**实验16**

**串口通信**

# **实验介绍**

通信协议是一种规定通信双方在信息交换过程中所采用的语法和语义规则的约定。 它定义了通信的格式、顺序、错误检测和纠正方法，以及参与通信实体的行为。 通信协议使得不同系统或设备能够有效地交换信息，确保了彼此通信 过程的正确性、可靠性和完整性。

在日常生活中，通信协议的应用场景已经十分广泛。例如， TCP/IP 协议是互联网上最常用的通信协议，它用于定义数据在网络上的传输方式， 包括数据包的格式、传输控制等；以太网协议是在局域网中广泛应用的通信协议； 蓝牙协议用于在短距离无线通信中连接设备，例如连接手机和耳机、键盘等； 串口协议用于在设备之间通过串口进行数据传输，如计算机核心处理器与外部设备的连接。

本实验中，我们将带大家实现一个基础的串口通信协议。

# **实验内容**

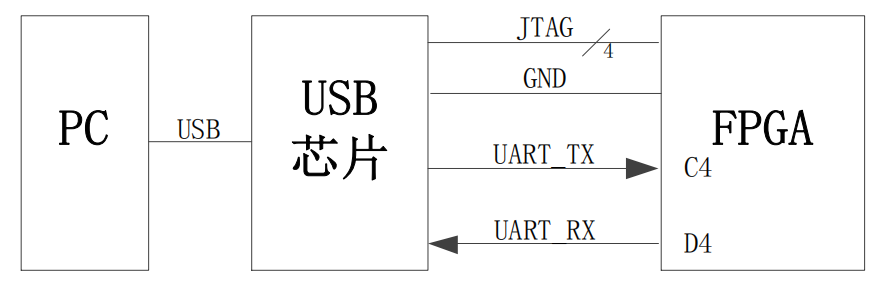
## 1 协议说明

从广义上来说，采用串行接口进行数据通信的接口都可以称为串口，如 SPI 接口、IIC 接口等，但我们所说的串口一般是指通用异步收发器（Universal Asynchronous Receiver/Transmitter），简称 UART，主要包含 RX、TX、GND 三个接口信号，其中 GND 为共地信号，TX、RX 信号负责数据的发送和接收。在嵌入式系统开发中，串口是一种必备的通信接口，在系统开发测试阶段和实际工作阶段都起着非常重要的作用。

通信协议是指双方实体完成通信或服务所必须遵循的规则和约定。在串口通信中，信道（也就是连接我们和开发板的网络通道）中传输的内容是特定的串行数据。而这些数据是无法被我们直接阅读的，因此就需要一些特定的操作，将我们日常使用的二进制数据转换成特定的串行数据后再进行发送，以及对接收到的串行数据解码后得到二进制数据。

除了编码和解码功能，通信协议同时还规定了双方通信的一些准则。即：如何知道此时信道中的数据是有效的数据、如何确定每一段数据的范围、双方的通信频率等。因此，通信协议是一段通信的核心，正确实现了通信协议也就保证了通信的正确性。

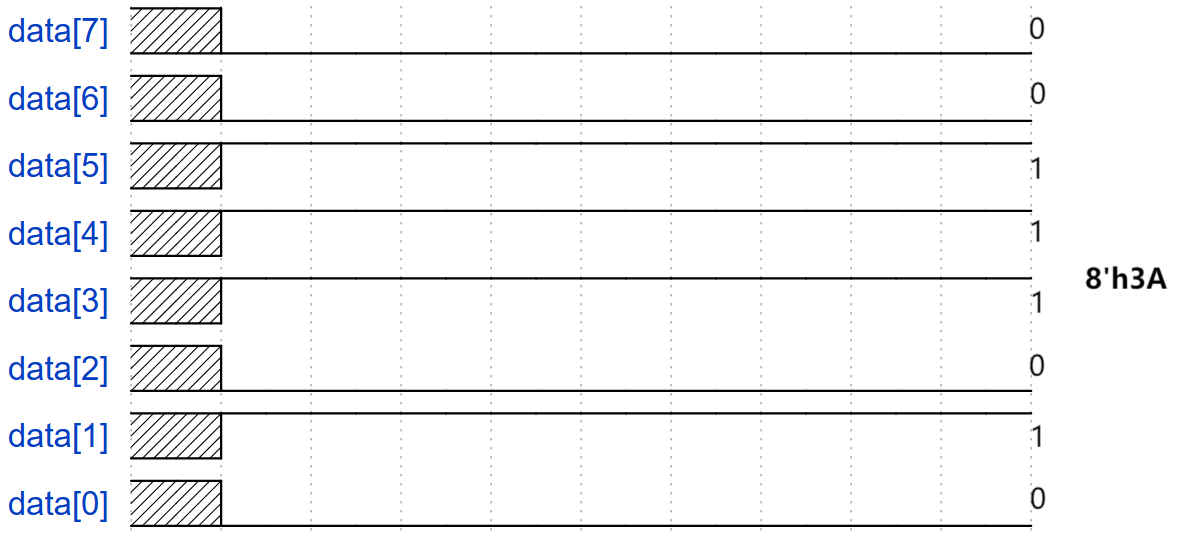
在 Nexys4DDR 开发板中，UART 通信与 USB 烧写功能集成在了一个 microUSB 接口中，如下图所示：

[](https://soc.ustc.edu.cn/Digital/lab8/figs/uart/uart.png)

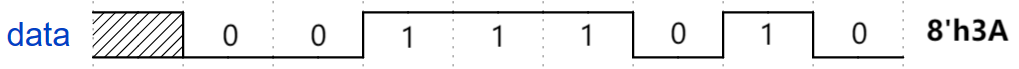
用户将 Nexys4DDR 开发板与 PC 设备相连，并连接电源之后，便可以在 PC 端的设备管理器中发现对应的串行接口。在 FPGAOL 平台上，我们已经在浏览器界面中集成了一个串口通信窗口，该窗口在浏览器端实现了串口通信协议，因此用户只需要在开发板上实现串口通信协议即可实现串口通信的功能。

回到我们的串口协议上。在串口通信中，所有的数据都是通过两个 1bit 位宽的信号 TX、RX 传输的。其中 TX 信号用于传输从用户到开发板的数据，RX 信号用于传输从开发板到用户的数据。两个信号（或者叫两条信道）之间是彼此独立的，也就是说，我们可以同时向开发板发送数据并接收来自开发板的数据。

假定现在有一个 8bits 位宽的数据 8'H3A。如果使用并行数据传输，我们将采用 8bits 位宽的通道同时传输所有的位；如果使用的是串行数据传输，我们将一位一位地传输数据的所有位。下面的波形图展示了二者的区别：

[](https://soc.ustc.edu.cn/Digital/lab8/figs/uart/mul.png)

并行数据传输

[](https://soc.ustc.edu.cn/Digital/lab8/figs/uart/sing.png)

串行数据传输

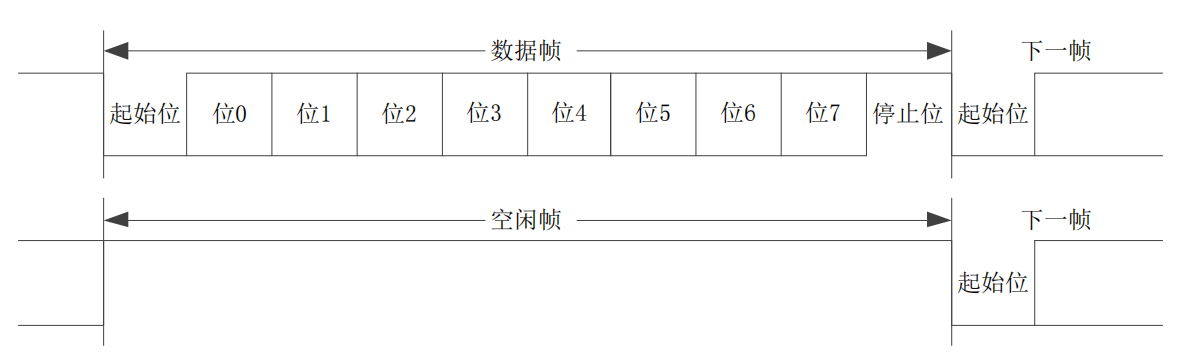
其中，波形图的横轴表示时间，因此左侧的波形比右侧的波形更早发出。我们可以直观感受到串口中「串」的含义。

为了规范串口的通信过程，我们很自然地提出了下面的问题：

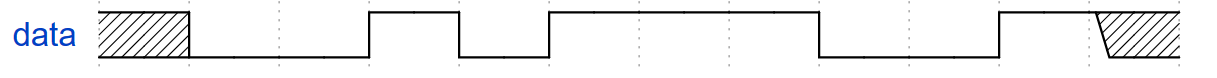
* 双方的工作频率需要一致。因为串行数据发送时每一位占据的时间应当是固定的，如果双方工作的频率不一致，就会导致解码时出现错位，也就无法得到正确的信息。
* 需要有特定的标识位表明数据的范围。发送序列可以看作一串长长的 0-1 串，我们需要对该串进行正确的分割，从而获取正确的数据。这就要求我们在通信时添加一些固定格式的位用于识别。

串口协议中支持的数据收发频率（又称波特率，bps）有多种，如 9600、19200、115200、256000 等，以 115200 为例，其表示 1s 时间内可以传送 115200 位的数据。在本实验中，我们约定使用 115200 的波特率进行讲解和设计。

串口的收发信号采用相同的数据格式，我们称之为数据帧；当没有数据需要发送时，可以发送空闲帧。在默认情况下，我们认为信道中始终为高电平，即一直发送空闲帧，仅在有数据传输时才会出现低电平。数据帧和空闲帧的格式如下图所示：

[](https://soc.ustc.edu.cn/Digital/lab8/figs/uart/frame_1.png)

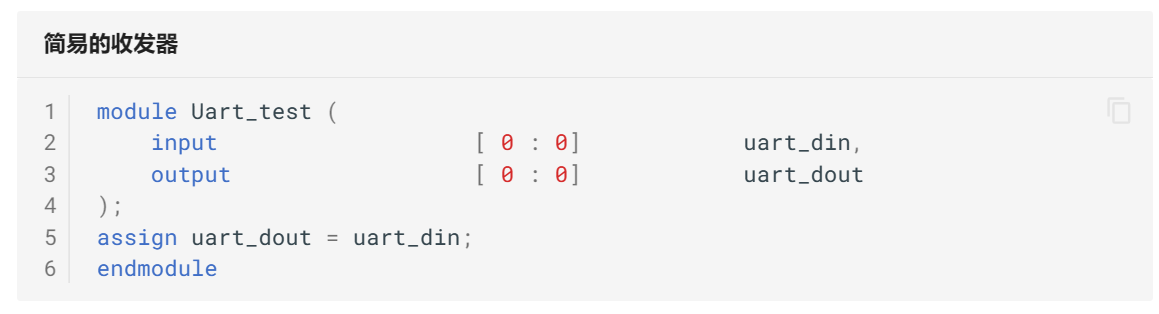
每一数据帧都由「起始位 + 数据位 + 停止位」三部分组成，相邻两个数据帧之间可以插入始终为高电平信号的空闲帧。串口通信协议规定：数据帧起始位为低电平、停止位为高电平，数据位长度可选择 5~8 中的任意数字。本实验中选择「1 位起始位 + 8 位数据位 + 1 位停止位」的数据帧结构。因此，8'h3A 可以转换为如下的波形图：

[](https://soc.ustc.edu.cn/Digital/lab8/figs/uart/example_msg.png)

## 2 简单使用

### ****2.1 串口回显****

下面，我们通过一个简单的程序来感受串口的使用过程。在 FPGA 内，将 UART\_TX（C4 引脚）输入到 FPGA 内的信号直接赋值给 UART\_RX（D4 引脚），这样我们就可以实时接收刚刚发送出去的数据（在 FPGA 侧不进行解码，直接原样返回，我发送给我自己.jpg）。对应的 Verilog 代码为：



与之对应的约束文件为

##USB-RS232 Interface

set\_property -dict { PACKAGE\_PIN C4 IOSTANDARD LVCMOS33 } [get\_ports { uart\_din }]; #IO\_L7P\_T1\_AD6P\_35 Sch=uart\_txd\_in

set\_property -dict { PACKAGE\_PIN D4 IOSTANDARD LVCMOS33 } [get\_ports { uart\_dout }]; #IO\_L11N\_T1\_SRCC\_35 Sch=uart\_rxd\_out

请在 Vivado 中创建一个项目，正确添加设计文件和约束文件，并烧写比特流，在 FPGAOL 平台上验证串口回显程序的功能。在正常情况下，在发送窗口输入数据并点击发送，之后会在接收窗口收到同样的内容。

### ****2.2 串口发送模块****

接下来，我们将实现串口的发送模块。简单来说，发送模块需要将来自开发板的 8 位数据转换成符合串口协议的数据帧。相比接收模块，发送模块需要考虑的事情更少，只需要将将对应位上的信号维持一定的时间即可，而无需考虑采样等实际的细节。

开发板的时钟频率为 100MHz，因此当波特率为 115200 时，发送时每一位持续的时间约为 1115200/1100×10^6≈868 个时钟周期。基于此，我们可以使用分频计数器在 0∼867 之间进行计数，保证数据帧中的每一位都能持续 868 个时钟周期。

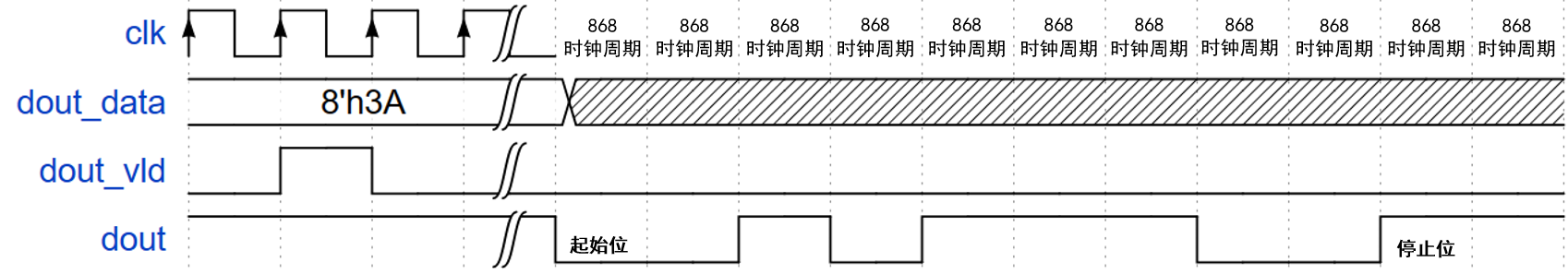
Send 模块的输入输出端口介绍如下：



其中

* dout 信号直接连接 UART 的 RX 端口，用于向用户发送来自开发板的串行数据；
* dout\_vld 信号用于指示当前 dout\_data 是否有效，持续一个时钟周期；
* dout\_data 信号用于存储即将发送的 8 位数据。

假定某时刻 dout\_data 的值为 8'h3A，那么 Send 模块就需要在一定的时钟周期内给出下图所示的结果：

[](https://soc.ustc.edu.cn/Digital/lab8/figs/uart/uart_send_example.png)

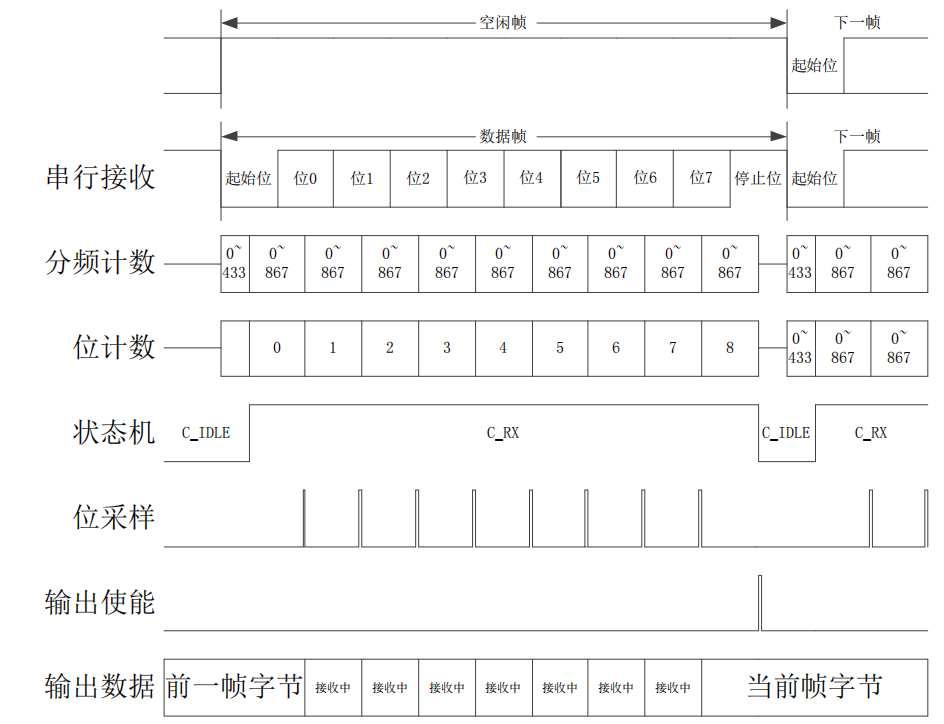
图中前半部分的间隔为 1 个时钟周期，外界的其他模块将 dout\_data 的值准备好后，给出一个周期的 dout\_vld 信号，随后 Send 模块开始进行转换。在转换的过程中，dout\_data 的改变并不影响转换的结果，也就是说 Send 模块在接收到 dout\_vld 信号后会暂存此时 dout\_data 的结果。

经过一定的时间后，Send 模块开始输出我们期待的数据帧。空闲帧始终为高电平信号，而数据帧的首位为低电平（起始位）。随后，Send 模块按照从低位到高位的顺序逐位发送 8'h3A 的结果，每一位持续 868 个时钟周期，最后以一个停止位结束该数据帧的发送过程。完成发送后，dout 信号将继续保持高电平，即持续发送空白帧。

基于上面的过程，我们可以分析出，发送模块需要一个状态机和对应的分频计数器、位计数器。分频计数器在发送状态下在 0~867 的范围内计数。每次分频计数器达到 867 时，位计数器就自增 1。位计数器用于指示当前发送的位的编号。我们一共有 1 + 8 + 1 = 10 位数据需要发送，因此位计数器的范围为 0 ~ 9。在发送时，我们需要根据位计数器的值确定当前发送的内容是起始位、中间数据还是终止位。

### ****2.3 串口接收模块****

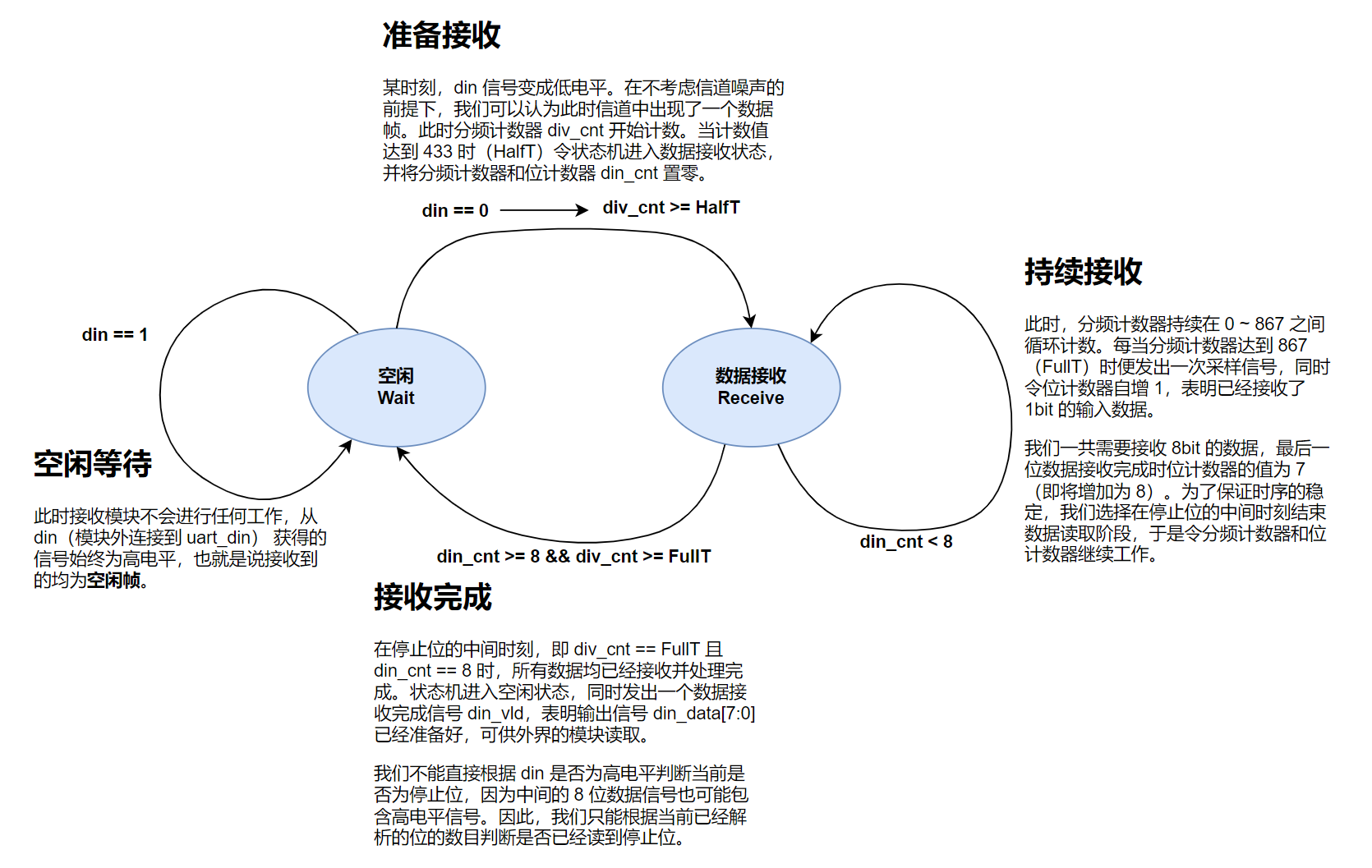
接下来，我们将实现一个简单的数据接收模块，在开发板上将 UART\_TX 发来的数据进行串并转换，并将结果输出到数码管上。模块工作流程可通过以下时序图说明：

[](https://soc.ustc.edu.cn/Digital/lab8/figs/uart/frame_2.png)

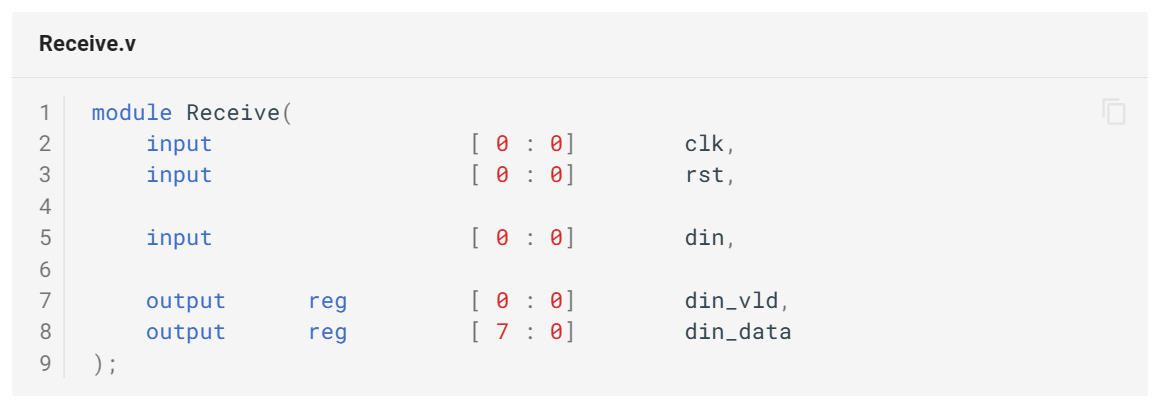
我们同样可以使用分频计数器进行计数：当接收信号为 0 时（起始位），分频计数器开始计数；当计数值达到 433 时（起始位中间时刻），状态机从空闲状态跳转到接收状态。

接下来，计数器将在 0~867 之间循环计数，同时启用位计数器进行位计数。从图中可以看出，当计数器值为 867 时，对应的就是串行接收信号某一位的最佳采样时刻（处于该位的中间时刻）。此时即采样信号接收 1bit 的数据，并保存到输出数据（8bits）的对应位中。当位计数器达到 8 时，表明当前的 8bits 数据已经接收完毕，将输出使能信号置位为高电平，并将接收到的整个字节输出出去。这样就完成了一个数据帧的串并转换过程。

下面的状态机展示了上面叙述的流程：

[](https://soc.ustc.edu.cn/Digital/lab8/figs/uart/uartFSM.png)

我们将使用 Verilog 实现串口接收模块。首先，我们明确模块的输入输出端口：



其中

* din 信号直接连接 UART 的 TX 端口，用于接收来自用户的串行数据；
* din\_vld 信号用于指示当前 din\_data 是否有效（数据帧内的 8 位数据全部解码完成），持续一个时钟周期；
* din\_data 信号用于存储数据帧内的 8 位数据。

# **思考与练习**

1. 实现Send模块的相关内容。
2. 实现Receive模块的相关内容。