P!!

EXAMPLE PROGRAMS 8085
Statement: Write an assembly language program to multiply 2 BCD numbers

MVI C, Multiplier : Load BCD multiplier
MVI B, 00 : Initialize counter
LXI H, 0000H : Result = 0000
MVI E, multiplicand : Load multiplicand
MVI D, 00H : Extend to 16-bits
BACK: DAD D : Result = Result + Multiplicand
MOV A, L : Get the lower byte of the result
ADI 00H
DAA : Adjust the lower byte of result to BCD.
MOV L, A : Store the lower byte of result
MOV A, H : Get the higher byte of the result
ACI 00H
DAA : Adjust the higher byte of the result to BCD
MOV L, A : Store the higher byte of the result.
MOV A, B : [Increment
ADI 01H : counter
DAA : adjust it to BCD and
MOV B, A : store it]
CMP C : Compare if count = multiplier
JNZ BACK : if not equal repeat
HLT : Stop

Statement: Subtract the BCD number stored in E register from the number stored in the D register.

Source Program:

MVI A, 99H
SUB E : Find the 99's complement of subtrahend
INR A : Find 100's complement of subtrahend
ADD D : Add minuend to 100's complement of subtrahend
DAA : Adjust for BCD
HLT : Terminate program execution
Note: When two BCD numbers are subtracted, we can use DAA instruction for adjusting the result to BCD. Therefore, the subtraction of BCD number is carried out 10's complement or 100's complement.

The 10's complement of a decimal number is equal to the 99's complement plus 1.

The 99's complement of a number can be found by subtracting the number from 99.

The steps for finding 100's complement BCD subtraction are:

Find the 100's complement of subtrahend
Add two numbers using BCD addition

Statement: Add two 4 digit BCD numbers in HL and DE register pairs and store result in memory locations, 2300H and 2301H. Ignore carry after 16 bit.

Example Problem:

(HL) = 3629
(DE) = 4738

Step 1: 29 + 38 = 61 and auxiliary carry flag = 1
:. add 06
61 + 06 = 67

Step 2: 36 + 47 + 0 (carry of LSB) = 7D

Lower nibble of addition is greater than 9, so add 6.

7D + 06 = 83

Result = 8367

Source program:

MOV A, L : Get lower 2 digits of no. 1
ADD E : Add two lower digits
DAA : Adjust result to valid BCD
STA 2300H : Store partial result
MOV A, H : Get most significant 2 digits of number
ADC D : Add two most significant digits
DAA : Adjust result to valid BCD
STA 2301H : Store partial result
HLT : Terminate program execution.
Statement: Pack the two unpacked BCD numbers stored in memory locations 4200H and 4201H and store result in memory location 4300H. Assume the least significant digit is stored at 4200H.
Sample problem:
\[
\begin{align*}
(4200H) &= 04 \\
(4201H) &= 09 \\
\text{Result} &= (4300H) = 94
\end{align*}
\]

Source program

LDA 4201H : Get the Most significant BCD digit
RLC
RLC
RLC
RLC : Adjust the position of the second digit (09 is changed to 90)
ANI FOH : Make least significant BCD digit zero
MOV C, A : store the partial result
LDA 4200H : Get the lower BCD digit
ADD C : Add lower BCD digit
STA 4300H : Store the result
HLT : Terminate program execution

NOTE:
BCD NO.: The numbers "0 to 9" are called BCD (Binary Coded Decimal).

Statement: Two digit BCD number is stored in memory location 4200H. Unpack the BCD number and store the two digits in memory locations 4300H and 4301H such that memory location 4300H will have lower BCD digit.
Sample problem
\[
\begin{align*}
(4200H) &= 58 \\
\text{Result} &= (4300H) = 08 \text{ and } (4301H) = 05
\end{align*}
\]

Source program

LDA 4200H : Get the packed BCD number
ANI FOH : Mask lower nibble
RRC
RRC
RRC
RRC : Adjust higher BCD digit as a lower digit
STA 4301H : Store the partial result
LDA 4200H: Get the original BCD number
ANI 0FH: Mask higher nibble
STA 4201H: Store the result
HLT: Terminate program execution

Statement: Assume the DAA instruction is not present. Write a sub routine which will perform the same task as DAA.

Sample Problem:

Execution of DAA instruction:
1. If the value of the low order four bits (03-00) in the accumulator is greater than 9 or if auxiliary carry flag is set, the instruction adds 6 '06) to the low-order four bits.
2. If the value of the high-order four bits (07-04) in the accumulator is greater than 9 or if carry flag is set, the instruction adds 6(06) to the high-order four bits.

Source Program:

LXI SP, 27FFH: Initialize stack pointer
MOV E, A: Store the contents of accumulator
ANI 0FH: Mask upper nibble
CPI 0A H: Check if number is greater than 9
JC SKIP: if no go to skip
MOV A, E: Get the number
ADI 06H: Add 6 in the number
JMP SECOND: Go for second check
SKIP: PUSH PSW: Store accumulator and flag contents in stack
POP B: Get the contents of accumulator in B register and flag register contents in C register
MOV A, C: Get flag register contents in accumulator
ANI 10H: Check for bit 4
JZ SECOND: if zero, go for second check
MOV A, E: Get the number
ADI 06: Add 6 in the number
SECOND: MOV E, A: Store the contents of accumulator
ANI FOH: Mask lower nibble
RRC
RRC
RRC
RRC: Rotate number 4 bit right
CPI 0AH: Check if number is greater than 9
JC SKIPI : if no go to skip 1
MOV A, E : Get the number
ADI 60 H : Add 60 H in the number
JMP LAST : Go to last
SKIP1: JNC LAST : if carry flag = 0 go to last
MOV A, E : Get the number
ADI 60 H : Add 60 H in the number
LAST: HLT

Note: To check auxiliary carry flag it is necessary to get the flag register contents in one of the registers and then we can check the auxiliary carry flag by checking bit 4 of that register. To get the flag register contents in any general purpose register we require stack operation and therefore stack pointer is initialized at the beginning of the source program.

http://www.8085projects.info/8085-free-programs.html