

第9章 串行扩展技术

(课时：6学时)

教学目的

- 了解串行扩展的分类和特点。
- 了解Microware、1-wire、USB和CAN总线。
- 掌握I²C总线的原理。
- 掌握SPI总线的原理。

学习重点和难点

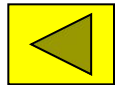
- I²C总线的软件模拟。
- SPI总线的软件模拟。

第9章 串行扩展技术

- 9.1 串行扩展概述
- 9.2 UART串行扩展接口
- 9.3 I²C串行扩展总线
- 9.4 SPI串行扩展接口
- 本章小结
- 习题

9.1 串行扩展概述

- 9.1.1 串行扩展的种类
- 9.1.2 串行扩展的特点



新一代单片机技术的显著特点之一就是串行扩展总线和接口的推出。过去常常通过并口扩展外设或是使用UART串口移位寄存器方式扩展(74LS165并入串出)，并行口连线复杂、外设工作方式各异、I/O与RAM的编址混合等，独立性较差。

常用的串行扩展总线和接口有I²C总线、SPI总线、Microware总线、1-wire总线和CAN总线等。

9.1.1 串行扩展的种类

常用的串行扩展总线和接口有I²C总线、SPI总线、Microware总线、1-wire总线和CAN总线等。

9.1.2 串行扩展的特点

(1) I²C总线 (Inter Integrated Circuit) 串行扩展总线

二线制，器件地址的硬件设置，软件寻址代替硬件寻址（片选法），硬件可灵活扩展。I²C总线简单，结构紧凑，易于实现模块化和标准化。

IC之间以及与I/O之间需要通信。

I²C主要用于内部IC控制应运而生。

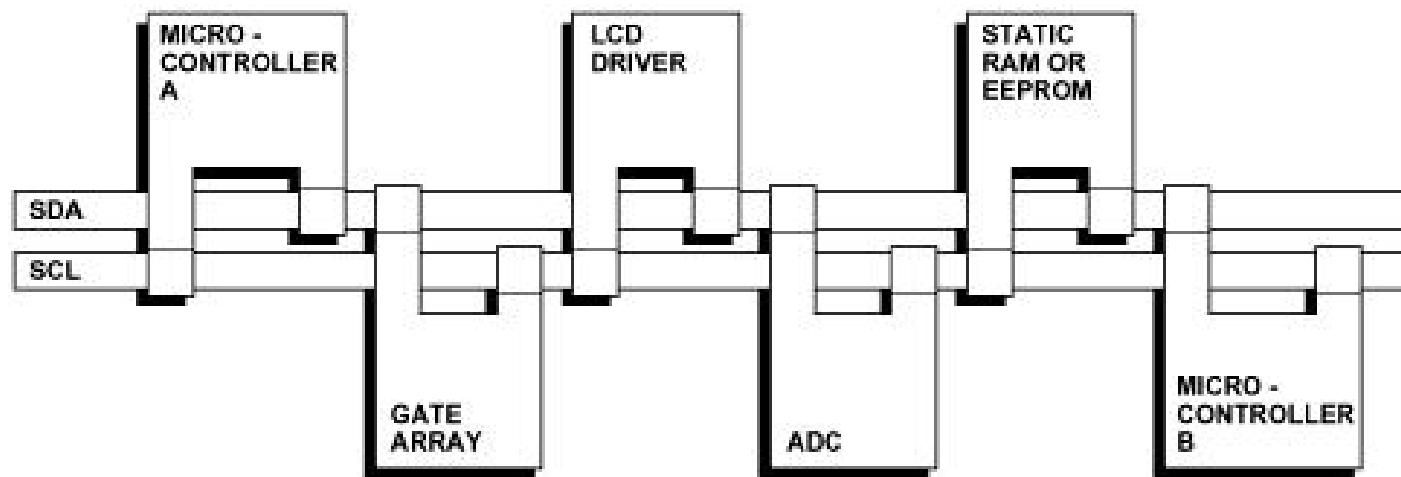
PHILIPS、兼容芯片繁多、世界工业标准。

I²C 总线上数据的传输速率

在标准模式下可达**100kbit/s**。

在快速模式下可达**400kbit/s**。

在高速模式下可达**3.4Mbit/s**。



I²C结构原理图

每个器件都有一个唯一的地址，而且可以是单接收器件(如**LCD**驱动器)或者可以是接收或发送器件(如存储器)

9.1.2 串行扩展的特点

(2) SPI (Serial Peripheral Interface) 串行扩展接口

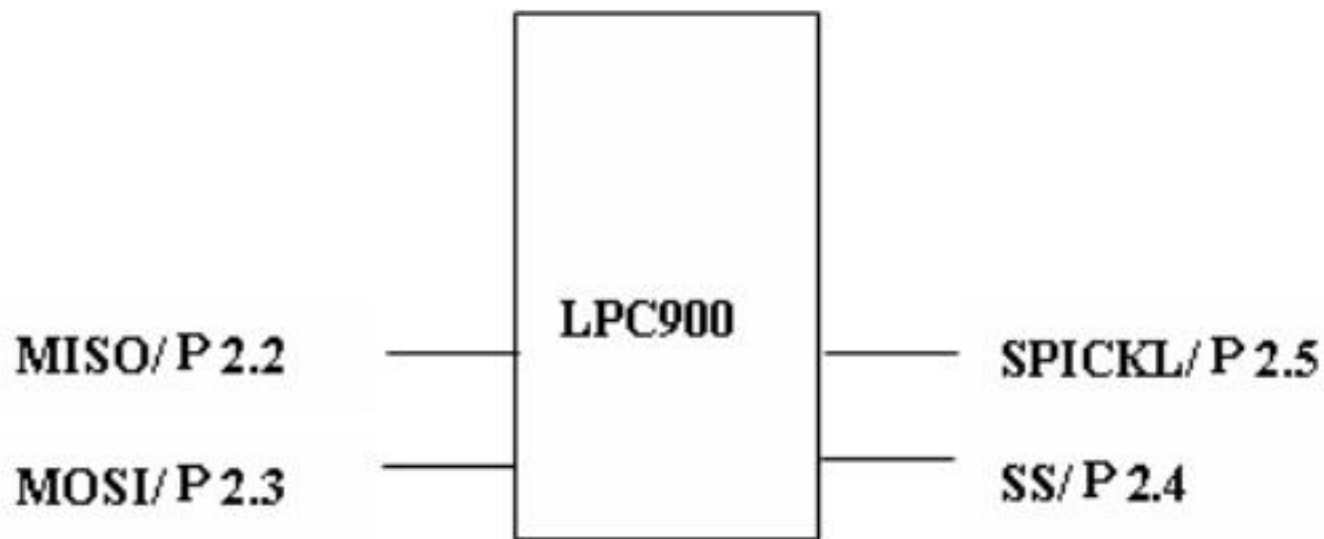
SPI总线是Motorola公司提出的一种同步串行外设接口。允许MCU与各种外围设备以同步串行方式进行通信。其外围设备种类繁多。

SPI总线是三线制，可直接与多种标准外围器件直接接口，从设备较少而没有总线扩展能力的单片机系统中使用特别方便。即使在有总线扩展能力的系统中采用SPI设备也可以简化电路设计，省掉很多常规电路中的接口器件，从而提高了设计的可靠性。

注意：请区分

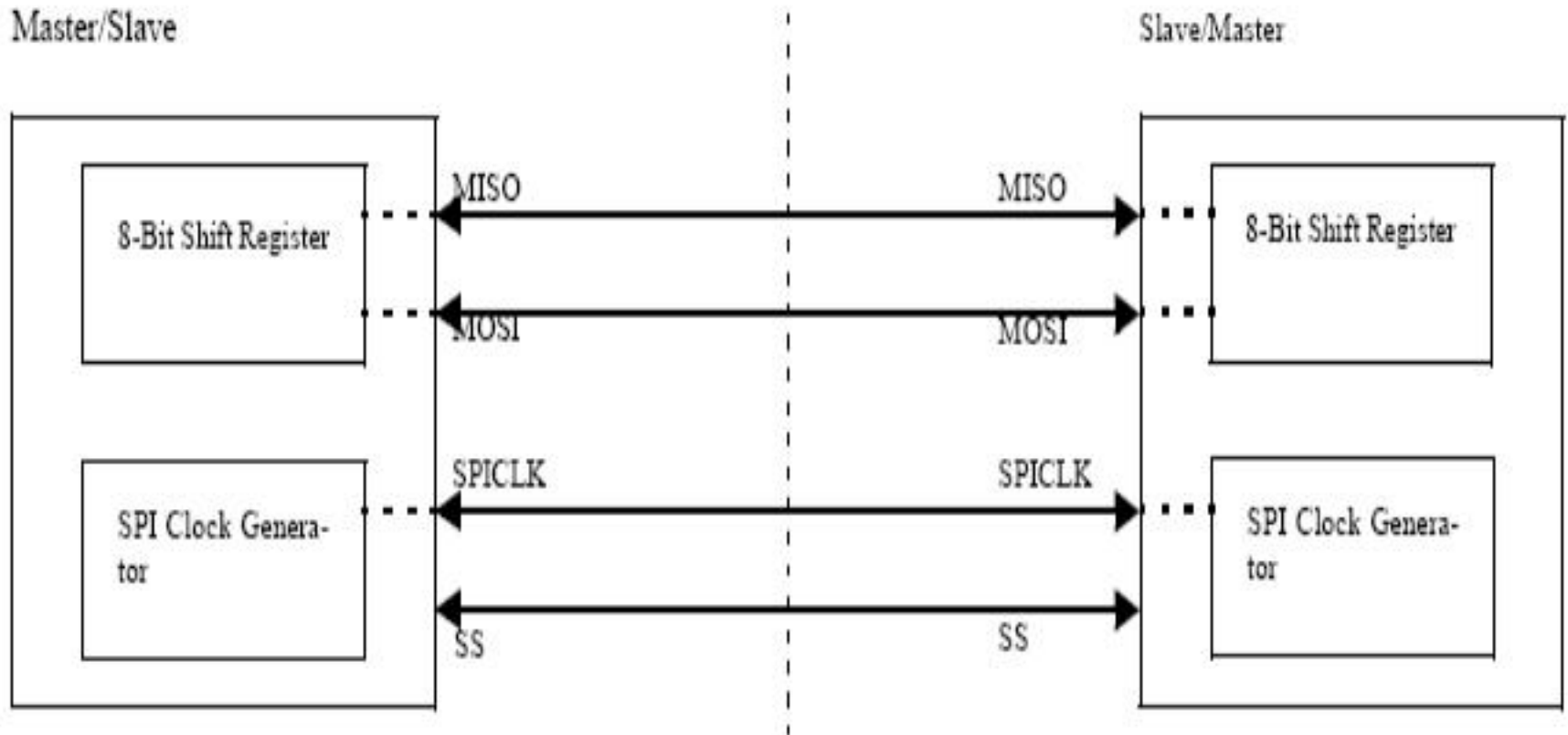
SPI系统总线一般用到三根公共时钟数据线和若干从设备选择线。

SPI接口一般是四根线与外设连接。



LPC900系列单片机引脚分布图

SPI可以有三种通信方式：单主单从、单主多从、互为主从。



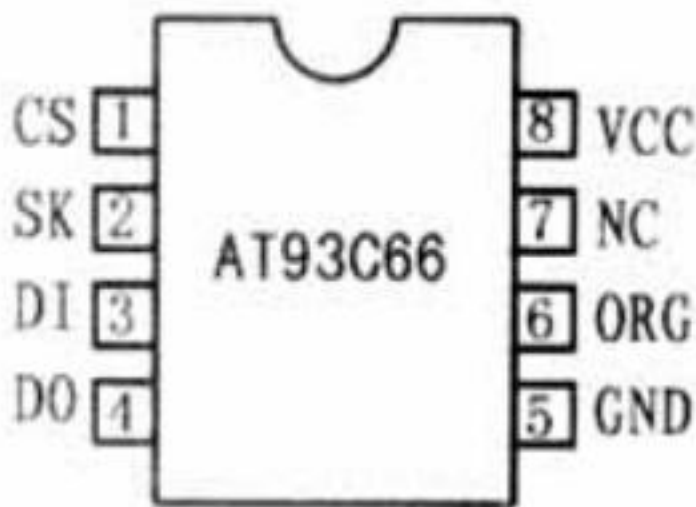
SPI互为主从模式原理图

9.1.2 串行扩展的特点

(3) Microware串行扩展接口

三线制Microware同步串行总线由NationalSemicondutor生产。

由一根数据输出（S0）线、一根数据输入（SI）线和一根时钟（SK）线组成。所有从器件的时钟线连接到同一根SK线上，主器件向SK线发送时钟脉冲信号，从器件在时钟信号的同步沿输出/输入数据。主器件的数据输出线S0和所有从器件的数据输入线相接，从器件的数据输出线都接到主器件的数据输入线SI上。



CS: 片选信号 SK: 串行时钟输入

DI: 数据输入 DO: 数据输出

ORG: 接VCC时16位为一个单元

接GND时8位为一个单元

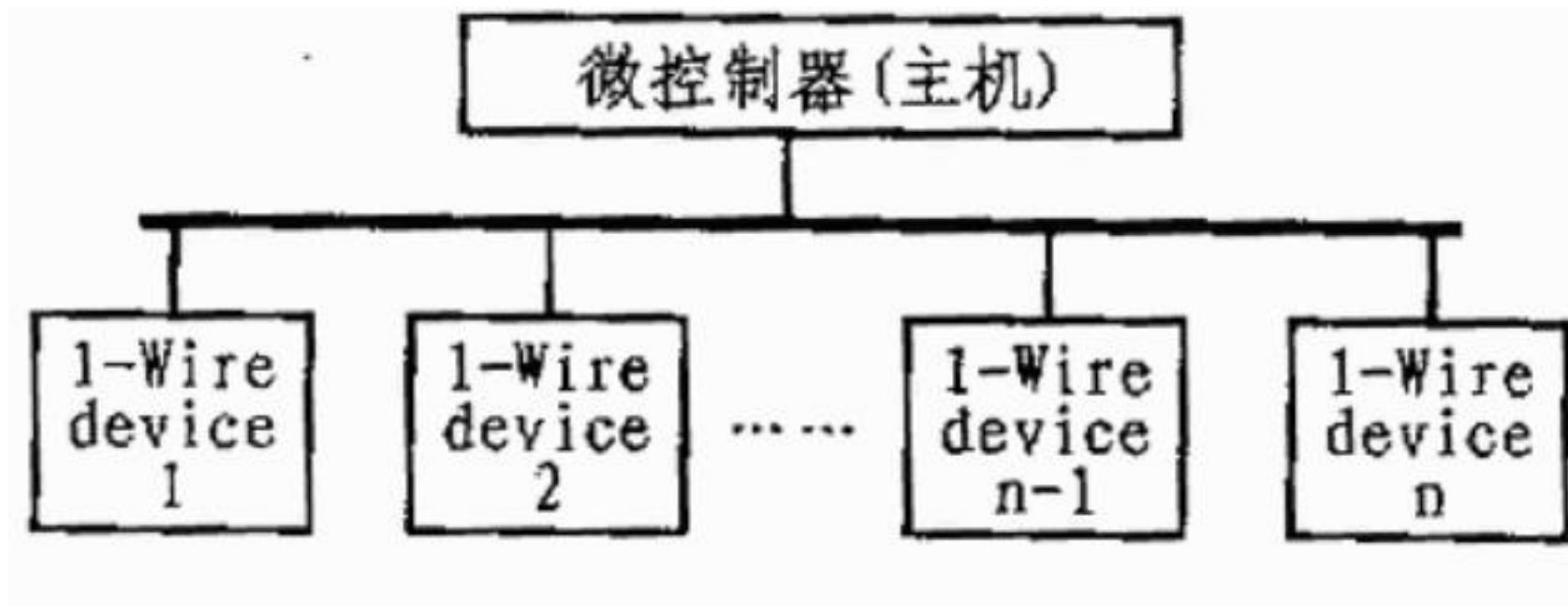
9.1.2 串行扩展的特点

(4) 1-wire总线

近年来，美国的达拉斯半导体公司（**DALLAS SEMICONDUCTOR**）推出了一项特有的单总线（**1-Wire Bus**）技术。它采用单根信号线，既可传输时钟，又能传输数据，而且数据传输是双向的，因而这种单总线技术具有线路简单，硬件开销少，成本低廉，便于总线扩展和维护等优点。

1-wire总线是利用一根线实现双向通信，由一个总线主节点、一个或多个从节点组成系统，通过一根信号线对从芯片进行数据的读取。每一个符合**1-wire**协议的从芯片都有一个唯一的地址，包括8位分类码、48位的序列号和8位**CRC**代码。主芯片对各个从芯片的寻找依据这64位的不同来进行。单总线节省I/O引脚资源、结构简单、成本低廉、便于总线扩展和维护。

单总线适用于单主机系统，能够控制一个或多个从机设备。主机可以是微控制器，从机可以是单总线器件，它们之间的数据交换只通过一条信号线。当只有一个从机设备时，系统可接单节点系统操作；当有多个从设备时，系统则按多节点系统操作。

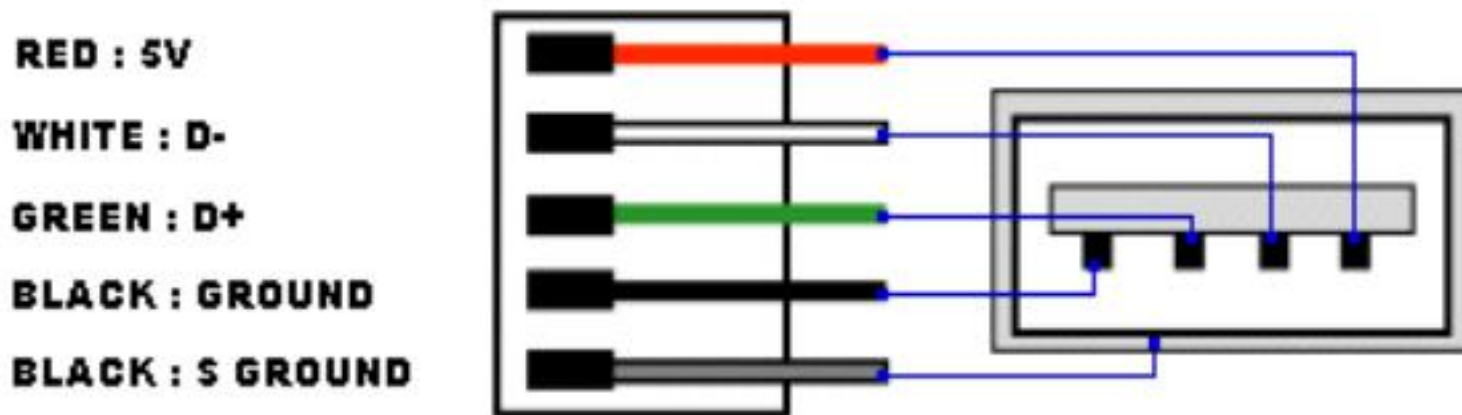


单主机多节点示意图

9.1.2 串行扩展的特点

(5) USB串行扩展接口

USB比较于其他传统接口的一个优势是即插即用的实现，即插即用（Plug-and-Play、PNP）也称为热插拔（Hot Plugging）。数据传输速度快，USB1.1接口的最高传输率可达12 Mb/s；USB2.0接口的最高传输率可达480 Mb/s。扩展方便，使用USB Hub扩展，可以连接127个USB设备，连接的方式十分灵活。



红色：电源+（+5v）白色：数据-绿色：数据+（+3.3v）
黑色：电源-（VBUS）灰色：USB外壳接地

USB接口规范

USB是一种统一的传输规范，但是接口有许多种，最常见的就是咱们电脑上用的那种了，扁平的，这叫做A型，里面有四根连线，根据主动被动关系分为正负接口，一般线上带的是正口，机器上带的是负口。



USBA型正口示意图

另外一种常见的接口就是**B**型口了，通常出现在扫描仪等设备上：



A型正口

B型正口

接下来就是在数码产品上常见的接口了，由于数码产品体积所限，所以通常用的是MINI B型接口，但是MINI B型接口有许多种：

Sony Cameras

Olympus Cameras

Canon Cameras

HP Cameras

SAMSUNG Cameras



MINI B型 4pin :

HP

Olympus

KODAK

SONY

RCA

EPSON

LEIXA

SAMSUNG



外设上常
常见到的
接口

MINI B型 4pin :

常见于富士机器
FUJI Models

FINEPIX 6900 Z
FINEPIX 4700 Z
FINEPIX 3800 Z
FINEPIX 2900
FINEPIX 2800 Z
FINEPIX 2400 Z
FINEPIX 2400
FINEPIX 1400 Z
FINEPIX 1300
FINEPIX 50i
FINEPIX 30i
FINEPIX A101
FINEPIX A200
FINEPIX A303
FINEPIX S602 Z



○

○

○

外设上常
常见到的
接口

MINI B型 8pin:

常见于nikon机型



外设上常
常见到的
接口

9.1.2 串行扩展的特点

(6) CAN总线

CAN（控制器局域网），全称为“**Controller Area Network**”，是应用最广泛的现场总线之一。20 世纪80 年代初，德国 **Bosch** 公司为了解决汽车中众多的控制与测试仪表之间的数据交换问题，开发出 **CAN** 总线。

CAN 总线能有效支持分布式控制或实时控制的串行通信网络，具有抗干扰性强和使用可靠等优点，最初主要应用汽车工业，现在广泛应用于汽车工业、航空工业和工业控制等自动化领域，如分布式环境监测系统、温室环境监控系统、变电站变监测系统。

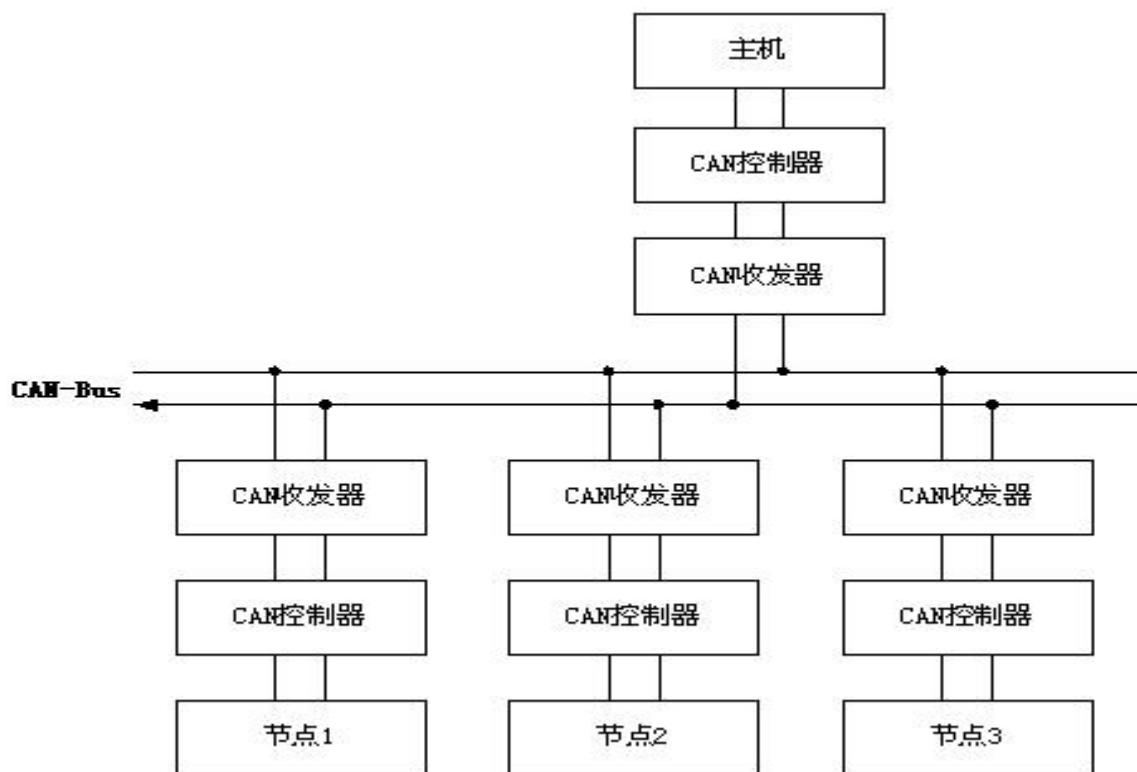
CAN 协议的一个最大特点是废除“站地址编码”，而代之以对通信数据块编码，采用这种方法可使网络内节点个数在理论上不受限制，还可使不同的节点同时收到相同的数据。

在短距离（**40m**） （**1Mbit/s**）

最大距离 **10000m** （**5kbits/s**）

极适合在高速的工业自控应用上

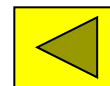
CAN 总线可在同一网络上连接多种不同功能的传感器（如位置,温度或压力）。



CAN总线应用系统

9.2 UART串行扩展接口

- 9.2.1 串行口工作方式
- 9.2.2 UART串行扩展应用实例



UART串行扩展接口

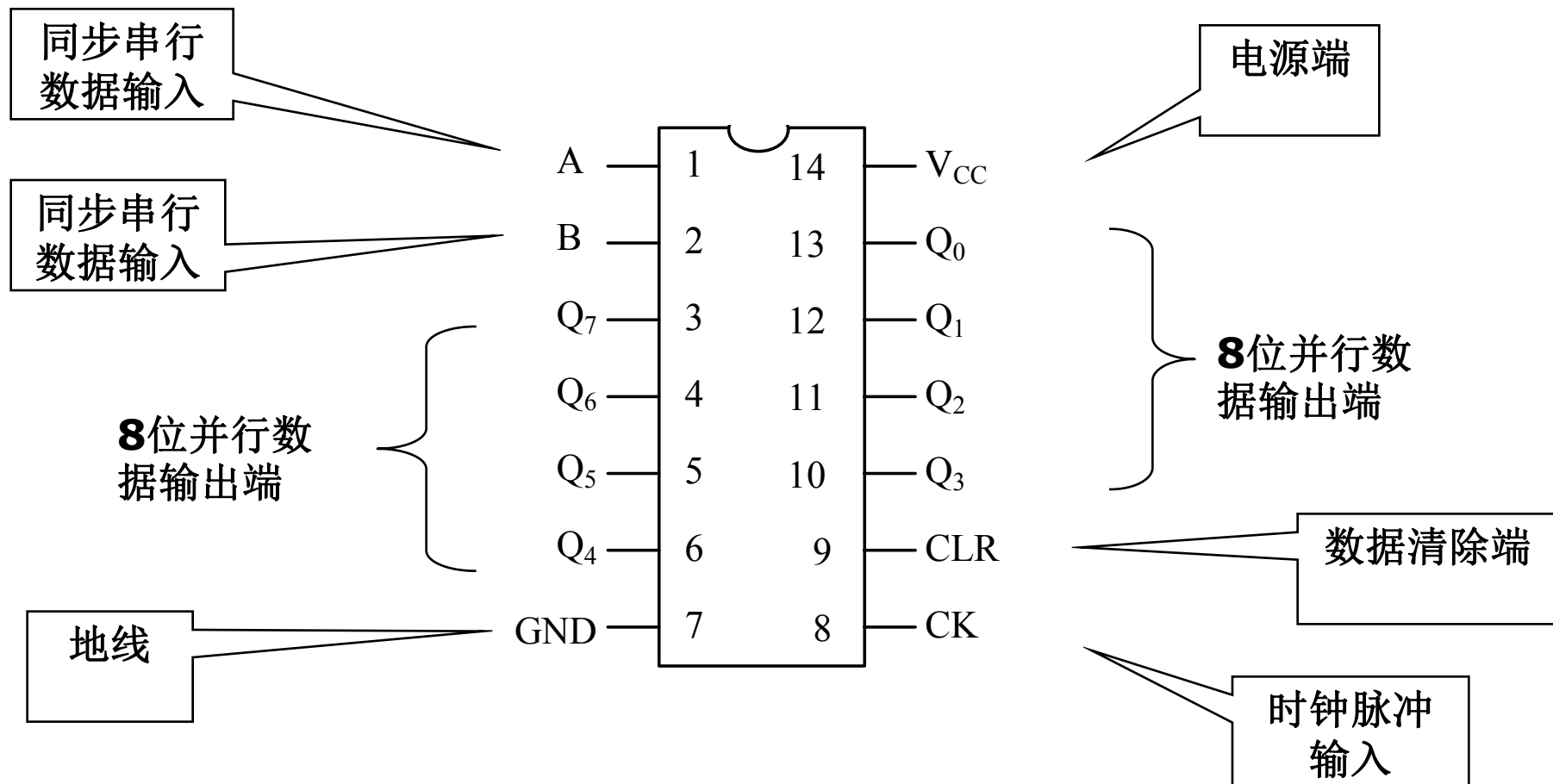
UART (Universal Asynchronous Receiver/Transmitter)

UART通用异步收发器，既能同步又能异步通信的硬件电路称为UASRT。UART是用于控制计算机与串行设备的芯片，它提供了RS-232C数据终端设备接口，这样计算机就可以和调制解调器或其它使用RS-232C接口的串行设备通信了。

9.2.1 串行口工作方式

- 串行口有四种工作方式。其中方式0是8位同步通信方式，用于串/并或并/串转换中，常用的串/并转换芯片有74LS164和并/串转换芯片74LS165等。
- 74LS164 是一个双列直插式8位串入/并出移位寄存器，其引脚如下图所示。其引脚定义如下：
 - A: 同步串行数据输入端
 - B: 同步串行数据输入端
 - $Q_0 \sim Q_7$: 8位并行数据输出端
 - CK: 时钟脉冲输入端
 - CLR: 数据清除端(清除输出数据，通常用在移位完成时)
 - GND: 接地端
 - V_{CC} : 电源端

74LS164引脚图：



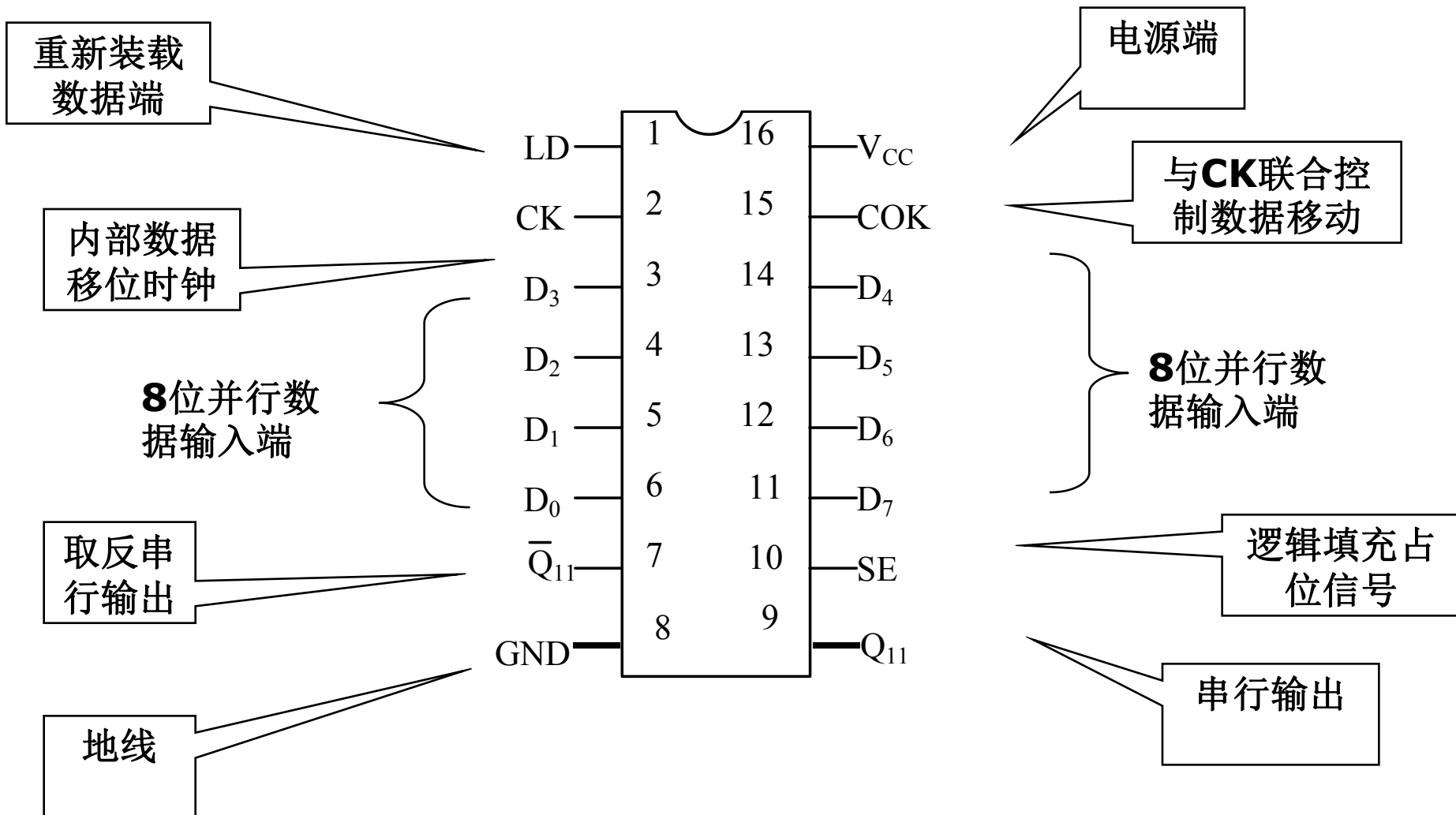
9.2.1 串行口工作方式

- **74LS165** 是一个双列直插式8位并入/串出移位寄存器，其引脚如下图所示。

其引脚定义如下：

- **LD**: 重新装载数据端(通常用在数据完全移出后)
- **CK**: 内部数据移位时钟脉冲输入端
- **D₀~D₇**: 并行数据输入端
- **Q₁₁**: 取反串行输出端
- **GND**: 接地端
- **Q₁₁**: 串行输出端
- **SE**: 用于填充数据移出后的空位的逻辑电平信号
- **COK**: 和**CK**联合控制数据移动
- **V_{CC}**: 电源端

74LS165引脚图：



9.2.2 UART串行扩展应用实例

【例1】 利用**74LS164**串行输入并行输出芯片作一个简单的电子钟，要求四个数码管显示时钟；其中**LED1**显示小时的十位，**LED2**显示小时的个位，**LED3**显示分钟的十位，**LED4**显示分钟的个位。

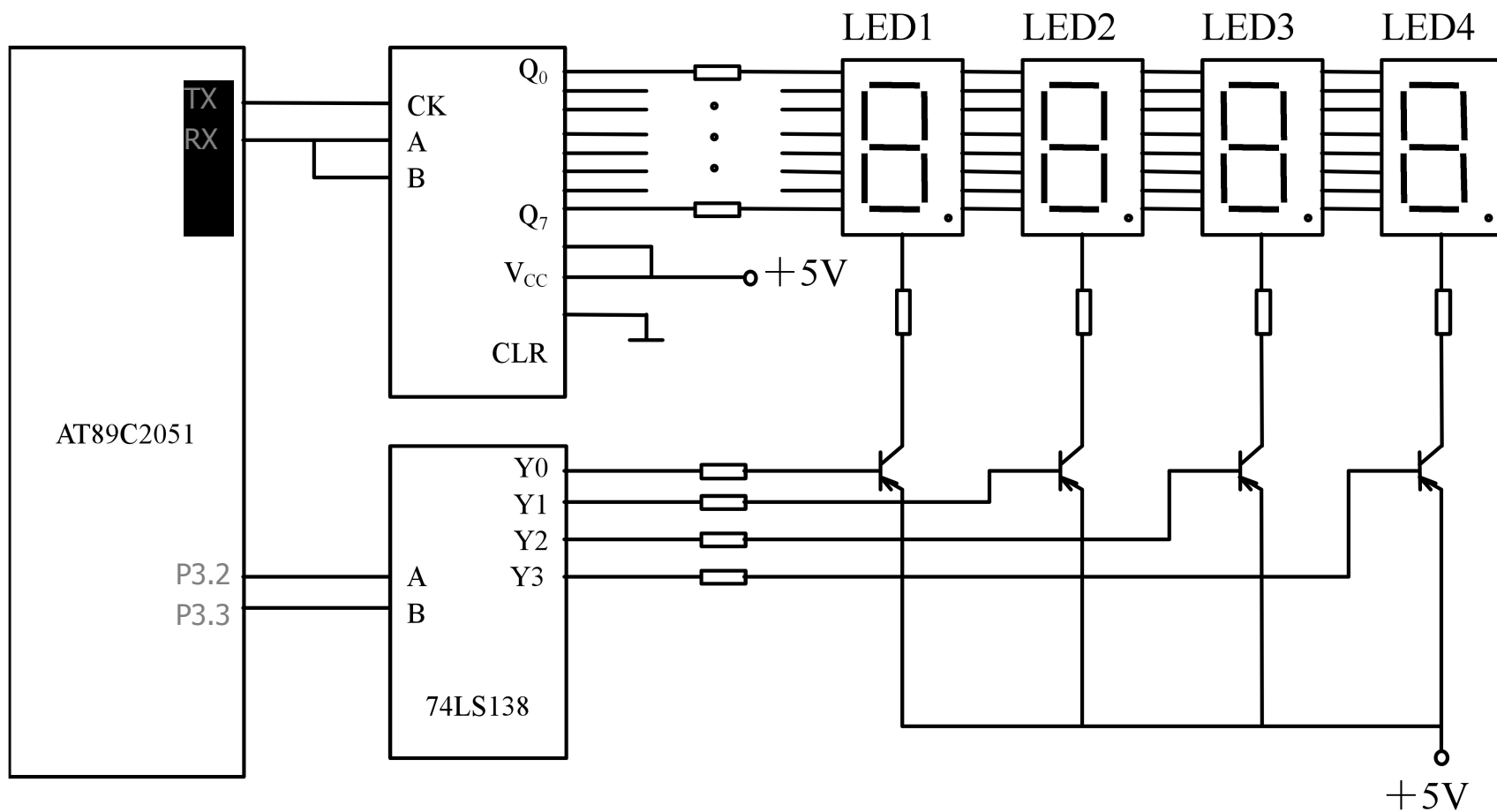
解：原理图如下图，采用单片机的串行口输出字形码，用**74LS164**和**74LS138**作为扩展芯片。

74LS164的功能是将**AT89C2051**串行通信口输出的串行数据译码并在其并口线上输出，从而驱动**LED**数码管。

74LS138是一个3线-8线译码器，它将单片机输出的地址信号译码后动态驱动相应的**LED**。因**74LS138**电流驱动能力较小，故用末级驱动三极管**9013**作为地址驱动。将4只**LED**的字段位都连在一起，它们的公共端则由**74LS138**分时选通，这样任何一个时刻，都只有一位**LED**在点亮，也即动态扫描显示方式，其优点使用串行口进行**LED**通信程序编写相当简单，用户只需将需显示的数据直接送串口发送缓冲器，等待串行中断即可。

9.2.2 UART串行扩展应用实例

串行动态LED扫描电路



9.2.2 UART串行扩展应用实例

ORG 0000H

LJMP MAIN

ORG 0040H

MAIN: MOV SCON, #00H ; 初始化串口为方式0

MOV R3, #00H

LOOP: MOV R4, #0E8H

DELAY: LCALL DISPLAY ; 动态扫描显示

DJNZ R4, DELAY

INC R3 ; 显示数字增1

CJNE R3, #0AH, LOOP ; 不等于10转移

LJMP MAIN

9.2.2 UART串行扩展应用实例

参考程序:

```
DISPLAY:  CLR P3.2                ; 显示LED1
          CLR P3.3
          LCALL DISP
          LCALL DELAY1
          SETB P3.2                ; 显示LED2
          LCALL DISP
          LCALL DELAY1
          SETB P3.3                ; 显示LED3
          CLR P3.2
          LCALL DISP
          LCALL DELAY1
          SETB P3.2                ; 显示LED4
          SETB P3.3
          LCALL DISP
          LCALL DELAY1
          RET
```

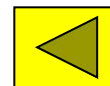
9.2.2 UART串行扩展应用实例

参考程序:

```
DISP:    MOV A, R3                                ; 将字形码送串口
          MOV DPTR, #TABLE
          MOVC A, @A+DPTR
          MOV SBUF, A
WAIT:     JNB TI, WAIT                            ; 等待串口传送
          CLR TI
          RET
DELAY1:   MOV R6, #10H                            ; 动态扫描的时间
LOOP1:    MOV R7, #38H                             ; 间隔
          DJNZ R7, $
          DJNZ R6, LOOP1
          RET
TABLE:    DB 3FH, 06H,, 5BH                        ; 0~9的字形码
          DB 4FH, 66H, 6DH
          DB 7DH, 07H,, 7FH,, 6FH
```

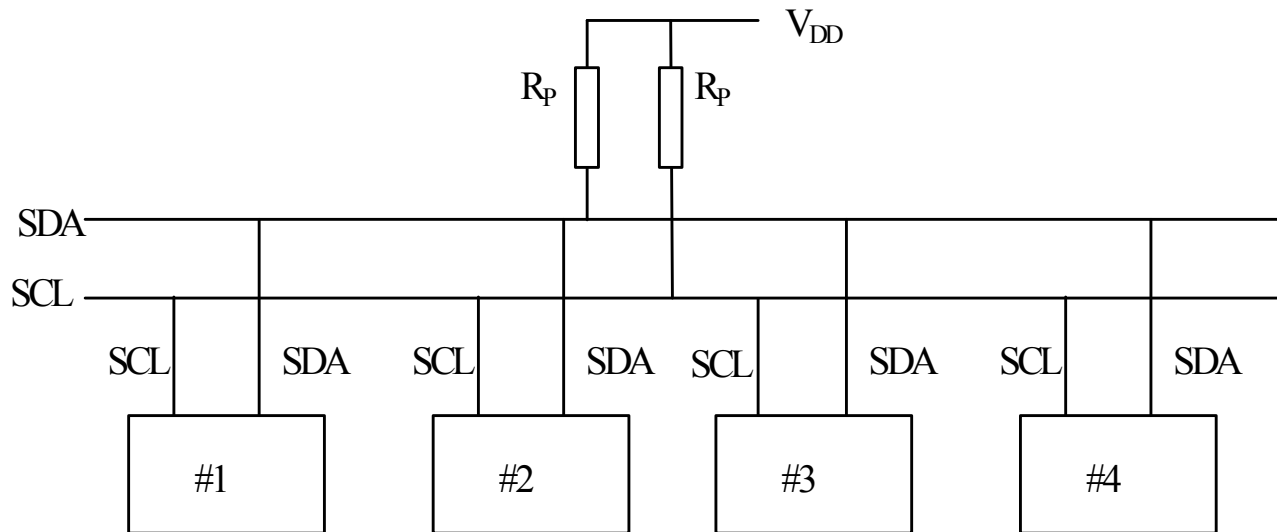
9.3 I²C串行扩展总线

- 9.3.1 I²C总线的结构原理
- 9.3.2 I²C总线的软件模拟
- 9.3.3 I²C串行扩展应用实例



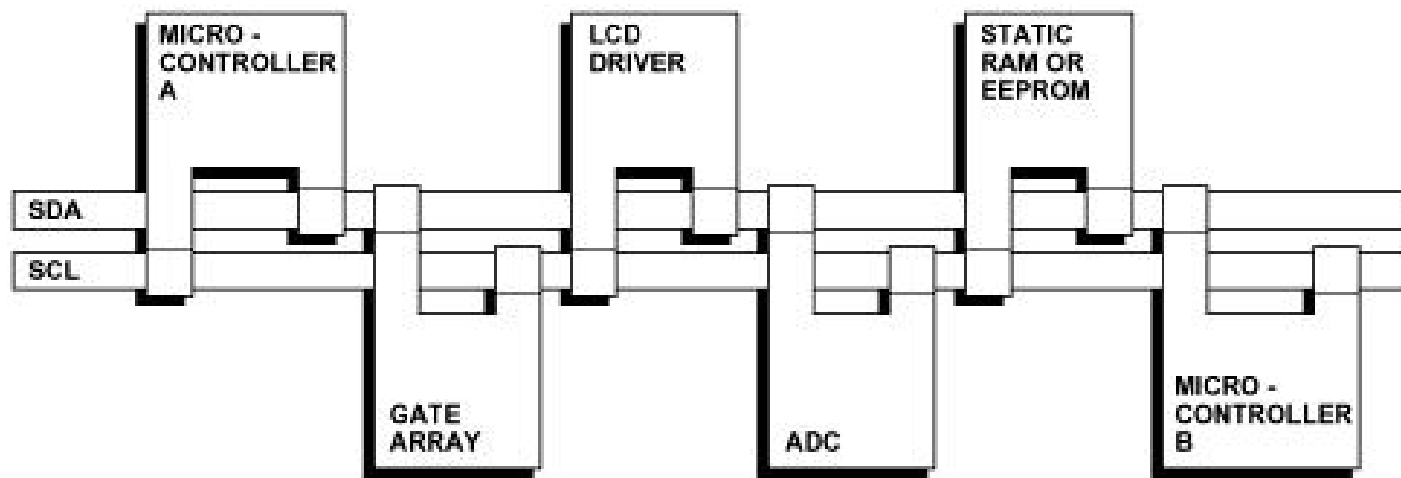
9.3.1 I²C总线的结构原理

- I²C总线是**芯片间串行传输总线**。它用**数据线SDA**和**时钟线SCL**两根线实现**全双工同步数据**传送，可方便地构成多机系统和外围器件扩展系统。



I²C总线接口电路结构

- **I²C**总线采用了**器件地址的硬件设置**方法，通过**软件寻址**完全避免了器件的片选线寻址方法，系统可简单扩展。



I²C结构原理图

9.3.1 I²C总线的结构原理

每个器件都有一个**唯一的地址**，而且可以是单接收器件(如**LCD**驱动器)或者可以是接收或发送器件(如存储器)。收发器可在主从模式下工作，这取决于芯片是否启动数据的传输还是仅仅被寻址。

按照I²C总线规范，总线传输中的所有状态都生成相对应的**状态码**，系统中的主机能够依照这些状态码**自动地进行总线管理**，用户只要在程序中装入这些标准处理模块，根据数据操作要求完成I²C总线的初始化，启动I²C总线，就能自动完成规定的的数据传送操作。

9.3.1 I²C总线的结构原理

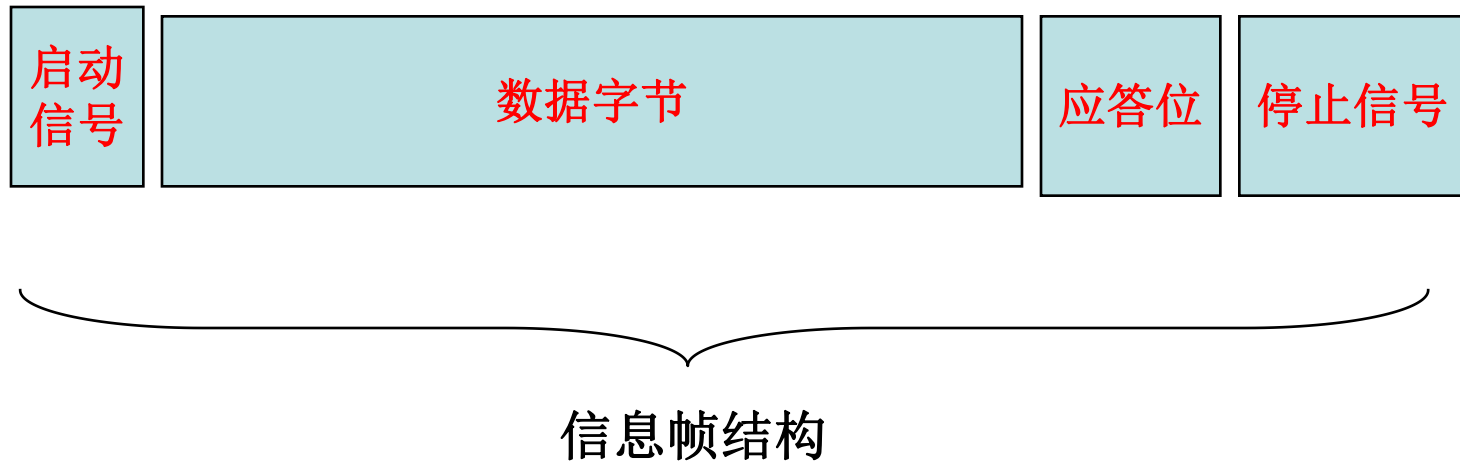
- I²C总线接口为开漏或开集电极输出，需加上拉电阻。
- 各器件数据线SDA和时钟线SCL的同名端相连在一起，各节点有对应地址。
- 在I²C总线上可以挂接各种类型的外围器件，如RAM/EEPROM、日历/时钟芯片、A/D转换器、D/A转换器、以及由I/O口、显示驱动器构成的各种模块。

9.3.1 I²C总线的结构原理

- SDA 和SCL 都是双向线路，都通过一个电流源或上拉电阻连接到正的电源电压；当总线空闲时，这两条线路都是高电平；连接到总线的器件输出级必须是漏极开路或集电极开路才能执行线与的功能。
- I²C 总线上数据的传输速率
 - 在标准模式下可达100kbit/s。 7位寻址
 - 在快速模式下可达400kbit/s。 10位寻址
 - 在高速模式下可达3.4Mbit/s 。 保持向下兼容
- 连接到总线的接口数量只由总线电容是400pF 的限制决定关于高速模式主机器件的信息。

9.3.1 I²C总线的结构原理

- I²C总线上数据传送的基本单位为字节，采用低位在前的格式。主从器件之间一次传输的数据称为一帧，由启动信号、若干个数据字节和应答位以及停止信号组成。



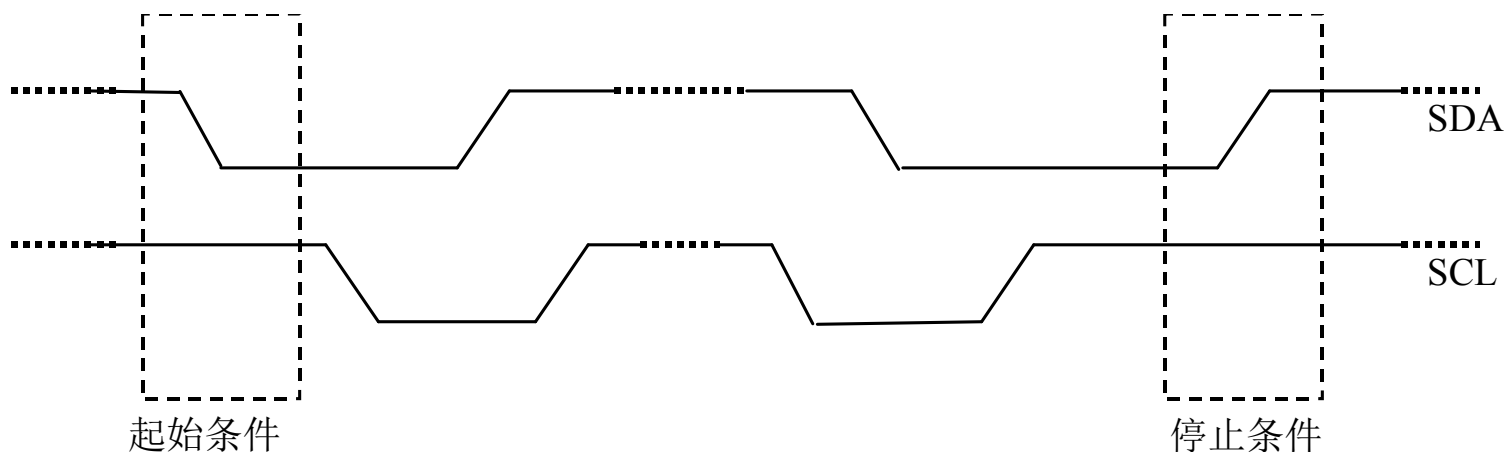
9.3.1 I²C总线的结构原理

- I²C总线最显著的特点是**规范的完整性、结构的独立性和用户使用时的“傻瓜”化。**
 - 1.** 电器特性、时序信号、信号定义、总线状态、总线规则管理
 - 2.** 各器件模块具有独立地址编号
 - 3.** 应用程序的支持

9.3.2 I²C总线的软件模拟

1. 产生起始位和停止位

I²C总线的起始和停止条件如下图所示。



分别用P1.0和P1.1模拟I²C总线的时钟线 and 数据线，则可给时钟线SCL和数据线SDA赋值。程序如下：

```
SDA EQU P1.0
```

```
SCL EQU P1.1
```

9.3.2 I²C总线的软件模拟

2. 发送起始条件

当时钟**SCL**为高电平时，数据线**SDA**从高电平向低电平切换表示起始条件，即启动I²C总线数据传送。模拟时序产生时钟**SCL**和**SDA**发送的起始条件子程序如下：

使用不同频率的晶体振荡器，则要相应增删程序段中**NOP**指令的条数，以满足时序的要求。

```
START:  SETB SDA
        SETB SCL
        NOP                               ; NOP的数目根据时钟频率确定
        NOP                               ; 此处用NOP来延时
        CLR SDA
        NOP
        NOP
        CLR SCL
        RET
```

9.3.2 I²C总线的软件模拟

3. 发送停止条件

当时钟SCL为高电平时，数据线SDA由低电平向高电平切换表示停止条件，即停止I²C总线数据传送。模拟时序产生时钟SCL和SDA发送的停止条件子程序如下：

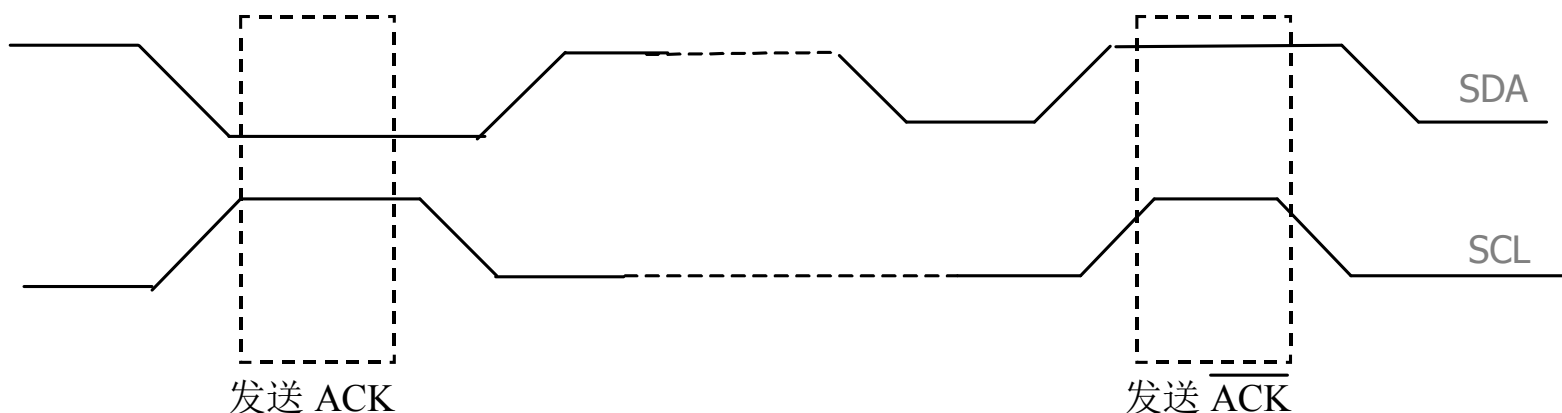
使用不同频率的晶体振荡器，则要相应增删程序段中NOP指令的条数，以满足时序的要求。

```
STOP:  CLR  SDA
        SETB SCL
        NOP                               ; NOP的数目根据时钟频率确定
        NOP                               ; 此处用NOP来延时
        SETB SDA
        NOP
        NOP
        ;CLR  SDA
        CLR  SCL
        RET
```

9.3.2 I²C总线的软件模拟

4. 发送应答位、非应答位子程序

I²C总线上的第9个时钟脉冲对应于应答位，当该位为低电平表示应答ACK，当该位为高电平表示非应答 $\overline{\text{ACK}}$ 。I²C总线的应答位和非应答位如下图所示。发送ACK和 $\overline{\text{ACK}}$ 子程序如下。



9.3.2 I²C总线的软件模拟

(1) 发送应答位ACK

由上图可知，当发送应答位ACK时，只需将SDA设为低电平、SCL设为高电平，其区间的长短根据晶振的频率确定，即改变下列子程序中的NOP数即可达到要求，发送ACK子程序如下：

```
YACK:  CLR SDA
```

```
        SETB SCL
```

```
        NOP
```

； NOP的数目根据时钟频率确定

```
        NOP
```

； 此处用NOP来延时

```
        CLR SCL
```

```
        SETB SDA
```

```
        RET
```

9.3.2 I²C总线的软件模拟

(2) 发送非应答位

由上图可知，当发送非应答位 $\overline{\text{ACK}}$ 时，只需将SDA设为高电平、SCL设为高电平，其区间的长短根据晶振的频率确定，即改变下列子程序中的NOP数即可达到要求，发送 $\overline{\text{ACK}}$ 子程序如下：

```
YNACK:  SETB SDA
```

```
        SETB SCL
```

```
        NOP                                ; NOP的数目根据时钟频率确定
```

```
        NOP                                ; 此处用NOP来延时
```

```
        CLR SCL
```

```
        CLR SDA
```

```
        RET
```

9.3.2 I²C总线的软件模拟

5. 应答位检查子程序

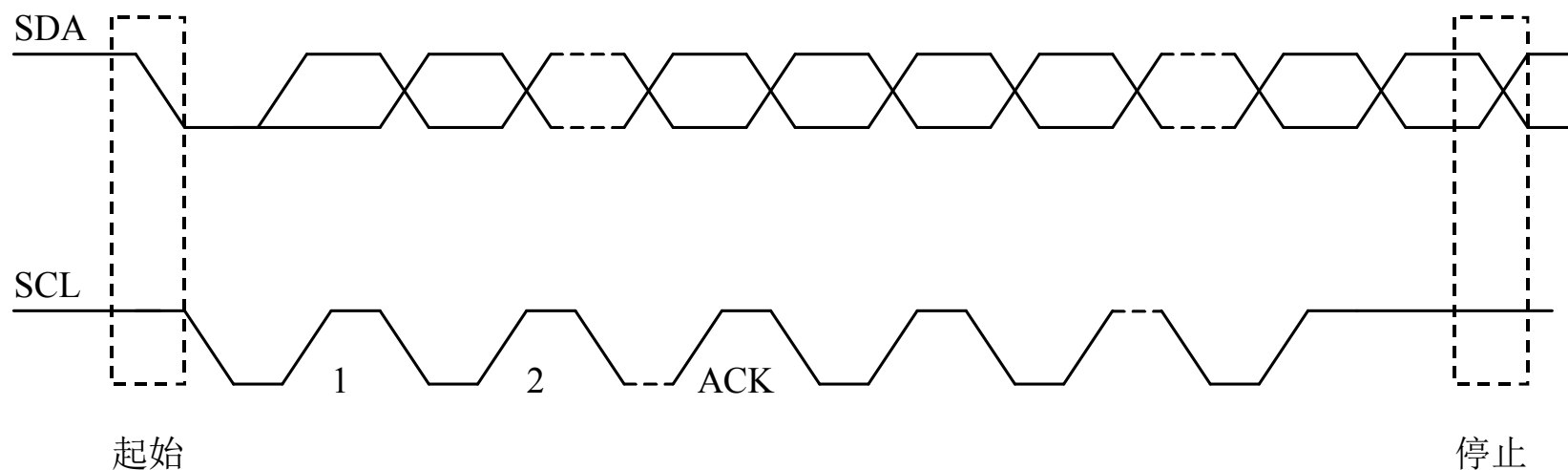
根据I²C总线协议，在接收到一个字节后，要发送一个应答位以供检查，此时可设置一个标志位表示应答状态。当检查结果为正常应答时，则标志位置0；否则，标志位置1。应答位检查子程序如下：

TACK:	SETB SDA	； 设置SDA为输入方式
	SETB SCL	； 产生第9个时钟脉冲
	NOP	； NOP的数目根据时钟频率确定
	NOP	； 此处用NOP来延时
	CLR 30H	
	MOV C, SDA	
	JNC TEND	； 若正常应答，则转移
	SETB 30H	
TEND:	CLR SCL	
	RET	

9.3.2 I²C总线的软件模拟

6. 字节数据发送子程序

根据I²C总线协议，数据在时钟线为低电平时变化，高电平时稳定，每一个时钟脉冲传送一位。模拟I²C总线的SDA接在并行口线，并口中无移位寄存器，要通过指令完成移位后再从SDA串行输出。I²C总线的字节数据传送如下图所示。将待发送的字节存于累加器A中，字节数据发送子程序如下：



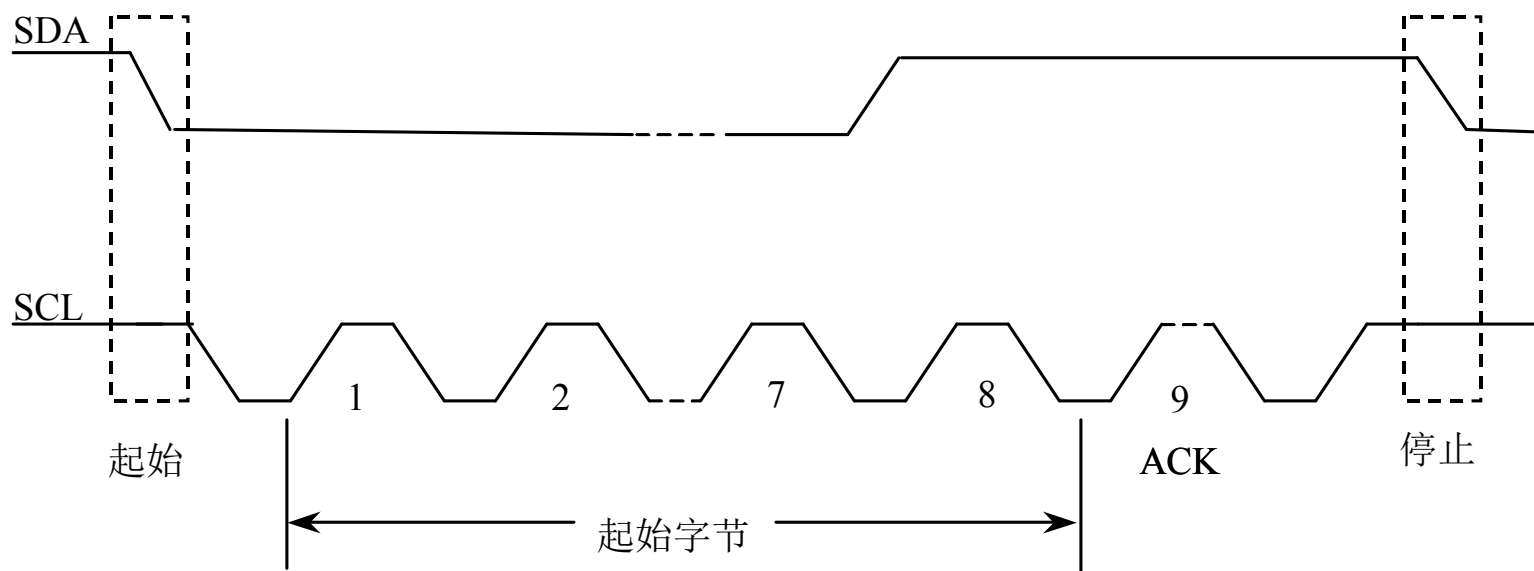
9. 3. 2 I²C总线的软件模拟

WOBYT:	MOV R4, #08H	； 要发送的数据长度为8位
LOOP2:	RLC A	； 将待发送的位送入位CY中
	JC LOOP1	； CY位为1转到LOOP1
	CLR SDA	； 置为低电平， 发送0
	SETB SCL	； 时钟高电平数据保持数据稳定
	NOP	； 调节NOP的个数， 使延时≥4.7u s
	NOP	
	CLR SCL	； 时钟脉冲变为低电平， 准备改变数据
	DJNZ R4, LOOP2	； 若8位未发送完， 转LOOP2继续
	RET	； 8位发送完成返回
LOOP1:	SETB SDA	； 置为高电平， 发送1
	SETB SCL	； 时钟为高电平保持数据稳定
	NOP	； 调节NOP的个数， 使延时≥4.7u s
	NOP	
	CLR SCL	； 时钟脉冲变为低电平， 准备改变数据
	CLR SDA	； 将数据改为低电平
	DJNZ R4, LOOP2	； 若8位未发送完， 转LOOP2继续
	RET	； 8位发送完成返回

9.3.2 I²C总线的软件模拟

7. 字节数据接收子程序

根据I²C总线协议，数据必须在时钟处于高电平期间，数据稳定时才能读取数据，在经过八次的时钟转换后，读出八位即一个字节数据。I²C总线起始字节如下图所示。将读出的字节数据存于R5中，接收字节数据子程序如下：



9.3.2 I²C总线的软件模拟

ROBYT:	MOV R4, #08H	； 要接收的数据长度为8位
LOOP3:	SETB SDA	； 设置SDA为输入方式
	SETB SCL	； SDA上数据有效
	MOV C, SDA	； 读入SDA引脚状态
	MOV A, R5	； 将存放结果送入A中
	RLC A	； 将读出的1位移入A中
	MOV R5, A	
	CLR SCL	； 一个脉冲结束，SDA上数据无效
	DJNZ R4, LOOP3	； 未读完8位，转到LOOP3
	RET	； 读完返回

9.3.2 I²C总线的软件模拟

8. 多个字节数据发送子程序

在完成上述模拟子程序后，根据I²C总线协议，可编写多个字节数据发送子程序。要发送的字节数存放在R5中，要发送的数据块的第一个数据的地址为40H，外围器件的地址存放在60H中，参考程序如下：

WDBYT:	LCALL START	; 起始
WLP1:	MOV A, 60H	; 设60H为存放的控制字
	LCALL WOBYT	; 发送读控制字节
	LCALL TACK	; 检查应答位
	JB 30H, WLP1	; 无应答位，重发
	MOV R0, #40H	; 40H为第一个数据的首地址
WLP2:	MOV A, @R0	; 读一个字节数据
	LCALL WOBYT	; 发送
	LCALL TACK	; 检查应答位
	JB 30H, WLP1	; 无应答位，重发
	INC R0	; 指向下一个要发送的数据
	DJNZ R5, WLP2	; 要发送数据未发完，再发送
	LCALL STOP	; 全部数据发完，停止
	RET	; 返回

9.3.2 I²C总线的软件模拟

9. 多个字节数据接收子程序

根据I²C总线协议，同样可编写多个字节数据接收子程序。要接收的字节数存放在R5中，要接收的数据块存放的首地址为50H，外围器件的地址存放在60H中，参考程序如下：

RDBYT:	LCALL START	； 起始
RLP1:	MOV A, 60H	； 设60H为存放的控制字
	LCALL WOBYT	； 发送读控制字节
	LCALL TACK	； 检查应答位
	JB 30H, RLP1	； 无应答位，重发
	MOV R0, #50H	； 50H为第一个数据的首地址
RLP2:	LCALL ROBYT	； 读入一个字节
	MOV @R0, A	
	DJNZ R5, RLP3	； 要接收数据未读完，再读入
	LCALL YNACK	； 发送非应答位
	LCALL STOP	； 全部数据发完，停止
	RET	； 返回
RLP3:	LCALL YACK	； 发送应答位
	INC R0	； 指向下一个要存放数据的地址
	SJMP RLP2	； 调节NOP的个数，使延时≥4.7u s

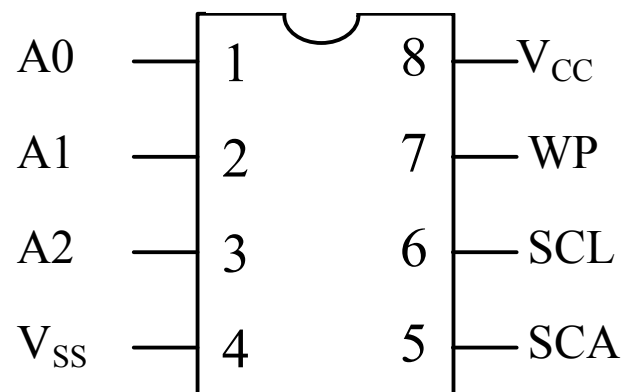
9.3.3 I²C串行扩展应用实例

【例2】 将AT89C2051片内RAM40H~47H单元中的八个8位数据通过I²C总线接口传送到存储器AT24C01的50H~57H单元中。

解：AT24C01是Atmel公司生产的具有I²C总线接口的E²PROM，是目前应用较多的AT24C××系列存储器中的一种，芯片存储容量为1Kb(128×8位)。AT24C01引脚如下图所示。

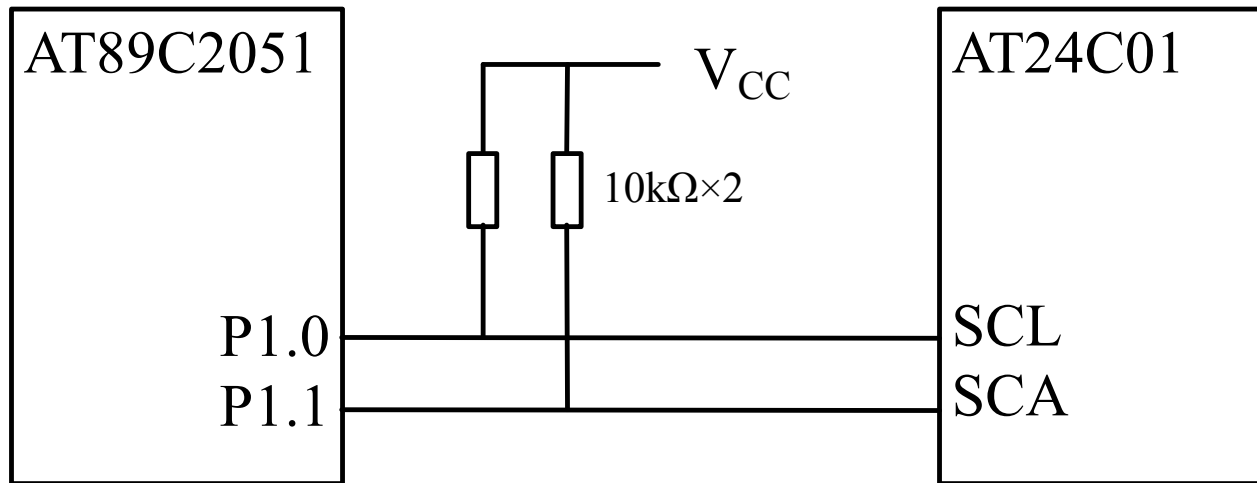
其引脚定义如下：

- **SCL**：串行时钟端。
- **SDA**：串行数据端，漏极开路，需接上拉电阻到V_{CC}。
- **WP**：写保护，
当WP为高电平时存储器只读；
当WP为低电平时存储器可读可写。
- **A2~A0**：芯片地址。



9.3.3 I²C串行扩展应用实例

AT89C2051与AT24C01通过串行总线接口传送数据的接线如下图所示。因AT89C2051没有I²C总线接口，可用软件的办法来模拟I²C总线操作。P1.0模拟I²C总线的时钟线SCL，P1.1模拟I²C总线的数据线SDA。



AT89C2051与AT24C01的接口接线

9.3.3 I²C串行扩展应用实例

AT24C01的读写操作有多种形式，写操作有两种类型：字节写和页面写；读操作有三种类型：读当前地址内容、读随机地址内容和读顺序地址内容。

AT24C01写N个字节数据的操作时序如下图所示，

8 位	8 位	8 位	8 位	8 位	8 位	8 位	8 位	8 位	8 位	8 位	8 位	8 位
起始信号	器件地址	响应信号	字地址	响应信号	数据 1	响应信号	数据 2	响应信号	...	数据 N	响应信号	结束信号
			写操作		写操作		写操作			写操作		

AT24C01读N个字节数据的操作时序如下图所示。

↑	8 位↑	↑	8 位↑	↑	↑	8 位↑	↑	8 位↑	↑	↑	8 位↑	↑	↑
起↑ 始↑ 信↑ 号↑	器件↑ 地址↑	响↑ 应↑ 信↑ 号↑	字地址↑	响↑ 应↑ 信↑ 号↑	起↑ 始↑ 信↑ 号↑	器件↑ 地址↑	响↑ 应↑ 信↑ 号↑	数据 1↑	响↑ 应↑ 信↑ 号↑	…↑	数据 N↑	响↑ 应↑ 信↑ 号↑	结↑ 束↑ 信↑ 号↑
↑	↑	↑	写操作↑	↑	↑	↑	↑	读操作↑	↑	↑	读操作↑	↑	↑

9.3.3 I²C串行扩展应用实例

上图中器件地址说明如下：

- ① 器件地址码的第7~4位——从器件地址位，用于确认器件的类型。
AT24C01的器件地址码为1010，1010 表示从器件为串行E²PROM。
- ② 器件地址码的第3~1位——1~8片的片选或存储器内的页面地址选择位。此三个控制位用于选片或者内部页面选择。
如在存储容量8Kb (1024×8位)的AT24C08内部，存储矩阵分为4个页面，每一页面有256个字节。通过器件地址码的第2位和第1位，可以选择数据读写的页面。
- ③ 器件地址码的第0位——读、写(R/W)操作控制码。若此位为1，下一字节进行读操作(R)；此位为0，下一字节进行写操作(W)。

9.3.3 I²C串行扩展应用实例

编程说明：

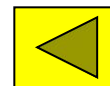
- AT24C01每接收一个字节后，都必须发送一个确认应答信号位ACK，即时序中的响应信号。此时AT89C2051必须产生一个与此确认位相应的时钟脉冲。
- AT24C01在读写操作时，具有地址自动加1功能，即读、写完某一地址空间后，会自动指向下一个地址单元。
- 将AT89C2051片内RAM40H~47H单元中的八个8位数据通过I²C总线接口传送到存储器AT24C01的50H~57H单元中，参考子程序如下。
- 程序中用到的子程序均为9.3.2节所编写的模拟I²C总线的子程序。

参考子程序

ICW:	LCALL START	; 发送起始信号
ICWLP1:	MOV A, #0A0H	; #10100000B器件地址码,
	LCALL WOBYT	; 发送器件地址
	LCALL TACK	; 检查应答位
	JB 30H, ICWLP1	; 无应答位, 重发
ICWLP2:	MOV A, #50H	; 50H为待写存储单元首地址
	LCALL WOBYT	; 发送待写存储单元地址
	LCALL TACK	; 检查应答位
	JB 30H, ICWLP1	; 无应答位, 重发
	MOV R5, #8	; 待发送数据块的长度
	MOV R0, #40H	; 40H为第一个数据的首地址
ICWLP3:	MOV A, @R0	; 读一个字节数据
	LCALL WOBYT	; 发送
	LCALL TACK	; 检查应答位
	JB 30H, ICWLP1	; 无应答位, 重发
	INC R0	; 指向下一个要发送的数据
	DJNZ R5, ICWLP3	; 要发送数据未发完, 再发送
	LCALL STOP	; 全部数据发完, 停止
	LCALL DELAY	; 延时, 等待AT24C01内部写操作
	RET	; 返回

9.4 SPI 串行扩展接口

- 9.4.1 SPI 总线的结构原理
- 9.4.2 SPI 总线的软件模拟
- 9.4.3 SPI 串行扩展应用实例

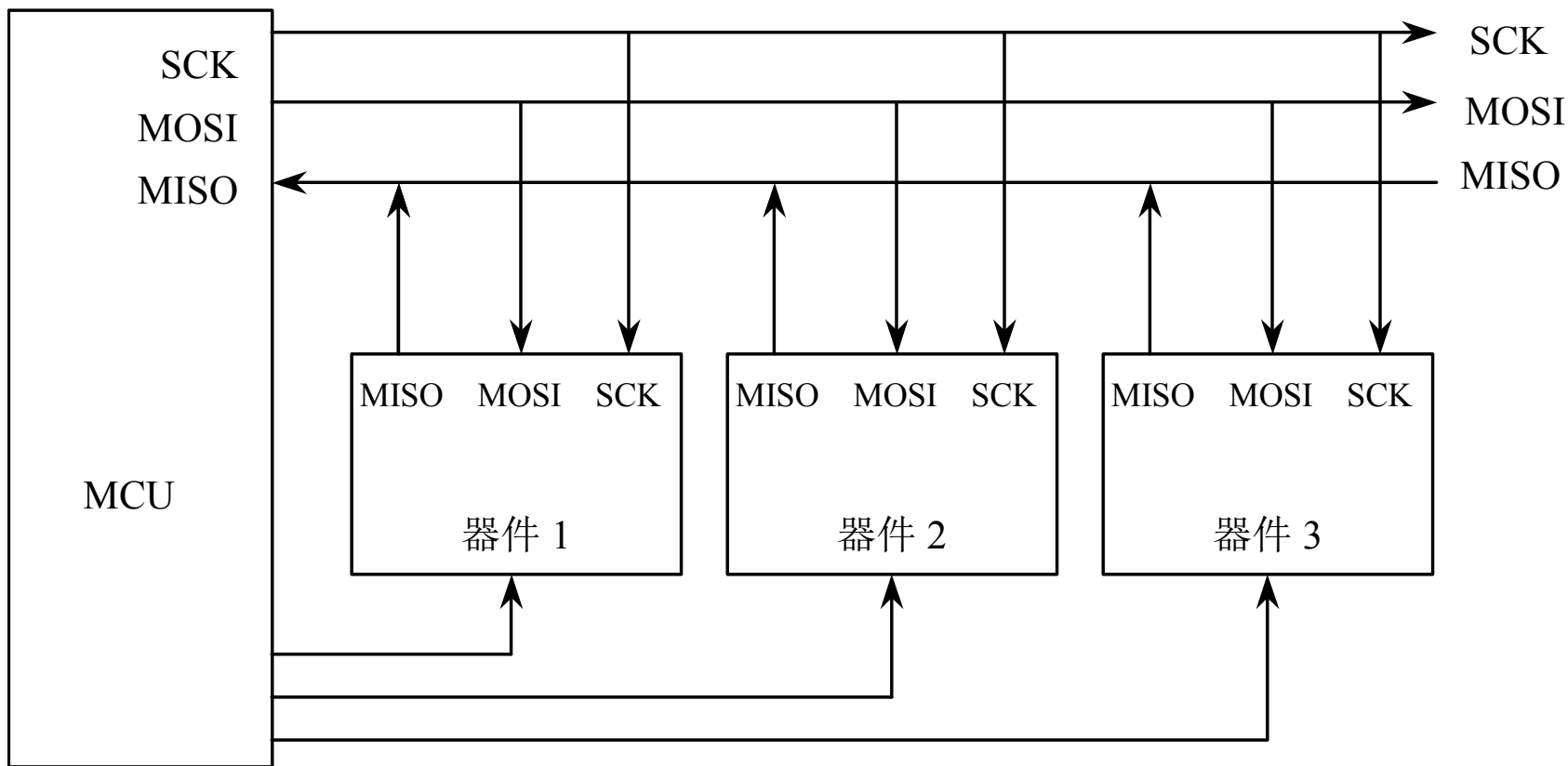


9.4.1 SPI总线的结构原理

- SPI总线系统是一种**同步串行外设接口**，允许MCU与各种外围设备以同步串行方式进行通信来交换信息。
- SPI总线接口一般使用4根线：**串行时钟线SCK、主机输入/从机输出数据线MISO、主机输出/从机输入数据线MOSI和低电平有效的从机选择线SS。**
- 由于SPI系统总线只需3根公共的时钟数据线和若干位独立的从机选择线，在SPI从设备较少而没有总线扩展能力的单片机系统中使用特别方便。
- SPI总线包括1根串行同步时钟信号线以及2根数据线。SPI总线接口电路结构如下图所示。

9.4.1 SPI 总线的结构原理

SPI 总线接口电路结构



9.4.1 SPI总线的结构原理

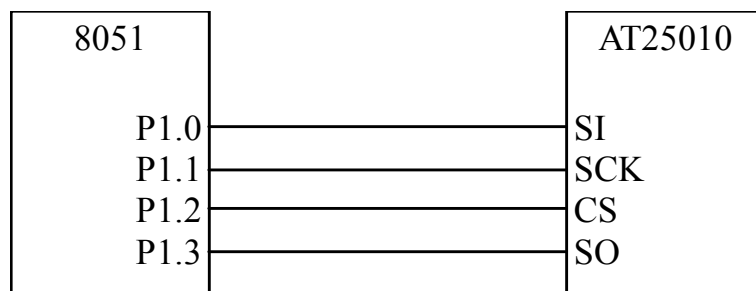
- SPI模块为了和外设进行数据交换，根据外设工作要求，其输出的串行同步时钟极性和相位可以进行配置。
- **时钟极性 (CPOL)** 对传输协议没有重大的影响。
 - 如果CPOL=0，串行同步时钟的空闲状态为低电平；
 - 如果CPOL=1，串行同步时钟的空闲状态为高电平。
- **时钟相位 (CPHA)** 能够用于选择两种不同的传输协议进行数据传输。
 - 如果CPHA=0，在串行同步时钟的第一个跳变沿(上升或下降)数据被采样；
 - 如果CPHA=1，在串行同步时钟的第二个跳变沿(上升或下降)数据被采样。
- SPI主模块和与之通信的外设间时钟相位和极性应该一致。

9.4.1 SPI总线的结构原理

- **特点：**由于SPI系统总线一共只需3~4位数据线和控制线即可实现与具有SPI总线接口功能的各种I/O器件进行连接，而扩展并行总线则需要8根数据线、8~16位地址线、2~3位控制线，因此，采用SPI总线接口可以简化电路设计，节省很多常规电路中的接口器件和I/O口线，提高设计的可靠性。
- **应用：**在MCS - 51系列等不具有SPI接口的单片机组成的智能仪器和工业测控系统中，当传输速度要求不是太高时，使用SPI总线可以增加应用系统接口器件的种类，提高应用系统的性能。

9.4.2 SPI总线的软件模拟

- 对于没有提供**SPI**接口的单片机而言，通常可使用软件的办法来模拟**SPI**的总线操作，包括串行时钟、数据输入和输出。
- **Atmel**公司生产的**EEPROM**具有**SPI**接口，存储容量为**4Kb**的**AT25040**就有**SPI**接口，**MCS - 51**系列单片机与**AT25010**的**SPI**总线接口接线如下图所示。



图中，**P1.0**模拟**SPI**的数据输出端(**MOSI**)，**P1.1**模拟**SPI**的**SCK**输出端，**P1.2**模拟**SPI**的从机选择端(**SS**)，**P1.3**模拟**SPI**的数据输入端(**MISO**)。下面给出模拟**SPI**串行输入、串行输出和串行输入/输出的3个子程序。

1. MCU串行输入子程序SPIIN

从AT25040的S0端接收8位数据并放入寄存器R0中。参考程序如下：

```
SPIIN:      SETB   P1.1                ;使P1.1(时钟)输出为1

            CLR    P1.2                ;选择从机

            MOV    R1,#08H            ;置循环次数

SPIIN1:     CLR    P1.1                ;使P1.1(时钟)输出为0

            NOP                     ;延时

            NOP

            MOV    C,P1.3              ;从机输出送进位C

            RLC    A                  ;左移至累加器A

            SETB   P1.1                ;使P1.0(时钟)输出为1

            DJNZ   R1,SPIIN1           ;判断是否循环8次 (8位数据)

            MOV    R0,A                ;8位数据送R0

            RET
```

2. MCU串行输出子程序SPIOUT

将MCS - 51单片机中R0寄存器的内容传送到AT25040的SI端。
参考程序如下：

SPIOUT:	SETB P1.1	;使P1.1 (时钟) 输出为1
	CLR P1.2	;选择从机
	MOV R1, #08H	;置循环次数
	MOV A, R0	;8位数据送累加器A
SPIOUT1:	CLR P1.1	;使P1.1 (时钟) 输出为0
	NOP	;延时
	NOP	
	RLC A	;左移至累加器A最高位至C
	MOV P1.0, C	;进位C送从机输入
	SETB P1.1	;使P1.1 (时钟) 输出为1
	DJNZ R1, SPIOUT1	;判是否循环8次 (8位数据)
	RET	

3. MCU串行输入/输出子程序SPIIO

将MCS - 51单片机R0寄存器的内容传送到AT25040的SI端，同时从AT25040的SO端接收8位数据。参考程序如下：

```
SPIIO:      SETB   P1.1           ;使P1.1(时钟)输出为1
            CLR    P1.2           ;选择从机
            MOV    R1,#08H        ;置循环次数
            MOV    A,R0           ;8位数据送累加器A
SPIIO1:     CLR    P1.1           ;使P1.1(时钟)输出为0
            NOP                    ;延时
            NOP
            MOV    C,P1.3         ;从机输出送进位C
            RLC    A              ;左移至累加器A最高位至C
            MOV    P1.0,C         ;进位C送从机输入
            SETB   P1.1           ;使P1.1(时钟)输出为1
            DJNZ   R1,SPIIO1      ;判断是否循环8次(8位数据)
            RET
```

9.4.2 SPI总线的软件模拟

- 这些子程序适用于在串行时钟的上升沿输入和下降沿输出的各种串行外围接口芯片(如D/A和A/D转换芯片、实时时钟芯片、LED显示驱动芯片等)。
- 对于下降沿输入、上升沿输出的各种串行外围接口芯片,只要改变P1.1的输出电平顺序,这些子程序也同样适用。如先置P1.1为低电平,之后再次置P1.1为高电平,再置P1.1为低电平等等。

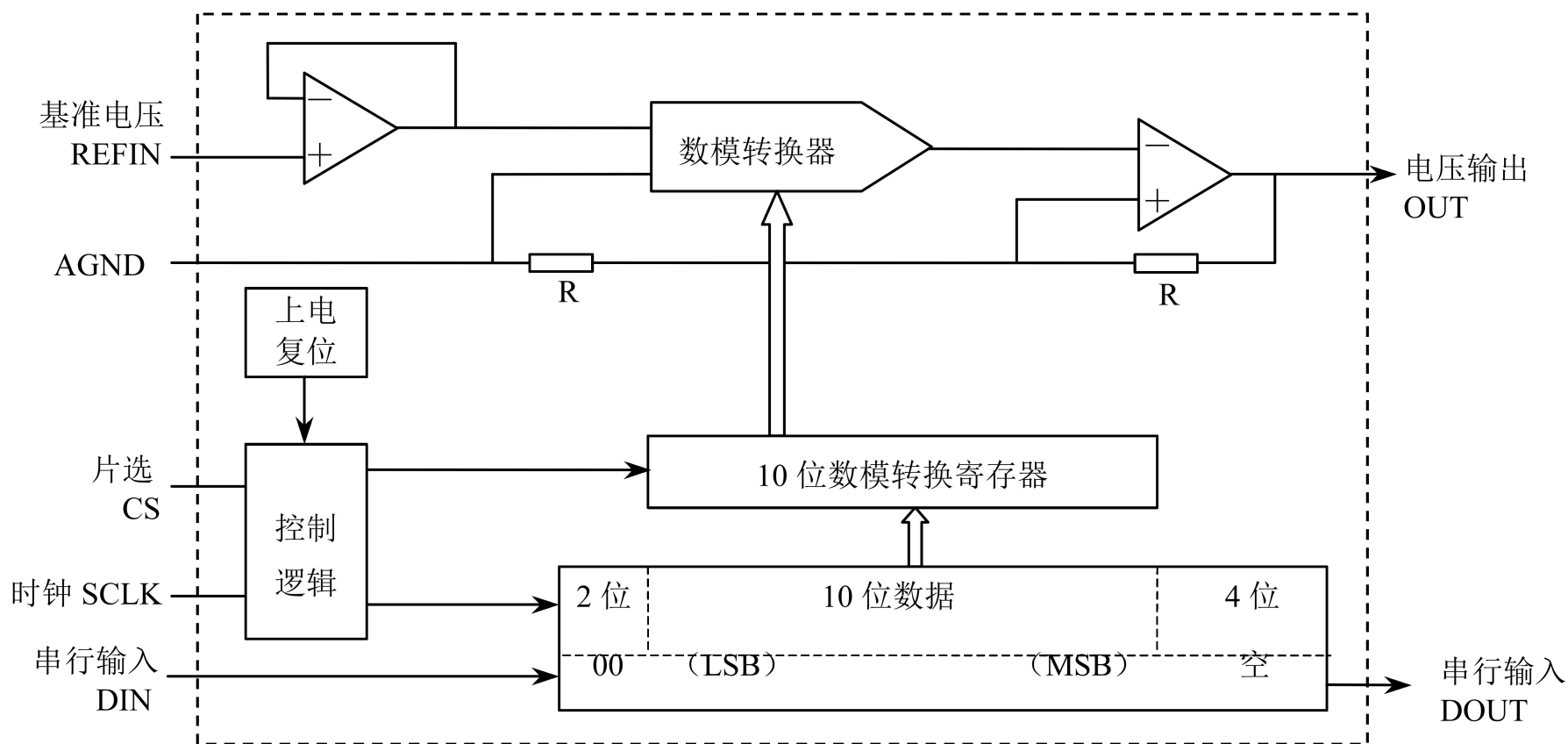
9.4.3 SPI 串行扩展应用实例

【例3】 将AT89C2051片内RAM30H、31H单元中的16位数据通过SPI总线接口传送到数/模转换器TLC5615。

解： TLC5615是3线串行总线接口10位电压输出数/模转换器，它既可与单片机的SPI总线接口相连接，又可与单片机的Microwire总线接口相连接。TLC5615内部结构如下图所示。

9.4.3 SPI 串行扩展应用实例

TLC5615内部结构



9.4.3 SPI 串行扩展应用实例

- TLC5615通过固定增益为2的运放缓冲电阻网络，把10位数字数据转换为模拟电压。在TLC5615芯片上电时，内部电路把D/A寄存器复位为0。其输出具有与基准输入相同的极性，表达式为

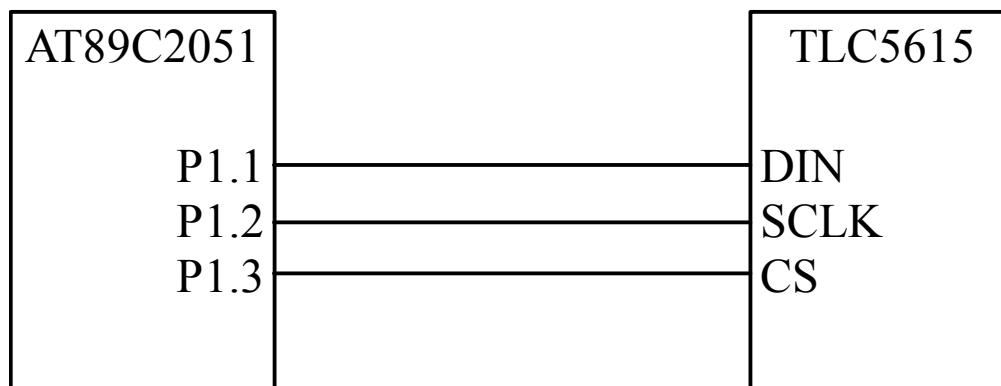
$$V_O = 2 \times \text{REF} \times \text{CODE} / 1024$$

其中，CMOD是通过串行总线接口输入的待转换的数据；REF是基准电压。

- TLC5615最大的串行时钟速率不超过14MHz，10位DAC的建立时间为12.5μs，通常更新速率限制至80kHz以内。
- TLC5615的16位移位寄存器在SCLK的控制下从DIN引脚输入数据，高位在前，低位在后。16位移位寄存器中间的10位数据在上升沿的作用下输入10位的D/A寄存器供给D/A转换。

9.4.3 SPI 串行扩展应用实例

- AT89C2051与TLC5615通过串行总线接口传送8位数据的接线如下图所示。



- 因AT89C2051没有SPI接口，可用软件的办法来模拟SPI的总线操作。P1.1模拟SPI的数据输出端(MOSI)，P1.2模拟SPI的SCK输出端，P1.2模拟SPI的从机选择端(SS)；TLC5615是数/模转换器，不会向AT89C2051发送数据，故不需要模拟SPI的数据输入端(MISO)。

9.4.3 SPI 串行扩展应用实例

AT89C2051将片内RAM30H、31H单元中的16位数据传送到TLC5615的参考程序如下：

```
        DIN  BIT  P1.1                ;定义I/O口
        SCLK  BIT  P1.2
        CS5615 BIT  P1.3
        DataH EQU  30H
        DataL EQU  31H
TLC5615: CLR  SCLK                    ;准备操作TLC5615
          CLR  CS5615                 ;选中TLC5615
          MOV  R7,#08H
          MOV  A, DataH               ;装入高8位数据
LPH:     LCALL DELAY                 ;延时
          RLC  A                      ;最高位移向TLC5615
          MOV  DIN,C
          SETB SCLK                   ;产生上升沿,移入1位数据
          LCALL DELAY
          CLR  SCLK
          DJNZ R7,LPH
```

9.4.3 SPI 串行扩展应用实例

参考程序

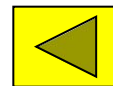
```

MOV    R7, #08H
MOV    A, DataL           ;装入低8位数据
LPL:   LCALL  DELAY       ;延时
      RLC    A            ;最高位移向TLC5615
      MOV    DIN, C
      SETB   SCLK         ;产生上升沿, 移入1位数据
      LCALL  DELAY
      CLR    SCLK
      DJNZ   R7, LPL
      SETB   CS5615       ;结束TLC5615的操作, 同时将转换数据代码存入10位DA寄存器, 启动新一轮的DA转换

      RET
DELAY: ... (略)
      RET
```

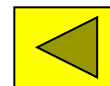
本章小结

- 单片机中目前常用的串行扩展总线和接口可分为以下几类：**UART**串行扩展接口、**I²C**总线、**SPI**总线、**Microware**总线、**1-wire**总线、**USB**总线和**CAN**总线。
- **I²C**总线是芯片间串行传输总线。它用两根线实现全双工同步数据传输，可方便地构成多机系统和外围器件扩展系统。**I²C**总线简单，结构紧凑，易于实现模块化和标准化。
- **SPI**总线是同步串行外设接口。允许**MCU**与各种外围设备以同步串行方式进行通信来交换信息。可直接与多种标准外围器件直接接口，采用**SPI**设备可简化电路设计。
- **模拟I²C**总线的应用程序可使没有**I²C**总线的单片机也能使用**I²C**总线技术。**模拟SPI**总线的应用程序也可使没有**SPI**总线的单片机也能使用**I²C**总线技术。
- **CAN**总线是多主机局域网，是国际上应用最广泛的现场总线之一。
- **USB**总线是计算机串行通信协议。是即插即用的，数据传输速度快，扩展方便，连接的方式十分灵活。
- **Microware**总线是串行同步双工通讯接口。**1-wire**总线也是一种串行通信协议。



习 题

1. 串行扩展与并行扩展相比的主要优点是什么？
2. 简述I²C总线的数据传输方法。
3. 常用模拟I²C总线的应用子程序有哪些？
4. 简述SPI总线的数据传输方法。
5. 常用模拟SPI总线的应用子程序有哪些？



Q & A?

Thanks!

