微机原理和接口技术

第五讲 指令系统与汇编程序3

提 纲

1. 指令系统概述
2. 寻址方式
3. 数据传送类指令
4. 算术运算类指令
5. 逻辑运算类指令
6. 控制转移类指令
7. 位操作指令
8. 查表指令的应用
9. 堆栈操作指令的应用
10. 十进制调整指令的应用
11. 逻辑指令与字节状态操作
12. 转移指令的应用

提 纲

4. 算术运算类指令

算数运算类指令

- **6.乘法指令(1条)：助记符MUL**(Multiply)

MUL AB ;(B)(A) ← (A)×(B)

➢ **功能：**将A和B中两个无符号8位二进制数相乘，所得的16位积的低8位存于A中，高8位存于B中。

如果乘积大于255时，即高位B不为0时，OV置位；否则OV置0。C总是清0。

- **7.除法指令(1条)：助记符DIV**(Divide)

DIV AB ;A/B; (A) ←商,(B) ← 余数

➢ **功能：**将A的内容除以B的内容，结果中的商保存于A，余数保存于B，并将C和OV置0。

当除数（B）=0时，结果不定，则OV置1。C总是清0。

算数运算类指令

- **8.十进制调整指令(1条)：助记符DA**(Decimal Adjustment)

  DA　A

➢ **功能：**对两个压缩BCD码（一个字节存放2位BCD码）数相加的结果进行十进制调整

- **注意：**

  1）只能用在ADD和ADDC指令之后，对相加后存放在A中的结果进行修正。

  2）两个压缩BCD码按二进制数相加之后，必须经过此指令的调整才能得到正确的BCD码累加和结果。

- **调整的条件和方法：**

  1）若（A0~3）>9或（AC）＝1，则（A0~3）←（A0~3）＋6，即低位加6调整。

  2）若（A4~7）>9或（Cy）＝1，则高位加6调整。

- **DA A 指令对C的影响是只能置位，不能清0。**

提 纲

5．逻辑运算类指令

算数运算类指令

- 逻辑操作类指令包含逻辑与、逻辑或、逻辑异或、求反、左右移位、清0等。该类指令不影响标志位，仅当其目的操作数为A时，对奇偶标志位P有影响。24条指令可分为5组。

  - 逻辑与操作指令: 6条
  - 逻辑或操作指令: 6条
  - 逻辑异或指令: 6条
  - 累加器清0和取反指令: 2条
  - 循环移位指令: 4条

算数运算类指令

- **1.逻辑与操作指令(6条)：助记符ANL**(AND Logic)

ANL A，Rn	；(A) ← (A) ∧(Rn)
ANL A，direct	；(A) ← (A) ∧(direct)
ANL A，@Ri	；(A) ← (A) ∧((Ri))
ANL A，#data	；(A) ← (A) ∧ #data
ANL direct，A	；(direct) ←(direct)∧(A)
ANL direct，#data	；(direct) ←(direct)∧ #data

➢ **功能：**将目的操作数和源操作数按“位”相“与”，结果存放到目的操作数单元中。

算数运算类指令

- **2.逻辑或操作指令(6条)：助记符ORL**(OR Logic)

```
ORL   A，Rn          ；(A) ←(A)∨(Rn)
ORL   A，direct      ；(A) ← (A) ∨(direct)
ORL   A，@Ri         ；(A) ← (A) ∨((Ri))
ORL   A，＃data      ；(A) ←(A) ∨#data
ORL   direct，A      ；(direct) ←(direct)∨(A)
ORL   direct，＃data ；(direct) ←(direct)∨＃data
```

➢ **功能：**将目的操作数和源操作数按“位”相“或”，结果存放到目的操作数单元中。

算数运算类指令

- **3.逻辑异或操作指令(6条)：助记符XRL**(Exclusive-OR Logic)

XRL A，Rn	;（A）←（A）⊕（Rn）
XRL A，drect	;（A）←（A）⊕（direct）
XRL A，@Ri	;（A）←（A）⊕（（Ri））
XRL A，rdata	;（A）←（A）⊕#data
XRL direct，A	;（direct）←（direct）⊕（A）
XRL direct，#data	;（direct）←（direct）⊕#data

➢ **功能：**将目的操作数和源操作数按“位”相“异或”，结果存放到目的操作数单元中。

➢异或运算规则：

0⊕0=0　　1⊕1=0

0⊕1=1　　1⊕0=1

算数运算类指令

- **4.累加器清0和取反指令(2条)：**
- **清0指令助记符CLR**（Clear)
- **取反指令助记符CPL**(Complement)
- CLR A ； (A) ←0

➢ **功能：**累加器A的内容清0（字节清0）。字节清0就此一条。

CPL A ； (A) ← (/A)

➢ **功能：**对累加器A的内容逐位取反，结果仍存在A中。

- 例： 设（A）=21H(0010 0001B)

执行CPL A指令后，（A）=DEH (1101 1110B)。

算数运算类指令

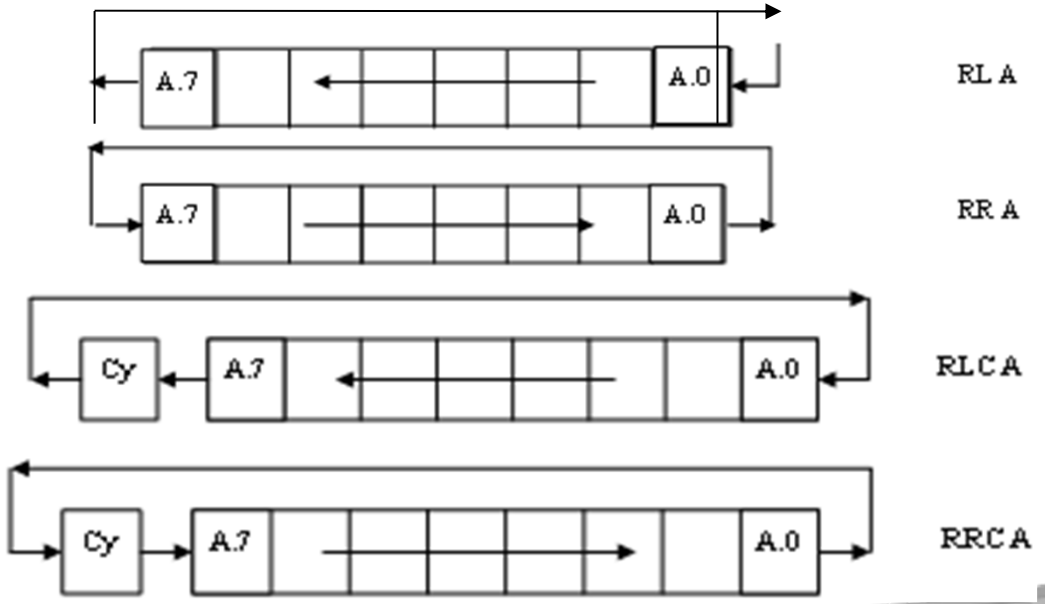
- **5.循环移位指令(4条)**

RL A ；循环左移指令；A的内容循环左移一位。(Rotate Left)

RR A ；循环右移指令； A的内容循环右移一位。(Rotate Right)

RLC A ；带C的循环左移；A的内容和C的内容整个左移一位。
; (Rotate Left through Carry)

RRC A ；带C的循环右移； A的内容和C的内容整个右移一位。



算数运算类指令

- **5.循环移位指令(4条)**
- 例：　MOV　A,#01H　　；（A）=0000 0001B
- 　　RL　　A　　；（A）=0000 0010B，（A）=02H
  　　RL　　A　　；（A）=0000 0100B，（A）=04H

左移一位相当于乘2；右移一位相当于除2。

例：2字节数R3R2中的内容（16位数）左移一位。（相乘结果不大于65536的情况）

```
PROG:  CLR     C
       MOV     A,R2
       RLC     A
       MOV     R2,A
       MOV     A,R3
       RLC     A
       MOV     R3,A
       RET
```

算数运算类指令

- **6.逻辑操作类指令举例**
- **例3-8：**已知A=85H，(45H)=A3H，分析执行“ANL A, 45H”指令、“ORL A, 45H”指令和“XRL A, 45H”指令的结果。

- 1）逻辑与：1 0 0 0 0 1 0 1
- ∧ 1 0 1 0 0 0 1 1
- 1 0 0 0 0 0 0 1

- **运行结果：**（A）=81H，(45H)=A3H，P=0。

- 2）逻辑或：1 0 0 0 0 1 0 1
- ∨1 0 1 0 0 0 1 1
- 1 0 1 0 0 1 1 1

- **运行结果：**A=0A7H，(45H)=0A3H，P=1。

- 3）逻辑异或：1 0 0 0 0 1 0 1
- ⊕1 0 1 0 0 0 1 1
- 0 0 1 0 0 1 1 0

- **运行结果：**A=26H，(45H)=0A3H，P=1。

算数运算类指令

- **6.逻辑操作类指令举例**
- **例3-9：**设(A)=B3H（10110011B），Cy=1，分析以下指令的执行结果。

执行“RLA”指令后，执行结果为（A）＝67H（01100111）。

执行“RRA”指令后，执行结果为（A）＝0D9H（11011001）。

执行“RLC A”指令后，执行结果为（A）＝67H（01100111），Cy=1。

执行“RRC A”指令后，执行结果为（A）=0D9H（11011001），Cy=1。

提 纲

6. 控制转移类指令

控制转移类类指令

- 程序的顺序执行是由程序计数器（PC）自动增1来实现的，要改变程序的执行顺序，控制程序的流向，必须通过控制转移类指令实现，所控制的范围为程序存储器的64KB空间。8051MCU的控制转移类指令，共17条，可分为4组。

  - 无条件转移指令：4条
  - 条件转移指令：8条
  - 子程序调用和返回指令：4条
  - 空操作指令：1条

控制转移类类指令

- **1.无条件转移指令(4条)**

```
LJMP   addr16   ；长跳转指令，跳转范围为64K； (PC) ← addr16
                ；3字节指令。（Long Jump)
AJMP   addr11   ；绝对跳转指令，跳转范围为2K;
                ； (PC)=(PC)+2，(PC)0-10 ← addr11。
                ；2字节指令。(Absolute Jump}
SJMP   rel      ；短跳转指令，跳转范围－128～＋127
                ; (Short Jump）
JMP   @A+DPTR   ；散转指令或间接跳转指令;
                ；（PC）← (A)+(DPTR);A的内容为8位无符号数。
                ；单字节指令。(Jump Indirect)
```

  - **区别：**跳转的范围不一样

- **原则上，**用SJMP或AJMP的地方都可以用LJMP替代。AJMP已很少使用。

控制转移类类指令

- **2.条件转移指令(8条)**
- **(1)判零转移指令:**

JZ rel ；A=0, 跳。（PC）←（PC）＋2＋rel

（Jump if ACC equal Zero） ；A≠0,不跳,继续向下执行。即(PC)←(PC)＋2

JNZ rel ；A≠0, 跳。（PC）←（PC）＋2＋rel

(Jump if ACC Not equal Zero) ；A＝0,不跳,继续向下执行。即(PC)←(PC)＋2

控制转移类类指令

- 问题：程序跳哪里去了？
- **(1)判零转移指令：**

JZ rel ；A=0, 跳。（PC）←（PC）＋2＋rel

（Jump if ACC equal Zero） ；A≠0,不跳,继续向下执行。即(PC)←(PC)＋2

JNZ rel ；A≠0, 跳。（PC）←（PC）＋2＋rel

(Jump if ACC Not equal Zero) ；A＝0,不跳,继续向下执行。即(PC)←(PC)＋2

例： 设（A）＝01H,

```
JZ    LABEL1    ;
DEC   A         ;
JZ    LABEL2    ;
LABEL1：------
LABEL2：------
```

控制转移类类指令

- 答案
- **(1)判零转移指令:**

JZ rel ；A=0, 跳。（PC）←（PC）＋2＋rel

（Jump if ACC equal Zero） ；A≠0,不跳,继续向下执行。即(PC)←(PC)＋2

JNZ rel ；A≠0, 跳。（PC）←（PC）＋2＋rel

(Jump if ACC Not equal Zero) ；A＝0,不跳,继续向下执行。即(PC)←(PC)＋2

例： 设（A）＝01H，执行程序

```
JZ    LABEL1        ；因为（A）≠ 0，程序继续执行
DEC   A      ；（A）-1=00H
JZ    LABEL2        ；因为（A）=00H，程序转向标号
                ；LABEL2 的地址执行
LABEL1：------
LABEL2：--这里！----
```

控制转移类类指令

- **2.条件转移指令(8条)**
- **(2)数值比较转移指令（均为3字节指令）**
- 助记符： CJNE（Compare and Jump if Not Equal）

CJNE A, direct, rel

CJNE A, #data，rel

CJNE Rn, ＃data，rel

CJNE @Ri, ＃data，rel

➢ **功能:**

- 对指定的两操作数进行比较，即(操作数1)－(操作数2)，比较结果仅影响标志位C，2个操作数的值不变；
- 比较不等，程序转移，目的地址=该指令的PC+3+rel，即 (PC) ←(PC) ＋3＋rel ，且:
  - 若(操作数1) ＞＝(操作数2)，C ← 0;
  - 若(操作数1) ＜(操作数2)，C ← 1 。
- 比较相等，程序继续顺序执行，即(PC) ← (PC)＋3

控制转移类类指令

- **2.条件转移指令(8条)**
- **(3)循环转移指令：** DJNZ（Decrement and Jump if Not equal Zero）

```
DJNZ  Rn，rel       ;(Rn) ← (Rn)-1;
                    ;若(Rn)≠0,跳,（PC） ←(PC)＋2＋rel
                    ;若(Rn)=0,不跳；即（PC）←  （PC）＋2
DJNZ  direct，rel   ; direct的内容为判断依据
```

➢ **功能：**

- Rn或direct的内容减1，判别其内容是否为0。
- 若不为0，跳转到目标地址，继续执行循环程序；
- 若为0，则结束循环程序段，程序往下执行。

控制转移类类指令

- **3.子程序调用和返回指令(4条)**
- **(1)长调用指令：助记符LCALL**(Long Subroutine Call)

LCALL addr16 ;(PC) ← (PC) ＋3 , 下条指令地址自动压栈保护;

;(SP) ← (SP) +1, ((SP)) ← PCL，PC低8位入栈;

;(SP)←(SP)+1 , ((SP)) ← PCH，PC高8位入栈;

; （PC） ← addr16（子程序首址）

该指令可调用存放在64KB ROM空间任何位置的子程序。

(2)绝对调用指令：助记符ACALL(Absolute Subroutine Call)

ACALL addr11 ;(PC) ← (PC) ＋2，PC自动压入堆栈保护, (SP) ←(SP) ＋2

; PC(10~0) ←addr11 ,PC 的高5位不变,执行子程序。

被该指令调用的子程序入口地址必须与ACALL下一条指令在相同的2KB ROM空间中。由于MCU的ROM容量不再是问题，所以这条指令很少使用。

控制转移类类指令

- **3.子程序调用和返回指令(4条)**
- **(3)子程序返回指令：助记符RET**(Return from Subroutine)
- RET ；（PCH）←（（SP）），（SP）←（SP）- 1
- ；（PCL）←（（SP）），（SP）←（SP）- 1

➢ **功能：**从堆栈顶部弹出子程序调用时，压入保护的断点地址到PC，即返回到主程序。子程序的最后一条指令必须是RET指令。

(4)中断程序返回指令：助记符RETI(Return from Interrupt Subroutine)

除具有RET的功能外,还可以恢复中断逻辑。

- RET 和RETI决不能互换使用
- 中断子程序的最后一条指令，必须是RETI指令。

4.空操作指令：助记符NOP(No Operation)

NOP ；（PC）←（PC）＋1

没有实际操作，只是花了一个机器周期时间。常用于软件延时。

控制转移类类指令

- **5.控制转移类指令举例**
- **例3-10：**已知(SP)=60H，标号地址MA为0123H，SUB子程序的地址为5060H。分析执行LCALL指令后，PC、SP以及堆栈顶部的数值。
- （0123H） MA： LCALL SUB
- （0126H） ……
- ……
- ORG 5060H
- （5060H）SUB: ……
- ……
- RET
- 问题：（PC）＝？，（SP）＝？，（61H）＝？，
- （62H）＝？

控制转移类类指令

- **5.控制转移类指令举例**
- **例3-10：**已知(SP)=60H，标号地址MA为0123H，SUB子程序的地址为5060H。分析执行LCALL指令后，PC、SP以及堆栈顶部的数值。
- （0123H） MA： LCALL SUB
- （0126H） ……
- ……
- ORG 5060H
- （5060H）SUB: ……
- ……
- RET
- 结果：（PC）＝5060H，（SP）＝62H，（61H）＝26H，（62H）＝01H。

Thank you!