# UNIT V - 8051 Instruction Set and Programming

**Instruction format**

- Instruction format of 8051 consist of Opcode and Operand
- The Opcode represents the type of operation to be performed, and
- The Operand represents the data upon which operation is performed.
- Example: MOV A, Rn; copies the content of the register Rn of selected register bank to A.

**INSRUCTION SETS OF 8051:**

Instruction set of 8051is broadly classified as follows:

**1.** Data Transfer Instruction:

**2.** Instruction to Access External Data Memory

**3.** Rotate and Swap Instruction

**4.** Logical Instructions

i. Byte Level Logical Instruction

ii. Bit Level Logical Instruction

**5.** Arithmetic Instruction:

i. Incrementing and Decrementing.

ii. Addition

iii. Subtraction

iv. Multiplication and Division

**6.** Jump Instruction

**7.** Call and Subroutine Instruction

**8.** Push and Pop Instruction

**9.** Data Exchange Instruction

**1. Data Transfer Instruction:**

- Data transfer Instruction set is used to perform data transfer function.
- Data can be transferred from or to external RAM or within the internal Memory itself.

| S. No | Instruction | Addressing Modes | Operation |
|---|---|---|---|
| **1.** | MOV A, | Rn Register | Copies the content of the register Rn to A |
| **2.** | MOV A,Direct | Direct byte | Copies the content of the address specified to A |
| **3.** | MOV A,@Ri | Register indirect | Copy the content of address Ri to A |
| **4.** | MOV A,#data | Immediate | Load data given in the instruction to A |
| **5.** | MOV Rn,A | Register | Copies the content of A to Register Rn of the selected register bank. |
| **6.** | MOV direct, A | Direct Byte | Copies the content of A to the address specified within the instruction. |
| **7.** | MOV@Ri, A | Register indirect | Copies the content of A to the address Ri of the selected register bank. |
| **8.** | MOV DPTR, #data 16 | Immediate | Load data pointer with a 16-bit constant. |

**Table 1: Data Transfer Instruction**

**2. Instruction to Access External Data Memory:**

- These instructions are used perform the data accessing from external memory.

| S. No | Instruction | Addressing Modes | Operation |
|---|---|---|---|
| **1.** | MOVX A ,Ri | Register | Copy the content of external address in Ri to A. |
| **2.** | MOVX A,@DPTR | Register indirect | Copy the content of external memory in DPTR to A. |
| **3.** | MOVX @Ri, A | Register indirect | Copy data from A to the external address inRi. |
| **4.** | MOVX @ DPTR, A | Register indirect | Copy data from A to the external address in DPTR. |

**Table 2: Instruction to Access External Data Memory**

**3. Rotate and Swap Instruction:**

- The instructions are used to rotate the bit position of the accumulator.

| S. No | Instruction | Operation |
|---|---|---|
| **1.** | RLA | Rotate Accumulator left |
| **2.** | RRA | Rotate Accumulator right |
| **3.** | RLCA | Rotate Accumulator left through carry flag |
| **4.** | RRCA | Rotate Accumulator Right through carry flag |
| **5.** | SWAPA | Swap with in the Accumulator |

**Table 3: Rotate and Swap Instruction**

**4. Logical Instructions:**

- The logical instructions are used to perform AND, OR, and XOR operations.
- Types:

i. Byte Level Logical Instruction

ii. Bit Level Logical Instruction

| S. No | Instruction | Addressing Modes | Operation | Byte |
|---|---|---|---|---|
| **1.** | ANL A, Rn | Register | Content of register is logically ANDed with that of a Accumulator. | 1 |
| **2.** | ANL A,Direct | Direct byte | Content of register byte is logically ANDed with that of Accumulator | 1 |
| **3.** | ORL A,@Ri | Register indirect | The content of register is an address whose content is logically ORed with that of accumulator. | 1 |
| **4.** | CLR A | Implied | Clear the accumulator | 1 |
| **5.** | CPL A | Implied | Complement the content of accumulator | 1 |

**Table 4: Byte Level Logical Instruction**

| S. No | Instruction | Addressing Modes | Byte |
|---|---|---|---|
| **1.** | CLRC | Clear carry flag | 1 |
| **2.** | CPLC | Complement carry flag | 1 |
| **3.** | ANL C , Bit | AND direct bit to carry flag | 2 |
| **4.** | ORL C, Bit | Or direct to carry flag | 2 |
| **5.** | MOV C , Bit | Move direct bit to carry flag | 2 |

**Table 5 : Bite Level Logical Instruction**

**5. Arithmetic Instruction:**

- The instructions are used to perform the arithmetic operations.
- The common arithmetic operations such as addition,subtraction, multiplicationand division are as possible with the 8051.
- According to operation, the instructions are:

i. Incrementing and Decrementing.

ii. Addition

iii. Subtraction

iv. Multiplication and Division

**i. Incrementing and Decrementing Instructions:**

| S. No | Instruction | Addressing Modes | Operation |
|---|---|---|---|
| **1.** | INC A | Register | Increment the content of Accumulator. |
| **2.** | INC Rn | Direct byte | Increment the content of direct byte address. |
| **3.** | DEC @ Ri | Register indirect | Decrement indirect RAM |

**Table 6 : Incrementing and Decrementing Instructions**

| S. No | Instruction | Addressing Modes | Operation |
|---|---|---|---|
| **1.** | ADD A,direct | Direct byte | Add the content of direct byte to accumulator |
| **2.** | ADD A,@Ri | Register indirect | Add the content of address register to content of accumulator |
| **3.** | ADD A,#data | Immediate | Add the data in the instruction to the content of accumulator |
| **4.** | ADDC A,Rn | Register | Add the content of register to the accumulator with carry. |
| **5.** | SUBB A,Rn | Register | Subtract the content of register to the accumulator with borrow |
| **6.** | SUBB A,direct | Direct byte | Subtract the content of direct byte to accumulator with borrow |
| **7.** | SUBB A,@Ri | Register indirect | Subtract the content of address of register to content of accumulator with borrow |
| **8.** | SUBB A,@data | Immediate | Subtract the data in the instruction to the content of accumulator with borrow |
| **9.** | MUL AB | Register | Multiply unsigned integer in the accumulator and the reg. B |
| **10.** | DIV AB | Register | Divide the unsigned 8-bit integer in the accumulator by unsigned integer in reg.B |

**Table 7 : Addition, Subtraction, Multiplication, Division Instructions**

**CONTROL INSTRUCTION:**

- The control instructions are: Jump Instruction, Call and Subroutine Instruction
- It supports the conditional jumping and call functions.
- If any conditional jump occurs, control instruction checks the status whether it is correct or wrong.
- According to this, it stores the current address in program counter and jump to the appropriate location.
- If the condition is true, it executes the next instruction.

**6. Jump Instructions, and**

**7. Call and Subroutine Instructions:**

| S. No | Instruction | Addressing Modes | Byte |
|---|---|---|---|
| **1.** | JZ rel | Jump if accumulator is zero<br>(pc)←(pc)+2<br>IF(A)=0 THEN<br>(pc)←(pc)+rel | 2 |
| **2.** | JNC rel | Jump if no carry (pc)←(pc)+2 If CY=0 THEN<br>(pc)←(pc)+rel | 2 |
| **3.** | JB bit | Jump if direct biis set<br>(pc)←(pc)+3 (pc)←(pc)+rel | 3 |
| **4.** | DJNZ byte | New byte =byte1<br>If new byte is not equal to zero then jumpto 16 bit address specified. | 2 or 3 |
| **5.** | LCALL addr 16 | Long jump (pc)←(pc)+3<br>(sp)←(sp)+1<br>(sp)←(pc)<br>(sp)←(sp)+1<br>(sp)←(pc) | 3 |
| **6.** | RET | Return from interrupt<br>(pc)←((sp))<br>(sp)←(sp)-1<br>(pc)←(sp)<br>(sp)←(sp)-1 | 1 |

**Table 8: Jump Instructions, and Call and Subroutine Instructions**

**PUSH and POP instruction:**

| S. No | Instruction | Operation | Example | Byte |
|---|---|---|---|---|
| **1.** | PUSH Direct | (SP)←(SP)+1;<br>((SP))←direct<br>The stack pointer is incremented by 1 | PUSH DPL<br>PUSH DPH | 2 |
| **2.** | POP direct | (direct)←((sp));<br>(sp)←(sp)-1<br>Stack pointer is decremented by 1 | POP DPL<br>POP DPH | 2 |

**Table 9: PUSH and POP instruction**

**Data Exchange instruction:**

| S. No | Instruction | Operation | Byte |
|---|---|---|---|
| **1.** | XCHA,Rn | Exchange data bytes b/w reg Rn and A | 1 |
| **2.** | XCHA,direct | Exchange data b/w directly within instruction and A | 2 |
| **3.** | XCH A,Ri | Exchange data bytes b/w A and Ri | 1 |

**Table 10: Data Exchange instruction**

**1. Program for addition of two 8-bit numbers**

```
CLR C
MOV A,# data 1
ADD A,# data 2
MOV DPTR, #8500
MOVX @ DPTR,A
LOOP: SJMP LOOP
```

**2. Program for Subtraction Of Two 8-bit Number**

```
CLR C
MOV A,# data 1
SUBB A,# data 2
MOV DPTR, #8500
MOVX @ DPTR,A
LOOP: SJMP LOOP
```

**3. Program for Multiplication Of Two 8-bit Number**

```
MOV A,# data 1
MOV B,# data 2
MUL AB
MOV DPTR, #8500
MOVX @ DPTR,A
INC DPTR
MOV A, B
MOVX @ DPTR,A
LOOP: SJMP LOOP
```

**4. Program for Division Of Two 8-bit Number**

```
MOV A,# dividend
MOV B,# divisor
DIV AB
MOV DPTR, #8500
MOVX @ DPTR,A
INC DPTR
MOV A, B
MOVX @ DPTR,A
LOOP: SJMP LOOP
```

**5. Write an assembly language program to perform the addition of two 16-bit numbers.**

```
MOV R0,#34H          //LOWER NIBBLE OF NO.1
MOV R1,#12H          //HIGHER NIBBLE OF NO.1
MOV R2,#0DCH         //LOWER NIBBLE OF NO.2
MOV R3,#0FEH         //HIGHER NIBBLE OF NO.2
CLR C
MOV A,R0
ADD A,R2
MOV 22H,A
MOV A,R1
ADDC A,R3
MOV 21H,A
MOV 00H,C
END
```

**6. Write an assembly language program to perform the subtraction of two 16-bit numbers.**

```
MOV R0,#0DCH         //LOWER NIBBLE OF NO.1
MOV R1,#0FEH         //HIGHER NIBBLE OF NO.1
MOV R2,#34H          //LOWER NIBBLE OF NO.2
MOV R3,#12H          //HIGHER NIBBLE OF NO.2
CLR C //
MOV A,R0
SUBB A,R2
MOV 22H,A
MOV A,R1
SUBB A,R3
MOV 21H,A
MOV 00H,C
END
```

**7. Program to find the sum of 10 numbers stored in the array**

```
MOV R0,#50H
MOV R2,#6
CLR A
MOV R7,A
XYZ: ADD A,@R0
JNC NEXT
INC R7
NEXT: INC R0
DJNZ R2 , XYZ
END
```

**8. Program to find the largest number in a set**

```
LABEL    MNEMONICS

         MOV DPTR,#4200
         MOV 50H,#00
         MOVX A,@DPTR
         MOV R0,A
         INC DPTR
  L1     MOVX A,@DPTR
         CJNE A,50H,4115
  L2     INC DPTR
         DJNE R0,L1
         MOV A,50H
         MOVX @DPTR,A
 L3      SJMP L3
         JC L2
        MOV 50H,A
        SJMP L2
```