

Compiling Esterel

Dumitru Potop-Butucaru

Stephen A. Edwards

Gérard Berry

 Springer

Contents

Preface

vii

I The Esterel Language

1	Introduction to Esterel	3
1.1	Reactive Systems	3
1.2	The Synchronous Hypothesis	4
1.3	Implementation Issues	5
1.4	Causality	6
1.5	Related work	6
1.6	A First Esterel Example	7
1.7	Causality Cycles	8
1.8	Code Generation	9
1.8.1	Translation to Explicit State Machines	9
1.8.2	Translation to Circuits	10
1.8.3	Direct Compilation to C Code	12
1.9	Executing the Generated Code	12
1.9.1	Existing Solutions	13
2	The Esterel Language	15
2.1	Syntax and Naïve Semantic Principles	15
2.2	The Kernel Esterel Language	20
2.3	Esterel Through Examples	24
2.4	Host Language	31
2.5	Program Structure and Interface	31
2.5.1	Data Handling	32
2.5.2	Signal and Signal Relation Declarations	34
2.5.3	The run Pseudo-Statement	34

II Formal Semantics

3	Introduction to Esterel Semantics	41
3.1	Intuition and Mathematical Foundations	41
3.1.1	The Constructive Approach	43
3.2	Flavors of Constructive Semantics	47
3.3	Conventions and Preliminary Definitions	50
3.3.1	Global Correctness of an Esterel Program	50
3.3.2	Restriction to Kernel Esterel	51
3.3.3	Signal Events	51
3.3.4	Trap Handling and Completion Codes	51
4	Constructive Behavioral Semantics	55
4.1	Behavioral Transitions	55
4.1.1	Transition syntax	55
4.1.2	States as Decorated Terms	56
4.1.3	State Syntax	57
4.2	Analysis of Potentials	58
4.2.1	The Definition of <i>Must</i> , <i>Can</i> , and <i>Can</i> ⁺	60
4.2.2	Elementary Properties	68
4.3	Semantic Rules	68
4.4	Proof	72
4.5	Determinism	72
4.6	Loop-Safe Programs. Completion Code Potentials	73
4.7	Program Behavior	75
5	Constructive Operational Semantics	79
5.1	Microsteps	80
5.2	COS Terms	80
5.2.1	Control Flow Propagation	81
5.2.2	State-Dependent Behavior	81
5.2.3	Syntax of Semantic Terms	82
5.3	Data Representation	84
5.4	Semantic Rules	85
5.4.1	Rules for Pure Esterel Primitives	85
5.4.2	Rules for Data-Handling Primitives	92
5.5	Analysis of Potentials	95
5.5.1	Reduction to Non-Dotted Terms	96
5.5.2	Non-Dotted Terms over Dataless Primitives	97
5.5.3	Non-Dotted Terms over Data-Handling Primitives	99
5.6	Behaviors as Sequences of Microsteps	99
5.7	COS versus CBS	101

6	Constructive Circuit Translation	103
6.1	Digital Circuits with Data	104
6.1.1	Circuit Semantics. Constructive Causality	104
6.1.2	Extension to Circuits with Data	107
6.1.3	Formal Definitions	110
6.2	Translation Principles	112
6.2.1	The Selection Circuit	114
6.2.2	The Surface and Depth Circuits	115
6.2.3	The Global Context	116
6.3	Translation Rules	117
6.3.1	Dataless Primitives	117
6.3.2	Data-Handling Primitives	126
6.4	Circuit Translation versus COS	128

III Compiling Esterel

7	Overview	135
7.1	Compiler Classes	135
7.2	A Brief History	136
7.3	The INRIA Compiler	137
7.4	The Synopsys Compiler	139
7.5	The Saxo-RT Compiler	140
7.6	The Columbia Esterel Compiler	144
8	The GRC Intermediate Format	145
8.1	Definition and Intuitive Semantics	146
8.1.1	The Hierarchical State Representation	146
8.1.2	The Control/Data Flowgraph	149
8.1.3	Implementation Issues	154
8.2	Esterel to GRC Translation	158
8.2.1	Translation Principles	158
8.2.2	Translation Rules	159
8.2.3	The Global Context	167
8.3	Formal Simulation Semantics and Translation Correctness	167
8.4	Format Optimizations	168
8.4.1	State Representation Analysis	170
8.4.2	Flowgraph Optimizations	173
9	Code Generation from GRC	179
9.1	Defining “Acyclic”	180
9.2	Code Generation for Acyclic Specifications	185
9.2.1	State Encoding	185
9.2.2	Flowgraph Transformations	190
9.2.3	Scheduling	192

9.3	Code Generation for Cyclic Specifications	193
9.4	Benchmarks	200
10	The Columbia Compiler	203
10.1	The Dynamic Technique	203
10.1.1	An Example	204
10.1.2	Sequential Code Generation	211
10.1.3	The Clustering Algorithm	214
10.2	The Program Dependence Graph Approach	215
10.2.1	Program Dependence Graphs	216
10.2.2	Scheduling	218
10.2.3	Restructuring the PDG	221
10.2.4	Generating Sequential Code	228
10.3	Benchmarks	229
A	Language Extensions	235
A.1	Signal Expressions	235
A.1.1	Syntactic Aspects and Limitations	236
A.1.2	Combinational Expressions	236
A.1.3	The <code>pre</code> Operator	238
A.1.4	Delay Expressions. Preemption Triggers	241
A.2	Traps and Trap Expressions	242
A.2.1	Concurrent Traps and Trap Expressions	243
A.2.2	Valued Traps	244
A.3	The <code>finalize</code> Statement	245
A.4	Tasks	247
A.4.1	Task Synchronization Semantics	248
A.4.2	Multiple <code>exec</code>	251
B	An Esterel Reference Manual	253
B.1	Lexical Conventions	253
B.1.1	Tokens	253
B.1.2	Comments	253
B.1.3	Identifiers	253
B.1.4	Reserved Words	254
B.1.5	Literals	254
B.2	Namespaces and Predefined Objects	255
B.2.1	Signals and Sensors	256
B.2.2	Variables and Constants	256
B.2.3	Traps	256
B.2.4	Types	257
B.2.5	Functions and Procedures	257
B.2.6	Tasks	257
B.3	Expressions	258
B.3.1	Data Expressions	258

B.3.2	Constant Atoms	258
B.3.3	Signal Expressions	259
B.3.4	Delay Expressions	259
B.3.5	Trap Expressions	259
B.4	Statements	260
B.4.1	Control Flow Operators	260
B.4.2	<code>abort</code> : Strong Preemption	261
B.4.3	<code>await</code> : Strong Preemption	262
B.4.4	<code>call</code> : Procedure Call	264
B.4.5	<code>do-upto</code> : Conditional Iteration (deprecated)	264
B.4.6	<code>do-watching</code> : Strong Preemption (deprecated)	264
B.4.7	<code>emit</code> : Signal Emission	264
B.4.8	<code>every-do</code> : Conditional Iteration	265
B.4.9	<code>exec</code> : Task Execution	265
B.4.10	<code>exit</code> : Trap Exit	266
B.4.11	<code>halt</code> : Wait Forever	266
B.4.12	<code>if</code> : Conditional for Data	266
B.4.13	<code>loop</code> : Infinite Loop	267
B.4.14	<code>loop-each</code> : Conditional Iteration	267
B.4.15	<code>nothing</code> : No Operation	268
B.4.16	<code>pause</code> : Unit Delay	268
B.4.17	<code>present</code> : Conditional for Signals	268
B.4.18	<code>repeat</code> : Iterate a Fixed Number of Times	269
B.4.19	<code>run</code> : Module Instantiation	270
B.4.20	<code>signal</code> : Local Signal Declaration	271
B.4.21	<code>suspend</code> : Preemption with State Freeze	272
B.4.22	<code>sustain</code> : Emit a Signal Indefinitely	273
B.4.23	<code>trap</code> : Trap Declaration and Handling	273
B.4.24	<code>var</code> : Local Variable Declaration	273
B.4.25	<code>weak abort</code> : Weak Preemption	274
B.5	Modules	275
B.5.1	Interface Declarations	276
C	The C Language Interface	281
C.1	Overview	281
C.2	C Code for Data Handling	283
C.2.1	Defining Data-handling Objects	283
C.2.2	Predefined Types	283
C.2.3	User-defined Types	283
C.2.4	Constants	286
C.2.5	Functions	286
C.2.6	Procedures	287
C.3	The Reaction Interface	288
C.3.1	Input Signals	288
C.3.2	Return Signals	289

C.3.3	Output Signals	289
C.3.4	Inputoutput Signals	289
C.3.5	Sensors	290
C.3.6	Reaction and Reset	290
C.3.7	Notes	291
C.4	Task Handling	292
C.4.1	The Low-level Layer: ExecStatus	292
C.4.2	The Functional Interface to Tasks	295
D	Esterel V7	297
D.1	Data Support	298
D.1.1	Basic Data Types	298
D.1.2	Arrays	298
D.1.3	Generic Types	299
D.1.4	Bitvectors	299
D.1.5	From Numbers to Bitvectors and Back	300
D.1.6	Data Units	300
D.2	Signals	301
D.2.1	Value-only Signals	301
D.2.2	Temporary Signals	301
D.2.3	Registered Signals	301
D.2.4	Signal Initialization	302
D.2.5	Oracles	302
D.3	Interfaces	303
D.3.1	Interface Declaration	303
D.3.2	Interfaces and Modules	303
D.3.3	Mirroring an Interface	304
D.3.4	Interface Refinement in Modules	304
D.4	Statements	304
D.4.1	Expressions and Tests	304
D.4.2	Static Replication	304
D.4.3	Enhanced Emit and Sustain Statements	305
D.4.4	Explicit and Implicit Assertions	306
D.4.5	Weak Suspension	307
D.4.6	Signal Connection by Module Instantiation	309
D.5	Multiclock Design	310
D.5.1	Clocks and Multiple Units	311
D.5.2	Simulation of Multiclock Designs by Single-clocked Designs	312
	Bibliography	315
	Index	323