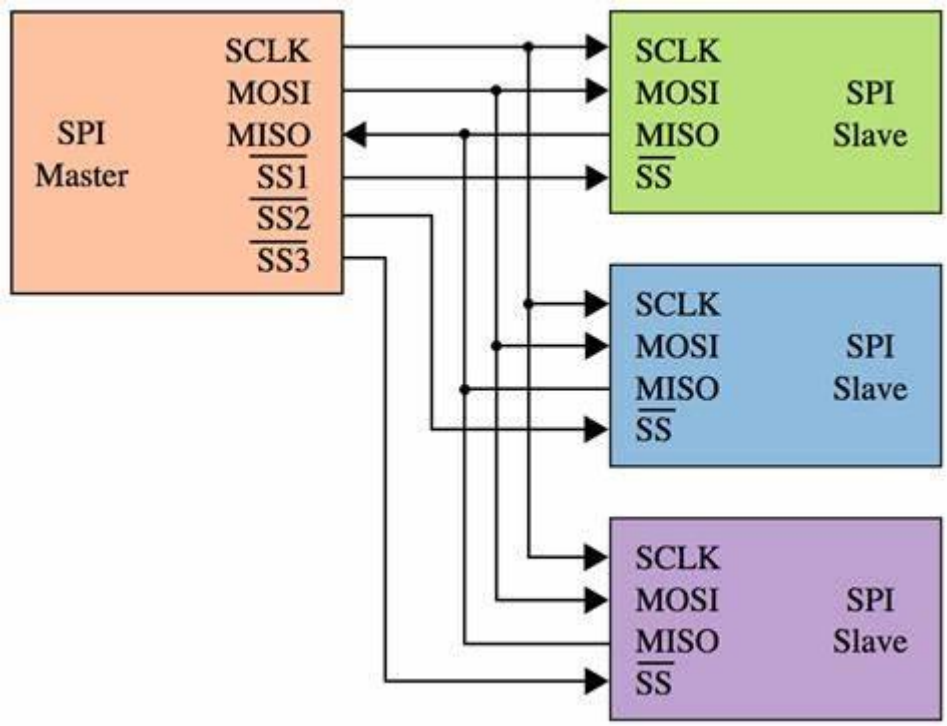
SPI is the abbreviation of Serial Peripheral Interface. It is a high-speed, full-duplex, synchronous communication bus, and only occupies four wires on the pins of the chip, saving the pins of the chip. At the same time, it saves space and provides convenience for the layout of the PCB. It is precisely because of this simple and easy-to-use feature that more and more chips integrate this communication protocol, such as AT91RM9200.



Catalogues

Basic agreement

SPI protocol overview

The SPI bus is a 4-wire bus. Because of its strong hardware function, the software related to SPI is quite simple, allowing the Central Processing Unit (CPU) to have more time to process other transactions. Because of this simple and easy-to-use feature, more and more chips integrate this communication protocol, such as AT91RM9200. SPI is a high-speed, high-efficiency serial interface technology. Usually consists of a master module and one or more slave modules. The master module selects a slave module for synchronous communication to complete the data exchange. SPI is a ring structure and requires at least 4 wires for communication (in fact, 3 wires are also acceptable for unidirectional transmission) .

The communication principle of SPI is very simple. It works in a master-slave mode. This mode usually has a master device and one or more slave devices. It requires at least 4 wires. In fact, 3 wires can also be used (one-way transmission). It is also common to all SPI-based devices. They are MISO (master device data input), MOSI (master device data output), SCLK (clock), and CS (chip select).

(1) MISO – Master Input Slave Output, master device data input, slave device data output;

(2) MOSI – Master Output Slave Input, master device data output, slave device data input;

(3) SCLK-Serial Clock, clock signal, generated by the master device;

(4) CS-Chip Select, slave device enable signal, controlled by the master device.

Among them, CS is the control signal of whether the slave chip is selected by the master chip, that is to say, only when the chip select signal is a predetermined enable signal (high potential or low potential), the master chip is effective for the operation of the slave chip. This makes it possible to connect multiple SPI devices on the same bus.

The next three lines are responsible for communication. Communication is done through data exchange. Here we must first know that SPI is a serial communication protocol, that is to say, data is transmitted bit by bit. This is the reason why the SCLK clock line exists. SCLK provides the clock pulse, and SDI and SDO complete the data transmission based on this pulse. The data output goes through the SDO line, and the data changes on the rising or falling edge of the clock and is read on the following falling or rising edge. To complete a one-bit data transfer, the input also uses the same principle. Therefore, at least 8 clock signal changes are required (upper and lower edges are one time) to complete the transmission of 8-bit data.

The SCLK signal line is only controlled by the master device, and the slave device cannot control the signal line. Similarly, in an SPI-based device, there is at least one master device. The characteristics of this transmission: This transmission method has an advantage. Unlike ordinary serial communication, ordinary serial communication continuously transmits at least 8 bits of data at a time, and SPI allows data to be transmitted one bit at a time, and even allows pause. The SCLK clock line is controlled by the master control device. When there is no clock transition, the slave device does not collect or transmit data. In other words, the master device can complete the control of the communication by controlling the SCLK clock line. SPI is also a data exchange protocol: because the data input and output lines of the SPI are independent, it is allowed to complete the data input and output at the same time. Different SPI devices have different implementation methods, mainly because the data changes and the acquisition time is different. There are different definitions on the upper or lower edge of the clock signal. For details, please refer to the relevant device documentation.

Finally, a disadvantage of the SPI interface: there is no specified flow control, and there is no response mechanism to confirm whether data is received.

The chip selection of SPI can be expanded to select 16 peripherals. At this time, PCS output = NPCS, which means that NPCS0~3 is connected to the 4-16 decoder. This decoder requires an external 4-16 decoder. The input of the decoder For NPCS0~3, the output is used to select 16 peripherals.

Examples of agreements

SPI is a ring bus structure, which is composed of ss (cs), sck, sdi, and sdo. It is mainly under the control of sck that two bidirectional shift registers exchange data.

Suppose the following 8-bit register contains the data to be sent 10101010, the rising edge sends, the falling edge receives, and the high bit is sent first.

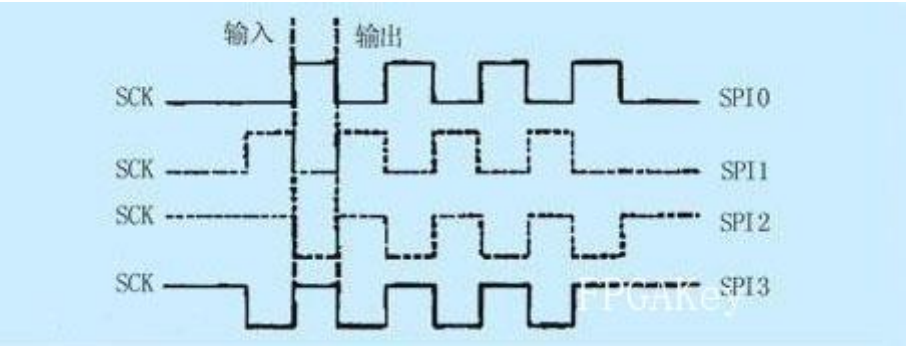
Then the data will be sdo=1 when the first rising edge comes; register=0101010x. Falling edge

When it arrives, the level on sdi will be latched into the register, then the register=0101010sdi, so that after 8 clock pulses, the contents of the two registers are exchanged with each other. This completes a spi sequence.

Example: Suppose the master and slave are ready to initialize: and the master's sbuff (serial port transceiver buffer) = 0xaa, the slave's sbuff (serial port transceiver buffer) = 0x55, the following will be divided into 8 clocks of spi Demonstrate the periodic data situation again:

In this way, the 8-bit exchange of the two registers is completed. The upper side indicates the rising edge and the lower side indicates the falling edge. Sdi and sdo are relative to the host. When the ss pin is used as a master, the slave can pull it low and passively select it as a slave. When used as a slave, it can be used as a chip select pin. According to the above analysis, a complete transfer cycle is 16 bits, that is, two bytes, because, first, the master must send a command, then the slave prepares the data according to the command of the master, and the master reads the data back in the next 8-bit clock cycle. .

The SPI bus is a three-wire synchronous interface launched by Motorola. It communicates in a synchronous serial three-wire mode: a clock line SCK, a data output line MOSI, and a data input line MISO; used for full-duplex and various peripheral devices CPU, Synchronous serial communication. The main features of SPI are: it can send and receive serial data at the same time; it can work as a master or slave; provide a frequency programmable clock; the end of transmission interrupt flag; write conflict protection; bus competition protection. The following figure shows the four modes of SPI bus operation, of which the most widely used are SPI0 and SPI3 (solid line):



In order to exchange data with peripherals, the SPI module can configure the polarity and phase of the output serial synchronous clock according to the peripheral work requirements. The clock polarity (CPOL) has no significant impact on the transmission protocol. If CPOL=0, the idle state of the serial synchronous clock is low; the phase and polarity of the SPI master module and the peripheral clock communicating with it should be consistent. The SPI interface timing is shown in Figure 3 and Figure 4.

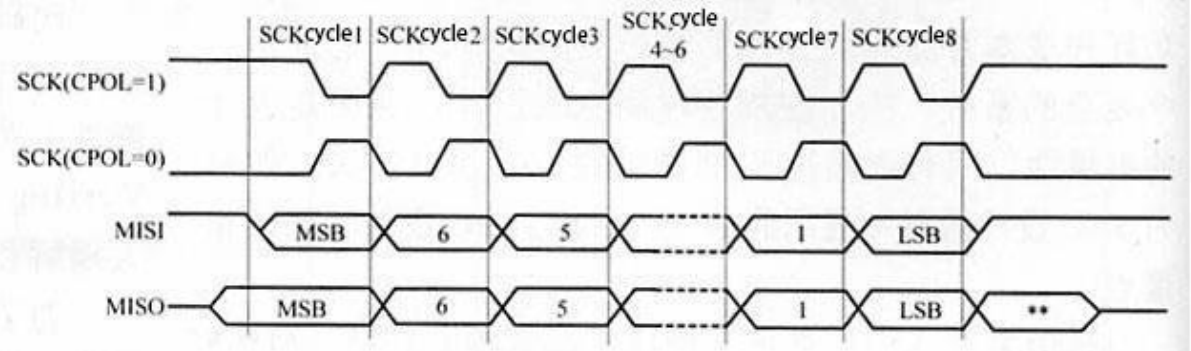


Fig. 3 Serial data transmission timing when CPHA=0

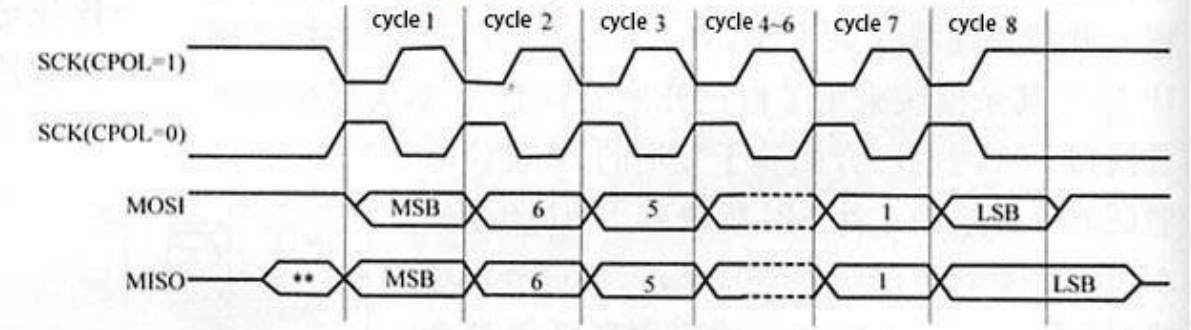


Fig. 4 Serial data transmission timing when CPHA=1

User logic

This module is designed for different applications of users, which is essentially the user's specific business applications, and has no direct relationship with the SPI-4 interface. This part is crucial when the application supports multiple ports. The following describes the design techniques of user logic by supporting two-port applications.

(1) User logic of Sink Core

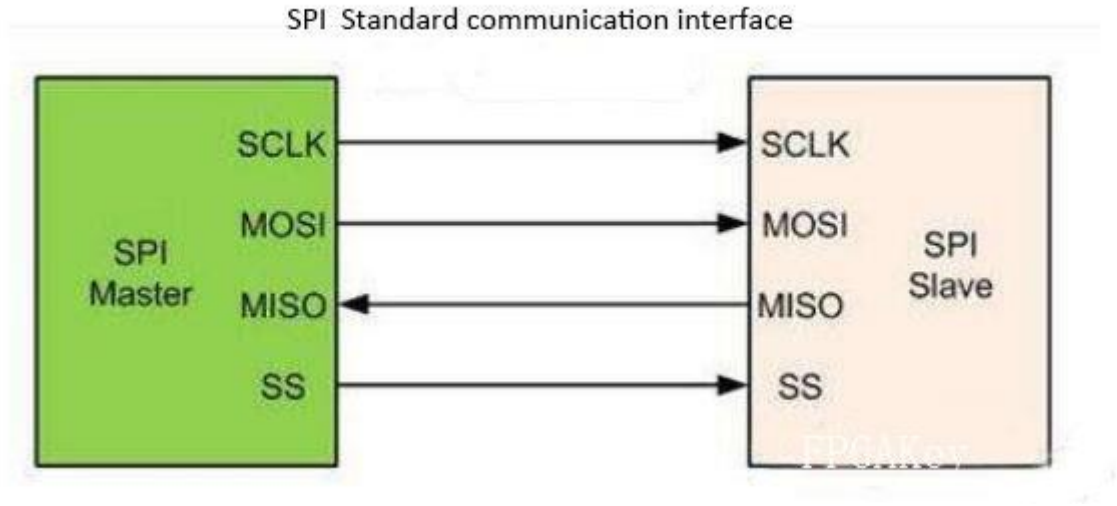
When there are two ports, the user logic needs to use two different FIFOs to buffer the user's two business data according to the port address and so on. At the same time, flow control information is sent to the SPI4 data interface according to the FIFO situation.

(2) User logic of Source Core

When there are two ports, the user logic needs to be arbitrated according to the flow control information and two different FffiOffJ situations, which user logic FIFO needs to be sent to the SPI4 data interface.

SPI principle

SPI has three specifications, as shown in the following figure for its model.



The processing flow of the three kinds of SPI is similar, and the most used SPI-4 is used as an example to illustrate the principle of SPI. It has its own data channel and flow control status information channel on the sending interface and the receiving interface. Its data channel and flow control status information channel are independent and point-to-point communication. Data is sent in the form of packets. Up to 256 ports can be supported according to the embedded address in the data packet. The following describes the basic protocol and the processing procedures of the data channel and flow control status information.

Timing diagram

The SPI interface has four different data transmission timings, depending on the combination of CPOL and CPHL.

CPOL is used to determine the level of the SCK clock signal when it is idle, CPOL=0, the idle level is low, and when CPOL=1, the idle level is high. CPHA is used to determine the sampling time, CPHA=0, sampling at the first clock edge of each cycle, CPHA=1, sampling at the second clock edge of each cycle.

Synchronous serial port

SPI: High-speed synchronous serial port. It is a standard four-wire synchronous bidirectional serial bus and a serial peripheral device interface. Motorola first defined it on its MC68HCXX series processors. The SPI interface is mainly used between EEPROM, FLASH, real-time clock, AD converter, digital signal processor and digital signal decoder.

The SPI bus system is a synchronous serial peripheral interface, which enables the MCU to communicate with various peripheral devices in a serial manner to exchange information. Peripheral settings FLASHRAM, network controller, LCD display driver, A/D converter and MCU, etc. The SPI bus system can directly interface with a variety of standard peripheral devices produced by various manufacturers. The interface generally uses 4 lines: serial clock line (SCLK), master input/slave output data line MISO, master output/slave input Data line MOSI and low-level slave selection line SS (some SPI interface chips have an interrupt signal line INT, and some SPI interface chips do not have a master output/slave input data line MOSI).

The communication principle of SPI is very simple. It works in a master-slave mode. This mode usually has a master device and one or more slave devices. At least 4 lines are required. In fact, 3 lines are also available (for unidirectional transmission, That is, half-duplex mode). It is also common to all SPI-based devices. They are SDI (data input), SDO (data output), SCLK (clock), and CS (chip select).

(1) MOSI- SPI Bus Master Output/Slave Input;

(2) MISO- SPI Bus Master Input/Slave Output;

(3) SCLK-clock signal, generated by the master device;

(4) CS-slave device enable signal, controlled by the master device (Chip select). Some ICs call this pin SS.

Among them, CS is to control whether the chip is selected, that is to say, only when the chip selection signal is a predetermined enable signal (high potential or low potential), the operation of this chip is effective. This allows multiple SPI devices to be connected on the same bus.

The next three lines are responsible for communication. Communication is done through data exchange. Here we must first know that SPI is a serial communication protocol, that is to say, data is transmitted bit by bit. This is the reason why the SCLK clock line exists. SCK provides the clock pulse, and SDI and SDO complete the data transmission based on this pulse. The data output goes through the SDO line, and the data changes on the rising or falling edge of the clock and is read on the following falling or rising edge. To complete a one-bit data transfer, the input also uses the same principle. In this way, at least 8 clock signal changes (upper edge and lower edge is once), you can complete the 8-bit data transmission.

In point-to-point communication, the SPI interface does not require addressing operations, and it is full-duplex communication, which is simple and efficient. In a system with multiple slave devices, each slave device requires an independent enable signal, which is slightly more complicated in hardware than an I2C system.

Hardware schematic

In a system with multiple slave devices, each slave device requires an independent enable signal, which is slightly more complicated in hardware than an I2C system.

The SPI interface is actually two simple shift registers in the internal hardware. The transmitted data is 8 bits. Under the slave device enable signal and shift pulse generated by the master device, the bit is transmitted bit by bit. The high bit is in the front and the low bit is in the back. . As shown in the figure below, the data changes on the falling edge of SCLK, while one bit of data is stored in the shift register.

Performance characteristics

The AT91RM9200's SPI interface is mainly composed of 4 pins: SPICLK, MOSI, MISO and NSS, where SPICLK is the common clock of the entire SPI bus, MOSI and MISO are used as the master, the input and output flag of the slave, MOSI is the output of the master, The input

of the slave, MISO is the input of the master, and the output of the slave. NSS is the flag pin of the slave. In two SPI bus devices that communicate with each other, the low level of the NSS pin is the slave, whereas the high level of the NSS pin is the master. In an SPI communication system, there must be a host. The SPI bus can be configured as single-master single-slave, single-master multi-slave, and mutual master-slave.

The chip selection of SPI can be expanded to select 16 peripherals. At this time, PCS output = NPCS, which means that NPCS0~3 is connected to 4-16 decoder. This decoder needs to be connected to 4-16 decoder. The input of the decoder For NPCS0~3, the output is used to select 16 peripherals.

One disadvantage of the SPI interface: there is no specified flow control, and there is no response mechanism to confirm whether data is received.

Examples of agreements

SPI is a ring bus structure, which is composed of ss (cs), sck, sdi, and sdo. Its timing is actually very simple, mainly under the control of sck, two bidirectional shift registers exchange data.

Suppose the following 8-bit register contains the data to be sent 10101010, the rising edge sends, the falling edge receives, and the high bit is sent first.

Then the data will be sdo=1 when the first rising edge comes; 10101010 in the register is shifted to the left by one bit, followed by the unknown bit x sent to become 1001010x. When the falling edge arrives, the level on sdi will be latched into the register, then the register=0101010sdi, so that after 8 clock pulses, the contents of the two registers are exchanged with each other. This completes a spi sequence.

Performance supplement

The last sentence in the above: the phase and polarity of the clock of the SPI master module and the external device communicating with it should be consistent. Personally understand this sentence has two meanings: First, the configuration of the SPI clock and polarity of the master device should be determined by the peripheral; second, the configuration of the two should be consistent, that is, the SDO of the master device and the SDO of the slave device The configuration is consistent, and the SDI of the master device is the same as the SDI configuration of the slave device. Because the master and slave devices are under the control of SCLK, send and receive data at the same time, and exchange data through two bidirectional shift registers. The working principle is shown below:

On the rising edge, the host SDO sends data 1 and the slave SDO sends data 0; immediately after the falling edge of SCLK, the SDI of the device receives the data 1 sent by the host, and the host also receives the data 0 sent from the device.

Agreement Experience

SPI interface clock configuration experience: When configuring the SPI interface clock on the master device side, be sure to understand the clock requirements of the slave device, because the clock polarity and phase on the master device side are based on the slave device.

Therefore, in the configuration of the clock polarity, it is necessary to understand whether the slave device receives data on the rising or falling edge of the clock and outputs data on the falling or rising edge of the clock. However, it should be noted that because the SDO of the master device is connected to the SDI of the slave device, and the SDO of the slave device is connected to the SDI of the master device, the data received from the SDI of the slave device is sent from the SDO of the master device, and the data received by the SDI of the master device is sent from The SDO is sent by the device, so the configuration of the SPI clock polarity of the master device (that is, the configuration of the SDO) is opposite to the polarity of the SDI received data of the slave device, and the polarity of the data sent by the SDO of the slave device is the same. The following paragraph is said on the Sychip Wlan8100 Module Spec, which fully explains how to configure the clock polarity:

The 81xx module will always input data bits at the rising edge of the clock, and the host will always output data bits on the falling edge of the clock.

Meaning: The master device sends data on the falling edge of the clock, and the slave device receives data on the rising edge of the clock. Therefore, the SPI clock polarity of the master device should be configured to be valid on the falling edge.

As another example, the following paragraph is taken from the LCD Driver IC SSD1289:

SDI is shifted into 8-bit shift register on every rising edge of SCK in the order of data bit 7, data bit 6 …… data bit 0.

Meaning: The slave device SSD1289 receives data on the rising edge of the clock, and receives data in the order from high bit to low bit. Therefore, the SPI clock polarity of the master device should also be configured to be valid on the falling edge.

After the clock polarity and phase are configured correctly, the data can be sent and received accurately. Therefore, the clock of the master device should be correctly configured according to the SPI interface timing of the slave device or Spec document description.

SPI fundamentals

Basically:

1. It is synchronous.
2. It is full-duplex serial.
3. It is not plug-and-play.
4. There is one (and only one) master, and one (or more) slaves.

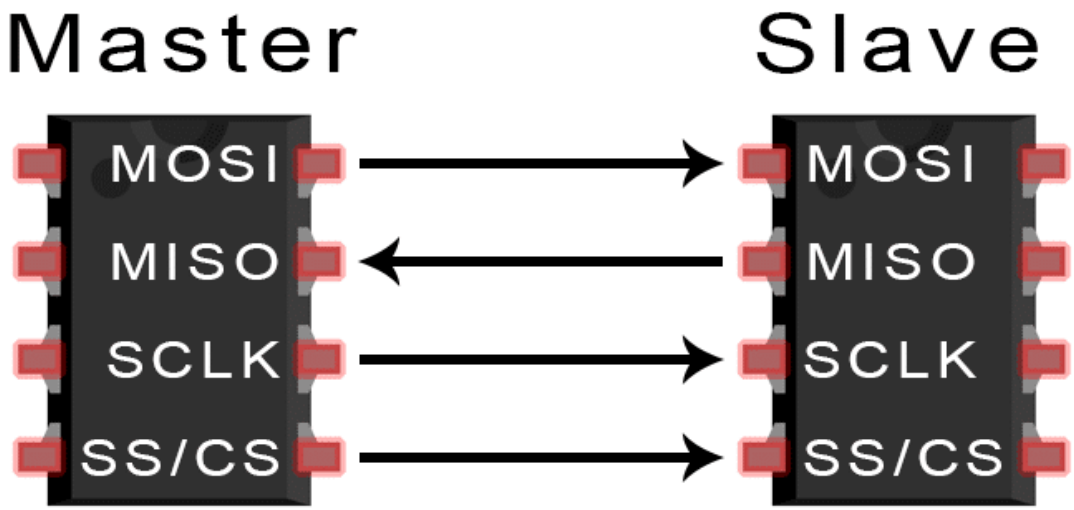
In more details:

1. Synchronous: a clock is generated by the master.
2. Full-duplex serial: data is serialized, one bit of data is transferred in each direction during each clock period, so two data wires are used (MOSI and MISO).

3. Not plug-and-play: The master and slave know beforehand the details of the communication (bit order, length of data words exchanged, etc...).
4. One master: slave(s) cannot initiates communication, only the master can. The slave(s) listen and respond.

Basics of the SPI Communication Protocol

Posted by Scott Campbell | DIY Electronics | 56



When you connect a microcontroller to a sensor, display, or other module, do you ever think about how the two devices talk to each other? What exactly are they saying? How are they able to understand each other?

Communication between electronic devices is like communication between humans. Both sides need to speak the same language. In electronics, these languages are called *communication protocols*. Luckily for us, there are only a few communication protocols we need to know when building most DIY electronics projects. In this series of articles, we will discuss the basics of the three most common protocols: Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I2C), and Universal Asynchronous Receiver/Transmitter (UART) driven communication.

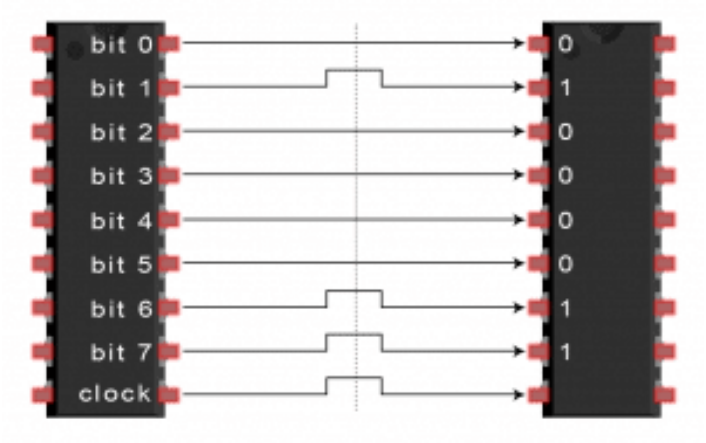
First, we'll begin with some basic concepts about electronic communication, then explain in detail how SPI works. In the next article, we'll discuss UART driven communication, and in the third article, we'll dive into I2C.

SPI, I2C, and UART are quite a bit slower than protocols like USB, ethernet, Bluetooth, and WiFi, but they're a lot more simple and use less hardware and system resources. SPI, I2C, and UART are ideal for communication between microcontrollers and between microcontrollers and sensors where large amounts of high speed data don't need to be transferred.

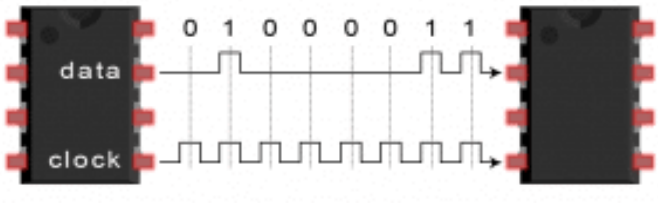
Serial vs. Parallel Communication

Electronic devices talk to each other by sending *bits* of data through wires physically connected between devices. A bit is like a letter in a word, except instead of the 26 letters (in the English alphabet), a bit is binary and can only be a 1 or 0. Bits are transferred from one device to another by quick changes in voltage. In a system operating at 5 V, a 0 bit is communicated as a short pulse of 0 V, and a 1 bit is communicated by a short pulse of 5 V.

The bits of data can be transmitted either in parallel or serial form. In parallel communication, the bits of data are sent all at the same time, each through a separate wire. The following diagram shows the parallel transmission of the letter “C” in binary (01000011):



In serial communication, the bits are sent one by one through a single wire. The following diagram shows the serial transmission of the letter “C” in binary (01000011):

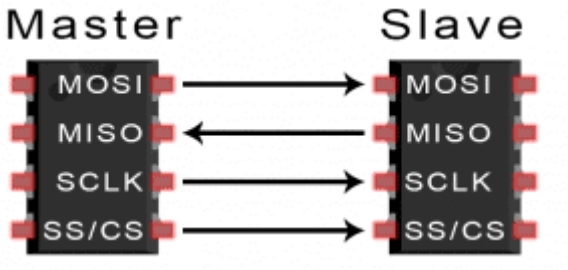


Introduction to SPI Communication

SPI is a common communication protocol used by many different devices. For example, SD card reader modules, RFID card reader modules, and 2.4 GHz wireless transmitter/receivers all use SPI to communicate with microcontrollers.

One unique benefit of SPI is the fact that data can be transferred without interruption. Any number of bits can be sent or received in a continuous stream. With I2C and UART, data is sent in packets, limited to a specific number of bits. Start and stop conditions define the beginning and end of each packet, so the data is interrupted during transmission.

Devices communicating via SPI are in a master-slave relationship. The master is the controlling device (usually a microcontroller), while the slave (usually a sensor, display, or memory chip) takes instruction from the master. The simplest configuration of SPI is a single master, single slave system, but one master can control more than one slave (more on this below).



MOSI (Master Output/Slave Input) – Line for the master to send data to the slave.

MISO (Master Input/Slave Output) – Line for the slave to send data to the master.

SCLK (Clock) – Line for the clock signal.

SS/CS (Slave Select/Chip Select) – Line for the master to select which slave to send data to.

Wires Used	4
Maximum Speed	Up to 10 Mbps
Synchronous or Asynchronous?	Synchronous
Serial or Parallel?	Serial
Max # of Masters	1
Max # of Slaves	Theoretically unlimited*

*In practice, the number of slaves is limited by the load capacitance of the system, which reduces the ability of the master to accurately switch between voltage levels.

How SPI Works

The Clock

The clock signal synchronizes the output of data bits from the master to the sampling of bits by the slave. One bit of data is transferred in each clock cycle, so the speed of data transfer is determined by the frequency of the clock signal. SPI communication is always initiated by the master since the master configures and generates the clock signal.

Any communication protocol where devices share a clock signal is known as *synchronous.* SPI is a synchronous communication protocol. There are also *asynchronous* methods that don't use a clock signal. For example, in UART communication, both sides are set to a pre-configured baud rate that dictates the speed and timing of data transmission.

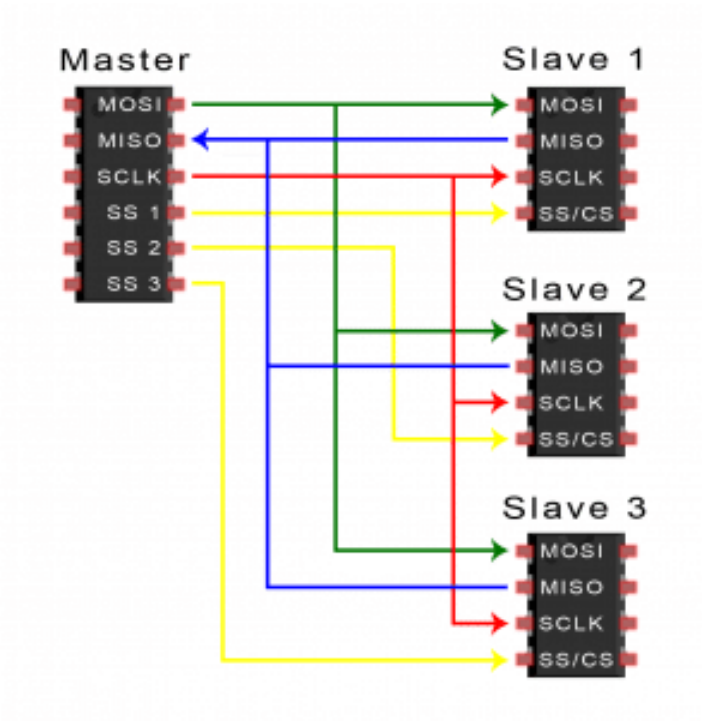
The clock signal in SPI can be modified using the properties of *clock polarity* and *clock phase*. These two properties work together to define when the bits are output and when they are sampled. Clock polarity can be set by the master to allow for bits to be output and sampled on either the rising or falling edge of the clock cycle. Clock phase can be set for output and sampling to occur on either the first edge or second edge of the clock cycle, regardless of whether it is rising or falling.

Slave Select

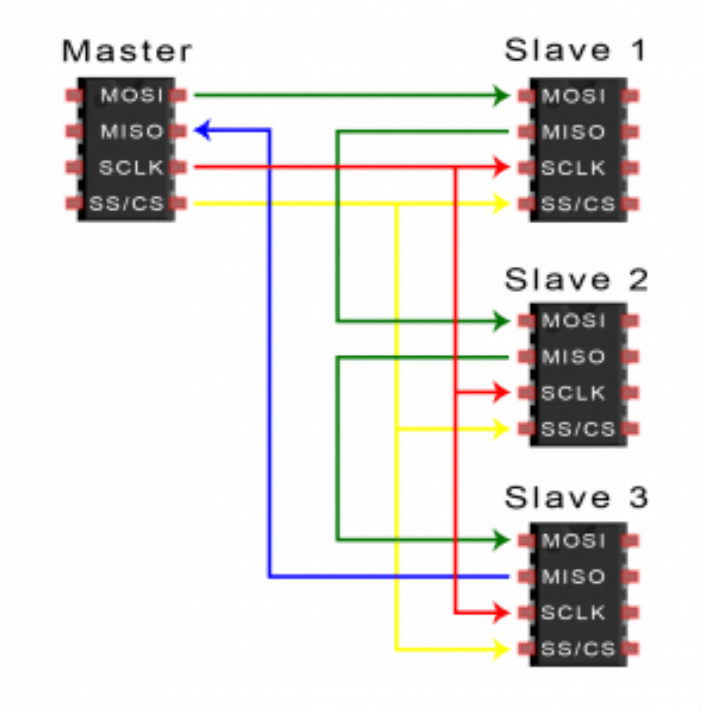
The master can choose which slave it wants to talk to by setting the slave's CS/SS line to a low voltage level. In the idle, non-transmitting state, the slave select line is kept at a high voltage level. Multiple CS/SS pins may be available on the master, which allows for multiple slaves to be wired in parallel. If only one CS/SS pin is present, multiple slaves can be wired to the master by daisy-chaining.

Multiple Slaves

SPI can be set up to operate with a single master and a single slave, and it can be set up with multiple slaves controlled by a single master. There are two ways to connect multiple slaves to the master. If the master has multiple slave select pins, the slaves can be wired in parallel like this:



If only one slave select pin is available, the slaves can be daisy-chained like this:



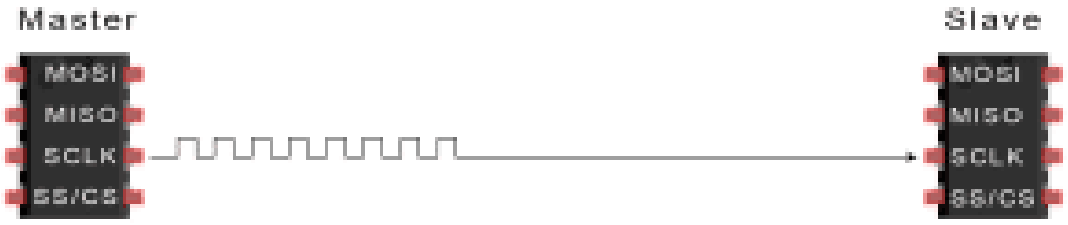
MOSI and MISO

The master sends data to the slave bit by bit, in serial through the MOSI line. The slave receives the data sent from the master at the MOSI pin. Data sent from the master to the slave is usually sent with the most significant bit first.

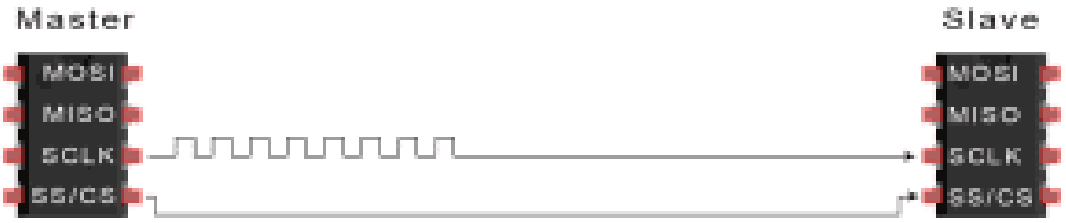
The slave can also send data back to the master through the MISO line in serial. The data sent from the slave back to the master is usually sent with the least significant bit first.

Steps of SPI Data Transmission

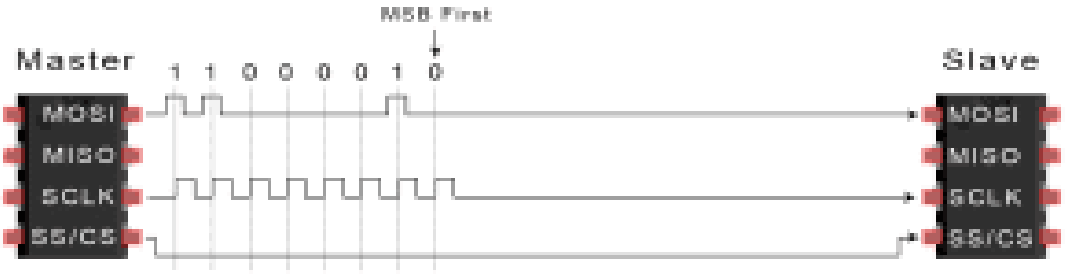
1. The master outputs the clock signal:



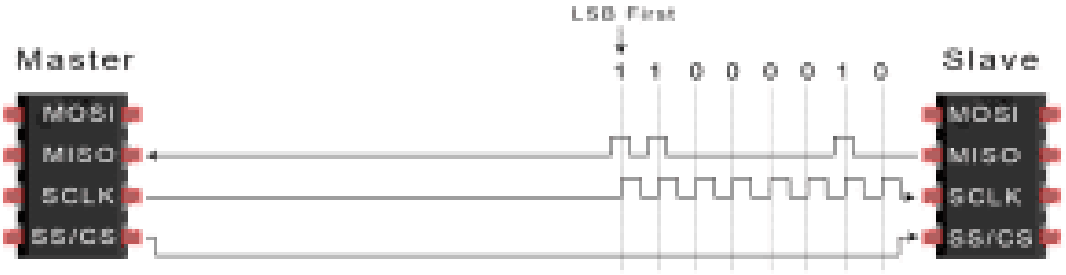
2. The master switches the SS/CS pin to a low voltage state, which activates the slave:



3. The master sends the data one bit at a time to the slave along the MOSI line. The slave reads the bits as they are received:



4. If a response is needed, the slave returns data one bit at a time to the master along the MISO line. The master reads the bits as they are received:



Advantages and Disadvantages of SPI

There are some advantages and disadvantages to using SPI, and if given the choice between different communication protocols, you should know when to use SPI according to the requirements of your project:

Advantages

- No start and stop bits, so the data can be streamed continuously without interruption
- No complicated slave addressing system like I2C
- Higher data transfer rate than I2C (almost twice as fast)
- Separate MISO and MOSI lines, so data can be sent and received at the same time

Disadvantages

- Uses four wires (I2C and UARTs use two)
- No acknowledgement that the data has been successfully received (I2C has this)

- No form of error checking like the parity bit in UART
- Only allows for a single master

Hopefully this article has given you a better understanding of SPI. Continue on to part two of this series to learn about UART driven communication, or to part three where we discuss the I2C protocol.