8051 Assembly Language

Objective

(1) Data transfer Instructions

(2) Logical Instructions

(3) Arithmetic Instructions

(4) Control transfer Instructions

(5) Examples

Introduction

- Spends more time in moving Data's
- Therefore we find more instructions are provided for moving data
- Data transfer can be broadly classified into

  MOV destination, source

  PUSH source / POP destination

  XCH destination, source
- Data transfer exists between following memories

  1. Internal RAM
  2. Internal SFRs
  3. External RAM (data memory)
  4. Internal and External code memory

5 type of opcodes used for Data transfer are

1. **MOV**
2. **MOVX**
3. **MOVC**
4. **PUSH and POP**
5. **XCH**

Addressing modes supported by data refer to immediate, register, direct & indirect

(1) Immediate addressing

Instruction using # Data

(2) Register Addressing

Instruction using R0 to R7

(3) Direct Addressing

Instruction using RAM Address

Address in RAM Source / Destination of Data

(4) Indirect Addressing

Instruction using@R0 / @R1

Reg R0 | R1 in current Bank Address of Data Address in RAM

Some Examples

MOV A, # 0F1h

MOV A, R0

MOV DPTR, # 0ABC h

MOV R0, 45 h

MOV R1 , @R0

Classification of Instruction set

(1) Data Moves / Transfer

(2) Arithmetic

(3) Logical

(4) Control transfer

(5) Boolean

1. Data Transfer Instructions :

a. Immediate Addressing :

MOV A, # n

MOV Rr, # n

MOV add, # n

MOV @Rp, # n

MOV DPTR, # nn

b. Register Addressing:

MOV A, Rr

MOV Rr, A

XCH A, Rr

c. Direct Addressing :

MOV A, add / MOV add, A

MOV Rr, add / MOV add, Rr

MOV add1, add2

PUSH add

POP add

XCH A, add

d. Indirect Addressing :

Register Indirect :

MOV A, @ Rp / MOV @Rp, A

XCH A, @Rp

MOV add, @Rp / MOV @Rp, add

Code Data Indirect :

MOVC A, @A+DPTR

External Indirect :

MOVX A, @Rp / MOVX @Rp, A

External Indirect :

MOVX @DPTR, A / MOVX A, @DPTR

2. Arithmetic Instructions :

a. Immediate Addressing :

ADD A, # n

ex) ADD A,#02H --- (A)+02H => A

ADDC A, # n ---- (A)+02H+(C) =>A

SUBB A, # n

b. Register Addressing :

ADD A, Rr | SUBB A, Rr

ADDC A, Rr |

DEC A | INC A

DEC Rr | INC Rr | INC DPTR

DIV AB

MUL AB

DA A

c. Direct Addressing :

ADD A, add

ADDC A, add

DEC add / INC add

SUBB A, add

d. Indirect Addressing :

ADD A, @Rp

ADDC A, @Rp

INC @Rp | DEC @Rp

SUBB A,@Rp

3. Logical Instructions :

a. Immediate Addressing :

ANL A, # n

ANL add, # n

ORL A, # n

ORL add, #n

XRL A, # n

XRL add, # n

b. Register Addressing :

ANL A, Rr

CLR A

CPL A

ORL A, Rr

XRL A, Rr

RLA / RRA

RLC A / RRC A

SWAP A

c. Direct Addressing :

ANL A, add / ANL add, A

ORL A, add / ORL add, A

XRL A, add / XRL add, A

d. Indirect Addressing :

ANL A, @Rp

ORL A, @Rp

XRL A, @Rp

4. Miscellaneous Instructions :

NOP

5. Control Transfer Instructions :

a. Immediate Addressing :

CJNE A, # n, radd

CJNE Rr, # n, radd

CJNE @Rp, #n, radd

b. Register Addressing :

DJNZ Rr, radd

c. Direct Addressing :

	SJMP	LJMP/LCALL	AJMP/ACALL
ACALL Sadd **LCALL Ladd** **SJMP radd** **AJMP Sadd** **LJMP Ladd**	Short jump, relative address is 8 bit it support 127 bytes (127locations relative to PC) forward/backward	Long jump range is 64 kb, address is 16bits	Absolute jump to anywhere within 2k block of program memory, address is 11bits

C JNE A, add, radd

DJNZ add, radd

JC radd

JNC radd

JB b, radd

JNB b, radd

JBC b, radd

JZ radd

JNZ radd

RET

RETI

d. Indirect Addressing :

JMP @A+DPTR (External Program Memory)

6. Boolean Instructions :

ANL c, b

ANL c, /b

CLR c

CLR b

CPL c

CPL b

ORL c, b

ORL c, /b

MOV c, b

SETB c

SETB b