifying vectors.....	47
3.3.2 Vector net accessibility	48
3.4 Strengths	48
3.4.1 Charge strength.....	48
3.4.2 Drive strength.....	48
3.5 Implicit declarations.....	49
3.6 Net initialization.....	49
3.7 Net types	49
3.7.1 Wire and tri nets.....	49
3.7.2 Wired nets	50

3.7.3	Trireg net.....	50
3.7.4	Tri0 and tri1 nets.....	54
3.7.5	Supply nets.....	55
3.8	regs.....	55
3.9	Integers, reals, times, and realsimes.....	55
3.9.1	Operators and real numbers.....	56
3.9.2	Conversion.....	56
3.10	Arrays.....	57
3.10.1	Net arrays.....	57
3.10.2	reg and variable arrays.....	57
3.10.3	Memories.....	57
3.11	Parameters.....	58
3.11.1	Module parameters.....	59
3.11.2	Local parameters—localparam.....	60
3.11.3	Specify parameters.....	61
3.12	Name spaces.....	62
4.	Expressions.....	64
4.1	Operators.....	64
4.1.1	Operators with real operands.....	65
4.1.2	Binary operator precedence.....	66
4.1.3	Using integer numbers in expressions.....	67
4.1.4	Expression evaluation order.....	67
4.1.5	Arithmetic operators.....	68
4.1.6	Arithmetic expressions with regs and integers.....	69
4.1.7	Relational operators.....	70
4.1.8	Equality operators.....	70
4.1.9	Logical operators.....	71
4.1.10	Bit-wise operators.....	71
4.1.11	Reduction operators.....	72
4.1.12	Shift operators.....	73
4.1.13	Conditional operator.....	74
4.1.14	Concatenations.....	75
4.1.15	Event or.....	76
4.2	Operands.....	76
4.2.1	Vector bit-select and part-select addressing.....	76
4.2.2	Array and memory addressing.....	78
4.2.3	Strings.....	79
4.3	Minimum, typical, and maximum delay expressions.....	81
4.4	Expression bit lengths.....	83
4.4.1	Rules for expression bit lengths.....	83
4.4.2	An example of an expression bit-length problem.....	84
4.4.3	Example of self-determined expressions.....	85
4.5	Signed expressions.....	86
4.5.1	Rules for expression types.....	86
4.5.2	Steps for evaluating an expression.....	86
4.5.3	Steps for evaluating an assignment.....	87
4.5.4	Handling X and Z in signed expressions.....	87
5.	Scheduling semantics.....	88
5.1	Execution of a model.....	88
5.2	Event simulation.....	88

5.3	The stratified event queue	88
5.4	The Verilog simulation reference model	89
5.4.1	Determinism	90
5.4.2	Nondeterminism	90
5.5	Race conditions	90
5.6	Scheduling implication of assignments	90
5.6.1	Continuous assignment	91
5.6.2	Procedural continuous assignment	91
5.6.3	Blocking assignment	91
5.6.4	Nonblocking assignment	91
5.6.5	Switch (transistor) processing	91
5.6.6	Port connections	92
5.6.7	Functions and tasks	92
6.	Assignments	93
6.1	Continuous assignments	93
6.1.1	The net declaration assignment	94
6.1.2	The continuous assignment statement	94
6.1.3	Delays	96
6.1.4	Strength	96
6.2	Procedural assignments	97
6.2.1	Variable declaration assignment	97
6.2.2	Variable declaration syntax	98
7.	Gate and switch level modeling	99
7.1	Gate and switch declaration syntax	99
7.1.1	The gate type specification	101
7.1.2	The drive strength specification	101
7.1.3	The delay specification	102
7.1.4	The primitive instance identifier	102
7.1.5	The range specification	102
7.1.6	Primitive instance connection list	103
7.2	and, nand, nor, or, xor, and xnor gates	105
7.3	buf and not gates	106
7.4	bufif1, bufif0, notif1, and notif0 gates	107
7.5	MOS switches	109
7.6	Bidirectional pass switches	110
7.7	CMOS switches	110
7.8	pullup and pulldown sources	111
7.9	Logic strength modeling	112
7.10	Strengths and values of combined signals	113
7.10.1	Combined signals of unambiguous strength	113
7.10.2	Ambiguous strengths: sources and combinations	114
7.10.3	Ambiguous strength signals and unambiguous signals	119
7.10.4	Wired logic net types	123
7.11	Strength reduction by nonresistive devices	126
7.12	Strength reduction by resistive devices	126
7.13	Strengths of net types	126
7.13.1	tri0 and tri1 net strengths	126
7.13.2	trireg strength	126
7.13.3	supply0 and supply1 net strengths	126

7.14	Gate and net delays	127
7.14.1	min:typ:max delays.....	128
7.14.2	trireg net charge decay.....	129
8.	User-defined primitives (UDPs)	131
8.1	UDP definition.....	131
8.1.1	UDP header.....	133
8.1.2	UDP port declarations.....	133
8.1.3	Sequential UDP initial statement	133
8.1.4	UDP state table	133
8.1.5	Z values in UDP.....	134
8.1.6	Summary of symbols	134
8.2	Combinational UDPs	135
8.3	Level-sensitive sequential UDPs	136
8.4	Edge-sensitive sequential UDPs	136
8.5	Sequential UDP initialization	137
8.6	UDP instances.....	139
8.7	Mixing level-sensitive and edge-sensitive descriptions.....	140
8.8	Level-sensitive dominance.....	141
9.	Behavioral modeling.....	142
9.1	Behavioral model overview	142
9.2	Procedural assignments.....	143
9.2.1	Blocking procedural assignments	143
9.2.2	The nonblocking procedural assignment.....	145
9.3	Procedural continuous assignments.....	148
9.3.1	The assign and deassign procedural statements.....	149
9.3.2	The force and release procedural statements	150
9.4	Conditional statement.....	151
9.4.1	If-else-if construct.....	152
9.5	Case statement	154
9.5.1	Case statement with don't-cares	157
9.5.2	Constant expression in case statement.....	157
9.6	Looping statements.....	158
9.7	Procedural timing controls.....	160
9.7.1	Delay control.....	161
9.7.2	Event control.....	162
9.7.3	Named events.....	162
9.7.4	Event or operator.....	163
9.7.5	Implicit event_expression list	164
9.7.6	Level-sensitive event control	165
9.7.7	Intra-assignment timing controls	166
9.8	Block statements	170
9.8.1	Sequential blocks	170
9.8.2	Parallel blocks.....	171
9.8.3	Block names.....	172
9.8.4	Start and finish times	172
9.9	Structured procedures	173
9.9.1	Initial construct	174
9.9.2	Always construct.....	174

10.	Tasks and functions.....	176
10.1	Distinctions between tasks and functions	176
10.2	Tasks and task enabling	176
10.2.1	Task declarations	177
10.2.2	Task enabling and argument passing	178
10.2.3	Task memory usage and concurrent activation.....	180
10.3	Functions and function calling.....	181
10.3.1	Function declarations	182
10.3.2	Returning a value from a function	183
10.3.3	Calling a function.....	184
10.3.4	Function rules.....	184
10.3.5	Use of constant functions.....	185
11.	Disabling of named blocks and tasks.....	187
12.	Hierarchical structures	190
12.1	Modules.....	190
12.1.1	Top-level modules	192
12.1.2	Module instantiation	192
12.1.3	Generated instantiation	194
12.2	Overriding module parameter values.....	204
12.2.1	defparam statement	206
12.2.2	Module instance parameter value assignment	207
12.2.3	Parameter dependence	209
12.3	Ports	209
12.3.1	Port definition	209
12.3.2	List of ports.....	209
12.3.3	Port declarations.....	210
12.3.4	List of ports declarations.....	212
12.3.5	Connecting module instance ports by ordered list.....	212
12.3.6	Connecting module instance ports by name	213
12.3.7	Real numbers in port connections.....	214
12.3.8	Connecting dissimilar ports	215
12.3.9	Port connection rules.....	215
12.3.10	Net types resulting from dissimilar port connections	216
12.3.11	Connecting signed values via ports.....	217
12.4	Hierarchical names	217
12.5	Upwards name referencing	220
12.6	Scope rules	222
13.	Configuring the contents of a design	224
13.1	Introduction.....	224
13.1.1	Library notation	224
13.1.2	Basic configuration elements.....	225
13.2	Libraries	225
13.2.1	Specifying libraries - the library map file.....	225
13.2.2	Using multiple library mapping files	227
13.2.3	Mapping source files to libraries.....	227
13.3	Configurations.....	227
13.3.1	Basic configuration syntax.....	227
13.3.2	Hierarchical configurations.....	230

13.4	Using libraries and configs	231
13.4.1	Precompiling in a single-pass use-model.....	231
13.4.2	Elaboration-time compiling in a single-pass use-model	231
13.4.3	Precompiling using a separate compilation tool	231
13.4.4	Command line considerations.....	231
13.5	Configuration examples	232
13.5.1	Default configuration from library map file	232
13.5.2	Using the default clause.....	232
13.5.3	Using the cell clause	233
13.5.4	Using the instance clause	233
13.5.5	Using a hierarchical config	233
13.6	Displaying library binding information	234
13.7	Library mapping examples	234
13.7.1	Using the command line to control library searching.....	234
13.7.2	File path specification examples	234
13.7.3	Resolving multiple path specifications	235
14.	Specify blocks	236
14.1	Specify block declaration.....	236
14.2	Module path declarations	237
14.2.1	Module path restrictions	238
14.2.2	Simple module paths.....	238
14.2.3	Edge-sensitive paths.....	239
14.2.4	State-dependent paths	240
14.2.5	Full connection and parallel connection paths.....	244
14.2.6	Declaring multiple module paths in a single statement	245
14.2.7	Module path polarity.....	246
14.3	Assigning delays to module paths.....	247
14.3.1	Specifying transition delays on module paths	248
14.3.2	Specifying x transition delays.....	249
14.3.3	Delay selection.....	250
14.4	Mixing module path delays and distributed delays.....	251
14.5	Driving wired logic.....	252
14.6	Detailed control of pulse filtering behavior.....	253
14.6.1	Specify block control of pulse limit values.....	254
14.6.2	Global control of pulse limit values.....	255
14.6.3	SDF annotation of pulse limit values.....	255
14.6.4	Detailed pulse control capabilities	256
15.	Timing checks	262
15.1	Overview.....	262
15.2	Timing checks using a stability window.....	265
15.2.1	\$setup	266
15.2.2	\$hold	266
15.2.3	\$setuphold.....	267
15.2.4	\$removal	269
15.2.5	\$recovery.....	270
15.2.6	\$recrém	271
15.3	Timing checks for clock and control signals	273
15.3.1	\$skew	274
15.3.2	\$timeskew	275
15.3.3	\$fullskew	277

15.3.4	\$width	279
15.3.5	\$period	280
15.3.6	\$nochange	281
15.4	Edge-control specifiers	283
15.5	Notifiers: user-defined responses to timing violations	284
15.5.1	Requirements for accurate simulation	286
15.5.2	Conditions in negative timing checks	288
15.5.3	Notifiers in negative timing checks	290
15.5.4	Option behavior	290
15.6	Enabling timing checks with conditioned events.....	290
15.7	Vector signals in timing checks	291
15.8	Negative timing checks.....	292
16.	Backannotation using the Standard Delay Format (SDF).....	294
16.1	The SDF annotator.....	294
16.2	Mapping of SDF constructs to Verilog.....	294
16.2.1	Mapping of SDF delay constructs to Verilog declarations.....	294
16.2.2	Mapping of SDF timing check constructs to Verilog.....	296
16.2.3	SDF annotation of specparams	297
16.2.4	SDF annotation of interconnect delays	298
16.3	Multiple annotations	299
16.4	Multiple SDF files.....	300
16.5	Pulse limit annotation	300
16.6	SDF to Verilog delay value mapping.....	301
17.	System tasks and functions.....	302
17.1	Display system tasks	302
17.1.1	The display and write tasks.....	303
17.1.2	Strobed monitoring	310
17.1.3	Continuous monitoring	311
17.2	File input-output system tasks and functions.....	311
17.2.1	Opening and closing files.....	311
17.2.2	File output system tasks.....	313
17.2.3	Formatting data to a string	314
17.2.4	Reading data from a file.....	315
17.2.5	File positioning	319
17.2.6	Flushing output	319
17.2.7	I/O error status	319
17.2.8	Loading memory data from a file	320
17.2.9	Loading timing data from an SDF file.....	321
17.3	Timescale system tasks.....	322
17.3.1	\$printtimescale.....	322
17.3.2	\$timeformat.....	323
17.4	Simulation control system tasks.....	326
17.4.1	\$finish	326
17.4.2	\$stop.....	326
17.5	PLA modeling system tasks.....	327
17.5.1	Array types.....	327
17.5.2	Array logic types.....	328
17.5.3	Logic array personality declaration and loading.....	328
17.5.4	Logic array personality formats.....	328
17.6	Stochastic analysis tasks	331

17.6.1	\$q_initialize.....	331
17.6.2	\$q_add.....	332
17.6.3	\$q_remove.....	332
17.6.4	\$q_full.....	332
17.6.5	\$q_exam.....	332
17.6.6	Status codes.....	333
17.7	Simulation time system functions.....	333
17.7.1	\$time.....	333
17.7.2	\$stime.....	334
17.7.3	\$realtime.....	334
17.8	Conversion functions.....	335
17.9	Probabilistic distribution functions.....	336
17.9.1	\$random function.....	336
17.9.2	\$dist_functions.....	337
17.9.3	Algorithm for probabilistic distribution functions.....	338
17.10	Command line input.....	345
17.10.1	\$test\$plusargs (string).....	346
17.10.2	\$value\$plusargs (user_string, variable).....	346
18.	Value change dump (VCD) files.....	349
18.1	Creating the four state value change dump file.....	349
18.1.1	Specifying the name of the dump file (\$dumpfile).....	349
18.1.2	Specifying the variables to be dumped (\$dumpvars).....	350
18.1.3	Stopping and resuming the dump (\$dumpoff/\$dumpon).....	351
18.1.4	Generating a checkpoint (\$dumpall).....	352
18.1.5	Limiting the size of the dump file (\$dumplimit).....	352
18.1.6	Reading the dump file during simulation (\$dumpflush).....	353
18.2	Format of the four state VCD file.....	354
18.2.1	Syntax of the four state VCD file.....	354
18.2.2	Formats of variable values.....	356
18.2.3	Description of keyword commands.....	357
18.2.4	Four state VCD file format example.....	363
18.3	Creating the extended value change dump file.....	364
18.3.1	Specifying the dumpfile name and the ports to be dumped (\$dumpports).....	364
18.3.2	Stopping and resuming the dump (\$dumpportsoff/\$dumpportson).....	365
18.3.3	Generating a checkpoint (\$dumpportsall).....	366
18.3.4	Limiting the size of the dump file (\$dumpportslimit).....	366
18.3.5	Reading the dump file during simulation (\$dumpportsflush).....	367
18.3.6	Description of keyword commands.....	367
18.3.7	General rules for extended VCD system tasks.....	368
18.4	Format of the extended VCD file.....	368
18.4.1	Syntax of the extended VCD file.....	368
18.4.2	Extended VCD node information.....	370
18.4.3	Value changes.....	372
18.4.4	Extended VCD file format example.....	373
19.	Compiler directives.....	375
19.1	`celldefine and `endcelldefine.....	375
19.2	`default_nettype.....	375
19.3	`define and `undef.....	376
19.3.1	`define.....	376
19.3.2	`undef.....	378

19.4	`ifdef, `else, `elsif, `endif, `ifndef	378
19.5	`include	382
19.6	`resetall.....	382
19.7	`line	383
19.8	`timescale	383
19.9	`unconnected_drive and `nounconnected_drive	385
20.	PLI overview.....	386
20.1	PLI purpose and history (informative).....	386
20.2	User-defined system task or function names	386
20.3	User-defined system task or function types	387
20.4	Overriding built-in system task and function names	387
20.5	User-supplied PLI applications.....	387
20.6	PLI interface mechanism	387
20.7	User-defined system task and function arguments	388
20.8	PLI include files.....	388
20.9	PLI Memory Restrictions.....	388
21.	PLI TF and ACC interface mechanism.....	389
21.1	User-supplied PLI applications.....	389
21.1.1	The sizetf class of PLI applications.....	389
21.1.2	The checktf class of PLI applications.....	389
21.1.3	The calltf class of PLI applications.....	390
21.1.4	The misttf class of PLI applications.....	390
21.1.5	The consumer class of PLI applications.....	390
21.2	Associating PLI applications to a class and system task/function name	390
21.3	PLI application arguments.....	391
21.3.1	The data C argument.....	391
21.3.2	The reason C argument.....	391
21.3.3	The paramvc C argument.....	392
22.	Using ACC routines.....	393
22.1	ACC routine definition.....	393
22.2	The handle data type.....	393
22.3	Using ACC routines.....	394
22.3.1	Header files.....	394
22.3.2	Initializing ACC routines.....	394
22.3.3	Exiting ACC routines.....	394
22.4	List of ACC routines by major category.....	394
22.4.1	Fetch routines.....	395
22.4.2	Handle routines.....	396
22.4.3	Next routines.....	397
22.4.4	Modify routines.....	399
22.4.5	Miscellaneous routines.....	399
22.4.6	VCL routines.....	400
22.5	Accessible objects.....	400
22.5.1	ACC routines that operate on module instances	402
22.5.2	ACC routines that operate on module ports	402
22.5.3	ACC routines that operate on bits of a port	403
22.5.4	ACC routines that operate on module paths or data paths	403
22.5.5	ACC routines that operate on intermodule paths	404

22.5.6	ACC routines that operate on top-level modules	404
22.5.7	ACC routines that operate on primitive instances	404
22.5.8	ACC routines that operate on primitive terminals	405
22.5.9	ACC routines that operate on nets	405
22.5.10	ACC routines that operate on reg types	406
22.5.11	ACC routines that operate on integer, real, and time variables	406
22.5.12	ACC routines that operate on named events	406
22.5.13	ACC routines that operate on parameters and specparams	407
22.5.14	ACC routines that operate on timing checks	407
22.5.15	ACC routines that operate on timing check terminals	407
22.5.16	ACC routines that operate on user-defined system task/function arguments	408
22.6	ACC routine types and fulltypes	408
22.7	Error handling	411
22.7.1	Suppressing error messages	412
22.7.2	Enabling warnings	412
22.7.3	Testing for errors	412
22.7.4	Example	412
22.7.5	Exception values	413
22.8	Reading and writing delay values	413
22.8.1	Number of delays for Verilog HDL objects	414
22.8.2	ACC routine configuration	414
22.8.3	Determining the number of arguments for ACC delay routines	415
22.9	String handling	419
22.9.1	ACC routines share an internal string buffer	419
22.9.2	String buffer reset	420
22.9.3	Preserving string values	421
22.9.4	Example of preserving string values	421
22.10	Using VCL ACC routines	421
22.10.1	VCL objects	422
22.10.2	The VCL record definition	422
22.10.3	Effects of acc_initialize() and acc_close() on VCL consumer routines	425
22.10.4	An example of using VCL ACC routines	425
23.	ACC routine definitions	428
23.1	acc_append_delays()	429
23.2	acc_append_pulse()	433
23.3	acc_close()	435
23.4	acc_collect()	436
23.5	acc_compare_handles()	438
23.6	acc_configure()	439
23.7	acc_count()	448
23.8	acc_fetch_argc()	449
23.9	acc_fetch_argv()	450
23.10	acc_fetch_attribute()	452
23.11	acc_fetch_attribute_int()	456
23.12	acc_fetch_attribute_str()	457
23.13	acc_fetch_defname()	458
23.14	acc_fetch_delay_mode()	459
23.15	acc_fetch_delays()	461
23.16	acc_fetch_direction()	465
23.17	acc_fetch_edge()	466
23.18	acc_fetch_fullname()	468
23.19	acc_fetch_fulltype()	470

23.20 acc_fetch_index()	473
23.21 acc_fetch_location()	475
23.22 acc_fetch_name()	477
23.23 acc_fetch_paramtype()	479
23.24 acc_fetch_paramval()	480
23.25 acc_fetch_polarity()	482
23.26 acc_fetch_precision()	483
23.27 acc_fetch_pulse()	484
23.28 acc_fetch_range()	487
23.29 acc_fetch_size()	488
23.30 acc_fetch_tfarg(), acc_fetch_itfarg()	489
23.31 acc_fetch_tfarg_int(), acc_fetch_itfarg_int()	491
23.32 acc_fetch_tfarg_str(), acc_fetch_itfarg_str()	492
23.33 acc_fetch_timescale_info()	493
23.34 acc_fetch_type()	495
23.35 acc_fetch_type_str()	497
23.36 acc_fetch_value()	498
23.37 acc_free()	503
23.38 acc_handle_by_name()	504
23.39 acc_handle_calling_mode_m()	506
23.40 acc_handle_condition()	507
23.41 acc_handle_conn()	508
23.42 acc_handle_datapath()	509
23.43 acc_handle_hiconn()	510
23.44 acc_handle_interactive_scope()	512
23.45 acc_handle_loconn()	513
23.46 acc_handle_modpath()	514
23.47 acc_handle_notifier()	516
23.48 acc_handle_object()	517
23.49 acc_handle_parent()	519
23.50 acc_handle_path()	520
23.51 acc_handle_pathin()	521
23.52 acc_handle_pathout()	522
23.53 acc_handle_port()	523
23.54 acc_handle_scope()	525
23.55 acc_handle_simulated_net()	526
23.56 acc_handle_tchk()	528
23.57 acc_handle_tchkarg1()	532
23.58 acc_handle_tchkarg2()	534
23.59 acc_handle_terminal()	535
23.60 acc_handle_tfarg(), acc_handle_itfarg()	536
23.61 acc_handle_tnst()	538
23.62 acc_initialize()	539
23.63 acc_next()	540
23.64 acc_next_bit()	544
23.65 acc_next_cell()	546
23.66 acc_next_cell_load()	547
23.67 acc_next_child()	549
23.68 acc_next_driver()	550
23.69 acc_next_hiconn()	551
23.70 acc_next_input()	553
23.71 acc_next_load()	555
23.72 acc_next_loconn()	557
23.73 acc_next_modpath()	558