单片机原理及应用

项目八：串行通信

1. 了解串行通信与并行通信的概定义、数据通信的传输方式；
2. 掌握波特率概念，波特率的计算方法；
3. 掌握特殊功能寄存器 SCON 和 PCON 的 SMOD 位；
4. 掌握串行口的 4 种工作方式；

1. 会阐述串行通信与并行通信的概定义、数据通信的传输方式；
2. 能正确计算串行通信的波特率；
3. 能够正确设置串行通信和寄存器，并使用不同的工作方式进行通信；
4. 会设计单片机与单片机之间的通信。

1. 能分析设计任务，正确使用串行口通信；
2. 能熟练编写单片机串行口通信的发送和接收数据程序；
3. 能完成单片机与单片机间串口通信；
4. 能使用 PROTUES 软件绘制电路原理图；
5. 能使用 Keil 软件编译程序对串行口通信的控制，并与 PROTUES 软件联调，实现控制电路仿真；
6. 能够完成单片机与单片机电路设计和仿真；
7. 能够使用串行口通信去解决实际问题。

4 课时

1.1 任务提出

设计一个单片机双向通信系统，系统有甲单片机和乙单片机，甲单片机和乙单片机都外接八个按键，通过按键输入数值，送入单片机的端口；甲单片机和乙单片机都外接显示器件，用于显示接收到对方发送过来的数值。利用单片机串行口通信，甲单片机可以接收到乙单片机发送过来的数值并显示；乙单片机可以接收到甲单片机发送过来的数值并显示。

1.2 方案设计

1. 单片机选择。采用 ATMEL 公司的 AT89C51 作为系统控制器的 CPU 方案。单片机算术运算功能强，软件编程灵活、自由度大，可以用软件编程实现各种算法和逻辑控制，并且由于其功耗低、体积小、技术成熟和成本低等优点，使其在各个领域应用广泛。

2. 显示电路的选择。根据题目要求，能够直观显示倒计时时间。采用 LED 数码管显示， LED 数码管价格适中，显示控制也方便。

3. 输入数值方法的选择。输入的方式可以使用矩阵键盘输入特定的数值，但是矩阵键盘比较复杂，实现起来较困难，因此输入数值方法采用八个独立式按键电路实现，通过八个按键的闭合断开状态的二进制数值作为输入的数值。

4. 通信的选择。这里采用直接的串行通信，电路方便，只需要两条线即可通信，程序编写也易于控制。

1.3 硬件设计

1. 串行通信

1.1 串行通信和并行通信

所谓“串行通信”是指外设和计算机间使用一根数据信号线（另外也需要地线，可能还需要控制线），数据在一根数据信号线上一位一位地进行传输，每一位数据都占据一个固定的时间长度。在一根传输线上即传送数据又传送联络信号，成本低，但速度比较慢。

所谓的并行通信，就是一组数据的各数据位在多条线上同时被传输。并行通信传输中有多个数据位，同时在两个设备之间传输。接收设备可同时接收到这些数据，不需要做任何变换就可直接使用。并行方式主要用于近距离通信。优点是传输速度快，缺点是数据线多。尤其使用于远距离通信。

1.3 硬件设计

1. 串行通信

1.2 串行通信传送方向

（1）单工方式：数据仅沿着一个固定的方向传送，具有单一方向性。一方只能处于传送状态，另一方只能处于接收状态，典型产品有收音机。

（2）双工方式：数据可以实现双向传送，但这种双方传送不能同时进行，需要借助某种协议实现收发转换，典型产品有对讲机。

（3）全双工方式：允许双方在同一时刻进行数据双向传送，线路和设备比较复杂，典型产品有手机。

（4）多工方式：以上三种传输方式都是同一线路传输一种频率信号，为了充分的利用线路资源，可通过使用多路复用器或多路集线器，采用频分、时分、或码分复用技术，即可实现在同一线路上资源共享功能。

1.3 硬件设计

1. 串行通信

1.3 串行通信方式

1. 同步通信。同步通信时要建立发送方时钟对接收方时钟的直接控制，让双方达到完成同步。同步通信是一种连续串行传输数据的通信方式，期间不允许出现空隙，没有起始位和停止位。具有通信传输效率高的特点。

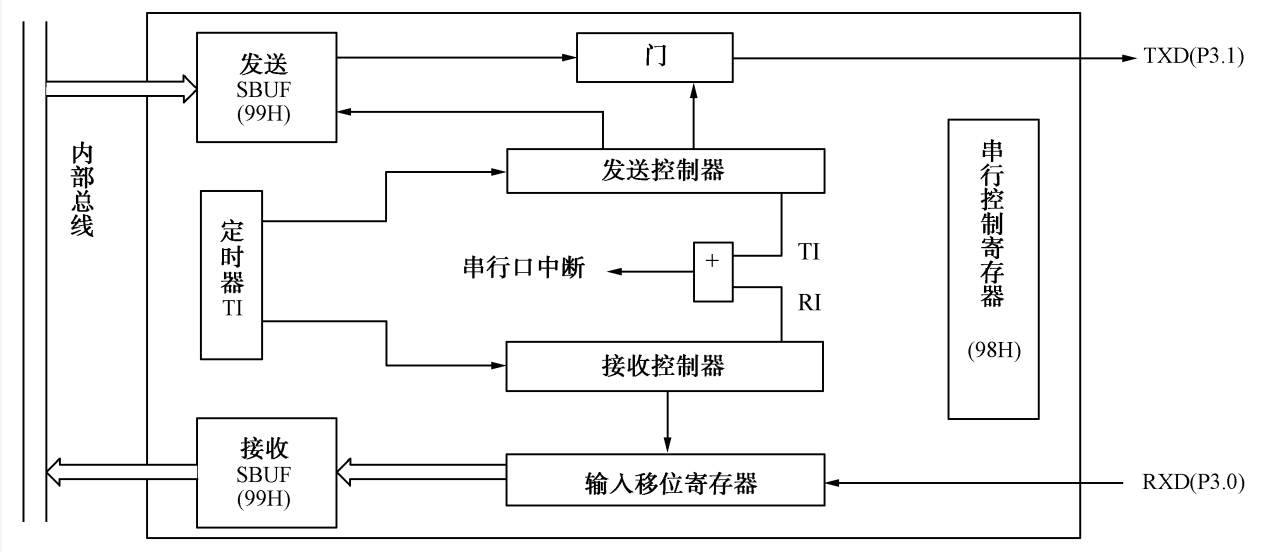
2. 异步通信。在异步通信中，发送端和接收端的工作是非同步的，由各自的时钟来控制，两个时钟可以彼此独立、互不同步。在异步通信方式中，发送方只发送数据帧，不传输时钟，发送和接收双方必须约定相同的传输率。当然双方实际工作速率不可能绝对相等，但是只要误差不超过一定的限度，就不会造成传输出错。发送的每一字符，都必须先按照通信双方约定好的格式进行格式化，否则会造成传输错误。一个字符一个字符地传输，每个字符一位一位地传输，并且传输一个字符时，总是以“起始位”开始，以“停止位”结束，字符之间没有固定的时间间隔要求。

1.3 硬件设计

2. 串行口通信

2.1 串行口结构

51 系列单片机集成一个功能很强的具有可编程的全双工串行通信接口，主要由发送数据缓冲器、发送控制器、接收数据缓冲器、接收控制器、输出控制门、输入移位寄存器等组成，可以接收也可以发射数据，但接收缓冲器只可以读出不能写入，而发送缓冲器只能写入不能读出。通过编程控制实现串口的发送和接收。串行通信数据帧的格式有 8 位、 10 位、 11 位，并能设置各种波特率，使用方便灵活。

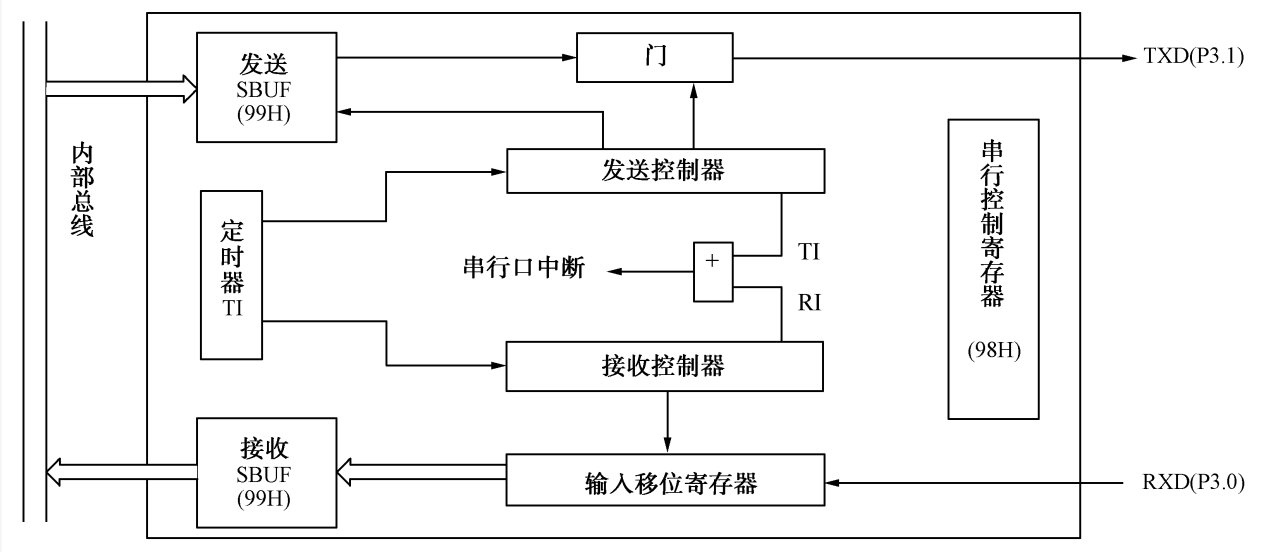


1.3 硬件设计

2. 串行口通信

2.1 串行口结构

由图中知可，串行口有一个接收信号引脚 RXD （ P3.0 ）和一个发送信号引脚 TXD （ P3.1 ），可以同时发送、接收数据，实现全双工通信。发送数据时，将数据按一定的传输格式转换后在 TXD 引脚上按规定的波特率逐位输出；而接收数据时，对引脚 RXD 不断查询，一旦出现起始位 0 ，就将外围设备送来的一定格式的串行数据转换成并行数据，送入单片机处理。



1.3 硬件设计

2. 串行口通信

2.2 串行口特殊寄存器

串行口特殊寄存器主要有串行口数据缓冲寄存器、串行控制寄存器 SCON 和电源控制寄存器 PCON 等三个组成，主要作用是对串行口的访问和设置。

1.3 硬件设计

2.2 串行口特殊寄存器

1．串行数据缓冲寄存器 SBUF

SBUF 是两个在物理上独立的接收、发送寄存器，一它既表示发送寄存器，又表示接收寄存器，可同时发送和接收数据。通过对 SBUF 的读、写指令来区别是对接收缓冲器还是发送缓冲器进行操作。发送缓冲器只能写入不能读出，接收缓冲器只能读出不能写入，两者均只能进行字节寻址。 CPU 在写 SBUF 时，就是修改发送缓冲器；读 SBUF ，就是读接收缓冲器的内容。接收或发送数据，是通过串行口对外的两条独立收发信号线 RXD （ P3.0 ）、 TXD(P3.1) 来实现的，可以同时发送、接收数据，其工作方式为全双工制式。

① 串行口的数据发送

当 TI 为 0 时，将数据送入 SBUF ，启动数据串行发送。这样，在发送时钟控制下，数据将由 TXD 引脚按照规定格式由低到高逐位发送，发送结束 TI 变为 1 。

② 串行口的数据接收

若设置单片机处于接收状态，必需要先设置 REN=1 。然后不断监测 RXD 引脚信号，一旦出现起始位 0 ，在接收时钟的控制下，将采集的数据存入输入移位寄存器，一直等到接收完成或检测到停止位时，再将接收到数据内容送入接收缓冲器 SBUF ，完成接收操作，设置 RI=1 。

1.3 硬件设计

2.2 串行口特殊寄存器

2. 串行控制寄存器（ SCON ）

SCON 控制寄存器是用于设置串行口的工作方式、监视串行口工作状态、发送与接收的状态控制等，是一个既可字节寻址又可位寻址的特殊功能寄存器。

表 8-1 寄存器 SCON 结构

SCON	D7	D6	D5	D4	D3	D2	D1	D0
	SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0、SM1：串行方式选择位，其定义如表 8-2 所示。

表 8-2 串行口方式选择位的定义

SM0 SM1	工 作 方 式	功 能	波 特 率
0 0	方 式 0	8 位同步移位寄存器	$f_{osc}/12$
0 1	方 式1	10 位 UART	可变
1 0	方 式2	11 位 UART	$f_{osc}/64$ 或 $f_{osc}/32$
1 1	方 式 3	11 位 UART	可变

2.2 串行口特殊寄存器

2. 串行控制寄存器（ SCON ）

SM2 ： 多机通信控制位，用于方式 2 和方式 3 中。当处于接收方式状态时，若 SM2=1 ， 且接收到的第 9 位数据 RB8 为 0 时， RI 不激活；若 SM2=1 ， 且 RB8=1 时，才把接收的前 8 位数据送入 SBUF ，且置位 RI 发出中断申请， RI=1 。若 SM2=0 ，不论接收到的第 9 位 RB8 为 0 还是为 1 ， TI 、 RI 都以正常方式被激活。在方式 1 处于接收时，若 SM2=1 ，则只有收到有效的停止位后， RI 置 1 。在方式 0 中， SM2 应为 0 。

REN ： 允许串行接收位。它由软件置位或清 0 。 REN=1 时，允许接收； REN=0 时，禁止接收。

TB8 ： 发送数据的第 9 位。在方式 2 和方式 3 中，由软件置位或复位，可作为奇偶校验位。在多机通信中，可作为区别地址帧或数据帧的标识位，一般约定地址帧时， TB8 为 1 ，约定数据帧时， TB8 为 0 。

RB8 ： 接收数据的第 9 位。其功能同 TB8 。在方式 1 中，若 SM2=0 ，则 RB8 是接收到的停止位。在方式 0 中，该位未用。

TI ： 发送中断标志位。在方式 0 中，发送完 8 位数据后，由硬件置位；在其他方式中，在发送停止位之初由硬件置位。 TI=1 时，也可向 CPU 申请中断。响应中断后，必须由软件清除 TI 。

RI ： 接收中断标志位。在方式 0 中，接收完 8 位数据后，由硬件置位；在其他方式中，在接收停止位的中间由硬件置位。 RI=1 时，也可申请中断。响应中断后，必须由软件清除 RI 。

1.3 硬件设计

2.2 串行口特殊寄存器

3. 电源及波特率选择寄存器 PCON

PCON：主要是 CHMOS 型单片机的电源控制而设置的专用寄存器，其结构格式如下表 8-3 所示。

PCON	D7	D6	D5	D4	D3	D2	D1	D0
	SMOD				GF1	GF0	PD	IDL

SMOD：串行口波特率倍增位。在工作方式 1 ～工作方式 3 时，若 SMOD=1，则串行口波特率增加一倍。若 SMOD=0，波特率不加倍。系统复位时，SMOD=0 。

在 CHMOS 型单片机中，除 SMOD 位外其他位均为虚设的，SMOD 是串行波特率倍增位，当 SMOD=1 时串行口波特率加倍，系统复位默认为 SMOD=0 。

1.3 硬件设计

2.2 串行口特殊寄存器

4. 中断允许寄存器 IE

中断允许寄存器这里重述一下对串行口有影响的位 ES 。 ES 为串行中断允许控制位， ES=1 允许串行中断， ES=0 ，禁止串行中断。

1.3 硬件设计

2.3 串行口工作方式

1．串行工作方式 0

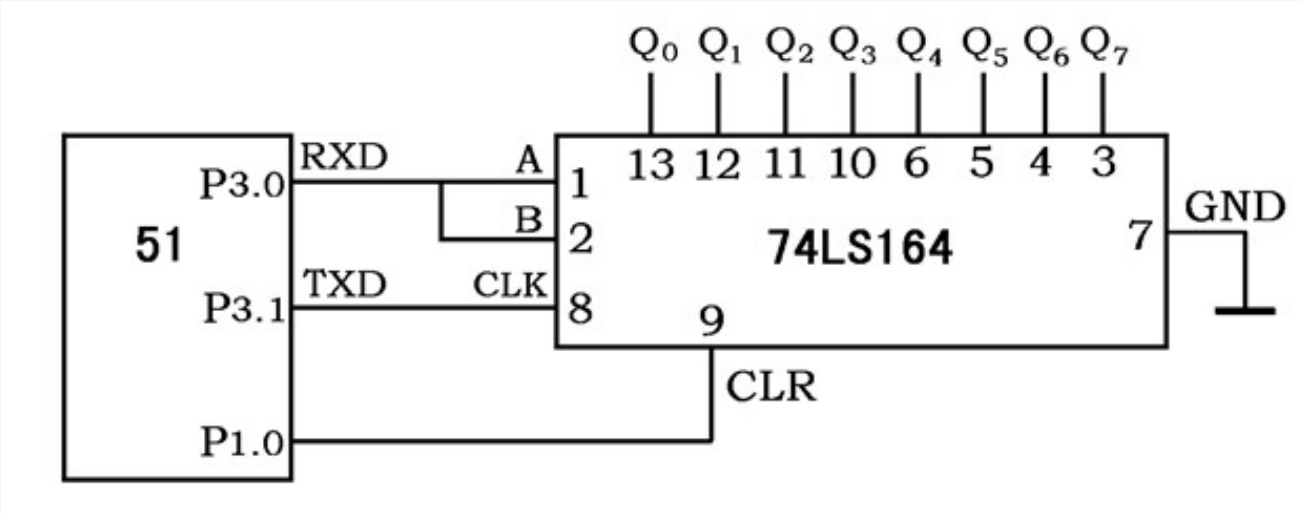
在方式 0 下，串行口用做同步移位寄存器，其波特率固定为 fOSC/12 。数据从 RXD(P3.0) 端输入或输出，从 TXD(P3.1) 送出，一般应用于 I/O 口的扩展，主要有串行口扩展为并行输出口和串行口扩展为并行输入口两种。

1.3 硬件设计

2.3 串行口工作方式

（1）串行口扩展为并行输出口

将数据写入接收缓冲器 SBUF，数据以 fOSC/12 的波特率从 RXD 端串行输出，在 TXD 的脉冲作用下进行的，数据从 74LS164 逐位装入，发送完 8 位数据后，置中断标志 TI 为 1，请求中断。在再次发送数据之前，必须由软件清 TI 为 0。

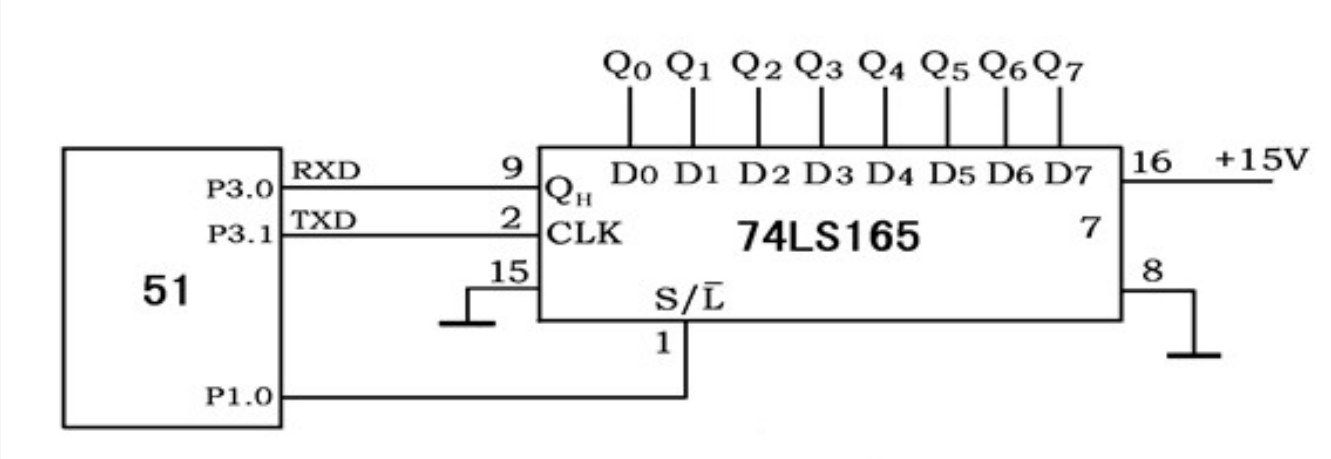


1.3 硬件设计

2.3 串行口工作方式

（2）串行口扩展为并行输入口

首先设置 REN=1 和 RI=0，进入数据接收状态，数据从 RXD 端以 fOSC/12 的波特率输入数据（低位在前）到接收缓冲器 SBUF，在 TXD 的脉冲作用下，接收的数据是从 74LS165 逐位取出，当接收完 8 位数据后，置中断标志 RI 为 1，请求中断。在再次接收数据之前，必须由软件清 RI 为 0。



1.3 硬件设计

2.3 串行口工作方式

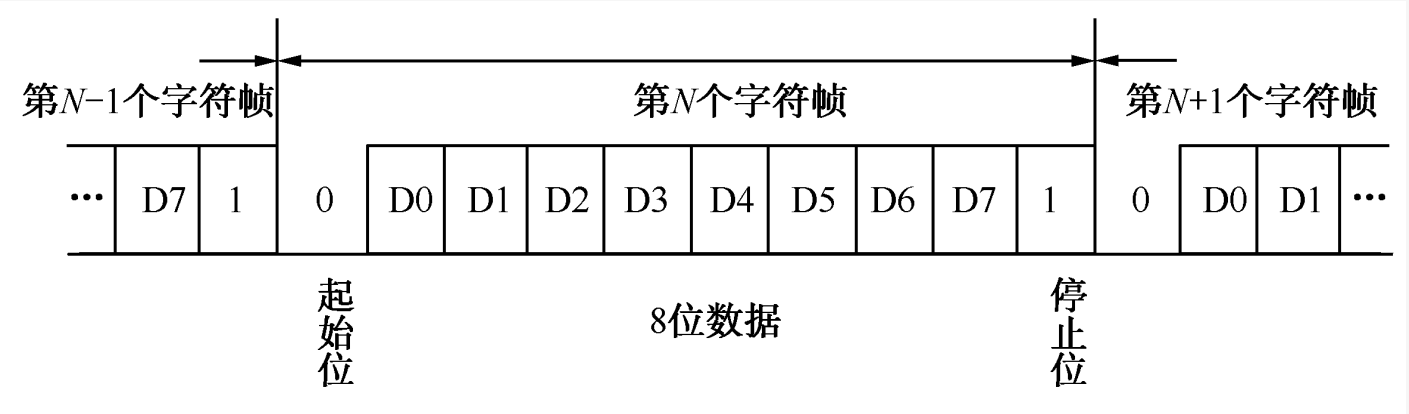
串行控制寄存器 SCON 中的 TB8 和 RB8 在方式 0 中未用。值得注意的是，每当发送或接收完 8 位数据后，硬件会自动置 TI 或 RI 为 1 。 CPU 响应 TI 或 RI 中断后，必须由用户用软件清 0 。在方式 0 时， SM2 必须为 0 。

1.3 硬件设计

2.3 串行口工作方式

2．串行工作方式 1

方式 1 为串行口波特率可调的 10 位通用异步接口 UART 。发送或接收 1 帧信息，包括 1 位起始位 0， 8 位数据位和 1 位停止位 1 。发送的条件是 TI ＝ 0, 发送结束后置 TI 为 1 。



1.3 硬件设计

2.3 串行口工作方式

2．串行工作方式 1

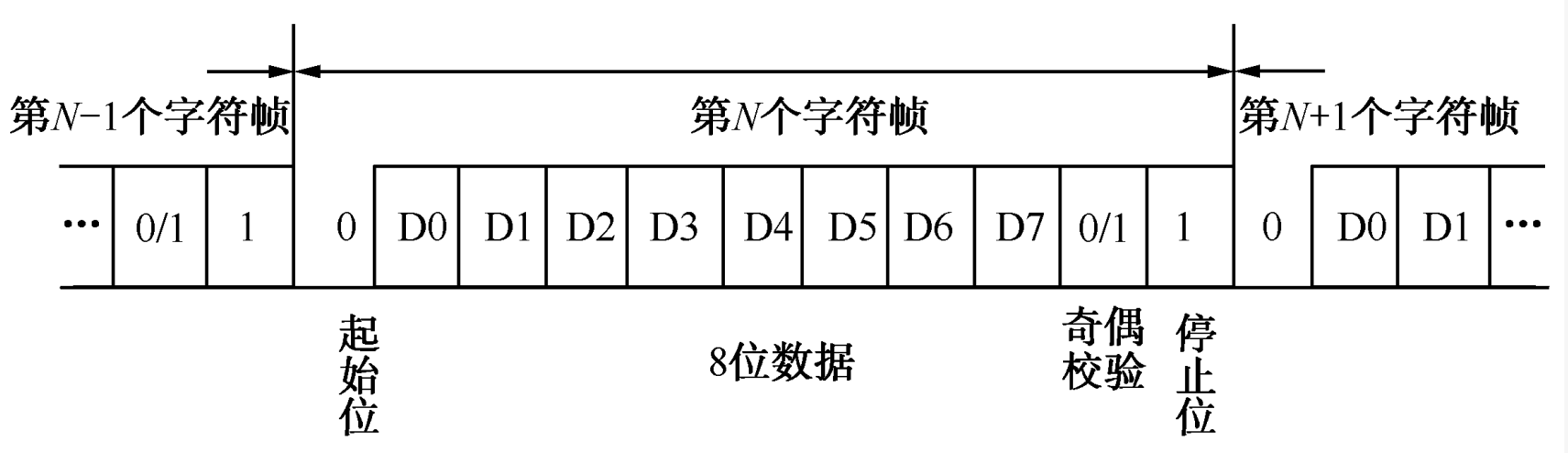
（1）发送数据时，数据从 TXD 端输出，启动发送时先把数据写入发送缓冲器 SBUF 后。发送结束置中断标志 TI 为 1。方式 1 所传送的波特率取决于定时器 1 的溢出率和 PCON 中的 SMOD 位。

（2）接收数据时，将 REN 置 1 允许接收，同时还要满足 RI=0、 SM2=0 和接收到停止位为 1。在 SBUF 中保存接收数据结果。当 RI=0，且停止位为 1 或 SM2=0 时，停止位进入 RB8 位，同时置中断标志 RI；否则信息将丢失。方式 1 接收时，应先用软件清除 RI 或 SM2 标志。详阅

2.3 串行口工作方式

3．串行工作方式 2

方式 2 下，串行口为 11 位 UART，传送波特率与 SMOD 有关。发送或接收 1 帧数据包括 1 位起始位 0， 8 位数据位， 1 位可编程位（用于奇偶校验）和 1 位停止位 1 。模式 2 的波特率是固定的，且有两种：一种是：$\frac{f_{osc}}{32}$，另一种是：$\frac{f_{osc}}{64}$。



2.3 串行口工作方式

3．串行工作方式 2

（1）发送时，先将第 9 位数据存入 TB8，然后将要发送的数据写入 SBUF，启动发送。当发送结束后，将 TI 置 1，在发送下 1 帧信息之前，TI 必须由中断服务程序或查询程序清 0。

（2）接收时，前提是 REN ＝ 1。当接收器接收到第 9 位数据后，必须是 RI ＝ 0 和 SM2 ＝ 0；接收到的第 9 位数据为 1，或者是 SM2=0 时，则接收数据有效，把数据送入 SBUF，第 9 位送入 RB8，并置 RI ＝ 1。

1.3 硬件设计

2.3 串行口工作方式

4．串行工作方式 3

方式 3 为波特率可变的 11 位 UART 通信方式，除了波特率以外，方式 3 和方式 2 完全相同。

1.3 硬件设计

2.4 波特率设置

比特率是指每秒时间内传送二进制数据的位数，来描述数据的传输速率，单位为bps （ bits per second ），是衡量串行数据速度快慢的重要指标。在串行通信中，收发双方对传送的数据传输速率必须有一定的约定。单片机的串行口通过编程可以有 4 种工作方式。其中，方式 0 和方式 2 的波特率是固定的，方式 1 和方式 3 的波特率可变，由定时器 T1 的溢出率决定。

1.3 硬件设计

2.4 波特率设置

1．串行工作方式 0

在方式 0 中，每个机器周期产生一个移位时钟，发送或接收一位数据。波特率为时钟频率的 1/12，固定不变，而且不受电源控制寄存器 PCON 中的波特率倍增位 SMOD 的影响，方式 0 的波特率由公式（ 8.1 ）计算得出。

$$波特率 = \frac{f_{osc}}{32}$$

其中：f_{osc}是单片机的振荡频率。

2.4 波特率设置

2．串行工作方式 2

在方式 2 中，波特率产生的模式与方式 0 有所区别，除了与系统的振荡频率有关外，还取决于 PCON 中的 SMOD 值，当 SMOD ＝ 0 时，波特率为 f_{osc} /64；当 SMOD ＝ 1 时，f_{osc} 波特率为 /32，方式 0 的波特率由公式（8.2）计算得出。

$$波特率 = \frac{2^{SMOD}}{32} f_{osc}$$

其中：f_{osc} 是单片机的振荡频率。

1.3 硬件设计

2.4 波特率设置

3．串行工作方式 1 和方式 3

方式 1 和方式 3 的波特率是可变的，由定时器 T1 的计数溢出率和 PCON 中的 SMOD 值共同决定。相应的计算如公式 8.3 所示。

$$\text{波特率} = \frac{2^{SMOD}}{32} \times \text{定时器 T1 的溢出率}$$

其中：f_o是单片机的振荡频率。

2.4 波特率设置

3．串行工作方式 1 和方式 3

定时器 T1 做波特率发生器使用时，通常采用定时方式 2 。当定时器的方式 2 是一个 8 位计数器，且初值自动加载模式。假如计数设置初值为 X ，当计数器经过“ 256 － X”个机器周期，定时器 T1 就会产生一次溢出中断。因此定时器 T1 的计数溢出率由公式 8.4 计算得出。

$$\text{定时器 T1 的溢出率} = \frac{f_{osc}}{12} \cdot \frac{1}{2^K - T1_{初值}}$$

其中：f_{osc}是单片机的振荡频率。

K 为定时器 T1 的位数，与定时器 T1 的工作方式有关。

1.3 硬件设计

2.4 波特率设置

3．串行工作方式 1 和方式 3

所以，方式 1 和方式 3 波特率由公式 8.5 计算得出。

$$波特率=\frac{2^{SMOD}}{32}\cdot\frac{f_{osc}}{12}\cdot\frac{1}{2^{K}-T1_{初值}}$$

由公式 8.5 推出定时器 T1 工作在模式 2 时的初值，由公式 8.6 计算。

$$X=256-\frac{f_{osc}(SMOD+1)}{384\cdot 波特率}$$

1.3 硬件设计

2.4 常用波特率及误差

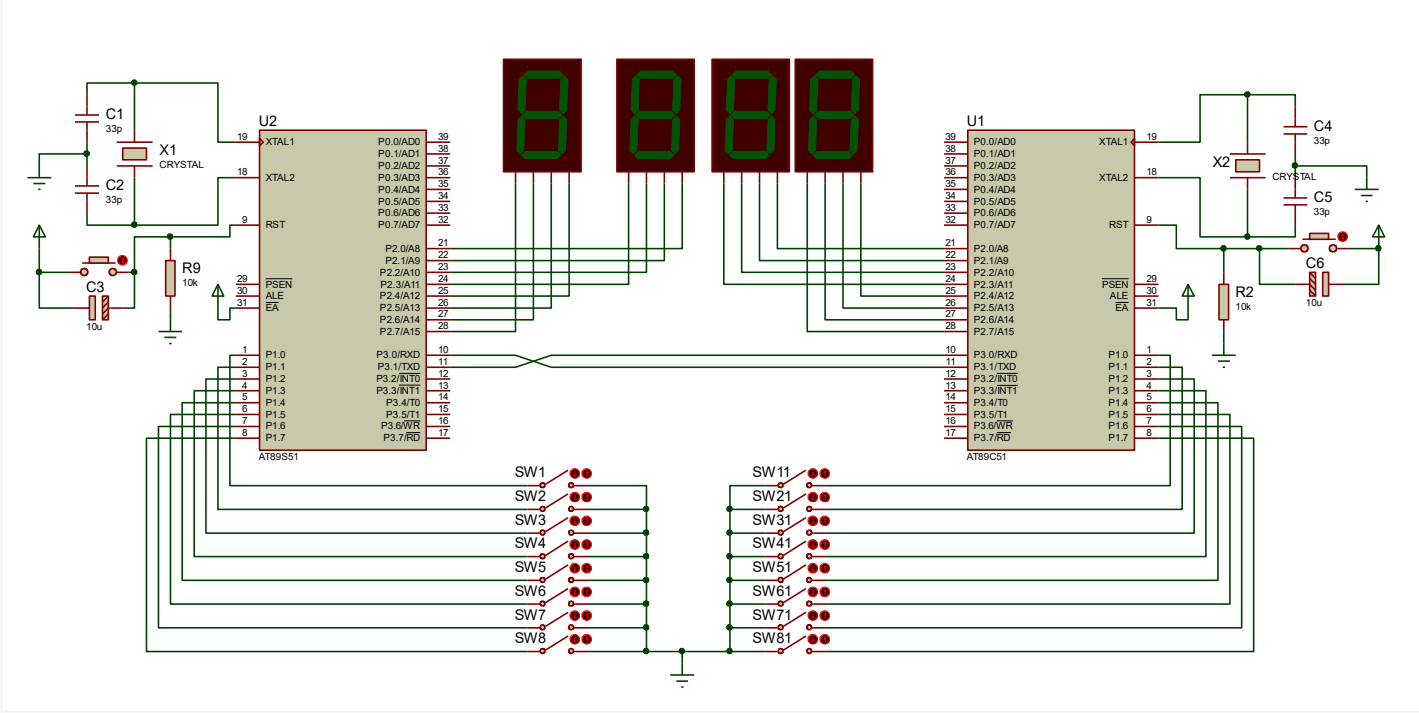
表 8-4 常用波特率及误差

晶振频率	波特率/Hz	SMOD 的值	TH1 值	实际波特率	对应误差
12.00	9600	1	F9H	8 823	7%
12.00	4800	0	F9H	4 460	7%
12.00	2400	0	F3H	2 404	0.16%
12.00	1200	0	E6H	1 202	0.16%
11.0592	19200	1	FDH	19 200	0
11.0592	9600	0	FDH	9 600	0
11.0592	4800	0	EAH	4 800	0
11.0592	2400	0	F4H	2 400	0
11.0592	1200	0	E8H	1 200	0

1.3 硬件设计

2. 硬件电路

由于甲乙两台单片机都能获取到按键输入的数值，输入数值方法采用八个独立式按键电路实现，通过八个按键的闭合断开状态的二进制数值作为输入的数值，给单片机端口送数，显示采用的数码管显示。每台单片机都外接有八个按键，一端共地，另一端分别连接到单片机的 P1.0 至 P1.7，模拟八位二进制数据的输入，使用两位数码管对输入的 P1 端口的数值进行显示，一位显示低四位数值，与单片机的 P2.0 至 P2.3 连接，另一位显示高四位数值，与单片机的 P2.4 至 P2.7，甲单片机的 RXD 连接到乙单片机的 TXD，甲单片机的 TXD 连接到乙单片机的 RXD 。



1.3 硬件设计

3. 软件设计

1. 编程思路

编程的思路主要有以下几个步骤：

（1）首先要对相关寄存器进行设置，并对其进行初始化，主要有确定 T1 的工作方式（设置 TMOD ）、计算初始值赋值给 TH1 和 TL1 。启动定时器 T1 。

（2）确定串行口控制（设置 SCON ）。

（3）设定接收 / 发送的波特率。

（4）设定 SMOD 的状态，以控制波特率是否加倍。

（5） 串行口要进行中断设置。 }

1.3 硬件设计

3. 软件设计

根据题目的要求，为了省去循环等待时间，而采用中断的方法。接收数据时，使用“中断方式”，清除 RI 后，用一个变量通知主函数，收到新数据。发送数据时，也用“中断方式”，清除 TI 后，用另一个变量通知主函数，数据发送完毕。

```
void  get_data (void) interrupt 4 using 0
{
   if (RI)                          // 如果是串口输入引起中断
{
   dis_port = SBUF;
   RI=0;
}
   else TI=0;                       // 否则就是串口输出引起的中断
}
```

在程序编写过程中，只有当端口发生数值变化时，再触发中断。

```
#include <reg51.h>
#define uchar unsigned c
har
#define uint unsigned int
#define key_In P1
#define dis_Out P2

void main (void)
{
        uchar  key_Por
t=0xff;
        SCON=0x50; //
MODER1,REN=1;
        TMOD=0x20; /
/TIMER1 MODER2;
        TH1=0xf3;  //b
ode=2400
        TL1=0xf3;
        ET1=1;
        TR1=1;
        EA=1;
        ES=1;
        while(1)
        {
        if (key_In != key_Port)
        {
        key_Port=key_In;
       SBUF=key_Port ;
        }
        }
}

void get_disp (void) interrupt 4 using 0
{
        if (RI)                          // 如果是串口输
入引起中断
        {
        dis_Out = SBUF;
        RI=0;
        }
         else TI=0;// 否则就是串口输出引起的中断
}
```

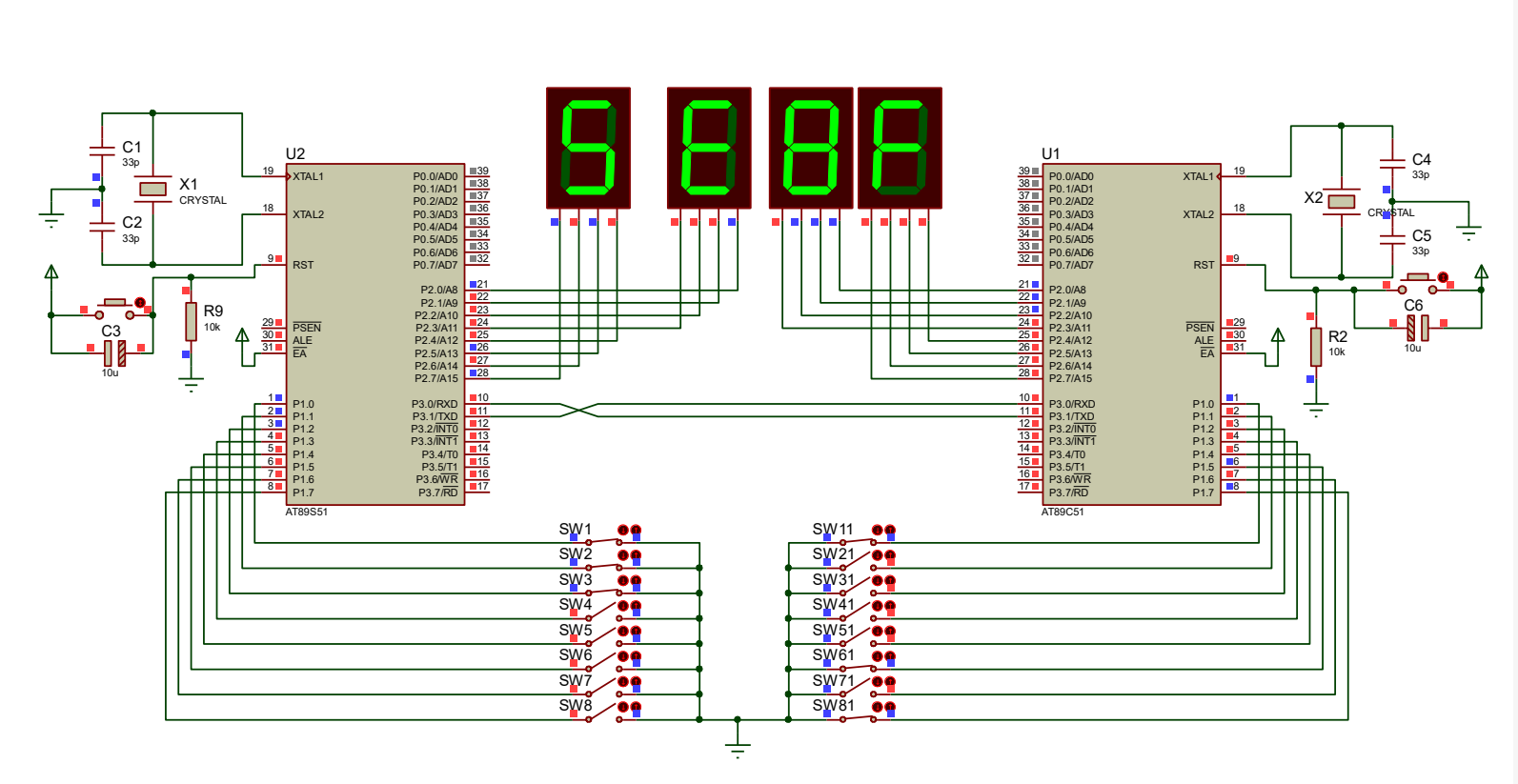
1.5 软件仿真

1. 使用 Proteus 软件，绘制硬件电路图，并保存到指定的位置。

2. 使用 KeilC51 建立一个单片机通信工程项目，建立一个 .c 文件，在编辑区中输入上述的源程序，并以“ mcu1.c” 为文件名存放“双单片机串口通信”文件夹中；进行编译，排除所有程序故障，直到无错误为止，目标代码文件“ mcu1.hex”。

3. 使用 Proteus 打开绘制好的按键电路设计图，双击电路图中 AT89S51 单片机，把编译好的“ mcu1.hex” 文件下载到单片机中。点击模拟调试按钮进行仿真。刚上电时，由于按键没有输入都是高电平，因此两个单片机的显示都是“ FF”。然后使用鼠标单击每台单片机的按键，调整单片机端口的输入数值，可以看到两个单片机都能正确显示对方发送过来的数值。

1.5 软件仿真


C1
33p
X1
CRYSTAL
C2
33p
C3
10u
R9
10k
U2
XTAL1
XTAL2
RST
PSEN
ALE
EA
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89S51
U1
AT89C51
X2
CRYSTAL
C4
33p
C5
33p
C6
10u
R2
10k
SW1
SW2
SW3
SW4
SW5
SW6
SW7
SW8
SW11
SW21
SW31
SW41
SW51
SW61
SW71
SW81

1.6 项目总结

本项目实现了单片机双向通信系统设计与仿真，主要是讲解串行通信相关概念以及主要寄存器，让学者掌握串行口通信的工作原理、相关的寄存器以及初始化以及初始值的设置方法等，重点阐述了单片机双向通信系统的编程控制的方法。

项目提升

1、数据通信的传输方式有哪些？

2、 MCS-51 单片机串行口有几种工作方式？各自的特点是什么？

3、串行口设有几个控制寄存器？它们的作用是什么？

4、编程的思路主要有哪些步骤？

5、串行口扩展主要分为哪两种？

1. 了解串行接口标准；
2. 掌握单片机与 PC 机通信的串口电路设计；
3. 掌握单片机与 PC 机通信的控制方法；

1. 会阐述串行接口标准；
2. 会设计单片机与 PC 机通信的串口的电路；
3. 会编程实现单片机与 PC 机通信的单片机功能；

1. 能分析设计任务，正确使用串行口通信；
2. 能熟练编写单片机与 PC 机通信的发送和接收数据程序；
3. 能完成单片机与 PC 机间的通信；
4. 能使用 PROTUES 软件绘制电路原理图；
5. 能使用 Keil 软件编译程序实现单片机与 PC 机的控制方法，并与 PROTUES 软件联调，实现控制电路仿真；
6. 能够完成单片机与 PC 机电路设计和仿真；
7. 能够使用串行口通信去解决实际问题。

4 课时

1.1 任务提出

设计一个单片机与 PC 机通信系统，具体功能如下：1．当按下 K1 时，单片机将采集 P1 端口的状态值，并发送给 PC 机，以十六进制形式显示出来；2．单片机接收 PC 机的数据（0-9 中任意一个数）传送到 P0 口，并用数码管进行显示。

1.2 方案设计

1. 单片机选择。采用 ATMEL 公司的 AT89C51 作为系统控制器的 CPU 方案。单片机算术运算功能强，软件编程灵活、自由度大，可以用软件编程实现各种算法和逻辑控制，并且由于其功耗低、体积小、技术成熟和成本低等优点，使其在各个领域应用广泛。

2. 显示电路的选择。根据题目要求，能够直观显示倒计时时间。采用 LED 数码管显示， LED 数码管价格适中，显示控制也方便。

3. 输入数值方法的选择。为了能够获取到的单片机端口的状态付，单片机端口外接逻辑状态输入，用于输入对应的高电平和低电平。

4. 通信的选择。这里采用直接的串行通信，电路方便，只需要两条线即可通信，程序编写也易于控制。但是为了能够与 PC 机进行通信，中间需要增加转换电路，可以采用 MAX232 电路，用于单片机和 PC 机的匹配。

1.3 硬件设计

1. 单片机通信技术

1.1 串行通信标准

串行正常的通信需要双方采用相同的接口标准。不同的设备的串行接口具有不同的信号线定义、电器规格，需经一个统一的串行通信接口标准才能使不同的设备间能够正常通信。串行接口标准是最广泛的一种串行接口，主要有 RS232 、 RS422 、 RS485 等串行接口标准。

RS-232C 是美国电子工业协会在 1969 年提出的串行通信接口的电气标准，是目前使用最多的一种异步串行通信总线标准。该标准定义了数据终端设备 (DTE) 和数据通信设备 (DCE) 间按位串行传输的接口信息，合理安排了接口的电气信号和机械要求。由于它采用了单端驱动非差分接收电路，传输距离受限，最大传输距离为 15m ，传输速率为 0 ～ 20Kbit/s 。如果想进行远距离的通信，需要额外增加 Modem(调制调节器) ，导致成本会增加。 RS-232-C 总线标准设有 25 条信号线，包括一个主通道和一个辅助通道。在多数情况下主要使用主通道，对于一般双工通信，仅需几条信号线就可实现，如一条发送线、一条接收线及一条地线。 RS-232-C 标准规定的数据传输速率为每秒 50 、 75 、 100 、 150 、 300 、 600 、 1200 、 2400 、 4800 、 9600 、 19200 波特。

1.3 硬件设计

1. 单片机通信技术

1.1 串行通信标准

RS-449 是美国电子工业协会在 1977 年提出的标准，主要是在传输速度、通信距离上作了改进，此标准保留了 RS-232 兼容特点外，还增加了 10 个控制信号。除此还推出了 RS-422 和 RS-423，是 RS-449 的标准子集。 RS-422 采用的是平衡驱动差分接收电路，驱动能力比 RS232 更强，同时允许连接多个接收节点，支持点对多的双向通信，传输速度最大为 10Mb/s，传输距离达到 1200m。在正常使用 RS422 时，需要接电阻进行电路匹配，阻值约等于传输电缆的特性阻抗，但是在传输距离 300m 以下，不需终接电阻。

1.3 硬件设计

1. 单片机通信技术

1.1 串行通信标准

RS485 是 RS422 改进型， RS-485 许多电气规定与 RS-422 相似。 RS-485 采用平衡发送和差分接收，因此具有抑制共模干扰的能力。 RS-485 采用半双工工作方式，任何时候只能有一点处于发送状态，因此，发送电路须由使能信号加以控制。 RS-485 可以采用二线制方式可实现真正的多点双向通信，四线方式只能实现点对多的通信，但是其连接负载的能力得到很大改进，应用 RS-485 可以联网构成分布式系统，其允许最多并联 32 台驱动器和 32 台接收器。 RS485 传输距离和最大传输速率与 RS422 一样，但是 RS485 需要 2 个终接电阻。

1.3 硬件设计

1. 单片机通信技术

1.2 RS-232C 串行通信协议

（ 1 ） RS-232C 电气特性

RS-232C 串行通信协议采用的负逻辑，这一点与 TTL 、 MOS 逻辑电平不同。规定逻辑“ 1” 的电平值是 -3V ～ -15V ， 逻辑“ 0” 的电平值是 +3 ～ +15V 。 RS-232C 需要加上适当的电平转换电路才能与 TTL 电路连接。否则将把 TTL 电路烧坏。实现这种变换的方法可用分立元件，也可用集成电路芯片。目前比较常用的是 MAX232 、 MC1488 和 MC1489 芯片等。

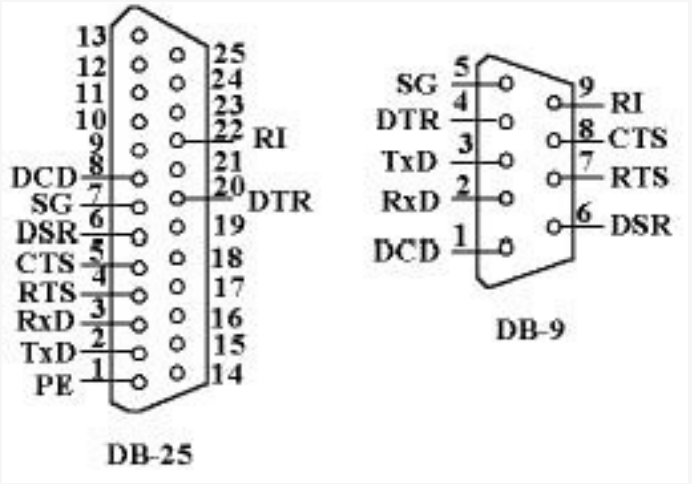
1.3 硬件设计

1. 单片机通信技术

1.2 RS-232C 串行通信协议

（ 2 ） RS-232C 引脚功能

RS-232C 采用的是 25 针的 D 形连接器，也有大部分采用的是 9 个引脚 。



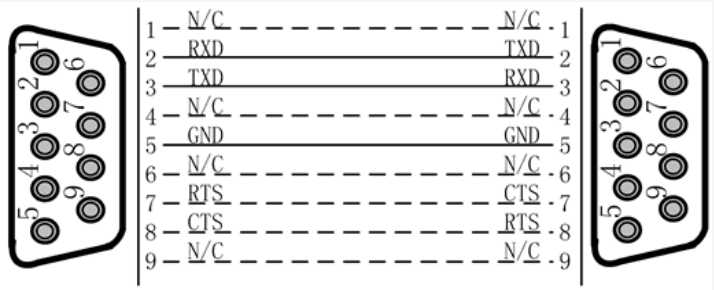
1.3 硬件设计

1. 单片机通信技术

1.2 RS-232C 串行通信协议

（ 2 ） RS-232C 引脚功能

在一些简单的通信系统中，只需要 TXD 、 RXD 和 GND （地） 3 个引脚就可以实现通信。因此，只要把控制端设备的发送（ TXD ）引脚与受控设备的接收（ RXD ）引脚相连，就能和受控设备通讯，如果需要接收受控设备的返回信息，那就需要找出受控设备串行接口的发送（ TXD ）脚与控制端设备串口的接收（ RxD ）脚相连。



1.3 硬件设计

1. 单片机通信技术

1.2 RS-232C 串行通信协议

（2） RS-232C 引脚功能

表 8-5 RS-232C 标准接口主要引脚定义和功能

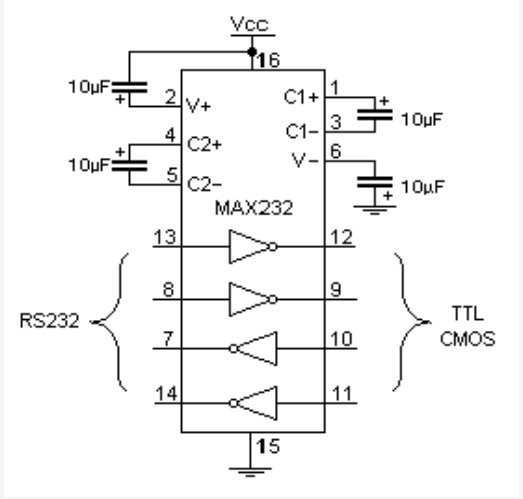
9针引脚	25针引脚	缩写	功能说明
1	8	DCD	数据载波检测
2	3	RXD	接收数据
3	2	TXD	发送数据
4	20	DTR	数据终端准备
5	7	GND	信号地
6	6	DSR	数据设备准备好
7	4	RTS	请求发送
8	5	CTS	清除发送
9	22	DELL	铃指示

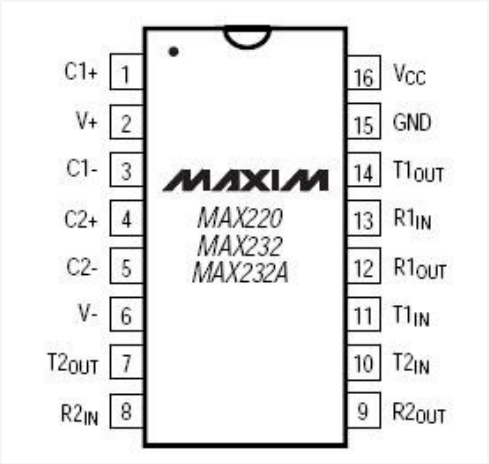
1.3 硬件设计

1. 单片机通信技术

1.3RS-232C 芯片

MAX232 是一种把电脑的串行口 RS232 信号电平（ -10 ， +10v ）转换为单片机所用到的 TTL 信号点平（ 0 ， +5 ） M 的芯片，价格便宜 。





1.3 硬件设计

1. 单片机通信技术

1.3RS-232C 芯片

MAX232 由三部分电路组成：

第一部分电路是电荷泵电路，主要是外接电容，作用是进行电压匹配和电源去耦。电路由 1 、 2 、 3 、 4 、 5 、 6 脚和 4 只电容组成，产生 +12V 和 -12V 电源供给 RS-232 串口电平的需要。

第二部分电路是数据转换通道。主要由 7 、 8 、 9 、 10 、 11 、 12 、 13 、 14 脚组成，有两个通道。第一通道由 13 、 12 、 11 和 14 脚组成，第二通道由 8 、 9 、 10 、 7 脚组成。 TTL/CMOS 数据从 11 、 10 脚输入从 14 、 7 脚输出； RS-232 数据从 13 、 8 脚输入转换成 TTL/CMOS 数据从 12 、 9 脚输出。

1.3 硬件设计

1. 单片机通信技术

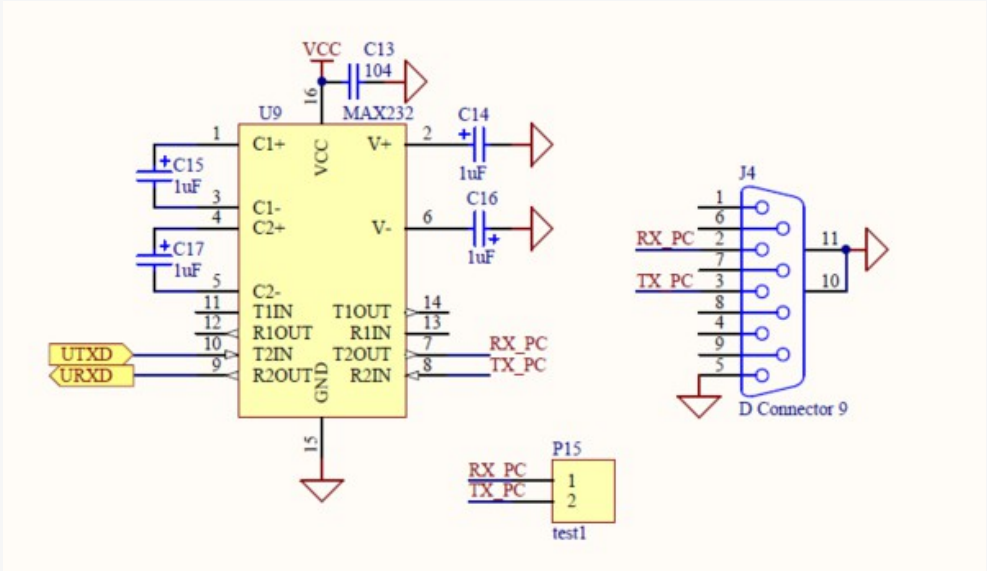
1.3RS-232C 芯片

第三部分是供电电路， 15 脚接 GND ， 16 脚接 VCC 。

通过 MAX232 的 TTL 和 RS232 的输入 / 输出端口，自动地调节了单片机串口的 TTL 电平信号和 RS-232 的串行通信信号的电平匹配。

MAX232 的电路图如图 8-11 所示。 MAX232 引脚 C1+ 与 C1- 、 C2+ 与 C2- 、 V+ 与 VCC 、 V- 与 GND 之间的 4 个 0.1uF 的电容不可缺少，一般选用陶瓷介质的电容。

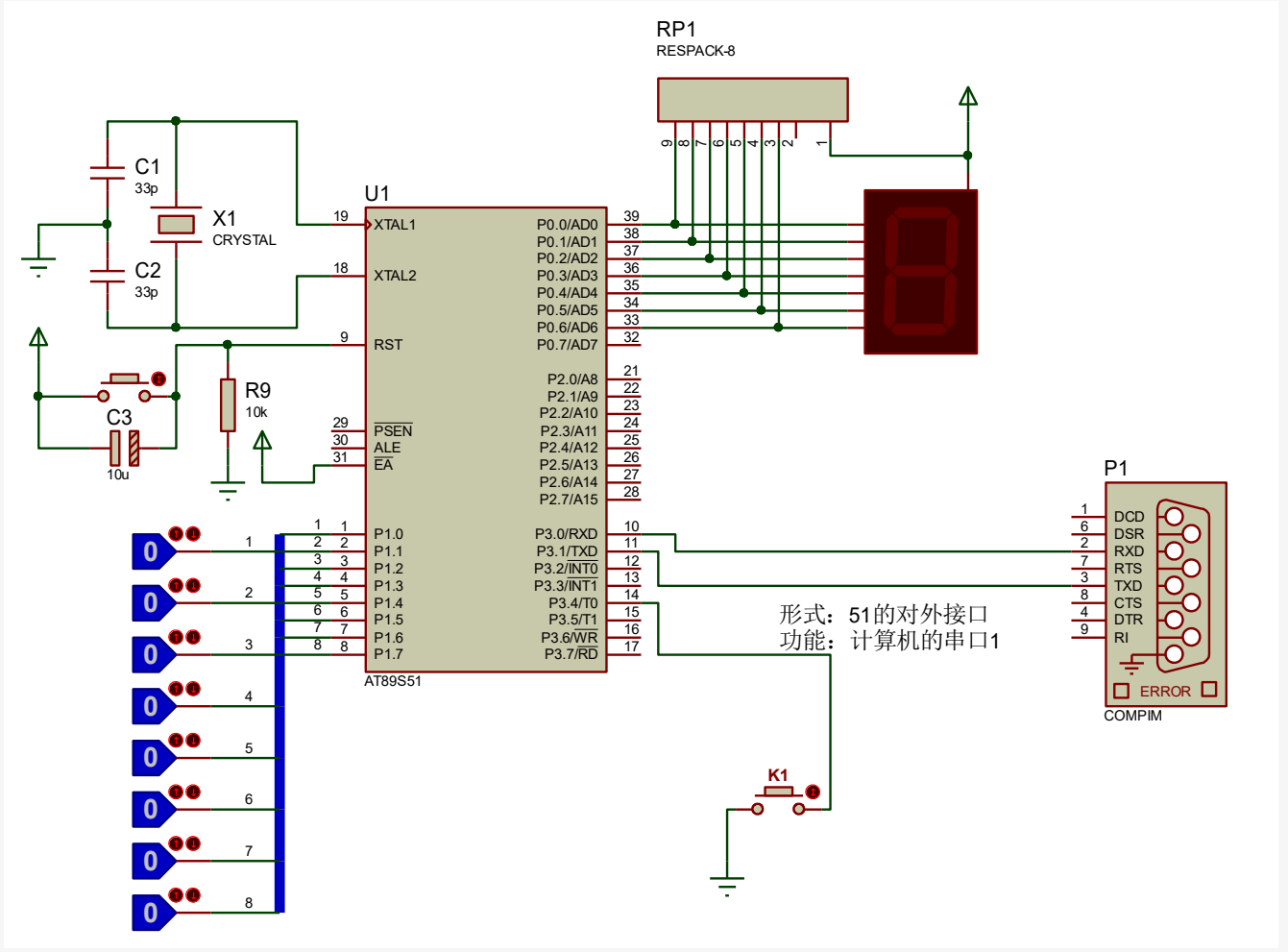
MAX232 可以用作单片机和单片机之间、单片机和 PC 机串口之间的符合 RS232 串行接口电路。只要将待进行串行传输的设备的发送和接收端相应的接上，编程即可。



1.3 硬件设计

2. 硬件电路

根据题目的实现功能，主要由四个部分组成：单片机最小系统、数码管显示电路、按键输入电路和 MAX232 转换电路。这里选择具有内部程序存储器的 AT89S51 单片机作为控制电路，按键 K1 接到单片机 P3.4 引脚，用于控制向 PC 机发送数据。数码管显示电路采用共阳极，需接上拉电阻后接到单片机 P0 端口。由于缺少实物， MAX232 转换电路直接使用一个虚拟终端来模拟实现，可以直接连接单片机的 RXD 和 TXD 引脚。



1.3 硬件设计

3. 软件设计

（1）串口初始化

串口初始化程序主要实现同对串口一系列参数的设置，包括产生波特率的定时器 1, 串行口控制和中断控制。具体步骤如下：

（1）确定定时器 T1 的工作方式（对 TMOD 寄存器进行编程）；

（2）计算定时器 T1 的初值，赋值 TH1 和 TL1；

（3）启动定时器 T1 （对 TCON 中的 TR1 进行编程）；

（4）确定串行口控制（对 SCON 寄存器编程），串行口在中断工作方式时，还要设置中断寄存器 IE 和优先寄存器 IP 。

串口初始化程序具体如下：

```
TMOD = 0x20;        // 定时器 1 工作于 8 位自动重载模式，用于产生波特率
TH1 = 0xFD;         // 波特率 9600
TL1 = 0xFD;
SCON = 0x50;                    // 设定串行口工作方式
PCON &= 0xef;                   // 波特率不倍增
TR1 = 1;                        // 启动定时器 1
IE = 0x00;                      // 禁止任何中断
```

1.3 硬件设计

3. 软件设计

按键的检测采用查询法实现，检测引脚的状态值是否为 0 即可判定按键是否按下。

```
if(K1==0)
{
  while(K1==0);
{
  // 执行发送操作
}
}
```

1.3 硬件设计

3. 软件设计

数码管采用的是共阳极数码管，只能显示数字 0-9 的数字。首先对 0-9 的数字所对应段码用一维数组保存，当接收到 PC 机发送过来的数字时，从数组调用对应段码送端口 P0 显示。由于 PC 机发送过来的是字符，需要将字符转换成数值才能显示，因此需要减去 0x30 。

```
Unsigned char dis []={0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80,0x90};
P0 = display_code[tmp-0x30];
```

1.3 硬件设计

3. 软件设计

串口接收或发送程序的功能主要是单片机接收 PC 机发送过来的数据或者单片机向 PC 发送数据，都采用查询法实现。具体如下：

```
if(K1==0)
{
    while(K1==0);
    {
     m = P1;
     SBUF = m;
     while(!TI);
    TI = 0;
    }
}
if(RI)
{
RI = 0;
tmp = SBUF;
delayms(10);
P0 = display_code[tmp-0x30];
}
```

1.5 软件仿真

1. 使用 Proteus 软件，绘制的硬件电路图，并保存到指定的位置。

2. 使用 KeilC51 建立一个单片机与 PC 通信工程项目，建立一个 .c 文件，在编辑区中输入上述的源程序，并以“ mcu1.c” 为文件名存放“单片机 PC 通信”文件夹中；进行编译，排除所有程序故障，直到无错误为止，目标代码文件“ mcu1.hex”。

3. 使用 Proteus 打开绘制好的按键电路设计图，双击电路图中 AT89S51 单片机，把编译好的“ mcu1.hex” 文件下载到单片机中。点击模拟调试按钮进行仿真。刚上电时，由于按键都设置成低电平（ 0 ），数码管没有接收到 PC 机发送的数据，没有显示。使用鼠标单击逻辑输入按键的状态，使部分状态值得变成“ 1”，再按一下 K1 ，单片机就向 PC 发送 P1 端口的数据值，使用串口调试工具，可以接收并显示单片机发送的数据。

```
#include <reg51.h>
#include <intrins.h>
unsigned char m,tmp;
unsigned char

display_code[]={0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80,0x90};

void delayms(unsigned char ms);

sbit        K1 = P3^4;

main()
{
        TMOD = 0x20;                    // 定时器 1 工作于 8 位
自动重载模式，用于产生波特率
        TH1 = 0xFD;                             // 波特率 9600
        TL1 = 0xFD;

        SCON = 0x50;                    // 设定串行口工作方式
        PCON &= 0xef;                   // 波特率不倍增

        TR1 = 1;                        // 启动定时器 1
        IE = 0x00;                      // 禁止任何中断
```

```
while(1)
        {
          if(K1==0)                    //  扫描按键
         {
           while(K1==0);
            { //key_v = key_s;         //  保存键值
                                        //proc_key();       //  键处理
            m = P1;
            SBUF = m;
             while(!TI);                                    //  等特数据传送
             TI = 0;                                        //  清除数据传送标志
                                }
        }
                    if(RI)                                  //  是否有数据到来
                   {            RI = 0;
                                tmp = SBUF;                 //  暂存接收到的数据
                                delayms(10);
                                P0 = display_code[tmp-0x30];//  数据传送到 P0 口
                   }
         }
       }
```

```
void delayms(unsigned c
har ms)
//  延时子程序
{

                unsigned char i
;
                while(ms--)
                {
                        for(i
= 0; i < 120; i++);
                }
}
```

1.5 软件仿真

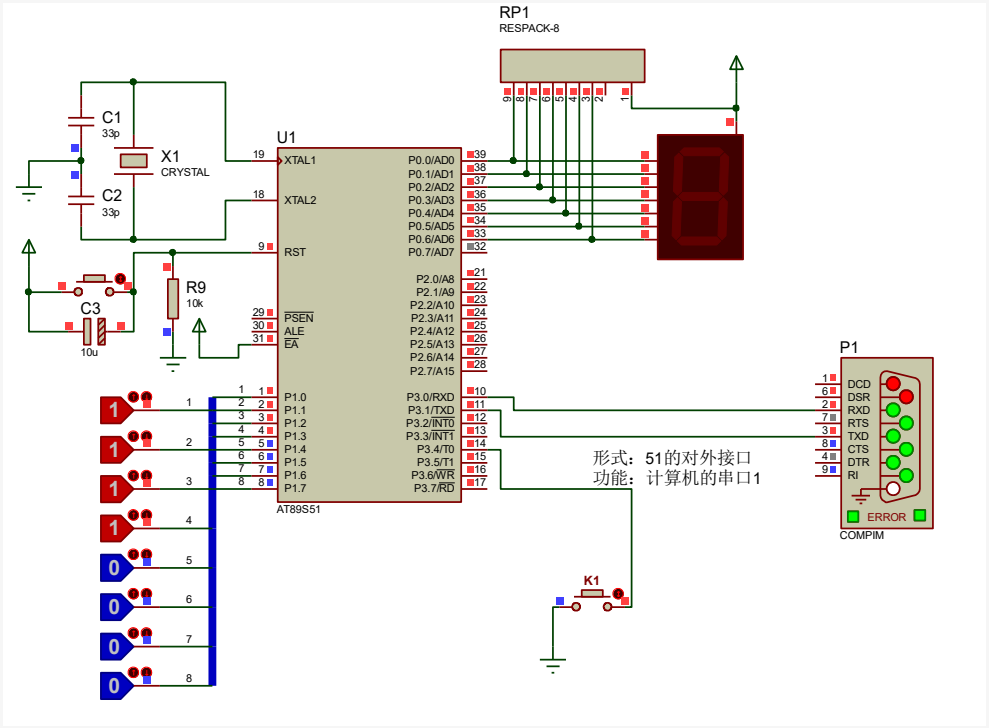


图 8-15 仿真效果图

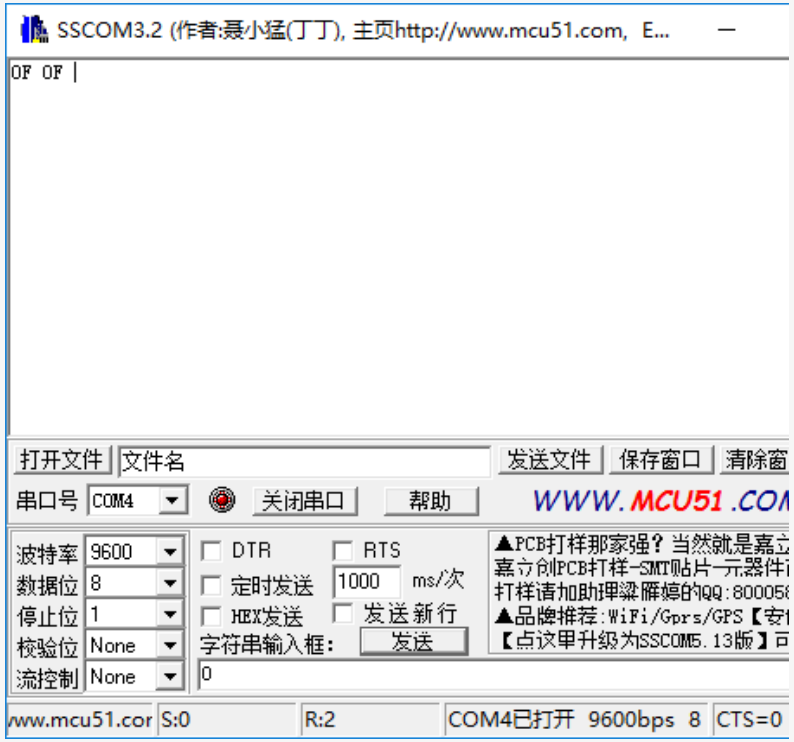


图 8-16 串口调试工具接收数据显示图

1.5 软件仿真

然后在串口调试工具的“字符口中输入框”输入“1”，点击发送，如图 8-17 所示。

图 8-17 串口调试工具发送数据

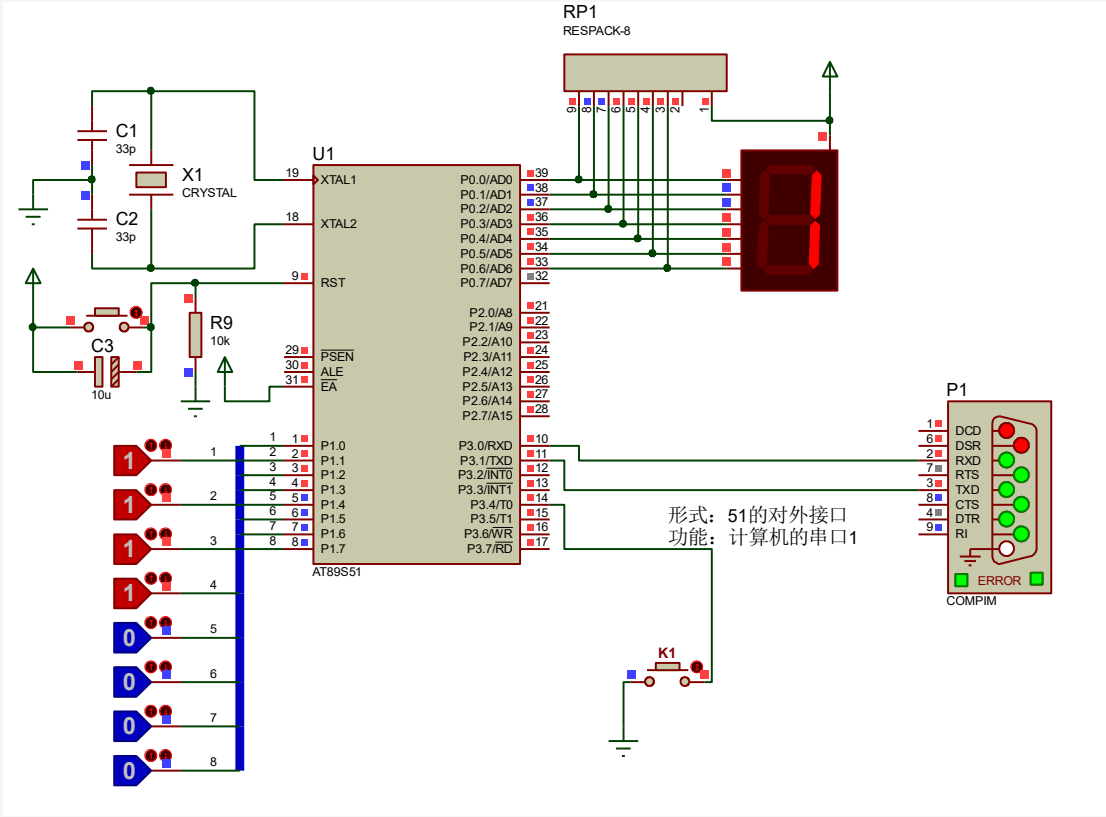


图 8-18 单片机接收 PC 发送的数据显示效果图

1.6 项目总结

本项目实现了单片机与 PC 通信系统设计与仿真，主要是串行口通信技术，让学者掌握单片机与 PC 机通信的工作原理、 MAX232 的电气特性、引脚功能以及电路设计等，重点阐述了单片机与 PC 通信的编程控制的方法。

项目提升

1、串行通信标准主要有哪几种？

2、单片机与 PC 通信时，常用哪种接口？

3、 MAX232 电路主要有哪几部分？

4、串口初始化主要有哪几个步骤？

5、 RS-232C 电气特性有哪些？