Contents

Preface

Chapter 1  Introduction And Overview

1.1  The Importance Of Architecture  1
1.2  Learning The Essentials  1
1.3  Organization Of The Text  2
1.4  What We Will Omit  3
1.5  Terminology: Architecture And Design  3
1.6  Summary  3

PART I  Basics

Chapter 2  Fundamentals Of Digital Logic

2.1  Introduction  7
2.2  Electrical Terminology: Voltage And Current  7
2.3  The Transistor  8
2.4  Logic Gates  9
2.5  Symbols Used For Gates  10
2.6  Construction Of Gates From Transistors  11
2.7  Example Interconnection Of Gates  12
2.8  Multiple Gates Per Integrated Circuit  14
2.9  The Need For More Than Combinatorial Circuits  15
2.10  Circuits That Maintain State  15
2.11  Transition Diagrams  16
2.12  Binary Counters  17
2.13  Clocks And Sequences  18
2.14  The Important Concept Of Feedback  20
2.15  Starting A Sequence  22
2.16  Iteration In Software Vs. Replication In Hardware  22
2.17  Gate And Chip Minimization  23
2.18  Using Spare Gates  24
Chapter 7  CPUs: Microcode, Protection, And Processor Modes  

7.1 Introduction 95  
7.2 A Central Processor 95  
7.3 CPU Complexity 96  
7.4 Modes Of Execution 97  
7.5 Backward Compatibility 97  
7.6 Changing Modes 98  
7.7 Privilege And Protection 99  
7.8 Multiple Levels Of Protection 99  
7.9 Microcoded Instructions 100  
7.10 Microcode Variations 102  
7.11 The Advantage Of Microcode 102  
7.12 Making Microcode Visible To Programmers 103  
7.13 Vertical Microcode 103  
7.14 Horizontal Microcode 104  
7.15 Example Horizontal Microcode 105  
7.16 A Horizontal Microcode Example 107  
7.17 Operations That Require Multiple Cycles 108  
7.18 Horizontal Microcode And Parallel Execution 109  
7.19 Look-Ahead And High Performance Execution 110  
7.20 Parallelism And Execution Order 111  
7.21 Out-Of-Order Instruction Execution 111  
7.22 Conditional Branches And Branch Prediction 112  
7.23 Consequences For Programmers 113  
7.24 Summary 113
Chapter 8 Assembly Languages And Programming Paradigm

8.1 Introduction 115
8.2 Characteristics Of A High-level Programming Language 115
8.3 Characteristics Of A Low-Level Programming Language 116
8.4 Assembly Language 117
8.5 Assembly Language Syntax And Opcodes 118
8.6 Operand Order 120
8.7 Register Names 121
8.8 Operand Types 122
8.9 Assembly Language Programming Paradigm And Idioms 122
8.10 Assembly Code For Conditional Execution 123
8.11 Assembly Code For A Conditional Alternative 124
8.12 Assembly Code For Definite Iteration 124
8.13 Assembly Code For Indefinite Iteration 125
8.14 Assembly Code For Procedure Invocation 125
8.15 Assembly Code For Parameterized Procedure Invocation 126
8.16 Consequence For Programmers 127
8.17 Assembly Code For Function Invocation 128
8.18 Interaction Between Assembly And High-Level Languages 128
8.19 Assembly Code For Variables And Storage 129
8.20 Two-Pass Assembler 130
8.21 Assembly Language Macros 131
8.22 Summary 134

PART III Memories

Chapter 9 Memory And Storage

9.1 Introduction 137
9.2 Definition 137
9.3 The Key Aspects Of Memory 138
9.4 Characteristics Of Memory Technologies 138
9.5 The Important Concept Of A Memory Hierarchy 140
9.6 Instruction And Data Store 140
9.7 The Fetch-Store Paradigm 141
9.8 Summary 141
12.21 Instruction And Data Caches 197
12.22 Virtual Memory Caching And A Cache Flush 198
12.23 Implementation Of Memory Caching 199
12.24 Direct Mapping Memory Cache 200
12.25 Using Powers Of Two For Efficiency 201
12.26 Set Associative Memory Cache 202
12.27 Consequences For Programmers 203
12.28 Summary 204

PART IV I/O

Chapter 13 Input/Output Concepts And Terminology 207
13.1 Introduction 207
13.2 Input And Output Devices 207
13.3 Control Of An External Device 208
13.4 Data Transfer 209
13.5 Serial And Parallel Data Transfers 209
13.6 Self-Clocking Data 210
13.7 Full-Duplex And Half-Duplex Interaction 210
13.8 Interface Latency And Throughput 211
13.9 The Fundamental Idea Of Multiplexing 211
13.10 Multiple Devices Per External Interface 212
13.11 A Processor's View Of I/O 213
13.12 Summary 213

Chapter 14 Buses And Bus Architectures 215
14.1 Introduction 215
14.2 Definition Of A Bus 215
14.3 Processors, I/O Devices, And Buses 216
14.4 Proprietary And Standardized Buses 216
14.5 Shared Buses And An Access Protocol 217
14.6 Multiple Buses 217
14.7 A Parallel, Passive Mechanism 217
14.8 Physical Connections 217
14.9 Bus Interface 218
14.10 Address, Control, And Data Lines 219
14.11 The Fetch-Store Paradigm 220
14.12 Fetch-Store Over A Bus 220
14.13 The Width Of A Bus 220
Chapter 15  Programmed And Interrupt-Driven I/O  237

15.1  Introduction  237
15.2  I/O Paradigms  237
15.3  Programmed I/O  238
15.4  Synchronization  238
15.5  Polling  239
15.6  Code For Polling  239
15.7  Control And Status Registers  241
15.8  Processor Use And Polling  241
15.9  First, Second, And Third Generation Computers  242
15.10  Interrupt-Driven I/O  242
15.11  A Hardware Interrupt Mechanism  243
15.12  Interrupts And The Fetch-Execute Cycle  243
15.13  Handling An Interrupt  244
15.14  Interrupt Vectors  245
15.15  Initialization And Enabling And Disabling Interrupts  246
15.16  Preventing Interrupt Code From Being Interrupted  246
15.17  Multiple Levels Of Interrupts  246
15.18  Assignment Of Interrupt Vectors And Priorities  247
15.19  Dynamic Bus Connections And Pluggable Devices  248
15.20  The Advantage Of Interrupts  249
15.21  Smart Devices And Improved I/O Performance  249
15.22 Direct Memory Access (DMA) 250
15.23 Buffer Chaining 251
15.24 Scatter Read And Gather Write Operations 252
15.25 Operation Chaining 252
15.26 Summary 253

Chapter 16 A Programmer's View Of Devices, I/O, And Buffering 255

16.1 Introduction 255
16.2 Definition Of A Device Driver 256
16.3 Device Independence, Encapsulation, And Hiding 256
16.4 Conceptual Parts Of A Device Driver 257
16.5 Two Types Of Devices 258
16.6 Example Flow Through A Device Driver 258
16.7 Queued Output Operations 259
16.8 Forcing An Interrupt 261
16.9 Queued Input Operations 261
16.10 Devices That Support Bi-Directional Transfer 262
16.11 Asynchronous Vs. Synchronous Programming Paradigm 263
16.12 Asynchrony, Smart Devices, And Mutual Exclusion 264
16.13 I/O As Viewed By An Application 264
16.14 Run-Time I/O Libraries 265
16.15 The Library/Operating System Dichotomy 266
16.16 I/O Operations The OS Supports 267
16.17 The Cost Of I/O Operations 268
16.18 Reducing The System Call Overhead 268
16.19 The Important Concept Of Buffering 269
16.20 Implementation of Buffering 270
16.21 Flushing A Buffer 271
16.22 Buffering On Input 272
16.23 Effectiveness Of Buffering 272
16.24 Buffering In An Operating System 273
16.25 Relation To Caching 274
16.26 An Example: The Unix Standard I/O Library 274
16.27 Summary 274
Chapter 19 Assessing Performance

19.1 Introduction 311
19.2 Measuring Power And Performance 311
19.3 Measures Of Computational Power 312
19.4 Application Specific Instruction Counts 313
19.5 Instruction Mix 314
19.6 Standardized Benchmarks 315
19.7 I/O And Memory Bottlenecks 316
19.8 Boundary Between Hardware And Software 316
19.9 Choosing Items To Optimize 317
19.10 Amdahl's Law And Parallel Systems 317
19.11 Summary 318

Chapter 20 Architecture Examples And Hierarchy

20.1 Introduction 319
20.2 Architectural Levels 319
20.3 System-Level Architecture: A Personal Computer 320
20.4 Bus Interconnection And Bridging 321
20.5 Controller Chips And Physical Architecture 322
20.6 Virtual Buses 323
20.7 Connection Speeds 325
20.8 Bridging Functionality And Virtual Buses 325
20.9 Board-Level Architecture 325
20.10 Chip-Level Architecture 327
20.11 Structure Of Functional Units On A Chip 328
20.12 Summary 329
20.13 Hierarchy Beyond Computer Architectures 329

Appendix 1 Lab Exercises For A Computer Architecture Course

A1.1 Introduction 331
A1.2 Digital Hardware For A Lab 332
A1.3 Solderless Breadboard 332
A1.4 Using A Solderless Breadboard 333
A1.5 Testing 334
A1.6 Power And Ground Connections 334
A1.7 Lab Exercises 335
   1 Introduction and account configuration 337
   2 Digital Logic: Use of a breadboard 339
   3 Digital Logic: Building an adder from gates 341