

Greg W. Scragg



COMPUTER ORGANIZATION

A TOP-DOWN APPROACH



McGRAW-HILL INTERNATIONAL EDITIONS
Computer Science Series

CONTENTS

Preface

xix

PART I The Higher-Level Language Model

1

Chapter 1 The Higher-Level Language Model

3

The Higher-Level Language Machine

3

Data Types

4

Simple Types

4

Structured Data Types

5

Variables, Constants, and Declarations

6

Executable Statements or Actions

8

Data Movement

8

Data Manipulation

10

Control Structures and Program Flow

12

Sequential (or Default)

12

Conditional Structures

12

Iteration Constructs

14

Modularity Constructs

17

Blocks

17

Subprograms

18

Documentation

20

The Higher-Level Language as Protector

21

Abstraction

21

Label and Naming Conventions

22

Hierarchical Nature

22

Preview

26

Summary	27
Suggested Reading	27
Exercises	28

PART II The Assembly Language Model

Chapter 2 Overview of the Assembly Language Model: A Higher-Level Language Perspective

Architectural Overview	33
Memory	34
Registers	34
Main Memory	35
Identifying Memory Locations	36
Organization of Data	36
The Arithmetic Logic Unit and Assembly Language Commands	39
Data Movement	40
Internal Data Transfer	40
External Data Transfer	41
A Longer Example	44
Data Manipulation	47
Arithmetic Operations	47
The Control Unit and Control Operations	47
Jumps	53
Conditional Jumps	54
Emulation of Hill Control Structures	55
Iteration	56
Subprograms	59
Local Variables, Modularity Constructs, and Good Programming	63
Built-in Subprograms	69
"Real" Assembly Languages	70
Generalizations	71
Memory and Data References	71
Identifier Names	73
Jump Mnemonics	73
Common Variations on Legal Semantics	74
Points to Ponder	76
Summary	77
Data Directives	77
Commands and Procedures	78
Exercises	78

Chapter 3 A Short Guide to Programming in Assembly Language

Why Program in Assembly Language?	81
Naked Programming	82

General Programming Guidelines	82
Problem Definition	84
Algorithm Development	84
Coding	85
Documentation	86
Testing and Debugging	86
Data Abstraction	87
Type-Checking	88
Global Variables	89
Data Errors	91
Procedural Abstraction	92
Control Structures	92
Global Program Structure	95
Documentation	95
Extending Language Capabilities	96
Errors	98
Classes of Error	99
Machine or Run-Time Errors	103
Logical Errors	106
Programming in "Real" Assembly Languages	107
Variations on Commands	107
Reading a Reference Manual	109
Finding Command Descriptions	110
Reading the Command Description	110
Summary	111
Exercises	112

Chapter 4	Numeric Representation and Computer Arithmetic	114
	<u>Internal Representation</u>	114
	Review of Decimal Notation	115
	Other Radices	116
	Binary Representation	117
	Conversion between Bases	120
	Multiplicative Method	120
	Additive Method	123
	Division Method	125
	Subtraction Method	127
	Hexadecimal Representations	127
	Storage and the Size of Numbers	130
	Negative Numbers	131
	First Attempt: Sign and Number	131
	A Second Attempt: One's Complement	132
	The Final Solution: Two's Complement	134
	Coexistence of Representations	136
	Implications for "Real" Languages	140
	Summary	141
	Exercises	141

Chapter 5	Structures: Data and Control	144
	Data Structures	144
	Details of Implementation: Addresses	145
	Pointers	145
	Arrays	147
	Stacks and Queues	154
	Record Structures	158
	Linked Lists	159
	Parameter Passing	161
	Control Structures	163
	Condition Codes and the Condition Code Register	163
	Simplified Test and Jump Situations	164
	Case	165
	Conditionals Involving Compound Tests	168
	Notes on "Real Machines"	170
	Operand Mismatch	170
	Machine Architectures	172
	Language Specifications	173
	Summary	174
	Exercises	174
PART III	The Machine Language Model	177
Chapter 6	The Machine Language Model	179
	Architectural Overview	179
	The Fetch/Execute Cycle	180
	Machine Language	181
	Operation (Opcode)	182
	Operands	183
	Size	185
	Mode	186
	Some Examples	189
	Implications for Real Machine Languages	192
	Architecture: Storage and Communication	193
	Registers	193
	The Bus	195
	An Expanded Fetch/Execute Cycle	195
	Implications for Other Instructions	202
	Data Manipulation Instructions	202
	Data Movement	203
	Control Operations	205
	The Condition Code Register and Conditional Jumps	207
	The Stack Pointer and Procedure Calls	213
	Jump-to-Subroutine	217
	Return	217
	Implications for the Assembly Language Model	217
	Types of Operands	217
	Understanding Errors	219

Implications for "Real" Languages	220
Summary	222
Exercises	222
Chapter 7 The Assembler and Related Software	225
Overview	225
The Assembler	226
The Two-Pass Assembler	228
Address Identifiers	228
Symbol Tables	229
The Two-Pass Solution	231
Executing the Assembled Program	233
Linkers and Loaders	236
Loaders	236
Linkers	240
Assembly, Addressing, and Machine Architecture	245
Binding	245
Relative Addressing	246
Based Addressing	248
Linking Higher-Level Language Programs to Assembly	251
Language Procedures	252
Directives to the Assembler	253
Variable and Constant Directives	254
Macros	258
Conditional Assembly	260
Miscellaneous Directives	262
Programming on Real Machines	262
Relative versus Absolute Addresses	262
Understanding Errors	262
Summary	264
Exercises	264
PART IV The Digital Logic Model	267
Chapter 8 Logical Operations	269
Boolean Algebra	269
Boolean Constructs in Higher-Level Languages	269
Basic Operations	270
Propositional Calculus	273
Constructed Operations	274
Logic Operations in Assembly Language	280
Boolean Logic and Control Concepts	281
Multiple-Bit Logical Operations	282
An Additional Logical Operation: xor	284
Uses of Logical Commands	284

Logic and Masks	285
And-Masks	286
Or-Masks	289
Xor Mask	291
Shift Operations	292
Variations	293
Shifts and Arithmetic	294
Rotate	294
Implications for "Real" Assembly Languages	296
Condition Codes	297
Further Variations	297
Summary	298
Exercises	298
Chapter 9 Digital Circuits	301
Introduction to Digital Circuits	301
Basic Constructions	302
Equivalence to Boolean Algebra	304
Building the Control Unit	306
The Bus	307
Signaling an Event in Time	308
Maintaining Values over Time	310
Signaling One Event of Many	313
Decoding Signals	314
Merging Signals	315
Constructing an ALU	316
Logical ALUs	317
The Adder ALU	317
A Logical Shifter	320
The Fetch/Execute Cycle Revisited	321
Time Requirements	323
Appendix: Why NANDS and NORS Rather than ANDS and ORS	325
Universal Operators	328
Transistors	329
Speed Factors	332
Summary	332
Exercises	332
Chapter 10 Memory	335
Primary Memory	335
A Quick Review of Register Memory Construction	336
RAM Memory	336
Reading from RAM Memory	337
Memory Write Operations	341
Interaction of Primary Memory and the Fetch/Execute Cycle	342
Bus Size and Memory Size as Conventional Machine Descriptions	345
Modern Variations on Primary Memory	347

The Memory Hierarchy	350
Cache Memory	351
Secondary Storage	355
Properties and Uses of Secondary Storage	355
Logical Organization of Secondary Memory	356
Access Methods	357
Random Access Devices	358
Sequential Access Devices	359
Data Migration	360
Error Conditions	361
Parity	361
Self-Correcting Codes	362
Hamming Codes	363
Core: An Historical Note	364
Summary	366
Exercises	367

PART V The Integrated System Model 369

Chapter 11 The Computer as a System 371

The Visible or Interface Services	373
Bootstraps	377
Process Termination	379
Normal Termination	379
Abnormal Termination	379
Traps and Interrupts	380
Interrupt Vectors	384
Priorities	385
System Calls	386
Virtual Machines: The Invisible Services	387
Time Sharing: A Virtual Machine Possessing	
Multiple Processors	388
Virtual Memory: Increasing the Apparent Size of RAM	392
Virtual Input and Output	395
Interrupt-Driven Programs	396
Postscript: Good Programming Practices Revisited	397
Summary	397
Suggest Reading	398
Exercises	398

Chapter 12 Input and Output 400

The Logic of I/O: Simple Read and Write Operations	401
First Attempt: I/O Registers	402
I/O Ports	406
Interrupt-Driven I/O	410
Buffers	415
Direct Memory Access	417

Devices for Human-Machine Communication	418
Keyboards	418
Screen Devices	418
Pointing Devices	422
Storage Devices	423
Random Access Devices	424
Sequential Access Devices	427
Intermachine Communication	429
Modems	429
Intercomputer Communication	430
Instrumentation and Control	430
Communications Links between Devices:	
A Potpourri of Communication Terms	431
Historical Notes	432
Punched Cards and Paper Tape	432
Pascal's Get and Put	433
Summary	435
Exercises	435

PART VI Extensions 439

Chapter 13 Debuggers, Debugging, and Antibugging 441

General Guidelines	442
Preparation	442
Assembly Error Correction Tools	442
Computer and Assembler Error Messages	443
Classes of Error	443
Thinking about Errors	444
Debuggers	444
Entering the Debugger	444
Basic Debugger Display	445
Examining Memory Content	446
Tracing	449
Using the Debugger Advantageously	452
Some Additional Powers	454
Common Causes of Error	455
Real Debuggers	455
Summary	456
Exercises	456

Chapter 14 Real Arithmetic 458

The Nature of Real Numbers	458
Scientific Notation	460
Computer Representations	462
Real Arithmetic Operations	465
Normal Forms and Standard Representations	467

Round-Off error	469
Numbers with Significantly Different Magnitudes	469
Decimal-to-Binary Conversion (and Back)	469
Underflow	470
Related Problems	470
Increased Precision	471
Co-processors	471
Summary	472
Exercises	472

Chapter 15 Variations on the von Neumann Model: Microprogrammed and RISC Machines 474

The Microprogrammed Model	476
Justification	477
Background, Review, and Observations	478
A Machine Language Interpreter	480
A Microprogrammed Computer	483
The Interpretation of the Machine Language Code:	
The μ -machine	486
Vertical versus Horizontal Microcode	488
A Simple Example	492
Reduced Instruction Set Computers	493
Potential Problems with the Microprogrammed Model	494
Empirical Studies of the Nature of Computation	495
Exploitable Properties of the Higher Models	496
Principles of RISC Architecture	497
Implementation of RISC Instructions	498
Pipelined Instructions	501
Delayed Instructions	502
Comparison	505
Summary	506
Suggested Reading	506
Exercises	506

Chapter 16 Alternatives to the von Neumann Model: Parallel Machines 508

Introduction	508
Semi-parallel Constructs	510
Communication between Processes	511
Topologies and Algorithms	512
Vector Processing	512
Pipelines	513
Grid or Matrix Problems	515
Trees	518
Hypercubes	520
Memory Organization	521
Shared Memory	522

A Compromise: Butterfly or Switching Networks	523
Architectures: Instruction and Data Configurations	523
Very Large Scale Parallelism	525
Efficiency	527
Summary	528
Suggested Reading	528
Exercises	529
Appendices	
1. Assembly Language Structure and Style Guide	530
Comments	531
Variables and Constants	533
Organization and Readability	533
Input/Output	534
Miscellaneous	534
2. Term Projects	536
Project I. Build an Emulator for the GEM Machine	
a. Build a clock-pulse level description	537
b. Build a "box"	537
c. Build the internal GEM machine	537
Notes and simplifying assumptions	538
Project II. Calculator: A Multi-step Project	539
Mini-Project 0: Pascal Calculator (Optional Starting Point, Chapter 1)	540
Mini-Project 1: Single-digit Calculator (Chapter 2)	541
Mini-Project 2: Controlled Calculator (Chapter 3)	541
Mini-Project 3: Multiple Digits (Chapter 4)	541
Mini-Project 4: More Control (Chapter 5)	541
Mini-Project 5: Multiple Operations (Chapter 5)	541
Mini-Project 6: Large Numbers (Chapter 8)	542
Mini-Project 7: Digital Logic (Chapter 9)	542
Mini-Project 8: Floating Point (Chapter 14)	543
Project III. A Microcode Emulation (A Variation on Project I)	543
Project IV. An Assembler or Disassembler	544
Assembler	544
Disassembler	545
Paired Projects	545
Index	546