UNIT 4

Chapter 1

CODE CONVERSION, BCD ARITHMETIC AND 16-BIT OPERATIONS

1.1 BCD to Binary conversion

Problem statement:
A BCD number between 0 and 99 is stored in an R/W memory location called Input Buffer (INBUF). Write a main program and a conversion subroutine (BCDBIN) to convert the BCD number into its equivalent binary number. Store the result in a memory location defined as the Output Buffer (OUTBUF).

```
START:  LXI SP, STACK
        LXI H, INBUF
        LXI B, OUTBUF
        MOV A, M
        CALL BCDBIN
        STAX B
        HLT

BCDBIN: PUSH B
        MOV B, A
        ANI 0FH
        MOV C, A
        MOV A, B
        ANI F0H
        JZ BCD1
        RRC
        RRC
        RRC
        RRC
        MOV D, A
        XRA A
        MVI E, 0AH
SUM:    ADD E
        DCR D
        JNZ SUM
BCD1:   ADD C
        POP B
        RET
```

1.2 Binary to BCD conversion

Problem statement:

- A binary number is stored in memory location BINBYT. Convert the number into BCD and store each BCD as two unpacked BCD digits in the Output Buffer (OUTBUF). To perform this task, write a main program and two subroutines: one to supply the powers of ten (PWRTEN), and the other to perform the conversion (BINBCD).

```
START:  LXI SP, STACK
        LXI H, BINBYT
        MOV A, M
        CALL PWRTEN
        HLT

PWRTEN: LXI H, OUTBUF
        MVI B,64H
        CALL BINBCD
        MVI B, 0AH
        CALL BINBCD
        MOV M, A
        RET

BINBCD: MVI M, FFH
NXTBUF: INR M
        SUB B
        JNC NXTBUF
        ADD B
        INX H
        RET
```

1.3 BCD to Seven-Segment LED conversion

Problem statement:

- A set of three packed BCD numbers (six digits) representing time and temperature are stored in memory locations starting at XX50H. The seven-segment codes of the digits 0 to 9 for a common-cathode LED are stored in memory locations starting at XX70H, and the Output-Buffer (OUTBUF) memory is reserved at XX90H. Write a main program and two subroutines, called UNPAK and LEDCOD, to unpack the BCD numbers and select an appropriate seven-segment code for each digit. The codes should be stored in the Output-Buffer memory.

```
START:  LXI SP, STACK
        LXI H, XX50H
        MVI D, 03H
        CALL UNPAK
        HLT

UNPAK:  LXI B, OUTBUF
NXTBCD: MOV A, M
        ANI F0H
        RRC
        RRC
        RRC
        RRC
        CALL LEDCOD
        INX B
        MOV A, M
        ANI 0FH
        CALL LEDCOD
        INX B
        INX H
        DCR D
        JNZ NXTBCD
        RET

LEDCOD: PUSH H
        LXI H, XX70H
        ADD L
        MOV L, A
        MOV A,M
        STAX B
        POP H
        RET

XX70H:  3F
XX71H:  06
XX72H:  5B
XX73H:  4F
XX74H:  66
XX75H:  6D
XX76H:  7D
XX77H:  07
XX78H:  7F
XX79H:  6F
```

1.4 Binary to ASCII conversion

Problem statement:

- An 8-bit binary number (eg 9FH) is stored in memory location XX50H. Write a program to
  - Transfer the byte to accumulator
  - Separate the two nibbles (as 09 and 0F)
  - Call the subroutine to convert each nibble into ASCII Hex code
  - Store the codes in memory locations XX60H and XX61H
- Write a subroutine to convert a binary digit (0 to F) into ASCII Hex code.

```
START: LXI SP, STACK
       LXI H, XX50H
       LXI D, XX60H
       MOV A, M
       MOV B, A
       RRC
       RRC
       RRC
       RRC
       CALL ASCII
       STAX D
       INX D
       MOV A, B
       CALL ASCII
       STAX D
       HLT

ASCII: ANI 0FH
       CPI 0AH
       JC CODE
       ADI 07H
CODE:  ADI 30H
       RET
```

1.5 ASCII to Binary conversion

Problem statement:

- Write a subroutine to convert an ASCII Hex number into its binary equivalent. A calling program places the ASCII number in the accumulator and the subroutine should pass the conversion back to the accumulator.

```
ASCBIN:   SUI 30H
          CPI 0AH
          RC
          SUI 07H
          RET
```

1.6 BCD Addition

Problem statement:

- A set of ten packed BCD numbers is stored in the memory location starting at XX50H. Write a main program with a subroutine (BCDADD) to add these numbers in BCD. If a carry is generated, save it in the register B, and adjust it for BCD. The final sum will be less than $(9999)_{BCD}$. Write a second subroutine (UNPAK) to unpack the BCD sum stored in registers A and B, and store them in the output-buffer memory starting at XX60H. The most significant digit (BCD_4) should be stored at XX60H and the least significant digit (BCD_1) at XX63H

```
START:    LXI SP, STACK
          LXI H, XX50H
          MVI C, COUNT
          XRA A
          MOV B, A
NXTBCD:   CALL BCDADD
          INX H
          DCR C
          JNZ NXTBCD
          LXI H, XX63H
          CALL UNPAK
          MOV A, B
          CALL UNPAK
          HLT
```

```
BCDADD:  ADD M
         DAA
         RNC
         MOV D, A
         MOV A, B
         ADI 01H
         DAA
         MOV B, A
         MOV A, D
         RET

UNPAK:   MOV D,A
         ANI 0FH
         MOV M, A
         DCX H
         MOV A, D
         ANI F0H
         RRC
         RRC
         RRC
         RRC
         MOV M, A
         DCX H
         RET
```

1.7 BCD Subtraction

Problem statement:

- Write a subroutine to subtract one packed BCD number from another BCD number. The minuend is placed in register B, and the subtrahend is placed is register C by the calling program. Return the answer in accumulator.

```
SUBBCD:  MVI A, 99H
         SUB C
         INR A
         ADD B
         DAA
         RET
```

1.8 INTRODUCTION TO ADVANCED INSTRUCTIONS AND APPLICATIONS

1.8.1 16-Bit Data Transfer and Data Exchange Group

LHLD, SHLD, XCHG: **Refer ppt and Excel file of data transfer instructions mailed to your class.**

1.8.2 Arithmetic Group

Addition with Carry (ADC R, ADC M, ACI data), Subtraction with Borrow (SBB R, SBB M), SBI data), Double Register Add (DAD R_P/ DAD SP): **Refer ppt and Excel file of Arithmetic instructions mailed to your class.**

1.8.3 Instructions related with Stack Pointer and Program Counter

Exchange the top of the stack with H and L (XTHL), Load Stack Pointer with content of HL pair (SPHL), Load Program Counter with content of HL pair (PCHL): **Refer ppt and Excel file of data transfer instructions mailed to your class.**

PROGRAMS:

1) Registers BC contain 8538H and registers DE contain 62A5H. Write instructions to subtract the contents of DE from contents of BC, and place the result in BC.

```
MOV  A, C
SUB  E
MOV  C, A
MOV  A, B
SBB  D
MOV  B, A
HLT
```

(B) 85	38 (C)
-	-
(D) 62	A5 (E)
- 1	93 with borrow 1
(B) 22	93 (C)

2) Registers BC contain 2793H and registers DE contain 3182H. Write instructions to add these two 16-bit numbers and place the result in memory locations 2050H and 2051H.

```
MOV  A, C
ADD  E
MOV  L, A
MOV  A, B
ADD  D
```

OR

```
XCHG
DAD  B
SHLD 2050H
HLT
```

```
MOV  H, A
SHLD 2050H
HLT
```

3) Write instructions to display the contents of the Stack Pointer at Output ports.

```
LXI  H, 0000H
DAD SP
MOV  A, H
OUT  PORT1
MOV  A, L
OUT  PORT2
HLT
```

4) Write subroutine to set the Zero flag and check whether the instruction JZ functions properly, without modifying any register contents other than flags.

```
CHECK:   PUSH  H
         MVI    L, FFH
         PUSH  PSW
         XTHL
         POP    PSW
         JZ     NOEROR
         JMP    ERROR
NOEROR:  POP    H
         RET
```