

**INTERNATIONAL EDITION**

**THE 80x86 IBM PC AND  
COMPATIBLE COMPUTERS  
(VOLUMES I & II)**

**ASSEMBLY LANGUAGE,  
DESIGN, AND INTERFACING**

4th Edition

**Muhammad Ali Mazidi  
Janice Gillispie Mazidi**

# CONTENTS

## PREFACE TO VOLUMES I AND II

### CHAPTER 0: INTRODUCTION TO COMPUTING 1

#### SECTION 0.1: NUMBERING AND CODING SYSTEMS 2

Decimal and binary number systems	2
Converting from decimal to binary	2
Converting from binary to decimal	2
Hexadecimal system	3
Converting between binary and hex	4
Converting from decimal to hex	4
Converting from hex to decimal	4
Counting in base 10, 2, and 16	6
Addition of binary and hex numbers	6
2's complement	6
Addition and subtraction of hex numbers	7
Addition of hex numbers	7
Subtraction of hex numbers	7
ASCII code	8

#### SECTION 0.2: INSIDE THE COMPUTER 9

Some important terminology	9
Internal organization of computers	9
More about the data bus	10
More about the address bus	10
CPU and its relation to RAM and ROM	11
Inside CPUs	11
Internal working of computers	12

#### SECTION 0.3: BRIEF HISTORY OF THE CPU 13

CISC vs. RISC	14
---------------	----

### CHAPTER 1: THE 80x86 MICROPROCESSOR 18

#### SECTION 1.1: BRIEF HISTORY OF THE 80x86 FAMILY 19

Evolution from 8080/8085 to 8086	19
Evolution from 8086 to 8088	19
Success of the 8088	19
Other microprocessors: the 80286, 80386, and 80486	19

SECTION 1.2: INSIDE THE 8088/8086	21
Pipelining	21
Registers	22
SECTION 1.3: INTRODUCTION TO ASSEMBLY PROGRAMMING	23
Assembly language programming	24
MOV instruction	24
ADD instruction	25
SECTION 1.4: INTRODUCTION TO PROGRAM SEGMENTS	26
Origin and definition of the segment	27
Logical address and physical address	27
Code segment	27
Logical address vs. physical address in the code segment	28
Data segment	29
Logical address and physical address in the data segment	30
Little endian convention	31
Extra segment (ES)	32
Memory map of the IBM PC	32
More about RAM	32
Video RAM	33
More about ROM	33
Function of BIOS ROM	33
SECTION 1.5: MORE ABOUT SEGMENTS IN THE 80x86	33
What is a stack, and why is it needed?	33
How stacks are accessed	34
Pushing onto the stack	34
Popping the stack	34
Logical address vs. physical address for the stack	35
A few more words about segments in the 80x86	36
Overlapping	36
Flag register	37
Bits of the flag register	38
Flag register and ADD instruction	38
Use of the zero flag for looping	40
SECTION 1.6: 80x86 ADDRESSING MODES	41
Register addressing mode	41
Immediate addressing mode	41
Direct addressing mode	42
Register indirect addressing mode	42
Based relative addressing mode	43
Indexed relative addressing mode	43
Based indexed addressing mode	44
Segment overrides	44

SECTION 2.1: DIRECTIVES AND A SAMPLE PROGRAM	50
Segments of a program	50
Stack segment definition	51
Data segment definition	51
Code segment definition	52
SECTION 2.2: ASSEMBLE, LINK, AND RUN A PROGRAM	54
.asm and .obj files	55
.lst file	55
PAGE and TITLE directives	56
.crf file	56
LINKing the program	57
.map file	57
SECTION 2.3: MORE SAMPLE PROGRAMS	57
Analysis of Program 2-1	58
Various approaches to Program 2-1	60
Analysis of Program 2-2	62
Analysis of Program 2-3	62
Stack segment definition revisited	62
SECTION 2.4: CONTROL TRANSFER INSTRUCTIONS	64
FAR and NEAR	64
Conditional jumps	64
Short jumps	64
Unconditional jumps	66
CALL statements	66
Assembly language subroutines	67
Rules for names in Assembly language	67
SECTION 2.5: DATA TYPES AND DATA DEFINITION	69
80x86 data types	69
Assembler data directives	69
ORG (origin)	69
DB (define byte)	69
DUP (duplicate)	70
DW (define word)	70
EQU (equate)	71
DD (define doubleword)	71
DQ (define quadword)	72
DT (define ten bytes)	72
SECTION 2.6: SIMPLIFIED SEGMENT DEFINITION	73
Memory model	74
Segment definition	74

SECTION 2.7: EXE VS. COM FILES	76
Why COM files?	76
Converting from EXE to COM	77

CHAPTER 3: ARITHMETIC AND LOGIC INSTRUCTIONS AND PROGRAMS 82

SECTION 3.1: UNSIGNED ADDITION AND SUBTRACTION	83
Addition of unsigned numbers	83
CASE 1: Addition of individual byte and word data	83
Analysis of Program 3-1a	84
CASE 2: Addition of multiword numbers	85
Analysis of Program 3-2	86
Subtraction of unsigned numbers	87
SBB (subtract with borrow)	88

SECTION 3.2: UNSIGNED MULTIPLICATION AND DIVISION	88
Multiplication of unsigned numbers	88
Division of unsigned numbers	90

SECTION 3.3: LOGIC INSTRUCTIONS AND SAMPLE PROGRAMS	93
AND	93
OR	93
XOR	94
SHIFT	95
COMPARE of unsigned numbers	96
IBM BIOS method of converting from lowercase to uppercase	99
BIOS examples of logic instructions	100

SECTION 3.4 BCD AND ASCII OPERANDS AND INSTRUCTIONS	101
BCD number system	101
Unpacked BCD	102
Packed BCD	102
ASCII numbers	102
ASCII to BCD conversion	102
ASCII to unpacked BCD conversion	102
ASCII to packed BCD conversion	103
Packed BCD to ASCII conversion	104
BCD addition and subtraction	104
BCD addition and correction	104
DAA	105
Summary of DAA action	105
BCD subtraction and correction	105
Summary of DAS action	107
ASCII addition and subtraction	109
Unpacked BCD multiplication and division	110
AAM	110
AAD	110

SECTION 3.5: ROTATE INSTRUCTIONS	111
Rotating the bits of an operand right and left	111
ROR rotate right	111
ROL rotate left	112
RCR rotate right through carry	113
RCL rotate left through carry	113
SECTION 3.6: BITWISE OPERATION IN THE C LANGUAGE	114
Bitwise operators in C	114
Bitwise shift operators in C	115
Packed BCD-to-ASCII conversion in C	116
Testing bits in C	116

## CHAPTER 4: BIOS AND DOS PROGRAMMING IN ASSEMBLY AND C 121

SECTION 4.1: BIOS INT 10H PROGRAMMING	122
Monitor screen in text mode	122
Clearing the screen using INT 10H function 06H	123
INT 10H function 02: setting the cursor to a specific location	123
INT 10H function 03: get current cursor position	124
Changing the video mode	124
Attribute byte in monochrome monitors	125
Attribute byte in CGA text mode	125
Graphics: pixel resolution and color	127
INT 10H and pixel programming	128
Drawing horizontal or vertical lines in graphics mode	128
Changing the background color	129
SECTION 4.2: DOS INTERRUPT 21H	130
INT 21H option 09: outputting a string to the monitor	130
INT 21H option 02: outputting a character to the monitor	130
INT 21H option 01: inputting a character, with echo	130
INT 21H option 0AH: inputting a string from the keyboard	131
Inputting more than the buffer size	132
Use of carriage return and line feed	134
INT 21H option 07: keyboard input without echo	135
Using the LABEL directive to define a string buffer	136
SECTION 4.3: INT 16H KEYBOARD PROGRAMMING	139
Checking a key press	139
Which key is pressed?	139
SECTION 4.4: INTERRUPT PROGRAMMING WITH C	141
Programming BIOS interrupts with C/C++	141
Programming INT 21H DOS functions calls with C/C++	143
Accessing segment registers	144
Accessing the carry flag in int86 and intdos functions	144
Mixing C with Assembly and checking ZF	145
C function kbhit vs. INT 16H keyboard input	146

SECTION 5.1: WHAT IS A MACRO AND HOW IS IT USED?	151
MACRO definition	151
Comments in a macro	152
Analysis of Program 5-1	154
LOCAL directive and its use in macros	155
INCLUDE directive	158
SECTION 5.2: MOUSE PROGRAMMING WITH INT 33H	161
INT 33H	161
Detecting the presence of a mouse	161
Some mouse terminology	162
Displaying and hiding the mouse cursor	162
Video resolution vs. mouse resolution in text mode	163
Video resolution vs. mouse resolution in graphics mode	163
Getting the current mouse cursor position (AX=03)	163
Setting the mouse pointer position (AX=04)	166
Getting mouse button press information (AX=05)	166
Monitoring and displaying the button press count program	167
Getting mouse button release information (AX=06)	168
Setting horizontal boundary for mouse pointer (AX=07)	168
Setting vertical boundary for mouse pointer (AX=08)	168
Setting an exclusion area for the mouse pointer (AX=10)	169
Getting mouse driver information (version) (AX=24H)	169

## CHAPTER 6: SIGNED NUMBERS, STRINGS, AND TABLES 173

SECTION 6.1: SIGNED NUMBER ARITHMETIC OPERATIONS	174
Concept of signed numbers in computers	174
Signed byte operands	174
Positive numbers	174
Negative numbers	174
Word-sized signed numbers	175
Overflow problem in signed number operations	176
When the overflow flag is set in 8-bit operations	176
Overflow flag in 16-bit operations	177
Avoiding erroneous results in signed number operations	178
IDIV (Signed number division)	179
IMUL (Signed number multiplication)	180
Arithmetic shift	182
SAR (shift arithmetic right)	182
SAL (shift arithmetic left) and SHL (shift left)	182
Signed number comparison	182

<b>SECTION 6.2: STRING AND TABLE OPERATIONS</b>	<b>184</b>
Use of SI and DI, DS and ES in string instructions	185
Byte and word operands in string instructions	185
DF, the direction flag	185
REP prefix	186
STOS and LODS instructions	186
Testing memory using STOSB and LODSB	187
The REPZ and REPNZ prefixes	187
SCAS (scan string)	189
Replacing the scanned character	189
XLAT instruction and look-up tables	190
Code conversion using XLAT	190

## **CHAPTER 7: MODULES; MODULAR AND C PROGRAMMING**     **193**

<b>SECTION 7.1: WRITING AND LINKING MODULES</b>	<b>194</b>
Why modules?	194
Writing modules	194
EXTRN directive	194
PUBLIC directive	194
END directive in modules	195
Linking modules together into one executable unit	196
SEGMENT directive	198
Complete stack segment definition	198
Complete data and code segment definitions	198
Analysis of Program 7-2 link map	200
Modular programming and the new segment definition	201

<b>SECTION 7.2: SOME VERY USEFUL MODULES</b>	<b>203</b>
Binary (hex)-to-ASCII conversion	203
ASCII (decimal)-to-binary (hex) conversion	204
Binary-to-ASCII module	205
ASCII-to-binary module	207
Calling module	207

<b>SECTION 7.3: PASSING PARAMETERS AMONG MODULES</b>	<b>208</b>
Passing parameters via registers	208
Passing parameters via memory	208
Passing parameters via the stack	208

<b>SECTION 7.4: COMBINING ASSEMBLY LANGUAGE AND C</b>	<b>210</b>
Why C?	210
Inserting 80x86 assembly code into C programs	211
C programs that call Assembly procedures	212
C calling convention	213
How parameters are returned to C	214
New assemblers and linking with C	215
Passing array addresses from C to the stack	216
Linking assembly language routines with C	217

**CHAPTER 8: 32-BIT PROGRAMMING FOR 386 AND 486 MACHINES 220**

<b>SECTION 8.1: 80386/80486 MACHINES IN REAL MODE</b>	<b>221</b>
General registers are pointers in 386/486	222
386/486 maximum memory range in real mode: 1M	224
Accessing 32-bit registers with commonly used assemblers	224
Little endian revisited	226
<b>SECTION 8.2: SOME SIMPLE 386/486 PROGRAMS</b>	<b>226</b>
Adding 16-bit words using 32-bit registers	226
Adding multiword data in 386/486 machines	228
Multiplying a 32-bit operand by a 16-bit operand	229
32-bit by 16-bit multiplication using 8086/286 registers	229
<b>SECTION 8.3: 80x86 PERFORMANCE COMPARISON</b>	<b>231</b>
Running an 8086 program across the 80x86 family	231

**CHAPTER 9: 8088, 80286 MICROPROCESSORS AND ISA BUS 235**

<b>SECTION 9.1: 8088 MICROPROCESSOR</b>	<b>236</b>
Microprocessor buses	236
Data bus in 8088	236
Address bus in 8088	238
8088 control bus	238
Bus timing of 8088	239
Other 8088 pins	240
<b>SECTION 9.2: 8284 AND 8288 SUPPORTING CHIPS</b>	<b>242</b>
8288 bus controller	242
Input signals	242
Output signals	243
8284 clock generator	244
Input pins	244
Output signals	245
<b>SECTION 9.3: 8-BIT SECTION OF ISA BUS</b>	<b>248</b>
A bit of bus history	246
Local bus vs. system bus	247
Address bus	247
Data bus	248
Control bus	249
One bus, two masters	249
AEN signal generation	249
Control of the bus by DMA	250
Bus boosting	250
8-bit section of the ISA bus	250

**SECTION 9.4: 80286 MICROPROCESSOR 251**

Pin descriptions 252

**SECTION 9.5: 16-BIT ISA BUS 255**

Exploring ISA bus signals 255

Address bus 256

Data bus 256

Memory and I/O control signals 256

Other control signals 258

ODD and EVEN bytes and BHE 259

A20 gate and the case of high memory area (HMA) 260

**CHAPTER 10: MEMORY AND MEMORY INTERFACING 265****SECTION 10.1: SEMICONDUCTOR MEMORY FUNDAMENTALS 266**

Memory capacity 266

Memory organization 266

Speed 267

ROM (read-only memory) 267

PROM (programmable ROM) or OTP ROM 268

EPROM (erasable programmable ROM) 268

EEPROM (electrically erasable programmable ROM) 269

Flash memory 270

Mask ROM 271

RAM (random access memory) 271

SRAM (static RAM) 271

DRAM (dynamic RAM) 273

Packaging issue in DRAM 273

DRAM, SRAM and ROM organizations 275

NV-RAM (nonvolatile RAM) 276

**SECTION 10.2: MEMORY ADDRESS DECODING 276**

Simple logic gate as address decoder 278

Using the 74LS138 as decoder 279

**SECTION 10.3: IBM PC MEMORY MAP 280**

Conventional memory: 640K of RAM 281

BIOS data area 282

Video display RAM (VDR) map 282

ROM address and cold boot on the PC 283

**SECTION 10.4: DATA INTEGRITY IN RAM AND ROM 284**

Checksum byte 284

Checksum program 286

Use of parity bit in DRAM error detection 286

DRAM memory banks 286

Parity bit generator/checker in the IBM PC 288

74S280 parity bit generator and checker 288

<b>SECTION 10.5: 16-BIT MEMORY INTERFACING</b>	<b>289</b>
ODD and EVEN banks	289
Memory cycle time and inserting wait states	291
Accessing EVEN and ODD words	292
Bus bandwidth	293

<b>SECTION 10.6: ISA BUS MEMORY INTERFACING</b>	<b>295</b>
Address bus signals	295
Memory control signals	295
ISA bus timing for memory	299
8-bit memory timing for ISA bus	299
ROM duplicate and x86 PC memory map	301
Shadow RAM	302
DIMM and SIMM memory modules	302

## **CHAPTER 11: I/O AND THE 8255; ISA BUS INTERFACING** 309

<b>SECTION 11.1: 8088 INPUT/OUTPUT INSTRUCTIONS</b>	<b>310</b>
8-bit data ports	310
How to use I/O instructions	311

<b>SECTION 10.2: I/O ADDRESS DECODING AND DESIGN</b>	<b>312</b>
Using the 74LS373 in an output port design	312
IN port design using the 74LS244	312
Memory map I/O	314

<b>SECTION 11.3: I/O ADDRESS MAP OF X86 PCS</b>	<b>316</b>
Absolute vs. linear select address decoding	316
Prototype addresses 300 - 31FH in the x86 PC	316
Use of simple logic gates as address decoders	316
Use of 74LS138 as decoder	318
IBM PC I/O address decoder	318
Use of the 8255 in the IBM PC/XT	341
Port 61H and time delay generation	319

<b>SECTION 11.4: 8255 PPI CHIP</b>	<b>320</b>
Mode selection of the 8255A	321

<b>SECTION 11.5: PC INTERFACE TRAINER AND BUS EXTENDER</b>	<b>325</b>
PC I/O Bus Extender	325
Buffering 300 - 31F address range	326
Installing the PC Bus Extender and booting the PC	327
Failure to boot	327
PC Interface Trainer	327
Design of the PC Trainer	328
The role of H1 and H2	328
Connecting the Module Trainer to the PC and testing	328
Testing the 8255 port	329
Testing Port A	330

**SECTION 11.6: I/O PROGRAMMING WITH C/C++ AND VB 332**

Visual C/C++ I/O programming	332
Visual C++ output example	332
Visual C++ input example	332
I/O programming in Turbo C/C++	334
I/O programming in Linux C/C++	335
Linux C/C++ program with I/O functions	335

**SECTION 11.7: 8-BIT AND 16-BIT I/O TIMING IN ISA BUS 338**

8-bit and 16-bit I/O in ISA bus	338
I/O signals of the ISA bus	339
8-bit timing and operation in ISA bus	341
16-bit I/O operation and timing in ISA bus	342
16-bit data ports instruction	342
16-bit I/O timing and operation via ISA bus	342
I/O bus bandwidth for ISA	343
Interfacing 8-bit peripherals to a 16-bit data bus	344

**CHAPTER 12: INTERFACING TO THE PC: LCD, MOTOR, ADC, AND SENSOR 351**

**SECTION 12.1: INTERFACING AN LCD TO THE PC 352**

LCD operation	352
LCD pin descriptions	352
Sending commands to LCDs	353
Sending data to the LCD	355
Checking LCD busy flag	356
LCD cursor position	357
LCD programming in Visual C/C++	358
LCD timing and data sheet	358

**SECTION 12.2: INTERFACING A STEPPER MOTOR TO A PC 362**

Stepper motors	362
Step angle	363
Stepper motor connection and programming	364
Steps per second and RPM relation	365
The four-step sequence and number of teeth on rotor	365
Motor speed	366
Holding torque	366
Wave drive 4-step sequence	367

**SECTION 12.3: INTERFACING DAC TO A PC 368**

Digital-to-analog (DAC) converter	368
MC1408 DAC (or DAC 808)	369
Converting IOOUT to voltage in 1408 DAC	369
Generating a sine wave	369

**SECTION 12.4: INTERFACING ADC AND SENSORS TO THE PC 373**

- ADC devices 373
- ADC 804 chip 373
- Selecting an input channel 376
- ADC0848 connection to 8255 377
- Interfacing a temperature sensor to a PC 378
- LM34 and LM35 temperature sensors 378
- Signal conditioning and interfacing the LM35 to a PC 379

**CHAPTER 13: 8253/54 TIMER AND MUSIC 386**

**SECTION 13.1: 8253/54 TIMER DESCRIPTION AND INITIALIZATION 387**

- Initialization of the 8253/54 388
- Control word 388

**SECTION 13.2: IBM PC 8253/54 TIMER CONNECTIONS AND PROGRAMMING 391**

- Using counter 0 392
- Using counter 1 393
- Using counter 2 393
- Use of timer 2 by the speaker 394
- Turning on the speaker via PB0 and PB1 of port 61H 394
- Time delay for 80x86 PCs 394
- Creating time delays in 8088/86-based computers 395
- Time delays in 80x86 IBM PC for 286 and higher processors 395

**SECTION 13.3: GENERATING MUSIC ON THE IBM PC 397**

- Playing "Happy Birthday" on the PC 399

**SECTION 13.4: SHAPE of 8253/54 OUTPUTS 401**

- OUT0 pulse shape in IBM BIOS 401
- OUT1 pulse shape in IBM BIOS 402
- OUT2 pulse shape in IBM BIOS 402
- 8253/54 modes of operation 402
- Testing the 8255/54 timer of the PC Interface Trainer 407

**CHAPTER 14: INTERRUPTS AND THE 8259 CHIP 410**

**SECTION 14.1: 8088/86 INTERRUPTS 411**

- Interrupt service routine (ISR) 411
- Difference between INT and CALL instructions 412
- Categories of interrupts 413
- Hardware interrupts 413
- Software interrupts 413
- Interrupts and the flag register 414
- Processing interrupts 414
- Functions associated with INT 00 to INT 04 415

**SECTION 14.2: IBM PC AND DOS ASSIGNMENT OF INTERRUPTS 417**

- Examining the interrupt vector table of your PC 417
- Analyzing an IBM BIOS interrupt service routine 419
- INT 12H: checking the size of RAM on the IBM PC 419

**SECTION 14.3: 8259 PROGRAMMABLE INTERRUPT CONTROLLER 420**

- 8259 control words and ports 421
- Masking and prioritization of IR0 - IR7 interrupts 426
- OCW (operation command word) 426
- OCW1 (operation command word 1) 427
- OCW2 (operation command word 2) 427
- Importance of the EOI (end of interrupt) command 429
- OCW3 (operation command word 3) 429

**SECTION 14.4: USE OF THE 8259 CHIP IN THE IBM PC/XT 430**

- Interfacing the 8259 to the 8088 in IBM PC/XT computers 430
- Initialization words of the 8259 in the IBM PC/XT 431
- Sequences of hardware interrupts with the 8259 432
- Sources of hardware interrupts in the IBM PC/XT 433
- Sources of NMI in the IBM PC 433

**SECTION 14.5: INTERRUPTS ON 80286 AND HIGHER 80x86 PCs 436**

- IBM PC AT hardware interrupts 436
- 8259 in master mode 436
- 8259 in slave mode 437
- AT-type computers interrupt assignment 438
- Case of missing IRQs on the AT expansion slot 438
- 80x86 microprocessor generated interrupts (exceptions) 439
- Interrupt priority 441
- More about edge- and level-triggered interrupts 441
- Interrupt sharing in the x86 PC 442

**CHAPTER 15: DIRECT MEMORY ACCESSING; THE 8237 DMA CHIP 447**

**SECTION 15.1: CONCEPT OF DMA 448**

**SECTION 15.2: 8237 DMA CHIP PROGRAMMING 450**

- 8237's internal control registers 453
- Command register 453
- Status register 454
- Mode register 456
- Single mask register 457
- All mask register 457
- Master clear/temporary register 458
- Clear mask register 459

**SECTION 15.3: 8237 DMA INTERFACING IN THE IBM PC/XT 459**

- 8237 and 8088 connections in the IBM PC 459
- Channel assignment of the 8237 in the IBM PC/XT 463
- DMA page register 463
- DMA data transfer rate of the PC/XT 464

**SECTION 15.4: REFRESHING DRAM USING CHANNEL 0 OF THE 8237 465**

- Refreshing DRAM with the 8237 467
- Refreshing in the IBM PC/XT 467
- DMA cycle of channel 0 467

**SECTION 15.5: DMA IN 80x86-BASED PC AT-TYPE COMPUTERS 468**

- 8237 DMA #1 468
- 8237 DMA #2 469
- Points to be noted regarding 16-bit DMA channels 470
- DMA channel priority 471
- I/O cycle recovery time 471
- DMA transfer rate 472

**CHAPTER 16: VIDEO AND VIDEO ADAPTERS 477**

**SECTION 16.1: PRINCIPLES OF MONITORS AND VIDEO ADAPTERS 478**

- How to judge a monitor 478
- Dot pitch 480
- Dot pitch and monitor size 480
- Phosphorous materials 480
- Color monitors 481
- Analog and digital monitors 481
- Video display RAM and video controller 481
- Character box 482

**SECTION 16.2: VIDEO ADAPTERS AND TEXT MODE PROGRAMMING 484**

- CGA (color graphics adapter) 484
- Video RAM in CGA 484
- Attribute byte in CGA text mode 485
- MDA (monochrome display adapter) 486
- Video RAM in MDA 486
- Attribute byte in IBM MDA 487
- EGA (enhanced graphics adapter) 487
- EGA video memory and attribute 487
- MCGA (multicolor graphics array) 488
- VGA (video graphics array) 489
- Video memory and attributes in VGA 489
- Super VGA (SVGA) and other video adapters 491

**SECTION 16.3: TEXT MODE PROGRAMMING USING INT 10H 491**

- Finding the current video mode 491
- Changing the video mode 491
- Setting the cursor position (AH=02) 493
- Getting the current cursor position (AH=03) 493
- Scrolling the window up to clear the screen (AH=06) 493

Writing a character in teletype mode (AH=0E)	494
Writing a string in teletype mode (AH=13H)	495
Character generator ROM	495
How characters are displayed in text mode	497
Character definition table in VGA	498
Changing the cursor shape using INT 10H	498

#### SECTION 16.4: GRAPHICS AND GRAPHICS PROGRAMMING 501

Graphics: pixel resolution, color, and video memory	501
The case of CGA	501
The case of EGA	502
Video memory size and color relation for EGA	502
The case of VGA	502
Video memory size and color relation for VGA	503
The case of SVGA graphics	503
INT 10H and pixel programming	504
Drawing horizontal or vertical lines in graphics mode	504

### CHAPTER 17: SERIAL DATA COMMUNICATION AND THE 16450/8250/51 CHIPS 508

#### SECTION 17.1: BASICS OF SERIAL COMMUNICATION 509

Half- and full-duplex transmission	510
Asynchronous serial communication and data framing	511
Start and stop bits	511
Data transfer rate	512
RS232 and other serial I/O standards	513
RS232 pins	513
Other serial I/O interface standards	514
Data communication classification	514
Examining the RS232 handshaking signals	514

#### SECTION 17.2: ACCESSING IBM PC COM PORTS USING DOS AND BIOS 516

IBM PC COM ports	516
Using the DOS MODE command	517
Data COM programming using BIOS INT 14H	520

#### SECTION 17.3: INTERFACING THE NS8250/16450 UART IN THE IBM PC 522

8250 pin descriptions	522
The 8250 registers	524
Limitation of the 8250/16450 UART and 16550	530

#### SECTION 17.4: INTEL 8251 USART AND SYNCHRONOUS COMMUNICATION 531

Intel's 8251 USART chip	531
Synchronous serial data communication	531
SDLC (serial data link control)	535
Cyclic redundancy checks	535

## CHAPTER 18: KEYBOARD AND PRINTER INTERFACING 541

### SECTION 18.1: INTERFACING THE KEYBOARD TO THE CPU 542

Scanning and identifying the key 542

Grounding rows and reading the columns 543

### SECTION 18.2: PC KEYBOARD INTERFACING AND PROGRAMMING 546

Make and break 546

IBM PC scan codes 546

BIOS INT 16H keyboard programming 549

Hardware INT 09 role in the IBM PC keyboard 551

Keyboard overrun 552

Keyboard buffer in BIOS data area 552

BIOS keyboard buffer 553

Tail pointer 553

Head pointer 553

PC keyboard technology 553

### SECTION 18.3: PRINTER AND PRINTER INTERFACING IN THE IBM PC 554

Centronics printer interface pins 554

Data lines and grounds 556

Printer status signals 556

Printer control signals 556

IBM PC printer interfacing 557

Programming the IBM PC printer with BIOS INT 17H 559

What is printer time-out? 560

ASCII control characters 560

Inner working of BIOS INT 17H for printing a character 561

### SECTION 18.4: BIDIRECTIONAL DATA BUS IN PARALLEL PORTS 562

SPP 562

PS/2 562

How to detect a PS/2-type bidirectional data bus 563

EPP 563

ECP 563

Using an LPT port for output 564

LCD connection to the parallel port 564

Stepper motor connection to the parallel port 564

Data input buffering 566

BIOS data area and LPT I/O address 566

## CHAPTER 19: FLOPPY DISKS, HARD DISKS, AND FILES 570

### SECTION 19.1: FLOPPY DISK ORGANIZATION 571

Capacity of the floppy disk 572

Formatting disks 572

Disk organization 572

Looking into the boot record	573
Directory	577
Bootable and nonbootable disks	579
FAT (file allocation table)	580
How to calculate sector locations of the FAT and the directory	582

## SECTION 19.2: HARD DISKS 583

Hard disk capacity and organization	583
Partitioning	585
Hard disk layout	585
Hard disk boot record	585
Hard disk FAT	585
Clusters	585
Hard disk directory	585
Speed of the hard disk	585
Data encoding techniques in the hard disk	586
Interfacing standards in the hard disk	588
Interleaving	591
Low- and high-level formatting	592
Parking the head	592
Disk caching	592
Disk reliability	592

## SECTION 19.3: DISK FILE PROGRAMMING 593

File handle and error code	593
----------------------------	-----

## CHAPTER 20: THE 80x87 MATH COPROCESSOR 600

### SECTION 20.1: MATH COPROCESSOR AND IEEE FLOATING-POINT STANDARDS 601

IEEE floating point standard	601
IEEE single-precision floating-point numbers	602
IEEE double-precision floating-point numbers	602
Other data formats of the 8087	604

### SECTION 20.2: 80x87 INSTRUCTIONS AND PROGRAMMING 605

Assembling and running 80x87 programs on the IBM PC	605
Verifying the Solution for Examples 20-1 to 20-4	605
80x87 registers	607
Trig functions	612
Integer numbers	614

### SECTION 20.3: 8087 HARDWARE CONNECTIONS IN THE IBM PC/XT 616

8087 and 8088 connection in the IBM PC/XT	616
How the 8088 and 8087 work together in the IBM PC/XT	618

## SECTION 20.4: 80x87 INSTRUCTIONS AND TIMING 620

Real transfers	620
Integer transfers	621
Packed decimal transfers	621
Addition	621
Subtraction	621
Reversed subtraction	622
Multiplication	622
Division	622
Reversed division	622
Other arithmetic instructions	622
Compare instructions	623
Transcendental instructions	623
Constant instructions	624
Processor control instructions	625

## CHAPTER 21: 386 MICROPROCESSOR: REAL vs. PROTECTED MODE 631

### SECTION 21.1: 80386 IN REAL MODE 632

What happened to the 80186/188?	632
80186/88 instructions	632
80286 Microprocessor	634
Major changes in the 80386	634
80386 Real mode programming	635
32-bit registers	635
Which end goes first?	636
General registers as pointers	636
Scaled index addressing mode	637
Some new 386 instructions	639
MOVSX and MOVZX instructions	639
Bit scan instructions	640

### SECTION 21.2: 80386: A HARDWARE VIEW 641

Overview of pin functions of the 80386	642
Bus bandwidth in the 386	645
Data misalignment in the 386	646
I/O address space in the 386	646

### SECTION 21.3: 80386 PROTECTED MODE 647

Protection mechanism in the 386	647
Virtual memory	647
Segmentation and descriptor table	648
Local and global descriptor tables	651
64 Terabytes of virtual memory	651
Paging	652
Going from a linear address to a physical address	653
The bigger the TLB, the better	654
Virtual 8086 mode	654

**CHAPTER 22: HIGH-SPEED MEMORY INTERFACING AND CACHE 659**

**SECTION 22.1: MEMORY CYCLE TIME OF THE 80X86 660**

Introducing wait states into the memory cycle 660

**SECTION 22.2: PAGE, STATIC COLUMN, AND NIBBLE MODE DRAMS 662**

Memory access time vs. memory cycle time 662

Types of DRAM 662

DRAM (standard mode) 663

DRAM interfacing using the interleaving method 663

Interleaved drawback 665

Page mode DRAM 667

Static column mode 669

Nibble mode 669

Timing comparison of DRAM modes 671

**SECTION 22.3: CACHE MEMORY 672**

Cache organization 673

Fully associative cache 673

Direct-mapped cache 674

Set associative 676

Updating main memory 678

Write-through 678

Write-back (copy-back) 678

Cache coherency 679

Cache replacement policy 679

Cache fill block size 679

**SECTION 22.4: EDO, SDRAM, AND RAMBUS MEMORIES 680**

EDO DRAM: origin and operation 680

SDRAM (synchronous DRAM) 682

Synchronous DRAM and burst mode 682

SDRAM and interleaving 683

Rambus DRAM 683

Overview of Rambus technology 683

Rambus protocol for block transfer 684

**CHAPTER 23: 486, PENTIUM, PENTIUM PRO AND MMX 690**

**SECTION 23.1: THE 80486 MICROPROCESSOR 691**

Enhancements of the 486 691

CLK in the 80486 694

High memory area (HMA) and the 80486 695

386, 486 Performance comparison 695

More about pipelining 695

**SECTION 23.2: INTEL'S PENTIUM 697**

Features of the Pentium 699

Intel's overdrive technology 703

<b>SECTION 23.3: RISC ARCHITECTURE</b>	<b>704</b>
Features of RISC	704
Comparison of sample program for RISC and CISC	707
IBM/Motorola RISC	709

<b>SECTION 23.4: PENTIUM PRO PROCESSOR</b>	<b>710</b>
Pentium Pro: internal architecture	710
Pentium Pro is both superpipelined and superscalar	711
What is out-of-order execution?	711
Branch prediction	714
Bus frequency vs. internal frequency in Pentium Pro	714

<b>SECTION 23.5: MMX TECHNOLOGY</b>	<b>715</b>
DSP and multimedia	715
Register aliasing by MMX	715
Data types in MMX	716

<b>SECTION 23.6: PROCESSOR IDENTIFICATION IN INTEL X86</b>	<b>717</b>
Program to identify the CPU	717
CPUID instruction and MMX technology	718

## **CHAPTER 24: MS DOS STRUCTURE, TSR, AND DEVICE DRIVERS 724**

<b>SECTION 24.1: MS DOS STRUCTURE</b>	<b>725</b>
DOS genealogy	725
From cold boot to DOS prompt	725
DOS standard device names	728
More about CONFIG.SYS and how it is used	728
What is AUTOEXEC.BAT and how is it used?	729
Types of DOS commands	730

<b>SECTION 24.2: TSR AND DEVICE DRIVERS</b>	<b>731</b>
Executing but not abandoning the program	731
How to make a program resident	731
Invoking the TSR	732
Hooking into hardware interrupts	732
Replacing the CS:IP values in the interrupt vector table	732
Writing a simple TSR	732
TSR with hot keys	734
Hooking into timer clock INT 08	735
DOS is not reentrant	736
Device drivers	736
Device driver categories	737

## **CHAPTER 25: MS DOS MEMORY MANAGEMENT 740**

<b>SECTION 25.1: 80x86 PC MEMORY TERMINOLOGY AND CONCEPTS</b>	<b>741</b>
Conventional memory	741
Upper memory area	741

IBM standard using ROM space in the upper memory area	742
Expanded memory	743
Extended memory	746
High memory area (HMA)	746
Shadow RAM	748
DOS MEM command	748

## SECTION 25.2: DOS MEMORY MANAGEMENT AND LOADING HIGH 749

Loading high into HMA	749
Finding holes in the upper memory area	750
EMM386.EXE options and switches	751
Loading high TSR and device driver into upper memory area	754
Emulating expanded memory and using UMB in 386/486/Pentium PC	755
How expanded memory is accessed	756

## CHAPTER 26: IC TECHNOLOGY AND SYSTEM DESIGN CONSIDERATIONS 759

### SECTION 26.1: OVERVIEW OF IC TECHNOLOGY 760

MOS vs. bipolar transistors	760
Overview of logic families	761
The case of inverters	761
CMOS inverter	762
Input, output characteristics of some logic families	762
History of logic families	763
Recent advances in logic families	764
Evolution of IC technology in Intel's 80x86 microprocessors	765

### SECTION 26.2: IC INTERFACING AND SYSTEM DESIGN CONSIDERATIONS 766

IC fan-out	766
Capacitance derating	768
Power dissipation considerations	770
Dynamic and static currents	771
Power-down option and Intel's SL series	771
Ground bounce	771
Filtering the transient currents using decoupling capacitors	774
Bulk decoupling capacitor	774
Crosstalk	774
Transmission line ringing	775

### SECTION 26.3: DATA INTEGRITY AND ERROR DETECTION IN DRAM 776

Soft error and hard error	776
Mean time between failure (MTBF) and FIT for DRAM	777
Error detection and correction	778
ECL and gallium arsenide (GaAs) chips	780

## CHAPTER 27: ISA, EISA, MCA, LOCAL, AND PCI BUS 784

### SECTION 27.1: ISA, EISA, AND IBM MICRO CHANNEL 785

Master and slave	785
Bus arbitration	785
Bus protocol	785
Bus bandwidth	786
ISA buses	786
36-pin part of the ISA bus	789
Limitations of the ISA bus	791
IBM Micro Channel Architecture (MCA)	793
Major characteristics of MCA	794
EISA bus	795
EISA slot numbering	797
Bus performance comparison	798

### SECTION 27.2: VL BUS AND PCI LOCAL BUSES 799

Definition and merits of local bus	799
VL bus (VESA local bus) characteristics	801
PCI local bus	801
PCI local bus characteristics	801
Plug and play feature	804
PCI connector	804
PCI performance	804

## CHAPTER 28: PROGRAMMING DOS, BIOS, HARDWARE WITH C/C++ 808

### SECTION 28.1: BIOS & DOS INTERRUPT PROGRAMMING WITH C 809

Programming BIOS interrupts with C/C++	809
Finding the conventional memory size with INT 12H	811
INT 16H and keyboard access	812
Programming INT 21H DOS function calls with C/C++	812
Accessing segment registers	812
Accessing the carry flag in int86 and intdos functions	814

### SECTION 28.2: PROGRAMMING PC HARDWARE WITH C/C++ 815

Accessing 80x86 SEGMENT:OFFSET memory addresses	815
Accessing BIOS data area with C	815
Programming input/output ports with C/C++	816
Revisiting playing music	816
Accessing parallel printer's (LPT1) data bus with C	816
Finding memory above 1MB: the extended memory size	819
Programming the CMOS RAM real-time clock (RTC)	820
Accessing the CMOS RAM bytes	820
Programming CMOS RAM with C/C++	822

APPENDIX A: DEBUG PROGRAMMING	825
APPENDIX B: 80x86 INSTRUCTIONS AND TIMING	847
APPENDIX C: ASSEMBLER DIRECTIVES AND NAMING RULES	883
APPENDIX D: DOS INTERRUPT 21H AND 33H LISTING	898
APPENDIX E: BIOS INTERRUPTS	924
APPENDIX F: ASCII CODES	940
APPENDIX G: I/O ADDRESS MAPS	941
APPENDIX H: IBM PC/PS BIOS DATA AREA	952
APPENDIX I: DATA SHEETS	959
REFERENCES	967
INDEX	969