

Index

IPC
see interprocess communication

Symbols

! 181
-- 71
cycle delay 110, 206, 377
\$ 39, 345–346
\$bits 307
\$cast 57, 62, 286, 288, 293, 307
\$clog2 30
\$dist_exponential 197
\$dist_normal 197
\$dist_poisson 197
\$dist_uniform 197
\$error 119
\$exit 112–113
\$fatal 119
\$fclose 72
\$feof 42, 72
\$finish 112–113, 430
\$fopen 42, 72
\$fscanf 42, 72–73
\$get_coverage 362
\$info 119
\$isunknown 29, 120
\$monitor 374
\$psprintf 64
\$random 197, 219
\$realtime 84–85
\$root 116
\$sformat 64
\$sformatf 64
\$size 32, 37, 43
\$strobe 374
\$system 456
\$test\$plusargs 392–393
\$time 29, 84–85
\$timeformat 29, 84–85
\$typename 307, 312
\$unit 116, 421, 452
\$urandom 197, 204, 253
\$urandom_range 43, 196–197, 204, 217
\$value\$plusargs 314, 392–393, 456
\$warning 119
%0t 29
%p 31–32, 43, 312
%t 84
++ 71
+= 71–72
+ntb_random_seed 226, 392
:: 56, 144, 146, 252
<< 57
<<-> 188, 192
-= 71
-> 186, 190–191, 194
> 192
>> 57–59, 163
?: operator 34, 45
[\$] 39
[] 37
^= 71, 76, 78
__FILE__ 176
__LINE__ 176
`define 51
'include 55, 95

`'SV_RAND_CHECK` 176

Numerics

2-state types 29, 420, 424, 427, 430

4-state types 29, 428–430

A

abstract class 297–298, 300–301, 314, 317–318

Accellera xxxii

accessor methods 164

Active region 103–105, 108, 114

agent 18, 165

AHB 130

always block 232

always block in programs 113

and method 43

anonymous enumerated type 59

API xxxiii

Application Programming Interface see API

arguments

default value 77–78

sticky 75

task and function 74

type 75

array

assignment 38

associative 40–44, 49

compare 34

constraint 205

copy 34

dynamic 37–38, 40, 48–49

fixed-size 30, 38, 40, 48–49

handle 157

literal 31, 37, 39, 42

locator methods 44

methods 43

multi-dimensional 30, 32–33, 38, 440

open 440–441

packed 35–36

queue 38, 40, 48–49

reduction methods 43

slice 34

unpacked 31, 36

assertion

concurrent 120

coverage 330

immediate 118–119, 288

in an interface 381

assignment pattern 31

associative array 40–44, 49, 52

at_least option 334, 359

ATM router 120–123, 125–127

atomic stimulus generation 213

auto_bin_max 341–342, 349, 357

automatic 76, 82–83, 105, 139–140, 143, 239

in threads 238

aval 428, 430

B

backdoor load 117

base class 278–279

BaseTr class 297, 306–307

begin...end 74, 232

optional in tasks and functions 74

Bergeron, Janick xxxvii, 4, 387, 459

BFM 15

bidirectional constraint see constraint bidirectional

bidirectional signal 110

binsof 353, 355

bit data type 29, 423, 434

bit streaming

see streaming operator

blueprint pattern 281

BNF 215

bounded mailbox 258–259

break 72

Bus Functional Model see BFM

bval 428, 430

byte data type 29, 422–423, 434

C

calc_csm method 277–278, 289

callback

coverage 337

creation 300

inject disturbance 299

scoreboard 302

usage 301

case 73

cast 56, 62, 65, 73

function return value 73

Cerny, Eduard 459

- chandle data type 423, 425–427, 434
- char 423
- checker 18, 165
- class 134, 136
- class constructor 138
- class scope resolution operator 146, 252
- clock generator 113–114
- clocking block 100–101, 105, 110–111, 125–126, 367
- Cohen, Ben 459
- command layer 17
- comment
 - covergroup 358
- comment option 335, 358
- compilation unit 115
- composition 275, 290–292, 294
- compound assignment 71–72, 76
- concatenation
 - bits 58
 - string 63
- concurrent assertion 120
- conditional operator 34, 45
- Config class 268
- configuration database 311
- const 78
- const type 63, 76, 116, 178
- constrained 8
- constrained-random test
 - see CRT
- constraint
 - array 205
 - bidirectional 186, 190
 - block 175
 - dist 180–181, 198, 361
 - implication 190
 - in extended class 286
 - inside 178, 181–185, 204–205
 - solve...before 191–192, 361, 398
- constraint_mode 193–195, 282
- constructor 138–139, 279, 297
- consumer 246
- containment 151
- context 448
- continue 72
- copy
 - deep 161
 - method 160–161, 282–283, 295–296
 - object 158, 295
 - shallow 160

- coverage event 337
- covergroup
 - comment 358
 - embedded 336
 - generic 356–357
 - option 357
 - sample 333, 337–339
 - trigger 337
- coverpoint 333, 336–337, 339–343, 345–353, 355–360
- cross coverage 350–356
- cross module reference 116, 372, 378
- cross_num_print_missing option 359
- CRT 4, 8–9, 171, 198, 325
- Cummings, Cliff xxxvii, 102, 104, 459
- cyclic random 175

D

- data type
 - bit 29
 - byte 29
 - chandle 423, 425–427, 434
 - int 29
 - integer 27
 - logic 28
 - longint 29
 - real 27
 - reg 27–28
 - shortint 29
 - time 27
 - wire 28
- DDR 100
- deallocation 141
- decrement 71
- default clocking block 110
- default coverage value 345, 347, 352–353
- default statement 100
- default value 77–78
- delete method 37, 39, 41, 50
- derived class 279
- design pattern 317
- Design Under Test
 - see DUT
- Direct Programming Interface
 - see DPI
- disable 233, 242–245
- disable fork 243–244
- disable label 244–245

display method 143–144, 277–278, 287
 dist constraint 180–181, 198, 361
 do...while 41, 61, 72, 176
 domain 341
 double data rate clock 100
 double data type 422–423
 double precision floating point 29
 downcasting 286
 DPI 419–457
 DPI-C 420, 427
 driver 18, 165
 Driver class 377, 381
 DUT 3, 90
 dynamic array 37–38, 40, 48–49
 dynamic cast
 see \$cast
 dynamic threads 237

E

e Reuse Methodology
 see eRM
 embedded covergroup 336
 enum 59–62
 enumerated types 59
 see enum 59
 enumerated values 60
 enumeration 59
 environment 19, 165
 Environment class 165
 equivalence operator 188, 192
 eRM xxxii
 event 246–251, 266
 event triggered 246–251, 263
 exists 42, 50
 expression width 64
 extended class 278–279
 extern
 see external routine declaration
 external constraint 201–202
 external routine declaration 144–145,
 268, 298

F

fabs routine 423
 file I/O 72
 file_exists method 421
 final block 112
 find_first method 45

find_index method 45–47
 find_last method 45
 find_last_index method 45
 first 41
 first method 41, 49, 61–62
 fixed-size array 30, 38, 40, 48–49
 float data type 422
 for loop 32, 41, 61, 71, 199
 force design signals 117
 foreach constraint 205, 209–210, 398
 foreach loop 32–33, 37, 39, 42
 fork...join 232–233
 fork...join_any 232, 235
 fork...join_none 232, 234, 236–237, 239–
 240
 four-state types 29
 see 4-state types
 function 73
 arguments 74
 functional coverage 19, 460
 using callbacks 303
 functional layer 18

G

garbage collection 141–142
 Generator 19
 generator 165
 Generator class 236, 250, 252, 257, 281,
 283, 306–307
 get method 253–256, 263, 265, 267
 get_coverage 362
 get_inst_coverage 362
 getc method 63–64
 goal option 360

H

handle 136–137, 141
 array 157
 Haque, Faisal 459
 Hardware Description Language
 see HDL
 Hardware Verification Language 2
 see HVL
 HDL 2, 85
 histogram 182
 hook 282, 299
 HVL xxxiii, 2, 13

I

IEEE 1800
 see LRM
 if constraint 187–188
 iff 120, 346
 ignore_bins 348–349, 353
 illegal_bins 349
 immediate assertion 118–119, 288
 Implication 186
 implication operator 186, 192
 implicit port connection 114
 import 55, 420–421
 incl.h 422
 increment 71
 inheritance 276, 290
 initial 31
 initialization in declaration 82
 in-line constraint 201
 inout
 argument type 75
 port type 128
 input
 argument type 75
 port type 128
 insert queue 39
 inside 73, 178, 182
 inside constraint 178, 181–185, 204–205
 instance 138
 instantiation 138
 int data type 29, 422–423, 434
 integer data type 27, 29, 53
 interface 93, 99, 115, 371
 connecting to port 95
 procedural code 381
 virtual 365–366, 368–370, 372–381
 interprocess 2
 interprocess communication 231, 246,
 266
 intersect 353, 355
 io_printf 426, 440
 iterator argument 45

L

Language Reference Manual
 see LRM
 last method 61
 local 164, 304–305, 310
 logic 51, 93

logic data type 28–29, 99, 423, 434
 long long int 441
 longint data type 29, 422–423, 434
 LRM xxxi–xxxii, 49, 78, 102, 104, 111,
 177, 202, 231, 335, 354, 365, 459

M

macro 50
 macromodule 115
 Magellan 460
 mailbox 254, 259, 264–266
 bounded 258–259
 unbounded 258
 makes Jack a dull boy xxxvi
 malloc 138, 426
 max method 44
 method 136, 279
 virtual 288–290, 295–296
 min method 44
 modport 96, 125
 module 135
 monitor 18, 165
 multi-dimensional array 30, 32–33, 38,
 440

N

name function 184
 name method 59
 new
 constructor 138–140
 copying objects 158
 new function 139
 new[] operator 37, 140
 next method 41, 49, 61
 nonblocking assignment 94, 100, 102–
 103, 108, 459
 null 138, 141–142, 200
 num method 41, 257

O

object 136–137
 copy 158, 295
 deallocation 141
 Object-Oriented Programming
 see OOP
 object-oriented programming
 see OOP

Observed region 104
 OOP xxxi, 133–134, 275
 analogy badge 141
 analogy car 134
 analogy house 137
 terminology 136, 279
 open array 438
 open array see array open
 Open Verification Methodology
 see OVM
 OpenVera xxxii
 option 354
 at_least 334, 359
 auto_bin_max 341–342, 349, 357
 comment 335, 358
 cross_num_print_missing 359
 goal 360
 per_instance 358, 362
 weight 353–355, 357
 or method 43
 output
 argument type 75
 port type 128
 OVM xxxii, 4, 311

P

pack
 method 162–163, 442–443
 operation 57
 package 51, 55, 63, 116, 135, 148, 202
 packed array 35–36
 packed structure 54
 parameter 51, 63, 116
 parameterized
 class 304
 interface 379
 mailbox 255
 module 379
 virtual interface 379–381
 parent class 279
 peek method 255
 per_instance option 358, 362
 Perl hash array 41
 physical interfaces 365
 PLI 419, 426
 Plyant, Tim
 a good guy xxxvii
 polymorphism 289–290

pop_back method 39
 pop_front method 39
 port
 connecting to interface 95
 post_randomize 195–197, 212, 214
 post-decrement 71
 post-increment 71
 Postponed region 104–107
 pre_randomize 195–197, 204
 pre-decrement 71
 pre-increment 71
 prev method 49, 61
 primitive 116
 print class 308
 PRNG 177, 219, 221
 process 231
 producer 246
 product method 43
 program 82, 105, 116, 135, 148, 240,
 269–270
 program blocks
 single vs. multiple 105
 property 136, 279
 protected 310
 prototype 136, 279, 297
 proxy class 317–318
 pseudo-random number generator
 see PRNG
 public 164
 pure
 imported method 444–445
 virtual method 297–298, 301, 314
 virtual methods 318
 push_back method 45, 301
 push_front 39
 push_front method 39, 301
 put method 253, 255–256, 259–260, 263,
 265, 267
 putc method 64

Q

queue 38, 40, 48–49
 literal 39–40

R

Ramanathan, Meyyappan 460
 rand 175
 rand_mode 193, 199

randc 175, 184, 205, 210, 283
 randcase 217–218
 random seed 12–13, 172, 177, 219, 221, 226
 random stability 219, 221
 randomize function 175, 178
 randomize() with 194–195, 201
 randomize(null) 200
 Reactive region 104–105, 108
 real data type 27, 29, 177, 422–423, 434
 realtime data type 84–85
 ref
 argument type 78, 422
 port type 128
 reg data type 27–28
 region
 scheduling 103
 regions
 scheduling 104
 return 80
 reverse method 47
 Rich, Dave 459
 routine arguments 74
 run method 281, 283–284
 RVM 4

S

s_vpi_vecval 428
 sample method 333, 337–339
 scenario generation 213
 scenario layer 19
 scheduling region 103
 scheduling regions 104
 scope operator 56, 144
 scoreboard 18, 47–48, 165, 299, 302–303
 using callbacks 299
 seed
 see random seed
 semaphore 252–254, 266
 shortint data type 29, 422–423, 434
 shortreal data type 422–423, 434
 shuffle method 47
 signal layer 17
 signature 290
 signed data types 29
 sine function 445
 singleton class 308, 310, 318
 size function 205

size method 37, 41
 solve...before see constraint solve...before
 sort method 47
 sparse matrix 40
 specialization 305, 307
 start method 346
 stat method 421
 state variables 200
 static 31
 cast 56
 class 308, 314
 method 147, 314
 storage 82
 variable 145–147, 314
 stop method 346
 streaming operator 57–58, 162–163
 string concatenation 63
 string data type 63–64, 423, 434
 stringify 176, 320
 struct 52–54, 59
 subclass 279
 substr method 64
 sum 57
 sum method 43, 45–46, 206–209, 398
 superclass 279
 Sutherland, Stuart 387, 460
 SV_PACKED_DATA_NELEMS 428
 SV_RAND_CHECK 176, 288
 SVA 120, 330, 340, 459–460
 svBit 423–424, 426, 431–432
 svBitVecVal 423–424, 427, 430, 439
 svDimension 439
 svdpi.h 422, 424, 426, 439
 svGetArrayElemPtr1 441
 svGetArrayPtr 438–439
 svGetArrElemPtr 439
 svGetArrElemPtr1 439
 svGetArrElemPtr2 439–440
 svGetArrElemPtr3 439
 svGetNameFromScope 454
 svGetScope 453
 svGetScopeFromName 454
 svHigh 439–440
 svIncrement 439
 svLeft 439–440
 svLogic 423, 430
 svLogicVecVal 423, 428–430, 439
 svLow 439–440
 SVM 317

svOpenArrayHandle 438–441
 svRight 439–440
 svScope 454
 -svseed 226
 svSetScope 453
 svSize 439
 svSizeOfArray 439
 synchronous drive 108
 system() 456
 SystemC 419
 SystemVerilog
 see LRM
 SystemVerilog too many to list
 SystemVerilog Assertion
 see SVA
 SystemVerilog Methodology 317

T

task 73
 arguments 74
 template - see parameterized class
 test class 314
 test layer 19
 The Matrix 1, 460
 this 150–151
 thread 231, 234–238, 240–242
 time data type 27, 29, 84
 time literals 84
 time step 104
 timeprecision 84
 timescale 83–85, 116
 timeunit 84
 TLM analysis port 304
 tolower method 63
 toupper method 63–64
 transactor 165–166
 transition coverage 347
 tri data type 28
 triggered 248
 triggered method 246–251, 263
 try_get method 253, 257
 try_put method 255
 two-state types
 see 2-state types
 type conversion 56
 type_option 354
 typedef 50–51
 array 52, 81

associative array index 52
 class 154
 enum 60
 struct 53–54, 59
 union 54
 virtual interface 377

U

uint user-defined type 51, 208
 unbounded mailbox 258
 union 53–54
 unique method 44
 Universal Verification Methodology
 see UVM
 unpack
 method 162–163, 442–443
 operation 57
 unpacked array 31, 35–36
 unsigned 29, 51, 197, 203, 210
 unsized array see array open
 user defined type - see typedef
 UVM xxxii, 4–5, 22, 162, 214, 281, 301,
 307, 311, 317, 339

V

van der Schoot, Hans 460
 vc_hdrs.h 422
 Verification IP see VIP
 Verification Methodology Manual for Sys-
 temVerilog
 see VMM
 Verification Procedural Interface
 see VPI
 Verilog-1995 xxxi–xxxii, 27, 30, 34, 74–
 75, 82, 121, 197, 365, 379
 Verilog-2001 xxxi–xxxii, 28, 30, 34–35,
 63–64, 72, 74, 82, 91, 95, 459
 veriuser.h 426
 Vijayaraghavan, Srikanth 460
 VIP 15
 virtual
 class - see abstract class
 interface 311, 365–381
 memory 289
 method 280, 288–290, 295–297
 VMM xxxii, 4–5, 18, 22, 99, 214–215,
 281, 301, 309, 317, 330, 373, 459
 void

- data type 73, 423, 434
- function 73, 197
- VPI 419, 428, 430
- vpi_control 430
- vpiFinish 430

W

- wait 233, 237, 243, 247–252, 270
- wait fork 240–241, 250
- weight option 353–355, 357
- wildcard 348, 356
- wire data type 28, 99
- with 44–46
- wrap_up method 284

X

- XMR 372–373, 378
- xor method 43

