

Paul Cockshott and Kenneth Renfrew

SIMD Programming Manual for Linux and Windows



Springer

Contents

List of Tables	xvii
List of Figures	xix
List of Algorithms	xxiii
Introduction	xxv
I SIMD Programming	1
Paul Cockshott	
1 Computer Speed, Program Speed	3
1.1 Clocks	3
1.2 Width	4
1.3 Instruction Speed	5
1.4 Overhead Instructions	6
1.5 Algorithm Complexity	8
2 SIMD Instruction-sets	11
2.1 The SIMD Model	11
2.2 The MMX Register Architecture	12
2.3 MMX Data-types	13
2.4 3DNow!	15
2.4.1 Cache Handling	17
2.4.2 Cache Line Length and Prefetching	18
2.5 Streaming SIMD	19
2.5.1 Cache Optimisation	21
2.6 The Motorola AltiVec Architecture	22
3 SIMD Programming in Assembler and C	23
3.1 Vectorising C Compilers	23
3.1.1 Dead for Loop Elimination	24
3.1.2 Loop Unrolling	25
3.2 Direct Use of Assembler Code	25
3.2.1 The Example Program	26
3.3 Use of Assembler Intrinsics	27

3.4	Use of C++ Classes	27
3.5	Use of the Nasm Assembler	28
3.5.1	General Instruction Syntax	29
3.5.2	Operand Forms	29
3.5.3	Directives	32
3.5.4	Linking and Object File Formats	34
3.5.5	Summing a Vector	35
3.6	Coordinate Transformations Using 3DNow!	38
3.7	Coordinate Transformations Using SSE Instructions	44
4	Intel SIMD Instructions	47
4.1	Types	47
4.2	shrl	51
4.3	saturate	51
4.4	Instructions	51
4.4.1	ADDPS	52
4.4.2	ADDSS	52
4.4.3	ANDNPS	52
4.4.4	ANDPS	52
4.4.5	CMPPS	53
4.4.6	CMPSS	54
4.4.7	COMISS	54
4.4.8	CVTPI2PS	55
4.4.9	CVTPI2PS	55
4.4.10	CVTTPS2PI	55
4.4.11	CVTSI2SS	56
4.4.12	CVTSS2SI	56
4.4.13	CVTTSS2SI	56
4.4.14	DIVPD	56
4.4.15	DIVPS	57
4.4.16	DIVSD	57
4.4.17	DIVSS	57
4.4.18	EMMS	57
4.4.19	FXRSTOR	58
4.4.20	FXSAVE	58
4.4.21	MASKMOVQ	59
4.4.22	MAXPD	59
4.4.23	MAXPS	60
4.4.24	MAXSD	60
4.4.25	MAXSS	60
4.4.26	MINPD	61
4.4.27	MINPS	61
4.4.28	MINSD	61
4.4.29	MINSS	61
4.4.30	MOVAPS_load	62
4.4.31	MOVAPS_store	62
4.4.32	MOVD_load	62

4.4.33	MOVD_store	63
4.4.34	MOVD_load_sse	63
4.4.35	MOVD_store_sse	63
4.4.36	MOVHPS	63
4.4.37	MOVHPS_load	64
4.4.38	MOVHPS_store	64
4.4.39	MOVLHPS	64
4.4.40	MOVLPS_load	64
4.4.41	MOVLPS_store	64
4.4.42	MOVMSKPS	65
4.4.43	MOVNTPS	65
4.4.44	MOVNTQ	65
4.4.45	MOVQ_load	66
4.4.46	MOVQ_store	66
4.4.47	MOVSS_load	66
4.4.48	MOVSS_store	66
4.4.49	MOVUPS_load	67
4.4.50	MOVUPS_store	67
4.4.51	MULPD	67
4.4.52	MULPS	67
4.4.53	MULSD	68
4.4.54	MULSS	68
4.4.55	ORPS	68
4.4.56	PACKSSDW	69
4.4.57	PACKSSWB	69
4.4.58	PACKUSWB	69
4.4.59	PADDB	70
4.4.60	PADDB_sse	70
4.4.61	PADDW	70
4.4.62	PADDW_sse	71
4.4.63	PADD	71
4.4.64	PADD_sse	71
4.4.65	PADDQ	72
4.4.66	PADDQ_sse	72
4.4.67	PADDSB	72
4.4.68	PADDSB_sse	73
4.4.69	PADDUSB	73
4.4.70	PADDUSB_sse	74
4.4.71	PAND	74
4.4.72	PAND_sse	74
4.4.73	PANDN	75
4.4.74	PANDN_sse	75
4.4.75	PAVGB	75
4.4.76	PAVGB_sse	76
4.4.77	PAVGW	76
4.4.78	PAVGW_sse	76
4.4.79	PCMPEQB	77
4.4.80	PCMPEQB_sse	77

4.4.81	PCMPEQW	77
4.4.82	PCMPEQW_sse	78
4.4.83	PCMPEQD	78
4.4.84	PCMPEQD_sse	79
4.4.85	PCMPGTB	79
4.4.86	PCMPGTB_sse	79
4.4.87	PCMPGTW	80
4.4.88	PCMPGTW_sse	80
4.4.89	PCMPGTD	80
4.4.90	PCMPGTD_sse	81
4.4.91	PEXTRW	81
4.4.92	PEXTRW_sse	81
4.4.93	PINSRW	82
4.4.94	PMADDWD	82
4.4.95	PMAXSW	82
4.4.96	PMAXUB	83
4.4.97	PMINSW	83
4.4.98	PMINUB	84
4.4.99	PMOVMSKB	84
4.4.100	PMULHUW	84
4.4.101	PMULHW	85
4.4.102	PMULLW	85
4.4.103	POR	86
4.4.104	PREFETCHNTA	86
4.4.105	PREFETCHT1	86
4.4.106	PREFETCHT0	86
4.4.107	PSADBW	87
4.4.108	PSHUFD	87
4.4.109	PSHUFW	87
4.4.110	PSxxf	88
4.4.111	PSUBx	89
4.4.112	PSUBSx	89
4.4.113	PSUBUSx	90
4.4.114	PSWAPD	90
4.4.115	PUNPCKHBW	90
4.4.116	PUNPCKLBW	91
4.4.117	PUNPCKHWD	91
4.4.118	PUNPCKLWD	91
4.4.119	PUNPCKHDQ	92
4.4.120	PUNPCKLDQ	92
4.4.121	PXOR	92
4.4.122	RCPPS	93
4.4.123	RCPSS	93
4.4.124	RSQRTPS	93
4.4.125	RSQRTSS	94
4.4.126	SFENCE	94
4.4.127	SQRTPS	95
4.4.128	SQRTSS	95

4.4.129	SUBPS	95
4.4.130	SUBSS	96
4.4.131	UNPCKHPS	96
4.4.132	UNPCLPS	96
4.4.133	XORPS	97
5	3DNow Instructions	99
5.0.1	FEMMS	99
5.0.2	PF2ID	99
5.0.3	PFACC	99
5.0.4	PFADD	100
5.0.5	PFCMPEQ	100
5.0.6	PFCMPGT	100
5.0.7	PFCMPGE	101
5.0.8	PFMAX	101
5.0.9	PFCMPGE	101
5.0.10	PFMUL	102
5.0.11	PFNACC	102
5.0.12	PFPNACC	102
5.0.13	PFRCP	103
5.0.14	PFRCPIT	103
5.0.15	PFSUB	104
5.0.16	PFSUBR	104
5.0.17	PI2FD	105
5.0.18	PI2FW	105
5.0.19	PREFETCH	105
II	SIMD Programming Languages	107
	Paul Cockshott	
6	Another Approach: Data Parallel Languages	109
6.1	Operations on Whole Arrays	109
6.1.1	Array Slicing	111
6.1.2	Conditional Operations	113
6.1.3	Reduction Operations	114
6.1.4	Data Reorganisation	114
6.2	Design Goals	116
6.2.1	Target Machines	118
6.2.2	Backward Compatibility	119
6.2.3	Expressive Power	119
6.2.4	Run-time Efficiency	120
7	Basics of Vector Pascal	121
7.1	Formating Rules	121
7.1.1	Alphabet	121

7.1.2	Reserved Words and Identifiers	122
7.1.3	Character Case	124
7.1.4	Spaces and Comments.	124
7.1.5	Semicolons.	124
7.2	Base Types	125
7.2.1	Booleans.	125
7.2.2	Integer Numbers.	125
7.2.3	Real Numbers.	125
7.2.4	Characters and Strings.	126
7.3	Variables and Constants.	127
7.3.1	Declaration Order.	127
7.3.2	Constant Declarations	128
7.3.3	Variable Declarations.	129
7.3.4	Assignment.	129
7.3.5	Predefined Types.	129
7.4	Expressions and Operators	130
7.4.1	Arithmetic	130
7.4.2	Operations on Boolean Values	132
7.4.3	Equality Operators	133
7.4.4	Ordered Comparison.	133
7.5	Matrix and Vector Operations	135
7.5.1	Array Declarations	135
7.5.2	Matrix and Vector Arithmetic	136
7.5.3	Array Input/Output.	139
7.5.4	Array Slices	140
7.6	Vector and Matrix Products	142
7.6.1	Inner Product of Vectors.	142
7.6.2	Dot Product of Non-real Typed Vectors	144
7.6.3	Matrix to Vector Product	145
7.6.4	Data-flow Hazards	146
7.6.5	Matrix to Matrix Multiplication	148
7.7	Typography of Vector Pascal Programs	149
8	Algorithmic Features of Vector Pascal	151
8.1	Conditional Evaluation.	151
8.2	Functions	152
8.2.1	User-defined Functions	152
8.2.2	Procedures	155
8.2.3	Procedure ReadAndValidate	156
8.2.4	Function H.	157
8.2.5	Function Log2.	157
8.3	Branching.	157
8.3.1	Two-way Branches.	157
8.3.2	Multi-way Branches.	158
8.4	Unbounded Iteration	159
8.4.1	While	159
8.4.2	Repeat	160

8.5	Bounded Iteration	161
8.5.1	For to	161
8.5.2	For Downto	162
8.6	Goto	163
9	User-defined Types	165
9.1	Scalar Types	165
9.1.1	SUCC and PRED	166
9.1.2	ORD	168
9.1.3	Input/Output of Scalars	168
9.1.4	Representation	168
9.2	Sub-range Types	169
9.2.1	Representation	170
9.3	Dimensioned Numbers	170
9.3.1	Arithmetic on Dimensioned Numbers	173
9.3.2	Handling Different Units of Measurement	174
9.4	Records	175
9.5	Pointers	177
9.5.1	Pointer Idioms	179
9.5.2	Freeing Storage	181
9.6	Set Types	182
9.6.1	Set Literals	182
9.6.2	Operations on Sets	183
9.7	String Types	183
10	Input and Output	187
10.1	File Types	187
10.1.1	Binary Files	187
10.1.2	Text Files	188
10.1.3	Operating System Files	188
10.2	Output	190
10.2.1	Binary File Output	190
10.2.2	Text File Output	190
10.2.3	Generic Array Output	193
10.3	Input	193
10.3.1	Generic Array Input	193
10.3.2	Binary File Input	194
10.3.3	Text File Input	194
10.4	File Predicates	195
10.5	Random Access to Files	195
10.5.1	Seek	195
10.5.2	filepos	195
10.5.3	Untyped i/o	196
10.6	Error Conditions	196
11	Permutations and Polymorphism	197
11.1	Array Reorganisation	198
11.1.1	An Example	200

11.1.2	Array Shifts	200
11.1.3	Element Permutation	200
11.1.4	Efficiency Considerations	202
11.2	Dynamic Arrays	202
11.2.1	Schematic Arrays	203
11.3	Polymorphic Functions	204
11.3.1	Multiple Uses of Parametric Units	205
11.3.2	Function dategt	206
 III Programming Examples		209
Paul Cockshott		
12	Advanced Set Programming	211
12.1	Use of Sets to Find Prime Numbers	211
12.1.1	Set Implementation	212
12.2	Ordered Sets	213
12.2.1	openfiles	215
12.2.2	loadset	216
12.3	Sets of Records	218
12.3.1	Retrieval Operations	219
12.4	Use of Sets in Text Indexing	219
12.5	Constructing an Indexing Program	222
12.5.1	dirlist: A Program for Traversing a Directory Tree	222
12.5.2	intodir	223
12.6	bloomfilter	224
12.6.1	hashword	225
12.6.2	setfilter	225
12.6.3	testfilter	226
12.7	The Main Program to Index Files	226
12.7.1	processfile	227
12.7.2	A Retrieval Program	227
13	Parallel Image Processing	229
13.1	Declaring an Image Data Type	229
13.2	Brightness and Contrast Adjustment	229
13.2.1	Efficiency in Image Code	230
13.3	Image Filtering	231
13.3.1	Blurring	233
13.3.2	Sharpening	233
13.3.3	Comparing Implementations	235
13.4	genconv	238
13.4.1	dup	240
13.4.2	prev	241
13.4.3	pm	241
13.4.4	doedges	242
13.4.5	freestore	242

13.5	Digital Half-toning	242
13.5.1	Parallel Half-tone	244
13.5.2	errordifuse	245
13.6	Image Resizing	247
13.7	Horizontal Resize	249
13.8	Horizontal Interpolation	251
13.9	Interpolate Vertically	251
13.10	Displaying Images	251
13.10.1	demoimg – An Example Image Display Program	251
13.11	The Unit BMP	257
13.11.1	Procedure initbmpheader	260
13.11.2	Procedure storebmpfile	261
13.11.3	Function loadbmpfile	261
13.11.4	Procedure adjustcontrast	262
13.11.5	Procedure pconv	263
13.11.6	Procedure convp	264
14	Pattern Recognition and Image Compression	265
14.1	Principles of Image Compression	265
14.1.1	Data Compression in General	265
14.1.2	Image Compression	266
14.1.3	Vector Quantisation of Images	266
14.1.4	Data Structures	268
14.1.5	encode	269
14.2	The <i>K</i> Means Algorithm	271
14.2.1	Vector Quantisation of Colour Images	277
15	3D Graphics	279
15.1	Mesh Representation	280
15.2	linedemo: An Illustration of 3D Projection	282
15.3	demo3d: Main Procedure of linedemo	283
15.3.1	Viewing Matrices	283
15.3.2	SDL Initialisation	285
15.4	Create a Rotation Matrix	287
15.4.1	Calculate $x \bmod 3$	288
15.5	2D Projection	288
15.5.1	Entry Point to Line Drawing	289
15.6	Bresenham Line Drawing Procedure	290
15.7	Performance	292
IV	VIPER	293
	Ken Renfrew	
16	Introduction to VIPER	295
16.1	Rationale	295
16.1.1	The Literate Programming Tool	295
16.1.2	The Mathematical Syntax Converter	296

16.2	A System Overview.	296
16.3	Which VIPER to Download?	297
16.4	System Dependencies	297
16.5	Installing Files	298
16.6	Setting Up the Compiler.	298
16.7	Setting Up the System	298
	16.7.1 Setting System Dependencies.	299
	16.7.2 Personal Set-up	300
	16.7.3 Dynamic Compiler Options	301
	16.7.4 VIPER Option Buttons	303
16.8	Moving VIPER	303
16.9	Programming with VIPER	303
	16.9.1 Single Files	303
	16.9.2 Projects	304
	16.9.3 Embedding L ^A T _E X in Vector Pascal	306
16.10	Compiling Files in VIPER.	306
	16.10.1 Compiling Single Files	306
	16.10.2 Compiling Projects.	306
16.11	Running Programs in VIPER	307
16.12	Making VPT _E X.	307
	16.12.1 VPT _E X Options.	307
	16.12.2 VPMath.	308
16.13	L ^A T _E X in VIPER	308
16.14	HTML in VIPER	309
16.15	Writing Code to Generate Good VPT _E X.	309
	16.15.1 Use of Special Comments	309
	16.15.2 Use of Margin Comments.	310
	16.15.3 Use of Ordinary Pascal Comments.	311
	16.15.4 Levels of Detail Within Documentation	311
	16.15.5 Mathematical Translation: Motivation and Guidelines.	312
	16.15.6 L ^A T _E X Packages	313
Appendix A Compiler Porting Tools		315
A.1	Dependencies.	315
A.2	Compiler Structure.	316
	A.2.1 Vectorisation	317
	A.2.2 Porting Strategy	320
A.3	IILCG	321
A.4	Supported Types	321
	A.4.1 Data Formats	321
	A.4.2 Typed Formats	322
	A.4.3 ref Types	322
A.5	Supported Operations.	322
	A.5.1 Type Casts	322
	A.5.2 Arithmetic	322
	A.5.3 Memory.	322

A.5.4	Assignment	323
A.5.5	Dereferencing	323
A.6	Machine Description	323
A.6.1	Registers	323
A.6.2	Register Sets	324
A.6.3	Register Arrays	324
A.6.4	Register Stacks	324
A.6.5	Instruction Formats	325
A.7	Grammar of ILCG	325
A.8	ILCG Grammar	326
A.8.1	Helpers	326
A.8.2	Tokens	327
A.8.3	Non-terminal Symbols	329
Appendix B Software Download		335
Appendix C Using the Command Line Compiler		337
C.1	Invoking the Compiler	337
C.1.1	Environment Variable	337
C.1.2	Compiler Options	337
C.1.3	Dependencies	338
C.2	Calling Conventions	338
C.3	Array Representation	341
C.3.1	Range Checking	341
References		343
Index		345