Executing the "MUL" instruction in MCS-51
Executing the “DIV” instruction in MCS-51
Write assembly code based on MCS-51 to perform the arithmetic operations for the following block diagram.

```assembly
INCLUDE 8051.mc

MOV P2, #FFH ; make P2 as input port
MOV A, P2 ; Read p2 to A
CLR C ; clear carry
SUBB A, #05H ; Subtract 5 from A
MOV B, #12H ; load 12h to reg-B
DIV AB ; divide A by reg-B
MOV P0, B ; Write Remainder to P0
MOV P1, A ; Write Quotient to P1
```
Write assembly code based on MCS-51 to carry out the arithmetic operations for the following block diagram.

```
INCLUDE 8051.mc
MOV P2, #FFH ; P2 as input port
MOV A, P2 ; Read p2 to A
ADD A, #03H ; Add 3H to A
MOV B, #A5H ; load A5H to reg-B
MUL AB ; multiply A with reg-B
MOV P0, B ; upper byte of multiplication
MOV P1, A ; lower order of multiplication
```
Write assembly code based on MCS-51 to achieve the operations for the following block diagram.

```
ORG 00H
MOV P1,#FFH
MOV P3,#FFH
MOV A, P1
ANL A, R1
MOV B, P3
ORL B, #33H
MUL AB
ADD A, #5H
MOV P0, A
MOV A,B
CLR C
SUBB A, R2
MOV P2, A
```
Write subprogram to add two 16-bit numbers for the following diagram (keep the data on “A” without change at the end of the program).

ORG 0
MOV P1,#0FFH
MOV P2,#0FFH
Push A ;Store byte
+++++
MOV A,P2
ADD A,#55H
MOV P0,A ;--
MOV A,P1
ADDC A,#22H
MOV P3,A
Pop A ;retrieve byte
SJMP$
Decimal Adjust (DA A) Instruction

- If $A_{0\rightarrow3} > 9$ or $AC = 1$ then $A_{0\rightarrow3} + 6h$
- If $A_{4\rightarrow7} > 9$ or $CY = 1$ then $A_{4\rightarrow7} + 6h$

If A has value DCh get the BCD value answer

Ch +6h = 12h $\rightarrow$ carry =1 and $A_{0\rightarrow3} = 2$
(Dh+ carry =1)+6=14h $\rightarrow$ CY =1 and $A_{4\rightarrow7} = 4$
DA(DCh)= 42
A has value 9Ah get the BCD value
answer
Ah +6h = 10h \( \rightarrow \) carry =1 and \( A_{0\rightarrow 3} = 0 \)
(9h+ carry=1)+6=10h \( \rightarrow \) CY =1 and \( A_{4\rightarrow 7} = 0 \)
DA(A)= 00

A has value 12h and CY=AC=1 get the BCD value
answer
Ac=1 then
2h +6h = 08h \( \rightarrow \) carry =0 and \( A_{0\rightarrow 3} = 8 \)
CY=1 then
(1h+ carry=0)+6=07h \( \rightarrow \) CY =0 and \( A_{4\rightarrow 7} = 7 \)
DA(A)= 78
A digital circuit based on MCS-51 is designed for adding 2 BCD-numbers that received from P1 and P2.

- Write assembly program to display the summation of these 2 data on 7-segments via any available port(s).

- If the 2 BCD-numbers were 81h and 70h, what is the output BCD-number on its 7-segments.
The assembly code

```
INCLUDE 8051.mc

MOV P1, #FFH          ; P1 as input port
MOV P2, #FFH          ; P2 as input port

MOV A, P1             ; Read p1
MOV B, P2             ; Read p2

ADD A, B              ; B + A
DA A                  ; Decimal adjusting

MOV P0, A             ; out lower 2 digits
MOV P3.7, C           ; out upper digits
```
ACC. has value **F1h**, **AC=Cy= ‘0’**

answer

1h+0 = 01h  \(\rightarrow\) carry=0 and  \(A_{0\rightarrow3} = 1\)

(Fh+ carry=0)+6=15h  \(\rightarrow\) CY =1 and  \(A_{4\rightarrow7} = 5\)

DA(A)= **51**

The total value = **151**
The Branching instructions of MCS-51

Instructions that responsible for changing the content of PC to follow the subsequent instruction in the program memory.
The **unconditional branching instruction**

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Branching</th>
</tr>
</thead>
<tbody>
<tr>
<td>LJMP addr16</td>
<td>long jump</td>
<td>64 KB</td>
</tr>
<tr>
<td>AJMP addr11</td>
<td>absolute jump</td>
<td>2KB</td>
</tr>
<tr>
<td>SJMP rel</td>
<td>short jump</td>
<td>Relative to PC</td>
</tr>
<tr>
<td>LCALL addr16</td>
<td>long call</td>
<td>64 KB</td>
</tr>
<tr>
<td>ACALL addr11</td>
<td>absolute call</td>
<td>2KB</td>
</tr>
<tr>
<td>RET</td>
<td>Return from the calling</td>
<td>64 KB</td>
</tr>
<tr>
<td>RETI</td>
<td>Return from the interrupt service</td>
<td>64 KB</td>
</tr>
<tr>
<td>Instruction</td>
<td>Condition</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>JZ rel</td>
<td>Jump if Accumulator = 0h</td>
<td></td>
</tr>
<tr>
<td>JNZ rel</td>
<td>Jump if Accumulator ≠ 0h</td>
<td></td>
</tr>
<tr>
<td>JC rel</td>
<td>Jump if carry flag = 1</td>
<td></td>
</tr>
<tr>
<td>JNC rel</td>
<td>Jump if carry flag ≠ 1</td>
<td></td>
</tr>
<tr>
<td>JB Bit, rel</td>
<td>Jump if addressable bit = 1</td>
<td></td>
</tr>
<tr>
<td>JNB Bit, rel</td>
<td>Jump if addressable bit ≠ 1</td>
<td></td>
</tr>
<tr>
<td>JBC Bit, rel</td>
<td>Jump if addressable bit = 1 and clear it</td>
<td></td>
</tr>
<tr>
<td>DJNZ Rx, rel</td>
<td>Dec Rx and jump</td>
<td></td>
</tr>
<tr>
<td>DJNZ Rn, rel</td>
<td>Dec Rn and jump</td>
<td></td>
</tr>
<tr>
<td>CJNE Rn, #data, rel</td>
<td>Compare and jump if Rn ≠ data</td>
<td></td>
</tr>
<tr>
<td>CJNE @Ri, #data, rel</td>
<td>Compare and jump if @Ri ≠ data</td>
<td></td>
</tr>
<tr>
<td>CJNE A, #data, rel</td>
<td>Compare and jump if A ≠ data</td>
<td></td>
</tr>
<tr>
<td>CJNE A, Rx, rel</td>
<td>Compare and jump if A ≠ Rx</td>
<td></td>
</tr>
</tbody>
</table>
Calculate the real delay time for the attached nested loop, when $F_{\text{crystal}} = 12 \text{ MHz}$
✓ No. of machine cycles of wait1 = (1+1+2) \times 250 \\
= 1000

✓ No. of machine cycles of wait2 = (1+2+\text{wait1}) \times 251 \\
= 1003 \times 251 = 251753

✓ Delay Time = (1+2+251753) \times \frac{12}{120000000} \approx 0.25 \text{ sec}
Calculate the real run-time for the attached subprogram, when $F_{crystal} = 6 \text{ MHz}$
Write assembly code to send the stored data in flash memory “The Microcontroller 8051” Via P2

```
org 0
SJMP main
org 30h
main:
MOV    P2, #0 ; clear Port2
MOV    R0,#0
MOV    DPTR,#My_MSG ;load data pointer with MSG's address

Get_B
```
Explanation

DB "8051", 0

“0” has Machine code = ASCII = 30h

Machine code = 00h

JZ is affected with the machine code 00h