Serial Peripheral Interface (SPI) – Motorola

Comparing SPI to I2C

How to decide?

Accelerometers, digital potentiometers, and displays, are available in both SPI and I2C versions.

SPI Advantages

- Can operate at high speeds (10 Mbps)
- Easier to work with
- No pull-up resistors needed
- Built-in Arduino hardware support

I2C Advantages

- Max speed is 5 Mbps
- Requires only 2 communication lines
- Built-in Arduino hardware support

Examples

- For example, SD card modules, RFID card reader modules, and 2.4 GHz wireless transmitter/receivers all use SPI to communicate with microcontrollers.
- The master is the controlling device (usually a microcontroller), while the slave (usually a sensor, display, or memory chip) takes instruction from the master.

Overview of SPI bus

- Originally created by Motorola, the SPI bus is a full-duplex serial communication standard that enables simultaneous bidirectional communication between a master device and one or more slave devices.
- Because the SPI protocol does not follow a formal standard, it is common to find SPI devices that operate slightly different (the number of transmitted bits may differ, or the slave select line might be omitted, among other things)
- Bear in mind that SPI implementations can vary, so reading the datasheet is extremely important

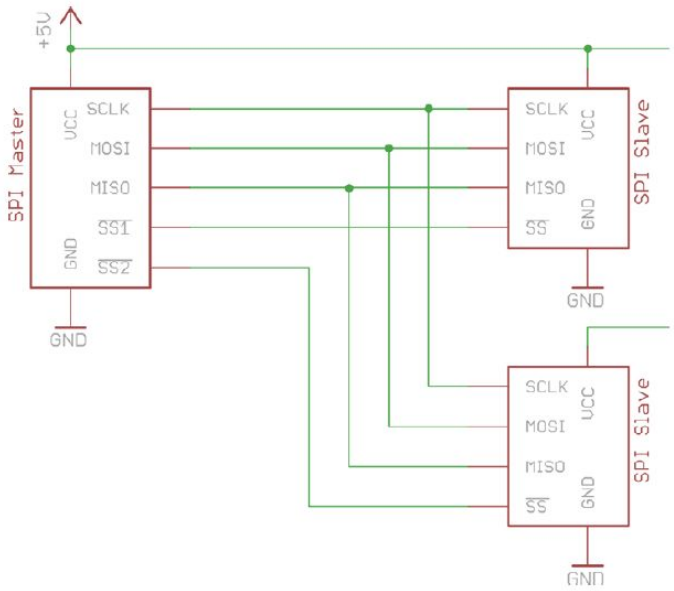
SPI Communication process

- The basic process for communicating with an SPI device is as follows:

  1. Set the SS pin low for the device you want to communicate with.
  2. Toggle the clock line up and down at a speed less than or equal to the transmission speed supported by the slave device.
  3. For each clock cycle, send 1 bit on the MOSI line, and receive 1 bit on the MISO line.
  4. Continue until transmitting or receiving is complete, and stop toggling the clock line.
  5. Return the SS pin to high state.

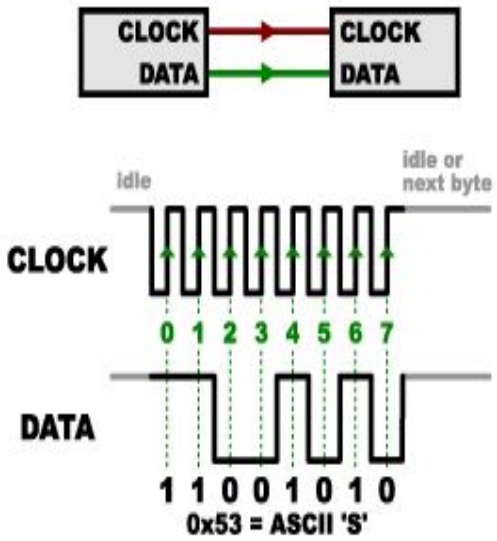
- NOTE: on every clock cycle a bit must be sent and received, but that bit does not necessarily need to mean anything

SPI Configuration



- Shared/Serial Clock (SCLK)
- Master Out Slave In (MOSI)
- Master In Slave Out (MISO)
- An additional slave select (SS) pin for each slave

SPI (How it works)

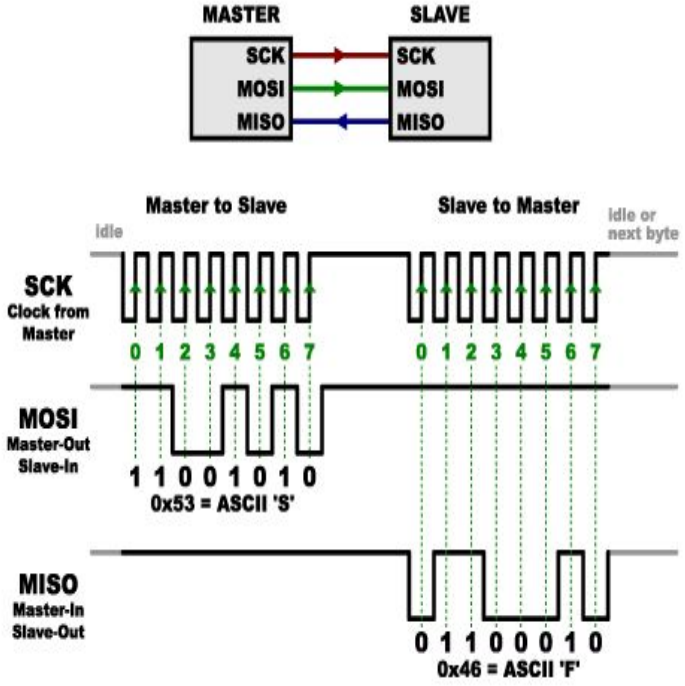


- Four-wire Interface
- **SCLK (Clock)** – Line for the clock signal.
- **MOSI (Master Output/Slave Input)** – Line for the master to send data to the slave.
- **MISO (Master Input/Slave Output)** – Line for the slave to send data to the master.
- **SS/CS (Slave Select/Chip Select)** – Line for the master to select which slave to send data to.

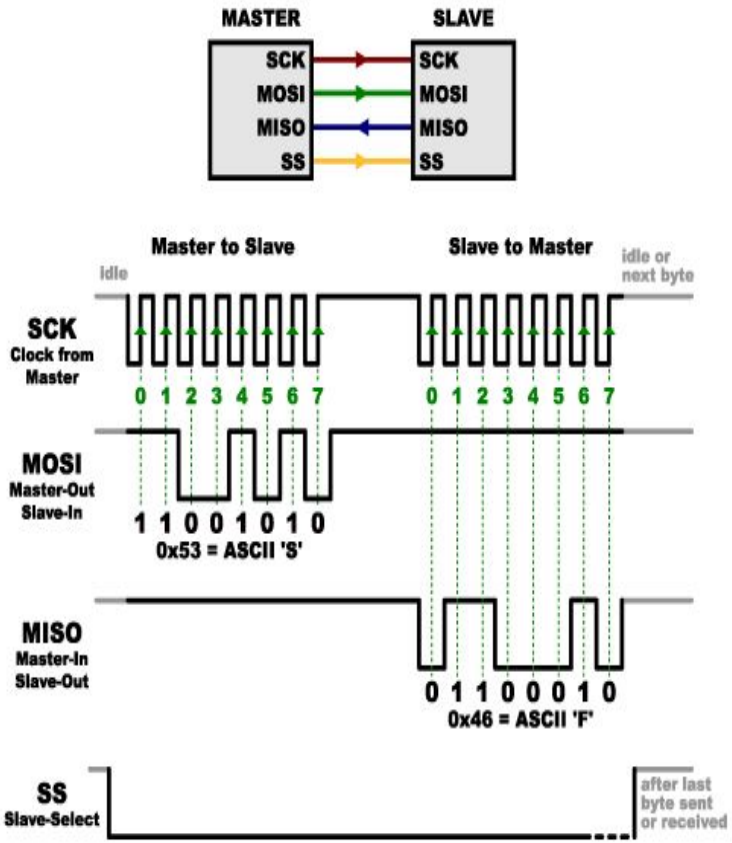
How SPI works - Clock

- 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.
- 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 - set by the master to allow for bits sampled on either the rising or falling edge of the clock cycle
- Clock phase - sampling to occur on either the first edge or second edge of the clock cycle, regardless of whether it is rising or falling.

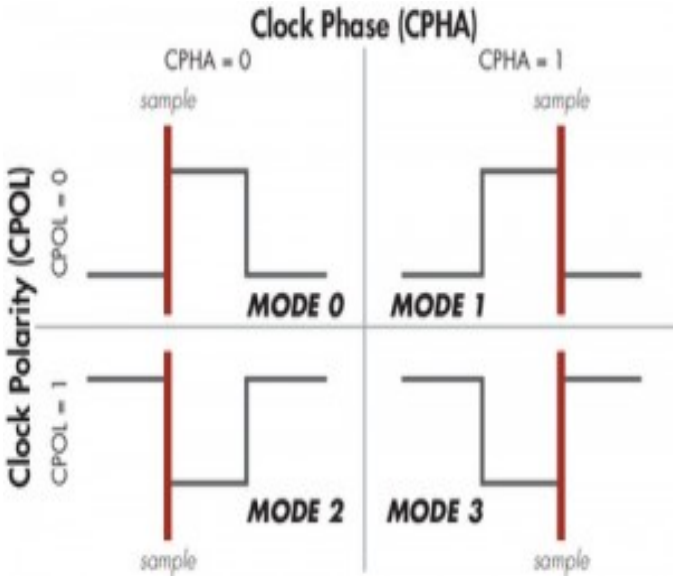
SPI (How it works)



SPI with Slave Select

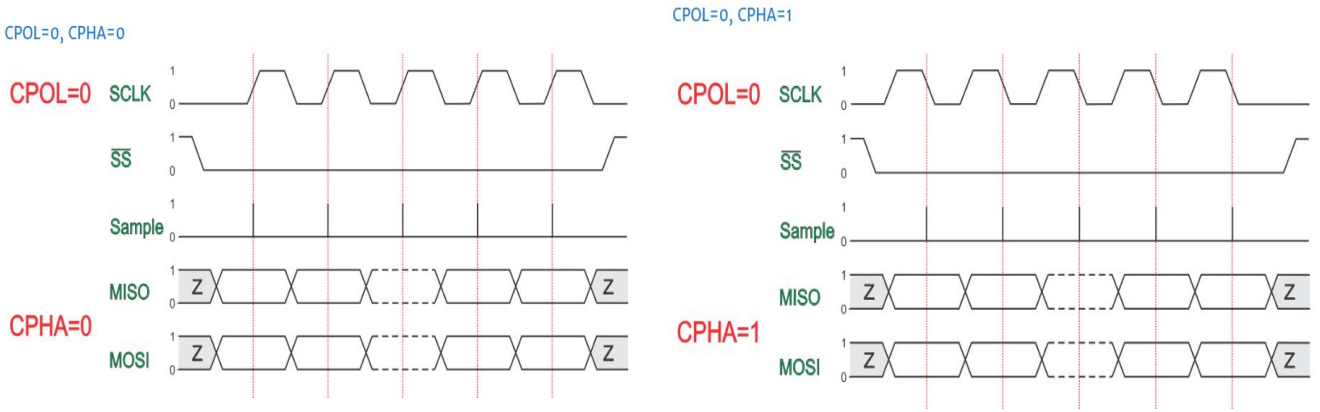


Clock Polarity and Clock Phase

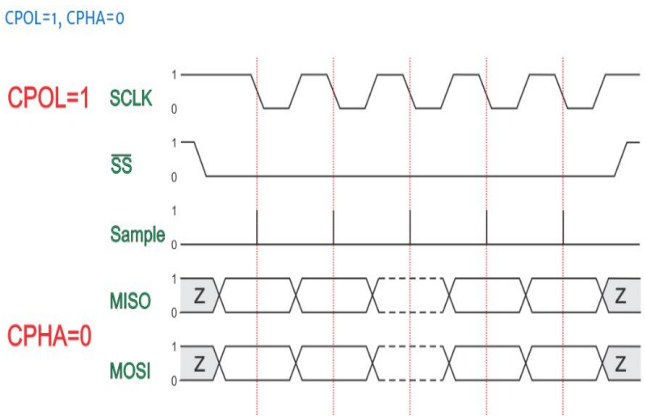


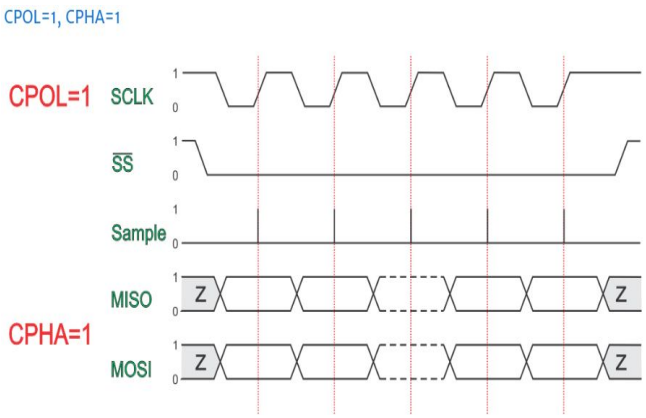
- **Rising/falling** is equivalent to CPOL = 0; the leading edge of the clock is rising and the trailing edge is falling.
- **falling/rising** is equivalent to CPOL = 1; the leading edge of the clock is falling and the trailing edge is rising.
- **sample/setup** is equivalent to CPHA = 0; data is sampled on the leading edge of the clock.
- **setup/sample** is equivalent to CPHA = 1; data is sampled on the trailing edge of the clock.

SPI Transfer Modes – Clock Polarity and Clock Phase

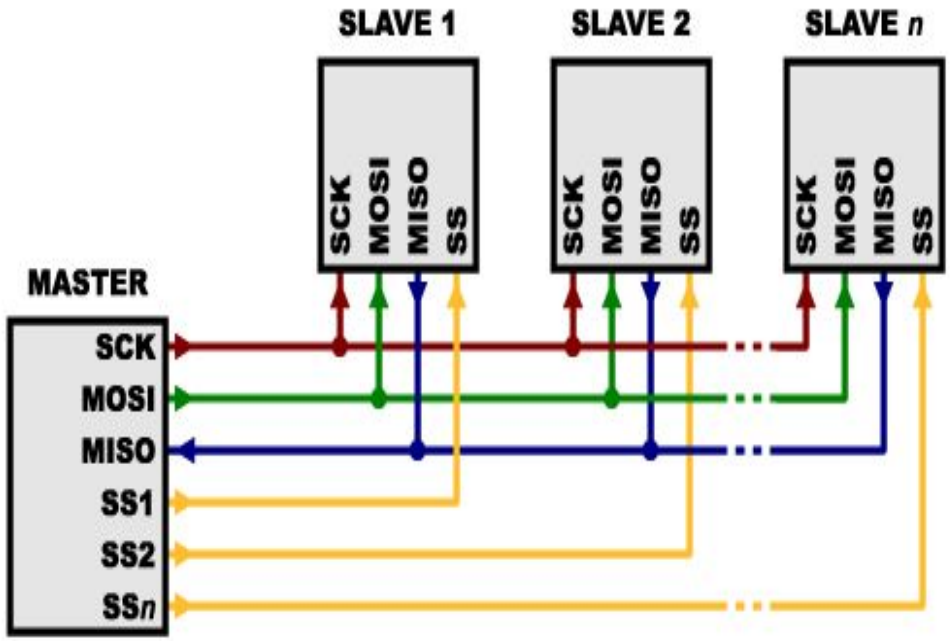


SPI Transfer Modes





SPI… Support for Multiple Slaves



Pros and Cons of SPI

Advantages of SPI:

- Higher data transfer rate than I2C (almost twice as fast) (Data Speed: 6-10 Mbps)
- The receive hardware can be a simple shift register
- It supports multiple slaves

Disadvantages of SPI:

- Uses four wires (I2C and UARTs use two)
- No acknowledgement that the data has been successfully received (I2C has this)
- The communications must be well-defined in advance (you can't send random amounts of data whenever you want)
- The master must control all communications (slaves can't talk directly to each other)
- It usually requires separate SS lines to each slave, which can be problematic if numerous slaves are needed.
- Only allows for a single master

Common Debugging Issues

- Signal Integrity
  - Clock Signal (SCLK): Ensure the clock signal is stable, with minimal jitter.
- Data Rate
  - Maximum Speed: Ensure that the SPI clock frequency does not exceed the slave's maximum supported speed.
- Clock Polarity and Phase
  - Ensure that both the master and slave devices are configured to the same clock mode.
  - Mismatched CPOL or CPHA settings will lead to data corruption.

Questions

Is clock stretching supported in SPI ?

- Answer: No

Clock Synchronization is not supported in SPI. If slave is slower in communication then solution is

- Lower the clock speed
- Use Handshaking with GPIO pins
- Use Flow control in software

Questions

Can GPIO pins be used for SPI or I2C ? (refer Wikipedia: bit-bashing or bit-banging)

- Bit banging is a technique in embedded systems and microcontroller programming where software directly manipulates GPIO (General Purpose Input/Output) pins to emulate a communication protocol, such as I2C, SPI, UART, or others, instead of using dedicated hardware peripherals.

This involves:

- Manually toggling GPIO pins to generate clock signals, data, or control signals.
- Writing code to read or write bits in a protocol-specific sequence.

Syllabus – Unit 3

- **COMMUNICATION INTERFACES :** Interfacing Buses - Serial Interfaces - RS232/UART - RS422/RS485 - I2C Interface - SPI Interface

Prasad K V K K , "Embedded/Real-Time Systems: Concepts, Design and Programming - The Ultimate Reference", Himal Impressions, New Delhi, 2003.