

CHAPTER 1

1. Components of a computer: ALU and Control Unit (CPU), Memory, Input, and Output.
2. Functions of various components:
 - .CPU: It processes and stores binary data, transfers data from and to memory and I/O devices, and provides timing to all the operations. It includes ALU, register arrays, and control unit. The ALU performs the arithmetic and logic operations, and the control unit provides timing.
 - .Input - provides binary data as an input to the CPU.
 - .Output - accepts binary data from the CPU.
3. A microprocessor functions as the CPU of a microcomputer, and includes the ALU, register arrays, and the control unit on one chip; it is manufactured using the LSI technology. On the other hand, the CPU is designed with various discrete boards. Functionally, both are similar; however, technology and processes used for designing is different.
4. A microprocessor is one component of a microcomputer, and the microcomputer is a complete computer consists of a microprocessor, memory, input, and output.
- 5, 6. See Summary: Scale of Integration
7. Four bytes.
8. The machine language of the 8085 are the commands to the microprocessor given in binary. These are the binary instructions the processor can understand and execute. The assembly language comprise of mnemonics (group of letters to represent commands) assigned by the manufacturer for the convenience of the users.
- 9, 10, 11. See Summary: Computer Languages
12. The assembly language mnemonics represent instructions to the microprocessor; therefore, when they are translated into machine language, there is one-to-one correspondence between the mnemonics and the machine code. The assembly language programs are compact, require less memory space, and are efficient. The high level languages are written in English- like statements, and when these statements are translated in machine language, the object code tends to be large, and requires large memory. The execution of the programs written in high level languages is less efficient than that of assembly language programs.

- 13, 14. See Summary: Computer Languages
15. ASCII codes in Hex: A = 41, Z = 5A, and m = 6D
16. See Summary: Computer Languages

CHAPTER 2

1. Memory Read, Memory Write, I/O Read, and I/O Write.
2. A bus is group of lines (wires or conductors) which carry digital information.
3. The function of the address bus is to carry a binary address of a memory location or an I/O device. The address bus is unidirectional, and the information flows from the MPU to peripherals and memory.
4. A microprocessor with 14 address lines is capable of addressing 16 K (2) memory locations.
5. 21 address lines.
6. Data bytes are transferred in both directions between the MPU and memory/peripherals.
7. IOR (I/O Read), IOW (I/O Write), MEMR (Memory Read), and MEMW (Memory Write).
8. In memory write operation, the control signal required is MEMW, and the direction of the data flow is from the MPU to memory.
9. The accumulator is an 8-bit register and it is a part of the ALU. All 8-bit arithmetic and logic operations are performed in relation to the accumulator content, and the result is stored in the accumulator (with a few exceptions).
10. A flag is the output of a given flip-flop to indicate certain data conditions.
11. The program counter and the stack pointer store memory addresses of 16 bits.
12. The program counter always points to the next memory location; therefore, the content of the program counter will be 2058H.
13. 128 registers and $128 \times 4 = 512$ memory cells.
14. 1024 bits are can be stored by this chip; however, it can not be specified as a 128-byte memory chip because the byte indicates 8-bit memory registers; this chip has 4-bit registers.

15. 8-bit word size.
16. 8 chips.
17. 4 chips.
18. 32 chips.
19. The WR signal enables the input buffer of a memory chip so that information can be stored (written) in the selected memory register.
20. 11 address lines.
21. 16 pages and the last location is 2FFFH.
22. The starting address is F800H, and the memory map is F800H to FBFFH.
23. The starting address is: E000H.
24. The address ranges from FF00H to FFFFH.
25. The address of the selected register: 1000 0000 0100 0111 = 8047H
26. The memory map ranges from 2000H to 23FFH.
27. The address of the selected register: 0010 0000 1111 1000 = 20F8H
28. 8 address lines are required for a peripheral I/O port, and 16 address lines are required for a memory-mapped I/O port.
29. Tri-state devices are logic devices with three states; the third state is high impedance. In a bus-oriented system, devices are connected in parallel, and the buses are capable of driving one TTL logic device. The MPU communicates with one peripheral at a time, and other peripherals are placed in high impedance to avoid bus loading.
30. High impedance state.
31. From B to A.
32. None. The decoder is not enabled; all output lines will be high.
33. The line 6 (O_6)
34. 0 0 1 (Complement of 1 1 0)

35. A transparent latch is a flip-flop; its output changes according to input when the clock signal is high, and it latches the input when the clock goes low. The latch is necessary for output devices to retain the result; otherwise, the result will disappear.
36. The high-order address lines: A12-A15, the low-order address lines: A0-A10, and the don't care line: A11.
37. This answer assumes the memory chips are 2048 X 4:
- | | | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|----|----|----|----|----|----|----|----|----|-----------|-----------|
| A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 = F000H |
| | | | | | | | | | | | | | | | | |
| 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 = F7FFH | |
38. The memory occupies the memory space from F000H to FFFFH. The don't care line A11 generates additional address range. This is a 2K memory chip that occupies 4K of memory space in the map; thus wasting 2K of memory space. If A11 is assumed to be at logic 0 as in Q. 37, the address range is: F000H to F7FFH and if it is assumed to be at logic 1, the address range (also called foldback memory space) is: F800H to FFFFH.

CHAPTER 3

- The ALE signal goes high at the beginning of each machine cycle indicating the availability of an address on the address bus, and the signal is used to latch the low-order address bus. The IO/M signal is a status signal indicating whether the machine cycle is I/O or memory operation. The IO/M signal is combined with the RD and WR control signals to generate IOR, IOW, MEMR and MEMW control signals.
- The low-order bus AD7-AD0 is used for two purposes. In the earlier part of a machine cycle, the bus is used for the low-order address of a memory location the 8085 is accessing, and in the latter part of the cycle the bus is used for data. By demultiplexing the bus, the address and the data are kept separate.
- In Fig. 3.22, the input signal RD and WR cannot be low at the same time. Therefore, the valid combinations of the input signals are:

IO/M	RD	WR	Output Signal		
0	0	0	O ₀	Invalid	RD and WR cannot be active simultaneously
0	0	1	O ₁	MEMR	M and RD active
0	1	0	O ₂	MEMW	M and WR active
0	1	1	O ₃	Irrelevant	Both RD and WR are inactive
1	0	0	O ₄	Invalid	RD and WR cannot be active simultaneously
1	0	1	O ₅	IOR	IO and RD active
1	1	0	O ₆	IOW	IO and WR active
1	1	1	O ₇	Irrelevant	Both RD and WR are inactive

4. See the answer of Q.3.
5. In Fig. 3.23, the 74LS139 is enabled when IO/M is low. Therefore, the following memory control signals can be generated.

RD WR Decoder Output

0	0	O ₀ - Invalid
0	1	O ₁ - MEMR
1	0	O ₂ - MEMW
1	1	O ₃ - No operation

6. The output of the latch will be 05H; however, it will not be latched until the ALE goes low.
7. The output of the latch is 05H. At T₂, the ALE is low; therefore, the latch will not be enabled, and it will continue to hold the previously latched byte (05H).
8. The crystal frequency should be = 2.2 MHz because the oscillator logic divides the input frequency by two.
9. See the steps on page 66/67, Example 3.1.
10. The sum of 87H + 79H = 100H. Therefore, the accumulator will have 00H, and the flags will be S = 0, CY = 1, Z = 1.
11. 2060H. The program counter always points to the next machine code to be fetched.
12. 18T X .2 micro-sec = 3.6 micro-sec.
13. (A15-A8) = 20H, (AD7-AD0) = 47H, (PC) = 2076H
14. RD and IO/M are asserted low.

15. The second machine cycle is Memory Read; the processor reads the contents of memory in register B, and the control signal is RD.
16. The fourth machine cycle is Memory Read; the processor reads the contents of memory in the accumulator.
17. (A15-A0) = 2050H
(AD7-AD0) as data bus = Contents of location 2050H
18. (Refer to Instruction Set on pages 696-699)
SUB B = OF (Opcode Fetch)
ADI 47H = OF, MR (Memory Read)
STA 2050H = OF, MR, MR, MW (Memory Write)
PUSH B = OF, MW, MW
19. Memory map: 6000H to 6FFFH
20. Memory map: 8000H to 8FFFH
21. OR gate
22. Connect RD to OE of the memory chip and IO/M to E2 of the decoder.
23. A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0

0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	= 2800H
									1	1	1	1	1	1	1	1	= 2FFFH
24. Total range = 16K. Map = 8000H to BFFFH
25. A data byte entered at location 2100H will be accepted and stored at location 2000H. The address lines A10, A9, and A8 are not being used for memory addressing; therefore, they can assume 0 or 1 (don't care) logic state which results into multiple addresses for the same memory locations.
26. Memory address: 0800H-08FFFH, and the foldback memory ranges from 0900 to 0FFFFH.
27. Memory map: 3800H - 3FFFH.
28. In Figure 3.19, three lines are don't care which can have (2³) eight combinations. Thus, the memory chip will occupy the memory space equal to eight times its size.
29. ROM1: 0000H - 1FFFH, ROM2: E000H - FFFFH, R/W1: 8000H - 83FFFH

30. Memory map: 8000H to 83FFH (Assume all don't care lines at 0)
Foldback Memory: 8400H to 9FFFH
31. The address range: 0000H to 3FFFH
32. The address range: 4000H to 7FFFH
33. The primary address range: 0000H to 1FFFH (Assumes A13 = 0)
The foldback or the mirror address range: 2000H to 3FFFH
34. The mirror address range: 8000H to 9FFFH
35. The address range when Y1 is asserted: 4000 to 7FFFH
36. The total address range is : 4000H to BFFFFH. For a 16K memory chip, when A14 = 1, the address ranges from 4000H to 7FFFH as in Q. 35. When A14 = 0, the address ranges from 8000H to BFFFFH. For a 32K memory chip, it is accessed either by Y1 or Y2; therefore, the address ranges from 4000H to BFFFFH.
37. The opcode fetch cycle begins immediately after MEMW signal.
1st MEMR ----> opcode fetch of the JMP instruction.
4th MEMR ----> opcode fetch of the MVI instruction.
6th MEMR ----> opcode fetch of the STA instruction.
38. The last MEMR is the third byte of the STA instruction. It reads FFH.

CHAPTER 4

1. The number of output ports in the peripheral I/O is restricted to 256 ports because the operand of the OUT instruction is 8 bit; it can have only 256 combinations.
2. Yes.
3. The 8085 differentiate between the input and the output ports of the same address by the control signal. The input port requires the RD and the output ports requires the WR control signals.
4. WR (low) and IO/M (high).
5. Pulse going from high to low.
6. Trailing edge.

7. Each LED requires 10 to 19 mA current for proper illumination. The latch cannot supply the necessary current when the output is logic high, but it can sink the necessary current when the logic level is low.
8. RD (low) and IO/M (high).
9. A latch is necessary to hold the output data for display; however, the input data byte is obtained by enabling a tri-state buffer and placed in the accumulator.
10. RD, WR, and IO/M (low).
11. No.
12. 78H.
13. No. An output byte will be displayed temporarily until the WR signal is active, and then, it will disappear.
14. Memory-mapped I/O. LE is enabled when IO/M is low.
15. 8000H.
16. Assuming A3 = 0, port address = F1H.
17. If A7 = 0, port address = 75H, and if A7 = 1, address = F5H.
18. If IO/M is connected to /E1 (active low), it will be a memory-mapped I/O. The port address = 00F5H.

Replace OUT F5H by STA 00F5H
19. MVI A, C0H ;Code for '0'
OUT F5H
HLT
20. In Q. 19, replace the code C0H by the code for letter 'H'.
Code for H = 89H.
21. If A7 is replaced by IO/M signal, the circuit will have three don't care address lines: A7, A4 and A3 resulting in eight different addresses.

If A7 = 0, the addresses are: 04H, 0CH, 14H and 1CH.
If A7 = 1, the addresses are: 84H, 8CH, 94H, and 9CH (as shown in Section 4.34).
22. The port will be a memory-mapped I/O with an address = 00F8H.

23. Port A = Memory-mapped Output Port
Port B = Memory-mapped Input Port
24. Both are memory-mapped I/O ports. Assuming the address lines A15-A8 are at logic 0, Port A and Port B will be 0085H.
25. In Figure 4.10, the output O5 is enabled by the address which is active for three T-states. On the other hand, the IOW signal requires WR signal which is active for approximately one and half T-states.
26. a. Machine Cycles: M1 M2 M3
- | | | | | |
|-----------|----|----|----|----|
| IN 84H | OF | MR | MR | MR |
| JMP START | OF | MR | MR | MR |
- b. $20T \times 0.5 = 10$ micro-sec.
- c. Six times.
- d. 10 micro-sec (from beginning to th next beginning)
- e. There is no WR pulse in the routine. IO/M high or IORD can be used to sync the scope.
27. a. Machine Cycles: M1 M2 M3 M4
- | | | | | |
|-----------|----|----|----|----|
| LDA FFF9H | OF | MR | MR | MR |
| STA FFF8H | OF | MR | MR | MW |
| MOV B,A | OF | | | |
| JMP START | OF | MR | MR | |
- b. FFF9H
- c. RD = 11 times and WR = 1 time.
- d. $40T \times 0.5$ micro-sec = 20 micro-sec.
28. In Figure 4.18, the address line A4 is don't care.
Assuming A4 = 0: Input Port = 2FH and Output Port = 8FH.
Assuming A4 = 1: Input Port = 3FH and Output Port = 9FH.
29. START: IN 2FH ;Read input port
ANI 0000011B ;Mask all bits except D1 and D0
JNZ START ;If a switch is open, read again
MVI A, 00 ;This instruction is unnecessary
;Used here for clarity

```

        OUT 8FH      ;Turn on all LEDs
        HLT

30.  START:  IN  2FH      ;Read input port
        ORI 11111100B ;Set all bits from D2 to D7
        MOV B, A     ;Save reading in B
        CPI FFH     ;Check whether both switches are open
        JZ  START   ;If both switches are open, read again
        MOV A, B     ;Get initial reading
        OUT 8FH     ;Turn on corresponding LED
        HLT

31.  START:  IN  2FH      ;Read input port
        ANI 00000011B ;Mask all bits except D0 and D1
        JZ  START   ;If both switches are open, read again
        CMA        ;Make all readings 1 except which is open
        OUT 8FH     ;Turn on corresponding LED
        HLT

32..  The address of the latch enabled by Y3 = F5H and the address of the latch enabled by
        Y2 = F4H.

        MVI A, 98H ;Common anode code for '9'
        OUT F5H
        MVI A, F8H ;Common anode code for '7'
        OUT F4H
        HLT

```

PART II: SOLUTIONS

CHAPTER 5

1. The four categories of instructions that manipulate data are: data transfer (copy), arithmetic, logic, and branch.
2. The task to be performed is called the opcode (operation code), and the data to be operated on is called the operand which may be specified as data, register or address.

Opcode: MOV and Operand: H,L

3. The machine code: 01 100 111 = 67H
4. (a) 2647H OPCODE = MVI OPERANDS = H, 47H
(b) C6F5H OPCODE = ADI OPERANDS = A (IMPLIED), F5H
(c) 91H OPCODE = SUB OPERANDS = A (IMPLIED), C
5. (a) HEX = 325020H OPCODE = STA OPERANDS = 2050H
(b) HEX = C27020H OPCODE = JNZ OPERANDS = 2070H
6. The SUB A instruction clears the accumulator. Z = 1, CY = 0

7. INSTRUCTION ADDRESS HEX

MVI B,4FH	2000	064F
MVI C,78H	2002	0E78
MOV A,C	2004	79
ADD B	2005	80
OUT 07H	2006	D307
HLT	2008	76

8. INSTRUCTION ADDRESS HEX

MVI A,8FH	2020	3E8F
MVI B,68H	2022	0668
SUB B	2024	90
ANI 0FH	2025	E60F
STA 2070H	2027	327020
HLT	202A	76

9. INSTRUCTION ADDRESS HEX

START:	IN F2H	2000	DBF2
	CMA	2002	2F
	ORA A	2003	B7
	JZ START	2004	CA0020

10. Logical steps to add two Hex numbers:

Load A2H in one register.
 Load 18H in second register.
 Copy A2H in the accumulator.
 Add the contents of the second register to the contents of the accumulator.
 End of the program.

11. MVI B, A2H
 MVI C, 18H
 MOV A, B
 ADD C
 HLT

12. Register contents:

Initial: B=28H, A=97H

After the execution : A=28H, B=28H, C=28H

13. In Q. 6, if the code 07H (port address) is omitted, the processor assumes the opcode of the next instruction 76H (HLT) as the address of the output port, outputs the contents of the accumulator to the address 76H, and continues to the next code. After the next code, results are indeterminate.
14. In Q. 8, if the byte 0FH is omitted, the processor assumes the opcode 32H of the next instruction (STA) as the second byte of the ANI instruction. The processor is a sequential machine; it assumes the next code 20H (the low-order address of 2070H) as the opcode of the next instruction and continues.

CHAPTER 6

Section 6.1: Data Transfer (Copy) Operations

- | | | | | | | | | |
|----|----------|----|----|----|---|----|----|----|
| 1. | | A | B | C | D | S | Z | CY |
| | MVI A,00 | 00 | | | | NA | NA | NA |
| | MVI B,F8 | 00 | F8 | | | | | |
| | MOV C,A | 00 | F8 | 00 | | | | |

- ```

MOV D,B 00 F8 00 F8
HLT

```
3.

```

MVI C, 65H
MVI A, 92H
OUT PORT1 ;Display 92H
MOV A, C
OUT PORT0 ;Display 65H
HLT

```
  4.

```

IN 07H
OUT 00H ;Display data from input port 07H
IN 08H
MOV B, A ;Store data from port 08H
HLT

```
  5. 82H
  6. 80H
  7. Both will be 80H

### Section 6.2: Arithmetic Operations

8.

|           | A  | B  | S  | Z  | CY |
|-----------|----|----|----|----|----|
|           | 00 | FF | 0  | 1  | 0  |
| MVI A,F2H | F2 | FF | NA | NA | NA |
| MVI B,7AH | F2 | 7A | NA | NA | NA |
| ADD B     | 6C | 7A | 0  | 0  | 1  |
| OUT PORT0 | 6C | 7A | NA | NA | NA |
| HLT       |    |    |    |    |    |
9. The instruction ADD A will add the content of the accumulator to itself; this is equivalent to multiplying by 2.
10. The instruction SUB A will clear the accumulator. The flag status will be: CY = 0, Z = 1.
11.

|           | A  | C  | S  | Z  | CY |
|-----------|----|----|----|----|----|
|           | XX | XX | 0  | 0  | 0  |
| MVI A,5EH | 5E | XX | NA | NA | NA |
| ADI A2H   | 00 | XX | 0  | 1  | 1  |

```
MOV C,A 00 00 NA NA NA
HLT
```

13. MVI A, 3AH  
ADI 48H  
OUT PORT#  
HLT

14. MVI A, 00H (A) = 00000000  
DCR A - 00000001  
OUT PORT# -----  
HLT X 11111111 = FFH

The instruction DCR does not set the CY flag.

15. A = 95H S=1 CY=0  
The S flag has no significance when subtracting unsigned numbers. If the the CY flag is set, it indicates a negative result.

| 16.     | A  | B  | S | Z  | CY    |
|---------|----|----|---|----|-------|
| SUB A   | 00 |    |   | 0  | 1 0   |
| MOV B,A | 00 | 00 |   | NA | NA NA |
| DCR B   | 00 | FF |   | 1  | 0 NA  |
| INR B   | 00 | 00 |   | 0  | 1 NA  |
| SUI 01H | FF | 00 |   | 1  | 0 1   |
| HLT     |    |    |   |    |       |

18. SUB A ;Clear acumulator  
ADI 47H  
SUI 92H  
OUT PORT0 ;Display result:(47H-92H)  
ADI 64H  
OUT PORT1

47H = 0100 0111

2'Com. = 0110 1110

of 92H -----

1 1011 0101 = B5H Borrow flag (CY) is set  
64H = 0110 0100 to indicate negative results.

1 0001 1001 = 19H Borrow flag is deleted by the CY of the result.

(PORT0) = B5H and (PORT1) = 19H

19. If a number is added before clearing the accumulator, the result will include the residual contents of the accumulator.

Section 6.3: Logic Operations

20. The instruction XRA A clears the accumulator, and the flag status is: CY = 0, Z = 1.
21. The instruction ADD B sets the CY flag, but the instruction ORA A resets the CY flag.
- A = 00 S = 0 Z = 1 CY = 0
23. The instruction ORA A will set the flag without affecting the content of the accumulator.

| 24. |           | A  | B  | S  | Z  | CY |
|-----|-----------|----|----|----|----|----|
|     | XRA A     | 00 |    | 0  | 1  | 0  |
|     | MVI B,4AH | 00 | 4A | NA | NA | NA |
|     | SUI 4FH   | B1 | 4A | 1  | 0  | 1  |
|     | ANA B     | 00 | 4A | 0  | 1  | 1  |
|     | HLT       |    |    |    |    |    |

26. MVI C, A8H  
 MOV A, C  
 ANI 0FH ;Masking byte to mask D7-D4  
 OUT PORT0  
 HLT
27. MVI D, 8EH  
 ANI 0FH ;Mask D7-D4  
 MOV D, A ;Save in D  
 MVI E, F7H  
 ANI 0FH ;Mask D7-D4 of second byte  
 XRA D ;Exclusive OR masked bytes  
 OUT PORT0  
 HLT
28. MVI B, 91H  
 MVI C, 87H  
 MOV A, B  
 ANI 01H ;Mask all bits of 91H except D0  
 MOV B, A ;Save D0 from first byte  
 MOV A, C  
 ANI 01H ;Mask all bits of 87H except D0  
 ANA B ;AND bits D0 of 91H and 87H

```
OUT PORT1 ;Turn on/off light connected to D0
HLT
```

29. IN 07H  
CMA ;Complement data from port 07H  
ORA A ;Set Z flag if all switches are open  
| ;Continue  
|  
|

#### Section 6.4: Branch Operations

30. 00  
31. 28H  
32. In the following program, explain the range of bytes that will be displayed at PORT2 for various values of BYTE1.

```
 MVI A, BYTE1
 MOV B, A
 SUI 50H
 JC DELETE
 MOV A, B
 SUI 80H
 JC DSPLAY
DELETE: XRA A
 OUT PORT1
 HLT
DSPLAY: MOV A, B
 OUT PORT2
 HLT
```

In this problem, all bytes from 50H to 7FH will be displayed at PORT2.

33. The address of the output port = F2H. All positive signed numbers and zero will be displayed at port F2H.  
34. 00  
35. This routine displays the absolute value (magnitude) of BYTE1.  
36. 59H



```

37. START: MVI D,9BH
 MVI E,A7H
 MOV A,D
 ADD E
 JC DSPLAY
 OUT 00H
 HLT
 DSPLAY: MVI A,01H
 OUT 00H
 HLT

```

```

38. XRA A ;Clear CY
 MVI B, FFH
 INR B
 MOV A, B
 JNC DSPLAY
 MVI A, 01H
 DSPLAY: OUT PORT# ;The output = 00H because INR does not
 HLT ;set CY flag.

```

To clear the CY flag, the instructions such as ANA A, SUB A, ORA A can be used instead of the instruction XRA A.

```

39. ORA A ;Clear CY
 MVI C, FFH
 MOV A, C
 ADI 01H
 JNC DSPLAY
 MVI A, 01H
 DSPLAY: OUT PORT# ;The output = 01H because ADI sets CY
 HLT ;flag.

```

```

40. MVI B, BYTE1
 MVI C, BYTE2
 MOV A, B
 SUB C
 JNC DSPLAY ;Jump if result is positive
 CMA ;Take one's complement
 ADI 01H ;Find two's complement
 DSPLAY: OUT PORT1
 HLT

```

## Section 6.6: Debugging a Program

### 41. Reference: Section 6.53

```
2000 ORG 2000H
2000 DBF1 START: IN 0F1H
2002 47 MOV B, A
2003 DBF2 IN 0F2H
2005 E680 ANI 80H
2007 4F MOV C, A
2008 78 MOV A, B
2009 E680 ANI 80H
200B A1 ANA C
200C C21720 JNZ SHTDWN
200F 78 MOV A, B
2010 E61F ANI 1FH
2012 D3F3 OUT 0F3H
2014 C30020 JMP START
2017 3E40 SHTDWN: MVI A, 40H
2019 D3F3 OUT 0F3H
201B 76 HLT
201C END
```

### 42. In the following program, the instructions IN F1 and IN F2 are replaced by loading two data bytes 97H and 85H in registers D and E respectively.

```
2000 ORG 2000H
2000 1697 START: MVI D,BYTE1 ;Simulate data from port F1
2002 1E85 MVI E,BYTE2 ;Simulate data from port F2
2004 7B MOV A,E
2005 E680 ANI 80H ;Mask S6'-S0'
2007 5F MOV E,A ;Save S7'
2008 7A MOV A,D
2009 E680 ANI 80H ;Mask S6-S0
200B A3 ANA E ;Check S7 & S7'
200C C21720 JNZ SHTDWN ;If S7 & S7' are on, jump to
; initiate shut down procedure
200F 7A MOV A,D ;If not, get data from port F1
2010 E61F ANI 1FH ;Mask bits D7-D5
2012 D3F3 OUT PORT ;Turn conveyer belts
2014 C30020 JMP START
2017 3E40 SHTDWN: MVI A,40H ;Set bit D6=1
2019 D3F3 OUT PORT ;Turn on emergency
201B 76 HLT
0097 = BYTE1 EQU 97H
```

```

0085 = BYTE2 EQU 85H
00F3 = PORT EQU 0F3H
201C END

```

43. This program turns on the LED indicator when the switch S7 is on.

```

2000 ORG 2000H
2000 DBF1 START: IN 0F1H ;Comments are same as illustrative
2002 47 MOV B,A ;program -- omitted here
2003 DBF2 IN 0F2H
2005 E680 ANI 80H
2007 4F MOV C,A ;Save S7'
2008 78 MOV A,B ;Get data from port F1
2009 E680 ANI 80H
200B CA1420 JZ TURNON ;If S7 =0, turn on belts
200E D3F3 OUT 0F3H ;Turn on LED to indicate S7 is on
2010 A1 ANA C ;Check S7 and S7'
2011 C21C20 JNZ SHTDWN
2014 78 TURNON: MOV A,B
2015 E61F ANI 1FH
2017 D3F3 OUT 0F3H
2019 C30020 JMP START
201C 3E40 SHTDWN: MVI A,40H ;Load byte to turn off belts and turn on emergency
201E D3F3 OUT 0F3H
2020 76 HLT
 END

```

## CHAPTER 7

The following programs assume the systems R/W memory begins at location 2000H. The symbols XX in the assignments are assumed as memory page 20H.

### Section 7.1

See Figures 7.1, 7.2, 7.3, & 7.4: pg. 81

### Section 7.2

5. Location 2075H will contain F7H

| 6.          | A | B  | C  | D  | E  | H  | L  |
|-------------|---|----|----|----|----|----|----|
| MVI C,FF    |   |    | FF |    |    |    |    |
| LXI H,2070H |   |    | FF |    |    | 20 | 70 |
| LXI D,2070H |   |    | FF | 20 | 70 | 20 | 70 |
| MOV M,C     |   |    | FF | 20 | 70 | 20 | 70 |
| LDAX D      |   | FF | FF | 20 | 70 | 20 | 70 |
| HLT         |   | FF | FF | 20 | 70 | 20 | 70 |

7. A = FFH (2070H) = FFH

8. 2075H and 2076H

9. A = 00H D = 00H HL = 209FH

10. Clears locations 2090H to 209FH

11.           LXI B,2090H  
               SUB A  
               MVI D,0FH  
 LOOP: STAX B  
               INX B  
               DCR D  
               JNZ LOOP  
               HLT

12. Infinite loop. DCX instruction does not affect Z flag.

13. 7 times.

14. DCX instruction does not affect Z flag.

15. START: LXI H, 2055H           ;Index for data source  
           LXI D, 2085H           ;Index for data destination, starting at last location  
           MVI B, 06H            ;Byte counter  
 NEXT: MOV A, M                ;Get data byte  
           STAX D                ;Store data byte  
           INX H                 ;Next location  
           DCX D  
           DCR B                 ;Next count  
           JNZ NEXT             ;If counter is not 0, go back to transfer next byte  
           HLT

16. START: LXI H, 205FH         ;Index pointing to last source byte  
           LXI D, 2064H         ;Index for destination

|     |        |                                                                                |                                                                                                             |
|-----|--------|--------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
|     | NEXT:  | MVI B, 10H<br>MOV A, M<br>STAX D<br>DCX H<br>DCX D<br>DCR B<br>JNZ NEXT<br>HLT | ;Byte counter<br>;Get data byte<br>;Store data byte<br>;Next location<br><br>;Next count                    |
| 17. | START: | LXI H, 2061H<br>LXI D, 2080H<br>MVI B, 05H                                     | ;Index pointing to low-order reading<br>;Index for storing low-order reading<br>;Counter for temp. readings |
|     | NEXT:  | MOV A, M<br>STAX D<br>INX H<br>INX H<br>INX D<br>DCR B<br>JNZ NEXT<br>HLT      | ;Get reading<br>;Store low-order reading<br><br>;Point to next low-order reading<br><br>;Next count         |
| 18. | START: | MVI B,6<br>LXI H,2050H<br>LXI D,2050H                                          | ;BYTE COUNT<br>;SOURCE<br>;DESTINATION                                                                      |
|     | LOOP:  | MOV A,M<br>ORA A<br>JNZ SKIP<br>STAX D<br>INX D                                | ;GET BYTE<br>;TEST IT FOR ZERO<br><br>;NOT ZERO, SO STORE IT                                                |
|     | SKIP:  | INX H<br>DCR B<br>JNZ LOOP<br>HLT                                              | ;GO ON TO NEXT                                                                                              |
| 19. | START: | LXI D, 2060H<br>MVI C, 05H<br>MVI B, 00H                                       | ;Index for data<br>;Counter for data<br>;Clear B to store partial sum                                       |
|     | NEXT:  | LDAX D<br>ADD B<br>MOV B, A<br>INX D<br>DCR C<br>JNZ NEXT<br>OUT PORT1<br>HLT  | ;Add data byte<br>;Save partial sum<br><br>;Next count<br><br>;Display sum                                  |

For the given five bytes, the sum is B7H

```

20. START: LXI H, 2060H ;Index for data
 MVI C, 05H ;Counter for data
 MVI B, 00H ;Clear B to store partial sum
NEXT: MOV A, M
 ADD B
 JC CARRY ;If sum > FF, display 01
 MOV B, A ;Save partial sum
 DCR C ;Next count
 JNZ NEXT
 OUT PORT1 ;Display sum
 HLT
CARRY: MVI A, 01H
 OUT PORT1
 HLT

```

First set display = 01H  
 Second set display = AFH

21. Clear register D to count the number of bytes added and insert the instruction INR D after the instruction JC CARRY. Display the contents of register D at PORT2 at the end.

First set display = 3 and Second set display = 5

Section 7.3:

22. Locations 2070H to 2074H will contain 01H, 02H, 03H, 04H, AND 05H

|             | A  | H  | L  | S  | Z  | CY | M 2055H |
|-------------|----|----|----|----|----|----|---------|
| LXI H,2055H |    | 20 | 55 | NA | NA | NA |         |
| MVI M,8AH   |    | 20 | 55 |    |    |    | 8A      |
| MVI A,76H   | 76 | 20 | 55 |    |    |    | 8A      |
| ADD M       | 00 | 20 | 55 | 0  | 1  | 1  | 8A      |
| STA 2055H   | 00 | 20 | 55 | NA | NA | NA | 00      |
| HLT         |    |    |    |    |    |    |         |

25. S = 0 Z = 1 CY is not affected (NA).

```

26. START: LXI D, 2050H ;Set up index pointer
 LXI H, 2040H ;Point to counter location in memory
 MVI M, 06H ;Load counter with the count
 XRA A ;Clear accumulator
 MOV C, A ;Clear C to save partial sum

```

|         |                                     |                                                      |
|---------|-------------------------------------|------------------------------------------------------|
|         | MOV B, A                            | ;Clear B to save carries                             |
| NEXT:   | LDAX D                              | ;Get data                                            |
|         | ADD C                               | ;Add previous partial sum                            |
|         | JNC SKIP                            | ;If no carry, do not increment CY register           |
| SKIP:   | INR B                               | ;Increment carry register                            |
|         | INX D                               | ;Point to next byte                                  |
|         | INX H                               |                                                      |
|         | DCR M                               | ;Next count                                          |
|         | JNZNEXT                             | ;If all numbers are not yet added, get the next byte |
|         | OUT PORT1                           | ;Display sum                                         |
|         | MOV A, B                            |                                                      |
|         | OUT PORT2                           | ;Display CY register                                 |
|         | HLT                                 |                                                      |
| 27.     | START: LXI D, 2050H                 | ;Set pointer for readings of furnace 1               |
|         | LXI H, 2060H                        | ;Set pointer for readings of furnace 2               |
|         | MVI C, 05H                          | ;Set up counter to count five readings               |
| NEXT:   | LDAX D                              | ;Place temp. reading from furnace 1 in A             |
|         | MOV B, M                            | ;Place temp. reading from furnace 2 in B             |
|         | SUB B                               |                                                      |
|         | JC SHTDWN                           | ;If (B) > (A), jump to display emergency             |
|         | INX D                               | ;Point to next memory location                       |
|         | INX H                               |                                                      |
|         | DCR C                               | ;Decrement reading count                             |
|         | JNZ NEXT                            | ;Get next temp. reading                              |
|         | MVI A, 01H                          | ;Set bit D0 = 1                                      |
|         | OUT PORT1                           | ;Turn on bit D0                                      |
|         | HLT                                 |                                                      |
| SHTDWN: | MVI A, 0FFH                         | ;Load emergency indicator                            |
|         | OUT PORT1                           | ;Display FF for emergency                            |
|         | HLT                                 |                                                      |
| 28.     | Add, after the statement JC SHTDWN: |                                                      |
|         | JNZ NOTEQ                           | ;If they are not same, skip                          |
|         | MVI A, 80H                          | ;If they are, load byte to turn on D7                |
|         | OUT PORT1                           | ;Turn on bit D7                                      |
| NOTEQ:  | continue                            |                                                      |
| 29.     | START: LXI H, 2070H                 | ;DATA LOCATION                                       |
|         | MVI B, 4                            | ;NUMBER OF PAIRS                                     |
| LOOP:   | MOV A, M                            | ;GET FIRST BYTE OF PAIR                              |
|         | INX H                               | ;POINT TO SECOND BYTE                                |
|         | ADD M                               | ;ADD 2ND TO 1ST                                      |
|         | DCX H                               | ;POINT TO LOCATION TO STORE SUM                      |
|         |                                     | ;NOTE: DCX AND INX DO NOT AFFECT CARRY               |

```

MOV M,A ;STORE SUM
INX H ;POINT TO CARRY POSITION
MVI A,0 ;GET CARRY INTO A
ADC A,A
MOV M,A ;STORE IT TO 2ND LOCATION
INX H ;POINT TO NEXT PAIR
DCR B ;NEXT COUNT
JNZ LOOP ;IF COUNT IS NOT ZERO, GET NEXT PAIR
HLT

30. START: LXI H,2070H ;Data location
 LXI D,2070H ;Storage location
 MVI B,4 ;Number of pairs
 LOOP: MOV A,M ;Get first byte of pair
 INX H ;Point to second byte
 SUB M ;Subtract 2nd from 1st
 STAX D ;Store sum
 INX D ;Bump sum pointer
 INX H ;Point to next pair
 DCR B ;Decrement count
 JNZ LOOP ;Go back for next operation
 HLT

31. START: LXI H,2070H ;Data location
 LXI D,2070H ;Storage location
 MVI B,4 ;Number of pairs
 LOOP: MOV A,M ;Get first byte of pair
 INX H ;Point to second byte
 SUB M ;Subtract 2nd from 1st
 JC SKIP ;Skip if < 0 (negative) unsigned
 STAX D ;Store sum
 INX D ;Bump sum pointer
 SKIP: INX H ;Point to next pair
 DCR B ;Decrement count
 JNZ LOOP ;Go back for next operation
 HLT

```

Section 7.4:

32. (a) A = 6FH, CY = 1      (b) A = 6EH, CY = 1

|     |           |    |   |   |    |
|-----|-----------|----|---|---|----|
| 33. |           | A  | S | Z | CY |
|     | MVI A,80H | 80 |   |   |    |
|     | ORA A     | 80 | 1 | 0 | 0  |
|     | RAR       | 00 | 0 | 1 | 1  |



34. (a) 

|           |    |   |
|-----------|----|---|
| MVI A,C5H | C5 |   |
| ORA A     | C5 | 0 |
| RAL       | 8A | 1 |
| RRC       | 45 | 0 |

 (b) 

|           |    |   |
|-----------|----|---|
| MVI A,A7H | A7 |   |
| ORA A     | A7 | 0 |
| RAR       | 4E | 1 |
| RAL       | A7 | 0 |
35. These instructions will move the MSD of a BCD number (7 in this case) from 10th to the 1's place. A = 07
36. (a) Multiply by 2 (b) Divide by 4
37. Multiply by 10.
38. 

|        |              |                                       |
|--------|--------------|---------------------------------------|
| START: | MVI D, 0AH   | ;Set up counter to count ten readings |
|        | LXI H, 2060H | ;Set pointer to data location         |
|        | MVI C, 00H   | ;Clear register C to save partial sum |
|        | MOV B, C     | ;Clear register B for carry           |
| LOOP:  | MOV A, M     | ;Get current reading                  |
|        | ORA A        | ;Set flags                            |
|        | JM NEXT      | ;If D7 = 1, reject the data byte      |
|        | ADD C        | ;Add the current reading              |
|        | MOV C, A     | ;Save partial sum of the readings     |
|        | JNC NEXT     | ;If no carry, do not increment (B)    |
|        | INR B        | ;Add 1 to previous carry count        |
| NEXT:  | INX H        | ;Go to next data location             |
|        | DCR D        | ;Next count                           |
|        | JNZ LOOP     | ;Go back to get next reading          |
|        | MOV A, C     | ;Get the final sum                    |
|        | OUT PORT1    | ;Display low-order byte of the sum    |
|        | MOV A, B     | ;Get the carry count                  |
|        | OUT PORT2    | ;Display carry count                  |
|        | HLT          |                                       |

For the given set of readings in assignment 38, the sum = 2A0H

39. 

|        |              |                                           |
|--------|--------------|-------------------------------------------|
| START: | MVI D, 0AH   | ;Set up counter to count ten readings     |
|        | LXI H, 2060H | ;Set pointer to data location             |
|        | MVI C, 00H   | ;Clear register C to save partial sum     |
|        | MOV B, C     | ;Clear register B for carry               |
|        | MOV E, C     | ;Clear regi. E to count positive readings |
| LOOP:  | MOV A, M     | ;Get current reading                      |
|        | ORA A        | ;Set flags                                |
|        | JM NEXT      | ;If D7 = 1, reject the data byte          |
|        | INR E        | ;Positive reading found; count it         |
|        | ADD C        | ;Add the current reading                  |

|       |        |              |                                       |
|-------|--------|--------------|---------------------------------------|
|       |        | MOV C, A     | ;Save partial sum of the readings     |
|       |        | JNC NEXT     | ;If no carry, do not increment (B)    |
|       |        | INR B        | ;Add 1 to previous carry count        |
| NEXT: |        | INX H        | ;Go to next data location             |
|       |        | DCR D        | ;Next count                           |
|       |        | JNZ LOOP     | ;Go back to get next reading          |
|       |        | MOV A, C     | ;Get the final sum                    |
|       |        | OUT PORT1    | ;Display low-order byte of the sum    |
|       |        | MOV A, B     | ;Get the carry count                  |
|       |        | OUT PORT2    | ;Display carry count                  |
|       |        | MOV A, E     | ;Get number for positive readings     |
|       |        | OUT PORT3    | ;Display number for positive reading  |
|       |        | HLT          |                                       |
| 40.   | START: | LXI D, 2060H | ;Set pointer to data storage location |
|       |        | LXI H, 2050H | ;Set pointer to data location         |
|       |        | MVI C, 08H   | ;Set counter to count eight bytes     |
| LOOP: |        | MOV A, M     | ;Get the byte                         |
|       |        | ORA A        | ;Set flag if D7 = 1                   |
|       |        | JM NEXT      | ;Jump to increment data pointer       |
|       |        | RRC          | ;Place D0 in CY position              |
|       |        | JC NEXT      | ;Jump to increment data pointer       |
|       |        | RLC          | ;Restore original byte                |
|       |        | STAX D       | ;Save the byte at storage location    |
|       |        | INX D        | ;Next storage location                |
| NEXT: |        | INX H        | ;Next data location                   |
|       |        | DCR C        | ;Next count                           |
|       |        | JNZ LOOP     | ;Jump back to get next byte           |
|       |        | HLT          |                                       |

Answer: The following memory locations should have the data bytes as shown.

2060 = 52H  
 2061 = 78H  
 2062 = 62H

Section 7.5:

|     |           |    |   |   |    |
|-----|-----------|----|---|---|----|
| 41. |           | A  | S | Z | CY |
|     | MVI A,7FH | 7F |   |   |    |
|     | ORA A     | 7F | 0 | 0 | 0  |
|     | CPI A2H   | 7F | 1 | 0 | 1  |

42. A S Z CY

```
LXI H,2070H
MVI M,64H
MVI A,8FH 8F
CMP M 8F 0 0 0
```

43. 00, 00, 7A, 87, 00, 00

44. 100 (64H) < BYTE <= 200 (C8H)

45. 20, 64, 8F

46. If PORT1 < 32 (20H), CY flag is set.

or if PORT1 = > 160 (A0H), Minus flag is set.

```
47. START: LXI H, 2050H ;Set index to point to data
 MVI C, 00H ;Clear register C to save sum
 MOV B, C ;Clear (B) to save carry
 MOV D, C ;Set register D to compare bytes
NXTBYT: MOV A, M ;Get data
 CMP D ;Check whether this the last byte
 JZ DSPLY ;If 'Yes', go to display section
 ADD C ;Add previous sum
 JNC SAVE ;If there is no carry, go to save the sum
 INR B ;Update carry register
SAVE: MOV C,A ;Save sum
 INX H ;Point to next reading
 JMP NXTBYT ;Go back to get next reading
DSPLY: MOV A, C ;Display low-order byte of the sum
 OUT PORT1 ;Copy the carry count to accumulator
 MOV A, B ;Display high-order byte of sum
 OUT PORT2
 HLT

48. START: LXI H, 2050H ;Set index to point to data location
 MVI D, 40H ;Byte to be found in data string
 MVI C, 08H ;Set up counter
NXTBYT: MOV A, M ;Get data byte
 CMP D ;Is the byte = 40H?
 JZ DSPLY ;If yes, go to display its location
 INX H ;Point to next data byte
 DCR C ;Next count
 JNZ NXTBYT ;Jump to get next byte
```

|          |                     |                                          |
|----------|---------------------|------------------------------------------|
|          | MVI A, FFH          |                                          |
|          | OUT PORT1           | ;The byte 40H is not in the set          |
|          | HLT                 |                                          |
| DISPLAY: | MOV A, H            | ;Load high-order memory address          |
|          | OUT PORT1           | ;Display memory page number              |
|          | MOV A, L            | ;Load low-order memory address           |
|          | OUT PORT2           | ;Display memory low-order address        |
|          | HLT                 |                                          |
| 49.      | START: LXI H, 2050H | ;Set index to point to data location     |
|          | MVI C, 08H          | ;Set up counter                          |
|          | MVI B, 00H          | ;Clear (B) to save the highest reading   |
| NXTBYT:  | MOV A, M            | ;Get data byte                           |
|          | CMP B               | ;Is (B) > (A)?                           |
|          | JNC NEXT            | ;If yes, replace (B) with (A)            |
|          | MOV B, A            | ;Save the larger number                  |
| NEXT:    | INX H               | ;Point to next data byte                 |
|          | DCR C               | ;Next count                              |
|          | JNZ NXTBYT          | ;Jump to get next byte                   |
|          | MOV A, B            | ;Load the largest byte                   |
|          | OUT PORT1           | ;Display the largest byte in the string  |
|          | HLT                 |                                          |
| 50.      | START: LXI H, 2050H | ;Set index to point to data location     |
|          | MVI C, 08H          | ;Set up counter                          |
|          | MVI B, 00H          | ;Clear (B) to save the highest reading   |
| NXTBYT:  | MOV A, M            | ;Get data byte                           |
|          | CMP B               | ;Is (B) < (A)?                           |
|          | JC NEXT             | ;If yes, replace (B) with (A)            |
|          | MOV B, A            | ;Save the larger number                  |
| NEXT:    | INX H               | ;Point to next data byte                 |
|          | DCR C               | ;Next count                              |
|          | JNZ NXTBYT          | ;Jump to get next byte                   |
|          | MOV A, B            | ;Load the largest byte                   |
|          | OUT PORT1           | ;Display the largest byte in the string  |
|          | HLT                 |                                          |
| 51.      | START: LXI H, 2050H | ;SOURCE POINTER                          |
|          | LXI D, 2050H        | ;SAVE POINTER                            |
|          | MVI B, 10           | ;BYTE COUNT                              |
| LOOP:    | MOV A, M            |                                          |
|          | CPI 60              |                                          |
|          | JC REJECT           | ;REJECT IF < 60                          |
|          | CPI 101             | ;NOTE: IF SUBTRACT 100, WOULD REJECT 100 |
|          | JNC REJECT          | ;REJECT IF > 100                         |

|     |         |             |                                          |
|-----|---------|-------------|------------------------------------------|
|     |         | STAX D      | ;OK, SO SAVE IT                          |
|     |         | INX D       |                                          |
|     | REJECT: | INX H       | ;LOOP FOR NEXT                           |
|     |         | DCR B       |                                          |
|     |         | JNZ LOOP    |                                          |
|     |         | HLT         |                                          |
| 52. | START:  | LXI H,2050H | ;SOURCE POINTER                          |
|     |         | LXI D,2050H | ;SAVE POINTER                            |
|     |         | MVI B,10    | ;BYTE COUNT                              |
|     |         | MVI C,0     |                                          |
|     | LOOP:   | MOV A,M     |                                          |
|     |         | CPI 60      |                                          |
|     |         | JC REJECT   | ;REJECT IF < 60                          |
|     |         | CPI 101     | ;NOTE: IF SUBTRACT 100, WOULD REJECT 100 |
|     |         | JNC REJECT  | ;REJECT IF > 100                         |
|     |         | STAX D      | ;OK, SO SAVE IT                          |
|     |         | INX D       |                                          |
|     |         | INR C       | ;AND COUNT IT                            |
|     | REJECT: | INX H       | ;LOOP FOR NEXT                           |
|     |         | DCR B       |                                          |
|     |         | JNZ LOOP    |                                          |
|     |         | MOV A,C     | ;DISPLAY COUNT                           |
|     |         | OUT PORT1   |                                          |
|     |         | HLT         |                                          |
| 53. | START:  | LXI H,2070H | ;SOURCE POINTER                          |
|     |         | LXI D,2090H | ;SAVE POINTER                            |
|     | LOOP    | MOV A,M     |                                          |
|     |         | CPI 0DH     | ;CHECK FOR END OF STRING                 |
|     |         | JZ ENDS     | ;IF =0DH, THEN END OF SRTING             |
|     |         | CPI 30H     |                                          |
|     |         | JC REJECT   | ;REJECT IF < 30H                         |
|     |         | CPI 3AH     | ;NOTE: IF SUBTRACT 39H, WOULD REJECT 39H |
|     |         | JNC REJECT  | ;REJECT IF > 39H                         |
|     |         | STAX D      | ;OK, SO SAVE IT                          |
|     |         | INX D       |                                          |
|     | REJECT: | INX H       | ;LOOP FOR NEXT                           |
|     |         | JMP LOOP    |                                          |
|     | ENDS:   | HLT         |                                          |
| 54. | START:  | LXI H,2070H | ;SOURCE POINTER                          |
|     |         | LXI D,2090H | ;SAVE POINTER                            |
|     |         | MVI C,0     |                                          |
|     | LOOP:   | MOV A,M     |                                          |

```

CPI 0DH ;CHECK FOR END OF STRING
JZ ENDS ;IF =0DH, THEN END OF SRTING
CPI 30H
JC REJECT ;REJECT IF < 30H
CPI 3AH ;NOTE: IF SUBTRACT 39H, WOULD REJECT 39H
JNC REJECT ;REJECT IF > 39H
STAX D ;OK, SO SAVE IT
INX D
INR C ;AND COUNT IT
REJECT: INX H ;LOOP FOR NEXT
JMP LOOP
ENDS: MOV A,C ;DISPLAY COUNT
 OUT PORT1
 HLT

```

```

55. START: LXI H,XXXXH ;SOURCE POINTER FOR SCANNER READINGS
 MVI C,0 ;COUNTER TO COUNT SETS
 LOOP: MOV A,M ;GET SCANNER READING
 ORA A ;SET ZERO FLAG
 JZ ENDS ;IF BYTE =0, THEN END OF STRING
 CPI A3H ;IS IT A 19" SET
 JNZ SKIP;IF NO, DO NOT COUNT IT
 INC C;COUNT THE SET
 SKIP: INX H
 JMP LOOP
 ENDS: HLT

```

56. This assignment is similar to the Illustrative Program 7.53 except that the order is descending and the number of bytes = 10. Therefore, the counter register C should be set for 9 and the instruction JC NXTBYT should be replaced by JNC NXTBYT.

#### Section 7.6:

57. (a) 256 (b) 2090H-2091H  
(c) 1) A was not initialized to 00 2) Missing INX H in loop

58. SET 1 SET 2

(a) E7H 9BH  
(d) DCR B should be DCR C and add the instruction MOV A,B just before (STA 2070H) storing the answer

59. SET 1

- (a) 016FH
- (d) Change JNZ 2008H to JNZ 2007H

SET 2

- (a) 039CH
- (d) ADC M should be ADD M

### CHAPTER 8

- 1. 1168 uSec
- 2. 234.67 mSec
- 3. 468.584675 mSec
- 4. Count 12FFH =  $1 \times 16 + 2 \times 16 + 15 \times 16 + 15 \times 16$   
= 4863

The delay in the loop:

$$\begin{aligned} &= \text{T-states} \times \text{Clock Period} \times \text{Count} \\ &= 64 \times (.33 \times 10^{-6}) \times 4863 \\ &= 102 \text{ ms} \end{aligned}$$

The above calculations are based on the assumption that the JMP instruction takes 10 T-states in the last iteration, and the initial instruction LXI B is not part of the delay loop. To account for the 7 T-states in the last iteration, the delay of 0.99 micro-sec ( $.33 \times 10^{-6} \times 3 = .99 \times 10^{-6}$ ) should be subtracted from the above delay.

- 5. 234.1313 mSec
- 6. (a) 4  
(b) infinite  
(c) infinite or just once if the flag is set initially.
- 7. (a) infinite (b) infinite (c) 1
- 8. 1.465 mSec
- 9. If the system frequency is 3.072 MHz, the clock period will be 325 ns. This will reduce the delay to .325 s.
- 10. COUNT = 0EH

11. BC = 34965<sub>10</sub> Insignificant difference when the delay is calculated with JNZ = 7 T-states in the last iteration.
12. 12799 (31FFH)

Note: In the following assignments, the delay calculations are based on the SDK-85 system with the frequency of 3.072 MHz (T = 325.5 ns).

|     | Mnemonics         | Comments                     | T-states |
|-----|-------------------|------------------------------|----------|
| 13. | START: MVI B, 00H | ;Initialize counter          |          |
|     | DSPLAY: OUT PORT1 | ;Display count               | 10       |
|     | LXI D, COUNT      | ;Load delay count            | 10       |
|     | LOOP: DCX D       |                              | 6        |
|     | MOV A, E          |                              | 4        |
|     | ORA D             | ;Set Z flag if (D) & (E) = 0 | 4        |
|     | JNZ LOOP          | ;If Z = 1, repeat the loop   | 10/7     |
|     | INR B             | ;Next count                  | 4        |
|     | MOV A, B          |                              | 4        |
|     | CPI 21H           | ;Is count = 21H?             | 7        |
|     | JNZ DSPLAY        | ;If not, go to display count | 10/7     |
|     | JMP START         | ;Reset counter, start again  |          |

Calculations for 100 ms delay:

Delay outside the loop:

$$T_o = 45 \text{ T-states} \times 325.5 \text{ ns} = 14648 \text{ ns}$$

Delay within the loop:

$$T = 24 \text{ T-states} \times 325.5 \text{ ns} \times \text{COUNT}$$

$$\text{Total Delay } T = T_o + T$$

$$100 \text{ ms} = 14648 \text{ ns} + (24 \times 325.5 \text{ ns} \times \text{COUNT})$$

$$\text{COUNT} = \frac{100 \times 10^{-3} - 14648 \times 10^{-9}}{7812 \times 10^{-9}} = 12,798 = 31\text{FEH}$$



```

14. START: MVI B, 00H ;Start Up-Counter
 DSPLY1: OUT PORT1
 MVI C, COUNT1 ;COUNT1 = 10
 LOOP1: LXI D, COUNT2 ;COUNT2 should provide 100 ms delay
 LOOP2: DCX D
 MOV A, E
 ORA D
 JNZ LOOP2
 DCR C
 JNZ LOOP1
 INR B
 MOV A, B
 CPI 0AH ;Check whether 9 is displayed
 JNZ DSPLY1
 DCR B
 DSPLY2: OUT PORT1
 MVI C, COUNT1 ;COUNT1 = 10
 LOOP3: LXI D, COUNT2 ;COUNT2 should provide 100 ms delay
 LOOP4: DCX D
 MOV A, E
 ORA D
 JNZ LOOP4
 DCR C
 JNZ LOOP3
 MOV A, B
 ORA A ;Set Z flag if (A) = 0
 JNZ DSPLY2
 JMP START

```

For delay calculations, see Assignment #7.

|     |         |               |                         |          |
|-----|---------|---------------|-------------------------|----------|
| 15. | START:  | MVI D, 00H    | ;Load bit pattern       | T-States |
|     | ROTATE: | MOV A, D      |                         | 4        |
|     |         | CMA           | ;Complement bit pattern | 4        |
|     |         | MOV D,A       |                         | 4        |
|     |         | ANI 80H       | ;Mask D6-D0             | 7        |
|     |         | OUT PORT1     |                         | 10       |
|     |         | MVI E, COUNT1 |                         | 7        |
|     | LOOP1:  | LXI B, COUNT2 |                         | 10       |
|     | LOOP2:  | DCX B         |                         | 6        |
|     |         | MOV A, C      |                         | 4        |
|     |         | ORA B         |                         | 4        |
|     |         | JNZ LOOP2     |                         | 10/7     |
|     |         | DCR E         |                         | 4        |
|     |         | JNZ LOOP1     |                         | 10       |
|     |         | JMP ROTATE    |                         | 10       |

To design 5-second delay using the loop within the loop technique, it is assumed that the LOOP2 will provide 100 ms delay, and it will be repeated 50 times by using the COUNT1 = 32H in the outer loop.

$$\begin{aligned} \text{LOOP1 Delay:} &= 24T \times (325.5 \text{ ns}) \times 50 \\ &= 0.39 \text{ ms} \end{aligned}$$

$$\text{Delay Outside the loop:} = 39T \times (325.5 \text{ ns}) = 0.0127 \text{ ms}$$

$$\text{Total Delay} = \text{LOOP1} + \text{LOOP2} \times 50 + \text{Outside Delay}$$

$$5 \text{ s} = .39 \text{ ms} + (24 \times 325.5 \text{ ns} \times \text{COUNT2}) 50 + .0128 \text{ ms}$$

$$\text{COUNT2} = 12799.78$$

The COUNT2 can be calculated by ignoring all the delays except in the LOOP2 as follows:

$$5 \text{ s} = \text{LOOP2} \times 50$$

$$= (24 \times 325.5 \text{ ns} \times \text{COUNT2}) \times 50$$

$$\text{COUNT2} = 12800$$

The difference between the two calculations is 0.22; it is insignificant.

|     |                   |                         |      |
|-----|-------------------|-------------------------|------|
| 16. | START: MVI D, 00H | ;Load bit pattern       |      |
|     | ROTATE: MOV A, D  |                         | 4    |
|     | CMA               | ;Complement bit pattern | 4    |
|     | MOV D,A           |                         | 4    |
|     | ANI 01H           | ;Mask D7-D1             | 7    |
|     | OUT PORT1         |                         | 10   |
|     | MVI B, COUNT      |                         | 7    |
|     | LOOP: DCR B       |                         | 4    |
|     | JNZ LOOP          |                         | 10/7 |
|     | JMP ROTATE        |                         | 10   |

$$\text{Total Delay } T = T_o + T$$

$$200 \text{ s} = 46 \times 325.5 \text{ ns} + 14 \times 325.5 \text{ ns} \times \text{COUNT}$$

$$\text{COUNT} = 42.5 \approx 42$$

|     |                    |                            |    |
|-----|--------------------|----------------------------|----|
| 17. | TURNON: MVI A, 01H | ;Bit pattern to turn on D0 | 7  |
|     | OUT PORT1          | ;On-period begins          | 10 |

|        |                                        |      |
|--------|----------------------------------------|------|
|        | MVI B, COUNT1 ;Count for 200 s pulse   | 7    |
| LOOP1: | DCR B                                  | 4    |
|        | JNZ LOOP1                              | 10/7 |
|        | MVI A, 00H ;Bit pattern to turn off D0 | 7    |
|        | OUT PORT1 ;Off-period begins           | 10   |
|        | MVI B, COUNT2 ;Count for 400 s delay   | 7    |
| LOOP2: | DCR B                                  | 4    |
|        | JNZ LOOP2                              | 10/7 |
|        | JMP TURNON                             | 10   |

On-period Delay:

$$T = T_o + T$$

$$200 = (24 T \times 325.5 \text{ ns}) + (14 T \times 325.5 \text{ ns} \times \text{COUNT1})$$

$$\text{COUNT1} = \frac{(200 - 7.81) \times 10}{(4.557 \times 10)} = 42$$

Off-period Delay:

$$T = T_o + T$$

$$400 = (34 T \times 325.5 \text{ ns}) + (14 T \times 325.5 \text{ ns} \times \text{COUNT2})$$

$$\text{COUNT2} = \frac{(400 - 11.06)}{(4.557 \times 10)} = 85$$

18. START: MVI L,10101010B ;ALTERNATING LIGHT PATTERN  
LIGHTS: MOV A,L  
RRC  
OUT PORT  
MOV L,A  
MVI B,50 ;20 x 50 mSec DELAY = 1 Sec  
OUTER: LXI D,2559 ;20 mSec DELAY  
INNER: DCX D  
MOV A,D  
ORA E  
JNZ INNER  
DCR B  
JNZ OUTER  
JMP LIGHTS