

目次

第1章 概要	1
1.1 SystemVerilog の歴史	1
1.2 SystemVerilog 概要	2
1.2.1 言語としての SystemVerilog	2
1.2.2 設計言語としての SystemVerilog	2
1.2.3 検証言語としての SystemVerilog	3
1.3 本書でのシンタックス記述法	4
1.4 SystemVerilog 言語ルール	4
1.5 コンパイルとエラーポレーション	6
1.6 宣言と定義	6
1.7 本書の対象者と目的	7
1.8 本書の構成	8
1.9 例題に関して	9
1.10 本書の記法	9
第2章 設計および検証のためのビルディングブロック	12
2.1 設計要素	12
2.2 モジュール	12
2.3 プログラム	14
2.4 インターフェース	15
2.5 チェッカー	17
2.6 パッケージ	17
2.7 ゲートとスイッチレベルのモデリング	18
2.8 primitive	19
2.9 configuration	20

2.10	コンパイルユニット	20
2.11	`timescale コンパイラディレクティブ	21
2.12	ガーベッジコレクション	23
2.12.1	automatic 変数	23
2.12.2	static 変数	23
第3章 データタイプ		24
3.1	データタイプとデータオブジェクト	24
3.2	logic 型	25
3.3	ネット型	26
3.4	変数	28
3.5	ネットと変数	30
3.6	4-state 型	31
3.7	2-state 型	32
3.8	integral データタイプ	33
3.9	real, shortreal と realtime	34
3.10	void 型	34
3.11	chandle 型	35
3.12	string データタイプ	35
3.13	イベントデータタイプ	38
3.14	typedef 文	39
3.15	enum データタイプ	40
3.16	定数	44
3.17	const 定数	47
3.18	cast オペレータ	48
3.19	\$cast ダイナミック型変換	49
3.20	便利な初期値設定	50
3.20.1	リテラルの拡張	50
3.20.2	インデックス指定	51
3.21	リファレンスポインタ	52
第4章 メンバーで構成されるデータタイプ		54
4.1	ストラクチャ	54
4.1.1	packed ストラクチャ	56
4.1.2	ストラクチャへの値の設定	57

4.2	ユニオン	58
4.2.1	packed ユニオン	59
4.2.2	タグ付きユニオン	60
4.3	packed アレイと unpacked アレイ	61
4.3.1	packed アレイ	61
4.3.2	unpacked アレイ	62
4.3.3	アレイの操作	63
4.3.4	packed アレイのアクセス	64
4.4	ダイナミックアレイ	65
4.4.1	ダイナミックアレイのメソッド	66
4.4.2	アレイのコピー	68
4.5	associative アレイ	70
4.5.1	associative アレイの概要	70
4.5.2	associative アレイの要素の登録	70
4.5.3	associative アレイのメソッド	72
4.5.4	associative アレイリテラル	72
4.6	キュー	74
4.6.1	キューの概要	74
4.6.2	キューの操作	75
4.6.3	キューを操作するメソッド	76
4.7	アレイ情報取得ファンクション	78
4.8	アレイ操作メソッド	78
4.8.1	アレイ検索メソッド	78
4.8.2	アレイ要素の順序を操作するメソッド	80
4.8.3	アレイを計算するメソッド	81
4.9	アレイの走査法	83
 第5章 クラス		 85
5.1	クラスの概要	85
5.2	シンタックス	87
5.3	クラスオブジェクト (クラスインスタンス)	89
5.4	クラスプロパティおよびメソッドへのアクセス	90
5.5	コンストラクタ	90
5.6	タイプ指定のコンストラクタ呼び出し	91
5.7	static クラスプロパティ	91
5.8	static クラスメソッド	94

5.9	this ハンドル	94
5.10	ハンドルのアレイ	95
5.11	クラスのコピー	97
5.12	クラス継承とサブクラス	98
5.13	\$cast	102
5.14	const クラスプロパティ	103
5.15	virtual メソッド	104
5.16	アブストラクトクラスと pure virtual メソッド	106
5.17	クラススコープオペレータ	107
5.18	メンバーへのアクセス制限	110
5.19	メソッドをクラスの外に記述する方法	111
5.20	パラメータによる汎用クラスの定義	112
5.20.1	概要	112
5.20.2	パラメータによる汎用クラスの実装	112
5.20.3	パラメータによる汎用クラスの開発手順	114
5.21	クラスのフォワード宣言	116
5.22	クラスのテストベンチへの応用	116
5.23	インターフェースクラス	118
5.23.1	概要	118
5.23.2	機能	119
第6章 プロセス		122
6.1	シミュレーションプロシージャ	122
6.1.1	initial プロシージャ	123
6.1.2	always プロシージャ	124
6.1.3	final プロシージャ	129
6.2	ブロック文	130
6.2.1	begin-end ブロック	130
6.2.2	fork-join ブロック	131
6.2.3	ブロック名	135
6.2.4	fork ブロックの効果的利用	136
6.3	タイミングによる実行制御	137
6.3.1	タイミングによる実行制御の概要	137
6.3.2	デイレイによる制御	138
6.3.3	エッジセンシティブイベント制御	138
6.3.4	代入内タイミング制御	140

6.3.5	レベルセンシティブイベント制御	142
6.3.6	イベント制御と解除	142
6.4	プロセス制御	144
6.4.1	wait 文	144
6.4.2	wait fork 文	144
6.4.3	disable fork 文	145
6.4.4	wait_order 文	146
6.5	プロセスと RNG	148
6.6	ユーザ固有のプロセス制御	149
 第7章 代入文		 153
7.1	連続代入文	154
7.2	ビヘイビア代入文	155
7.2.1	ブロッキング代入文	156
7.2.2	ノンブロッキング代入文	156
7.3	パターン指定による代入	159
 第8章 オペレータと式		 161
8.1	オペレータ	161
8.1.1	代入オペレータ	163
8.1.2	インクリメントおよびデクリメントオペレータ	163
8.1.3	算術オペレータ	164
8.1.4	比較オペレータ	165
8.1.5	ワイルドカード比較オペレータ	166
8.1.6	論理オペレータ	168
8.1.7	bitwise オペレータ	169
8.1.8	計算オペレータ	169
8.1.9	シフトオペレータ	170
8.1.10	conditional オペレータ	171
8.1.11	結合オペレータ	172
8.1.12	inside オペレータ	173
8.1.13	ビットストリームオペレータ	174
8.2	オペランド	177
8.2.1	パートセレクト	177
8.2.2	unpacked アレイ	179

8.3	タグ付きメンバーの操作	179
第9章	実行文	181
9.1	if文	181
9.1.1	全ての条件を列挙	181
9.1.2	unique-if文と unique0-if文	182
9.1.3	priority-if文	183
9.2	case文	184
9.2.1	unique-case文と unique0-case文	185
9.2.2	priority-case文	186
9.2.3	casezと casex	186
9.3	inside オペレータと if文および case文	187
9.3.1	if文と inside オペレータ	187
9.3.2	case文と inside オペレータ	188
9.4	ループ文	189
9.4.1	for文	190
9.4.2	repeat文	190
9.4.3	foreach文	192
9.4.4	while文	194
9.4.5	do-while文	195
9.4.6	forever文	196
9.5	return文	196
9.6	break文	197
9.7	continue文	198
第10章	タスクとファンクション	199
10.1	タスク	199
10.1.1	ポートリスト	199
10.1.2	タスク内の記述	200
10.2	ファンクション	201
10.2.1	ファンクションの制限	201
10.2.2	ポートリスト	201
10.2.3	ファンクション内の記述	202
10.3	引数に標準値を指定する方法	203
10.4	値を戻すファンクションの使用	204

10.5	再帰呼び出し	204
10.6	クラスのメソッドと再帰呼び出し	205
10.7	メソッド内での変数の初期化	206
10.8	引数としてのアレイ	209
10.9	インポートとエクスポート	210
第 11 章 クロッキングブロック		212
11.1	最も簡単なクロッキングブロック	213
11.2	クロッキングキュー	213
11.3	クロッキングイベントと Observed 領域	216
11.4	サイクルディレイ	217
第 12 章 プロセス間の同期と交信		220
12.1	セマフォ	220
12.2	メールボックス	223
12.3	パラメータ化したメールボックス	227
12.4	名称付きイベント	228
12.4.1	概要	228
12.4.2	triggered メソッド	231
12.4.3	引数としてのイベントオブジェクト	233
12.4.4	イベント資源の解放	233
12.4.5	イベントの比較	233
12.4.6	イベントの別名	234
第 13 章 チェッカー		235
13.1	概要	235
13.2	チェッカーインスタンス	236
13.3	自由変数	238
13.4	DUT 出力のサンプリング	239
第 14 章 プログラム		242
14.1	シンタックス	242
14.2	プログラムの特徴	244

14.3	プログラムの制御	246
14.4	シミュレーションの終了	246
第 15 章	インターフェース	248
15.1	シンタックス	248
15.2	インターフェースの機能概要	250
15.3	ジェネリックインターフェースによる接続	251
15.4	modport	251
15.5	パラメータ化したインターフェース	252
15.6	virtual インターフェース	253
第 16 章	パッケージ	258
16.1	シンタックス	258
16.2	パッケージの定義法	259
16.3	パッケージの使用法	260
16.4	std パッケージ	264
第 17 章	モジュール	267
17.1	概要	267
17.2	モジュールの定義	268
17.3	ポートリスト	271
17.3.1	Verilog スタイルと SystemVerilog スタイル	271
17.3.2	ポートの方向に関するルール	272
17.4	パラメータ化したモジュール	272
17.5	トップレベルモジュール	275
17.6	モジュールインスタンス	275
17.7	インターフェースを使用するモジュール記述	276
17.8	未定義モジュールの宣言	278
17.9	階層名称	279
第 18 章	システムタスクとシステムファンクション	281
18.1	$\$display$ および $\$write$ タスク	281
18.2	$\$sformat$ タスクと $\$sformatf$ ファンクション	283

18.3	モニタリング	284
18.4	シミュレーション時間取得ファンクション	285
18.5	\$printtimescale	286
18.6	値の変換	287
18.7	情報取得ファンクション	287
18.8	ビット vector システムファンクション	290
18.9	サンプル値を参照するためのファンクション	292
18.10	エラー処理タスク	293
18.11	確率分布ファンクション	295
18.12	シミュレーション制御	295
18.13	その他のシステムタスクおよびシステムファンクション	296
18.14	コマンドラインの操作	297
18.15	VCD ファイル	299
18.15.1	VCD ファイルの指定	299
18.15.2	VCD ファイルへの記録	299
18.15.3	VCD ファイルへの記録の一時的停止と再開	300
18.15.4	VCD ファイル作成例	300
 第 19 章 制約によるランダムスティミュラスの生成		303
19.1	概要	303
19.2	ランダム変数	305
19.2.1	ランダム変数の概要	305
19.2.2	rand 修飾子	306
19.2.3	randc 修飾子	306
19.2.4	ランダム変数定義例	306
19.3	乱数発生メソッド	307
19.4	制約	309
19.4.1	inside オペレータ	309
19.4.2	dist オペレータ	312
19.4.3	unique オペレータ	313
19.4.4	implication オペレータ	315
19.4.5	foreach 制約	316
19.4.6	乱数決定順序	317
19.5	実行時に制約を定義する方法	319
19.6	ランダム変数の制御	320
19.7	制約の制御	322

19.8	randomize() メソッドによるランダム変数の制御	323
19.9	条件の否定	325
19.10	ストラクチャ	326
19.11	キューに乱数を発生	327
19.12	チェッカーとしての制約	328
19.13	制約をテストケースごとに指定する方法	331
19.14	制約をクラス外部に定義する方法	332
19.15	std::randomize() ファンクション	333
19.16	システムファンクション	334
第 20 章	SystemVerilog の検証機能	336
20.1	ファンクショナルカバレッジ	336
20.1.1	概要	336
20.1.2	カバレッジ計算	337
20.1.3	カバレッジ計算例	338
20.2	アサーション	342
20.2.1	概要	343
20.2.2	アサーションの種類	344
20.2.3	アサーションの式	345
20.2.4	アサーション記述例	346
第 21 章	モデリングと検証	350
21.1	組み合わせ回路	351
21.1.1	組み合わせ回路の記述ルール	351
21.1.2	組み合わせ回路を検証するタイミング	352
21.1.3	デコーダ	353
21.1.4	エンコーダ	355
21.1.5	ALU	357
21.1.6	コンパレータ	358
21.1.7	Gray コードをバイナリーコードに変換する回路	360
21.1.8	バレルシフタ	362
21.1.9	符号付き整数の加減算	363
21.2	シーケンシャル回路	366
21.2.1	シーケンシャル回路の記述ルール	367
21.2.2	シーケンシャル回路の検証	367

21.2.3	バイナリーカウンター	369
21.2.4	JK-フリップフロップ	371
21.2.5	Johnson カウンター	372
21.2.6	ユニバーサルシフトレジスタ	374
21.2.7	Gray カウンター	376
21.2.8	リングカウンター	379
21.2.9	Gated clock の記述例	380
21.3	FSM	383
21.3.1	概要	383
21.3.2	Moore FSM モデリング	384
21.3.3	Mealy FSM モデリング	387
21.4	FSM とビットシーケンスの認識	390
21.4.1	ビットシーケンス認識問題	390
21.4.2	Moore FSM モデリング	390
21.4.3	Mealy FSM モデリング	392
第 22 章 UVM 概説		395
22.1	UVM とは何か	395
22.2	検証技術のトレンドと UVM	396
22.3	UVM の検証要素	396
22.3.1	トランザクションとシナリオに関連する UVM クラス	396
22.3.2	メソドロジークラス	398
22.4	TLM	399
22.5	UVM シミュレーション	400
22.5.1	シミュレーションフェーズ	400
22.5.2	run_test() メソッド	400
22.6	UVM 検証コンポーネントの開発	401
22.7	トップモジュール	402
第 23 章 コンパイラディレクティブ		404
23.1	<code>`include</code> 文	404
23.2	<code>`define</code> 文	405
23.2.1	定数を定義する場合	405
23.2.2	接頭辞および接尾辞を持つ名称の創成	406
23.3	文字列内のパラメータ展開	407

23.4	`endif 文	408
23.5	`__FILE__と`__LINE__	408
第 24 章	シミュレーション実行モデル	410
24.1	スケジューリング領域	410
24.2	#0 デイレーの効果	412
参考文献		415
索引		417

本書で使用する略語一覧

略語	定義
ALU	Arithmetic Logic Unit
BNF	Backus-Naur Form
CRT	Constrained Random Test
DDR	Double Data Rate
DMA	Direct Memory Access
DPI	Direct Programming Interface
DT	Directed Test
DUT	Design Under Test または Device Under Test
EDA	Electronic Design Automation
FIFO	First In First Out
FSM	Finite State Machine
HDL	Hardware Description Language
LHS	Left Hand Side
LRM	Language Reference Manual, すなわち IEEE Std 1800-2017
LSB	Least Significant Bit
MOS	Metal-Oxide-Silicon または Metal-Oxide-Semiconductor
MSB	Most Significant Bit
OOP	Object Oriented Programming
RNG	Random Number Generator
RTL	Register Transfer Level
TLM	Transaction Level Modeling
UDP	User-Defined Primitive
UVM	Universal Verification Methodology
VCD	Value Change Dump

各種目次

目 次

図 1-1	SystemVerilog の機能範囲	2	図 12-3	名前付きイベントによるプロセス間の同期	228
図 1-2	SystemVerilog と Verilog HDL の関係	3	図 13-1	DUT とチェッカーの実行するタイミング	239
図 2-1	FullAdder の構成	14	図 13-2	チェッカーによる DUT のレスポンスのサンプリング	239
図 2-2	3つのファイルから構成されるコンパイルユニット	21	図 14-1	program ブロックによる DUT のレスポンスのサンプリング	245
図 3-1	wand の効果	28	図 15-1	検証環境で使用されるインターフェースの例	248
図 3-2	連続代入文 my_and = a & b の効果	31	図 15-2	インターフェースのインスタンスと virtual インターフェース	254
図 4-1	pack1 のレイアウト	56	図 15-3	virtual インターフェースを使用する検証環境	254
図 4-2	packed 次元と unpacked 次元	61	図 16-1	パッケージをインクルードファイルで構成する方法	259
図 4-3	bytes のレイアウト	62	図 16-2	パッケージを使用する方法	260
図 4-4	unpacked アレイ name は不連続	63	図 17-1	パラメータ宣言を明示的な宣言形態に変換する手法	274
図 4-5	associative アレイの機能	70	図 17-2	階層構造の例	280
図 4-6	キューはサイズが [0:\$] であるアレイ	74	図 18-1	次元数の順序	289
図 5-1	static メンバー (counter) と automatic メンバー (addr, data, name) の領域確保の差異	92	図 18-2	\$rose()==1'b1 となる状態	292
図 5-2	sample_array[10] を定義した時点のアレイの状態	95	図 19-1	検証環境におけるランダムステイミュラス生成の役割	303
図 5-3	クラス継承	99	図 19-2	制約定義をクラス外で記述する例	333
図 5-4	クラス間の関連	100	図 20-1	検証におけるカバレッジ計算の流れ	337
図 5-5	virtual メソッドの効果	104	図 20-2	check_a_b が仕様を満たす状況	344
図 5-6	同一クラスタイプのオブジェクトを管理するリスト	109	図 21-1	3ビットのコードを8ビット one-hot コードに変換するデコーダ	353
図 5-7	typedef とクラスパラメータの対応	115	図 21-2	8ビットの情報を3ビットのコードに変換するエンコーダ	355
図 6-1	begin-end および fork-join 文を利用して複数の文を一つの文として扱う例	130	図 21-3	ALU のブロックダイアグラム	357
図 6-2	プロセス生成時の実行環境の保存	134	図 21-4	コンパレータ	359
図 6-3	子プロセスが親プロセスの実行環境を共有している状況	135	図 21-5	Gray コードをバイナリーコードに変換する回路のブロックダイアグラム	360
図 6-4	プロセスと RNG	148	図 21-6	バレルシフタの機能 (N==2)	362
図 7-1	連続代入文の効果	153	図 21-7	adder_subtractor のブロックダイアグラム	364
図 7-2	連続代入文による組み合わせ回路の生成	153			
図 7-3	シーケンシャル回路の生成例	158			
図 10-1	import と export	210			
図 11-1	クロッキングスキュー	214			
図 12-1	セマフォの概念図	220			
図 12-2	メールボックス使用の概念図	224			

図 21-8	符号付き加減算器の構造 ([9])	365	カーの状態遷移図	385	
図 21-9	バイナリーカウンターのブロックダイアグラム	369	図 21-16	Mealy FSM ([9])	387
図 21-10	ユニバーサルシフトレジスタのブロックダイアグラム	374	図 21-17	Mealy FSM によるパリティチェッカーの状態遷移図	388
図 21-11	Gated clock (望ましくない構造)	380	図 21-18	パターン認識回路	390
図 21-12	Gated clock の実装例 (望ましい構造)	382	図 21-19	認識問題の Moore FSM 状態遷移図	391
図 21-13	パリティチェッカーのブロックダイアグラム	383	図 21-20	認識問題の Mealy FSM 状態遷移図	392
図 21-14	Moore FSM([9])	384	図 22-1	UVM の成り立ち	395
図 21-15	Moore FSM によるパリティチェッカーの状態遷移図	385	図 22-2	UVM のテストベンチ構成 ([2])	397
			図 22-3	典型的なトランザクション処理の例	399
			図 22-4	UVM を使用した環境での実行の流れ	400
			図 24-1	イベントスケジューリング領域	411

表

表 1-1	SystemVerilog 言語仕様の変遷	1	表 4-4	associative アレイのメソッド	72
表 1-2	シンタックス要素の意味	5	表 4-5	キューを操作するメソッド	77
表 1-3	本書の構成	8	表 4-6	with を必要とするアレイ検索メソッド	79
表 1-4	英語表記と和訳表記の対応	10	表 4-7	検索条件を省略できるアレイ検索メソッド	79
表 2-1	コンパイルユニット, コンパイルユニットスコープ, \$unit	20	表 4-8	アレイ要素の順序を操作するメソッド	81
表 2-2	`timescale コンパイラディレクティブ	22	表 4-9	アレイを計算するメソッド	82
表 2-3	SystemVerilog の時間単位名称	22	表 4-10	アレイ走査法のまとめ	84
表 2-4	\$time の値	22	表 5-1	クラスの代表的なメンバー	86
表 3-1	標準データタイプの使用例の効果	25	表 5-2	シンタックス要素の対応	88
表 3-2	ネット型の種類	26	表 5-3	宣言されたプロパティの lifetime と効果	93
表 3-3	ネットと変数の差異	30	表 5-4	クラスハンドルの状態	97
表 3-4	4-state 型の種類	31	表 5-5	アクセス制御に使用するキーワードとその機能	110
表 3-5	2-state 型の種類と意味	32	表 5-6	変数と設定されたアクセス制限	110
表 3-6	2-state 型の宣言例の結果	33	表 6-1	シミュレーションプロシージャの種類	123
表 3-7	integral データタイプの種類	34	表 6-2	always_comb と always @*の差異	128
表 3-8	string データタイプで使用できるオペレータ	36	表 6-3	fork ブロックの機能	131
表 3-9	string メソッド	37	表 6-4	エッジセンシティブイベント制御	138
表 3-10	enum データタイプとして宣言された変数の型と内容	42	表 6-5	代入内タイミング制御と同等な表現 ([1])	141
表 3-11	enum データタイプのメソッド	43	表 6-6	イベント制御と解除	143
表 3-12	リテラルの記述例	44	表 6-7	process クラスのメソッド	150
表 3-13	可変長リテラルの種類	50	表 7-1	ノンブロッキング代入文の効果	157
表 4-1	ダイナミックアレイに使用できるメソッドとコンストラクタ	66	表 8-1	SystemVerilog のオペレータと優先順序および結合法則	162
表 4-2	アレイ宣言の状態	67	表 8-2	算術オペレータ	164
表 4-3	associative アレイの index_type	71	表 8-3	比較オペレータ	165

表 8-4	ワイルドカード比較オペレータ	167	表 18-13	\$system システムファンクション	296
表 8-5	AND(&) オペレータ	169	表 18-14	コマンドラインを操作するファンク ション	297
表 8-6	OR() オペレータ	169	表 19-1	乱数発生メソッド	307
表 8-7	XOR(^) オペレータ	169	表 19-2	dist_weight の機能	312
表 8-8	XNOR(~~,~^~) オペレータ	169	表 19-3	rand_mode() メソッド	321
表 8-9	NEGATION (~) オペレータ	169	表 19-4	constraint_mode() メソッド	322
表 8-10	シフトオペレータ	170	表 19-5	乱数発生に使用できるシステムファン クション	335
表 8-11	conditional オペレータ	171	表 20-1	アサーションスレッドの活動	349
表 8-12	expr1==expr2 が真でない場合の conditional オペレータの解消法	171	表 21-1	バイナリーコード, one-hot コード, Gray コード, Johnson コード	350
表 11-1	サンプリングとドライブする時期	215	表 21-2	組み合わせ回路を検証するの手段	353
表 12-1	セマフォのメソッド	221	表 21-3	ALU の機能	357
表 12-2	メールボックスのメソッド	225	表 21-4	b3 を求めるための Gray コード	360
表 12-3	実行順序とイベント解除	229	表 21-5	adder_subtractor の仕様	364
表 15-1	検証環境で使用する要素	255	表 21-6	シーケンシャル回路の動作の代表的ス ケジューリング領域	368
表 16-1	std パッケージのクラスとメソッド	264	表 21-7	競合状態を避ける事ができる代表的ス ケジューリング領域	368
表 17-1	シンタックス要素の対応	269	表 21-8	JK-フリップフロップの機能 ([9])	371
表 17-2	モジュールヘッダの記述例	273	表 21-9	ユニバーサルシフトレジスタの機能	375
表 18-1	書式の意味 (例 18-2)	283	表 21-10	ユニバーサルシフトレジスタの動作	377
表 18-2	書式とプリント結果	283	表 21-11	8 ビットリングカウンターの動作	379
表 18-3	\$sformat と\$sformatf	283	表 21-12	Moore FSM と Mealy FSM	383
表 18-4	シミュレーション時間取得ファンク ション	285	表 22-1	階層的テストベンチを構成するレイ ヤーと検証要素	396
表 18-5	値変換用のファンクション	287	表 22-2	トランザクションとシナリオに関連す るクラス	397
表 18-6	情報取得ファンクション	288	表 22-3	メソドロジークラス	398
表 18-7	ビット vector を調べるファンクシ ョン	291	表 22-4	シミュレーションフェーズ	401
表 18-8	サンプル値の状態を参照するファンク ション	292	表 22-5	検証環境を構成する主な検証コンポー ネント	402
表 18-9	エラー処理タスク	293	表 23-1	代表的なコンパイラディレクティブ	404
表 18-10	確率分布ファンクション	295			
表 18-11	シミュレーションの実行を制御する ためのタスク	296			
表 18-12	\$stop と\$finish の引数 n の意味	296			

例

例 1-1	名称の指定例	5	例 3-2	変数が複数のドライバを持つ例	25
例 1-2	エスケープシーケンスの指定例	6	例 3-3	ネット型の宣言例	27
例 2-1	2 入力の multiplexer のゲートによる 実装例	19	例 3-4	変数の定義例	29
例 2-2	`timescale の使用例 ([1])	22	例 3-5	モジュールのポートとして変数を指定 する例	29
例 2-3	automatic 変数のメモリ割り当ての例	23	例 3-6	ネット型の使用例	30
例 3-1	標準データタイプの例	24	例 3-7	2-state 型の使用例	32

例 3-8	bit 型の使用例	33	例 4-22	アレイ要素の順序を操作するメソッド の使用例	81
例 3-9	void 型の使用例	34	例 4-23	アレイを計算するメソッドの使用例	82
例 3-10	chandle 型の使用例	35	例 4-24	アレイ要素の和を求める sum() メソッド の使用例	82
例 3-11	string 型の使用例	35	例 5-1	クラスの例	85
例 3-12	文字列結合オペレータの例	36	例 5-2	super.new() が必要な例	90
例 3-13	string メソッドの使用例	37	例 5-3	static メンバーへのアクセス	92
例 3-14	イベントデータタイプの使用例	38	例 5-4	static クラスメソッドの例	94
例 3-15	typedef 文の使用例	40	例 5-5	ハンドルのアレイの初期化例	96
例 3-16	enum 型の使用例	41	例 5-6	クラスのコピー例	97
例 3-17	enum 型のデータタイプの宣言例	42	例 5-7	クラス継承の例 (その 1)	99
例 3-18	enum で定義されている全てのラベル を走査する例	43	例 5-8	クラス継承の例 (その 2)	100
例 3-19	parameter の使用例	45	例 5-9	\$cast の使用例	102
例 3-20	パラメータポートリストの parameter	46	例 5-10	virtual メソッドの例	104
例 3-21	タイプをパラメータとして指定する例	46	例 5-11	アブストラクトクラスの定義例	106
例 3-22	cast オペレータの使用例 ([1])	48	例 5-12	クラススコープオペレータの使用例	108
例 3-23	ダイナミック型変換の例	49	例 5-13	アクセス制御例	110
例 3-24	可変長リテラルの使用例	50	例 5-14	メソッド定義をクラス外に記述する例	111
例 3-25	インデックス指定による初期化	51	例 5-15	辞書の実装例	113
例 3-26	ref タイプのポートが必要な例	52	例 5-16	クラスによる DUT の検証	117
例 4-1	ストラクチャの定義例	55	例 5-17	インターフェースクラスの定義と使用 例	120
例 4-2	packed ストラクチャの定義例	56	例 6-1	マルチプレクサの記述例	124
例 4-3	ストラクチャへの値の設定例 ([1])	57	例 6-2	fork-join の使用例	131
例 4-4	ユニオンの定義例	58	例 6-3	fork-join_any の使用例	132
例 4-5	packed ユニオンの定義例	60	例 6-4	fork-join_none の使用例	133
例 4-6	タグ付きユニオンの定義例	60	例 6-5	fork-join_none の間違った使用例	134
例 4-7	packed アレイの例	62	例 6-6	fork-join_none の正しい使用例	135
例 4-8	unpacked アレイの例	62	例 6-7	fork ブロックの効果的利用	136
例 4-9	アレイリテラルによる初期化の例	63	例 6-8	iff の使用例	139
例 4-10	アレイのインデックスを使用したアレ イリテラルの例	64	例 6-9	代入内タイミング制御の簡単な使用例	141
例 4-11	packed アレイへのアクセス	64	例 6-10	wait fork の使用例	144
例 4-12	new[] コンストラクタの使用例	67	例 6-11	disable fork の使用例	146
例 4-13	アレイのコピー	68	例 6-12	wait_order の使用例	147
例 4-14	ダイナミックアレイの初期化	69	例 6-13	子プロセスに異なるシードが与えられ る例	148
例 4-15	associative アレイの要素登録例	71	例 6-14	プロセス制御の例 ([1])	150
例 4-16	associative アレイの初期化例	73	例 6-15	suspend() メソッドの使用例	151
例 4-17	キューの使用例	74	例 7-1	連続代入文の使用例 ([1])	154
例 4-18	キューの操作例	76	例 7-2	ノンブロッキング代入文の例 ([1])	157
例 4-19	キューメソッドの使用例	76	例 7-3	ノンブロッキング代入文によるシーケ ンシャル回路の記述例	158
例 4-20	検索条件を伴うアレイ検索メソッドの 使用例	78	例 7-4	パターン代入文の例	159
例 4-21	検索条件を省略できるアレイ検索メ ソッドの使用例	80	例 8-1	オペレータの使用例	162

例 8-2	インクリメントオペレータの使用例 ..	163	例 10-2	ファンクションの定義例	202
例 8-3	算術オペレータの例	164	例 10-3	標準値を伴う引数の使用例	203
例 8-4	比較オペレータの使用例	165	例 10-4	再帰呼び出しの例 ([1])	204
例 8-5	ワイルドカード比較オペレータの例 ..	167	例 10-5	クラスメソッドにおける再帰呼び出し の例	206
例 8-6	論理オペレータの使用例	168	例 10-6	変数の初期化例	206
例 8-7	計算オペレータの使用例	170	例 10-7	automatic メソッド内での static 変 数の初期化	208
例 8-8	cond_predicate が x または z の例 ..	171	例 10-8	引数としてダイナミックアレイを指定 する例	209
例 8-9	結合オペレータの使用例	172	例 11-1	標準クロッキングの使用例	213
例 8-10	inside オペレータの使用例	173	例 11-2	skew の使用例	215
例 8-11	ビットストリームオペレータ使用例 (その 1)	175	例 11-3	クロッキングブロックによるイベント 待ちの使用例	216
例 8-12	ビットストリームオペレータ使用例 (その 2)	176	例 11-4	サイクルディレーの使用例	217
例 8-13	ビットストリームオペレータ使用例 (その 3)	176	例 11-5	標準クロッキングとアサーション ..	218
例 8-14	ビットストリームオペレータ使用例 (その 4)	177	例 12-1	セマフォの使用例	221
例 8-15	インデックス付きパートセレクトの使 用例	178	例 12-2	二つのプロセスが共有資源にアクセス する例	222
例 8-16	タグ付きメンバーの使用例	179	例 12-3	メールボックスの使用例	224
例 9-1	unique-if の使用例	182	例 12-4	二つの並列プロセスがメールボックス を使用する例	226
例 9-2	priority-if 文の使用例	183	例 12-5	パラメータ化したメールボックス ..	227
例 9-3	case 文の例	184	例 12-6	複数のイベント解除を待つ例	230
例 9-4	unique-case 文の使用例	185	例 12-7	イベント待ちと解除が同時に発生する 例	231
例 9-5	casez 文の使用例 ([1])	186	例 12-8	triggered メソッドの使用例	232
例 9-6	casex 文の使用例 ([1])	187	例 12-9	別名の使用例	234
例 9-7	if 文における inside オペレータの使 用例	187	例 13-1	チェッカーの使用例 ([1])	236
例 9-8	case 文における inside オペレータの 使用例	188	例 13-2	チェッカーインスタンスをプロシー ジャで使用する例	237
例 9-9	for 文の使用例	190	例 13-3	チェッカーの自由変数の例	238
例 9-10	repeat 文の使用例	191	例 13-4	チェッカーによる DUT 出力のサンプ リング	240
例 9-11	foreach 文の使用例 (その 1)	192	例 14-1	Reactive 領域の特徴	244
例 9-12	foreach 文の使用例 (その 2)	193	例 14-2	\$exit 文の使用例	246
例 9-13	foreach 文の使用例 (その 3)	194	例 14-3	シミュレーションの終了	247
例 9-14	while 文の使用例	195	例 15-1	virtual インターフェースの使用例 ..	254
例 9-15	do-while 文の使用例	195	例 16-1	パッケージの使用例 (その 1)	261
例 9-16	return 文の使用例	196	例 16-2	パッケージの使用例 (その 2)	262
例 9-17	foreach ループ内で break 文を使用す る例	197	例 16-3	process クラスの使用例	265
例 9-18	forever ループ内で break 文を使用す る例	197	例 16-4	std::randomize() ファンクション の使用例	266
例 9-19	foreach ループ内で continue 文を使 用する例	198	例 17-1	データタイプをパラメータに持つモ ジュールの定義例	274
例 10-1	タスクの定義例	200			

例 17-2	インターフェースを使用したバイナリーカウンター	276	例 19-16	キューに乱数を発生する例	327
例 17-3	階層名称の使用例	279	例 19-17	制約をチェッカーとして利用する例 (その 1)	328
例 18-1	<code>\$display</code> タスクの例	281	例 19-18	制約をチェッカーとして利用する例 (その 2)	330
例 18-2	string 型の書式例	282	例 19-19	制約をクラス外部で定義する例	331
例 18-3	10 進数をプリントする書式 (%d) の使用例	283	例 19-20	制約をクラスの外部に定義する例	332
例 18-4	<code>\$sformatf</code> の使用例	284	例 19-21	<code>std::randomize()</code> ファンクションによる乱数発生例	334
例 18-5	<code>\$monitor</code> システムタスクの使用例	285	例 19-22	システムファンクションの例	335
例 18-6	<code>\$realtime</code> の使用例 ([1])	286	例 20-1	ピンを定義しないカバレッジ計算例	338
例 18-7	<code>\$printtimescale</code> の使用例 ([1])	286	例 20-2	制約を考慮するカバレッジの定義	339
例 18-8	<code>\$typename</code> の使用例	287	例 20-3	テストのカバレッジ計算	341
例 18-9	情報取得ファンクションの使用例	289	例 20-4	Johnson カウンターを検証するアサーションの例	346
例 18-10	<code>\$left()</code> , <code>\$right()</code> , <code>\$increment()</code> の使用例	289	例 20-5	標準クロッキングを使用したアサーション記述例	348
例 18-11	ビット vector システムファンクションの使用例	291	例 21-1	デコーダの記述例	353
例 18-12	<code>\$rose()</code> の使用例	292	例 21-2	エンコーダの記述例	355
例 18-13	<code>\$fatal()</code> タスク使用例	294	例 21-3	ALU の記述例	357
例 18-14	<code>\$random()</code> の使用例	295	例 21-4	コンパレータの記述例	359
例 18-15	<code>\$system</code> の使用例	296	例 21-5	Gray コードをバイナリーコードに変換する記述例	360
例 18-16	<code>\$test\$plusargs()</code> の使用例	297	例 21-6	バレルシフタの記述例	362
例 18-17	<code>\$value\$plusargs()</code> の使用例	298	例 21-7	符号付き加減算器の記述例	364
例 18-18	<code>\$dumpfile</code> と <code>\$dumpvars</code> の使用例	300	例 21-8	バイナリーカウンターの記述例	369
例 19-1	<code>rand</code> と <code>randc</code> の使用例	306	例 21-9	JK-フリップフロップの記述例	371
例 19-2	<code>pre_randomize()</code> と <code>post_randomize()</code> の使用例	308	例 21-10	Johnson カウンターの記述例	373
例 19-3	inside オペレータの使用例 (ランダム変数の値域)	310	例 21-11	ユニバーサルシフトレジスタの記述例	375
例 19-4	アレイを使用した inside オペレータ	311	例 21-12	Gray カウンターの記述例	377
例 19-5	dist オペレータの使用例	312	例 21-13	リングカウンターの記述例	379
例 19-6	unique オペレータの使用例 ([1])	314	例 21-14	Gated clock の記述例 (望ましくない方法)	380
例 19-7	implication オペレータの使用例	315	例 21-15	Gated clock の記述例 (望ましい方法)	382
例 19-8	foreach アレイの使用例 (ダイナミックアレイへの適用)	316	例 21-16	パリティチェッカーの実装例 (Moore FSM)	385
例 19-9	solve 制約の使用例	318	例 21-17	パリティチェッカーの実装例 (Mealy FSM)	388
例 19-10	実行時に制約を定義する例	319	例 23-1	共通の接頭辞を簡略化する例	407
例 19-11	ランダム変数の状態を制御する例	320	例 23-2	<code>`_FILE__</code> と <code>`_LINE__</code> の使用例	408
例 19-12	制約を制御する例	322			
例 19-13	randomize() メソッドによるランダム変数の制御例	324			
例 19-14	条件の否定の例	325			
例 19-15	ストラクチャのメンバーに乱数を発生する例	326			