

---

# Contents

<b>1</b>	<b>Introduction</b> .....	1
1.1	Introduction .....	1
1.2	Interpreters .....	3
1.3	Landin's SECD Machine .....	3
1.4	The Organisation of this Book .....	5
1.5	Omissions .....	7
<b>2</b>	<b>VMs for Portability: BCPL</b> .....	11
2.1	Introduction .....	11
2.2	BCPL the Language .....	12
2.3	VM Operations .....	15
2.4	The OCODE Machine .....	17
2.5	OCODE Instructions and their Implementation .....	18
2.5.1	Expression Instructions .....	18
2.5.2	Load and Store Instructions .....	20
2.5.3	Instructions Relating to Routines .....	20
2.5.4	Control Instructions .....	22
2.5.5	Directives .....	23
2.6	The Intcode/Cintcode Machine .....	24
<b>3</b>	<b>The Java Virtual Machine</b> .....	27
3.1	Introduction .....	27
3.2	JVM Organisation: An Overview .....	28
3.2.1	The stack .....	29
3.2.2	Method areas .....	30
3.2.3	The PC register .....	31
3.2.4	Other structures .....	32
3.3	Class Files .....	32
3.4	Object Representation at Runtime .....	40
3.5	Initialisation .....	42
3.6	Object Deletion .....	44

3.7	JVM Termination .....	45
3.8	Exception Handling .....	45
3.9	Instructions .....	46
3.9.1	Data-manipulation instructions .....	48
3.9.2	Control instructions .....	51
3.9.3	Stack-manipulating instructions .....	54
3.9.4	Support for object orientation .....	56
3.9.5	Synchronisation .....	59
3.10	Concluding Remarks .....	59
<b>4</b>	<b>DIY VMs</b> .....	<b>61</b>
4.1	Introduction .....	61
4.2	ALEX .....	62
4.2.1	Language Overview .....	62
4.2.2	What the Virtual Machine Must Support .....	65
4.2.3	Virtual Machine—Storage Structures .....	66
4.2.4	Virtual Machine—Registers .....	68
4.2.5	Virtual Machine—Instruction Set .....	70
4.2.6	An Example .....	79
4.2.7	Implementation .....	81
4.2.8	Extensions .....	85
4.2.9	Alternatives .....	88
4.2.10	Specification .....	93
4.3	Issues .....	96
4.3.1	Indirect and Relative Jumps .....	97
4.3.2	More Data Types .....	98
4.3.3	Higher-Order Routines .....	106
4.3.4	Primitive Routines .....	106
4.4	Concluding Remarks .....	107
<b>5</b>	<b>More Stack-Based VMs</b> .....	<b>109</b>
5.1	Introduction .....	109
5.2	A Simple Object-Oriented Language .....	110
5.2.1	Language Overview .....	110
5.2.2	Virtual Machine—Storage Structures .....	111
5.2.3	Virtual Machine—Registers .....	113
5.2.4	Virtual Machine—Instruction Set .....	113
5.2.5	Extensions .....	116
5.2.6	Alternatives .....	116
5.3	A Parallel Language .....	117
5.3.1	Language Overview .....	117
5.3.2	Virtual Machine—Storage Structures .....	119
5.3.3	Virtual Machine—Registers .....	121
5.3.4	Virtual Machine—Instruction Set .....	122
5.3.5	Implementation .....	124

5.3.6	Extensions .....	126
5.3.7	Alternatives .....	128
5.3.8	Issues .....	129
5.4	Concluding Remarks .....	129
5.4.1	Some Optimisations .....	129
5.4.2	Combining the Languages .....	130
<b>6</b>	<b>Case Study: An Event-Driven Language .....</b>	<b>131</b>
6.1	Introduction .....	131
6.2	The Structure of Rules .....	133
6.3	Events .....	136
6.4	Execution Cycle .....	136
6.5	Interpretation Rules .....	138
6.6	VM Specification .....	141
6.6.1	States and Notational Conventions .....	142
6.6.2	Infra-Rule Transitions .....	145
6.6.3	Extra-Rule Transitions .....	148
6.6.4	VM-Only Transitions .....	150
6.6.5	Introspective Operations .....	151
6.7	Rule Equivalences .....	153
6.8	Concluding Remarks .....	154
<b>7</b>	<b>Register-Based Machines .....</b>	<b>157</b>
7.1	Introduction .....	157
7.2	The Register-Transfer Model .....	158
7.3	Register Machine Organisation .....	161
7.4	Parrot—General Organisation .....	165
7.5	Parrot Instruction Set .....	168
7.5.1	Control instructions .....	169
7.5.2	Data management instructions .....	169
7.5.3	Register and stack operations .....	170
7.6	DIY Register-Based Virtual Machine .....	171
7.6.1	Informal Design .....	172
7.6.2	Extensions .....	176
7.6.3	Transition Rules .....	177
7.7	Translating ALEXVM into RTM .....	183
7.8	Example Code .....	186
7.9	Correctness of the Translation .....	186
7.10	More Natural Compilation .....	196
7.11	Extensions .....	200

<b>8</b>	<b>Implementation Techniques</b> .....	201
8.1	Stack-Based Machines .....	202
8.1.1	Direct Implementation .....	202
8.1.2	Translation .....	203
8.1.3	Threaded Code .....	207
8.2	Register Machines .....	209
8.2.1	Register sets .....	210
8.2.2	Addressing .....	210
8.2.3	Translation to Another VM .....	212
8.3	Using Transitions .....	212
8.4	Concluding Remarks .....	213
<b>9</b>	<b>Open Issues</b> .....	215
9.1	Security .....	215
9.2	New Languages .....	216
9.3	Typed Instruction Sets and Intermediate Codes .....	216
9.4	High-Level Instructions .....	218
9.5	Additivity and Replacement .....	218
9.6	Compiler Correctness .....	218
9.7	Dynamic Code Insertion .....	219
9.8	Instrumentation .....	220
9.9	Including more Information about Source Code .....	221
9.10	Integration with Databases .....	222
9.11	Increased Inter-Operability .....	222
9.12	Code Mobility .....	223
9.13	Small Platforms .....	224
9.14	Real-Time VMs .....	226
9.15	Code Morphing .....	227
9.16	Greater Optimisation .....	227
9.17	Operating System Constructs .....	228
9.18	Virtual Machines for more General Portability .....	229
9.19	Distributed VMs .....	229
9.20	Objects and VMs .....	229
9.21	Virtual VMs .....	230
9.22	By Way of a Conclusion .....	231
<b>A</b>	<b>Compiling ALEX</b> .....	233
A.1	Introduction .....	233
A.2	Notational Conventions .....	233
A.3	Compilation Rules .....	235
<b>B</b>	<b>Harrison Machine Compilation Rules</b> .....	241
B.1	Introduction .....	241
B.2	Compilation Rules .....	241

**C Harrison Machine Instruction Set**..... 257  
**References** ..... 261  
**Index** ..... 265