6

Programming of 8085

In this chapter assembly language programming of different programs will be discussed. It will help readers to go into the details of 8085 programming operations and their applications. For this, one is supposed to be well acquainted with 8085 instruction set discussed in the preceding chapters. Though some programming examples have also been discussed in these chapters, yet this chapter will exclusively deal with some more programming examples. The assembly language program written by the programmer in mnemonic form is also called as source program. The instructions, operand and data may be converted to binary (machine language) either by hand assembler or machine assembler. In case of hand assembler the hexadecimal data (op code / operand and data) are fed to the microprocessor kit through the hexadecimal key board. But in machine assembler, instructions (in mnemonic form) with data or operand are directly fed to the microprocessor kit through Alpha-numeric key board (computer key board). Here we are concerned with the assembly language program (source program).

In computers high level language like C, C++, Pascal, BASIC etc are used, which are easier than the assembly language. The computer is used for the conversion of high level language to assembly language. The advantage of using assembly language is that it can directly go into the details of the microprocessor's register and manipulate as the requirement.

6.1 SIMPLE PROGAMS

The simple programs based on the instruction set of 8085 microprocessor are given in the following examples.

Example 6.1. *Write an assembly language program of 8085 to find 1's complement of the data stored in memory location 2500 H and the result is to be stored in memory location 2501 H.*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LDA	2500 H	; Load the data from 2500 H memory location to accumulator.
	CMA		; Complement the contents of accumulator (Converts in 1's complement).
	STA	2501 H	; Store the result in memory location 2501 H.
	HLT		; Stop processing.

Example 6.2. *Write an assembly language program of 8085 to find 2's complement of the data stored in memory location 2100 H and the result is to be stored in memory location 2101 H.*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LDA	2100 H	; Load the data from 2100 H memory location to accumulator.
	CMA		; Complement the contents of accumulator (Converts in 1's complement).
	ADI	01 H	; 01 is added to the accumulator contents to get the 2's complement.
	STA	2101 H	; Store the result in memory location 2101 H.
	HLT		; Stop processing.

Example 6.3. *Write an assembly language program of 8085 to find 1's complement of n (decimal number) bytes of data stored in memory location starting at 2501 H. The number n (decimal number) is stored in memory location 2500 H. Store the result in memory locations starting at 2601 H.*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI D,	2601 H	; Get first address of the destination in D-E register pair.
	LXI H,	2500 H	; Get H-L pair with 2500 H.
	MOV C,	M	; Data in 2500 H is loaded to C-register which will be used as counter.
	INXH		; Increment H-L register pair.
LOOP	MOV A,	M	; Accumulator is loaded with the data.
	CMA		; Complement the contents of accumulator (Converts in 1's complement)
	STAX D		; Store the result in memory location addressed by D-E register pair.
	INX D		; Increment D-E register pair.
	INX H		; Increment H-L register pair.
	DCR C		; Decrement the C-register data.
	JNZ	LOOP	; If data in C-reg. is not zero jump to LOOP for next conversion.
	HLT		; Stop processing.

Example 6.4. *Write an assembly language program of 8085 to find 9's complement of n bytes (in BCD) of data stored in memory location starting from 2501 H. The number n*

(decimal number) is stored in memory location 2500 H. Store the result in memory locations starting from 2601 H.

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI D,	2601 H	; Get first address of the destination in D-E register pair.
	LXI H,	2500 H	; Get H-L pair with 2500 H.
	MOV C,	M	; Data in 2500 H is loaded to C-register which will be used as counter.
	INX H		; Increment H-L register pair.
LOOP	MVI A,	99 H	; Get 99 H in accumulator.
	SUB M		; Subtract each byte by 99 H to get 9's complement.
	STAX D		; Store the result in memory location addressed by D-E register pair.
	INX D		; Increment D-E register pair.
	INX H		; Increment H-L register pair.
	DCR C		; Decrement the C-register data.
	JNZ	LOOP	; If data in C-reg. is not zero jump to loop for next conversion.
	HLT		; Stop processing.

It is clear from this example that for 9's complement each BCD byte is subtracted from 99.

Example 6.5. *Write an assembly language program of 8085 to find 2's complement of a 16 bit number stored in memory locations 2101 H and 2102 H. The least significant byte is in 2101 H. The result is to be stored in memory locations 2103 H and 2104 H.*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2101 H	; Get H-L pair with 2101 H.
	MVI B,	00 H	; Store 00 to register B which will be used to store carry.
	MOV A,	M	; Accumulator is loaded with the data.
	CMA		; Complement the contents of accumulator (Converts in 1's complement).
	ADI	01 H	; Add 01 H to the accumulator content to get 2's complement.
	STA	2103 H	; Store the result in memory location 2103 H.
	JNC	NXT	; If there is no carry jump to NXT.
	INR B		; Increment 1 to B-register as carry.

```
      NXT   INX H                  ; Increment H-L register pair for the
                                     second byte.
            MOV A,       M         ; Move the second byte to
                                     accumulator.
            CMA                    ; Complement the contents of
                                     accumulator.
            ADD B                  ; Add carry stored in register B.
            STA          2504 H    ; Store in 2504 H.
            HLT                    ; Stop processing.
```

From this example it is clear that 1 is added to the 1’s complement of LS byte and to the 1’s complement of MS byte 1 is added if there is a carry from the previous byte.

Example 6.6. *Write an assembly language program of 8085 to find 10’s complement of a 16 bit number (BCD) stored in memory locations 2101 H and 2102 H. The least significant byte is in 2101 H. The result is to be stored in memory locations 2103 H and 2104 H.*

Solution.

Program:

```
Label       Mnemonics    Operand   Comments
            LXI  H,      2101 H    ; Get H-L pair with 2101 H.
            MVI B,       00 H      ; Store 00 to register B which will be
                                     used to store carry.
            MVI A,       99 H      ; Store 99 in accumulator
            SUB M                  ; Subtract each byte by 99 H to get
                                     9’s complement.
            ADI          01 H      ; Add 01 H to the accumulator
                                     content to get 10’s complement.
            STA          2103 H    ; Store the result in memory location
                                     2103 H.
            JNC          NXT       ; If there is no carry jump to NXT.
            INR B                  ; Increment 1 to B-register as carry.
      NXT   INX H                  ; Increment H-L register pair for the
                                     second byte.
            MVI A,       99 H      ; Store 99 in accumulator
            SUB M                  ; Subtract each byte by 99 H to get
                                     9’s complement.
            ADD B                  ; Add carry stored in register B.
            STA          2104 H    ; Store in 2104 H.
            HLT                    ; Stop processing.
```

From this example it is clear that 1 is added to the 9’s complement of LS byte and to the 9’s complement of MS byte 1 is added if there is a carry from the previous byte.

Example 6.7. *Write an assembly language program of 8085 to find 2’s complement of N bytes ($N \geq 2$). The number of bytes N in hexadecimal is stored in 2100 H. The bytes are stored in memory locations starting at 2101 H. The least significant byte is in 2101 H. The result is to be stored in memory locations starting at 2201 H.*

Solution.
Program:

```
Label       Mnemonics    Operand     Comments
            LXI  H,      2100 H      ; Get H-L pair with 2100 H.
            LXI D,       2201 H      ; Get D-E pair with 2201 H.
            MOV C,       M           ; Get the number N in c-register to be
                                       used as counter.
            INX  H                   ; Increment H-L pair.
            MOV A,       M           ; Accumulator is loaded with first
                                       data.
            CMA                      ; Complement the contents of
                                       accumulator (Converts in 1’s
                                       complement).
            ADI          01 H        ; Add 01 H to the accumulator
                                       content to get 2’s complement.
            STAX D                   ; Store the result in memory location
                                       addressed by D-E pair.
            JNC          NXT         ; If there is no carry jump to NXT.
            MVI B,       01 H        ; Store 01 to B-register as carry.
            JMP          NXT1        ; Jump to NXT1.
NXT         MVI B,       00 H        ; Store 00 to B-register as carry.
NXT1        DCR C                    ; Decrement C-reg.
LOOP        INX  H                   ; Increment H-L pair.
            INX  D                   ; Increment D-E pair.
            MOV A,       M           ; Move the next byte to accumulator.
            CMA                      ; Complement the contents of
                                       accumulator.
            ADD B                    ; Add carry stored in register B.
            STAX D                   ; Store in the memory location
                                       addressed by D-E pair.
            JNC          NXT2        ; If there is no carry jump to NXT2.
            MVI B,       01 H        ; Store 01 to B-register as carry.
            JMP          NXT3        ; Jump to NXT3.
NXT2        MVI B,       00 H        ; Store 00 to b-register as carry.
NXT3        DCR C                    ; Decrement C-reg.
            JNZ          LOOP        ; If not zero jump to LOOP.
            HLT                      ; Stop processing.
```

Example 6.8. *Write an assembly language program of 8085 to find 10’s complement of N bytes ($N \geq 2$). The number of bytes N in hexadecimal is stored in 2100 H. The bytes are stored in memory locations starting at 2101 H. The least significant byte is in 2101 H. The result is to be stored in memory locations starting at 2201 H.*

Solution.
Program:

```
Label       Mnemonics    Operand     Comments
            LXI  H,      2100 H      ; Get H-L pair with 2100 H.
            LXI D,       2201 H      ; Get D-E pair with 2201 H.
```

```
          MOV C,     M          ; Get the number N in c-register to be
                                  used as counter.
          INX  H                ; Increment H-L pair.
          MVI A,     99 H       ; Accumulator is loaded with 99 H.
          SUB M                 ; Get the 9's Complement of the
                                  number stored in location
                                  addressed by H-L pair.
          ADI        01 H       ; Add 01 H to the accumulator
                                  content to get 2's complement.
          STAX D                ; Store the result in memory location
                                  addressed by D-E pair.
          JNC        NXT        ; If there is no carry jump to NXT.
          MVI B,     01 H       ; Store 01 to B-register as carry.
          JMP        NXT1       ; Jump to NXT1.
NXT       MVI B,     00 H       ; Store 00 to b-register as carry.
NXT1      DCR C                 ; Decrement C-reg.
LOOP      INX  H                ; Increment H-L pair.
          INX  D                ; Increment D-E pair.
          MVI A,     99 H       ; Get 99 in accumulator.
          SUB M                 ; Get 9's Complement of the
                                  contents of accumulator.
          ADD B                 ; Add carry stored in register B to
                                  get 10's complement.
          STAX  D               ; Store in the memory location
                                  addressed by D-E pair.
          JNC        NXT2       ; If there is no carry jump to NXT2.
          MVI B,     01 H       ; Store 01 to B-register as carry.
          JMP        NXT3       ; Jump to NXT3.
NXT2      MVI B,     00 H       ; Store 00 to b-register as carry.
NXT3      DCR C                 ; Decrement C-reg.
          JNZ        LOOP       ; If not zero jump to LOOP.
          HLT                   ; Stop processing.
```

Example 6.9. *Write an assembly language program of 8085 to combine two hex nibbles stored in 2500 H and 2501 H memory locations, to form a byte. The least significant nibble is stored in 2500 H and most significant nibble is stored in 2501 H. The byte thus combined should be stored in 2502 H. (Let 08 H is stored in 2500 H and 09 H is stored in 2501 H, after the program is executed 2502 H should be loaded with the combined byte 89 H).*

Solution.

Program:

```
Label     Mnemonics  Operand    Comments
          LXI  H,    2500 H     ; Get H-L pair with 2500 H.
          MOV A,     M          ; Data in 2500 H is loaded to
                                  Accumulator.
          RLC                   ; Rotate left
          RLC                     ; Rotate left
          RLC                     ; Rotate left
          RLC                     ; Rotate left (Rotated left four times
                                    so that it is moved to MS nibble).
          INX H                   ; Increment H-L register pair.
          ORA M                   ; Oring of two nibbles combines
                                    them to form a byte in
                                    accumulator.
          INX H                   ; Increment H-L register pair.
          MOV M,      A           ; Move the accumulator content
                                    (required byte) to the memory
                                    location addressed by H-L register
                                    pair.
          HLT                     ; Stop processing.
```

Example 6.10. *Write an assembly language program of 8085 to separate a hexadecimal number into two nibbles. The hexadecimal number is stored in 2501 H memory location. The least significant nibble of the byte is to be stored in 2502 memory location and the most significant nibble is to be stored in 2503 H memory location. (Suppose 3A H is a byte stored in memory location 2501 H and the lower nibble 0A should be stored in 2502 H and 03 should be stored in 2503 H)*

Solution.

Program:

```
Label     Mnemonics   Operand     Comments
          LXI  H,     2501 H      ; Get H-L pair with 2501 H.
          MOV A,      M           ; Data in 2501 H is loaded to
                                    Accumulator.
          MOV B,      A           ; Accumulator content are also
                                    loaded to B register.
          ANI         0F H        ; Mask off the first four digits
                                    (higher nibble).
          INX H                   ; Increment H-L register pair.
          MOV M,      A           ; Lower nibble is loaded to the
                                    memory location addressed by H-L
                                    register pair.
          MOV A,      B           ; Get the byte again in the
                                    accumulator.
          ANI         F0 H        ; Mask off the lower nibble.
          INX  H                  ; Increment H-L register pair.
          MOV  M,     A           ; Higher nibble is loaded to the
                                    memory location addressed by H-L
                                    register pair.
          HLT                     ; Stop processing.
```

Example 6.11. *Write an assembly language program of 8085 to check a hexadecimal number stored in 2500 H memory location for odd or even parity. If the parity is even store data EE H to memory location 2501H otherwise store 00 H in 2501 H.*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2500 H	; Get H-L pair with 2500 H.
	MOV A,	M	; Data in 2500 H is loaded to Accumulator.
	ORA A		; Set the flag.
	JPE	EVEN	; Check for even parity, if parity is even jump to EVEN.
	INX H		; Increment H-L register pair.
	MVI M,	00 H	; Load 00 H to the memory location addressed by H-L register pair.
	JMP	DONE	; Jump to DONE.
EVEN	INX H		; Increment H-L register pair.
	MVI M,	EE H	; Load EE H to the memory location addressed by H-L register pair.
DONE	HLT		; Stop processing.

Example 6.12. *n (decimal number) data bytes are stored in the memory locations starting at 2501 H. Write an assembly language program of 8085 to check if 12 H is stored in any of the given locations. If any of the locations has 12 H then store 12 H in that location else load 00 H in that memory location.*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2500 H	; Get H-L pair with 2500 H.
	MOV C,	M	; Data in 2500 H is loaded to C reg. which is used as the counter.
LOOP	INX H		; Increment H-L register pair.
	MOV A,	M	; Load the content of M_{HL} to accumulator.
	CPI	12 H	; Compare if the accumulator data are 12 H.
	JZ	NXT	; If it is 12 H then jump to NXT.
	MVI M,	00 H	; Else load 00 H to the memory location addressed by H-L register pair.
	JMP	DONE	; Jump to DONE.
NXT	MVI M,	12 H	; Load 12 H to the memory location addressed by H-L register pair.

DONE	DCR C		; Decrement the content of C-reg. for next byte.
	JNZ	LOOP	; If the contents of C-reg. are not zero then jump to LOOP.
	HLT		; Stop processing.

Example 6.13. *16 data bytes are stored in memory locations 2001 H to 2010 H. Write an assembly language program of 8085 to transfer this block of data bytes to memory locations 2501 to 2510 H.*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2001 H	; Get H-L pair with 2001 H.
	LXI D,	2501 H	; Get the address of the memory location (destination) in D-E register pair
	MVI C,	10 H	; Get 10 H (16_{10}) Data in C register which is used as the counter.
LOOP	MOV A,	M	; Load the content of M_{HL} to accumulator.
	STAX D		; Load the content of accumulator to the memory locations addressed by D-E register pair.
	INX H		; Increment H-L register pair.
	INX D		; Increment D-E register pair.
	DCR C		; Decrement the content of C-reg. for next byte.
	JNZ	LOOP	; If the contents of C-reg. are not zero then jump to LOOP.
	HLT		; Stop processing.

Example 6.14. *16 data bytes are stored in memory locations 2001 H to 2010 H. Write an assembly language program of 8085 to transfer this block of data bytes to memory locations 2501 to 2510 H in the reverse order (i.e. the data of 2001 is to transferred to 2510 H and data of 2002 H to 250F H and so on).*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2001 H	; Get H-L pair with 2001 H.
	LXI D,	2510 H	; Get the address of the memory location (destination) in D-E register pair
	MVI C,	10 H	; Get 10 H (16_{10}) Data in C register which is used as the counter.

LOOP	MOV A,	M	; Load the content of M_{HL} to accumulator.
	STAX D		; Load the content of accumulator to the memory locations addressed by D-E register pair.
	INX H		; Increment H-L register pair.
	DCX D		; Decrement D-E register pair.
	DCR C		; Decrement the content of C-reg. for next byte.
	JNZ	LOOP	; If the contents of C-reg. are not zero then jump to LOOP.
	HLT		; Stop processing.

Example 6.15. *Write an assembly language program of 8085 to clear the memory locations starting at 2001 (i.e. each location should have 00 H). The length of data bytes is given in 2000 H memory location.*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2000 H	; Get H-L pair with 2000 H.
	MOV C,	M	; Use C register as counter.
	XRA A		; Clear the accumulator.
LOOP	INX H		; Increment H-L register pair.
	MOV M,	A	; Load the accumulator content to the memory location.
	DCR C		; Decrement the content of C-reg. for next byte.
	JNZ	LOOP	; If the contents of C-reg. are not zero then jump to LOOP.
	HLT		; Stop processing.

Example 6.16. *Write an assembly language program of 8085 to add five bytes of data stored in memory locations starting at 2000 H. If the sum generates a carry, stop addition and store 01 H to the memory location 2501 H; else continue addition and store the sum at 2501 memory location..*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2000 H	; Get H-L pair with 2000 H.
	MVI C,	05 H	; Store the number of data bytes in C register as counter.
	XRA A		; Clear the accumulator and CY flag.
LOOP	ADD M		; Add the byte in accumulator.

```
            JC          NXT          ; If there is a carry jump to NXT.
            INX H                    ; Increment the H-L register pair.
            DCR C                    ; Decrement the content of C-reg.
                                       for next byte.
            JNZ         LOOP         ; If the contents of C-reg. are not
                                       zero then jump to LOOP.
            STA         2501 H       ; Store the sum in memory location
                                       2501 H.
            HLT                      ; Stop processing.
  NXT       MVI A,      01 H         ; If carry occurs load 01 to
                                       accumulator.
            STA         2501 H       ; Store 01 H to 2501 H.
            HLT                      ; Stop processing.
```

Example 6.17. *Write an assembly language program of 8085 to add five bytes of data stored in memory locations starting at 2000 H. The sum may be more than one byte. Store the result at two consecutive memory locations2500 H and 2501 H.*

Solution.

Program:

```
Label       Mnemonics   Operand      Comments
            LXI  H,     2000 H       ; Get H-L pair with 2000 H.
            MVI C,      05 H         ; Store the number of data bytes in C
                                       register as counter.
            XRA A                    ; Clear the accumulator and CY flag.
            MOV B,      A            ; Store the accumulator contents to
                                       B-register also for the carry.
  LOOP      ADD M                    ; Add the byte in accumulator.
            JNC         NXT          ; If there is a no carry jump to NXT.
            INR B                    ; Increment B as carry is generated.
  NXT       INX H                    ; Increment the H-L register pair.
            DCR C                    ; Decrement the content of C-reg.
                                       for next byte.
            JNZ         LOOP         ; If the contents of C-reg. are not
                                       zero then jump to LOOP.
            LXI  H,     2500 H       ; Store H-L pair with 2500 H
                                       (destination address).
            MOV  M,     A
            INX  H                   ; Store the sum to 2500 H.
            MOV M,      B            ; Content of B are stored in 2501 H.
            HLT                      ; Stop processing.
```

Example 6.18. *Write a program in assembly language of 8085 to test 6th bit (D_6 bit) of a byte stored at 2500 H memory location. If the bit D_6 is zero then store 00 H at 2501 H else store the same number at 2501 H.*

Solution.

Program:

Label	Mnemonics	Operand	Comments
	LXI H,	2500 H	; Get H-L pair with 2500 H.
	MOV A,	M	; Store the byte in accumulator
	ANI	40 H	; Reset all the bits except D_6.
	JZ	END	; If this bit is zero jump to END.
	MOV A,	M	; Store the byte in accumulator again.
	INX H		; Increment H-L pair.
	MOV M,	A	; The byte is stored in 2501 H.
	HLT		; Stop processing.
END	XRA A		; Clear accumulator.
	INX H		; Increment H-L pair.
	MOV M,	A	; 00 H is stored in 2501 H.
	HLT		; Stop processing.

Example 6.19. *Write an ALP (Assembly Language Program) of 8085 to find logical OR and exclusive OR of two numbers stored at 2501 H and 2502 H memory locations. The results should be stored at 2503 H (logical OR operation) and 2504 H (XOR operation) memory locations.*

Solution.

Program:

Label	Mnemonics	Operand	Comments
	LXI H,	2501 H	; Get H-L pair with 2501 H.
	MOV B,	M	; Store the byte in B-register.
	MOV A,	B	; Store this byte in accumulator also.
	INX H		; Increment H-L pair.
	MOV C,	M	; Store second byte to C register.
	ORA C		; Logical OR of two bytes.
	INX H		; Increment H-L pair.
	MOV M,	A	; Store the result (OR) in 2503 H.
	MOV A,	B	; Load the first number in accumulator.
	XRA C		; Ex-OR the two number.
	INX H		; Increment H-L pair.
	MOV M,	A	; Store the result (XOR)
	HLT		; Stop processing.

Example 6.20. *Write an ALP (Assembly Language Program) for 8085 to shift an 8-bit number left by one bit. The number is stored in 2101 H memory location. The result is to be stored in 2102 H memory location.*

Solution.

Program:

Label	Mnemonics	Operand	Comments
	LDA	2101 H	; Get the number is accumulator.
	ADD A		; Shift it left by one bit.
	STA	2102 H	; Store the result in 2502 H.
	HLT		; Stop processing.

Example 6.21. *Write an ALP 8085 to shift a 16-bit number left by one bit. The number is stored in 2101 H and 2102 H memory locations. The result is to be stored in 2103 H and 2104 H memory locations.*

Solution.

Program:

Label	Mnemonics	Operand	Comments
	LHLD	2101 H	; Get the 16 bit number in H-L register pair.
	DAD H		; Shift left by one bit.
	SHLD	2103 H	; Store the result in 2103 H and 2104 H memory locations.
	HLT		; Stop processing.

6.2 PROGAMS ON CODE CONVERSION

The programs on code conversion will now be discussed.

6.2.1 BCD to Binary Conversion

Suppose we have 0 to 99 decimal numbers (BCD numbers) and we wish to convert any of these numbers to its equivalent binary number, we proceed in the following way:

For example 49 (4 x 10 + 9) is the given decimal number, its binary equivalent is ($00011\ 0001_2$ = 31 H).

Step I	Unpack the number in MSD and LSD form.	
	0000 1001	LSD in four least significant nibble in a register.
	0000 0100	MSD in four least significant nibble in anther register.
Step II	Multiply MSD by decimal number 10.	
	or MSD is added 10 times with 0000.	
	So we get (4 x 10 = 0010 1000).	
Step III	ADD LSD to the result of step II.	
	i.e. 00101000 + 00001001 = 00110001	
	Hence we get the result.	

Example 6.22. *A BCD number (0 to 99) is stored in a memory location 2500 H, write an ALP of 8085 to convert the BCD number into its equivalent binary number. Store the result in 2600 H memory location.*

Solution.

Main Program:

Label	Mnemonics	Operand	Comments
	LXI SP,	XXXX H	; Initialize the stack pointer.
	LXI H,	2500 H	; Get H-L pair with 2500 H.
	LXI D,	2600 H	; Get D-E pair with 2600 H.
	MOV A,	M	; Load the byte to accumulator.
	CALL	CONV	; Call conversion subroutine program.
	STAX D		; Store the result in 2600 H.
	HLT		

Subroutine Program:

```
Label       Mnemonics   Operand     Comments
  CONV      PUSH D                  ; Push the contents of D-E register
                                      pair to stack.
            PUSH H                  ; Push the contents of H-L register
                                      pair to stack.
            ANI         0F H        ; Separate LSD
            MOV C,      A           ; Store it to C-register.
            MOV  A,     M           ; Load the byte to accumulator again.
            ANI         F0 H        ; Separate MSD.
            RRC                     ; Rotate right four times
            RRC                     ; to shift MSD to four
            RRC                     ; least significant nibble
            RRC                     ; of accumulator.
            MOV D,      A           ; Store MSD to D register.
            XRA A                   ; Clear accumulator.
            MVI E,      0A H        ; Get 0A (decimal number 10) to E-
                                      register.
  SUM       ADD D                   ; Add MSD ten times to 00 H.
            DCR E                   ; Decrement C
            JNZ         SUM         ; If addition not complete then move
                                      to SUM.
            ADD C                   ; Add LSD to 10 X MSD
            POP H                   ; Pop the contents of H-L pair.
            POP D                   ; Pop the contents of D-E pair.
            RET                     ; Go back to main program.
```

6.2.2 Binary to BCD (Unpacked) Conversion

For example binary number is 1111 1111 (FF H). Its decimal equivalent is 255_{10} having the unpacked BCD numbers as

05	(0000 0101)	BCD 1
05	(0000 0101)	BCD 2
02	(0000 0010)	BCD 3

The procedure for converting the 8 bit binary number (0000 0000 to 1111 1111) into unpacked BCD number is given in the following steps.

Step I	If the number is less than 100 go to step II. If the number is more than 100, divide the number by 100 or subtract 100 repeatedly till the reminder is less than 100. The quotient is MS BCD (BCD 3).
Step II	If the number (or remainder) is less than 10, go to step III. If number is $>10 < 100$, then subtract 10 repeatedly till the remainder is less than 10. The quotient is BCD 2.
Step III	The remainder from step II is BCD 1.

Example 6.23. *A binary number (between 00 H to FF H) is stored in a memory location 2500 H, write an ALP of 8085 to convert this number into BCD number. Store each BCD as unpacked BCD digit in the memory locations at 2601 H to 2603 H.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI SP,	XXXX H	; Initialize the stack pointer.
	LXI H,	2500 H	; Get H-L pair with 2500 H.
	MOV A,	M	; Load the byte to accumulator.
	CALL	CONV	; Call conversion subroutine program.
	HLT		

Subroutine Program 1:

Label	**Mnemonics**	**Operand**	**Comments**
CONV	LXI H,	2601 H	; Get H-L pair with 2601 H.
	MVI B,	64 H	; Load 100 in B-reg.
	CALL	CONV1	; Call another subroutine program for the division by 100.
	MVI B,	0A H	; Load 10 in B-reg.
	CALL	CONV1	; Call the subroutine again for the division by 10.
	MOV M,	A	; Load the accumulator contents to the memory location.
	RET		; Go back to main program.

Subroutine Program 2:

Label	**Mnemonics**	**Operand**	**Comments**
CONV1	MVI M,	FF H	; Get FF H in the $M_{H\text{-}L}$.
NXT	INR M		; Increment $[M_{H-L}]$
	SUB B		; Subtract the content of B from accumulator.
	JNC	NXT	; Jump to NXT if no carry.
	ADD B		; Add the contents of B to accumulator.
	INX H		; Increment H-L register pair.
	RET		; Go back.

Example 6.24. *Write ALP of 8085 for the following statement:*

A set of three packed BCD numbers (six digits or three bytes) are stored in memory locations starting at 2500 H. The seven segment codes of the digits 0 to 9 for common cathode LED are stored in memory locations starting at 2050 H; and the answer is to be stored in memory locations starting at 2070 H.

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI SP,	XXXX H	; Initialize the stack pointer.
	LXI H,	2500 H	; Get H-L pair with 2500 H.

	MVI D,	03 H	; Number of bytes to be converted is placed in D-register.
	CALL	CONV	; Call conversion subroutine program.
	HLT		

Subroutine Program 1:

Label	Mnemonics	Operand	Comments
CONV	LXI B,	2070 H	; Get B-C pair with 2070 H.
NXT	MOV A,	M	; Move the content of $[M_{H-L}]$to accumulator.
	ANI	F0 H	; Separate MSD.
	RRC		; Rotate right four times
	RRC		; to shift MSD to four
	RRC		; least significant nibble
	RRC		; of accumulator.
	CALL	SEGMENT	; Call subroutine program for the conversion of unpacked BCD to seven segment.
	INX B		; Increment B-C register pair.
	MOV A,	M	; Get the byte again.
	ANI	0F H	; Separate LSD.
	CALL	SEGMENT	; Call subroutine program for the conversion of unpacked BCD (LSD) to seven segment.
	INX B		; Increment B-C register pair.
	INX H		; Increment H-L register pair.
	DCR D		; Decrement the contents of D register.
	JNZ	NXT	; If the content in D register is not then jump to NXT for next byte.
	RET		; Return to main program.

Subroutine Program 2:

Label	Mnemonics	Operand	Comments
SEGMENT	PUSH H		; Push the contents of H-L register pair to stack.
	LXI H,	2050 H	; Get H-L pair with 2050 H.
	ADD L		; Add the content of L register to accumulator to get right location for the digit from the look-up table.
	MOV L,	A	; Store it to L-register.
	MOV A,	M	; Load the corresponding data from the look-up table to accumulator.
	STAX B		; Store the result in the memory location addressed by B-C register pair.

```
            POP H                   ; Get the contents of H-L register
                                      pair from the stack.
            RET                     ; Go back
```

DATA (in the form of look-up table) is stored in the following memory locations:

2050 H	3F H	(Digit 0)
2051 H	06 H	(Digit 1)
2052 H	5B H	(Digit 2)
2053 H	4F H	(Digit 3)
2054 H	66 H	(Digit 4)
2055 H	6D H	(Digit 5)
2056 H	7D H	(Digit 6)
2057 H	07 H	(Digit 7)
2058 H	7F H	(Digit 8)
2059 H	6F H	(Digit 9)

The 3F H data is given for the digit 0. It is clear from the figure 6.1.

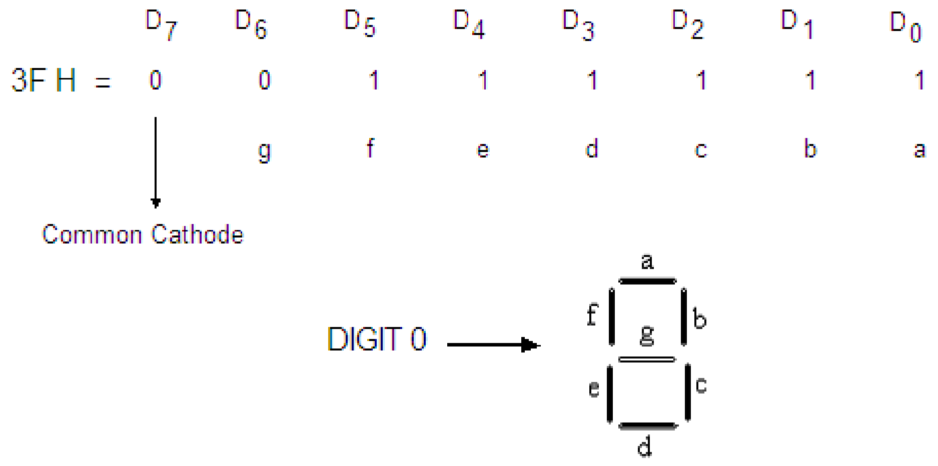

Fig. 6.1

6.2.3 Binary to ASCII Conversion

ASCII (American Standard Code for Information Interchange) is a most commonly used alphanumeric code. It is a seven bit code. The hexadecimal numbers 30 to 39 (0011 0000 to 0011 1001) represent 0 to 9 ASCII decimal numbers. Similarly, 41 H to 5A H (0100 0001 to 0101 1010) represent capital letters A to Z in ASCII.

For example, the ASCII representation of a binary number 8E is 38 H and 45 H (as 8 is represented by 0011 1000 or 38 H; and E is represented by 0100 0101 or 45 H).

The following steps are carried out for the conversion of Binary to ASCII

Step I	First separate the LSD and MSD of the given binary number (or byte). Each digit is then converted to ASCII.
Step II	If the digit is less than 10 (0A H) then add 30 H (0011 0000) to the binary number else add 37 H (0011 0111).

For binary number 8E H, 30 is added to 08 and we get 38 H (the ASCII Code for 8) and 37 is added to 0E H to get 45 H (the ASCII Code for E).

Example 6.25. *An eight bit binary number is stored in 2500 H. Write an ALP for 8085 to convert the binary number to its equivalent ASCII code. The ASCII code for most* *significant binary digit should be stored to 2601 H location; and the code for least significant binary digit should be stored to 2602 H location.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI SP,	XXXX H	; Initialize the stack pointer.
	LXI H,	2500 H	; Get H-L pair with 2500 H.
	LXI D,	2601 H	; Get D-E pair with 2601 H.
	MOV A,	M	; Move the content of $[M_{H-L}]$to accumulator.
	MOV B,	A	; Move the accumulator content to B-register.
	RRC		; Rotate right four times
	RRC		; to shift MSD to four
	RRC		; least significant nibble
	RRC		; of accumulator.
	CALL	ASCII	; Call the subroutine to convert MSD to ASCII.
	STAX D		; Store the ASCII code of MS digit to the memory location addressed by D-E register pair.
	INX D		; Increment D-E register pair.
	MOV A,	B	; Move the binary number again to accumulator.
	CALL	ASCII	; Call the subroutine to convert LSD to ASCII.
	STAX D		; Store the ASCII code of LS digit to the memory location addressed by D-E register pair.
	HLT		; Stop processing.

Subroutine Program:

Label	**Mnemonics**	**Operand**	**Comments**
ASCII	ANI	0F H	; Isolate MS digit.
	CPI	0A	; Compare with 10 (0A H).
	JC	NXT	; If it less than 10, jump to NXT.
	ADI	07 H	; Else add 07 H.
NXT	ADI	30 H	; Add 30 H.
	RET		; Go back to main program.

6.2.4 ASCII to Binary Conversion

The following steps are carried out to convert the ASCII code to binary (Hex):

Step I	Subtract 30 H (0011 0000) from the given binary (Hex) number.
Step II	If the difference after subtraction in step I is less than 10, then it is the required binary (Hex) number.

Step III	If the difference after subtraction in step II is more than 10 then subtract 07 also from it. This will then give the required binary (Hex) number.

Example 6.26. *An ASCII number is stored in 2500 H memory location. Write an ALP for 8085 to convert this number into its equivalent binary (Hex) number and store it to 2501 memory location.*

Solution.

Main Program:

Label	Mnemonics	Operand	Comments
	LXI SP,	XXXX H	; Initialize the stack pointer.
	LXI H,	2500 H	; Get H-L pair with 2500 H.
	MOV A,	M	; Move the content of $\left[M_{H-L}\right]$to accumulator.
	CALL	CONV	; Call the subroutine to convert ASCII to binary.
	INX H		; Increment H-L register pair.
	MOV M,	A	; Store the binary number to required memory location.
	HLT		; Stop processing.

Subroutine Program:

Label	Mnemonics	Operand	Comments
CONV	SUI	30 H	; Subtract 30 H from accumulator.
	CPI	0A	; Compare with 10 (0A H).
	RC		; If it less than 10 go back to main program.
	SUI	07 H	; Else subtract 07 H.
	RET		; Go back to main program.

6.3 PROGAMS ON ADDITION AND SUBTRACTION

Now we shall take up the problems on BCD or decimal addition and subtraction of two bytes or multibyte. These problems are self explanatory and can be understood by considering proper logic.

Example 6.27. *Write an ALP for 8085 to add two 16-bit numbers. The augend's LS byte is stored in 2101 H and MS byte in 2102 H. The addend's LS and MS byte are stored in 2103 H and 2104 H respectively. The result is to be stored in 2105 H to 2107 H.*

Solution.

Main Program:

Label	Mnemonics	Operand	Comments
	LHLD	2101 H	; Load the H-L pair direct with the contents of 2101 H and 2102 H Locations.
	XCHG		; Exchange the contents of H-L and D-E pair.
	LHLD	2103 H	; Load the H-L pair direct with the contents of 2103 H and 2104 H Locations.

```
            MVI  C,     00 H        ; Store 00 H to C register for carry.
            DAD  D                  ; Add the contents of H-L and D-E
                                      register pair and result is in H-L
                                      pair.
            JNC         NXT         ; If there is no carry after the
                                      addition jump to NXT.
            INR  C                  ; Increment the carry.
  NXT       SHLD        2105 H      ; Store the contents in the required
                                      locations.
            MOV  A,     C           ; Move carry to accumulator.
            STA         2107 H      ; Store carry in 2107 H.
            HLT                     ; Stop processing.
```

Example 6.28. *Write an ALP for 8085 to add two N byte numbers. The augend's bytes are stored in memory locations starting at 2101 H and the addend's bytes are stored in memory locations starting at 2201 H. The number N is stored in memory location 2100 H. The result is to be stored in the memory locations starting at 2201 H.*

Solution.

Main Program:

```
Label       Mnemonics   Operand     Comments
            LXI  H,     2100 H      ; Get H-L pair with 2100 H.
            MOV C,      M           ; Move the number N to C-register
                                      for counter.
            INX  H                  ; Increment the H-L pair.
            LXI  D,     2201 H      ; Get D-E pair with 2201 H.
            XRA  A                  ; Clear the accumulator and carry
                                      flag.
  LOOP      LDAX  D                 ; Get the addend byte to
                                      accumulator.
            ADC  M                  ; Add with carry the two bytes.
            MOV  M,     A           ; Store the result in the location
                                      addressed by H-L pair.
            INX  D                  ; Increment the D-E pair.
            INX  H                  ; Increment the H-L pair.
            DCR  C                  ; Decrement C.
            JNZ         LOOP        ; If C is not zero, then jump to
                                      LOOP.
            HLT                     ; Stop processing.
```

Example 6.29. *Write an ALP for 8085 to add two N byte numbers. The augend's bytes are stored in memory locations starting at 2101 H and the addend's bytes are stored in memory locations starting at 2201 H. The number N is stored in memory location 2100 H. The result should in decimal form and is to be stored in the memory locations starting at 2201 H.*

Solution.

Main Program:

```
Label       Mnemonics   Operand     Comments
```

```
            LXI  H,     2100 H      ; Get H-L pair with 2100 H.
            MOV C,      M           ; Move the number N to C-register
                                      for counter.
            INX  H                  ; Increment the H-L pair.
            LXI  D,     2201 H      ; Get D-E pair with 2201 H.
            XRA  A                  ; Clear the accumulator and carry
                                      flag.
  LOOP      LDAX  D                 ; Get the addend byte to
                                      accumulator.
            ADC  M                  ; Add with carry the two bytes.
            DAA                     ; Decimal adjust the accumulator for
                                      the result in decimal form.
            MOV  M,     A           ; Store the result in the location
                                      addressed by H-L pair.
            INX  D                  ; Increment the D-E pair.
            INX  H                  ; Increment the H-L pair.
            DCR  C                  ; Decrement C.
            JNZ         LOOP        ; If C is not zero, then jump to
                                      LOOP.
            HLT                     ; Stop processing.
```

It may be noted from this example that DAA (decimal adjust the accumulator) instruction is used after ADC M for the conversion of the result in decimal form.

Example 6.30. *Write an ALP for 8085 for multibyte (N bytes) subtraction. The number of bytes (N) is stored in 2100 H location. The minuend bytes are stored in the memory locations starting at 2101 H and the subtrahend bytes are stored in the memory locations starting at 2201 H. The answer should be stored in the memory locations starting at 2101 H.*

Solution.

Main Program:

```
Label       Mnemonics   Operand     Comments
            LXI  H,     2100 H      ; Get H-L pair with 2100 H.
            MOV C,      M           ; Move the number N to C-register
                                      for counter.
            INX  H                  ; Increment the H-L pair.
            LXI  D,     2201 H      ; Get D-E pair with 2201 H.
            XRA  A                  ; Clear the accumulator and carry
                                      flag.
  LOOP      LDAX  D                 ; Get the addend byte to
                                      accumulator.
            SBB  M                  ; Subtract with borrow the contents
                                      of [M_{H-L}] from the accumulator
                                      contents.
            MOV  M,     A           ; Store the result in the location
                                      addressed by H-L pair.
            INX  D                  ; Increment the D-E pair.
            INX  H                  ; Increment the H-L pair.
```

	DCR C		; Decrement C.
	JNZ	LOOP	; If C is not zero, then jump to LOOP.
	HLT		; Stop processing.

Decimal Subtraction

It may be noted from the above example that SBB M (subtract with borrow) instruction is used for the subtraction. The result will not be obtained in the decimal form as DAA instruction can not be used after Subtract instructions. For decimal subtractions, the subtrahend number should be converted to its equivalent 10's complement which will then be added to minuend.

Example 6.31. *Write an ALP for 8085 to subtract 78 from 96. The answer in decimal form should be stored in 2100 H memory location.*

Solution.

Main Program:

Label	Mnemonics	Operand	Comments
	MVI C,	96 H	; Get 96 in C register.
	MVI B,	78 H	; Get 78 in B register.
	MVI A,	99 H	; Get 99 in accumulator.
	SUB B		; Subtract the content of B-register from 99 to get the 9's complement of 78.
	INR A		; Increment accumulator to get 10's complement.
	ADD C		; Add the content of C to the accumulator.
	DAA		; Decimal adjust the accumulator, to get the answer in decimal for.
	STA	2100 H	; Store the answer in decimal form at 2100 H location.
	HLT		; Stop processing.

Example 6.32. *Write an ALP for 8085 for multibyte (N bytes) decimal subtraction. The number of bytes (N) is stored in 2100 H location. The minuend bytes are stored in the memory locations starting at 2101 H and the subtrahend bytes are stored in the memory locations starting at 2201 H. The answer obtained in decimal form should be stored in the memory locations starting at 2101 H.*

Solution.

Main Program:

Label	Mnemonics	Operand	Comments
	LXI SP,	XXXX H	; Initialize the stack pointer.
	LXI H,	2100 H	; Get H-L pair with 2100 H.
	MOV C,	M	; Move the number N to C-register for counter.
	INX H		; Increment the H-L pair.
	LXI D,	2201 H	; Get D-E pair with 2201 H.
	LDAX D		; Get the subtrahend byte in accumulator.

	MOV B,	A	; Move the accumulator content to B-register.
	MVI A,	99 H	; Store 99 to accumulator.
	SUB B		; Get 9’s complement of the subtrahend.
	INR A		; Get 10’s complement of the subtrahend.
	ADD M		; Add 10’s complement of the subtrahend to minuend.
	DAA		; Give the answer in decimal form.
	PUSH PSW		; Push the carry to stack.
	MOV M,	A	; Store the result in the location addressed by H-L pair.
LOOP	INX D		; Increment the D-E pair.
	INX H		; Increment the H-L pair.
	LDAX D		; Get the second number to accumulator.
	MOV B,	A	; Store it to B-register.
	MVI A,	99 H	; Store 99 to accumulator.
	SUB B		; Get 9’s complement.
	MOV B,	A	; Store it to B-register.
	POP PSW		; Get the carry of the previous addition from the stack.
	MOV A,	B	; Get the B-register content to accumulator.
	ADC M		; Add with carry.
	DAA		; Give the answer in decimal form.
	PUH PSW		; Store the carry to stack for further addition.
	MOV M,	A	; Store the result in the memory location addressed by H-L register pair.
	DCR C		; Decrement the content of C-register.
	JNZ	LOOP	; If C is not zero, then jump to LOOP.
	HLT		; Stop processing.

6.4 PROGAMS TO FIND LARGEST OR SMALLEST NUMBER

Example 6.33 *Write an ALP for 8085 to find the larger of two numbers stored in memory locations 2101 H and 2102 H. Store the result in memory location 2103 H.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2101 H	; Get H-L pair with 2101 H.
	MOV A,	M	; Move the first number in accumulator.

	INX H		; Increment the H-L pair.
	CMP M		; Compare the second number with first number.
	JNC	NXT	; If the accumulator content is larger then jump to NXT.
	MOV A,	M	; Else move $[M_{H-L}]$ to accumulator.
NXT	INX H		; Increment the H-L pair.
	MOV M,	A	; Store the result in the required memory location.
	HLT		; Stop processing.

Note: To get the smaller number from the given two numbers, use the instruction JC in place of JNC.

Example 6.34. *Write an ALP for 8085 to find the largest number among three numbers stored in memory locations staring at 2101 H. Store the result in memory location 2104H.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2101 H	; Get H-L pair with 2101 H.
	MOV A,	M	; Move the first number in accumulator.
	INX H		; Increment the H-L pair.
	CMP M		; Compare the second number with first number.
	JNC	NXT	; If the accumulator content is larger then jump to NXT.
	MOV A,	M	; Else move $[M_{H-L}]$ to accumulator.
NXT	INX H		; Increment the H-L pair.
	CMP M		; Compare the third number with the larger of first two numbers.
	JNC	NXT1	; If the accumulator content is larger then jump to NXT1.
	MOV A,	M	; Else move $[M_{H-L}]$ to accumulator.
NXT1	INX H		; Increment the H-L pair.
	MOV M,	A	; Store the result in the required memory location.
	HLT		; Stop processing.

Example 6.35. *Write an ALP for 8085 to find the largest number among a series of N numbers stored in memory locations staring at 2101 H. The number N is stored in memory location 2100 H. Store the result in memory location 2201 H.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**

```
          LXI  H,     2100 H     ; Get H-L pair with 2100 H.
          MOV  C,     M          ; Store the number N in C-register
                                   which will be used as the counter.
          INX  H                 ; Increment the H-L pair.
          MOV  A,     M          ; Move the first number in
                                   accumulator.
          DCR  C                 ; Decrement count.
  LOOP    INX  H                 ; Increment the H-L pair.
          CMP  M                 ; Compare the second number with
                                   first number.
          JNC         NXT        ; If the accumulator content is larger
                                   then jump to NXT.
          MOV  A,     M          ; Else move $[M_{H-L}]$ to accumulator.
  NXT     DCR  C                 ; Decrement count.
          JNZ         LOOP       ; Jump to LOOP if not zero.
          STA         2201 H     ; Store the result in 2201 H.
          HLT                    ; Stop processing.
```

6.5 PROGAMS TO ARRANGE A GIVEN SERIES IN ASCENDING OR DESCENDING ORDER

Example 6.36. *Write an ALP for 8085 to arrange a series of N-numbers in descending order. The number N is stored in memory location 2100 H. The series is stored in memory location starting at 2101 H. The result is to be stored in memory locations starting at 2201 H.*

Solution.

Main Program:

```
Label     Mnemonics   Operand    Comments
          LXI  SP,    XXXX H     ; Initialize stack pointer.
          LXI  D,     2201 H     ; Get D-E pair with 2201 H.
          LXI  H,     2100 H     ; Get H-L pair with 2100 H.
          MOV  B,     M          ; Store the count N in B-register.
START     LXI  H,     2100 H     ; Get H-L pair with 2100 H.
          MOV  C,     M          ; Store the number N in C-register
                                   also.
          INX  H                 ; Increment the H-L pair.
          MOV  A,     M          ; Move the first number in
                                   accumulator.
          DCR  C                 ; Decrement count.
  LOOP    INX  H                 ; Increment the H-L pair.
          CMP  M                 ; Compare the second number with
                                   first number.
          JNC         NXT        ; If the accumulator content is larger
                                   then jump to NXT.
          MOV  A,     M          ; Else move $[M_{H-L}]$ to accumulator.
  NXT     DCR  C                 ; Decrement count.
          JNZ         LOOP       ; Jump to LOOP if not zero.
          STAX  D                     ; Store the largest number in
                                        memory location addressed by D-E
                                        register pair.
          CALL       SUBR             ; Call the subroutine SUBR to put
                                        00 H to the memory location where
                                        the largest number was found.
          INX  D                      ; Increment D-E register pair.
          DCR  B                      ; Decrement B.
          JNZ        START            ; Jump to START if not zero.
          HLT                         ; Stop processing.
```

Subroutine Program:

```
Label     Mnemonics  Operand          Comments
SUBR      LXI  H,    2100 H           ; Get H-L pair with 2100 H.
          MOV  C,    M                ; Store the number N is C-register
                                        which will be used as the counter.
 AGAIN    INX  H                      ; Increment the H-L pair.
          CMP  M                      ; Compare the number with the
                                        largest number obtained in the
                                        main program.
          JZ         NXT              ; If it is largest jump to NXT.
          DCR C                       ; Decrement count.
          JNZ        AGAIN            ; If counts not complete jump to
                                        AGAIN.
          MVI  A,    00 H             ; Store 00 H to the accumulator.
          MOV  M,    A                ; Put 00 H to the location at which
                                        the largest number was found.
          RET                         ; Go back to main program.
```

Example 6.37. *Write an ALP for 8085 to arrange a series of N-numbers in ascending order. The number N is stored in memory location 2100 H. The series is stored in memory location starting at 2101 H. The result is to be stored in memory locations starting at 2201 H.*

Solution.

Main Program:

```
Label     Mnemonics  Operand          Comments
          LXI  SP,   XXXX H           ; Initialize stack pointer.
          LXI  D,    2201 H           ; Get D-E pair with 2201 H.
          LXI  H,    2100 H           ; Get H-L pair with 2100 H.
          MOV  B,    M                ; Store the count N in B-register.
START     LXI  H,    2100 H           ; Get H-L pair with 2100 H.
          MOV  C,    M                ; Store the number N in C-register
                                        also.
          INX  H                      ; Increment the H-L pair.
          MOV  A,    M                ; Move the first number in
                                        accumulator.
          DCR  C                      ; Decrement count.
 LOOP     INX  H                      ; Increment the H-L pair.
```

```
            CMP  M                      ; Compare the second number with
                                          first number.
            JC          NXT             ; If the accumulator content is
                                          smaller then jump to NXT.
            MOV  A,     M               ; Else move [M_{H-L}] to accumulator.
  NXT       DCR  C                      ; Decrement count.
            JNZ         LOOP            ; Jump to LOOP if not zero.
            STAX  D                     ; Store the largest number in
                                          memory location addressed by D-E
                                          register pair.
            CALL        SUBR            ; Call the subroutine SUBR to put
                                          FF H to the memory location
                                          where the smallest number was
                                          found.
            INX  D                      ; Increment D-E register pair.
            DCR  B                      ; Decrement B.
            JNZ         START           ; Jump to START if not zero.
            HLT                         ; Stop processing.
```

Subroutine Program:

```
Label       Mnemonics   Operand         Comments
SUBR        LXI  H,     2100 H          ; Get H-L pair with 2100 H.
            MOV C,      M               ; Store the number N is C-register
                                          which will be used as the counter.
  AGAIN     INX  H                      ; Increment the H-L pair.
            CMP M                       ; Compare the number with the
                                          smalest number obtained in the
                                          main program.
            JZ          NXT             ; If it is smallest jump to NXT.
            DCR C                       ; Decrement count.
            JNZ         AGAIN           ; If counts not complete jump to
                                          AGAIN.
            MVI  A,     FF H            ; Store FF H to the accumulator.
            MOV  M,     A               ; Put FF H to the location at which
                                          the smallest number was found.
            RET                         ; Go back to main program.
```

6.6 PROGRAMS ON MULTIPLICATION

There are two methods for multiplication of two numbers.

1. Repetitive Addition Method
2. Shift and Add Method

Repetitive Addition Method

In this method multiplicand is added by the multiplier times.

For example, we wish to multiply 8 and 4 numbers. So add 08 to 00 for four times. i.e.

```
     0 0
  +  08          1 time
    ----------------------
     08
  +  08          2 times
    ----------------------
     16
  +  08          3 times
    ----------------------
     24
  +  08          4 times
    ----------------------
     32          Answer
    ----------------------
```

Shift and Add Method

For example, we wish to multiply two decimal numbers 32 and 12 as:

```
     3 2     Multiplicand
  x  1 2     Multiplier
  ------------------
      6 4
    3 2      Shift by one digit
  ------------------
    3 8 4     Answer
  ------------------
```

In this example, first of all multiplicand 32 is multiplied by 2 and the result 64 is copied in the temporary register (or written on the scratch pad). Again 32 is multiplied by 1 and result 32 is shifted left by one digit to make it 32o. Now 64 and 320 are added and we get the answer 384.

Similar method used in binary multiplication. In binary multiplication, when a multiplicand is multiplied by 1, we get the product equal to multiplicand. When a multiplicand is multiplied by 0, we the product zero. For example 08 and 04 are multiplied as:

```
                  0 0 0 0 1 0 0 0
              X   0 0 0 0 0 1 0 0
           --------------------------
                  0 0 0 0 0 0 0 0
                0 0 0 0 0 0 0 0
              0 0 0 0 1 0 0 0
            0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0
        0 0 0 0 0 0 0 0
      0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0
  ----------------------------------
   0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
  ----------------------------------
```

The answer is 0020 H (32_{10}). It may be noted that the answer is not in BCD.
This method may further be illustrated using the following flow chart.

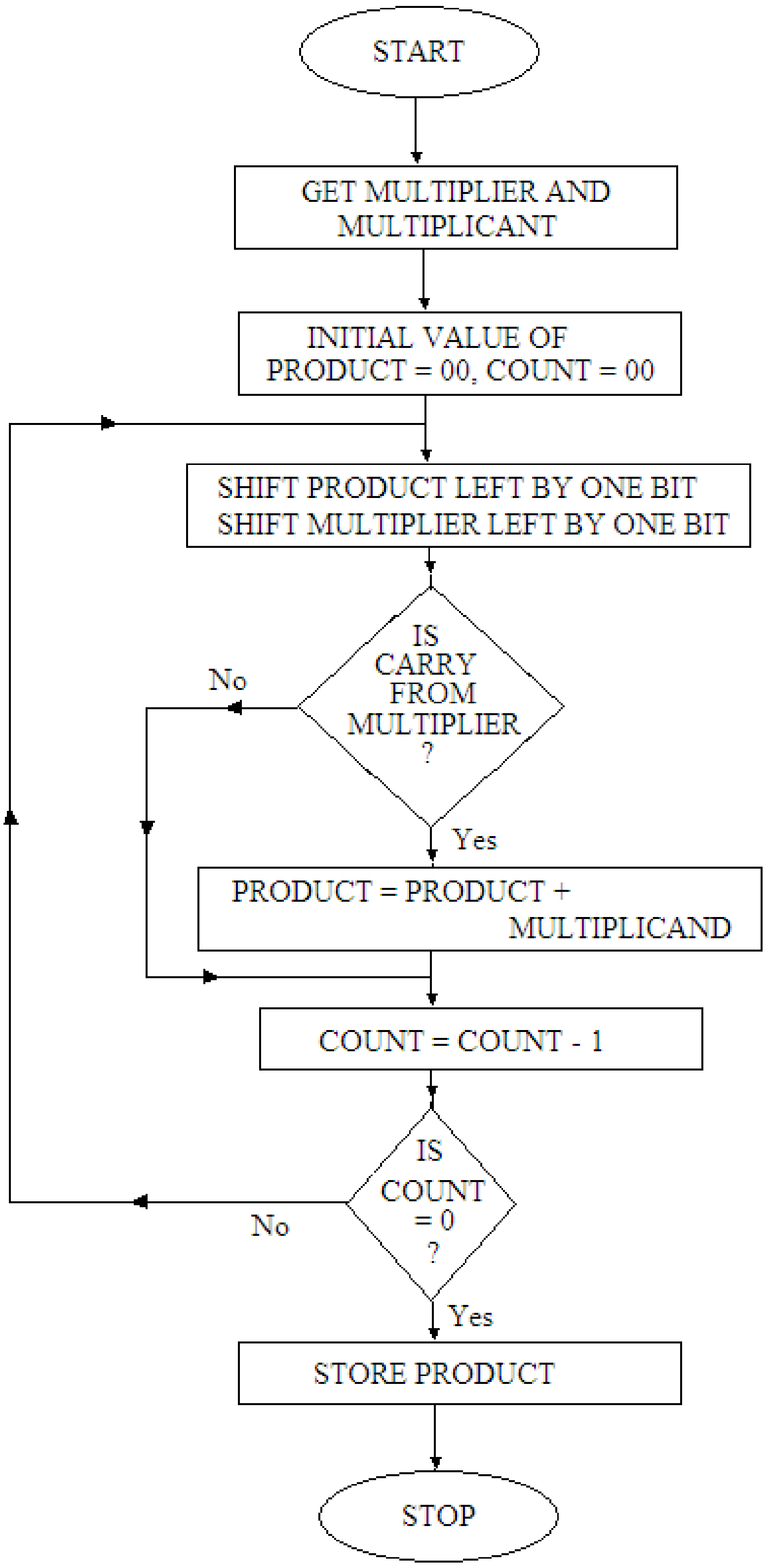
START
GET MULTIPLIER AND MULTIPLICANT
INITIAL VALUE OF PRODUCT = 00, COUNT = 00
SHIFT PRODUCT LEFT BY ONE BIT
SHIFT MULTIPLIER LEFT BY ONE BIT
IS CARRY FROM MULTIPLIER ?
No
Yes
PRODUCT = PRODUCT + MULTIPLICAND
COUNT = COUNT - 1
IS COUNT = 0 ?
No
Yes
STORE PRODUCT
STOP

Fig. 6.1

Example 6.38. *Write an ALP for 8085 to multiply any number (stored in memory location 2101 H) by 2 and store the result in 2102 H memory location.*

Solution.

Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2101 H	; Get the H-L pair with 2101 H.
	MOV A,	M	; Move the number in accumulator.
	STC		; Set the carry flag.
	CMC		; Complement the carry (it clear the carry.
	RAL		; Rotate accumulator with carry i.e. it multiplies the accumulator content by two.
	INX H		; Increment the H-L pair.
	MOV M,	A	; Store the result in 2102 H memory location.
	HLT		; Stop processing.

Example 6.39. *Write an ALP for 8085 to multiply two numbers using repetitive addition method. The two numbers are in memory locations 2101 H and 2102 H. Store the result in 2103 H and 2104 H.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI H,	2101 H	; Get H-L pair with 2101 H.
	MOV B,	M	; Get first number in B-register.
	INX H		; Increment H-L pair.
	MOV A,	M	; Get the second number in accumulator.
	CMP B		; Compare the two numbers.
	JNC	NXT	; If A < B then use A as counter.
	MVI D,	00 H	; Move 00 to D register.
	MOV E,	B	; Move first number to E register.
	JMP	NXT1	; Jump to NXT1.
NXT	MVI D,	00 H	; If A>B, so B should be used as counter.
	MOV E,	A	; Store accumulator content in E-register.
	MOV A,	B	; Store B-register content in accumulator.
NXT1	LXI H,	0000 H	; Put 00 to H and L registers.
	ORA A		; ORing of A with A to find if A = 00 H.
	JZ	END	; If zero jump to End.
	DAD D		; Add the contents of H-L pair with D-E pair and the result is in H-L pair.

```
              DCR  A                     ;  Decrement count.
              JNZ         LOOP           ;  If not zero jump to LOOP.
END           SHLD        2103 H         ;  Store the result.
              HLT                        ; Stop processing.
```

Example 6.40. *Write an ALP for 8085 to multiply two numbers using shift and add method. The two numbers are in memory locations 2101 H and 2102 H. Store the result in 2103 H and 2104 H.*

Solution.

Main Program:

```
Label         Mnemonics   Operand        Comments
              LXI  H,     2101 H         ;  Get H-L pair with 2101 H.
              MOV  E,     M              ;  Get first number in E-register.
              MVI  D,     00 H           ;  Extend to 16 bits.
              INX  H                     ;  Increment H-L pair.
              MOV  A,     M              ;  Get the multiplier in accumulator.
              LXI  H,     0000 H         ;  Put 00 to H and L registers.
              MVI  B,     08 H           ;  Get count = 08.
MULT          DAD  H                     ; Multiplicand = 2 x Multiplicand..
              RAL                        ;  Rotate accumulator left to find if
                                            most significant bit is 1.
              JNC         NXT            ;  If no carry jump to NXT.
              DAD  D                     ;  Get   Product   =   product   +
                                            multiplicand.
NXT           DCR  B                     ;  Decrement B.
              JNZ         MULT           ;  If not zero jump to MULT.
              SHLD        2103 H         ;  Store  the result in 2103 H and
                                            2104 H .
              HLT                        ; Stop processing.
```

6.7 PROGRAM ON 8-BIT DIVISION

To divide 16 bit dividend with 8 bit divisor, the divisor is subtracted from the 8 most significant bits of the dividend. If there is no borrow, the bit of the quotient is set to 1, otherwise 0. The dividend is then shifted left by one bit before each trial of subtraction. The dividend and quotient share a register pair. Due to shift of dividend one bit of the register falls vacant in each subtraction. The quotient is stored in vacant bit positions. It is illustrated by the flow chart shown in figure 6.2.

If the dividend of the division is an 8-bit number, then it is extended to a 16-bit number by placing zeros in the MSBs positions.

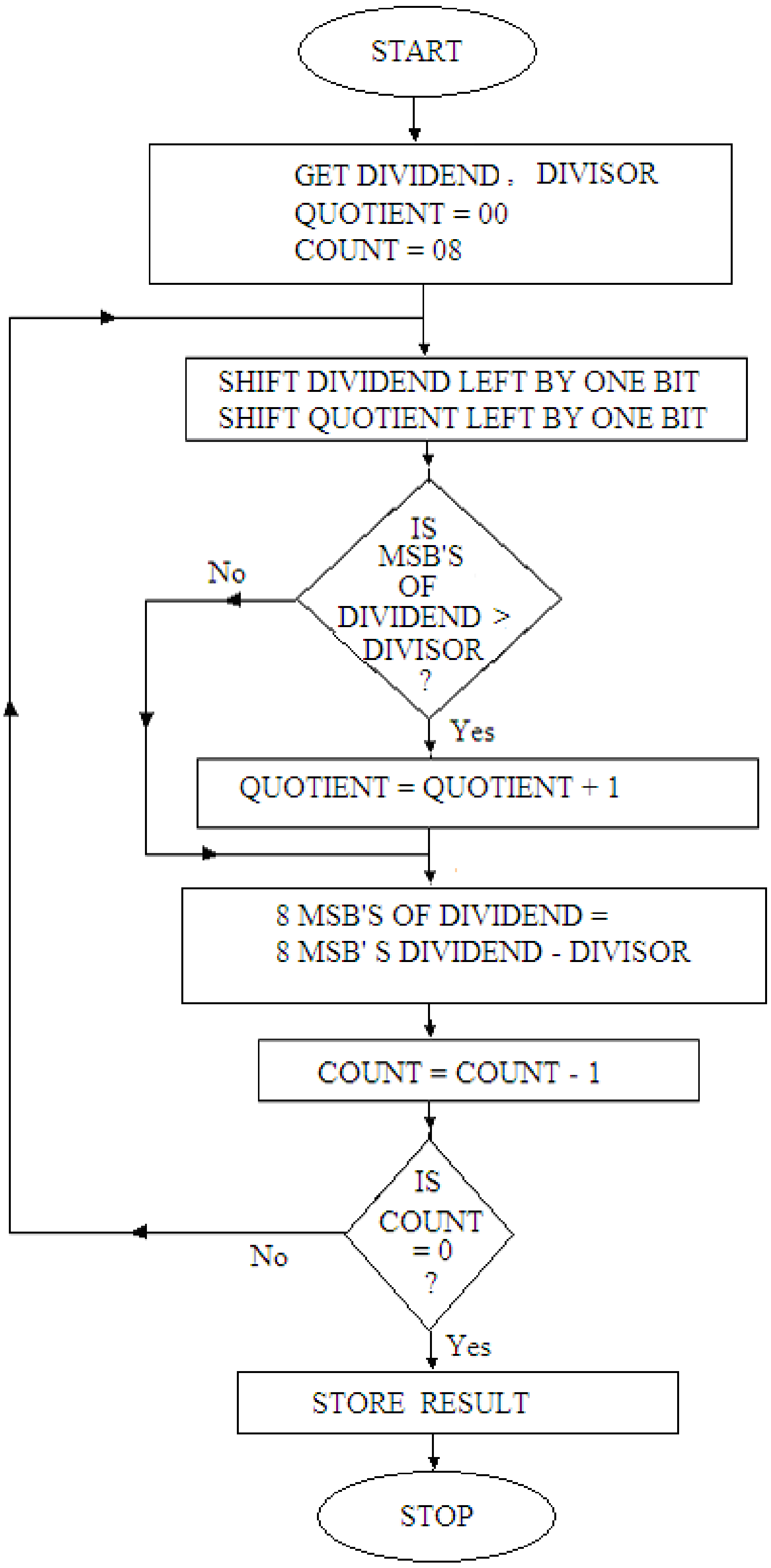
START
GET DIVIDEND , DIVISOR
QUOTIENT = 00
COUNT = 08
SHIFT DIVIDEND LEFT BY ONE BIT
SHIFT QUOTIENT LEFT BY ONE BIT
IS MSB'S OF DIVIDEND > DIVISOR ?
No
Yes
QUOTIENT = QUOTIENT + 1
8 MSB'S OF DIVIDEND =
8 MSB' S DIVIDEND - DIVISOR
COUNT = COUNT - 1
IS COUNT = 0 ?
No
Yes
STORE RESULT
STOP

Fig. 6.2

Example 6.41. *Write an ALP for 8085 to divide a 16-bit number by an 8-bit number. The dividend bytes are stored in memory locations 2101 H and 2102 H. (LS byte in 2101 H and MS byte in 2102 H). The divisor is stored in memory location 2103 H. The quotient is to be stored in the memory location 2104 H and the remainder is to be stored in the memory location 2105 H.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LHLD	2101 H	; Get 16-bit dividend in H-L pair.
	LDA	2103 H	; Get divisor in accumulator.
	MVI C,	08 H	; Get count 08 in C-register.
	MVI D,	00 H	; Quotient = 00.
	MOV B,	A	; Keep the divisor in B-register also.
LOOP	DAD H		; Shift dividend left by one bit.
	MOV A,	D	; Keep the quotient in accumulator.
	ADD A		; Shift quotient left by one bit.
	MOV D,	A	; Store the quotient in D-register.
	SUB B		; Perform trial subtraction.
	JC	NXT	; If carry then dividend = previous dividend.
	MOV H,	A	; Put most significant bits of dividend in register H.
	INR D		; Increment quotient by 1.
NXT	DCR C		; Decrement C.
	JNZ	LOOP	; If not zero jump to LOOP.
	MOV L.	D	; Store the quotient in L-register.
	SHLD	2104 H	; Store the result in 2104 H and 2105 H .
	HLT		; Stop processing.

6.8 MISCELLANEOUS PROGRAMS

Example 6.42. *Write an ALP for 8085 to find the square of a given number stored in memory location 2101 H and the result is to be stored in memory location 2102 H. Use Look-up table starting at 2200 H.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LXI D,	2200 H	; Get D-E pair with 2200 H (Starting address of the look-up table).
	LDA	2101 H	; Get the number in accumulator.
	MOV L,	A	: Load the number to L-register.
	MVI H,	00 H	; Get 00 H to H-register.
	DAD D		; Get effective address in H-L register pair.
	MOV A,	M	; Get the result in accumulator.

	STA	2102 H	; Store the result in the required memory location.
	HLT		; Stop processing.

Look-up table:

<u>Location</u>	<u>Data</u>
2200 H	00 H
2201 H	01 H
2202 H	04 H
2203 H	09 H
2204 H	10 H
2205 H	19 H
2206 H	24 H
2207 H	31 H
2208 H	40 H
2209 H	51 H
220A H	64 H
220B H	79 H
220C H	90 H
220D H	A9 H
220E H	C4 H
220F H	E1 H

Example 6.43. *Write an ALP for 8085 to find the square of a given number stored in memory location 2101 H and the result is to be stored in memory location 2102 H. Use the addition of successive odd integers.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LDA	2101 H	; Get the number in accumulator.
	MVI L,	00H	: Load 00 H to L-register.
	ORA L		; Find if number is zero.
	JZ	END	; If number is zero then no need to find the square and jump to END.
	MOV C,	A	; Move the number to C-register.
	MOV A,	L	; Move the content of L-reg to accumulator so that square is zero.
	MVI B,	01 H	; Move 01 to B-register (first odd number).
LOOP	ADD B		; Add next odd number.
	INR B		; Find next odd number to add
	INR B		; 02.
	DCR C		; Decrement C-register contents.
	JNZ	LOOP	; Continue if not zero.
END	STA	2102 H	; Store the result in the memory location 2102 H.

	HLT		; Stop processing.

Example 6.44. *Write an ALP for 8085 to find the square root of a given number (perfect positive square) stored in memory location 2101 H and the result is to be stored in memory location 2102 H. Use the subtraction of successive odd integers.*

Solution.

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	LDA	2101 H	; Get the number in accumulator.
	ORA A		; Find if number is zero.
	JZ	END	; If number is zero then no need to find the square root and jump to END.
	MVI B,	01 H	; Move 01 to B-register (first odd number).
	MVI C,	01 H	; Initial value of square root as one.
LOOP	SUB B		; Subtract next odd number.
	JZ	END	; If number is zero then jump to END.
	INR B		; Find next odd number to add
	INR B		; 02.
	INR C		; Decrement square root value by 1.
	JNC	LOOP	; Continue if no carry.
END	MOV A,	C	; Store the result in accumulator.
	STA	2102 H	; Store the result in the memory location 2102 H.
	HLT		; Stop processing.

Example 6.45. *Write an ALP for 8085 to find the Fibonacci series. Sixteen terms of this series are to be store in the memory locations starting at 2101 H. Let 00 H and 01 H data are stored in memory locations 2101 H and 2102 H respectively before the execution of the program.*

Solution. The terms of Fibonacci series are given as:

1, 1, 2, 3, 5, 8, 13, 21,. . . . .

This sequence is defined by assuming first two terms have the same value 1, and each term afterwards is the sum of the previous two terms, that is,

1 + 1 = 2, 1 + 2 = 3, 2 + 3 = 5, 3 + 5 = 8, and so on.

The required terms of the

Main Program:

Label	**Mnemonics**	**Operand**	**Comments**
	MVI C,	10 H	; Decimal number 16 (10 H) is stored in C-register which will be used as counter.
	LXI H,	2100 H	; Get H-L pair with 2100 H.
	MOV A,	M	; Move first data to accumulator.
	INX H		; Point to next data.
LOOP	ADD M		; Get next term of the series.

```
INX  H                      ; Point to the next memory location
                              for storing the next term.
MOV  M,      A              ; Store the next term.
DCX  H                      ; Get previous term of the Fibonacci
                              series.
MOV  A,      M              ; Get this term in accumulator.
INX  H                      ; Point to the next memory location.
DCR  C                      ; Decrement count.
JNZ          LOOP           ; If not zero jump to LOOP.
HLT                         ; Stop processing.
```

Data stored before the execution of the program.

```
2100 H       00 H
2101 H       01 H
```

PROBLEMS

1. Write an assembly language program of 8085 to fill the RAM area from 2500 H to 25FF H with a byte 33 H.
2. Sixteen bytes of data are stored in memory locations 2001 H to 2010 H. Write an assembly language program of 8085 to transfer this block of data to 2006 H to 2015 H.
   (Hint: In this case data will be transferred in the reverse order otherwise some of the data will coincide.)
3. Write a program (WAP) in assembly language of 8085 to compare two hexadecimal numbers, stored in 2001 H and 2002 H memory locations, for equality. If the numbers are equal, the number itself will be stored in 2003 H memory location, else 00 H will be stored in the memory location 2003 H.
4. WAP in assembly language of 8085 to test 3rd bit (D_3 bit) of a byte stored at 2000 H memory location. If the bit D3 is zero store 00 at 2101 H else store the same number at 2101 H.
5. Write an assembly language program (ALP) of 8085 to find the logical AND and Logical OR of 26 H and 39 H. Store the result in 2500 H and 2501 H.
6. Write an ALP of 8085 to load EE H to the memory locations starting at 2501 H. The length of data bytes is given in 2500 H memory location.
7. Write an ALP for 8085 to add two 8-bit numbers stored in memory locations 2101 H and 2102 H. The result should be stored in 2103 H and the carry if any should be stored in 2104 H.
8. Write an assembly language program for 8085 to add two 8 bit numbers stored in memory locations 2101 H and 2102 H. The answer is required in decimal form and it should be stored in 2103 H and the carry if any should be stored in 2104 H.
9. Write an ALP for 8085 to add 833 and 776 decimal numbers. The result should be stored in 2101 H to 2103 H memory locations. The memory location 2103 H should store the carry bit if any.
   (Hint: First convert the decimal numbers 833 and 776 to hexadecimal numbers.)
10. Write an ALP for 8085 to subtract an 8 bit number (stored in 2101 H memory location) from another 8 bit number (stored in 2102 H memory location). The

answer should be stored in 2103 H and the borrow bit if any should be stored in 2104 H.

11. Repeat the problem 10 to get the result in decimal form.
12. Write an ALP for 8085 to find smaller of two numbers stored in 2301 H and 2302 H. Store the smaller number in memory location 2303 H.
13. Write an ALP for 8085 to find the smallest of the three numbers stored in memory locations starting at 2301 H. Store he smallest number in 2304 H.
14. Write an ALP for 8085 to find the smallest number among a series of 16 numbers stored in the memory locations starting at 2301 H. The number 16 (10 H) is stored in 2300 H. The smallest number should be should be stored in 2401 H.
15. Write an ALP to find 13 terms of Fibonacci series. The terms of the series are to be stored in the memory locations starting at 2501 H.
16. Write an ALP for 8085 to shift an 8-bit number left by two bits. The number is stored in memory location 2501 H and the result is to be stored in 2502 H memory location.
    (Hint: Keep the number in accumulator and use ADD A instruction twice.)
17. Write an ALP for 8085 to shift a 16-bit number left by two bits. The number is stored in memory locations 2501 H and 2502 H. The result is to be stored in 2503 H and 2504 H memory locations.