

## Contents

1.	Overview.....	1
	1.1 Scope.....	1
	1.2 Purpose.....	1
	1.3 Verification environments .....	2
	1.4 Basic concepts relating to this standard.....	3
	1.5 Conventions used .....	8
	1.6 Use of color in this standard .....	10
	1.7 Contents of this standard.....	10
2.	Normative references .....	13
3.	Definitions, acronyms, and abbreviations.....	13
	3.1 Definitions .....	13
	3.2 Acronyms and abbreviations .....	15
4.	e basics .....	17
	4.1 Lexical conventions .....	17
	4.2 Syntactic elements .....	24
	4.3 Struct hierarchy and name resolution .....	30
	4.4 Ranges.....	36
	4.5 Operator precedence .....	37
	4.6 Evaluation order of expressions.....	38
	4.7 Bitwise operators .....	38
	4.8 Boolean operators .....	40
	4.9 Arithmetic operators .....	42
	4.10 Comparison operators .....	43
	4.11 String matching.....	47
	4.12 Extraction and concatenation operators.....	49
	4.13 Scalar modifiers .....	53
	4.14 Parentheses.....	54
	4.15 list.method() .....	54
	4.16 Special-purpose operators.....	55
5.	Data types .....	59
	5.1 e data types.....	59
	5.2 Untyped expressions .....	64
	5.3 Assignment rules.....	65
	5.4 Precision rules for numeric operations .....	68
	5.5 Automatic type casting .....	70
	5.6 Defining and extending scalar types.....	70
	5.7 Type-related constructs.....	73
6.	Structs, subtypes, and fields.....	79
	6.1 Structs overview .....	79
	6.2 Defining structs: struct .....	79
	6.3 Extending structs: extend type .....	80

6.4	Restrictions on inheritance.....	81
6.5	Extending subtypes.....	81
6.6	Creating subtypes with when.....	81
6.7	Extending when subtypes.....	83
6.8	Defining fields: field.....	84
6.9	Defining list fields.....	85
6.10	Projecting list of fields.....	88
6.11	Defining attribute fields.....	88
7.	Units.....	91
7.1	Overview.....	91
7.2	Defining units and fields of type unit.....	93
7.3	Unit attributes.....	97
7.4	Predefined methods of any_unit.....	98
7.5	Unit-related predefined methods of any_struct.....	100
7.6	Unit-related predefined routines.....	102
8.	<i>e</i> ports.....	105
8.1	Introduction to <i>e</i> ports.....	105
8.2	Using simple ports.....	106
8.3	Using buffer ports.....	108
8.4	Using event ports.....	109
8.5	Using method ports.....	110
8.6	Defining and referencing ports.....	112
8.7	Port attributes.....	117
8.8	Buffer port methods.....	130
8.9	MVL methods for simple ports.....	132
8.10	Global MVL routines.....	138
8.11	Comparative analysis of ports and tick access.....	141
9.	Constraints and generation.....	143
9.1	Types of constraints.....	143
9.2	Generation concepts.....	143
9.3	Defining constraints.....	157
9.4	Invoking generation.....	164
10.	Events.....	167
10.1	Causes of events.....	167
10.2	Scope of events.....	167
10.3	Defining and emitting named events.....	167
10.4	Predefined events.....	168
11.	Temporal expressions.....	171
11.1	Overview.....	171
11.2	Temporal operators and constructs.....	174
11.3	Success and failure of a temporal expression.....	189
12.	Temporal struct members.....	191

12.1 on .....	191
12.2 expect   assume .....	192
13. Time-consuming actions.....	193
13.1 Synchronization actions.....	193
13.2 Concurrency actions .....	194
13.3 State machines .....	196
14. Coverage constructs.....	201
14.1 Defining coverage groups: cover .....	201
14.2 Defining basic coverage items: item .....	203
14.3 Defining cross coverage items: cross .....	207
14.4 Defining transition coverage items: transition .....	209
14.5 Extending coverage groups: cover ... using also ... is also .....	211
14.6 Extending coverage items: item ... using also .....	212
14.7 Coverage API.....	212
14.8 Coverage methods for the covers struct.....	217
15. Macros .....	223
15.1 Syntax overview .....	223
15.2 Defining a macro .....	223
15.3 Match expression structure .....	224
15.4 Interpretation of match expressions.....	225
15.5 Replacement code.....	226
16. Print, checks, and error handling .....	229
16.1 print .....	229
16.2 Handling DUT errors .....	229
16.3 Handling user errors.....	234
16.4 Handling programming errors: assert .....	236
17. Methods .....	237
17.1 Rules for defining and extending methods .....	237
17.2 Invoking methods .....	245
17.3 Parameter passing .....	248
17.4 Using the C interface .....	250
18. Creating and modifying <i>e</i> variables .....	253
18.1 About <i>e</i> variables .....	253
18.2 var .....	253
18.3 = .....	254
18.4 op= .....	254
18.5 <=.....	255
19. Packing and unpacking .....	257
19.1 Basic packing.....	257
19.2 Predefined pack options.....	260

19.3	Customizing pack options.....	261
19.4	Packing and unpacking specific types .....	261
19.5	Implicit packing and unpacking.....	266
20.	Control flow actions.....	269
20.1	Conditional actions .....	269
20.2	Iterative actions .....	271
20.3	File iteration actions.....	275
20.4	Actions for controlling the program flow .....	276
21.	Importing and preprocessor directives.....	279
21.1	Importing e modules .....	279
21.2	#ifdef, #ifndef .....	280
21.3	#define .....	281
21.4	#undef .....	282
22.	Encapsulation constructs.....	283
22.1	package: package-name .....	283
22.2	package: type-declaration .....	283
22.3	package   protected   private: struct-member .....	284
22.4	Scope operator (::) .....	285
23.	Simulation-related constructs .....	287
23.1	force .....	287
23.2	release .....	287
23.3	Tick access: 'hdl-pathname' .....	288
23.4	simulator_command().....	288
23.5	stop_run() .....	289
24.	Messages.....	291
24.1	Overview.....	291
24.2	The message model.....	291
24.3	message and messagef .....	292
24.4	Message loggers.....	294
24.5	Configuring message loggers with constraints .....	295
24.6	Messaging procedural interface (PI).....	296
24.7	Examples.....	304
25.	Sequences.....	307
25.1	Overview.....	307
25.2	Sequence statement .....	309
25.3	do sequence action .....	311
25.4	Sequence struct types and members .....	312
25.5	BFM-driver-sequence flow diagrams .....	317
26.	List pseudo-methods library .....	321
26.1	Pseudo-methods overview .....	321

26.2	Using list pseudo-methods .....	321
26.3	Pseudo-methods to modify lists .....	321
26.4	General list pseudo-methods .....	330
26.5	Math and logic pseudo-methods .....	344
26.6	List CRC pseudo-methods .....	346
26.7	Keyed list pseudo-methods .....	348
27.	Predefined methods library .....	351
27.1	Predefined methods of sys .....	351
27.2	Predefined methods of any_struct .....	351
27.3	Methods and predefined attributes of unit any_unit .....	354
27.4	Pseudo-methods .....	354
27.5	Coverage methods .....	356
28.	Predefined routines library .....	357
28.1	Deep copy and compare routines .....	357
28.2	Arithmetic routines .....	360
28.3	bitwise_op() .....	364
28.4	get_all_units() .....	365
28.5	String routines .....	365
28.6	Output routines .....	373
28.7	Operating system interface routines .....	375
28.8	set_config() .....	378
29.	Predefined file routines library .....	379
29.1	File names and search paths .....	379
29.2	File handles .....	379
29.3	Low-level file methods .....	379
29.4	General file routines .....	384
29.5	Reading and writing structs .....	390
30.	Reflection API .....	395
30.1	Introduction .....	395
30.2	Type information .....	396
30.3	Aspect information .....	403
30.4	Value query and manipulation .....	408
31.	Predefined resource sharing control structs .....	413
31.1	Semaphore methods .....	413
31.2	How to use the semaphore struct .....	414
32.	Type constraints .....	419
32.1	keep type .....	420
32.2	Type constraints and struct fields .....	422
32.3	Type constraints and list fields .....	423
32.4	Type constraints and like subtypes .....	424
33.	Real data types .....	427

33.1	Real data type usage .....	427
33.2	Real literals .....	427
33.3	Real constants .....	427
33.4	Conversion between real and integer data types.....	428
33.5	Conversions using the as_a() operator.....	429
33.6	Real data type precision, data conversion, and sign extension.....	429
33.7	Packing values for real types .....	429
33.8	Printing real values .....	430
33.9	Arithmetic routines supporting real type .....	430
33.10	Random routines .....	431
33.11	C interface macros that support real type objects .....	431
33.12	Real type limitations .....	431
34.	Template types .....	433
34.1	Defining a template type .....	433
34.2	Instantiating a template type .....	435
34.3	Template types and reflection.....	436
34.4	Template errors .....	437
34.5	Limitations .....	438
34.6	Templates vs. macros.....	438
Annex A	.....	441
Bibliography	.....	441
Annex B	.....	443
Source code serialization	.....	443
Annex C	.....	451
Comparison of when and like inheritance	.....	451
Annex D	.....	459
Name spaces	.....	459
Annex E	.....	467
Reflection API examples	.....	467