第9章 串行通信及其接口电路

第9章 串行通信及其接口电路

9.1 串行通信概述

9.2 可编程串行接口芯片8251A

9.3 8251A芯片的应用

学习目的

通过对本章的学习，应该能够达到下列要求：

- 串行通信的概念
- 8251A芯片的编程结构
- 8251A控制器的应用

学习目的

重 点

- 串行通信的概念
- 异步方式与同步方式
- 波特率
- 8251A芯片结构与命令
- 8251A应用编程

9.1 串行通信概述

- **并行通信:** 各位数据都是并行传输的，它以字节（或字）为单位与I/O设备或被控对象进行数据交换。
  - **特点：** 传输速度快；硬件开销大；只适合近距离传输。

- **串行通信：** 串行通信是通过一位一位地进行数据传输来实现通信。
  - **特点：** 具有传输线少，成本低等优点，适合远距离传送；缺点是速度慢。

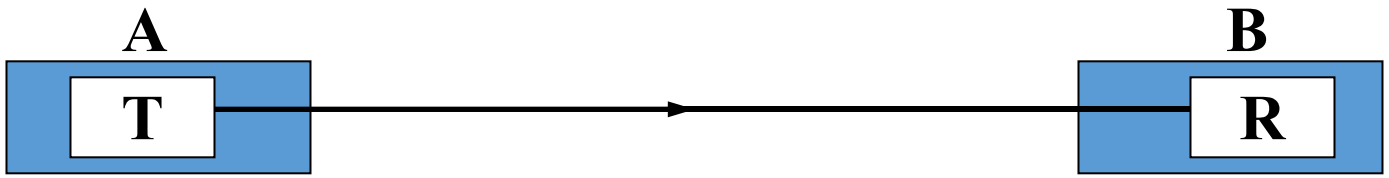
9.1 串行通信概述

1.串行数据传送方式

- 串行通信数据传送方式分为：单工通信方式、半双工通信方式和全双工通信方式。

(1)单工通信方式

传输的线路用一根线，通信的数据只允许按照一个固定的方向传送。如图：只能从A站点传送到B站点。



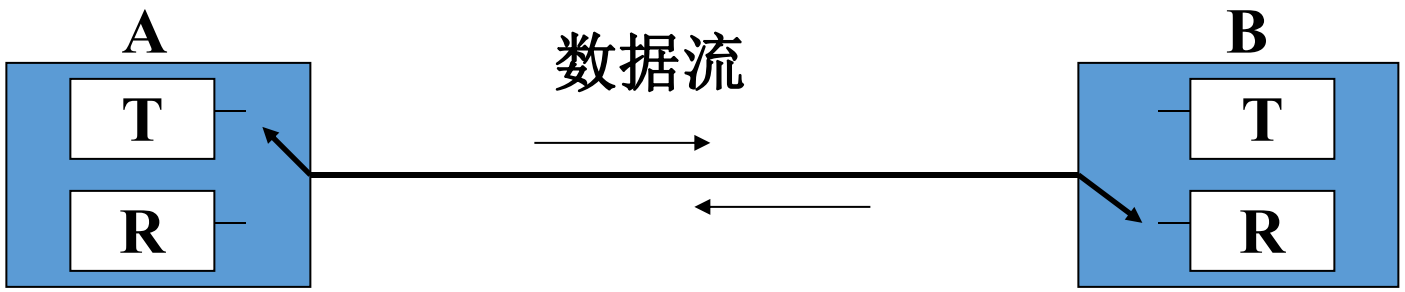
例： 单工通信类似无线电广播，电台发送信号，收音机接收信号，收音机永远不能发送信号。

9.1 串行通信概述

(2)半双工通信方式

传输的过程中依然用一根线连接，在某个时刻，只能进行发送，或只能进行接收，即发送和接收不可能同时进行。

如图:



例：半双工通信工方式类似对讲机，某时刻A方发送B方接收，另一时刻B方发送A方接收，双方不能同时进行发送和接收。

9.1 串行通信概述

(3)全双工通信方式

相互通信的双方，都可以是接收器也都可以是发送器。分别用2根独立的传输线(一般是双绞线，或同轴电缆)来连接发送信号和接收信号，这样发送方和接收方可同时进行工作。如下图所示。

全双工通信工方式类似电话机，双方可以同时进行发送和接收。

9.1 串行通信概述

2. 串行传送的两种基本工作方式

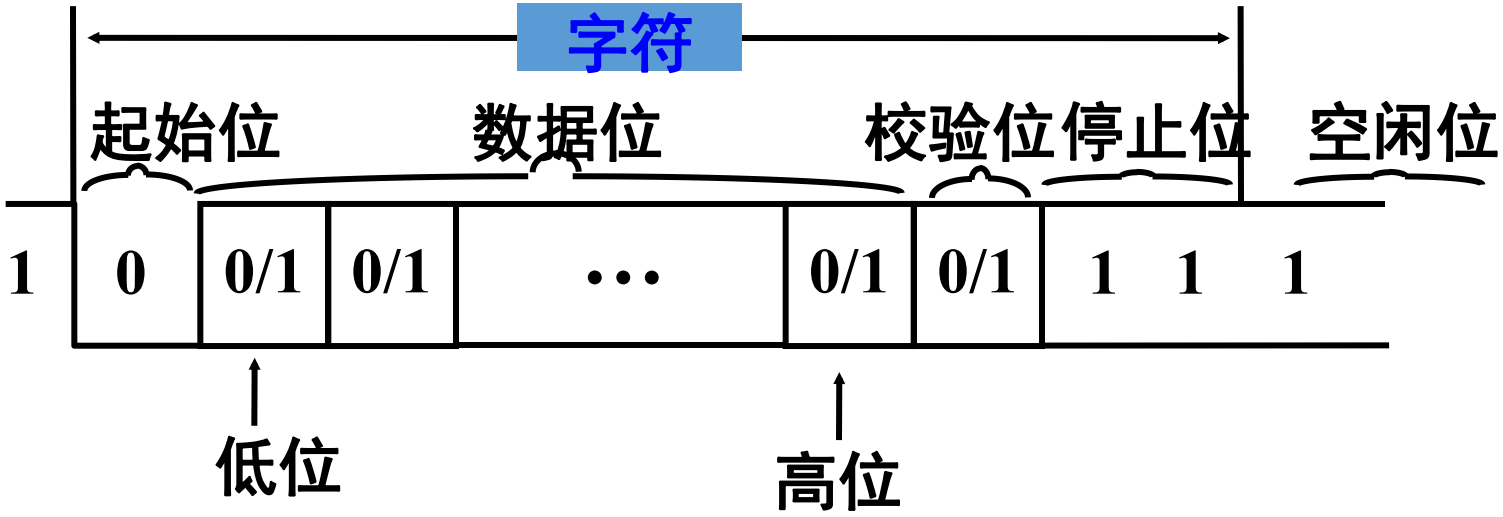
- 串行通信分为两种类型：一种是同步通信方式，另一种是异步通信方式。

(1)异步通信方式

- 它是以字符为单位进行传输的，字符之间没有固定的时间间隔要求，而每个字符中的各位则以固定的时间传送。
- 收、发双方取得同步的方法是采用在字符格式中设置起始位和停止位。
- 在一个有效字符正式发送前，发送器先发送一个起始位，然后发送有效字符位，在字符结束时再发送一个停止位，起始位至停止位构成一帧。

9.1 串行通信概述

图9.1 异步通信数据格式

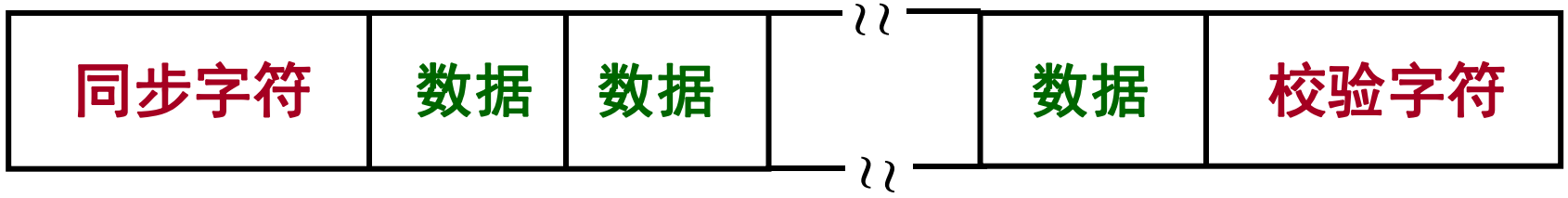


- 空闲位——传送字符之间的逻辑1电平，表示没有进行传送

9.1 串行通信概述

(2)同步通信方式

- 由一个统一的时钟控制发送方和接收方，若干字符组成一个信息组，字符要一个接着一个传送；
- 没有字符时，也要发送专用的“空闲”字符或同步字符，因为同步传输要求连续传送，字符中间不允许有间隔。
- 同步传输的特征是：在每组信息的开始(常称为帧头)要加上1—2个同步字符，后面跟着8位的字符数据。



9.1 串行通信概述

(2)同步通信方式

以一串字符为一个传送单位，字符间不加标识位，在一串字符开始用同步字符表示，硬件要求高，通信双方须严格同步。

（a）单同步字符帧结构

（b）双同步字符帧结构

9.1 串行通信概述

3. 串行传送速率

- **波特率(Baud Rate)：**波特率作为串行传输中数据传输速度的衡量单位，用每秒传输数据的位数(位/秒)来表示。

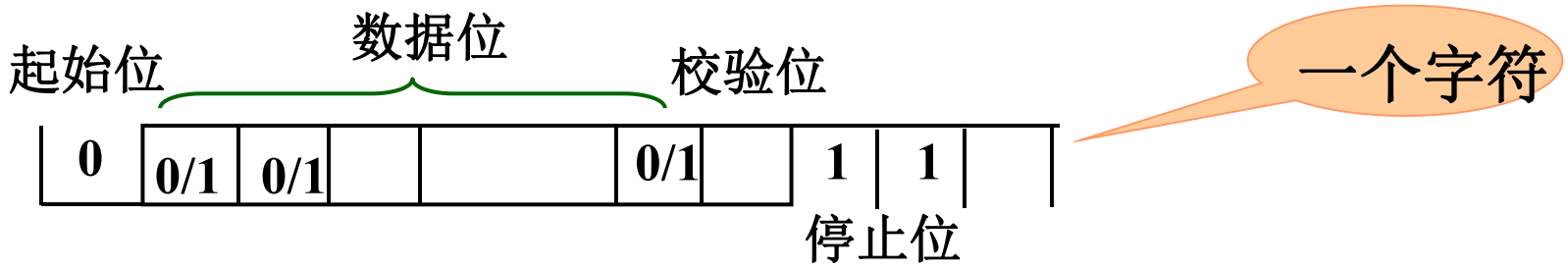
  例：10位/字符×120字符/秒=1200位/秒=1200波特

- **国际上规定的一个标准的波特率系列是：**110，300，600，1200，1800，2400，4800，9600，19200。
- 同步传送的波特率高于异步，可达64000波特。
- 异步通信允许发送方和接收方的时钟误差或波特率误差在4%～5%。

9.1 串行通信概述

例1：一个异步串行发送器，发送具有8位数据位的字符，在系统中使用一个奇偶校验位和两个停止位。若每秒发送100个字符，则其波特率为多少？

格式



100*（1+8+1+2）=1200 bps

例2：一个异步串行发送器，发送具有7位数据位的字符，传送波特率为1800，字符格式为：1个奇偶校验位，1个停止位，问：十秒钟内传送了多少个字符?

10 * 1800/(1+7+1+1) = 1800

9.1 串行通信概述

4.发送时钟和接收时钟

- 二进制数据序列称为比特组，由发送器发送到传输线上，再由接收器从传输线上接收。
- 二进制数据序列在传输线上是以数字信号形式出现，即用高电平表示二进制数1，低电平表示二进制数0。
- 每一位持续的时间是固定的，在发送时是以发送时钟作为数据位的划分界限，在接收时是以接收时钟作为数据位的检测。

9.1 串行通信概述

每个数据位的传送时间Td为波特率的倒数：

Td ＝1/1200＝0.000 833 s＝0.833 ms

发送/接收时钟：

用时钟来检测每一位数据的位宽度。

波特率因子K：

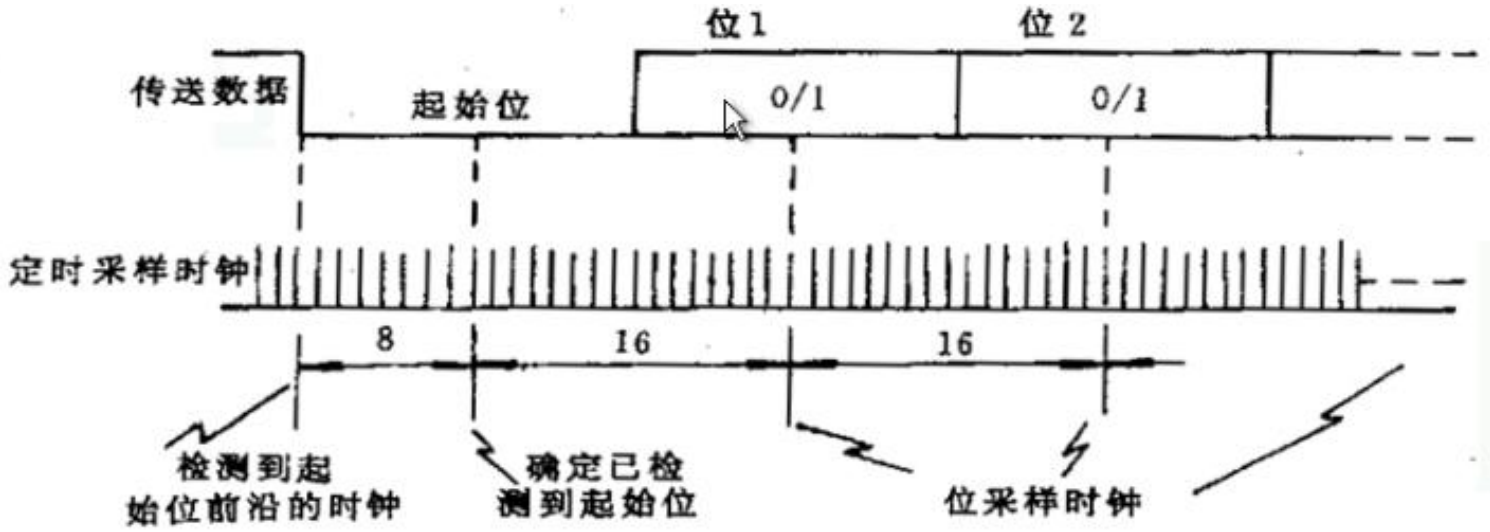
每BIT占用的时钟周期数。

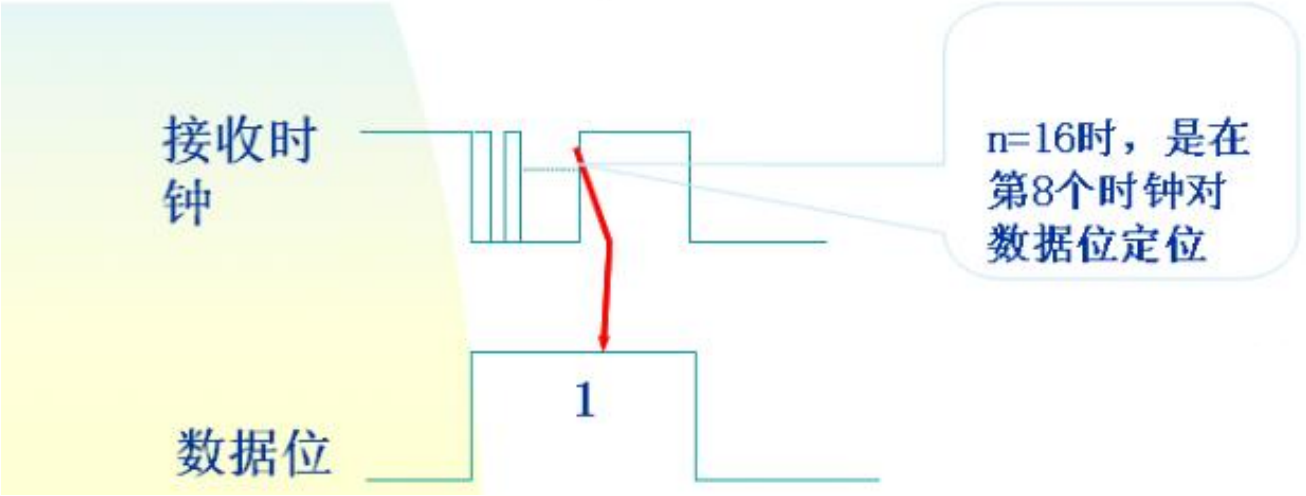
9.1 串行通信概述

F（时钟频率）＝波特率因子 × 波特率

- 波特率因子：时钟频率与数据传输率（波特率）之间的比例系数。
- 例如，波特率因子为16，则16个时钟脉冲移位1次。
- 无信号时，RxD端为高电平，一旦检测到RxD端为低电平，则启动接收控制中的内部计数器，计数到半个数位传输时间8个RxC，再次检测RxD引脚，若仍为低电平，则确认起始位到来，之后，每隔16个RxC，进行一次采样。
- 使用波特率因子高的时钟，可以使移位寄存器在信号的中间同步，而不是在信号的起始边沿同步，这样可以减少信号噪声在信号起始处引起读错数据的问题。

9.1 串行通信概述

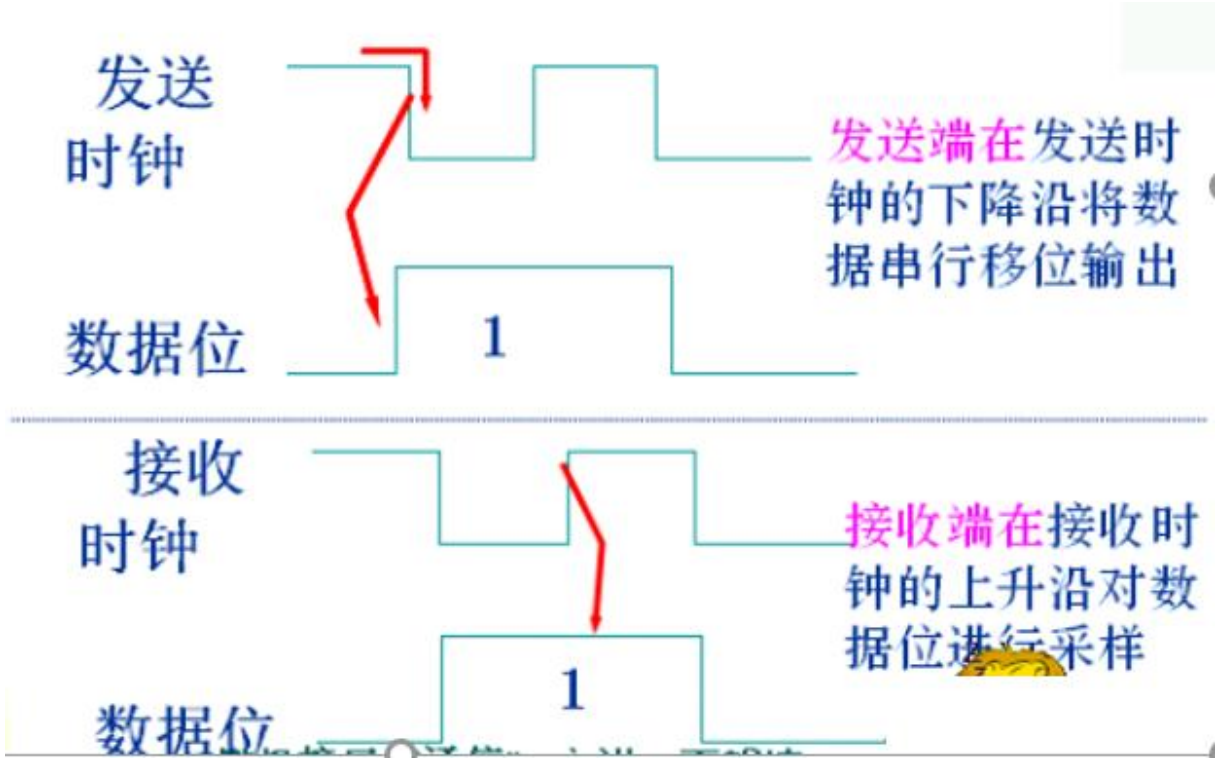

位1
位2
传送数据
起始位
0/1
0/1
定时采样时钟
8
16
16
检测到起始位前沿的时钟
确定已检测到起始位
位采样时钟


接收时钟
n=16时，是在第8个时钟对数据位定位
1
数据位

9.1 串行通信概述

收发时钟

无论是接收还是发送数据，都是在时钟的作用下进行的。二进制数据序列在串行传送过程中是以数字信号波形形式出现，无论是接收还是发送，都要对传送数据进行定位，定位方法如下：



9.1 串行通信概述

5.信号的调制与解调

- 计算机对数字信号的通信，要求传输线的频带很宽，但在实际的长距离传输中，若利用电话线来传输，电话线的频带一般都比较窄。为保证信息传输的正确，要采用**调制解调器(modem)**来实现远距离的信息传输。
- 调制解调器，顾名思义主要是完成调制和解调的功能。经过**调制器(modulator)**可把**数字信号转换为模拟信号**，经过**解调器(demodulator)**把**模拟信号转换为数字信号**。

9.1 串行通信概述

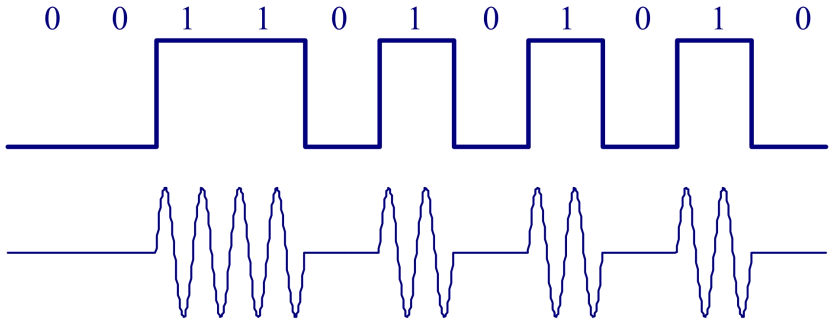
信号的调制与解调

数据通信传输的是数字信号，要求传送线的频带很宽，若传输带宽很窄，直接传输数字信号，信号就要发生畸变。 因此，需用调制器将数字信号转换成模拟信号，经传输后再用解调器将其转换成数字信号。

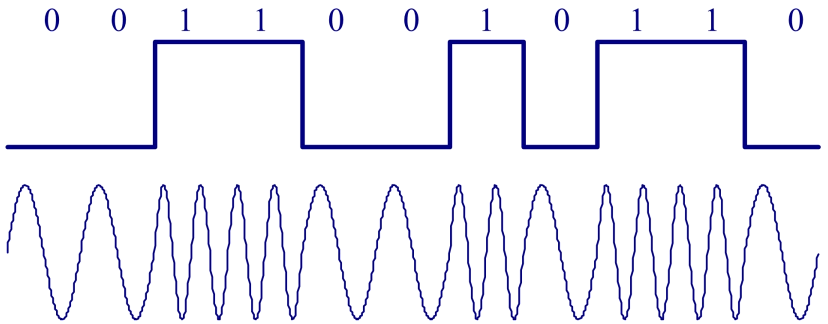
根据载波 $A\sin(\omega t + \varphi)$ 的三个参数：**幅度、频率、相位**，常用的调制技术：

- **幅度调制** Amplitude-Modulating (AM)
- **频移键控法** Frequency-Shift Keying (FSK)

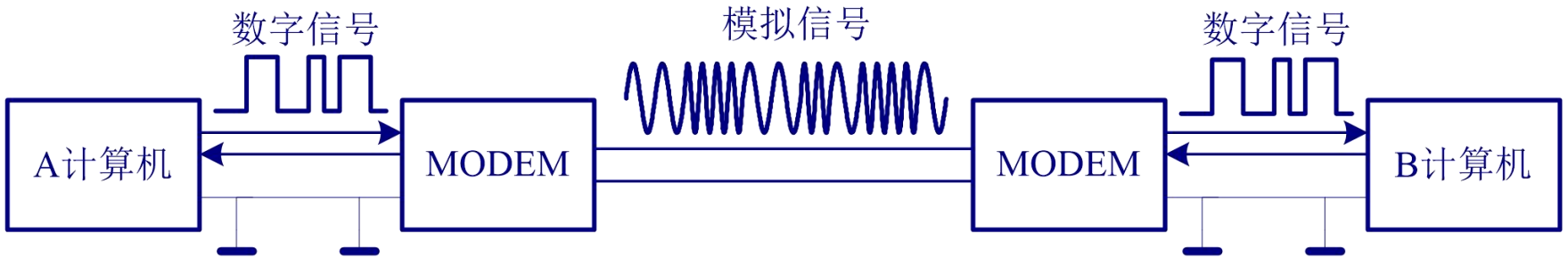
9.1 串行通信概述



用调幅正弦波表示数字1和0



用两种不同频率正弦波表示数字1和0



9.1 串行通信概述

6.串行通信接口RS-232C标准

- 美国电子工业协会EIA制定的通用标准串行接口：
  - 设计目的是用于连接调制解调器，现是最常用的串行通信接口标准之一，是PC机的标准配置。
  - 是数据终端设备DTE（例如计算机）与数据通信设备DCE（例如调制解调器）的标准接口。
  - 可实现远距离通信，也可近距离连接两台微机。
  - 属于网络层次结构中的最低层：物理层。

9.1 串行通信概述

• RS-232C的电气特性

- 232C接口采用EIA电平
  - 逻辑低电平为＋3V～＋15V
  - 逻辑高电平为－3V～－15V
  - 实际常用±12V或±15V

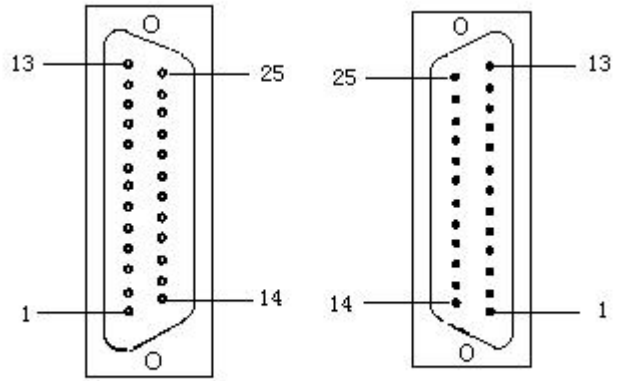
相互转换

- 标准TTL电平
  - 低电平：0V～0.8V
  - 高电平：＋2V～＋5V

9.1 串行通信概述

- **RS-232C的引脚定义**

RS-232C是一种标准接口，D型插座，采用25芯引脚或9芯引脚的连接器，如图所示。



9.1 串行通信概述

- **主要引脚定义：**

  TxD（2）：发送数据，串行数据的发送端。

  RxD（3）：接收数据，串行数据的接收端。

  RTS（4）：请求发送，当数据终端设备准备好送出数据时，就发出有效的RTS信号，用于通知数据通信设备准备接收数据。

  CTS（5）：清除发送（允许发送），当数据通信设备已准备好接收数据终端设备的传送数据时，发出CTS有效信号来响应RTS信号。

  RTS和CTS信号逻辑0为有效状态，是数据终端设备与数据通信设备间一对用于数据发送的联络信号。

9.1 串行通信概述

DTR（20）：数据终端准备好，通常当数据终端设备一加电，该信号就有效，表明数据终端设备准备就绪。

DSR（6）：数据装置准备好，通常表示数据通信设备（即数据装置）已接通电源连到通信线路上，并处在数据传输方式。

DTR和DSR信号逻辑0为有效状态，可用做数据终端设备与数据通信设备间的联络信号，例如应答数据接收。

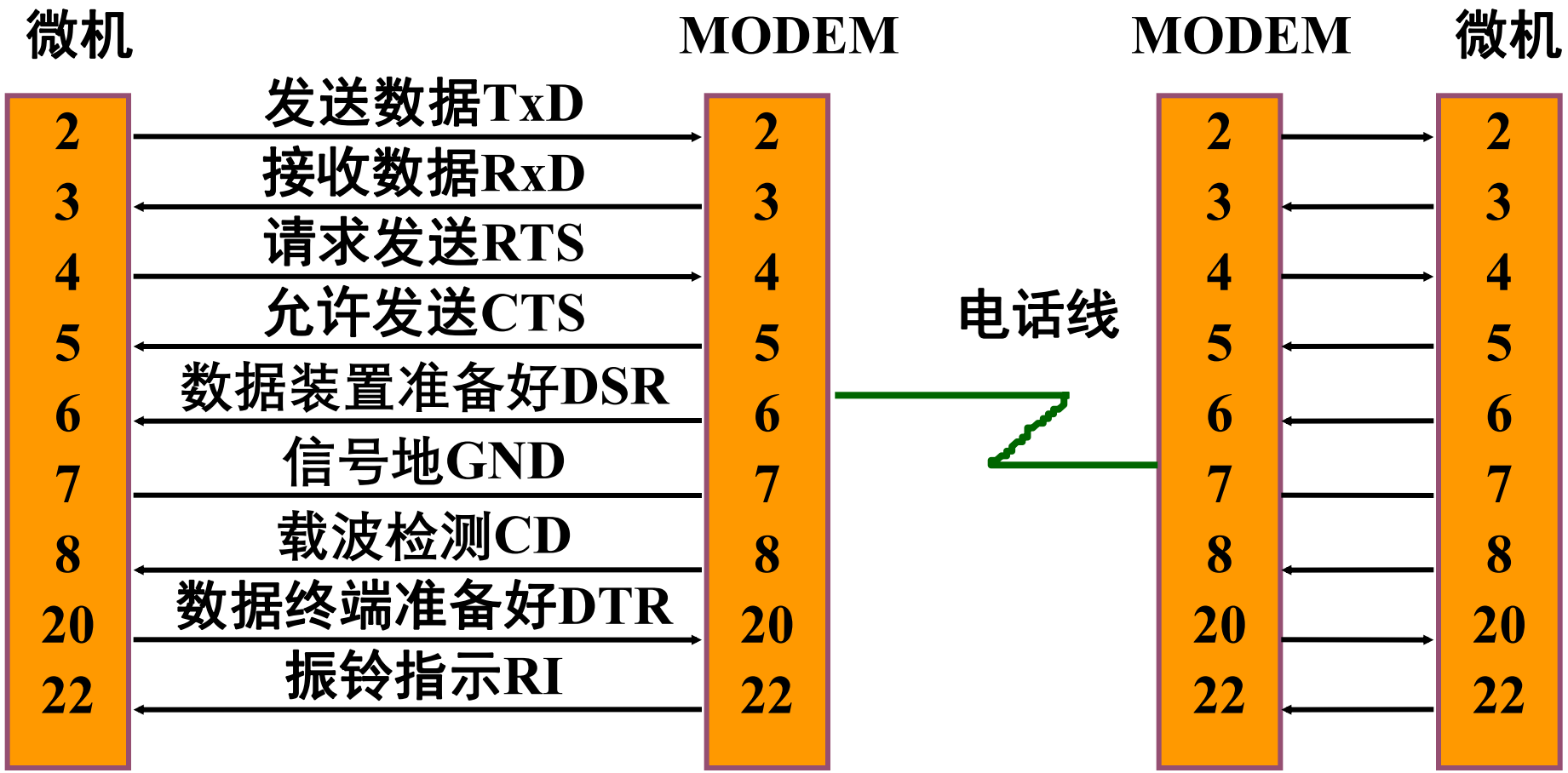
9.1 串行通信概述

GND（7）：信号地，为所有的信号提供一个公共的参考电平。

DCD（8）：载波检测，当本地调制解调器接收到来自对方的载波信号时，该引脚向数据终端设备提供有效信号。

RI（22）：振铃指示，当调制解调器接收到对方的拨号信号期间，该引脚信号作为电话铃响的指示、保持有效。

9.1 串行通信概述



连接及通信原理

9.1 串行通信概述

串行接口电路的基本任务：

- 实现数据格式化；
- 进行串并，并串转换；
- 控制数据传输率；
- 进行错误检验；
- 进行TTL与EIA电平转换；
- 提供符合EIA—RS—232C接口标准所要求的信号线。

9.2 可编程串行接口电路8251A

- **8251A的主要性能和内部结构**

- 825lA是可编程的串行通信接口芯片，是Intel公司生产的一种通用同步/异步数据收发器（USART）,它的基本性能如下：

  (1) 可工作在同步方式，也可工作在异步方式。同步方式下波特率为0～64,000波特，异步方式下波特率为0～19,200波特。

  (2) 在同步方式时，每个字符可定义为5、6、7或8位。两种方法实现同步，由内部自动检测同步字符或由外部给出同步信号。允许同步方式下增加奇/偶校验位进行校验。

9.2 可编程串行接口电路8251A

(3) 在异步方式下，每个字符可定义为5、6、7或8位，用1位作奇偶校验。时钟速率可用软件定义为波特率的l、16或64倍。另外，8251A在异步方式下能自动为每个被输出的数据增加1个起始位，并能根据软件编程为每个输出数据设置1位、1.5位或2位停止位。

(4) 能进行出错检测。带有奇偶、溢出和帧错误等检测电路，用户可通过输入状态寄存器的内容进行查询。

9.2 可编程串行接口电路8251A

常设的错误标志

✧奇偶错误

- 接收字符中“1”的个数与奇偶性不一致。

✧帧错误

- 接收的字符格式不符合规定(如无停止位等)。

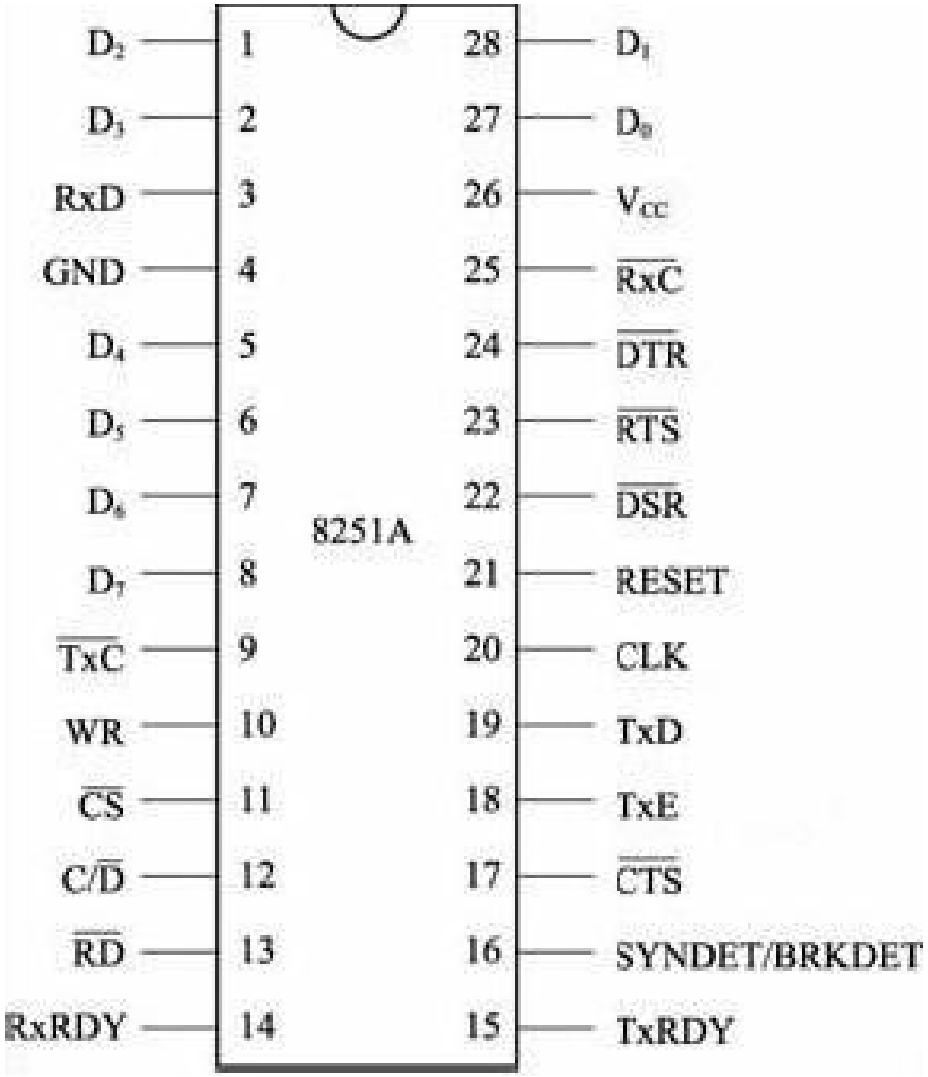
✧溢出(丢失)错误

- 接收到第二个字符的停止位时，前一个字符还未取走。

9.2 可编程串行接口电路8251A

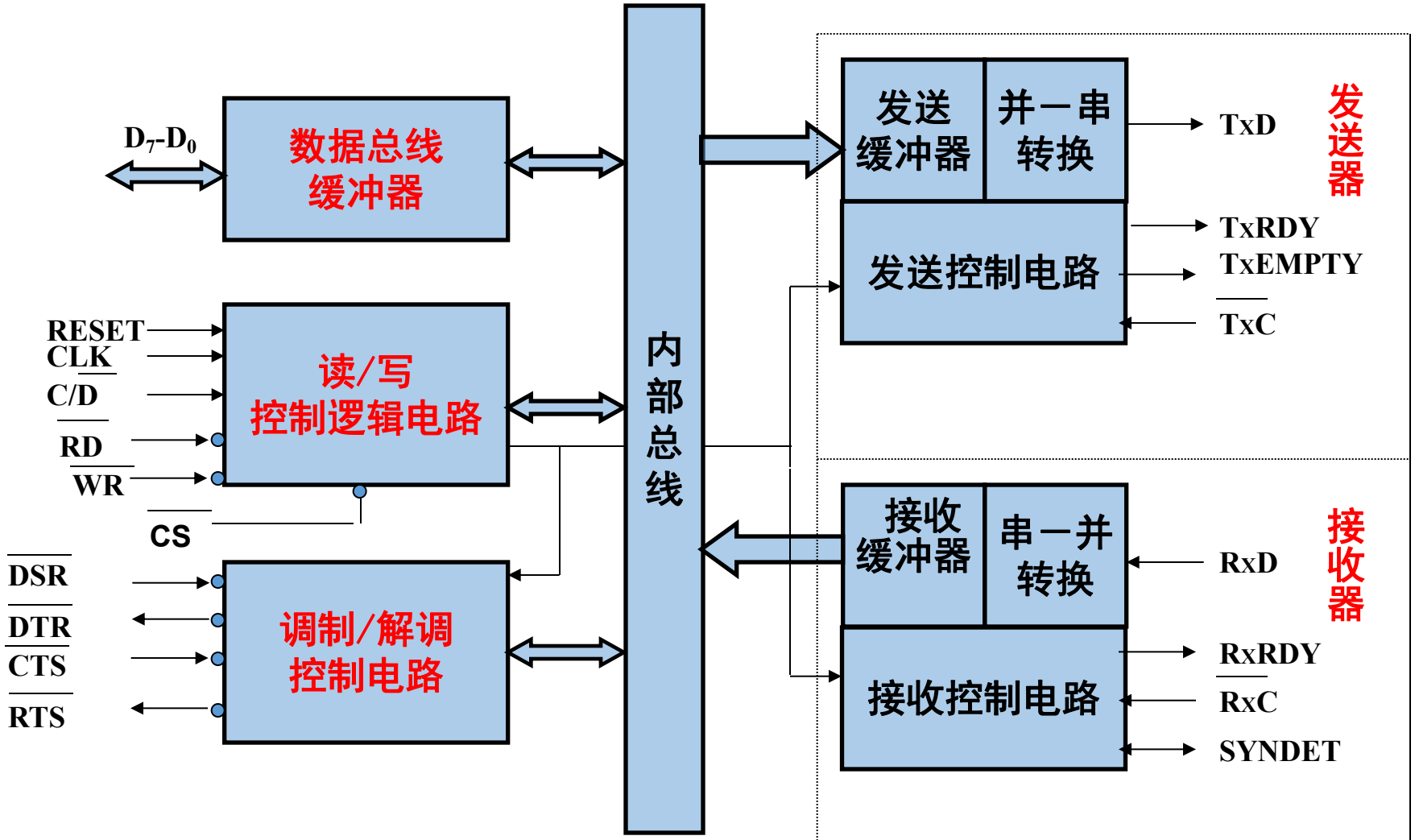
芯片封装

- 双列直插
- 28根引脚



8251A的内部结构和引脚

9.2 可编程串行接口电路8251A



8251A的内部结构框图

9.2 可编程串行接口电路8251A

（1） 数据总线缓冲器

- 数据总线缓冲器通过8位数据线D_7～D_0和CPU的数据总线相连，负责与CPU交换信息。
- 还可随时把状态寄存器中的内容读到CPU中，在8251A初始化时，分别把方式字、控制字和同步字符送到方式寄存器、控制寄存器和同步字符寄存器中。

9.2 可编程串行接口电路8251A

（2）读/写控制逻辑

8251A的控制信号与执行的操作之间的对应关系表

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	C/$\overline{D}$	执行的操作
0	0	1	0	CPU由8251A输入数据
0	1	0	0	CPU向8251A输出数据
0	0	1	1	CPU读取8251A的状态
0	1	0	1	CPU向8251A写入控制命令

9.2 可编程串行接口电路8251A

（3）接收缓冲器与接收控制器（异步方式）

- 接收缓冲器包括**接收移位寄存器**（接收R_XD管脚的串行数据）和**数据输入寄存器**（转为并行格式数据等待CPU取走）。
- 接收控制电路是用来控制数据接收工作。接收数据的速率取决于RxC引脚上接的时钟频率。异步方式下，接收时钟的频率可以是波特率的1、16或64倍，即**波特率系数（因子）**为1、16或64。
- 当CPU发出允许接收数据的命令时，接收缓冲器就一直监视着数据引脚RxD上的电平信号，一旦检测到下降沿，就启动接收过程。

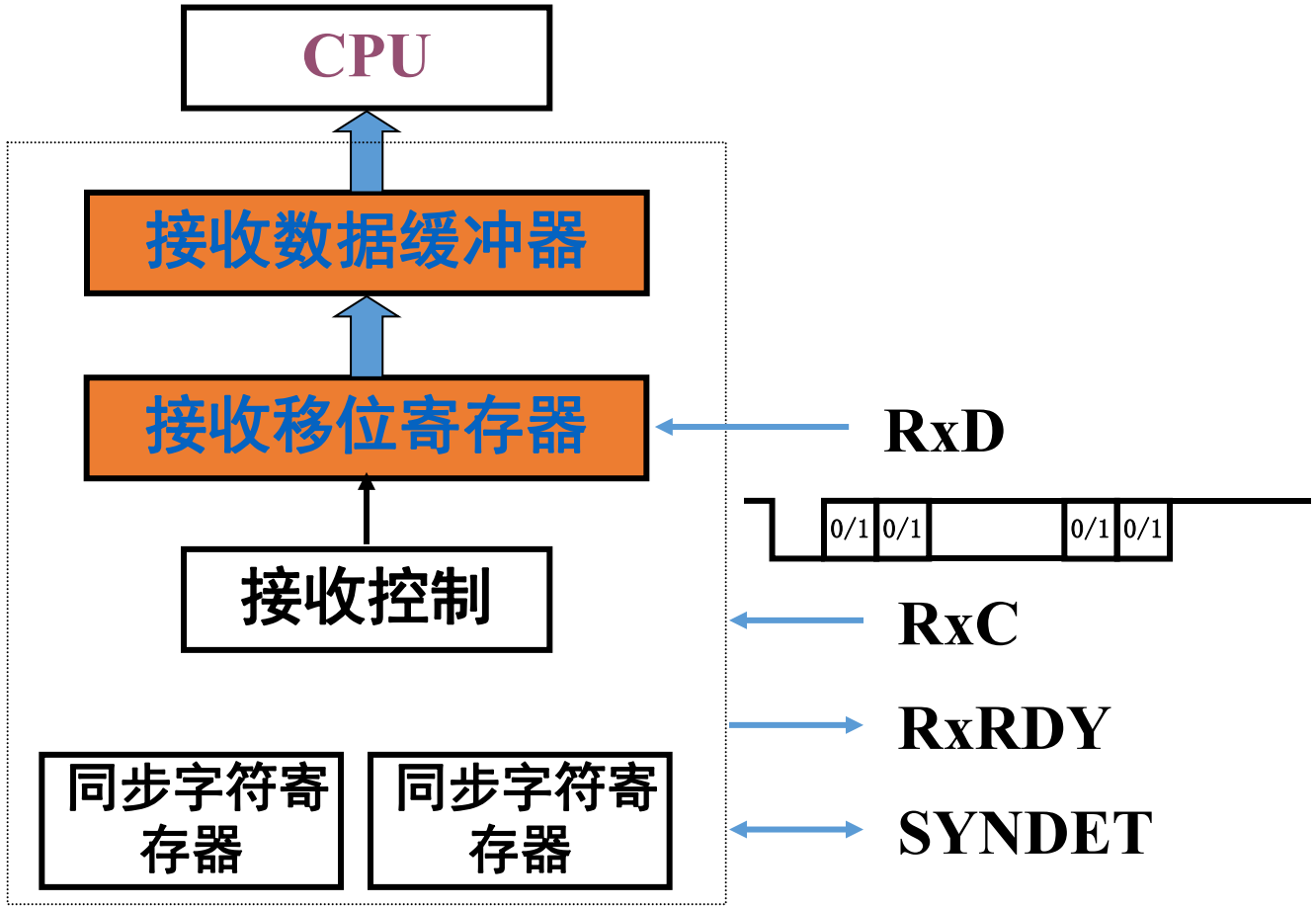
9.2 可编程串行接口电路8251A

同步传送方式分为内同步和外同步

工作于外同步：由外部电路监测同步字符，当发现同步字符后，从同步输入端SYNDET输入高电平，告知8251A，8251A就脱离对同步字符的搜索过程，高电平需维持一个接收时钟周期。

达到同步后， 8251A利用接收时钟采样RxD，接收同步帧格式数据。采得的数据送往移位寄存器，当位数达到一个字符规定的数位时，移位寄存器的内容通过片内总线送往接收数据缓冲器，同时RxRDY引脚置高电平，且状态寄存器的RxRDY位为1，表示已经收到一个可用字符。

9.2 可编程串行接口电路8251A



工作于**内同步**：CPU发出允许接收和进入**搜索**命令，监测RxD引脚，将接收的数据位送入移位寄存器，并与同步字符寄存器的内容比较，若不同，不断接收并且进行移位比较操作，直到相同出现，则**SYNDET**置高电平，表明同步实现。若为双同步，则需两个同步字符均一致。

9.2 可编程串行接口电路8251A

有关的引脚包括：

- **接收数据线RxD**
- **接收数据准备好RxRDY：**

RxRDY＝1时，表明8251A已经从串行输入线接收了一个字符，正等待CPU将此信号取走。在中断方式时，RxRDY可作为向CPU申请中断的请求信号；在查询方式时，RxRDY的状态供CPU查询之用。 CPU对接收器的读操作，将清除RxRDY信号，从而使得接收器继续滚动接收工作。

- **接收时钟RxC：**

是接收器的工作时钟，它控制8251A接收字符的速度，在上升沿采集串行输入线。在同步方式下， RxC的频率即为接收数据的波特率；在异步方式下，该频率可为波特率的1倍、16倍或者64倍。

9.2 可编程串行接口电路8251A

- **同步检测/断点检测SYNDET/BRKDET：**

内同步和外同步的检测不能同时进行

内同步SYNDET作为输出，CPU执行一次读，该信号被自动复位。

外同步时当引脚SYNDET由低电平变为高电平，使8251A在下一个RxC的上升沿开始接收字符。

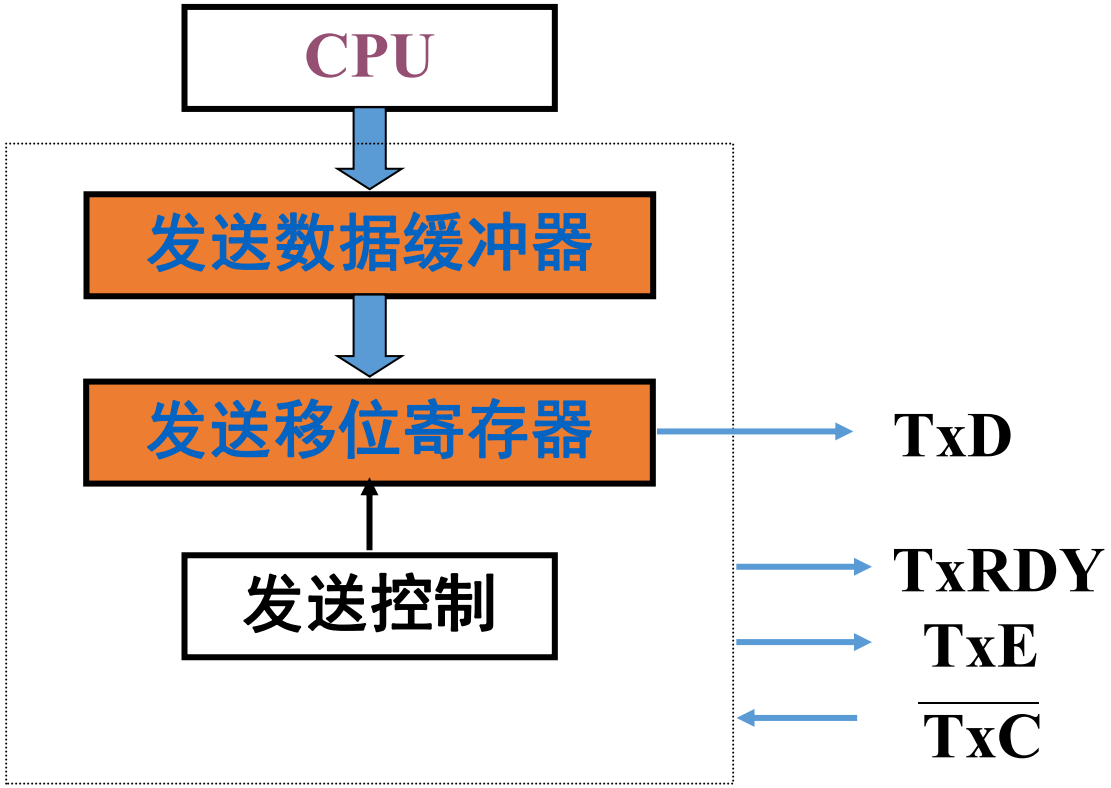
在异步工作方式下，该引脚为断点检测BRKDET。当8251A连续收到两个全“0”组成的字符，该引脚输出高电平，表明当前没有数据可读。直到收到“1”或8251A复位，BRKDET变低电平。

9.2 可编程串行接口电路8251A

(4) 发送缓冲器与发送控制器（异步方式）

- 发送缓冲器包括**数据输出寄存器**（寄存来自CPU的数据）和**发送移位寄存器**（将串行数据从T_XD管脚发送出去）。
- 发送控制电路能**按程序规定的字符格式**，给发送数据自动加上起始位、奇偶校验位和停止位对串行数据实行**逐位发送**。发送速率取决于TxC引脚上接的发送时钟频率。

9.2 可编程串行接口电路8251A



CPU用OUT指令将要发送的数据送入到8251A的数据总线缓冲器，再并行送入发送数据缓冲器中。再将数据送移位寄存器将并行数据转换为串行数据并格式化后，经TxD引脚串行输出。

9.2 可编程串行接口电路8251A

- 异步方式下：

发送控制器按照程序规定的字符格式，给发送数据加上起始位、奇偶校验位、停止位，从起始位开始，经移位寄存器移位后，从TxD引脚送出，送出频率取决于发送时钟和波特率因子。

- 同步方式下：

发送器在发送字符之前，首先送出1~2个同步字符，再逐位送出串行数据。同步发送字符之间不允许有空隙，若处于某种原因使CPU中断发送过程，8251A将不断自动插入同步字符，直到CPU送来新的数据再重新输出数据。同步传送速率等同于发送时钟频率。

9.2 可编程串行接口电路8251A

有关的引脚包括：

- 发送数据线TxD：
- 发送时钟TxC：

确定了串行数据的发送速率。若为同步方式，TxC的输入频率等于发送数据的波特率。若为异步方式，可由软件定义该时钟为波特率的1倍、16倍或64倍。

- 发送数据准备好TxRDY

当允许8251A开始发送数据，且数据总线缓冲器中的发送数据/命令缓冲器为空时，TxRDY为高电平有效，表示发送缓冲器已准备好从CPU接收数据。该信号可以作为中断申请信号或者查询信号，当CPU向8259A送出一个数据后，TxRDY被清为低电平。

9.2 可编程串行接口电路8251A

•发送器空TxE：

TxE有效表明发送移位寄存器空，也就是完成了一次发送操作，当前缓冲器已经无数据向外发送。则异步方式下，TxD输出空闲位，同步方式下送出同步字符。一旦从CPU接收到数据，TxE变低电平。

8251没有内置的波特率发生器，必须由外部产生建立波特率的时钟信号，所以TXC、RXC通常与8253连接

9.2 可编程串行接口电路8251A

（5）调制/解调器控制逻辑

- 利用8251A进行远距离通信时，发送方要通过调制解调器将输出的串行数字信号变为模拟信号，再发送出去。接收方也必须将模拟信号经过调制解调器变为数字信号，才能由串行接口接收。
- 调制解调器控制电路是专为调制解调器提供控制信号用的。

9.2 可编程串行接口电路8251A

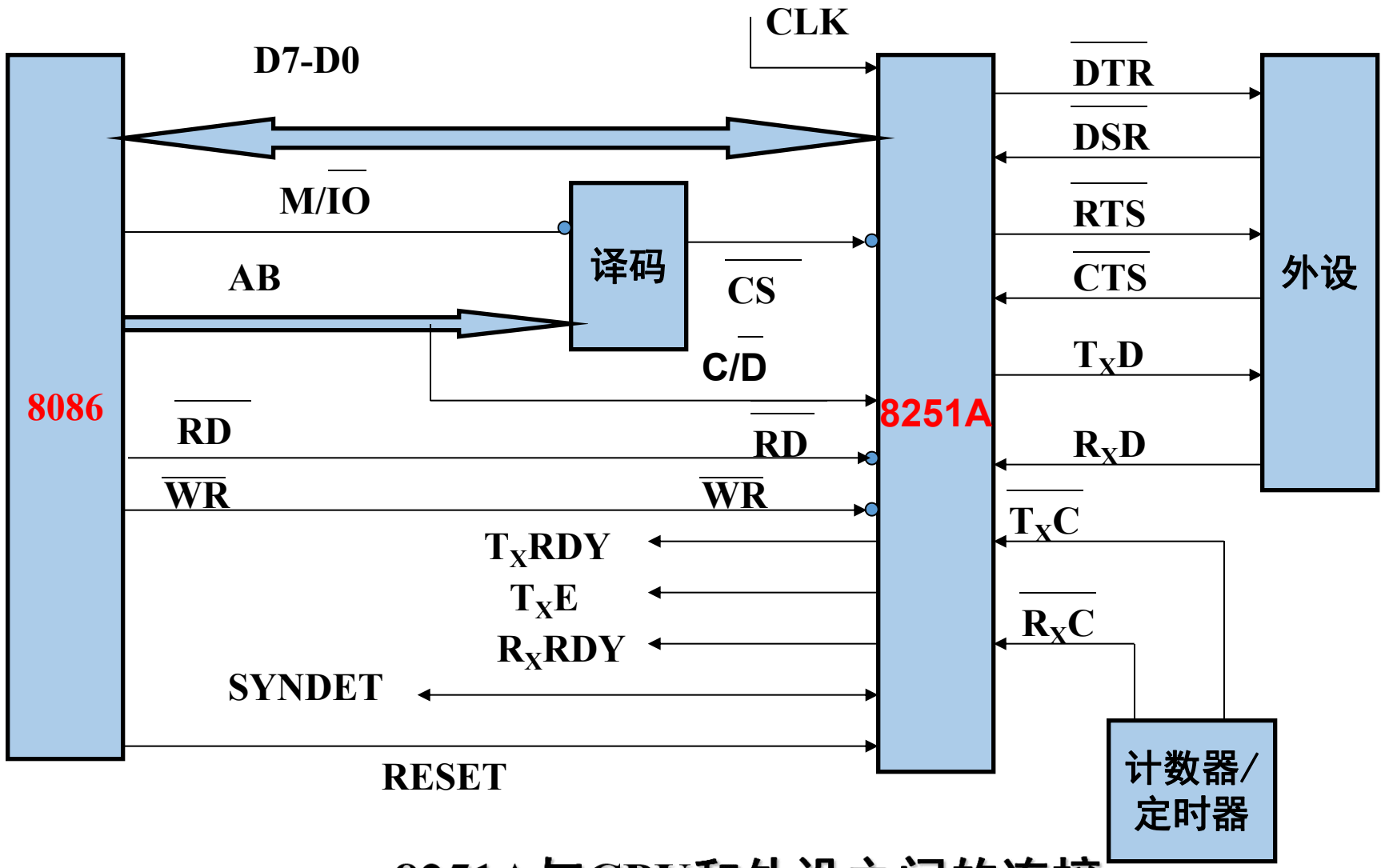
$\overline{\text{DTR}}$：数据终端准备好。告诉MODEM数据准备好。控制字中的DTR为1，就能使该引脚产生低电平有效信号。

$\overline{\text{DSR}}$：数据装置准备好。回复信号，表示MODEM准备好，可通过状态寄存器DSR位来查询， DSR=1，该引脚为低电平状态。

$\overline{\text{RTS}}$：请求发送。告诉MODEM数据准备好，可以发送。通过命令字将RTS位置1，就能使该引脚产生有效低电平信号。

$\overline{\text{CTS}}$：清除发送。MODEM对请求发送的应答信号，表明发送器可以发送数据。发送过程结束后，CTS变高。

9.2 可编程串行接口电路8251A



8251A与CPU和外设之间的连接

9.2 可编程串行接口电路8251A

1、8251A的编程地址

- 8251A只需要两个端口地址：一个用于数据端口，一个用于控制端口。数据输入输出用读信号RD和写WR信号区分；状态端口只能读不能写，控制端口只能写不能读。

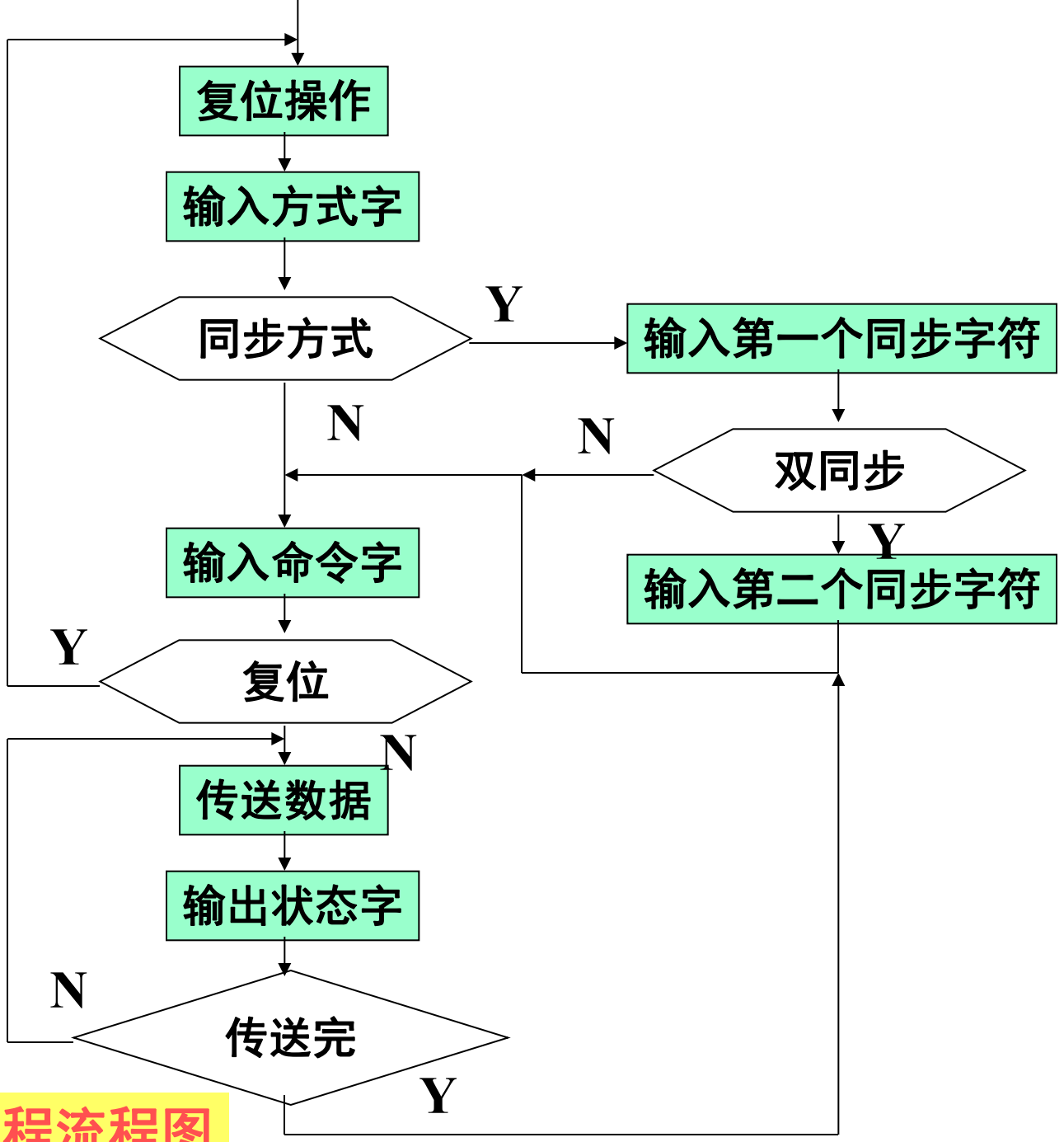
2、8251A初始化的编程流程

- 8251A是一种多功能的串行接口芯片，使用前必须向它写入方式字及命令字等，对它进行初始化编程后，才能收发数据。
- 初始化编程主要是对8251A的方式字、命令字和状态字进行编程设置。

方式字：确定8251A的工作方式；（异步，波特率，字符长度，奇偶校验）

命令字：控制8251A按方式字所规定的方式工作；（允许，禁止收发数据，启动搜索同步字符，8251复位）

状态字：了解8251A的工作状态。



编程流程图

9.2 可编程串行接口电路8251A

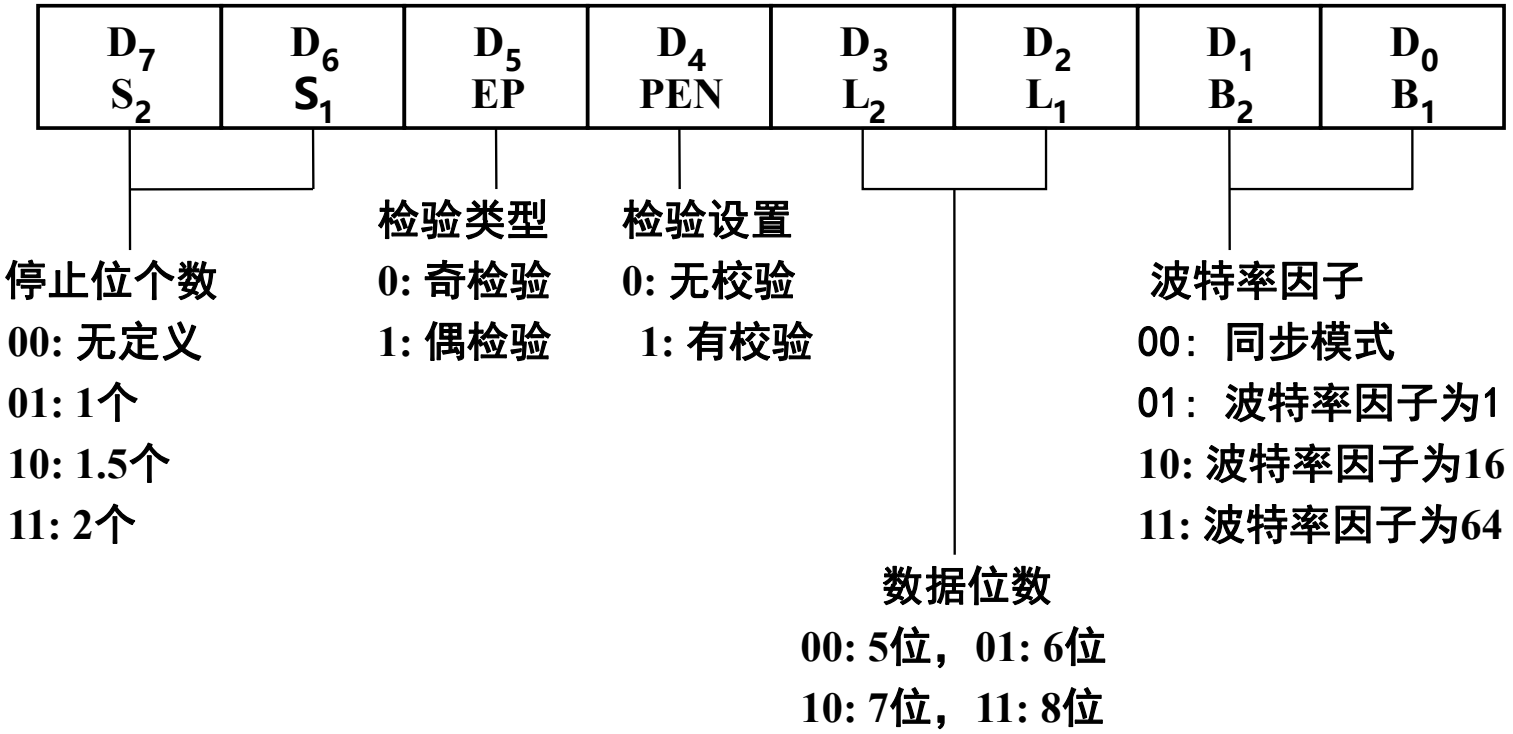
8251A的控制字

1. 方式字

- 方式字（8位）是8251A在初始化时，用来写入方式选择字用的。
- 方式选择有两种：同步方式和异步方式。
- 方式字最低2位全为0时表示是同步方式，最低2位不全为0时表示是异步方式。

9.2 可编程串行接口电路8251A

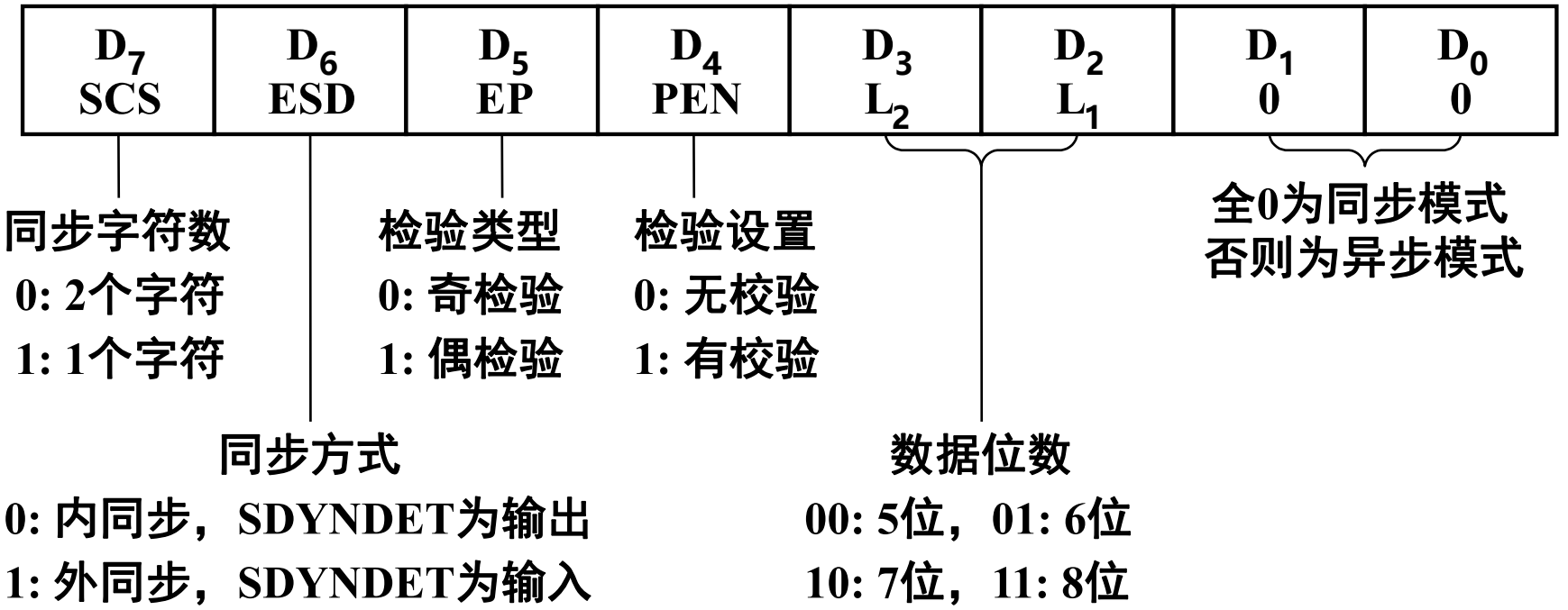
（1）8251A工作在异步方式下



8251A异步方式下方式字的格式

9.2 可编程串行接口电路8251A

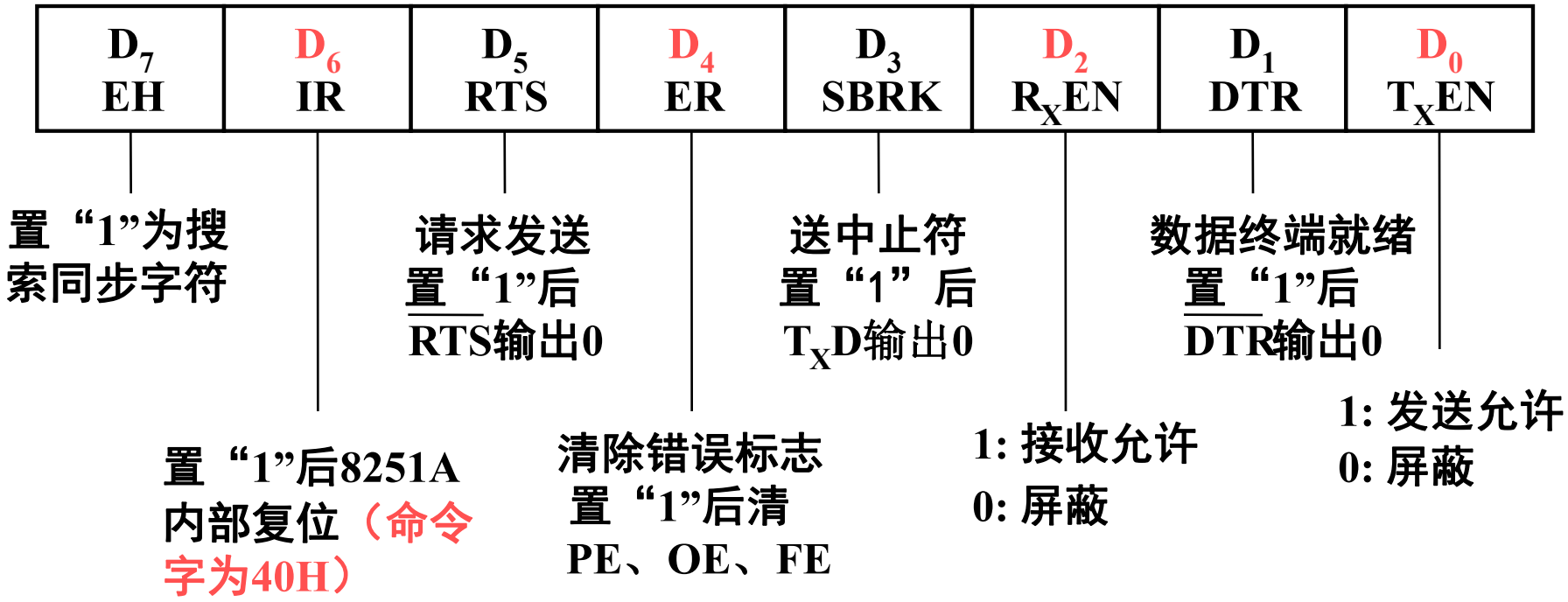
（2）8251A工作在同步方式下



8251A同步方式下方式寄存器的格式

2. 命令字

对8251A初始化时，写入了方式选择字后，接着要写入的是命令字，由命令字来规定8251A的工作状态。



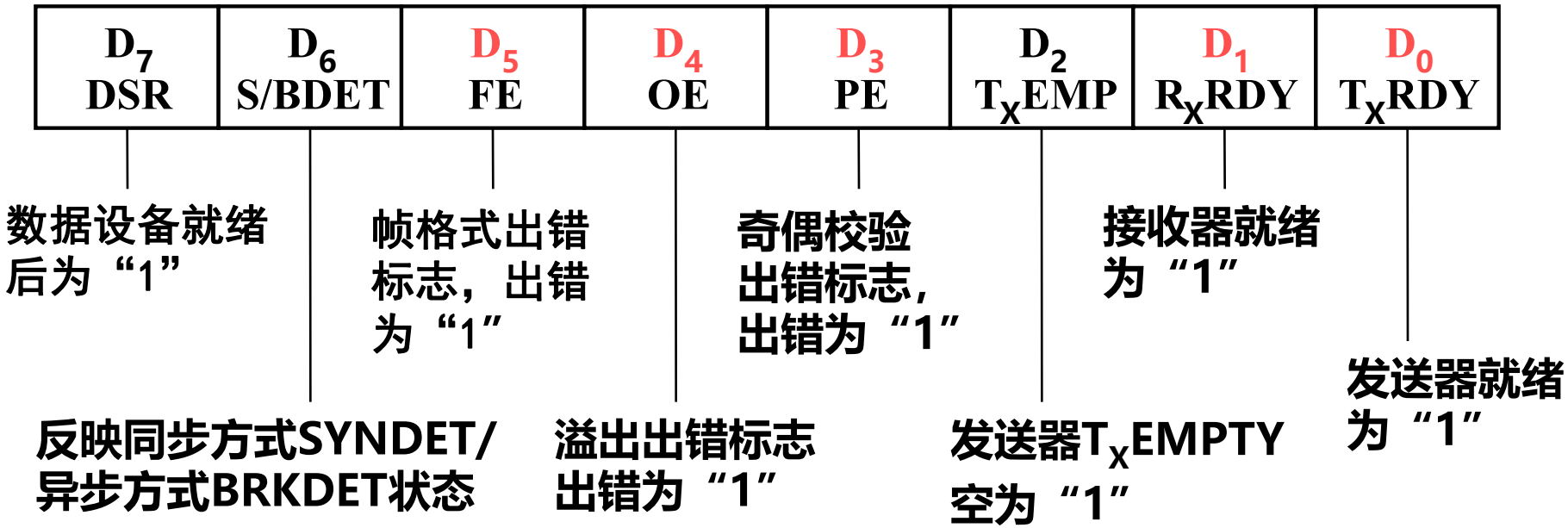
8251A命令字格式

9.2 可编程串行接口电路8251A

- 当对8251A初始化时，使用同一个地址，先写入方式字，接着写入同步字符(异步方式时不写入同步字符)，最后写入的才是命令字，这个顺序不能改变，否则将出错。
- 而初始化以后，再通过这个地址写入的字都是命令字，因此命令字可以随时写入。
- 如果要重新设置工作方式，写入方式字，必须先要将控制寄存器的D6位置1（命令字为40H），进行内部复位返回到初始化前的状态。当然，外部的RESET也可使8251A复位，而在正常的传输过程中D6=0。

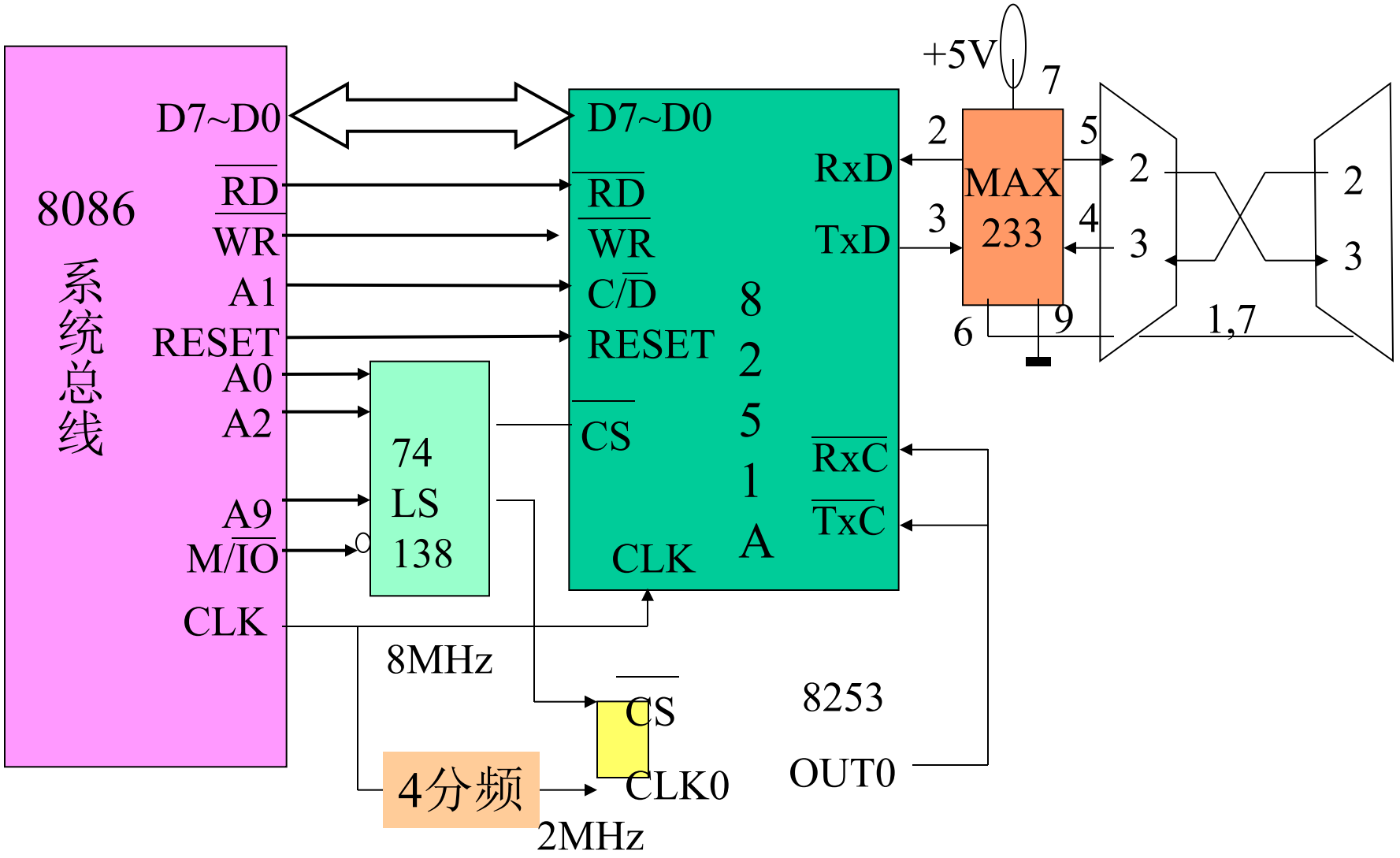
3. 状态字

状态字（8位）是只读的。CPU可用IN指令来读取状态寄存器的内容。每位的定义如下：



8251A状态字格式

9.3 8251A应用举例



利用RS-232C近距离串行通信。

9.3 8251A应用举例

在实际使用中，当未对8251A设置方式控制字时，如果要使8251A复位，一般采用先送3个“00H”，再送一个40H（命令控制字，IR＝1）的方法，这是8251A的编程约定。其实现程序如下：

```
        XOR AX, AX
        MOV CX, 0003
        MOV DX, PORTC
NEXT:   OUT DX, AL
        CALL DELAY
        LOOP NEXT
        MOV  AL, 40H
        OUT  DX, AL
        CALL DELAY
          ……
```

9.3 8251A应用举例

例9.1 8251A工作在异步方式下的初始化编程。要求字符用7位二进制表示，采用奇校验，1.5个停止位，波特率系数为16。清除出错标志，让各出错标志处于初始状态，使请求发送有效电平，命令控制字使数据终端就绪(DTR)处于有效电平，发送允许和接收允许均为有效电平。设端口地址为208H和20AH

9.3 8251A应用举例

初始化程序：

```
        XOR AX, AX
        MOV CX, 0003
        MOV DX, PORTC
NEXT:   OUT DX, AL
        LOOP NEXT
        MOV  AL, 40H
        OUT  DX, AL
        MOV  AL，9AH    ；方式字准备
MOV  DX，20AH
OUT  DX，AL   ；写入方式字
MOV  AL，37H    ；命令字准备
OUT  DX，AL   ；写入命令字
```

9.3 8251A应用举例

例9.2 8251A工作在同步方式下的初始化编程。要求采用内同步方式，同步字符为两个相同的16H值，字符长度为8位二进制，采用偶校验，要求对同步字符进行检索，复位3个出错标志，允许发送和接收，命令控制字使数据终端就绪(DTR)处于有效电平。

初始化程序：

```
        XOR AX, AX
        MOV CX, 0003
        MOV DX, PORTC
NEXT:   OUT DX, AL
        LOOP NEXT
        MOV  AL, 40H
        OUT  DX, AL
        MOV  AL，3CH    ；方式字准备
        MOV  DX，20AH
        OUT  DX，AL    ；写入方式字
        MOV  AL，16H     ；同步字符准备
        OUT   DX, AL     ；写入第一同步字符
        OUT  DX, AL     ；写入第二同步字符
        MOV  AL，97H     ；命令字准备
        OUT  DX，AL    ；写入命令字
```

9.3 8251A应用举例

8251与CPU的数据交换

查询方式/中断方式

- 采用查询方式，在数据交换前应读取状态寄存器。
- 状态寄存器D0=1(TxRDY=1)，CPU可以向8251数据端口写入数据，完成串行数据的发送
- 状态寄存器D1=1 (RxRDY=1)，CPU可以从8251数据端口读出数据，完成一帧数据的接收
- 采用中断方式则用TxRDY、RxRDY引脚信号提出中断申请

9.3 8251A应用举例

例9.3 8251设计要求：波特率=9600，波特率系数=16，定时器分频系数（定时器2MHz）？

OUT0输出频率=9600*16=0.1536MHz

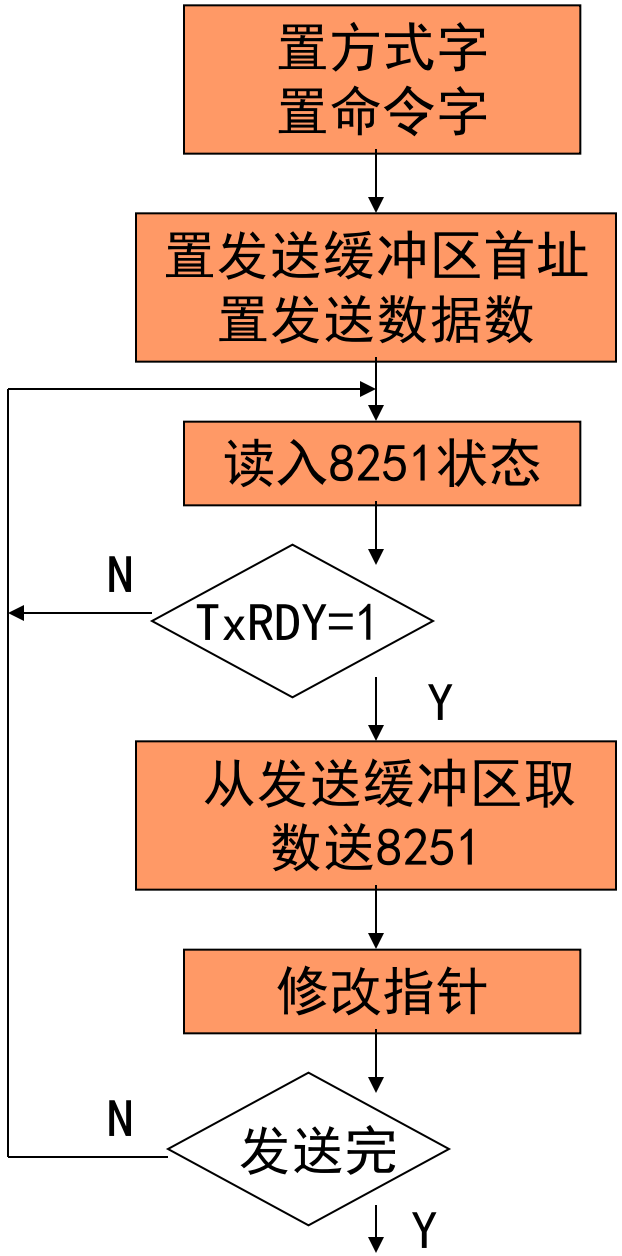
分频系数n=2MHz/0.1536MHz=13

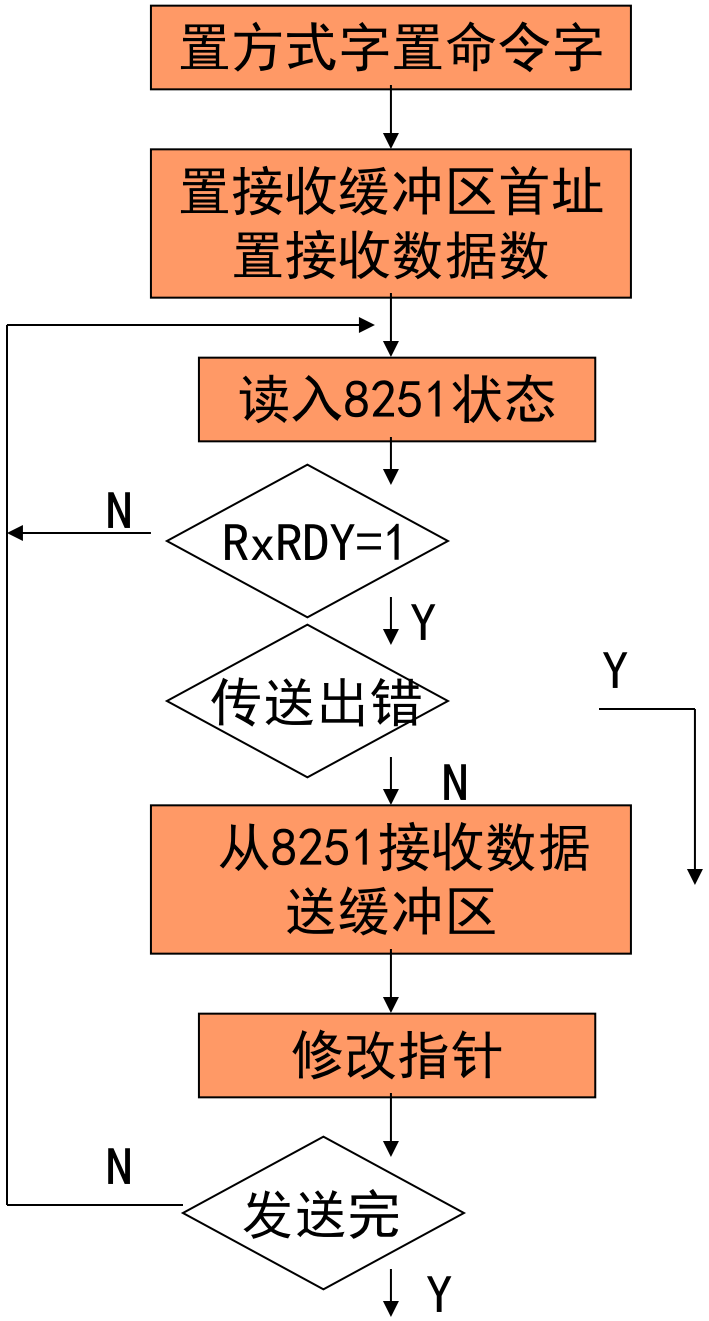
例9.4 设8251A数据口和控制口地址分别为1F0H和1F2H，异步方式，7位数据，1位停止位，偶校验，波特率系数16，

方式字：7AH

发送与接收程序如下：

```
BEG-T:  MOV   DX，1F2H
        MOV  AL, 40H
        OUT  DX, AL
        MOV   AL，7AH
        OUT   DX，AL
        MOV   CX，02H
D1：    LOOP  D1
        MOV   AL，11H
        OUT   DX，AL
        MOV   CX，02H
D2：    LOOP  D2
        LEA   DI，BUFF-T
        MOV   CX，COUNT-T
NEXT-T：MOV   DX，1F2H;
        IN    AL，DX
        TEST  AL，01H
        JZ      NEXT-T
        MOV   DX，1F0H
        MOV   AL，[DI]
        OUT   DX，AL
        INC     DI
        LOOP  NEXT-T
              …
```






```
BEG-R:   MOV   DX，1F2H
         MOV  AL, 40H
         OUT  DX, AL
         MOV   AL，7AH
         OUT   DX，AL
         MOV   CX，02H
D3:      LOOP  D3
         MOV   AL，14H
         OUT   DX，AL
         MOV   CX，02H
D4:      LOOP  D4
         LEA   DI，BUFF-R
         MOV   CX，COUNT-R
NEXT-R:  MOV   DX，1F2H;
         IN    AL，DX
         TEST  AL，02H
         JZ    NEXT-R
         TEST  AL，38H
         JNZ     ERROR
         MOV   DX，1F0H
         IN    AL，DX
         MOV   [DI]，AL
         INC    DI
         LOOP   NEXT-R
```