8051 Assembly Language Programming(ALP)

ADDITION OF TWO 8 bit Numbers

ADDRESS	LABEL	MNEMONICS
9100:		MOV A,#05
		MOV B,#03
		ADD A,B
		MOV DPTR,#9200
		MOVX @DPTR,A
	HERE	SJMP HERE

After execution: A=08

SUBTRACTION OF TWO 8 bit Numbers

ADDRESS	LABEL	MNEMONICS
9100:		CLR C
		MOV A,#05
		MOV B,#03
		SUBB A,B
		MOV DPTR,#9200
		MOVX @DPTR,A
	HERE	SJMP HERE

After execution: A=02

MULTIPLICATION OF TWO 8 bit Numbers

Address	Label	Mnemonics
9000	START	MOV A,#05
		MOV B,#03
		MUL AB
		MOV DPTR,#9200
		MOVX @ DPTR,A
		INC DPTR
		MOV A,B
		MOVX @DPTR,A
	HERE	SJMP HERE

After execution: **A=0F , B=00**

DIVISION OF TWO 8 bit Numbers

Address	Label	Mnemonics
9000	START	MOV A,#05
		MOV B,#03
		DIV AB
		MOV DPTR,#9200
		MOVX @ DPTR,A
		INC DPTR
		MOV A,B
		MOVX @DPTR,A
	HERE	SJMP HERE

After execution: **A=01 , B=02**

Average of N (N=5) 8 bit Numbers

```
        MOV 40H, #02H        store  1st number in location 40H
        MOV 41H, #04H
        MOV 42H, #06H
        MOV 43H, #08H
        MOV 44H, #01H
        MOV R0, #40H         store 1 st number address 40H in R0
        MOV R5, #05H         store the count {N=05} in R5
        MOV B,R5             store the count {N=05} in B
        CLR A                Clear Acc
LOOP:   ADD A,@R0
        INC R0
        DJNZ R5,LOOP
        DIV AB
        MOV 55H,A            Save the quotient in location 55H
HERE    SJMP  HERE
```

Answer: 02+04+06+08+01 = 21(decimal) = 15 (Hexa)

SUM = 15 H Average = 21(decimal) / 5 = 04 (remainder) , 01 (quotient)

55 | **01** | **quotient**

INSTRUCTION SET OF 8051

8051 Instruction Set

- The instructions are grouped into 5 groups
  - Arithmetic
  - Logic
  - Data Transfer
  - Boolean
  - Branching

1. Arithmetic Instructions

- **ADD A, source**

  A ← A + <operand>.

- **ADDC A, source**

  A ← A + <operand> + CY.

- **SUBB A, source**

  A ← A - <operand> - CY{borrow}.

- **INC**
  - Increment the operand by one. Ex: **INC DPTR**
- **DEC**
  - Decrement the operand by one. Ex: **DEC B**
- **MUL AB**

Multiplication	A*B	**Result**
8 byte * 8 byte		A=low byte, B=high byte

- **DIV AB**

Division	A/B	**Quotient**	**Remainder**
8 byte /8 byte		A	B

Multiplication of Numbers

MUL AB ; A × B, place 16-bit result in B and A

```
A=07 , B=02
MUL   AB          ;07 * 02 = 000E where B = 00 and A = 0E
```

Division of Numbers

DIV AB ; A / B , 8-bit Quotient result in A &
8-bit Remainder result in B

```
A=07 , B=02
DIV   AB          ;07 / 02 = Quotient 03(A)  Remainder  01 (B)
```

2. Logical instructions

- **ANL D,S**

  -Performs **logical AND** of destination & source

  - Eg: **ANL A,#0FH** **ANL A,R5**

- **ORL D,S**

  -Performs **logical OR** of destination & source

  - Eg: **ORL A,#28H** **ORL A,@R0**

- **XRL D,S**

  -Performs **logical XOR** of destination & source

  - Eg: **XRL A,#28H** **XRL A,@R0**

- **CPL A**

  -Compliment accumulator

  -gives 1's compliment of accumulator data

- **RL A**

  -Rotate data of accumulator towards left **without carry**

- **RLC A**

  - Rotate data of accumulator towards left **with carry**

- **RR A**

  -Rotate data of accumulator towards right **without carry**

- **RRC A**

  - Rotate data of accumulator towards right **with carry**

3. Data Transfer Instructions

MOV Instruction

- **MOV destination, source** ; copy source to destination.

- MOV A,#55H
  MOV R0,A

  MOV R1,A

  MOV R2,A

  MOV R3,#95H

  MOV A,R3

- **MOVX**
  - Data transfer between the accumulator and a byte from external data memory.
    - **MOVX A, @DPTR**
    - **MOVX @DPTR, A**

- **PUSH / POP**
  - Push and Pop a data byte onto the stack.

    - PUSH DPL
    - POP 40_H

- **XCH**
  - Exchange accumulator and a byte variable
    - XCH A, Rn
    - XCH A, direct
    - XCH A, @Ri

4.Boolean variable instructions

CLR:

- The operation **clears** the specified bit indicated in the instruction
- Ex: **CLR C** clear the carry

SETB:

- The operation **sets** the specified bit to **1**.

CPL:

- The operation **complements** the specified bit indicated in the instruction

- **ANL C,<Source-bit>**

  -Performs **AND** bit addressed with the **carry bit.**

  - Eg: **ANL C,P2.7 AND carry flag with bit 7 of P2**

- **ORL C,<Source-bit>**

  -Performs **OR** bit addressed with the **carry bit.**

  - Eg: **ORL C,P2.1 OR carry flag with bit 1 of P2**

- **XORL C,<Source-bit>**

-Performs **XOR** bit addressed with the **carry bit.**

- Eg: **XOL C,P2.1 OR carry flag with bit 1 of P2**

- MOV P2.3,C
- MOV C,P3.3
- MOV P2.0,C

5. Branching instructions

Jump Instructions

- **LJMP** (long jump):
  - Original 8051 has only **4KB on-chip ROM**
- **SJMP** (short jump):
  - 1-byte relative address: **-128 to +127**

Call Instructions

- LCALL (long call):
  - Target address within **64K-byte range**
- ACALL (absolute call):
  - Target address within **2K-byte range**

- 2 forms for the return instruction:
  - **Return from subroutine – RET**
  - **Return from ISR – RETI**

8051 conditional jump instructions

Instructions	Actions
JZ	Jump if A $= 0$
JNZ	Jump if A $\neq 0$
DJNZ	Decrement and Jump if A $\neq 0$
CJNE A,byte	Jump if A $\neq$ byte
CJNE reg,#data	Jump if byte $\neq$ #data
JC	Jump if CY $= 1$
JNC	Jump if CY $= 0$
JB	Jump if bit $= 1$
JNB	Jump if bit $= 0$
JBC	Jump if bit $= 1$ and clear bit