# 8051 INSTRUCTION SET

**8051 has about 111 instructions. These can be grouped into the following categories**

- Arithmetic Instructions
- Logical Instructions
- Data Transfer instructions
- Boolean Variable Instructions
- Program Branching Instructions

## The following nomenclatures for register, data, address and variables are used while write instructions

- A: Accumulator
- B: "B" register
- C: Carry bit
- Rn: Register R0 - R7 of the currently selected register bank
- Direct: 8-bit internal direct address for data. The data could be in lower 128bytes of RAM (00 - 7FH) or it could be in the special function register (80 - FFH).
- @Ri: 8-bit external or internal RAM address available in register R0 or R1. This is used for indirect addressing mode.
- #data8: Immediate 8-bit data available in the instruction.
- #data16: Immediate 16-bit data available in the instruction.
- Addr11: 11-bit destination address for short absolute jump. Used by instructions AJMP & ACALL. Jump range is 2 kbyte (one page).
- Addr16: 16-bit destination address for long call or long jump.
- Rel: 2's complement 8-bit offset (one - byte) used for short jump (SJMP) and all conditional jumps.
- bit: Directly addressed bit in internal RAM or SFR

**Some Simple Instructions:**

```
MOV dest,source         ; dest = source
    MOV A,#72H          ;A=72H
    MOV R4,#62H ;R4=62H
    MOV   B,0F9H              ;B=the content of F9'th byte of RAM
    MOV   DPTR,#7634H
    MOV   DPL,#34H
    MOV   DPH,#76H
    MOV   P1,A          ;mov A to port 1
```

Note 1:

MOV A,#72H ≠ MOV A,72H

After instruction "MOV A,72H " the content of 72'th byte of RAM will replace in Accumulator.

Note 2:

MOV A,R3 ≡ MOV A,3

```
ADD   A, Source         ;A=A+SOURCE
    ADD   A,#6          ;A=A+6
    ADD   A,R6          ;A=A+R6

    ADD   A,6           ;A=A+[6] or A=A+R6
    ADD         A,0F3H        ;A=A+[0F3H]
SUBB  A, Source         ;A=A-SOURCE-C
    SUBB  A,#6          ;A=A-6
    SUBB  A,R6          ;A=A+R6
```

**MUl & Div:**

- MUL AB ;B|A = A*B

```
    MOV   A,#25H
    MOV   B,#65H
    MUL   AB            ;25H*65H=0E99
                              ;B=0EH, A=99H
```

- DIV AB ;A = A/B, B = A mod B

```
MOV   A,#25
      MOV   B,#10
      DIV   AB              ;A=2, B=5
SETB  bit             ; bit=1
CLR   bit             ; bit=0
SETB  C               ; CY=1
SETB  P0.0            ;bit 0 from port 0 =1
SETB  P3.7            ;bit 7 from port 3 =1
SETB  ACC.2           ;bit 2 from ACCUMULATOR =1
SETB  05              ;set high D5 of RAM loc. 20h
```

Note:

CLR instruction is as same as
SETB i.e.:

```
      CLR         C       ;CY=0
```

But following instruction is only for CLR:

```
      CLR         A       ;A=0
DEC   byte            ;byte=byte-1
INC         byte            ;byte=byte+1
INC         R7
DEC         A
DEC         40H             ; [40]=[40]-1
```

RR – RL – RRC – RLC A

**EXAMPLE:**
**RR A**

**RR:**

**RRC:**

**RL:**

**RLC:**

ANL - ORL – XRL

Bitwise Logical Operations:

AND, OR, XOR

EXAMPLE:

```
      MOV   R5,#89H
      ANL   R5,#08H
```

```
CPL     A       ;1's complement
```

Example:

```
        MOV     A,#55H ;A=01010101 B
L01:    CPL     A
        MOV     P1,A
        ACALL   DELAY
        SJMP    L01
```