MCS-51与D/A转换器的接口

- 嵌入式系统与A/D、D/A转换器
- D/A转换器原理
- DAC0832芯片介绍
- MCS-51与DAC芯片的接口

1

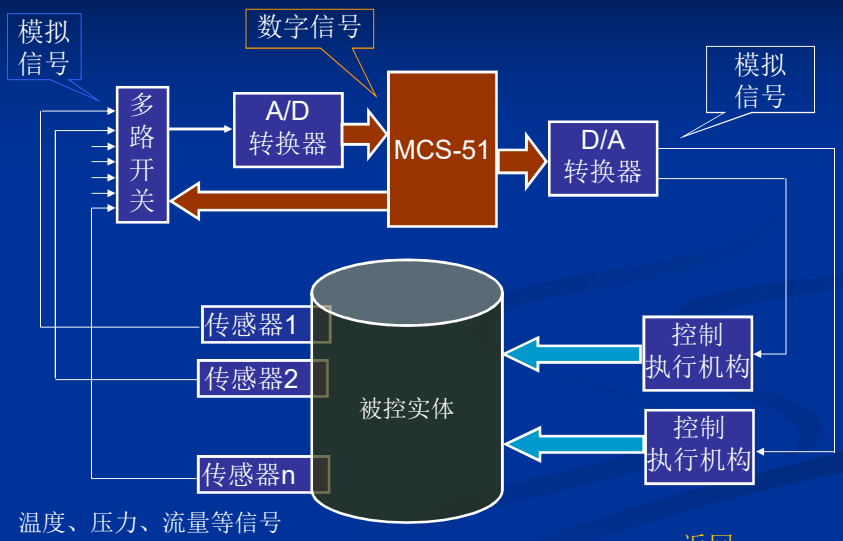
嵌入式系统与 A/D、D/A转换器

- 单片机作为嵌入式控制器嵌入在仪表、家电和各种智能化设备中。在这些嵌入式系统中，要对各种输入的模拟信号进行采集、分析和运算后产生对应的操作。
- CPU不能直接处理模拟信号，必须使用A/D转换器将模拟信号转化为CPU能识别、处理的数字信号;

  同理，CPU通过D/A转换器将数字信号经DAC转换为模拟量送给模拟伺服机构进行输出控制。
- A/D、D/A转换器的应用使CPU与输入、输出连成一个有机的整体，使单片机具有了真正意义上的“微控制器”的作用。

2

单片机在嵌入式系统中的应用示意图



返回

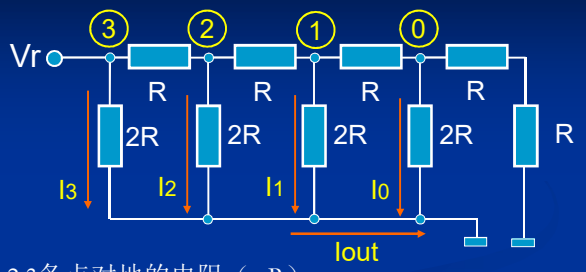
3

D/A转换器原理

将数字量N（D7～D0）转换为模拟量Vout的器件称之为D/A转换器。

4

T型电阻网络构成的D/A转换器结构与原理



$I_{out} = I_3+I_2+I_1+I_0$

- 分析0,1,2,3各点对地的电阻（=R）
- 根据各点电压求出I_3,I_2,I_1,I_0电流：

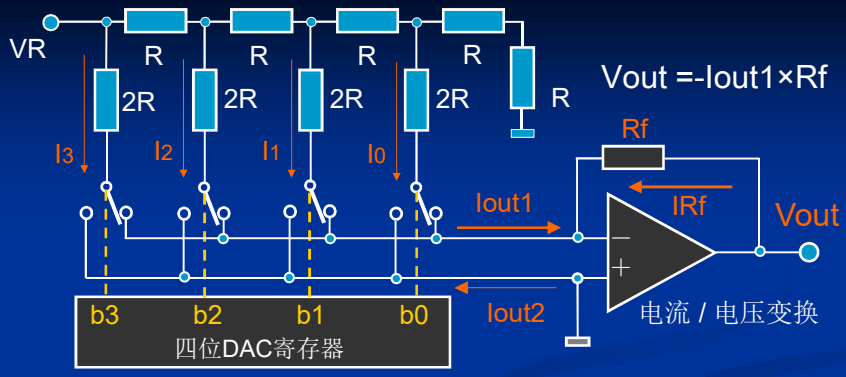
因为$I_3 = I_{总} \times 1/2$ 所以$I_3 = Vr/(2\times R) =$ $2^3 \times \frac{Vr}{2^4 \times R}$

$U_2 = Vr/2$ 所以$I_2 = Vr/(2\times 2R) =$ $2^2 \times \frac{Vr}{2^4 \times R}$

$U_1 = U2/2 = Vr/4$ 所以$I_1 = Vr/(4\times 2R) =$ $2^1 \times \frac{Vr}{2^4 \times R}$

$U_0 = U1/2 = Vr/8$ 所以$I_0 = Vr/(\times 2R) =$ $2^0 \times \frac{Vr}{2^4 \times R}$

5



$$Iout1 = b3 * I3 + b2 * I2 + b1 * I1 + b0 * I0$$

$$= \left(b3 * 2^3 \times \frac{Vr}{2^4 \times R} + b2 * 2^2 \times \frac{Vr}{2^4 \times R} + b1 * 2^1 \times \frac{Vr}{2^4 \times R} + b0 * 2^0 \times \frac{Vr}{2^4 \times R} \right)$$

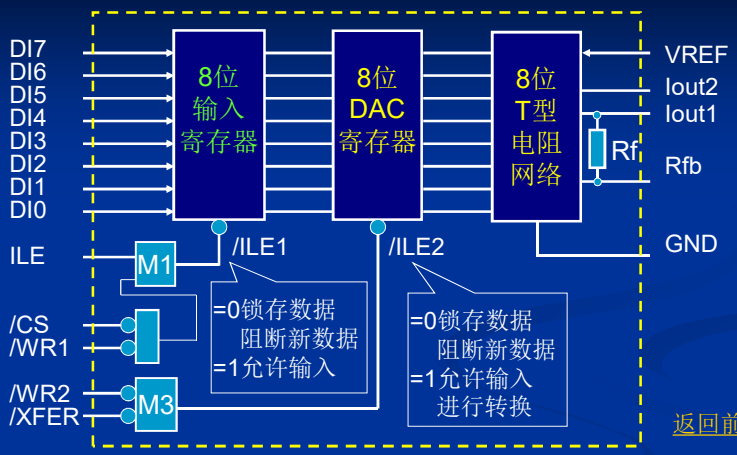
$$= (b3 * 2^3 + b2 * 2^2 + b1 * 2^1 + b0 * 2^0) \times \frac{Vr}{2^4 \times R} = B \times \frac{Vr}{2^4 \times R}$$

返回

其中：bi=0或=1 B:为4位二进制数 因此：Iout1与B成正比

6

DAC0832芯片介绍



返回前一次

- 8位输入数据寄存器:存放CPU送来得数据，由/ILE1控制；
- 8位DAC寄存器:存放待转换的数据，由/ILE2控制；
- 带有开关的8位T型电阻网络: 输出与数字量对应的模拟电流。

7

DAC0832芯片引脚介绍

- 数字量输入线DI_7～DI_0；
- 第1级缓冲控制线：

1. ILE:允许数字量输入线，高电平有效。
2. /CS:片选线，低电平有效。
3. /WR1:写命令控制线，低电平有效。

【锁存原理】

ILE= 1, /CS= /WR1=0时：M1为1。数据进入输入寄存器；

反之，条件不满足时：M1= 0。锁存器锁存数据。

8

- 第2级缓冲控制线（控制DAC新的转换时刻）：

1. /WR2:写命令控制线，低电平有效。
2. /XFER:输入传送控制线，低电平有效。

【锁存原理】

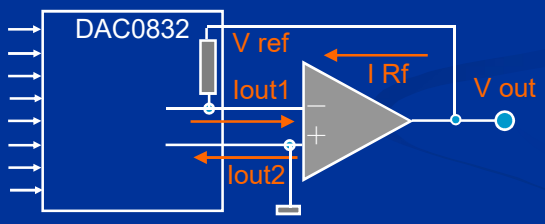
/WR2=0，/XFER=0时：M3为1。数据进入DAC寄存器,通过T型电阻网络实现DAC转换；

反之，条件不满足时：M3=0。锁存器锁存数据。DAC寄存器不接收输入寄存器的数据。

9

- 输出线：

1. R fb:与外接运算放大器的输出端相连。
2. Iout1、Iout2:模拟电流输出线，分别与运算放大器的反相端、同相端连接。



电流 / 电压变换

返回

10

- 电源线：

1. VCC:电源输入线，+5V～+15V之间。
2. VREF:参考电压输入线，-10V～+10V范围内，由基准电源提供。
3. DGND:数字电源地。
4. AGND:模拟电源地。

11

MCS-51与DAC0832芯片的接口

- 根据DAC0832结构特点，DAC0832的接法有：

1. **无缓冲的直通方式：** 用于无CPU的普通仪表场合。
2. **单 缓 冲 方 式：** 在单片机系统中常用的使用模式。
3. **双 缓 冲 方 式：** 用于多DAC的应用场合。

转内部框图

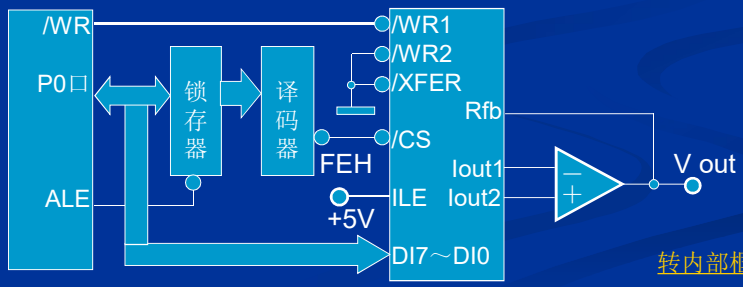
12

1，单缓冲方式

- 将DAC两个缓冲器中的后级直通（/XFER，/WR2=0）。前级的/WR1与单片机的/WR连接，/CS与地址译码器连接。这样，执行MOVX指令时，DAC就开始进行转换。

MOV R0,#0FEH ；DAC0832的地址送R0

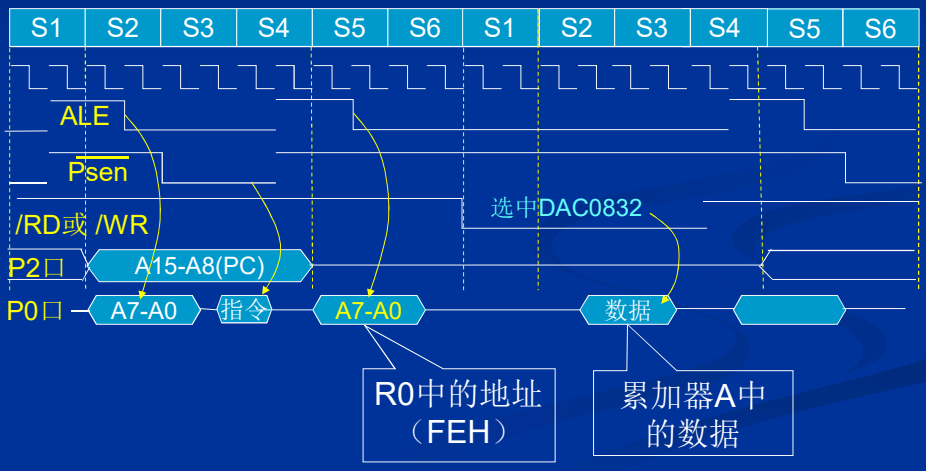
MOVX @R0,A ；产生/WR、/CS信号使0832进行转换。



转内部框图

13

MOVX @R0,A 指令时序图



14

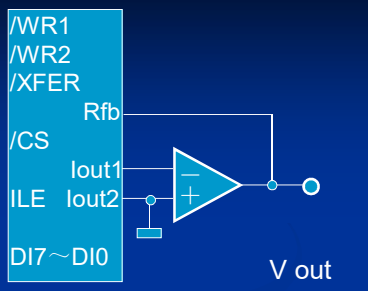
- 使用DAC0832作波形发生器，输出锯齿波、三角波和方波。

1，锯齿波发生器程序：

```
        ORG 1000H
START:MOV R0,#0FEH  ；DAC口地址
        MOVX @R0,A      ；数据送DAC
        INC A           ；数据加一
        SJMP START      ；返回继续
        END
```

【说明】：单极性输出Vout

$$= -B * \frac{V_{rfe}}{2^n} = -B * \frac{V_{rfe}}{256}$$



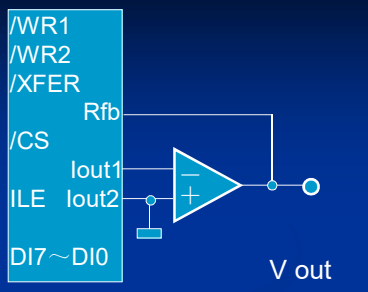
15

2，三角波发生器程序：

```
        ORG 1000H
START: CLR A
        MOV R0,#0FEH
DOWN: MOVX @R0,A
        INC A
        JNZ  DOWN
UP:     DEC A
        MOVX @R0,A
        JNZ UP
        SJMP DOWN
        END
```

【说明】：单极性输出Vout

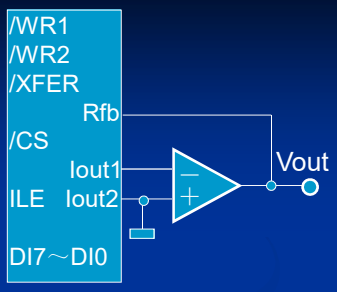
$$= -B * \frac{Vrfe}{2^n} = -B * \frac{Vrfe}{256}$$



16

3,方波发生器程序：

```
        ORG 1000H
START:MOV R0,#0FEH ；DAC口地址
LOOP:  MOV  A,#33H    ；数据为33H
       MOVX@R0,A      ；DAC转换
       ACALL  DELAY   ；调延时
       MOV A,#0FEH    ；数据为FEH
       MOVX @R0,A     ；DAC转换
       ACALL  DELAY   ；调延时
       SJMP  LOOP     ；返回继续
DELAY:
        :
       END
```

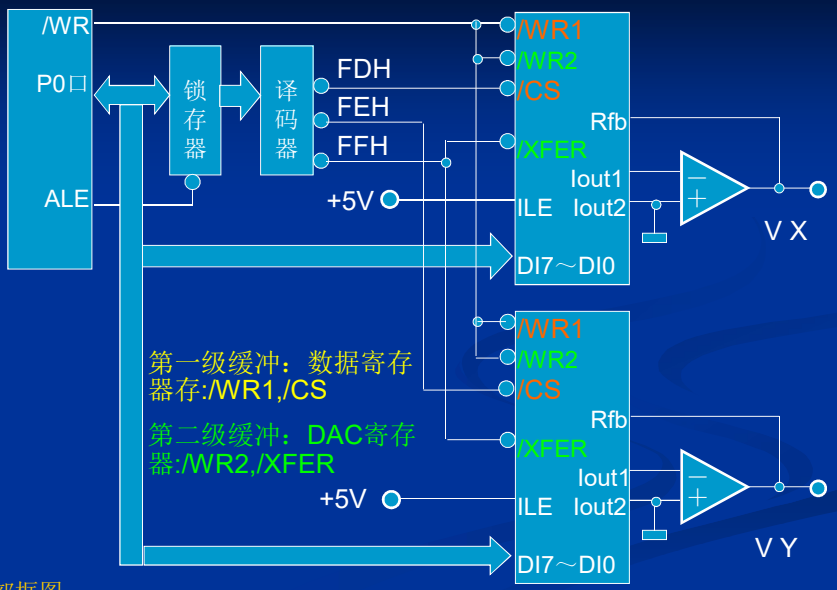


【说明】：单极性输出Vout

$$=-B*\frac{Vrfe}{2^n}=-B*\frac{Vrfe}{256}$$

17

2，双缓冲方式



转内部框图

18

- 设内部RAM单元有两个长度为30的数据块，起始地址分别为DA1和DA2，试编程将DA1,DA2中的数据分别从1#和2#DAC0832输出的程序。

- 【解】：FDH:1#DAC0832数字量输入控制口；
  FEH:2#DAC0832数字量输入控制口；
  FFH:1#,2#DAC0832的DAC转换控制口；

- R2为数据块长度计数器
- 0区R1为DA1数据块指针，1区R1为DA2数据块指针；
- R0用于存放DAC口地址。

19

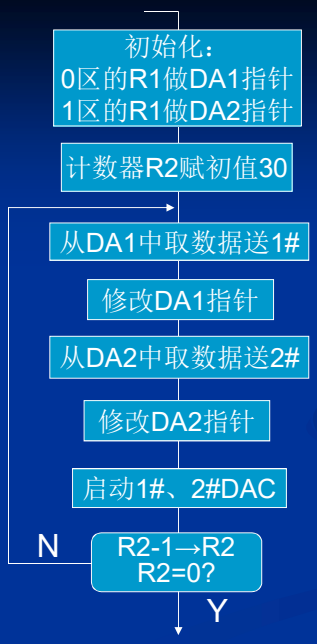
程序清单

```
        ORG 1200H
DA1     DATA 20H        ;DATA为数据、地址赋值伪指令
DA2     DATA 40H        ;为左边的“字符”名称赋值
DTOUT: MOV R1,#DA1      ;0区的R1指向DA1数据区
       MOV R2,#30       ;数据区长度30送计数器R2
       SETB RS0         ;CPU转向1#工作寄存器区
       MOV R1,#DA2      ;1#区R1指向DA2数据区
       CLR RS0          ;返回0#工作寄存器区
```

20

```
NEXT: MOV R0,#0FDH     ;指向1#输入数据口
      MOV A,@R1        ;取DA1数据
      MOVX @R0,A       ;数据送1#的数据寄存器
      INC R1           ;修改DA1数据区指针
      SETB RS0         ;CPU转向1#工作寄存器区
      MOV R0,#0FEH     ;指向2#输入数据口
      MOV A,@R1        ;取DA2数据
      MOVX @R0,A       ; 数据送2#的数据寄存器
      INC R1           ;修改DA2数据区指针
      INC R0           ;R0=FFH,指向1#、2#后级
      MOVX @R0,A       ;同时启动1#、2# DAC进行转换
      CLR RS0          ;CPU转回0# 工作寄存器区
      DJNE R2,NEXT     ;30个数据是否完成
      SJMP DTOUT       ;未完时，转DTOUT继续
      END
```

21



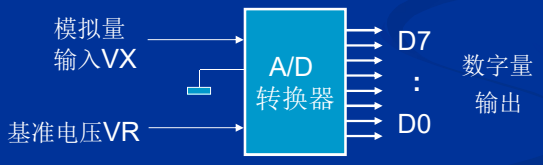
程序流程图

本章目录

22

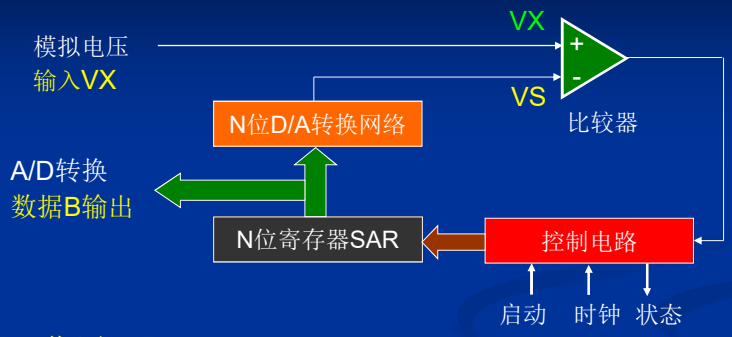
MCS-51与A/D转换器的接口

- 逐次比较式A/D转换器原理逐次比较
- A/D转换器ADC0809MCS-51
- 与ADC0809的接口电路



23

逐次比较式A/D转换器原理



【工作原理】：

N位寄存器中的二进制数 B 通过DAC电路转换为对应的模拟信号VS，将VS与VX相比较。若VS=VX，则与VS对应的二进制数B就作为VX的数字量。（如何寻找B?）

返回

24

逐次比较（逐次逼近）原理

1，N位寄存器首先形成10000000；经DAC转换后送比较器与VX相比较；

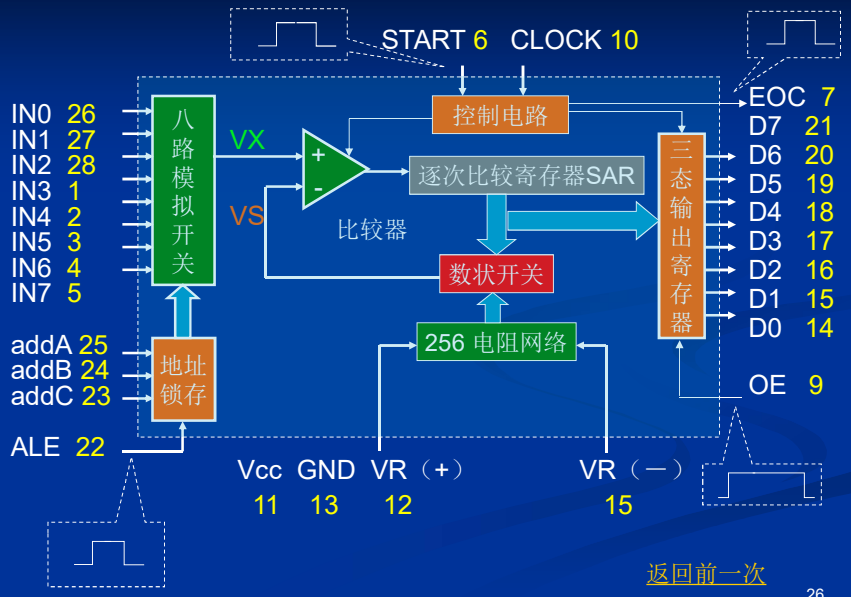
2，若VX>VS时,比较器通知N位寄存器保留最高位“1”；
若VX<VS时,比较器通过控制器将最高位“1”清除
（因为VX小于量程的一半127）。

3，N位寄存器对次高位置1，再重复上面的过程，确定该位是“1”或“0”。

4，经过8次比较确定了N位计数器从D7～D0的8位数据。整个过程由输入一个“启动”信号开始，到“状态”端输出一个标志信号结束。

25

逐次比较式A/D转换器ADC0809



返回前一次

26

ADC0809 芯片的引脚

1. IN0～IN7:　　八路模拟电压输入端；
2. ALE: 地址锁存控制信号，上升沿送入、下降沿锁存；
3. addA～ addC:　地址输入线；
4. START:　启动输入端，上升沿清SAR,下降沿启动ADC。
5. EOC:转换结束标志输出. 高电平表明转换完成。再次启动ADC时该引脚变低电平，直到转换完成后再次变高电平。

结构图

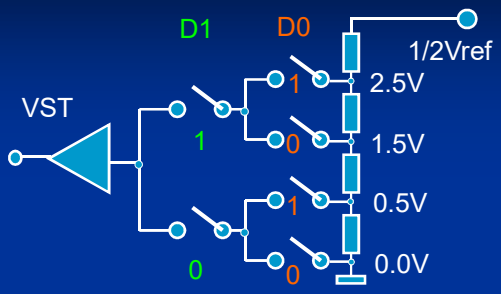
27

6. D7～D0:数字量输出线。
7. OE:输出三态控制线.置高电平时数据经D7～D0向外输出。
8. CLOCK:时钟输入端。 提供640KH逐次比较脉冲时序。
9. Vref（+）、 Vref（-）:参考电压输入，为电阻网络提供电压。 Vref（+）、 Vref（-）可以与Vcc和GND 连接。
10. Vcc、GND:电源和地。

结构图

28

简化的两位电阻阶梯和树状开关



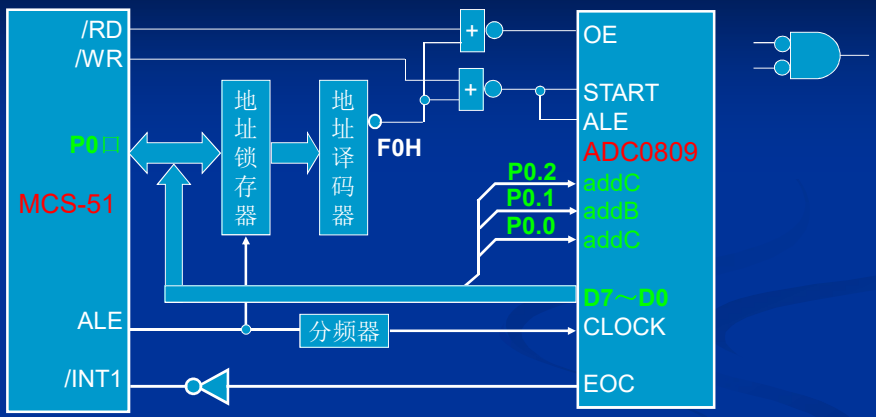
D1 D0	VST
0 0	0.0V
0 1	0.5V
1 0	1.5V
1 1	2.5V

- 由左边的两组树状电子开关，右边四只分压电阻构成；
- 树状开关D1、D0由SAR（逐次比较寄存器）对应控制；
- 树状开关D1、D0的状态与DAC输出VST的关系见表；
- 实际电路为8位,既256个分压电阻，形成256个标准电压供树状开关使用

返回

29

MCS-51与ADC0809的接口电路

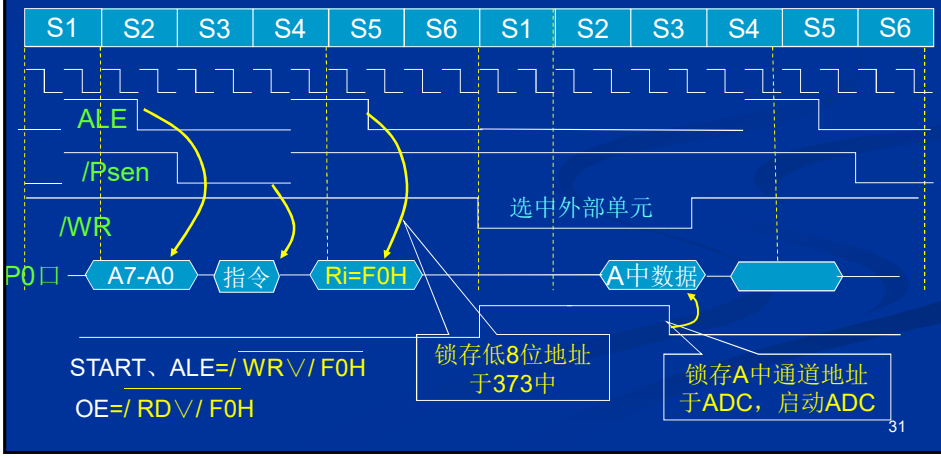


1. ADC的启动信号有哪个器件控制，单片机如何启动ADC？
2. 单片机如何知道ADC转换是否结束？
3. ADC数据输出三态门由谁控制，单片机如何读ADC数据？

30

MOVX @Ri ,A 指令的时序

正确理解 MOVX @Ri，A指令，是设计ADC与单片机接口的关键。指令执行周期：P0先输出Ri中的地址F0H（地址信号所存在373中），后输出A中通道代码（数据信号锁存到ADC中）。



31

系统的工作过程与原理

- 使用MOVX @Ri，A 指令启动ADC。此时：
  - ① Ri中内容为ADC的口地址=F0H；
  - ② A中数据为通道代码；
  - ③ 指令产生 / WR 信号。
- 当ADC转换完成后，EOC的正脉冲经反相后送单片机的INT1并引发中断（如果中断是开放的）。在中断服务程序中使用 MOVX A ,@Ri 指令读取ADC的数据；
- 使用MOVX A ,@Ri 指令来读ADC的数据，Ri=F0H,指令产生 / RD 信号。

32

对IN0-IN7上的模拟电压采集并送到内部RAM30H开始的单元（采用中断方式）

```
        ORG 0000H
        LJMP START
        ORG 0013H
        LJMP  CINT1
        ORG 0A00H
START:  MOV R1,#30H     ;数据区指针R1赋初值30H
        MOV R4,#08H     ;计数器R4赋初值08H
        MOV R2,#00H     ;通道代码送R2
        SETB  EA
        SETB  EX1               ;开/INT1中断
        SETB  IT1               ;设/INT1为边沿触发
        MOV  R0,#0F0H           ;ADC地址送间址寄存器R0
        MOV  A,R2               ;通道代码送累加器A
        MOVX @R0,A      ;送通道代码并启动ADC
        SJMP  $                 ;等待中断
```

TCON寄存器

33

```
        ORG  0100H
CINT1:  MOV  R0,#0F0H   ;中断服务程序
        MOVX  A,@R0     ;读入ADC数据
        MOV  @R1,A      ;存入转换的数据
        INC  R1         ;修改数据区指针R1
        INC  R2         ;修改通道代码寄存器R2
        MOV  A,R2       ;通道代码送累加器A
        MOVX  @R0,A     ;送下一个通道代码并启动ADC
        DJNZ R4,LOOP    ;若未采集完转LOOP
        CLR  EX1        ;采集完时，关中断
LOOP:   RETI            ;中断返回
        END
```

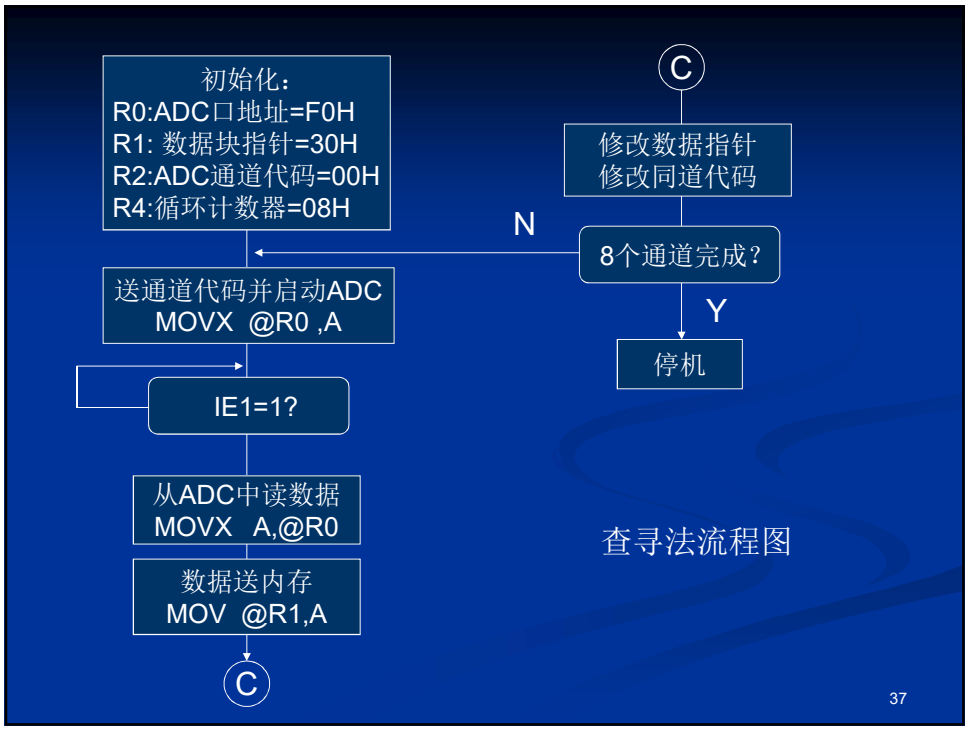
34

35

采用查询方式的ADC采集程序

```
        ORG 0000H
        LJMP 0100H
        ORG 0100H
START:  MOV R1,#30H     ;数据区指针R1赋初值30H
        MOV R4,#08H     ;计数器R4赋初值08H
        MOV R2,#00H     ;通道代码送R2
        MOV A,#00H      ;通道代码送A
        MOV R0,#0FEH
LOOP:   MOVX @R0,A      ;送通道代码，启动ADC
        JB  IE1,$       ;ADC转换是否结束
        MOVX A,@R0
        MOV A,@R1       ;取ADC的转换数据
        INC R1
        INC A
        DJNZ R4,LOOP
        SJMP $
        END
```

36


初始化：
R0:ADC口地址=F0H
R1: 数据块指针=30H
R2:ADC通道代码=00H
R4:循环计数器=08H
送通道代码并启动ADC
MOVX @R0 ,A
IE1=1?
从ADC中读数据
MOVX A,@R0
数据送内存
MOV @R1,A
C
C
修改数据指针
修改同道代码
N
8个通道完成？
Y
停机
37

查寻法流程图