Instructor's Manual

to accompany

The Intel Microprocessors

8086/8088, 80186/80188, 80286, 80386, 80486,
Pentium, Pentium Pro Processor,
Pentium II, Pentium III, and Pentium 4

Seventh Edition

Barry B. Brey
DeVry University

PEARSON
Prentice Hall

Upper Saddle River, New Jersey
Columbus, Ohio

10 9 8 7 6 5 4 3 2 1

PEARSON
Prentice
Hall

ISBN 0-13-170448-6

Instructor's Manual for

The Intel Microprocessors

8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Pentium Pro Processor, Pentium II, Pentium III, and Pentium 4

Architecture, Programming, and Interfacing

Seventh Edition

Barry B. Brey
DeVry University

Preface,

This is the seventh edition of this text and since its inception there have been many changes in the coverage. The Intel architecture and the personal computer have proved to be resilient and ever improving technology with no end in sight. Over the years there have been many attempts at displacing this technology, but none have succeeded. What may not have been understood is that the hardware is relatively inexpensive, especially today, and software continues to become more expensive. Whether this is the best technology is a moot point. The software has caused it to survive and thrive and as time passes the assaults become fewer and weaker. The Intel architecture has truly become the standard to master.

In the beginning of this architecture we had a relatively primitive machine (8086/8088) that has evolved into a very powerful machine (Pentium 4 with hyper-technology). What the future holds is parallel processing and higher clock frequencies and applications that communicate through electromagnetic waves in place of wires. Even though I write of this wonderful technology I sometimes doubt my sanity since I first learned digital technology using vacuum tubes. I recall building my first decade counter using four dual triode vacuum tubes for the flip-flops, neon lamps as indicators, and a power supply voltage of 200 volts. I recall when the 7400 NAND gate first appeared for $19.95. I was amazed when the Intel 4004 appeared in 1971, a year after I started teaching digital electronics and computers. If you are relatively young, can you imagine what you will see in your lifetime in this incredible field?

I thank each and every one of you for your continued support. If you have any comments or suggestions, please do not hesitate to write because I do answer all my email.

bbrey@ee.net

or

bbrey@devrycols.edu

You might also enjoy visiting my website at:

http://members.ee.net/brey

The publisher also has a set power-point slides for this text for instructors only. If you need them contact your representative.

Chapter One

1. Charles Babbage
3. Herman Hollerith
5. To decipher the enigma code (Zuse's machine was electromechanical relay logic)
7. Intel Corporation
9. Grace Hopper
11. 8080
13. 8086
15. 4G bytes
17. 1995
19. The Pentium Pro through the Pentium 4
21. Complex Instruction Set Computer
23. 1024
25. 1025M
27. TPA (transient program area) and a system area
29. 640K bytes
31. 1M
33. The 80386 through the Pentium
35. The BIOS is the Basic I/O system that contains I/O software and the setup dialog for the computer.
37. The AT is an updated version of the XT.
39. 8-bit
41. Advanced Graphics Port
43. In DOS a driver is stored in the TPA and in Windows the system area
45. BIOS makes every computer compatible
47. The microprocessor is the controlling element in a microprocessor-based computer system.
49. Address bus
51. Causes a read from an I/O device
53. (a) DB defines a byte (8-bits) or bytes, (b) DW defines a word (16-bits) or words,
(c) DD defines a doubleword (32-bits) or doublewords, and (d) DQ defines a quadword (64-bits) or quadwords.
55. (a) 13.25, (b) 57.1875, (c) 43.3125, and (d) 7. 0.0625.
57. (a) 163.1875, (b) 297.75, (c) 172.859375, (d) 4011.1875, and (e) 3000.05078125
59. (a) 0.101_2, 0.5_8, and $0.A_{16}$, (b) 0.00000001_2, 0.002_8, and 0.01_{16}, (c) 0.10100001_2, 0.502_8, and $0.A1_{16}$, (d) 0.11_2, 0.6_8, and $0.C_{16}$, and (e) 0.1111_2, 0.74_8, and $0.F_{16}$
61. (a) C2, (b) 10FD, (c) BC, (d) 10, and (e) 8BA
63. (a) 0111 1111, (b) 0101 0100, (c) 0101 0001, and (d) 1000 0000
65. (a) 46 52 4F 47, (b) 41 72 63, (c) 57 61 74 65 72, and (d) 57 65 6C 6C
67. Unicode is the standard alphanumeric code since 1995 for Windows programming
69. (a) 0010 0000, (b) 1111 0100, (c) 0110 0100, and (d) 1010 0100
71. TEMP DB -34
73. (a) 34 12, (b) 22 A1, and (c) 00 B1
75. DATA1 DW 123AH

77. (a) –128, (b) +51, (c) –110, and (d) –118
79. (a) 0 01111111 10000000000000000000000
    (b) 1 10000010 01010100000000000000000
    (c) 0 10000101 10010001000000000000000
    (d) 0 10001001 00101100000000000000000

Chapter Two

1. The program visible registers are registers that can be specified in instructions.
3. EAX, EBX, ECX, EDX, ESP, EBP, EDI, and ESI
5. ECX, CX, and CL
7. INC and DEC
9. Odd
11. The 80386 through the Pentium 4
13. (a) 10000H–1FFFFH, (b) 12340H–2233FH, (c) 23000H–32FFFH, (d) E0000H–EFFFFH, and (e) AB000H–BAFFFH
15. 100000H
17. EAX, EBX, ECX, EDX, EDI, or ESI
19. Stack
21. (a) 23000H, (b) 1C000H, (c) CA000H, (d) 89000H, and (e) 1CC90H
23. Any location in the 4G bytes address range of the Pentium 4
25. 8,192
27. 01000000H–0100FFFFH
29. Descriptor number 4
31. Descriptor number 20H, local descriptor table, requested privilege level 1
33.

0000 0000	0000 0000
1001 0010	0010 0001
0000 0000	0000 0000
0000 0000	0001 1111

35. The global descriptor table register
37. The microprocessor accesses the descriptor structure and caches the base address, limit, and access rights for the protected mode segment
39. The GDTR addresses the global descriptor table
41. The leftmost bit of control register zero (CR0)
43. 4M
45. 30000000H

Chapter Three

1. (a) copies BX into AX (b) copies AX into BX (c) copies CH into BL (d) copies EBP into ESP (e) copies CS into AX
3. AX, BX, CX, DX, SP, BP, DI, and SI

5. CS, DS, ES, FS, GS, and SS
7. You may not move one segment register into another.
9. (a) MOV AL,12H (b) MOV AX,123AH (c) MOV CL,0CDH
   (d) MOV SI,1000H (e) MOV EBX, 1200A2H
11. The TINY model sets up a program to function in one segment and as a .COM file.
13. A label is a symbolic memory location.
15. A letter, @, $, -, or ?
17. No, as a command (.COM) program.
19. A displacement is a distance. In the instruction listed, the distance is 2000H byte above the start of the data segment.
21. (a) 03234H (b) 02300H (c) 02400H
23. MOV BYTE PTR DATA1, 3
25. MOV DWORD PTR DATA1, 3
27. (a) 11100H (b) 10100H (c) 21000H
29. (a) 12100H (b) 12350H (c) 12200H
31. (a) 15700H (b) 15400H (c) 13000H
33. (a) 03100H (b) 05100H (c) 07100H
35. An example is MOV AX, FIELDS.F3
37. 5 bytes. The first bytes is the opcode, the second and third are the offset address, and the fourth and fifth bytes are the segment address.
39. Since a segment is cyclic, that is the top and bottom touch, one location below the bottom is the top and one location above the top is the bottom. Because of this, a signed displacement can reach any location within the current segment.
41. A real mode far jump is a jump to any memory location in the real memory system.
43. (a) short (b) near (c) short (d) far
45. JMP NEAR PTR TABLE
47. The contents of the data segment memory word addressed by offset address DI is pushed onto the stack.
49. The instruction pushes all the doubleword-sized registers onto the stack.

Chapter Four

1. Opcode
3. The MOD field determines how the R/M field is used in an instruction.
5. Nothing special needs to be done in assembly language, but in machine language the register-size override prefix is used in the 16-bit mode to select 32-bit registers.
7. (a) Stack (b) Data (c) Data (d) Stack (e) Data
9. MOV BX,[BP+4C00H]
11. 67 66 8B 30
13.

```
        MOV  AX,1000H
        MOV  DS,AX
```

15. CS

17. EAX, EBX, ECX, EDX, ESP, EBP, EDI and ESI
19. The BH register is moved to memory location 020FFH and the BL register is moved to location 020FEH then SP is changed to 00FEH.
21. 2
23. The MOV DI,NUMB instruction copies the 16-bit number in the data segment location NUMB into DI while the LEA DI,NUMB loads DI with the offset address of location NUMB.
25. The MOV with the OFFSET directive.
27. LDS loads DS and LSS loads SS along with another 16-bti register for the offset address.
29. If the direction flag is cleared it selects auto-increment for the string instructions and if the direction flag is set is selects auto-decrement.
31. MOVS
33. The STOSB instruction copies AL into the extra segment memory location addressed by DI then DI is either incremented or decremented by one as dictated by the direction flag.
35. The REP prefix repeats a string instruction CX number of times.
37. DX register
39. Probably not in modern software.
41.

```
TABLE  DB     30H, 31H, 32H, 33H
       DB     34H, 35H, 36H, 37H, 38H, 39H
BCD2A  PROC  NEAR
       MOV    BX,OFFSET TABLE
       XLAT
       RET
BCD2A  ENDP
```

43. The OUT DX,AX instruction outputs AX to the 16-bit I/O port addressed by DX.
45. MOV AH,ES:[BX]
47. If the zero flag indicates a not equal condition DX is copied into CX.
49. DB reserves byte(s), DW reserves word(s), and DD reserves doubleword(s)
51. The EQU directive equates labels and values to labels.
53. Selects the model type for a program.
55. Full segments
57. The start of a procedure is indicated with PROC and the end by ENDP.
59. In the assembler the .686 follows the .MODEL statement to specify 16-bit instruction mode.
61.

```
COPS   PROC   NEAR
       MOV    AX,CS:DATA4
       MOV    BX,AX
       MOV    CX,BX
       MOV    DX,BX
       MOV    SI,DX
       RET
COPS   ENDP
```

Chapter Five

1. (a) ADD AX,BX (b) ADD AL,12H (c) ADD EBP,EDI (d) ADD CX,22H (e) ADD AL,[SI] (f) ADD FROG,CX

3. No you may not add segment registers.

5.
```
ADD  AH,AL
ADD  AH,BL
ADD  AH,CL
ADD  AH,DL
MOV  DH,AH
```

7.
```
MOV  EDI,ECX
ADD  EDI,EDX
ADD  EDI,ESI
```

9. INC SP

11. (a) SUB CX,BX (b) SUB DH,0EEH (c) SUB SI,DI (d) SUB EBP,3322H (e) SUB CH,[SI] (f) SUB FROG,AL

13.
```
SUB  AX,DI
SUB  AX,SI
SUB  AX,BP
MOV  BX,AX
```

15. DX and carry are subtracted from the word contents of the data segment memory location addressed by DI minus 4.

17. AX

19. In an 8-bit multiplication, if the most-significant 8 bits of the result are zero, both C and O flag bits equal zero. These flag bits show that the result is 8-bits wide (C = 0) or 16-bits wide (C = 1). In a 16-bit multiplication, if the most-significant 16-bits part of the product is 0, both C and O clear to zero. In a 32-bit multiplication, both C and O indicate that the most-significant 32 bits of the product are zero.

21. IMUL performs signed multiplication, while MUL performs unsigned multiplication.

23. DX is multiplied by 100H and the 16-bit product is in BX

25. DX

27. IDIV performs signed division and DIV performs unsigned division

29.
```
MOV  AL,BL
DIV  DL
MOV  DL,AL
MOV  DH,0
ADD  DX,DX
```

31. AAA, AAS, AAD, and AAM

33.
```
MOV  CX,10000
DIV  CX
MOV  [BX],AL
XCHG DX,AX
MOV  CX,100
DIV  CX
AAM
MOV  [BX+1],AH
MOV  [BX+2],AL
XCHG DX,AX
AAM
MOV  [BX+3],AH
MOV  [BX+4],AL
```

35. (a) AND BX,DX (b) AND DH,0EAH (c) AND DI,BP
(d) AND EAX,1122H (e) AND [BP],CX (f) AND DX,[SI–8]
(g) AND WHAT,AL

37. (a) OR AH,BL (b) OR ECX,88H (c) OR SI,DX (d) OR BP,1122H
(e) OR [BX],CX (f) OR AL,[BP+40] (g) OR WHEN,AH

39. (a) XOR AH,BH (b) XOR CL,99H (c) XOR DX,DI (d) XOR ESP,1A23H
(e) XOR [EBX],DX (f) XOR DI,[BP+60] (g) XOR DI,WELL

41. The only difference is that the logical product is lost after TEST.

43. NOT is one's complement and NEG is two's complement.

45. AL is compared with the byte contents of the extra segment memory location addressed by DI.

47. The D flag selects whether SI/DI are incremented (D = 0) or decremented (D = 1).

49. An equal condition or if CX decrements to 0.

51.

```
        MOV   DI,OFFSET LIST
        MOV   CX,300H
        CLD
        MOV   AL,66H
        REPNE SCASB
```

Chapter Six

1. A short jump can jump ahead in the memory up to 127 bytes or back in memory up to –128 bytes.
3. Far jump
5. ±2G bytes
7. A label followed by a colon identifies a near memory location.
9. IP/EIP and CS
11. A JMP DI jumps to the offset address stored in DI and a JMP [DI] jumps to the offset address stored at the data segment memory location addressed by DI.
13. C, Z, O, S, and P
15. The JO instruction jumps on an overflow condition.
17. JE, JNE, JA, JAE, JB, and JBE
19. When CX is zero.
21. CX
23. The LOOPE instruction decrements CX, if CX is not zero it next tests to see if the zero flag indicates a not zero condition and if its not zero a jump to the operand address occurs.
25.

```
              MOV  SI,OFFSET BLOCK
              MOV  UP,0
              MOV  DOWN,0
              MOV  CX,100H
              MOV  AL,42H
              CLD
        L1:   SCASB
              JE   L3
              JA   L2
              INC  DOWN
              JMP  L3
        L2:   INC  UP
        L3:   LOOP L1
```

27. An infinite loop is created.
29. A .BREAK can be used to break out of a .WHILE construct.
31. A procedure is a group or code that can be used many times from a program.
33. A RET instruction retrieves the return address from the stack so the microprocessor can return from a procedure.
35. PROC
37. The RET 6 deletes 6 bytes from the stack before returning from a procedure.
39.

```
MULT   PROC NEAR USES CX DX
       MOV  AX,SI
       MUL  DI
       MOV  CX,100H
       DIV  CX
       RET
MULT   ENDP
```

41. An interrupt is either a hardware or software initiated CALL to a procedure.
43. 256
45. An interrupt vector contains the offset address followed by the segment address in 4 bytes of memory.
47. The IRETD instruction pops the flags, a 32-bit offset address, and the protected mode selector for the CS register.
49. 100H–103H
51. WAIT
53. 16
55. ESC

Chapter Seven

1. No, macro sequences and dot commands are not supported by the inline assembler.
3. Labels are defined in the inline assembler exactly as they are in the assembler.
5. EAX
7. Dot commands are not usable in the inline assembler.
9. The program uses SI and SI is not saved by the inline assembler so it must be saved and restored using a PUSH and POP.
11. The main difference is that when using the 16-bit version a program should attempt to use only 8- and 16- bit registers, while when using the 32-bit version a program should attempt to use 8- and 32-bit registers.
13. The conio header allows the putch() getche() functions to be used in a program.
15. Embedded applications use different I/O than the PC so the conio library would not be used in an embedded application.
17. The disp procedure divides by the number base and saves the remainders to generate a number in any number base.
19. The PUBLIC statement identifies a label as being available outside of the module.
21. It defines that the GetIt function has a single integer passed to it and returns nothing.
23. A control is usually some visible object that is obtained from the tool bar or inserted as an ActiveX control.
25. It is a 32-bit pointer.

27. External procedures are defined using the extern prototype.
29. It uses a 32-bit (DWORD) number.
31.

```
int  RotateLeft3 (int number)
{
       if ( ( number & 0x20000000 ) == 0x20000000 )
       {
              number <<= 3;
              number |= 1;
       }
       else
              number <<= 3;
       return number;
}
```

33. Find the blue arrow at the top of the screen and click on it.
35. An ActiveX control is an object that is a collection of functions and member variables that control the behavior of some, usually, visible object.

Chapter Eight

1. Object (.obj)
3. Library (.lib)
5. EXTRN indicates that a label is outside of the current program module.
7. Only the function used from the library file are placed in a program.
9. A macro sequence is a short list of instruction placed in a program when the macro is invoked.
11.

```
ADD32  MACRO
       ADD  AX,CX
       ADC  BX,DX
       ENDM
```

13.

```
ADDLIST MACRO PARA1,PARA2
       PUSH AX
       PUSH DI
       PUSH SI
       PUSH BX
       MOV  BX,OFFSET PARA1
       MOV  DI,PARA2
       .REPEAT
              MOV AL,[DI]
              ADD AL,[BX]
              MOV [DI],AL
              INC DI
              INC BX
       .UNTILCXZ
       POP  CX
       POP  BX
       POP  DI
       POP  AX
       ENDM
```

15. The include directive allows a file containing macros to be included in a program.
17.

```
//add a class char variable called Random for a random number
BOOL CssssDlg::PreTranslateMessage(MSG* pMsg)
{
```

```
          Random++;            //crank random number
          if ( pMsg->message == WM_CHAR )
          {
                 unsigned int key = pMsg->wParam;          //get key
                 if ( key >= 'a' && key <= 'f' )           //make uppercase
                        key -= 32;
                 if ( (key < '0' || key > 'F') || (key > '9' && key < 'A') )
                        return true;                       //ignore these
                 pMsg->wParam = key;                       //replace key
          }
          return CDialog::PreTranslateMessage(pMsg);
   }
```

19.
```
   BOOL CssssDlg::PreTranslateMessage(MSG* pMsg)
   {
   if ( pMsg->message == WM_KEYDOWN && pMsg->wParam == 13 )  //is it enter?
   {
          CString temp;
          int temp1 = 0;
          char temp2;
          GetDlgItemText(IDC_EDIT1, temp);
          for ( int a = 0; a < temp.GetLength(); a++)
          {
                        temp2 = temp.GetAt(a);
                        _asm
                        {
                               mov  al,temp2   ;convert ASCII digit to binary
                               sub  al,30h
                               cmp  al,9
                               jbe  L1
                               sub  al,27h
L1:
                               shl  temp1,4
                               or  byte ptr temp1,al
                        }
          }
          SetDlgItemInt(IDC_EDIT2,temp1);
          return true;
   }
   else if ( pMsg->message == WM_CHAR )
   {
   unsigned int key = pMsg->wParam;
          if ( key >= 'A' && key <= 'F' )
                 key += 32;
          if ( key < '0' || key > 'a' || key > '9' && key < 'f' )
                 return true;
          pMsg->wParam = key;
   }
   return CDialog::PreTranslateMessage(pMsg);
   }
```

21. Refer to Table 8-2.
23. Catch message WM_LBUTTONDBLCLK or message WM_RBUTTONDBLCLK.
25. Magenta
27. AAM
29. Divide by 8 and save each remainder as significant digits in the octal answer.
31. 30H
33. Subtract 30 from each digit, multiply the result (initial value of 0) by 10, add a digit and continue this for all three digits.

35.

```
char GetIt (char temp)
{
      char lookup[] = {'0','1','2','3', '4','5','6','7',
      '8','9','A','B','C','D','E','F'};
      return lookup[temp];
}
```

37. The master file table contains descriptors that describe the location of the file or folder.

39. The boot record (track zero, sector zero) contains the bootstrap loader program. The bootstrap loader program loads the operating system from the disk into the system.

41. 4K

43. Unicode

45. 3

47.

```
CString FileName = "C:\\TEST.LST";
CFile File;
CFileException ex;

unsigned char Array[512];
unsigned int FileByteCount;

CWaitCursor wait;                //display hour glass

if ( !File.Open( FileName, CFile::modeRead, &ex ) )
{
      TCHAR Cause[255];          //if file error
      CString Str;
      ex.GetErrorMessage(Cause, 255);
      Str = _T("File could not be opened.  Error = ");
      Str += Cause;
      AfxMessageBox(Str);        //display error message
}
else                             //no file error
{
      FileByteCount = File.Read( Buffer, sizeof( Buffer ) );
      File.Close();

}

wait.Restore();                  //return to normal cursor
```

49. The remove function removes a file or folder from the disk.

51.

```
// This program assumes that the decimal number is in IDC_Edit1 (an edit
// control) and placed in IDC_Edit2 (an edit control) as a binary value.
void DispBin(void)
{
      CString temp;
      char temp1;
      int temp2 = GetDlgItemInt(IDC_EDIT1);
      for ( int a = 0; a < 32; a++ )
      {
            _asm
            {
                  shl temp2,1
                  mov al,30h
                  adc al,0
                  mov temp1,al
```

```
        }
        temp += temp1;
    }
    SetDlgItemText(IDC_EDIT2,temp);
}
```

53.

```
// Random is a class variable
// MaxNumber is a class variable
// Flag is a class bool variable initialized to false
// Numbers are display in a list box called List1;
BOOL CssssDlg::PreTranslateMessage(MSG* pMsg)
{
    Random++;           //crank random number
    if ( Random = MaxNumber + 1)
        Random = 0;
    if ( pMsg->message == WM_KEYDOWN && pMsg->wParam == 13 )
    {
        Flag = true;
        return true;
    }
    return CDialog::PreTranslateMessage(pMsg);
}
void draw(char numbersdrawn)
{
    if ( numbersdrawn < 21 )
    {
        Char temp[10];
        char array[20];     //at most 20 random numbers
        for ( int a = 0; a < numbersdrawn; a++ )
        {
            while (!flag);      //wait for enter
            Flag = false;
            array[a] = Random;
            if ( a != 0 )       //check for repeat
            {
                for ( int b = 0; b < a - 1; b++ )
                {
                    if ( array[a] = array[b] )
                    {
                        a--;
                        break;
                    }
                }
            }
            if ( b == a )
            {
                itoa(array[a], temp, 10);
                List1.InsertString(a, temp “ ”);
            }
        }
    }
}
```

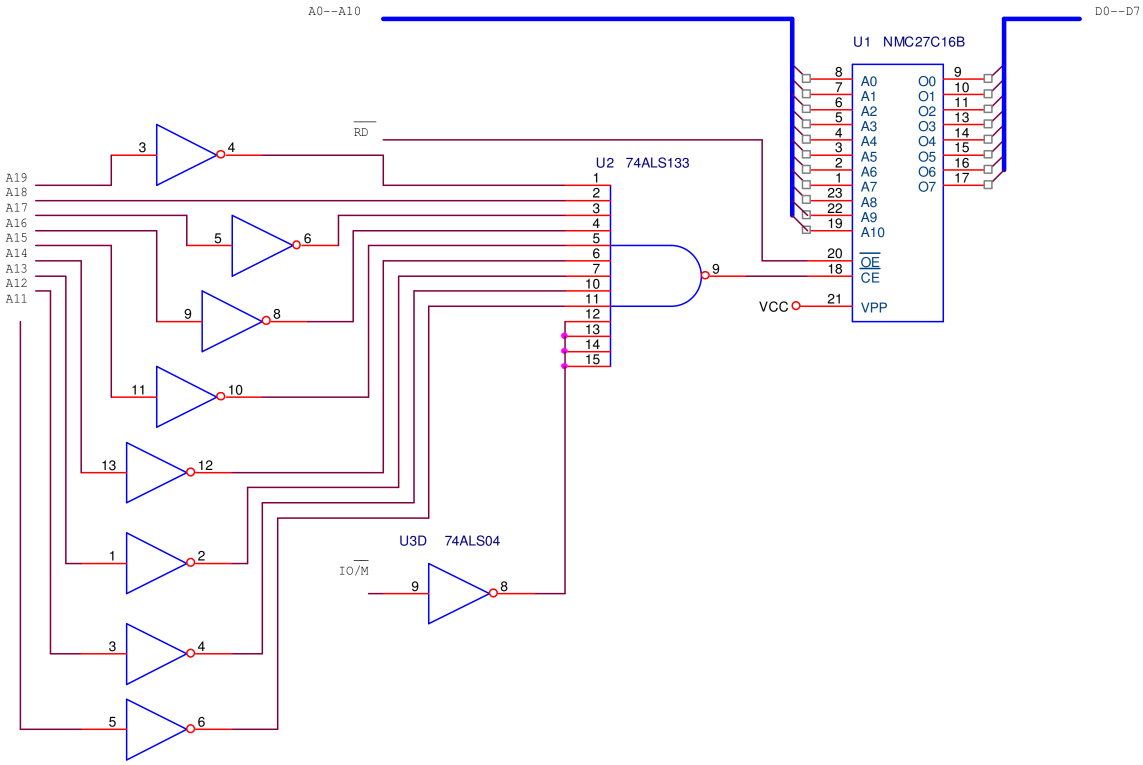
Chapter Nine

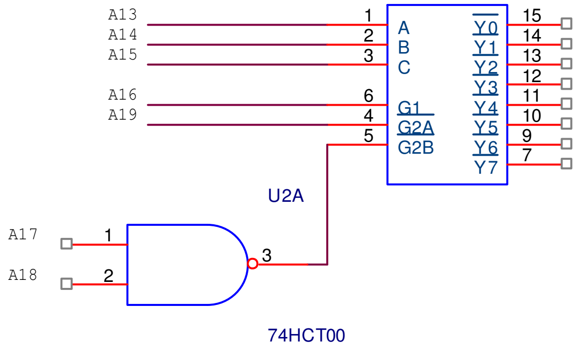
1. The main differences are the data bus width and the $IO/\overline{M}$ signal.
3. (a) 1 (b) 5 (c) 5
5. These bits indicate the segment being addressed by the current instruction.
7. The WAIT instruction waits for the $\overline{TEST}$ pin to become a logic zero.
9. Maximum mode
11. Never
13. During a HOLD, the microprocessor stops processing instructions and places the address, data, and controls buses at the high-impedance state.
15. The $\overline{LOCK}$ pin becomes a logic zero during instructions that pre prefixed with the LOCK: prefix.
17. The clock signal is provided, the RESET input is synchronized, and the READY input is synchronized.
19. EFI input
21. zero
23. Address/Data bus
25. The $\overline{BHE}$ signal is shared with a status bit (S7).
27. $DT/\overline{R}$
29. 1.0 μs
31. 2.5 MIPS
33. 600 ns – 110 ns – 30 ns = 460 ns
35. ∞
37. 0
39. It generates system control signals

Chapter Ten

1. All memory devices have address, data, and control connections.
3. (a) 2048 four bit numbers (b) 1024 one bit numbers (c) 4096 eight bit numbers (d) 16,384 one bit numbers (e) 65,536 four bit numbers
5. It causes the memory device to read data from a location.
7. (a) 1K (b) 2K (c) 4K (d) 8K (e) 64K
9. Flash memory requires an extended amount of time to accomplish an erase and write.
11. The $\overline{G}$ input cause a read, the $\overline{W}$ input causes a write, and the $\overline{S}$ input selects the chip.
13. Dynamic random access memory.
15. These inputs strobe the column and row addresses into a DRAM.
17. Memory rarely fills the entire memory, which requires some form of decoder to select the memory device for a specific range of memory addresses.

19.





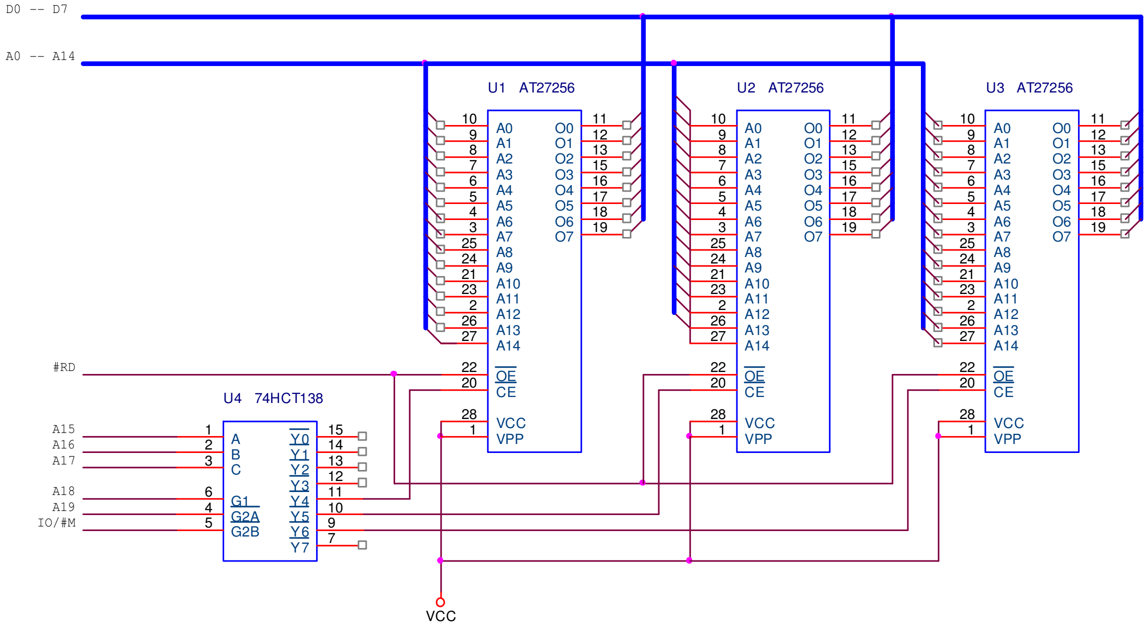
23. The 74LS139 is a dual 2-to-4 line decoder.

25. and or nand nor not

27.

```
begin

        ROM  <= A19 or (not A18) or A17 or MIO;
        RAM  <= A18 and A17 and (not MIO);
        AX19 <= not A19;

end V1;
```

29.



33. Single bit error flag.

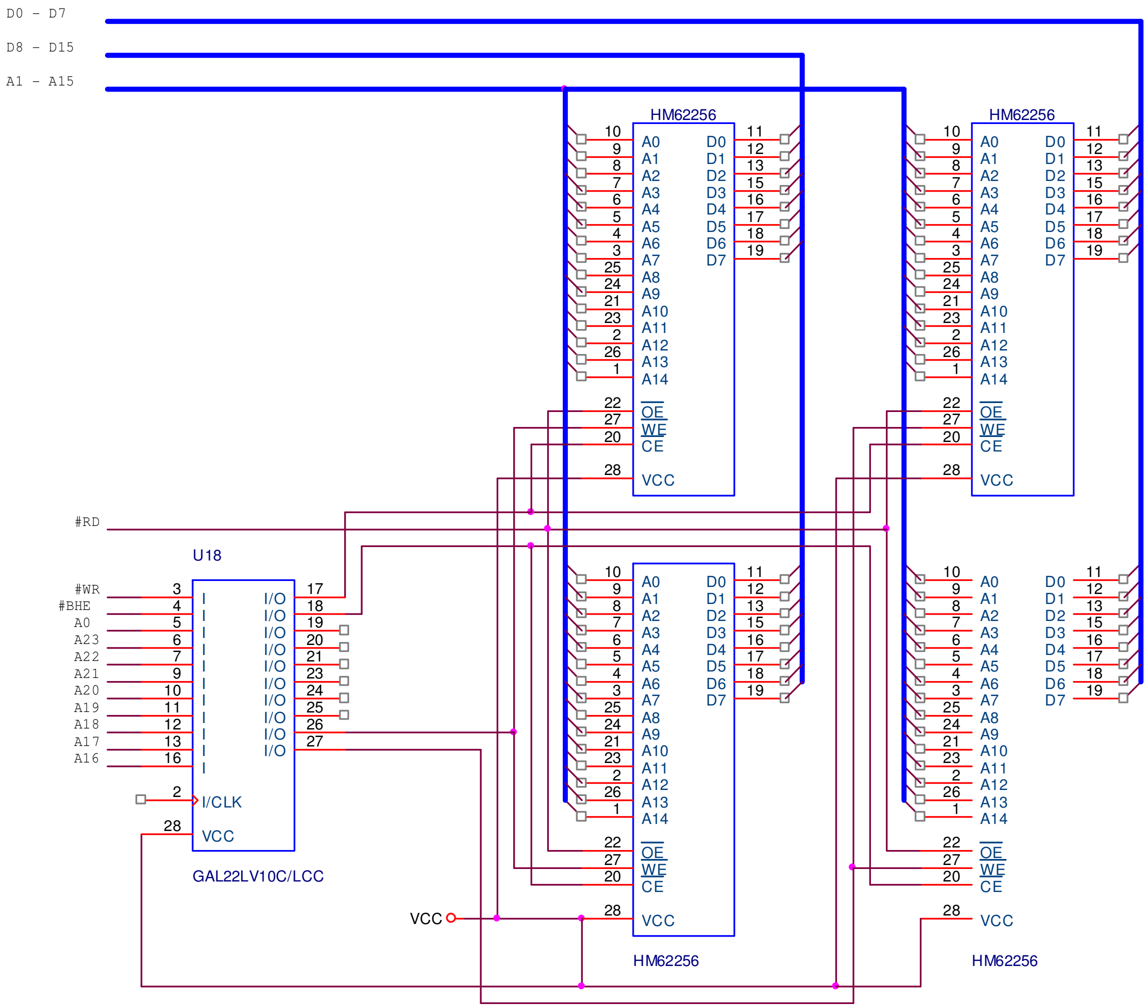
35. The main differences are the data bus size and the I/O, memory control signal.

37. Bank low enable has replaced the A0 pin.

39. Upper memory bank

41. It does not matter if 16-bit or 8-bit are read because the microprocessor just ignores any data bus bits that are not needed.

43.



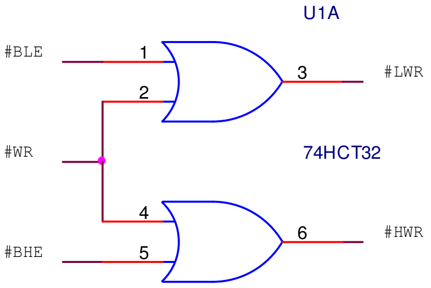
47. A cycle that does not read data, it only refreshes a row of memory.
49. 15.625 µs

Chapter Eleven

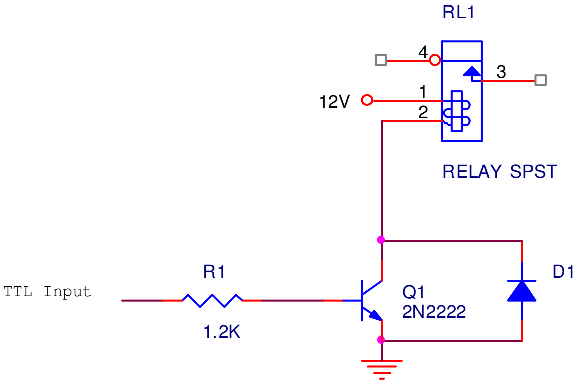
1. The IN instruction inputs data from an external device into the accumulator and the OUT instruction sends data out to an external device from the accumulator.
3. DX
5. AX
7. The INSW inputs data from the I/O port addressed by DX, as a word, into the extra segment memory location addressed by DI; it then increments DI by 2.
9. The basic input interface is a three-state buffer that is enabled for the IN instruction. When the buffer is enabled data is gated onto the data bus and into the accumulator.
11. Handshaking is the act of synchronizing two systems that operate asynchronously.

13. D8–D15

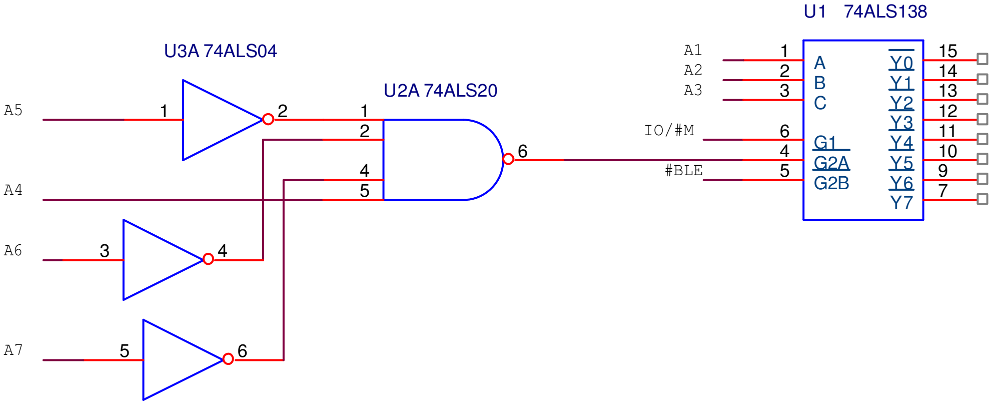
15.



17.



19.



21.

```
library ieee;
use ieee.std_logic_1164.all;

entity DECODER_21 is

port (
      A15, A14, A13, A12, A11, A10, A9, A8, A7, A6, A5, A4, A3, A2,
            A1: in STD_LOGIC;
      S1000, S1002, S1004, S1006, S1008, S100A, S100C, S100E: out STD_LOGIC
);

end;

architecture V1 of DECODER_21 is
```

```
begin

       S1000 <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or A3 or A2 or A1;
       S1002 <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or A3 or A2
              or (not A1);
       S1004 <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or A3 or (not A2)
              or A1;
       S1006 <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or A3 or (not A2)
              or (not A1);
       S1008 <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or (not A3) or A2
              or A1;
       S100A <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or (not A3) or A2
              or (not A1);
       S100C <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or (not A3) or (not A2)
              or A1;
       S100E <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or (not A3) or (not A2)
              or (not A1);

end V1;
```

23.

```
library ieee;
use ieee.std_logic_1164.all;

entity DECODER_23 is

port (
       BHE, A15, A14, A13, A12, A11, A10, A9, A8, A7, A6, A5, A4, A3, A2,
              A1: in STD_LOGIC;
       S300D, S300B, S1005, S1007: out STD_LOGIC
);

end;

architecture V1 of DECODER_23 is

begin

       S300D <= A15 or A14 or (not A13) or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or (not A3)
              or (not A2) or A1 or BHE;
       S300B <= A15 or A14 or (not A13) or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or (not A3)
              or A2 or (not A1) or BHE;
       S1005 <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or A3 or (not A2)
              or A1 or BHE;
       S1007 <= A15 or A14 or A13 or (not A12) or A11 or A10 or A11
              or A10 or A9 or A8 or A7 or A6 or A5 or A4 or A3 or (not A2)
              or (not A1) or BHE;

end V1;
```

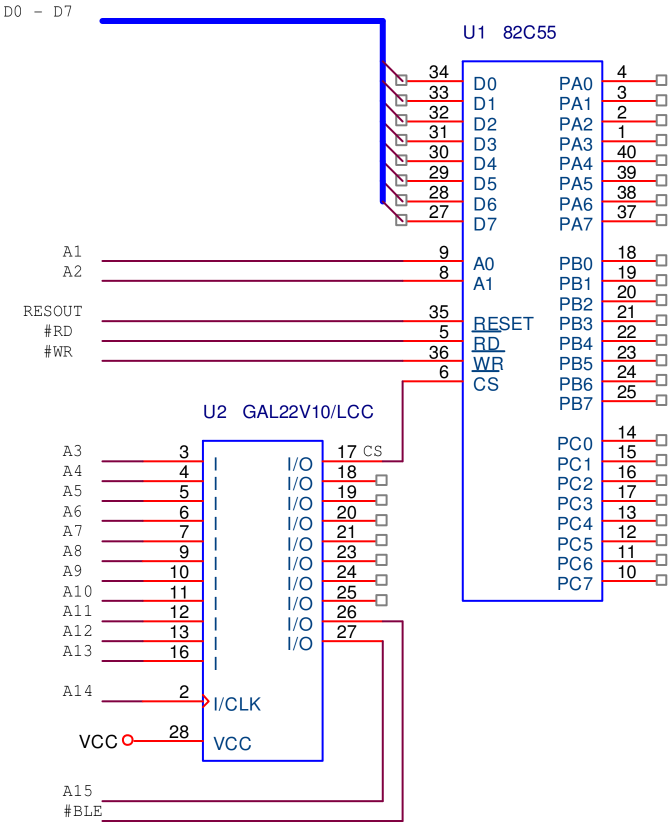
25. D0–D7

27. 24

29. A0 and A1

31.

```
library ieee;
use ieee.std_logic_1164.all;

entity DECODER_31 is

port (
        BLE, A15, A14, A13, A12, A11, A10, A9, A8, A7, A6, A5, A4, A3:
                in STD_LOGIC;
        CS: out STD_LOGIC
);

end;

architecture V1 of DECODER_31 is

begin

        CS <= A15 or A14 or A13 or A12 or A11 or A10 or (not A9) or (not A8)
                or (not A7) or A6 or A5 or A4 or A3 or BLE

end V1;
```



33. Modes 0, 1, and 2.

35.

```
DELAY  PROC   NEAR
       MOV  ECX, 479904
       .REPEAT
       .UNTIL ECX == 0
       RET
DELAY  ENDP
```

37. The 4-coil stepper is moved by activating (passing current through) a single coil at a time in round-robin fashion to move the armature a step at a time.

39.

```
IN   AL,PORTC
OR   AL,80H
OUT  PORTA,AL
```

41. The ACK signal is used by the I/O device to inform the 8255 that the output data has been processed by the output device.

43.

```
IN    AL,PORTC
BT    AL,4
JZ    IF_ZERO
```

45. PC0, PC1, and PC2

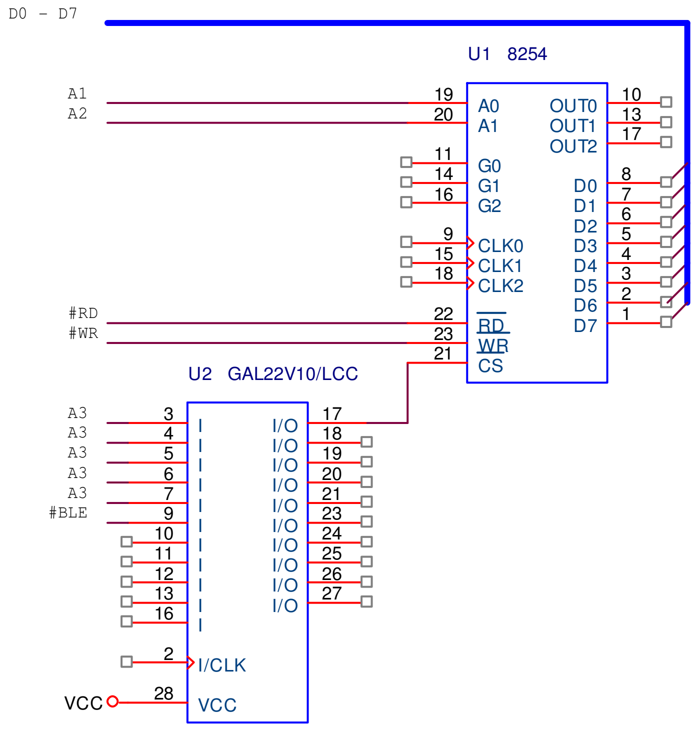
47. A display position is select by sending a command that contains the 7-bit address with the 8th and most significant bit set.

49. A read command is issued and the leftmost bit of the data read from the LCD display is the BUSY bit.

51. 10–20 ms.

53. 10 MHz

55.



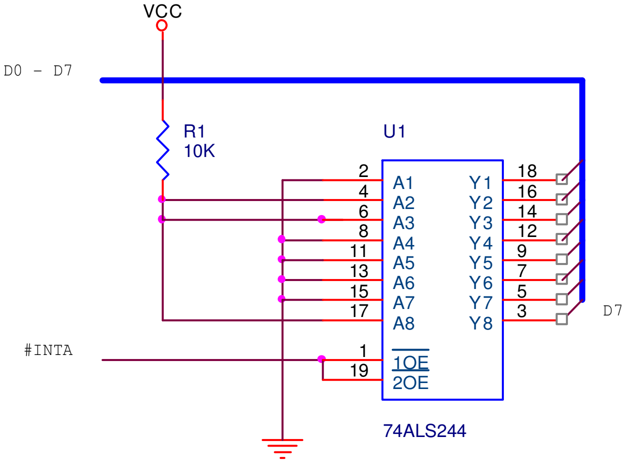
```
library ieee;
use ieee.std_logic_1164.all;

entity DECODER_55 is

port (
      BLE, A7, A6, A5, A4, A3: in STD_LOGIC;
      CS: out STD_LOGIC
);

end;

architecture V1 of DECODER_55 is

begin

      CS <= A7 or A6 or A5 or (not A4) or A3 or BLE

end V1;
```

57. 300
59. The counter is latched then the counter read-back control reads the counter at the time of the latching.
61. The motor attempts to move forward and reverse for equal amounts of time. This causes it to remain stationary.
63. The number of transmitted bits per second including data, start, stop and any other bits that are transferred.
65. 614,400 Hz
67. The MR input pin resets the device.
69. 1.0 μs
73. 100 μs
75. Start conversion

Chapter Twelve

1. An interrupt interrupts the currently executing program.
3. The interrupt service procedure is called by an interrupt.
5. NMI, INTR, and $\overline{\text{INTA}}$.
7. The interrupt vector is the address of the interrupt service procedure.
9. 256
11. INT 0 occurs for a divide error.
13. A real mode interrupt vector is 4 bytes in length and contains the segment and offset address of the interrupt service procedure, while a protected mode interrupt vector is 8 bytes in length and contains the selector and 32-bit offset address of the interrupt service procedure.
15. The BOUND instruction tests the contents of a 16-bit register with two numbers stored in the memory. If the register contains a number that is outside of the boundaries set by the memory data, INT 5 occurs.
17. INT 44H is stored at vector locations 110H–113H.

19. INT 7 is used to emulate a coprocessor.
21. The I flag controls whether the INTR pin si enabled or disabled.
23. CLI and STI clear and set the interrupt flag.
25. INT 2
27. Level
29. Vector
31.



33. The pull-ups force the inputs of the data bus to FFH when the interrupt acknowledge cycle executes.
35. Since the signals are ORed together to generate the interrupt, the only way to determine which device caused the interrupt is to ask (poll) the devices.
37. 9
39. The CAS pins are used to cascade the 8259.
41. The ICW is the initialization control word.
43. 3
45. LTIM in ICW1
47. The nonspecific end of interrupt is used to clear the most recent interrupt request.
49. The interrupt request register can be used to determine the levels found on the interrupt inputs.
51. INT 70H through INT 77H

Chapter Thirteen

1. HOLD and HLDA
3. Memory to I/O
5. A0–A7 and D0–D7 (where address bits A8–15 appear).
7. A memory-to-memory DMA transfer occurs when one channel addresses the source address and another channel address a destination address. Data are then transferred from source to destination.
9. The DMA controller is in its hold state and the microprocessor operates normally.

11. 2002H and 2003H
13. 64K
17. Micro
19. Sectors
21. NRZ recording is used because it erases old data when it records new data.
25. The disk heads must be parked over a landing zone when power is removed so the heads do not damage the surface of the disk.
27. A write once optical disk such as a CD-R or DVD-R.
29. 4.7G bytes.
31. Red, green, and blue
33. The smallest video picture element.
35. By using 2 levels of brightness for each of the three primary colors.
37. Because the analog signal are continuously variable an infinite number fo colors are possible.
39. 540

Chapter Fourteen

1. Integer, BCD, and floating-point.
3. A BCD number is stored in 10 bytes of memory with 9 bytes containing the BCD integer magnitude as packed BCD and the 10th byte containing only the sign of the number.
5. (a) 0 10000011 11001100000000000000000
(b) 0 10001000 00111000000000000000000
(c) 1 11111110 01000000000000000000000
(d) 0 00000000 00000000000000000000000
(e) 1 10001000 11110100010000000000000
7. The coprocessor may be idle or it may execute a coprocessor instruction at the same time.
9. These bits indicate the relative size of a number after a test or compare instruction as well as if the number is valid or invalid.
11. An error bit.
13. By programming the rounding control bits in the coprocessor control register.
15. FFF8H–FFFFH
17. A NAN (not a number) is a number with an exponent of all ones and a significand not equal to zero.
19. Truncate
21. ESC
23. (a) FROG DQ 23.44 (b) DATA3 DD –123 (c) DATAL DD –23.8
(d) DATA2 DQ ?
25. An integer is loaded from memory location DATA to the top of the stack.
27. FADD (no operands) pops the top two stack elements and adds them then returns the sum (pushes) to the top of the stack.

29. It stores the BCD version of the top of the stack into memory location DATA hen it pops the stack.
31. The FCOMI instruction replaces the FCOM, FSTSW AX, and SAHF instructions
33. Usually an FCOMI instruction must appear before an FCMOV.
35. FTST compares ST against zero, while FXAM changes the status flags to indicate the type of number at ST (positive, negative, a NAN, etc.).
37. IE
39. FLD1
41. FSTENV

```
43. AREA   PROC  NEAR
           FLD   L
           FMUL  W
           FSTP  A
           RET
    AREA   ENDP
```

```
45. ROOT   PROC  NEAR
           MOV  ECX,9
           MOV  EBX,OFFSET ROOTS
           .REPEAT
                 MOV  EAX,11
                 SUB  EAX,ECX
                 MOV  TEMP,EAX       ;TEMP is defined as DD
                 FILD TEMP
                 FSQRT
                 FSTP DWORD PTR [EBX]
                 ADD  EBX,4
           .UNTILCXZ
           RET
    ROOT   ENDP
```

47. One does a wait the other does not.

```
49. COS    PROC  NEAR
           MOV  TEMP,EAX
           FLD  TEMP
           FLDPI
           FADD ST,ST(0)
           FDIV
           FCOS
           FSPT TEMP
           MOV  EAX,TEMP
           RET
    COS     ENDP
```

```
51. MULT   PROC  NEAR
           MOV  TEMP,EBX
           FLDPI
           FLD  TEMP
           FMUL
           FSTP TEMP
           MOV  EBX,TEMP
           RET
    MULT   ENDP
```

```
53. LOG10  PROC  NEAR
           FLD1
           FXCH  ST(1)
           FYL2X
```

```
         FILD   TEN
         FLD1
         FXCH   ST(1)
         FYL2X
         FLD1
         FDIVR
         FMUL
         RET
TEN      DW     10
```

55. The multimedia extension allows integer arithmetic and logic on multiple data with a single instruction.

57. The MM registers use the coprocessor stack registers.

59. Unsigned saturation is where the carry is dropped after the addition or borrow after a subtraction.

61.

```
         MOV  ECX,64
         .REPEAT
               PMOV   MM0,QWORD PTR ARRAY1[ECX*8-8]
               PMULLW MM0,QWORD PTR ARRAY2[ECX*8-8]
               PMOV   QWORD PTR ARRAY3[ECX*8-8]
               PMOV   MM0,QWORD PTR ARRAY1[ECX*8-8]
               PMULHW MM0,QWORD PTR ARRAY2[ECX*8-8]
               PMOV   QWORD PTR ARRAY3[ECX*8-8+256]
         .UNTILCXZ
```

63. Streaming SIMD extensions.

65. 4

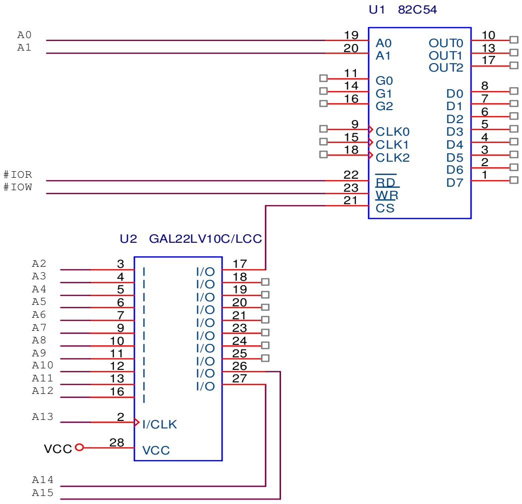
67. An octal word is a 128-bit wide number.

Chapter Fifteen

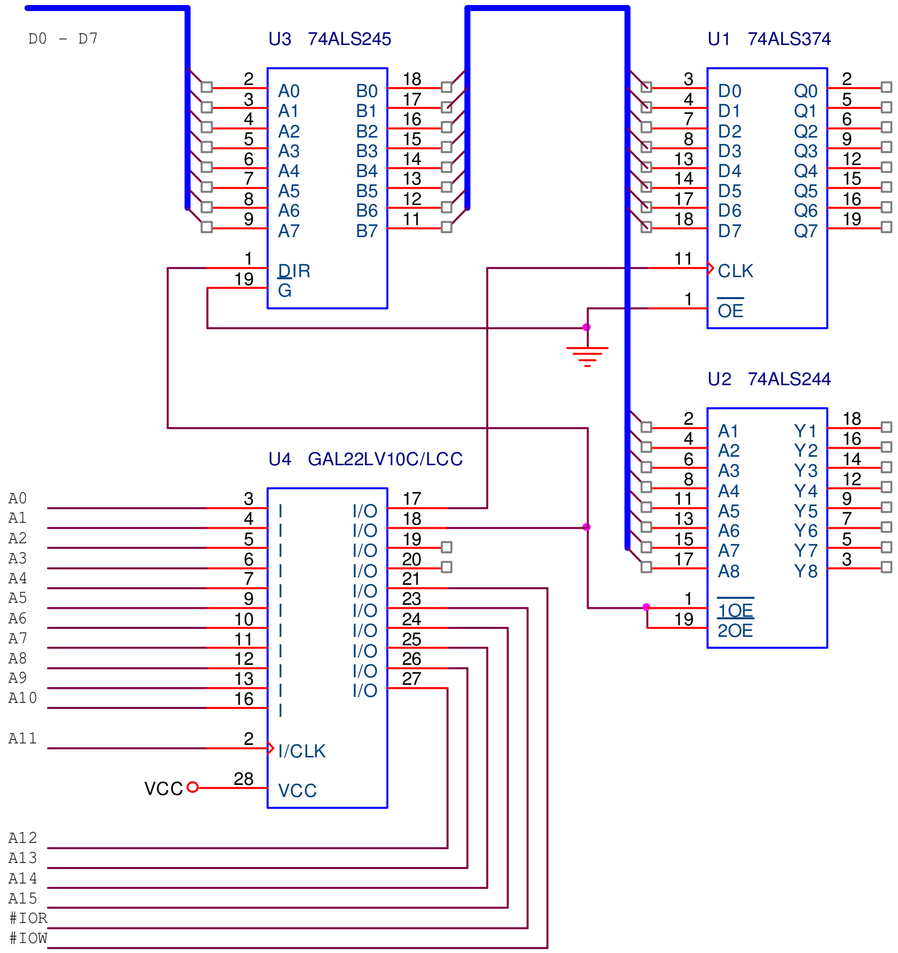
1. Industry Standard Architecture

3. It was long ago, but today because of its relatively low speed, it is only suited to I/O expansion.

5.



7.



13. On the first positive edge of the clock after $\overline{\text{FRAME}}$ goes low.

15. Plug and Play is where the computer polls the PCI cards in a system to determine what interrupts are required and also the type of the card.

17. If operating in DOS, the BIOS is tested for PCI if an 0B101H is placed in AX followed by an INT 1AH. If carry is set upon return PCI is not present.

19. Speed and data width.

21. 378H–37FH

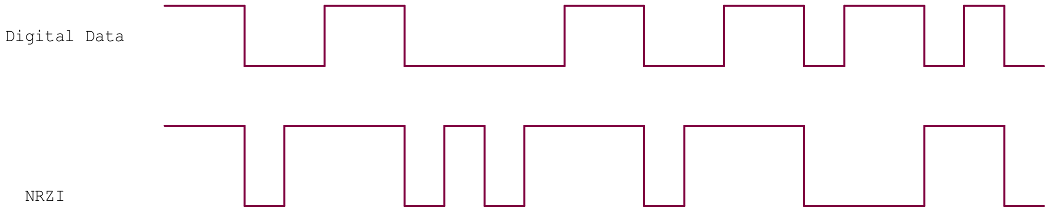
23. 25 pins

29. NRZ

31. For many applications it has replaced the ISA and PCI bus.

33. Non-return to zero inverted

35.



37. ACK acknowledges the receipt of data and NAK does not acknowledge the receipt of data.
39. 2 GBps

Chapter Sixteen

1. The main differences are the internal timers, the chip selection unit, the additional interrupt inputs, and in some models the serial communications ports and the enhanced 4-channel DMA controller.
3. Leadless chip carrier (LCC) and pin grid array (PGA)
5. The main difference is that the EB version contains 10 chip selection pins and a pair of serial communications ports.
7. 4
9. Memory access time is the amount of time that the microprocessor allows the memory to look up data. If not enough time exists, wait states are inserted to allow additional time for access.
11. I/O ports FF00H–FFFFH
13. INT 12/INT 0CH
15. Master and slave modes are available.
17. 1
19. The EOI register is used to clear the interrupt from the microprocessor. If not, the interrupt will never occur again.
21. Times 0 and 1
23. If both compare registers are used one determines the length for the logic 0 output and the other determines the length of the logic 1 output.
25. The P bit selects the system clock as the input to the timer.
27. The timer output pins are used to provide wither a single pulse or an output with a selectable logic 1 and logic 0 time.
29.

```
MOV  AX,0
MOV  DX,0FF50H
OUT  DX,AX
MOV  AX,105
MOV  DX,0FF52H
OUT  DX,AX
MOV  AX,0C008H
MOV  DX,0FF56H
OUT  DX,AX
```

31. 20
33. 6
35. FFFFH
37. 00000H
39.

```
MOV  AX,1F44H
MOV  DX,0FFA8H
OUT  DX,AX
```

41.

```
MOV  AX,2002H
MOV  DX,0FF8CH
OUT  DX,AX
MOV  AX,300AH
```

```
          MOV   DX,0FF8EH
          OUT   DX,AX
```

43. 16M
45. 8086
47. Loads the segment limit
49. Multiple threads are handled by a scheduler that starts a new thread on each tick of the scheduler.

Chapter Seventeen

1. 4G
3. The DX has a full 32 bit address bus, while the SX is a scaled down version with a 24-bit address bus.
5. 4 or 5 mA depending on the pin compared to the 8086 which has 2 mA on each output pin.
7. A hardware reset causes the address bus to start at memory location FFFFFFF0H.
9. A cache memory is a high-speed store of data and/or instructions. Because the main memory is relatively slow, when data or instructions are accessed a second time, they are accessed from the cache at a high speed increasing system performance.
11. 800000F8H–800000FFH
13. 40MHz
15. CR0 mainly selects protected mode and paging, CR1 is reserved by Intel, CR2 contains the linear fault address for debugging, and CR3 contains the base address of the page directory.
17. `INT 1` or type 1
19. `MOV  EAX,CR0`
21. `MOV  FS:[DI],EAX`
23. Yes
25. Coprocessor not available interrupt.
27. The double fault interrupt occurs when two interrupts occur simultaneously.
29. A descriptor describes a memory segment, or a gate.
31. The TI bit in the selector is set to select the local descriptor table.
33. 8K
35. A segment descriptor defines a memory segment and a system descriptor defines a memory location for a call or interrupt or a task state segment.
37. The TSS is address by the task register.
39. The switch occurs when a 0 is placed into the PE bit of CR0.
41. Where ever he programmer decides to place it as dictated by CR3.
43. The entry in the page table and entry that corresponds to address D0000000H contains a C0000000H.
45. The $\overline{\text{FLUSH}}$ input causes the internal cache to be erased.
47. The flags are almost identical except for the AC flag.
49. Even
51. 16

53. A cache write-through is when data are written into the cache and the DRAM at the same time.

Chapter Eighteen

1. 4G bytes
3. 64 bits
5. 66 MHz
7. Address parity
9. $\overline{\text{BRDY}}$
11. If the instructions are not dependent then two can be executed simultaneously, one by each integer unit.
13. 5.8 ns
15. The $\overline{\text{SMI}}$ input causes an interrupt to the system memory management interrupt at address 38000H unless changed to some other location in the first 1M byte of memory.
17. The SMM is exited by using the RSM instruction.
19. Modify the dump base address register at locations 3FEF8H–3FEFBH.
21. 1
23. CR4
25. The TSC counts system clock pulses in a 64-bit counter located within the microprocessor. It can be used to time events by storing its value when the event begins and at the end of the event read TCS and subtract the stored number to obtain the count in clock pulses.
27. The bank enable signals are multiplexed with address (A15–A8) information and must be extracted from the address bus during the second clock cycle of a bus cycle.
29. PAE and PSE have been added to control the additional address bits (A32–A35).
31. Error correction code.

Chapter Nineteen

1. 32K
3. The Level 2 cache operated at the bus speed (66 MHz) in the Pentium and at ½ the microprocessor speed in the Pentium II.
5. 2
7. No, the Pentium II is on a cartridge.
9. Used for serial messages between the Pentium II and APIC.
11. 66 MHz or 100 MHz
13. 72 bits
15. Version number and features have been added to CPUID.

17.

```
MOV  ECX, 175H
MOV  EDX,0
MOV  EAX,12H
WRMSR
```

19. SYSEXIT
21. Ring 0
23. FSAVE saves the state of the coprocessor and FXSAVE saves the state of the MMX unit.
25. SIMD extension SSE2.