23VLS1401: Microcontroller and Computer architecture

Lecture 5 (U5)

Programming for Microcontroller 8051 based on Real Life problem statements.

A presentation by

Dr. Shubhangi Rathkanthiwar

Professor

Department of Electronics Engineering, YCCE, Nagpur, India

Session objectives

- To overview Logical instructions for the Microcontroller 8051
- To develop the programming technique in assembly language for given problem statement, store the source data, execute the program and observe the result in destination register or memory location.

Logical instructions

- ANL (Logical AND)
- ORL (Logical OR)
- XRL (Logical Exclusive OR)
- CPL (Complement)
- CLR (Clear)
- RL (Rotate left), RLC (Rotate left through carry)
- RR (Rotate Right), RRC (Rotate Right through carry)
- SWAP (Exchange nibbles)

Related Branching instructions

- CJNE
- DJNZ
- JC/JNC
- JZ/JNZ

Logical instructions

Opcode	Operand	Operation
ANL	A,#n	Logically AND immediate data byte with A. Store the result in A
ANL	A,addr	Logically AND data byte in Addr with A. Store the result in A
ANL	A,Rr	Logically AND data byte in register with A. Store the result in A
ANL	A,@Rp	Logically AND data byte in memory with address in Rp with A. Store the result in A
ANL	Addr,A	Logically AND immediate data byte at address with A. Store the result in A
ANL	Addr,#n	Logically AND immediate data byte with data at address. Store the result in A

Logical instructions

Opcode	Operand	Operation
ORL	A,#n	Logically OR immediate data byte with A. Store the result in A
ORL	A,addr	Logically OR data byte in Addr with A. Store the result in A
ORL	A,Rr	Logically OR data byte in register with A. Store the result in A
ORL	A,@Rp	Logically OR data byte in memory with address in Rp with A. Store the result in A
ORL	Addr,A	Logically OR immediate data byte at address with A. Store the result in A
ORL	Addr,#n	Logically OR immediate data byte with data at address. Store the result in A

Logical instructions

Opcode	Operand	Operation
XRL	A,#n	Logically EXOR immediate data byte with A. Store the result in A
XRL	A,addr	Logically EXOR data byte in Addr with A. Store the result in A
XRL	A,Rr	Logically EXOR data byte in register with A. Store the result in A
XRL	A,@Rp	Logically EXOR data byte in memory with address in Rp with A. Store the result in A
XRL	Addr,A	Logically EXOR immediate data byte at address with A. Store the result in A
XRL	Addr,#n	Logically EXOR immediate data byte with data at address. Store the result in A

Logical instructions

Opcode	Operand	Operation
RL	A	Rotate A one position left
RLC	A	Rotate A one position left through Carry
RR	A	Rotate A one position right
RRC	A	Rotate A one position right through Carry
SWAP	A	Exchange Higher order nibble with lower order

Related branching instructions

Opcode	Operand	Operation
CJNE	A,Addr,raddr	Compare A with data byte at Addr. If A < Addr, carry flag is set. If A >Addr, carry flag is reset.
CJNE	A,#n,raddr	Compare A with data byte n. If A < n, carry flag is set. If A >n, carry flag is reset.
CJNE	Rn,#n,raddr	Compare Rn with data byte n. If Rn< n, carry flag is set. If Rn>n, carry flag is reset.
CJNE	@Rp,#n,raddr	Compare data byte pointed by Rp with immediate data 'n'. If @Rp < n, carry flag is set. If @Rp>n, carry flag is reset.
DJNZ	Rn,raddr	Decrement Rn by 1 and jump to relative address if Rn is not equal to 0
DJNZ	Addr,raddr	Decrement data at Addr by 1 and jump to relative address if data at Addr is not equal to 0

Problem statement 1: 5 data bytes stored in external RAM locations starting at 2501H and is given in Sign Magnitude representation. WAP

a) to represent all the numbers as Negative numbers without modifying magnitude part of the number.

b) to represent all the numbers as Positive numbers without modifying magnitude part of the number.

```
ORG 0000H
MOV DPTR,#2501H
MOV R0,#05H
L1: MOVX A,@DPTR
    ORL A,#80H
    MOVX @DPTR,A
INC DPTR
DJNZ R0,L1
END
```

```
ORG 0000H
MOV DPTR,#2501H
MOV R0,#05H
L1: MOVX A,@DPTR
    ANL A,#7FH
    MOVX @DPTR,A
INC DPTR
DJNZ R0,L1
END
```

Problem statement 2: A string of 100 data bytes are stored from 2501H and represented in Sign Magnitude representation. WAP

a) To count no. of negative data bytes in the string and put the count in register R5

b) Transfer all positive data bytes in internal RAM starting at 25H

```
ORG 0000H
MOV DPTR,#2501H
MOV R0,#64H
MOV R1,#00H
L1: MOVX A,@DPTR
 RLC A
JNC AHEAD
INC R1
AHEAD: INC DPTR
DJNZ R0,L1
MOV A,R1
MOV R5,A
END
```

```
ORG 0000H
MOV DPTR,#2501H
MOV R0,#25H
MOV R1,#64H
L1: MOVX A,@DPTR
 RLC A
JC AHEAD
RRC A
MOV @R0,A
INC R0
AHEAD: INC DPTR
DJNZ R1,L1
END
```

Problem statement 3: WAP to reject the odd data bytes from a series of 10 numbers stored in the external RAM locations starting at 2501H and transfer even data bytes in internal RAM starting at 25H

```
ORG 0000H
MOV DPTR,#2501H
MOV R0,#25H
MOV R1,#0AH
L1: MOVX A,@DPTR
    RRC
JC REJECT
RLC
MOV @R0,A
INC R0
REJECT: INC DPTR
DJNZ R1,L1
HLT
```

Problem statement 4: WAP to exchange the nibbles of 10 data bytes stored from 2501H. Place the result in same memory locations.

```
ORG 0000H
MOV DPTR,#2501H
MOV R0,#0AH
L1: MOVX A,@DPTR
   SWAP A
  MOVX @DPTR,A
INC DPTR
DJNZ R0,L1
END
```

Problem statement 5: WAP to set D5, reset D4 bit of 5 numbers stored from 2501H. Place the result in same memory locations.

```
ORG 0000H
MOV DPTR,#2501H
MOV R1,#05H
L1: MOVX A,@DPTR
   ORL A,#20H
   ANL A,#0EFH
  MOVX @DPTR,A
INC DPTR
DJNZ R1,L1
END
```

Thank you