北京大学
PEKING UNIVERSITY

10010010

嵌入式系统编程与实践

3-嵌入式编程基础-接口

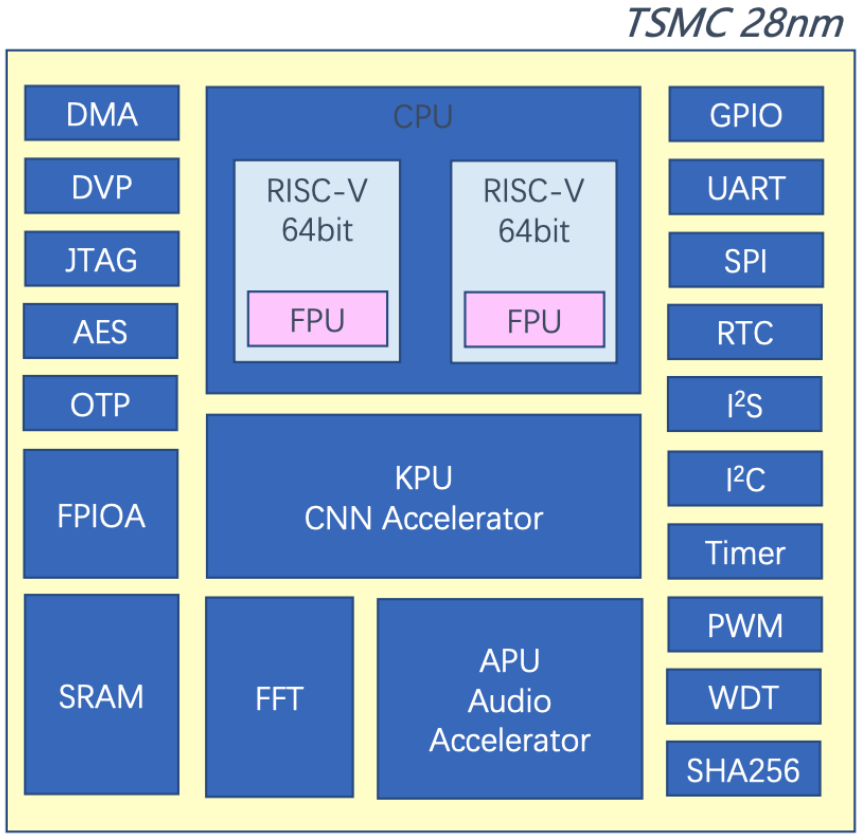
燕博南

2024秋

今天的第一个小任务

- 分配开发板（每人一份，实名发放）
  - 注：学期末回收（除了互评第一）
  - 注：注意不要损坏
- 分配实践课带班

接口无处不在



片间接口：

- UART
- SPI
- I^2C

其他：

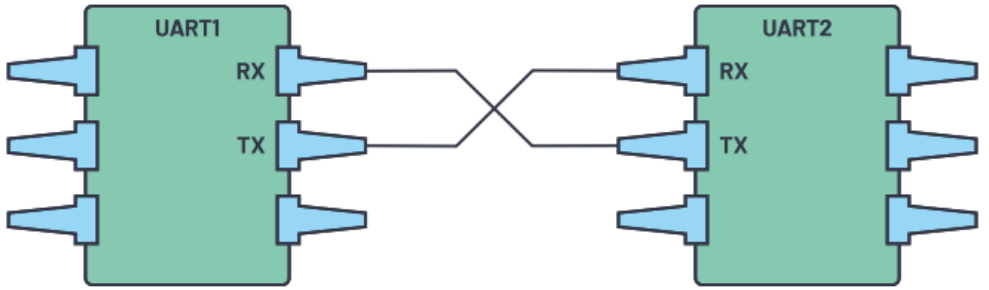
- PCIe
- DDR
- CXL
- UXL
- Ethernet

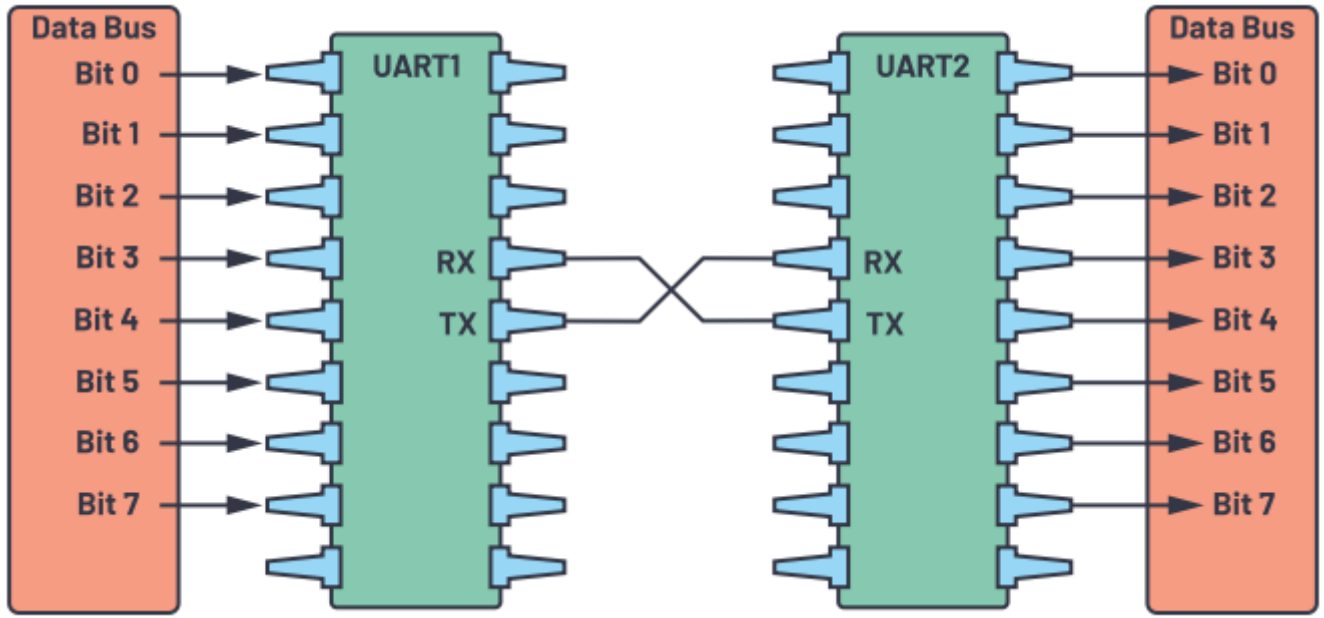
片内接口：

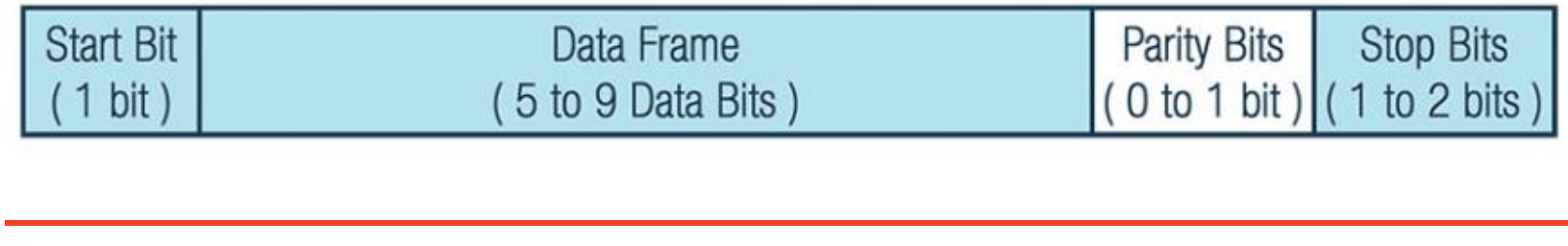
- AMBA接口
  - AXI
  - AHB
  - APB

UART-物理连接

- 物理接口

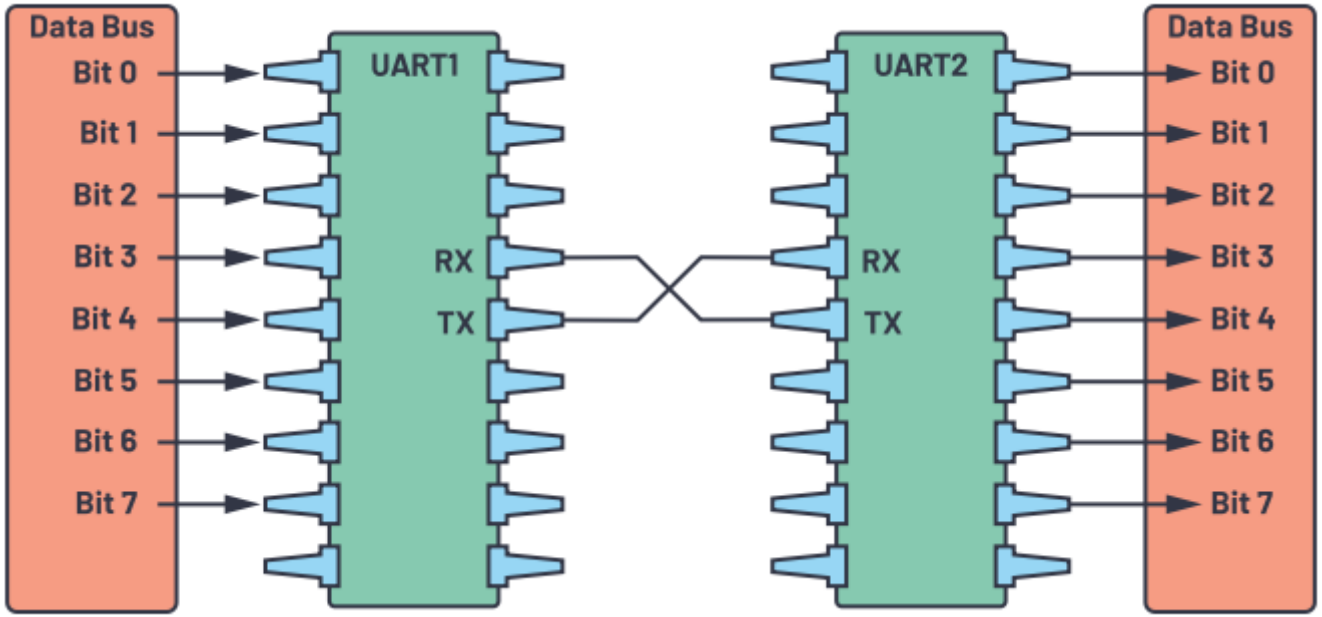






UART-物理接口

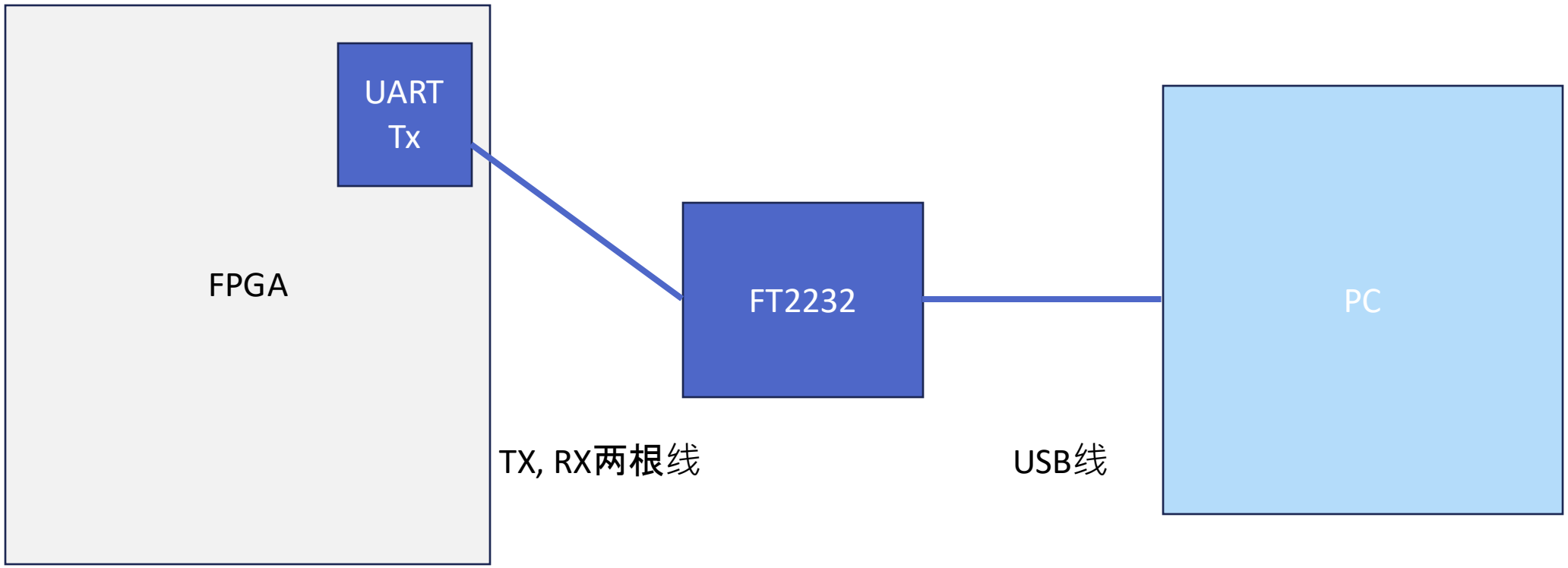
```
// Set Parameter CLKS_PER_BIT as follows:
// CLKS_PER_BIT = (Frequency of
i_Clock)/(Frequency of UART)
// Example: 25 MHz Clock, 115200 baud UART
// (25000000)/(115200) = 217
```



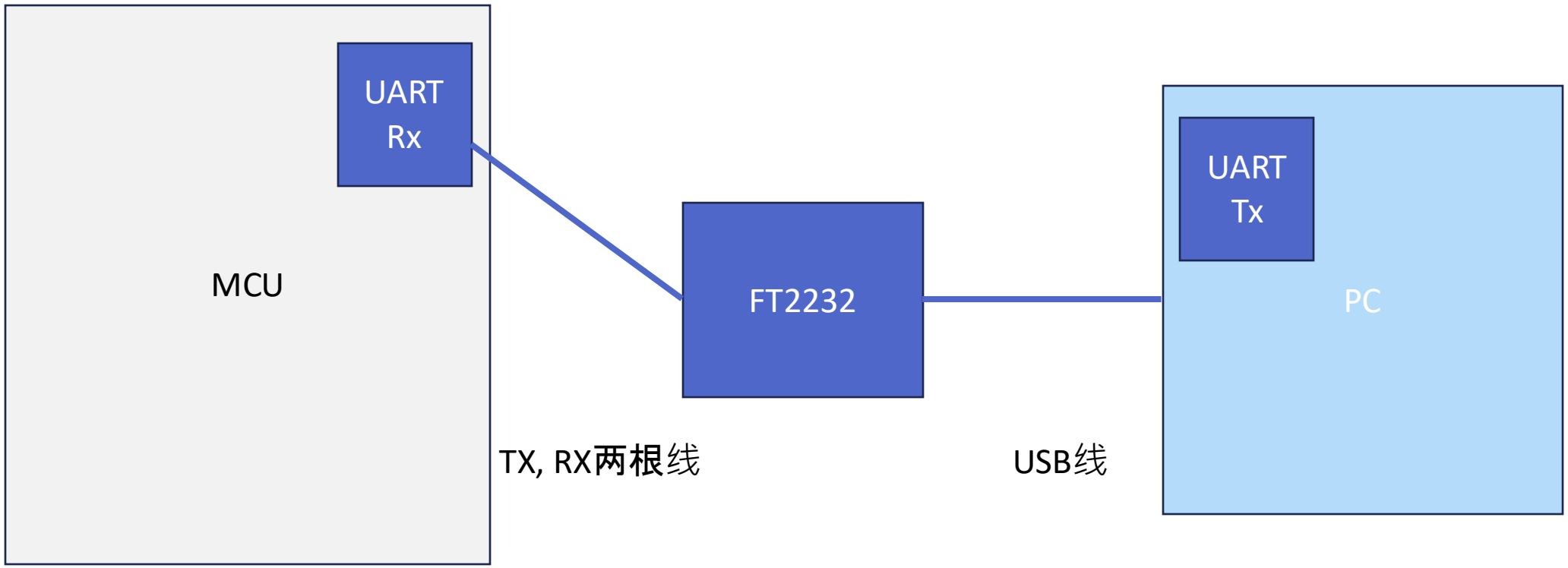

```
module UART_TX
  #(parameter CLKS_PER_BIT = 217)
  (
   input        i_Rst_L,
   input        i_Clock,
   input        i_TX_DV,
   input [7:0]  i_TX_Byte,
   output reg   o_TX_Active,
   output reg   o_TX_Serial,
   output reg   o_TX_Done
   );
```

```
module UART_RX
  #(parameter CLKS_PER_BIT = 217)
  (
   input         i_Clock,
   input         i_RX_Serial,
   output        o_RX_DV,
   output [7:0]  o_RX_Byte
   );
```

UART-连接与编程（FPGA to PC）



UART-连接与编程（MCU to PC）

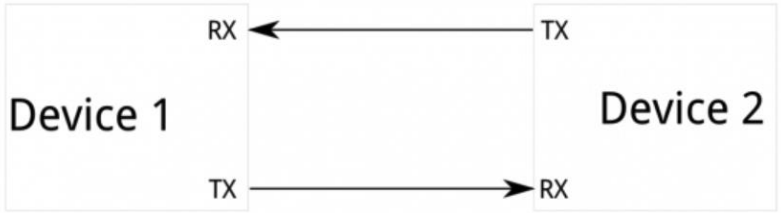


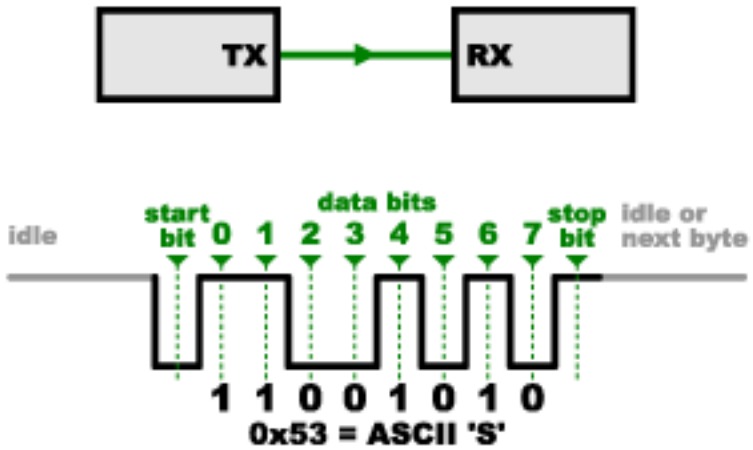
UART编程实例

- https://github.com/kendryte/PaddlePi
- PaddlePi / standalone-demos / uart

SPI

异步传输接口

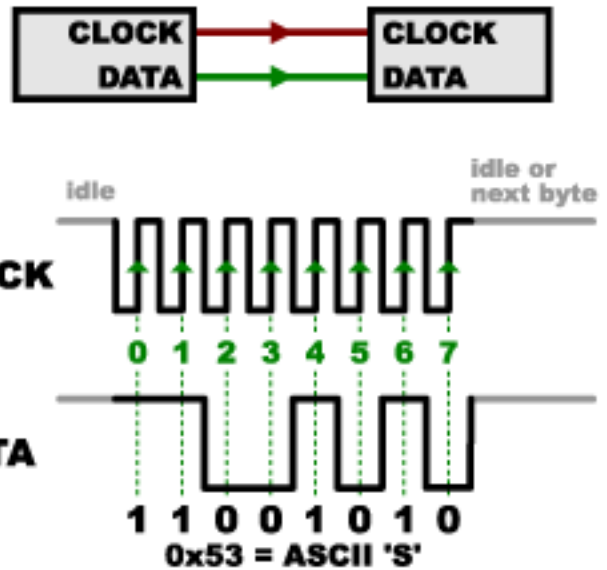




问题：

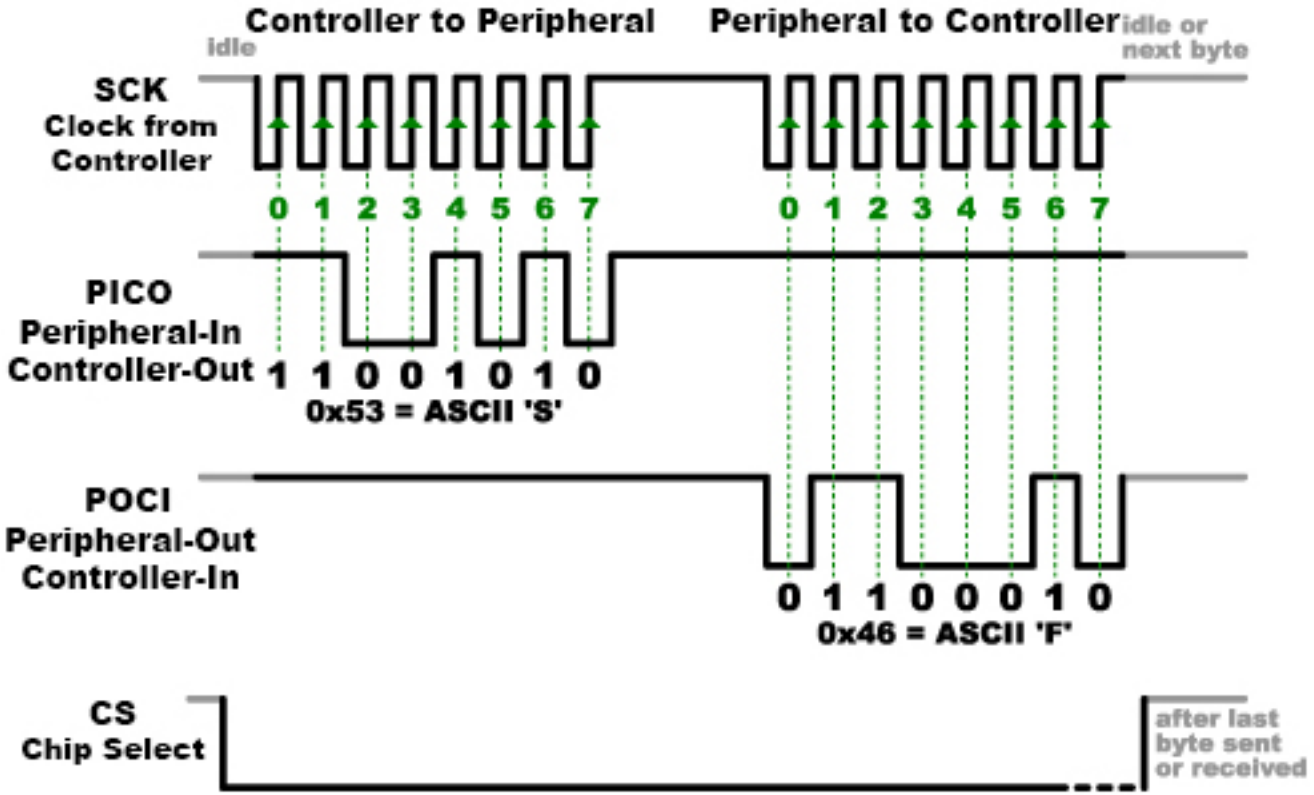
- 传输线没有公共的控制（异步）
- TX、RX的本地时钟必须差不多
- 一对一传输

改进版本



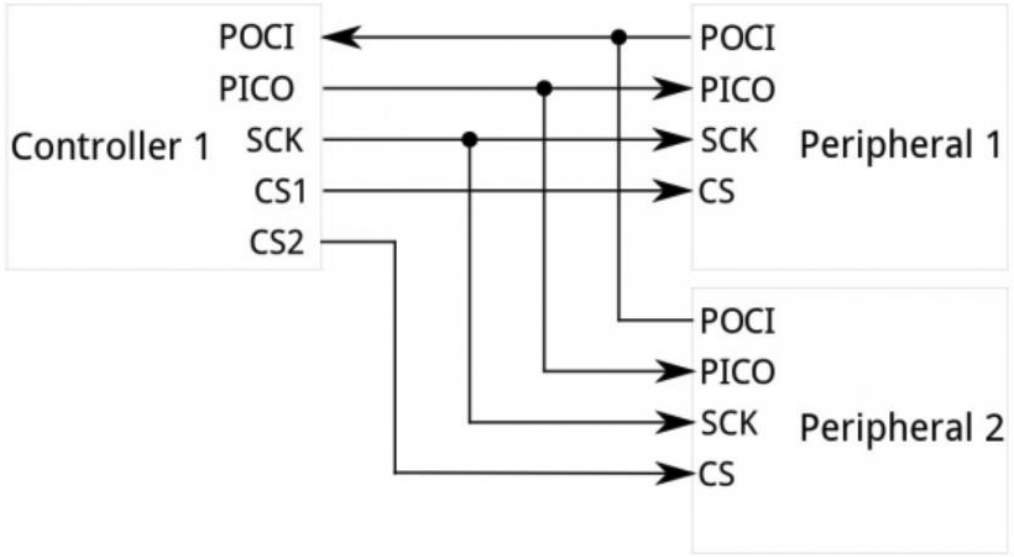
SPI – 一对一

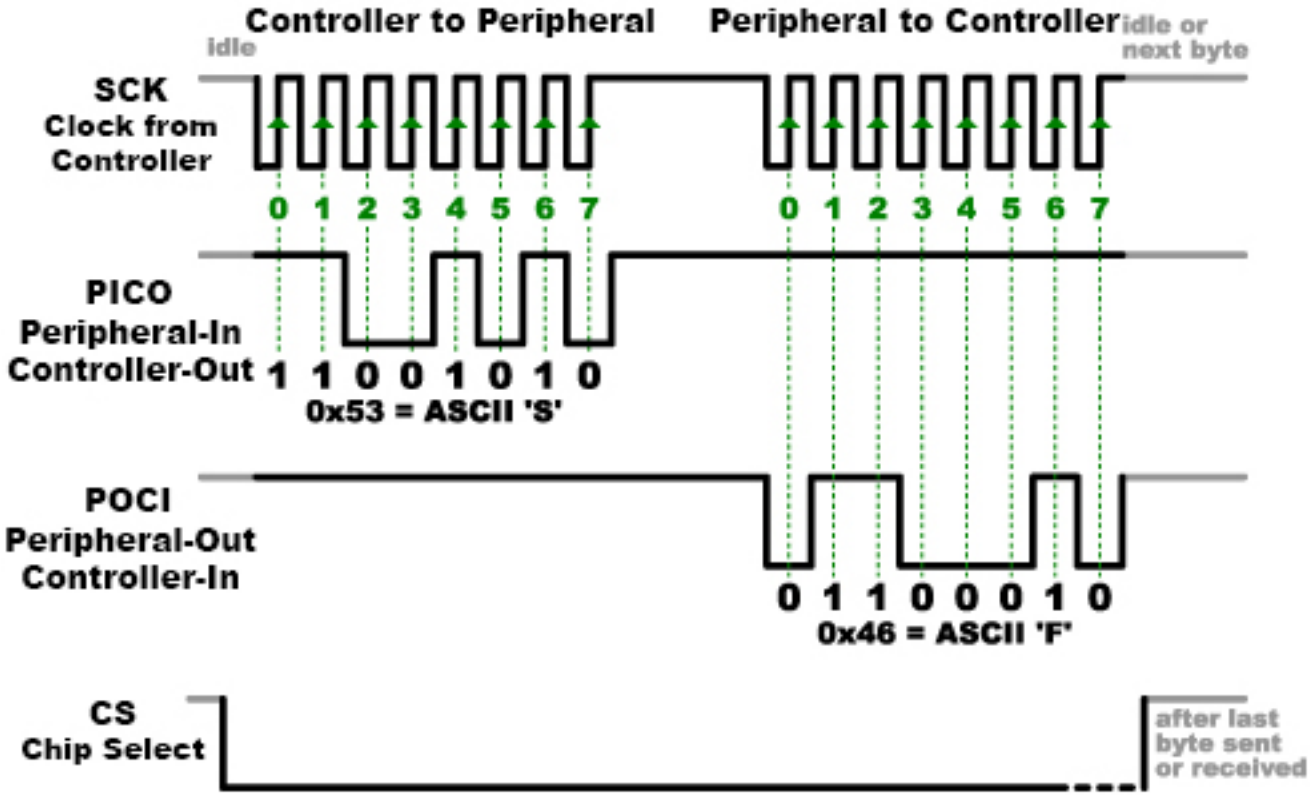
Obsolete Name	Replacement Name
Master	Controller
Slave	Peripheral
MISO	POCI
MOSI	PICO
SS	CS



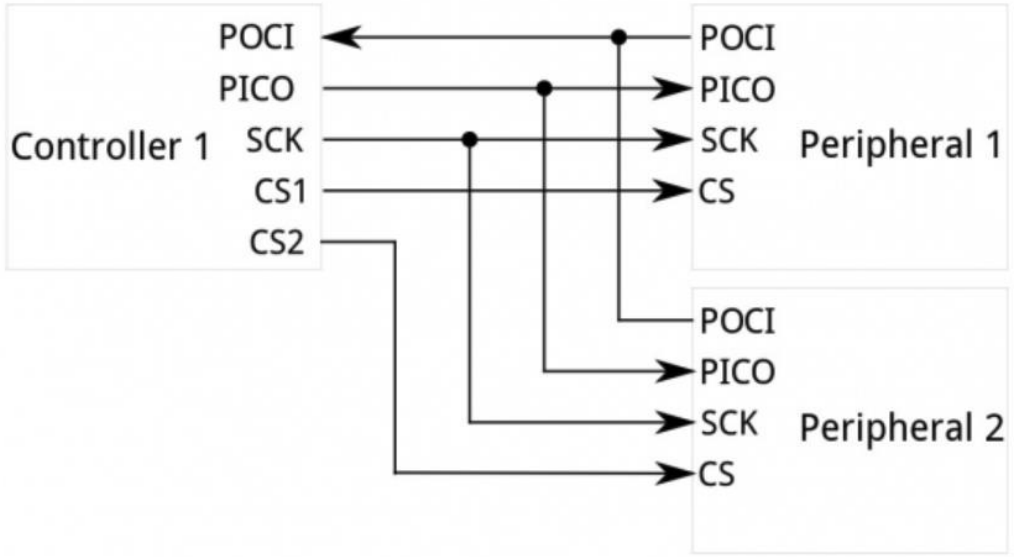
SPI – 一对多

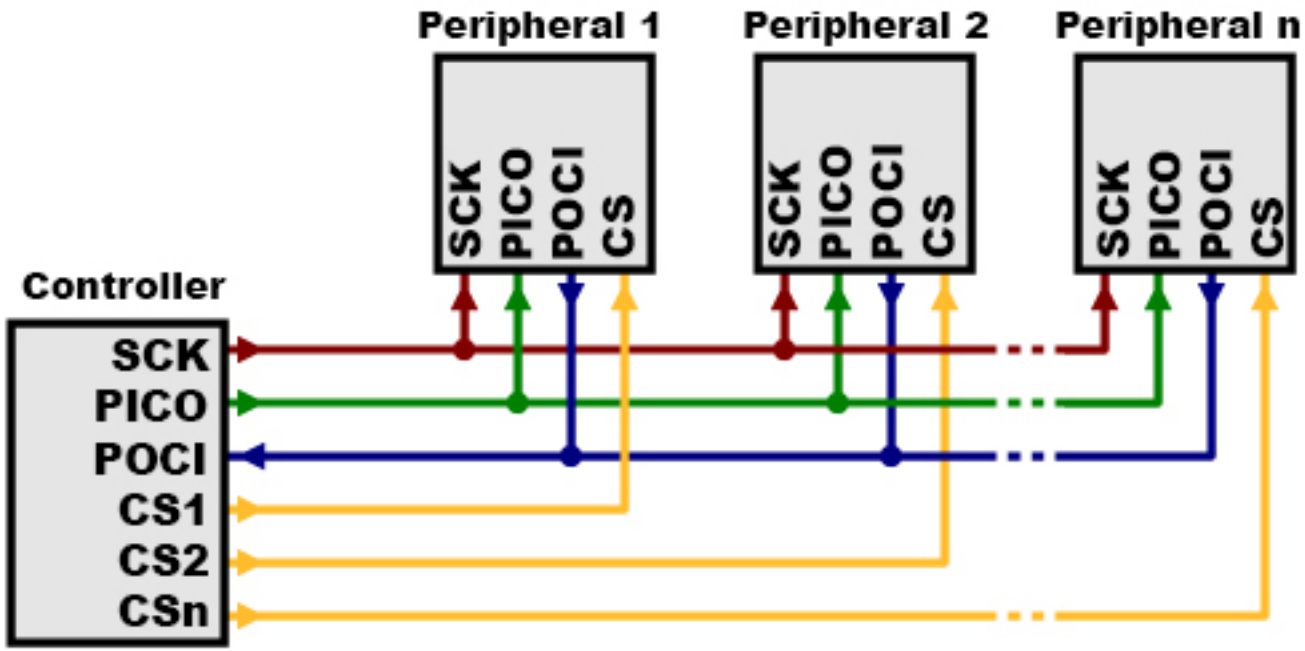

SCK
PICO
POCI
CS
SCK
PICO
POCI
CS


Controller 1
POCI
PICO
SCK
CS1
CS2
POCI
PICO
SCK
CS
Peripheral 1
POCI
PICO
SCK
CS
Peripheral 2

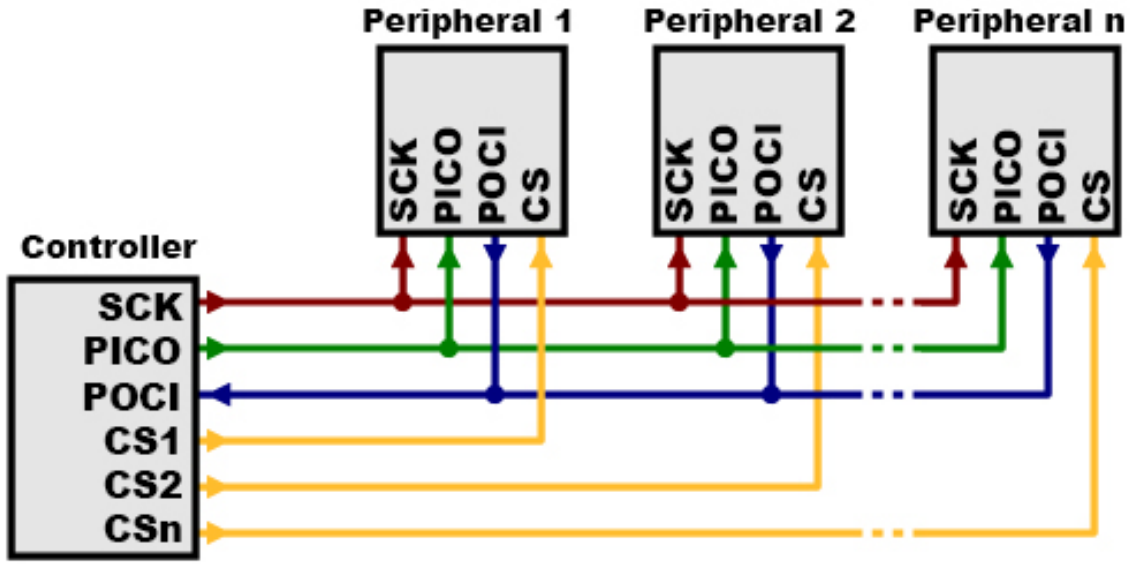

Controller to Peripheral
Peripheral to Controller
idle
idle or next byte
SCK
Clock from Controller
0 1 2 3 4 5 6 7
0 1 2 3 4 5 6 7
PICO
Peripheral-In
Controller-Out
1 1 0 0 1 0 1 0
0x53 = ASCII 'S'
POCI
Peripheral-Out
Controller-In
0 1 1 0 0 0 1 0
0x46 = ASCII 'F'
CS
Chip Select
after last byte sent or received

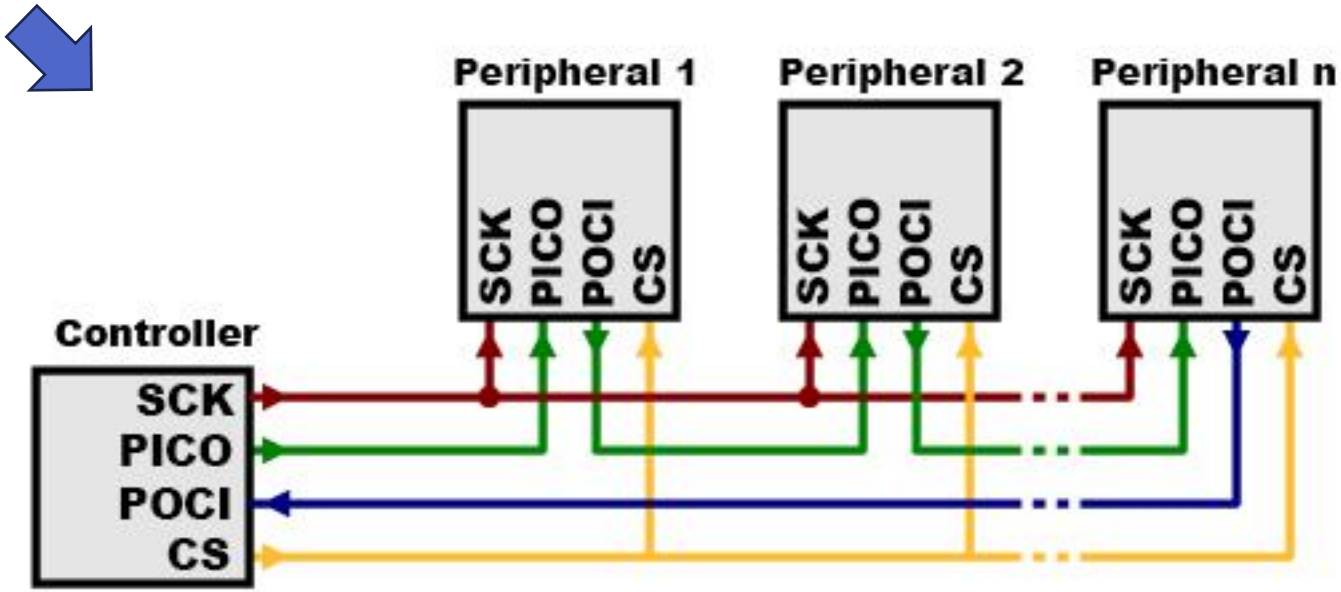
SPI – 一对多


Controller 1
POCI
PICO
SCK
CS1
CS2
Peripheral 1
POCI
PICO
SCK
CS
Peripheral 2
POCI
PICO
SCK
CS


Peripheral 1
Peripheral 2
Peripheral n
SCK
PICO
POCI
CS
Controller
SCK
PICO
POCI
CS1
CS2
CSn

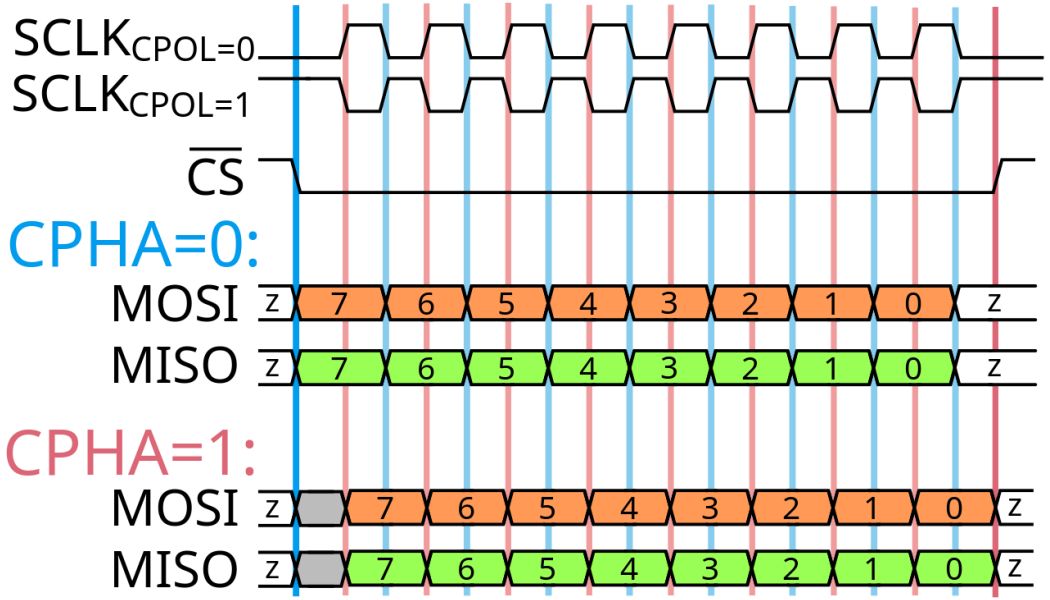
SPI – 一对多，但是daisy-chained





SPI Mode

SPI mode	Clock polarity (CPOL)	Clock phase (CPHA)	Data is shifted out on	Data is sampled on
0	0	0	falling SCLK, and when CS activates	rising SCLK
1	0	1	rising SCLK	falling SCLK
2	1	0	rising SCLK, and when CS activates	falling SCLK
3	1	1	falling SCLK	rising SCLK



SPI编程

4_kendryte_standalone_programming_guide.pdf

16.3.1 spi_init

16.3.1.1 描述

设置 SPI 工作模式、多线模式和位宽。

16.3.1.2 函数原型

```
void spi_init(spi_device_num_t spi_num, spi_work_mode_t work_mode, spi_frame_format_t
    frame_format, size_t data_bit_length, uint32_t endian)
```

16.3.1.3 参数

参数名称	描述	输入输出
spi_num	SPI 号	输入
work_mode	极性相位的四种模式	输入
frame_format	多线模式	输入
data_bit_length	单次传输的数据的位宽	输入
endian	大小端 0：小端 1：大端	输入

16.3.3 spi_send_data_standard

16.3.3.1 描述

SPI 标准模式传输数据。

16.3.3.2 函数原型

```
void spi_send_data_standard(spi_device_num_t spi_num, spi_chip_select_t chip_select,
    const uint8_t *cmd_buff, size_t cmd_len, const uint8_t *tx_buff, size_t tx_len)
```

16.3.3.3 参数

参数名称	描述	输入输出
spi_num	SPI 号	输入
chip_select	片选信号	输入
cmd_buff	外设指令地址数据，没有则设为 NULL	输入
cmd_len	外设指令地址数据长度，没有则设为 0	输入
tx_buff	发送的数据	输入
tx_len	发送数据的长度	输入

16.3.3.4 返回值

无

16.3.5 spi_receive_data_standard

16.3.5.1 描述

标准模式下接收数据。

16.3.5.2 函数原型

```
void spi_receive_data_standard(spi_device_num_t spi_num, spi_chip_select_t chip_select,
    const uint8_t *cmd_buff, size_t cmd_len, uint8_t *rx_buff, size_t rx_len)
```

16.3.5.3 参数

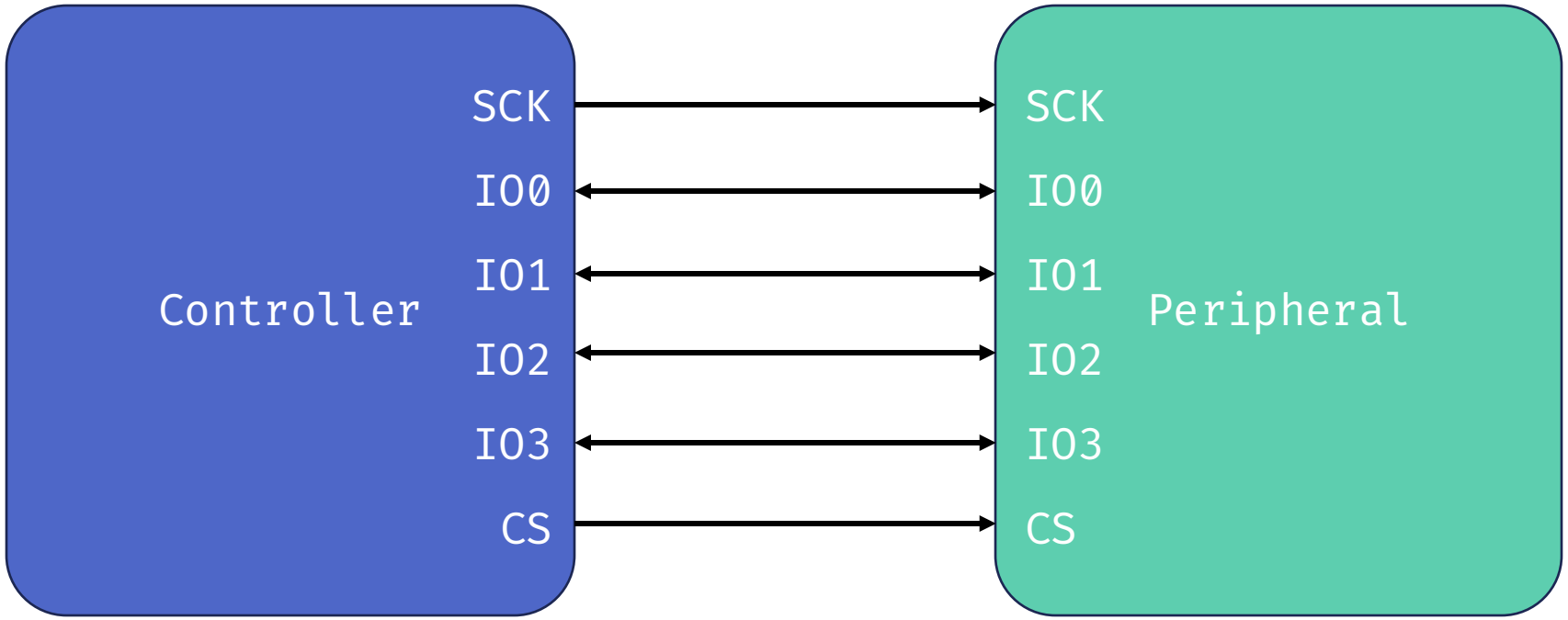
参数名称	描述	输入输出
spi_num	SPI 号	输入
chip_select	片选信号	输入
cmd_buff	外设指令地址数据，没有则设为 NULL	输入
cmd_len	外设指令地址数据长度，没有则设为 0	输入
rx_buff	接收的数据	输出
rx_len	接收数据的长度	输入

16.3.5.4 返回值

无

QSPI： Quad serial peripheral interface

- 传输1组线变成4组线
- 共用1个SCK

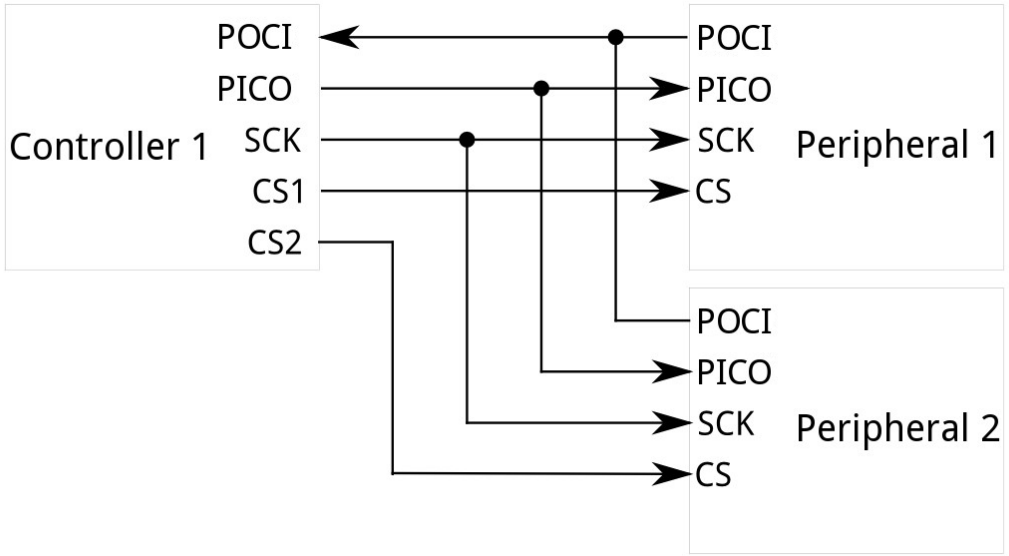


从 SPI 到 I^2C

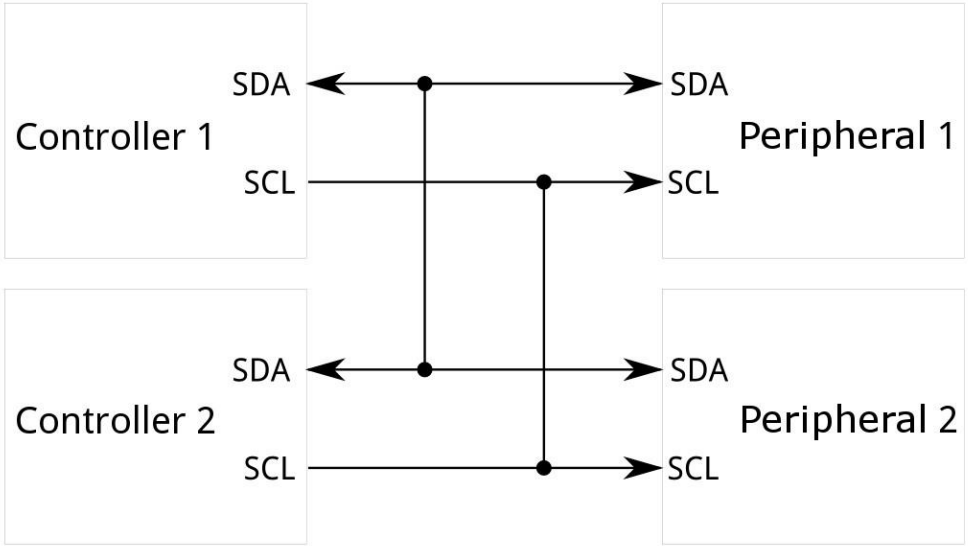
- SPI的特点：
- good for high data rate full-duplex (simultaneous sending and receiving of data) connections
- supporting clock rates upwards of 10MHz (and thus, 10 million bits per second) for some devices, and the speed scales nicely.
- The hardware at either end is usually a very simple shift register, allowing easy implementation in software.
- 但是！
  - SPI走线过多：每个interface 4根线

从 SPI 到 I²C

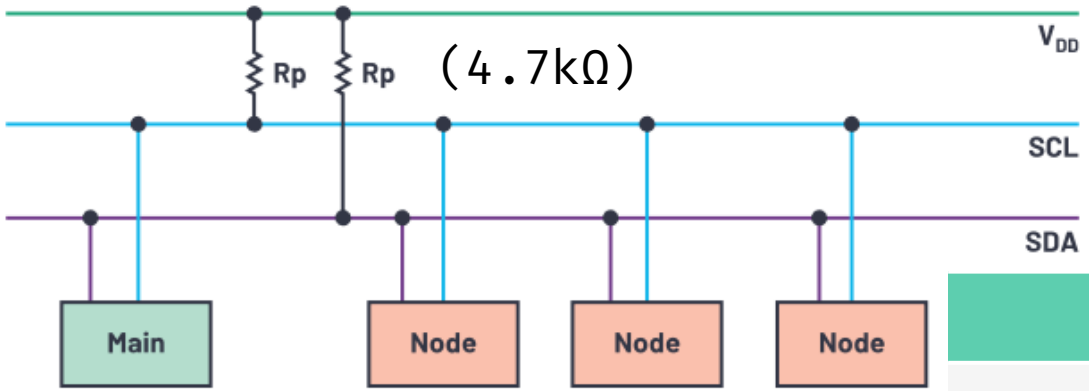
SPI



I²C



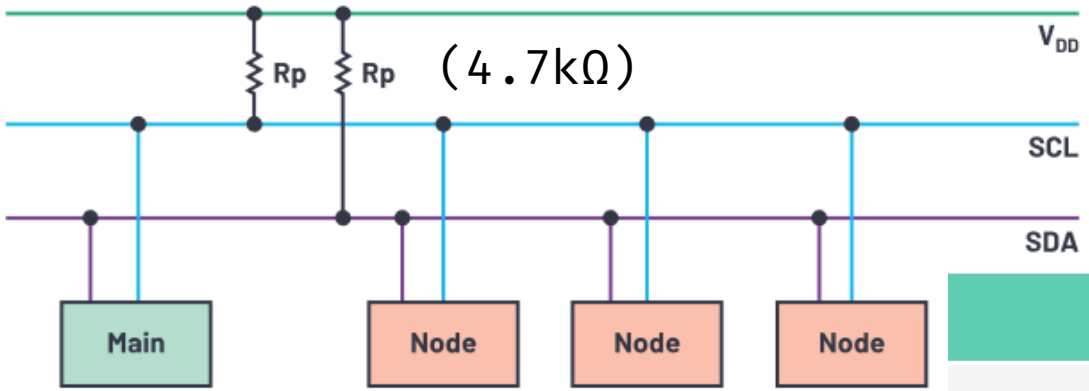
I²C Interface




```
module I2C_main
  (
   input        clk,
   input        rst,
   input [6:0] data_in,
   input [6:0] address,
   output       scl,
   inout        sda
   );
```

Features	Specs
Wires	2
Maximum Speed	Standard mode = 100 kbps Fast mode = 400 kbps High speed mode = 3.4 Mbps Ultrafast mode = 5 Mbps
Synchronous or Asynchronous?	Synchronous
Serial or Parallel?	Serial
Maximum Number of Mains	Unlimited
Maximum Number of Nodes	1008

I²C Interface

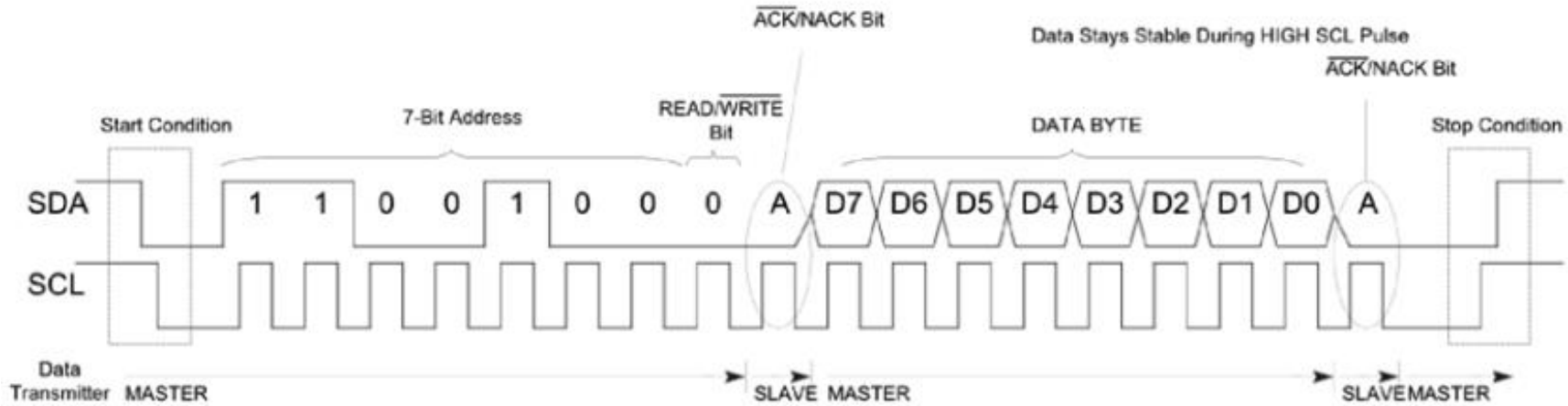



```
module I2C_node
  (
   input        clk,
   input        rst,
   input        scl,
   inout        sda,
   output
   );
```

Features	Specs
Wires	2
Maximum Speed	Standard mode = 100 kbps Fast mode = 400 kbps High speed mode = 3.4 Mbps Ultrafast mode = 5 Mbps
Synchronous or Asynchronous?	Synchronous
Serial or Parallel?	Serial
Maximum Number of Mains	Unlimited
Maximum Number of Nodes	1008

I²C 时序


Start Condition
Address Frame (7 or 10 Bits)
Read/Write Bit
ACK/NACK Bit
Data Frame 1 (8 Bits)
ACK/NACK Bit
Stop Condition


ACK/NACK Bit
Data Stays Stable During HIGH SCL Pulse
ACK/NACK Bit
Start Condition
7-Bit Address
READ/WRITE Bit
DATA BYTE
Stop Condition
SDA
1 1 0 0 1 0 0 0 A D7 D6 D5 D4 D3 D2 D1 D0 A
SCL
Data Transmitter MASTER
SLAVE MASTER
SLAVE MASTER

I²C-连接与编程