

Contents

Preface	vii
PART I: Overview	1
1 Introduction	3
1.1 A User's View of Computer Systems	4
1.2 What Is Assembly Language?	5
1.3 Advantages of High-Level Languages	8
1.4 Why Program in the Assembly Language?	9
1.5 Typical Applications	10
1.6 Why Learn the Assembly Language?	11
1.7 Performance: C Versus Assembly Language	12
1.7.1 Multiplication Algorithm	12
1.8 Summary	14
1.9 Exercises	14
1.10 Programming Exercises	15
1.11 Program Listings	15
2 Basic Computer Organization	19
2.1 Basic Components of a Computer System	20
2.2 The Processor	22
2.2.1 The Execution Cycle	22
2.2.2 The System Clock	23
2.3 Number of Addresses	24
2.3.1 Three-Address Machines	24
2.3.2 Two-Address Machines	25
2.3.3 One-Address Machines	26
2.3.4 Zero-Address Machines	26

2.3.5	The Load/Store Architecture	26
2.3.6	Processor Registers	27
2.4	Flow of Control	28
2.4.1	Branching	28
2.4.2	Procedure Calls	30
2.5	Memory	32
2.5.1	Two Basic Memory Operations	33
2.5.2	Types of Memory	35
2.5.3	Storing Multibyte Data	37
2.6	Input/Output	38
2.7	Performance: Effect of Data Alignment	41
2.8	Summary	43
2.9	Exercises	43
PART II: Pentium Assembly Language		46
3	The Pentium Processor	47
3.1	The Pentium Processor Family	47
3.2	The Pentium Registers	49
3.2.1	Data Registers	50
3.2.2	Pointer and Index Registers	50
3.2.3	Control Registers	51
3.2.4	Segment Registers	53
3.3	Protected-Mode Memory Architecture	53
3.3.1	Segment Registers	54
3.3.2	Segment Descriptors	55
3.3.3	Segment Descriptor Tables	57
3.3.4	Segmentation Models	58
3.4	Real-Mode Memory Architecture	59
3.5	Mixed-Mode Operation	62
3.6	Which Segment Register to Use	63
3.7	Initial State	64
3.8	Summary	65
3.9	Exercises	65
4	Overview of Assembly Language	67
4.1	Assembly Language Statements	68
4.2	Data Allocation	69
4.3	Where Are the Operands?	74
4.3.1	Register Addressing Mode	75
4.3.2	Immediate Addressing Mode	75

4.3.3	Direct Addressing Mode	76
4.3.4	Indirect Addressing Mode	77
4.4	Data Transfer Instructions	78
4.4.1	The MOV Instruction	78
4.4.2	Ambiguous Moves	79
4.4.3	The XCHG Instruction	79
4.4.4	The XLAT Instruction	80
4.5	Overview of Assembly Language Instructions	80
4.5.1	Simple Arithmetic Instructions	80
4.5.2	Conditional Execution	83
4.5.3	Iteration Instruction	86
4.5.4	Logical Instructions	87
4.5.5	Shift Instructions	89
4.5.6	Rotate Instructions	91
4.6	Defining Constants	93
4.6.1	The EQU Directive	93
4.6.2	The %assign Directive	94
4.6.3	The %define Directive	94
4.7	Macros	95
4.8	Illustrative Examples	98
4.9	Performance: When to Use XLAT Instruction	108
4.9.1	Experiment 1	108
4.9.2	Experiment 2	110
4.10	Summary	110
4.11	Exercises	111
4.12	Programming Exercises	113
5	Procedures and the Stack	117
5.1	What Is a Stack?	118
5.2	Pentium Implementation of the Stack	118
5.3	Stack Operations	120
5.3.1	Basic Instructions	120
5.3.2	Additional Instructions	121
5.4	Uses of the Stack	123
5.4.1	Temporary Storage of Data	123
5.4.2	Transfer of Control	124
5.4.3	Parameter Passing	124
5.5	Procedures	124
5.6	Pentium Instructions for Procedures	126
5.6.1	How Is Program Control Transferred?	127
5.6.2	The ret Instruction	128

5.7	Parameter Passing	128
5.7.1	Register Method	129
5.7.2	Stack Method	132
5.7.3	Preserving Calling Procedure State	135
5.7.4	Which Registers Should Be Saved	136
5.7.5	ENTER and LEAVE Instructions	137
5.7.6	Illustrative Examples	138
5.8	Handling a Variable Number of Parameters	146
5.9	Local Variables	150
5.10	Multiple Source Program Modules	156
5.11	Performance: Procedure Overheads	159
5.11.1	Procedure Overheads	159
5.11.2	Local Variable Overhead	160
5.12	Summary	160
5.13	Exercises	162
5.14	Programming Exercises	163
6	Addressing Modes	167
6.1	Introduction	167
6.2	Memory Addressing Modes	169
6.2.1	Based Addressing	171
6.2.2	Indexed Addressing	171
6.2.3	Based-Indexed Addressing	173
6.3	Illustrative Examples	173
6.4	Arrays	180
6.4.1	One-dimensional Arrays	180
6.4.2	Multidimensional Arrays	181
6.4.3	Examples of Arrays	184
6.5	Performance: Usefulness of Addressing Modes	186
6.6	Summary	190
6.7	Exercises	191
6.8	Programming Exercises	192
7	Arithmetic Flags and Instructions	197
7.1	Status Flags	197
7.1.1	The Zero Flag	198
7.1.2	The Carry Flag	200
7.1.3	The Overflow Flag	203
7.1.4	The Sign Flag	205
7.1.5	The Auxiliary Flag	206
7.1.6	The Parity Flag	208

7.1.7	Flag Examples	209
7.2	Arithmetic Instructions	211
7.2.1	Multiplication Instructions	211
7.2.2	Division Instructions	215
7.3	Illustrative Examples	218
7.3.1	PutInt8 Procedure	218
7.3.2	GetInt8 Procedure	221
7.4	Multiword Arithmetic	224
7.4.1	Addition and Subtraction	224
7.4.2	Multiplication	225
7.4.3	Division	229
7.5	Performance: Multiword Multiplication	232
7.6	Summary	233
7.7	Exercises	234
7.8	Programming Exercises	235
8	Selection and Iteration	239
8.1	Unconditional Jump	239
8.2	Compare Instruction	242
8.3	Conditional Jumps	244
8.3.1	Jumps Based on Single Flags	244
8.3.2	Jumps Based on Unsigned Comparisons	246
8.3.3	Jumps Based on Signed Comparisons	247
8.3.4	A Note on Conditional Jumps	249
8.4	Looping Instructions	250
8.5	Implementing High-Level Language Decision Structures	252
8.5.1	Selective Structures	252
8.5.2	Iterative Structures	255
8.6	Illustrative Examples	257
8.7	Indirect Jumps	263
8.7.1	Multiway Conditional Statements	266
8.8	Summary	266
8.9	Exercises	268
8.10	Programming Exercises	269
9	Logical and Bit Operations	271
9.1	Logical Instructions	272
9.1.1	The <code>and</code> Instruction	272
9.1.2	The <code>or</code> Instruction	275
9.1.3	The <code>xor</code> Instruction	276
9.1.4	The <code>not</code> Instruction	278

9.1.5	The test Instruction	278
9.2	Shift Instructions	278
9.2.1	Logical Shift Instructions	279
9.2.2	Arithmetic Shift Instructions	281
9.2.3	Why Use Shifts for Multiplication and Division?	282
9.2.4	Doubleshift Instructions	283
9.3	Rotate Instructions	284
9.3.1	Rotate Without Carry	284
9.3.2	Rotate Through Carry	285
9.4	Logical Expressions in High-Level Languages	286
9.4.1	Representation of Boolean Data	286
9.4.2	Logical Expressions	286
9.4.3	Bit Manipulation	287
9.4.4	Evaluation of Logical Expressions	288
9.5	Bit Instructions	290
9.5.1	Bit Test and Modify Instructions	291
9.5.2	Bit Scan Instructions	291
9.6	Illustrative Examples	291
9.7	Summary	297
9.8	Exercises	297
9.9	Programming Exercises	299
10	String Processing	301
10.1	String Representation	301
10.1.1	Explicitly Storing String Length	302
10.1.2	Using a Sentinel Character	303
10.2	String Instructions	303
10.2.1	Repetition Prefixes	304
10.2.2	Direction Flag	305
10.2.3	String Move Instructions	306
10.2.4	String Compare Instruction	309
10.2.5	Scanning a String	311
10.3	Illustrative Examples	312
10.4	Testing String Procedures	321
10.5	Performance: Advantage of String Instructions	323
10.6	Summary	324
10.7	Exercises	325
10.8	Programming Exercises	326
11	ASCII and BCD Arithmetic	329
11.1	ASCII and BCD Representations of Numbers	330

11.1.1	ASCII Representation	330
11.1.2	BCD Representation	330
11.2	Processing in ASCII Representation	331
11.2.1	ASCII Addition	332
11.2.2	ASCII Subtraction	333
11.2.3	ASCII Multiplication	334
11.2.4	ASCII Division	334
11.2.5	Example: Multidigit ASCII Addition	335
11.3	Processing Packed BCD Numbers	336
11.3.1	Packed BCD Addition	336
11.3.2	Packed BCD Subtraction	338
11.3.3	Example: Multibyte Packed BCD Addition	338
11.4	Performance: Decimal Versus Binary Arithmetic	340
11.5	Summary	341
11.6	Exercises	342
11.7	Programming Exercises	344
PART III: MIPS Assembly Language		345
12	MIPS Processor	347
12.1	Introduction	347
12.2	Evolution of CISC Processors	349
12.3	RISC Design Principles	351
12.4	MIPS Architecture	354
12.5	Summary	358
12.6	Exercises	359
13	MIPS Assembly Language	361
13.1	MIPS Instruction Set	361
13.1.1	Instruction Format	362
13.1.2	Data Transfer Instructions	363
13.1.3	Arithmetic Instructions	364
13.1.4	Logical Instructions	368
13.1.5	Shift Instructions	369
13.1.6	Rotate Instructions	370
13.1.7	Comparison Instructions	371
13.1.8	Branch and Jump Instructions	371
13.2	SPIM Simulator	373
13.2.1	SPIM System Calls	373
13.2.2	SPIM Assembler Directives	375
13.2.3	MIPS Program Template	377

13.3	Illustrative Examples	378
13.4	Procedures	386
13.5	Stack Implementation	391
13.6	Summary	393
13.7	Exercises	394
13.8	Programming Exercises	395
PART IV: Pentium Interrupt Processing		399
14	Protected-Mode Interrupt Processing	401
14.1	Introduction	401
14.2	A Taxonomy of Interrupts	402
14.3	Interrupt Processing in the Protected Mode	403
14.4	Exceptions	407
14.5	Software Interrupts	409
14.6	File I/O	410
	14.6.1 File Descriptor	410
	14.6.2 File Pointer	410
	14.6.3 File System Calls	410
14.7	Illustrative Examples	414
14.8	Hardware Interrupts	418
14.9	Summary	419
14.10	Exercises	420
14.11	Programming Exercises	420
15	Real-Mode Interrupts	423
15.1	Interrupt Processing in the Real Mode	424
15.2	Software Interrupts	425
15.3	Keyboard Services	426
	15.3.1 Keyboard Description	426
	15.3.2 DOS Keyboard Services	427
	15.3.3 Extended Keyboard Keys	431
	15.3.4 BIOS Keyboard Services	434
15.4	Text Output to Display Screen	440
15.5	Exceptions: An Example	441
15.6	Direct Control of I/O Devices	444
	15.6.1 Accessing I/O Ports	444
15.7	Peripheral Support Chips	446
	15.7.1 8259 Programmable Interrupt Controller	446
	15.7.2 8255 Programmable Peripheral Interface Chip	448
15.8	I/O Data Transfer	449

15.8.1	Programmed I/O	450
15.8.2	Interrupt-driven I/O	452
15.9	Summary	457
15.10	Exercises	458
15.11	Programming Exercises	458
PART V: Advanced Topics		461
16 Recursion		463
16.1	Introduction	463
16.2	Recursion in Pentium Assembly Language	464
16.3	Recursion in MIPS Assembly Language	471
16.4	Recursion Versus Iteration	478
16.5	Summary	479
16.6	Exercises	479
16.7	Programming Exercises	479
17 High-Level Language Interface		483
17.1	Why Program in Mixed Mode?	484
17.2	Overview	484
17.3	Calling Assembly Procedures from C	486
17.3.1	Illustrative Examples	488
17.4	Calling C Functions from Assembly	493
17.5	Inline Assembly	495
17.5.1	The AT&T Syntax	496
17.5.2	Simple Inline Statements	497
17.5.3	Extended Inline Statements	498
17.5.4	Inline Examples	500
17.6	Summary	504
17.7	Exercises	504
17.8	Programming Exercises	504
18 Floating-Point Operations		507
18.1	Introduction	507
18.2	Floating-Point Unit Organization	508
18.2.1	Data Registers	508
18.2.2	Control and Status Registers	510
18.3	Floating-Point Instructions	512
18.3.1	Data Movement	513
18.3.2	Addition	514
18.3.3	Subtraction	514

18.3.4	Multiplication	515
18.3.5	Division	516
18.3.6	Comparison	517
18.3.7	Miscellaneous	518
18.4	Illustrative Examples	519
18.5	Summary	524
18.6	Exercises	524
18.7	Programming Exercises	525
Appendices		527
A	Internal Data Representation	529
A.1	Positional Number Systems	529
A.1.1	Notation	531
A.2	Number Systems Conversion	532
A.2.1	Conversion to Decimal	532
A.2.2	Conversion from Decimal	534
A.2.3	Conversion Among Binary, Octal, and Hexadecimal	536
A.3	Unsigned Integer Representation	538
A.3.1	Arithmetic on Unsigned Integers	539
A.4	Signed Integer Representation	545
A.4.1	Signed Magnitude Representation	546
A.4.2	Excess-M Representation	547
A.4.3	1's Complement Representation	547
A.4.4	2's Complement Representation	550
A.5	Floating-Point Representation	551
A.5.1	Fractions	552
A.5.2	Representing Floating-Point Numbers	555
A.5.3	Floating-Point Representation	556
A.5.4	Floating-Point Addition	560
A.5.5	Floating-Point Multiplication	561
A.6	Character Representation	561
A.7	Summary	563
A.8	Exercises	564
A.9	Programming Exercises	566
B	Assembling and Linking	567
B.1	Introduction	567
B.2	Structure of Assembly Language Programs	568
B.3	Input/Output Routines	569
B.4	Assembling and Linking	574

B.4.1	The Assembly Process	574
B.4.2	Linking Object Files	581
B.5	Summary	581
B.6	Web Resources	581
B.7	Exercises	581
B.8	Programming Exercises	582
C	Debugging Assembly Language Programs	583
C.1	Strategies to Debug Assembly Language Programs	583
C.2	Preparing Your Program	586
C.3	GNU Debugger	586
C.3.1	Display Group	587
C.3.2	Execution Group	592
C.3.3	Miscellaneous Group	595
C.3.4	An Example	595
C.4	Data Display Debugger	597
C.5	Summary	602
C.6	Web Resources	603
C.7	Exercises	603
C.8	Programming Exercises	603
D	SPIM Simulator and Debugger	605
D.1	Introduction	605
D.2	Simulator Settings	608
D.3	Running and Debugging a Program	610
D.3.1	Loading and Running	610
D.3.2	Debugging	610
D.4	Summary	613
D.5	Exercises	613
D.6	Programming Exercises	613
E	IA-32 Instruction Set	615
E.1	Instruction Format	615
E.1.1	Instruction Prefixes	615
E.1.2	General Instruction Format	617
E.2	Selected Instructions	618
F	MIPS/SPIM Instruction Set	651
G	ASCII Character Set	673
	Index	677



<http://www.springer.com/978-0-387-20636-3>

Introduction to Assembly Language Programming
For Pentium and RISC Processors

Dandamudi, S.P.

2005, XXIV, 692 p., Hardcover

ISBN: 978-0-387-20636-3