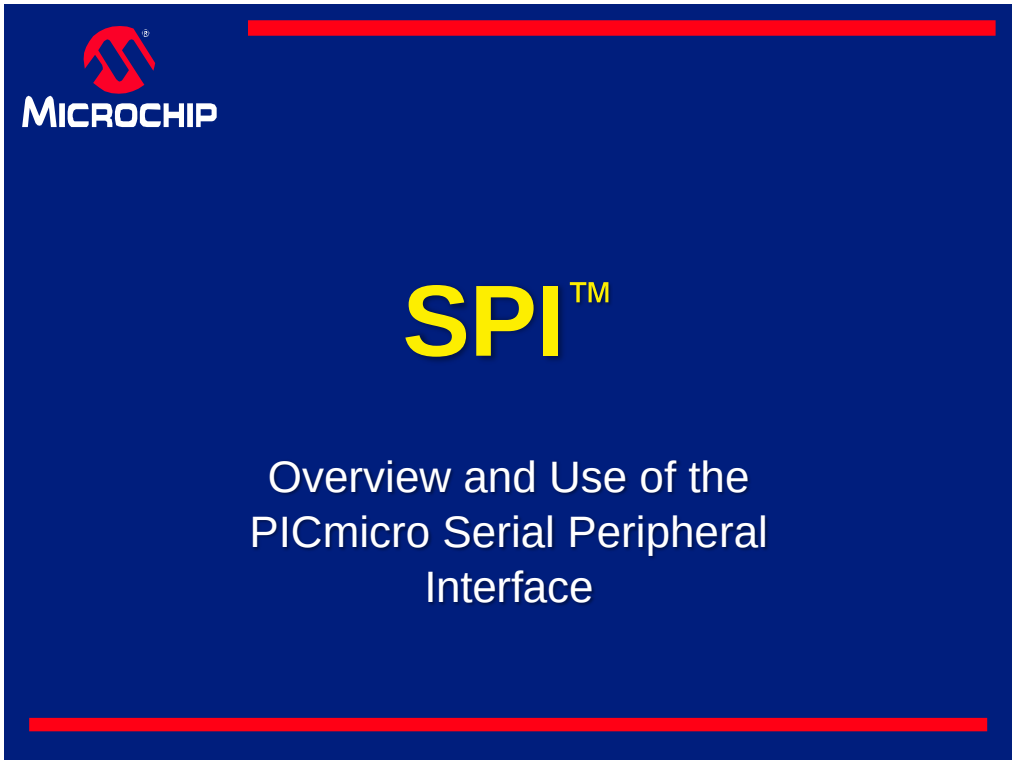
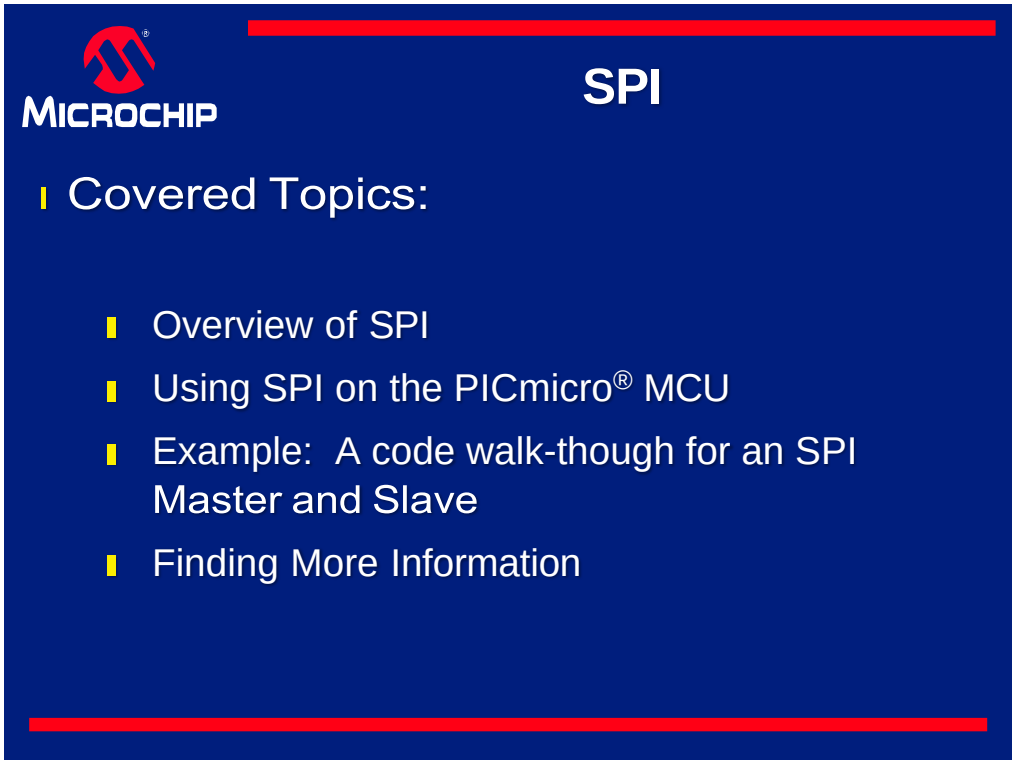


In this presentation, we will look at what the Serial Peripheral Interface, otherwise known as the SPI, is, and how it is used to communicate data to and from the PICmicro microcontroller.

SPI is frequently used when few I/O lines are available, but communication between two or more devices must be fast and easy to implement.



In this presentation, we will discuss the following topics:

We will first cover an Overview of SPI.

This section of the tutorial will introduce you to SPI and its concepts.

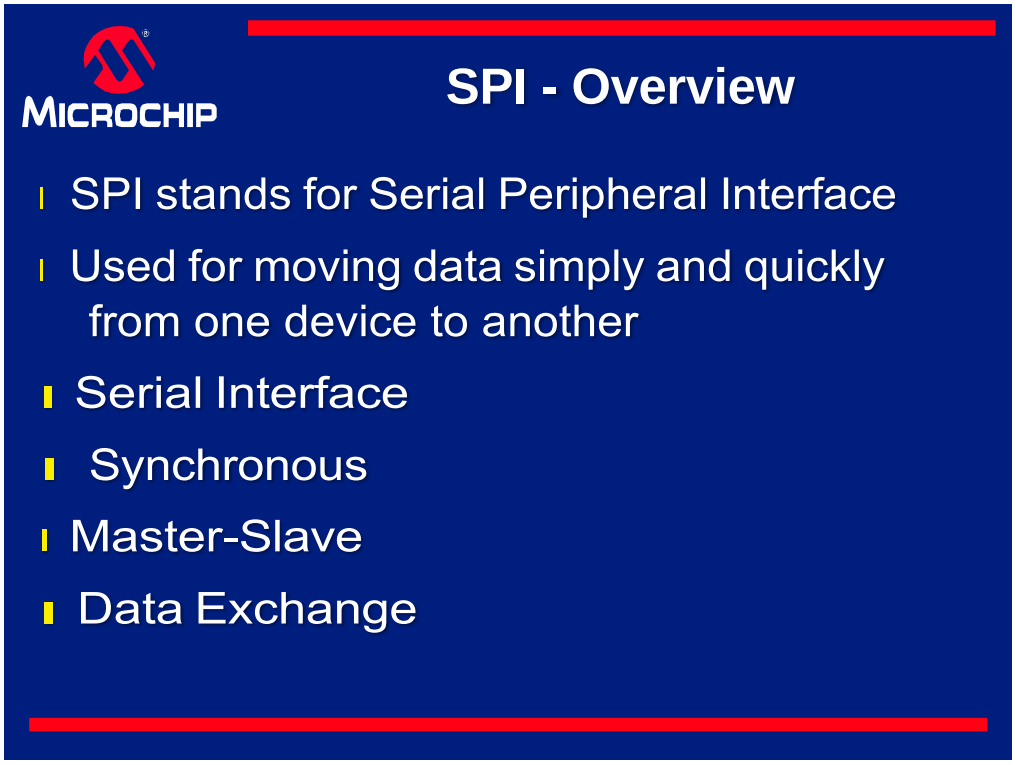
Next, we will examine the use of SPI on the PICmicro MCU.

The details of how SPI is implemented on a PICmicro device will be examined.

Then, a Code Walkthrough will be given.

The walkthough will explore code for both an SPI Master and SPI Slave. The example will have the Master generate data for sending to the slave.

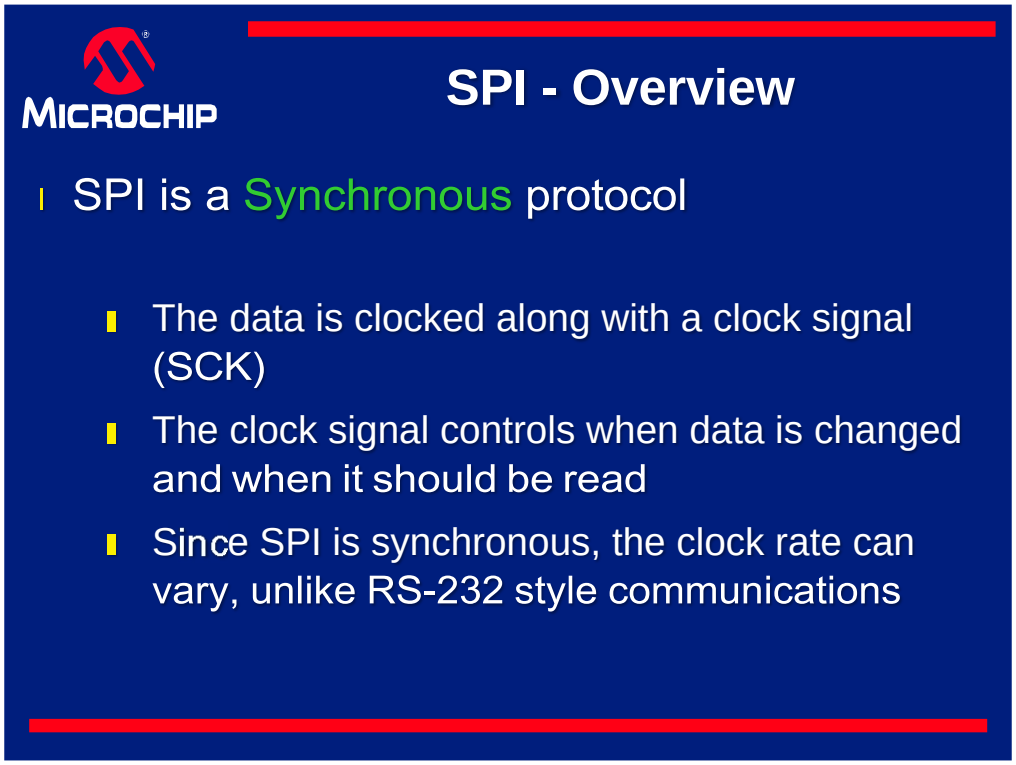
Finally, there will be a few resources given at the end of the presentation. These resources will allow you to explore in more detail the SPI interface.



SPI stands for Serial Peripheral Interface.

SPI is a synchronous protocol that allows a master device to initiate communication with a slave device. Data is exchanged between these devices. We will look at this more in detail as we progress though this tutorial.

SPI is implemented in the PICmicro MCU by a hardware module called the Synchronous Serial Port or the Master Synchronous Serial Port. This module is built into many different PICmicro devices. It allows serial communication between two or more devices at a high speed and is reasonably easy to implement.



SPI is a Synchronous protocol.

The clock signal is provided by the master to provide synchronization. The clock signal controls when data can change and when it is valid for reading.

Since SPI is synchronous, it has a clock pulse along with the data. RS-232 and other asynchronous protocols do not use a clock pulse, but the data must be timed very accurately.

Since SPI has a clock signal, the clock can vary without disrupting the data. The data rate will simply change along with the changes in the clock rate. This makes SPI ideal when the microcontroller is being clocked imprecisely, such as by a RC oscillator.

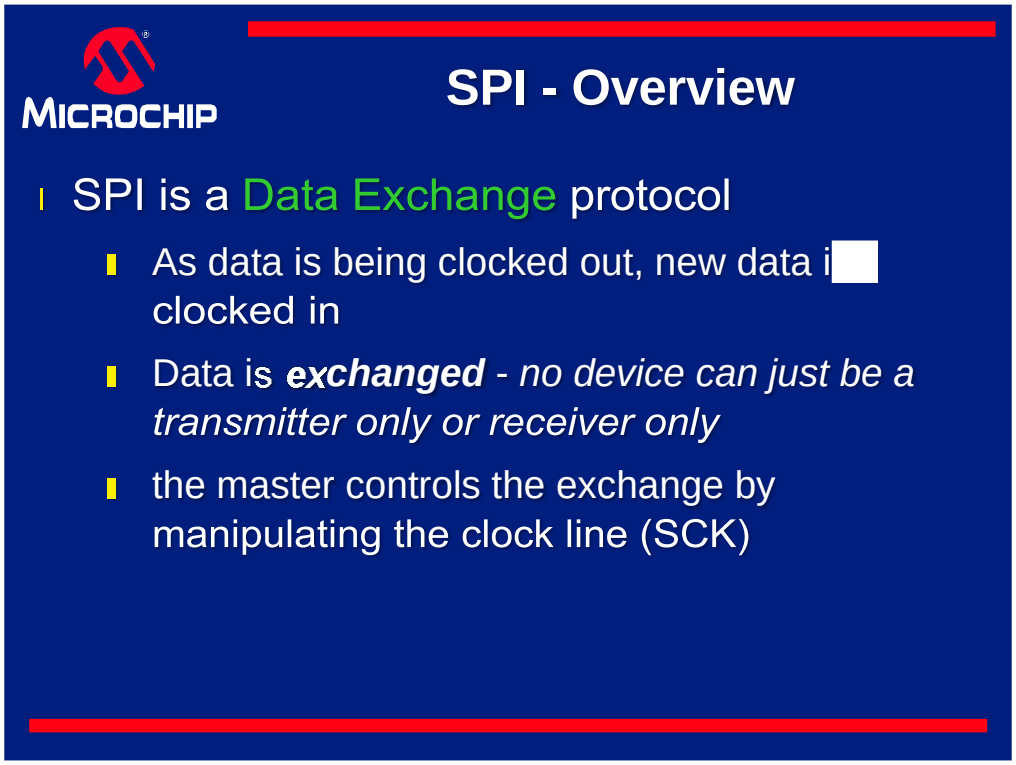
SPI is a Master-Slave protocol.

Only the master device can control the clock line, SCK.

No data will be transferred unless the clock is manipulated.

All slaves are controlled by the clock which is manipulated by the master device.

The slaves may not manipulate the clock. The SSP configuration registers will control how a device will respond to the clock input.

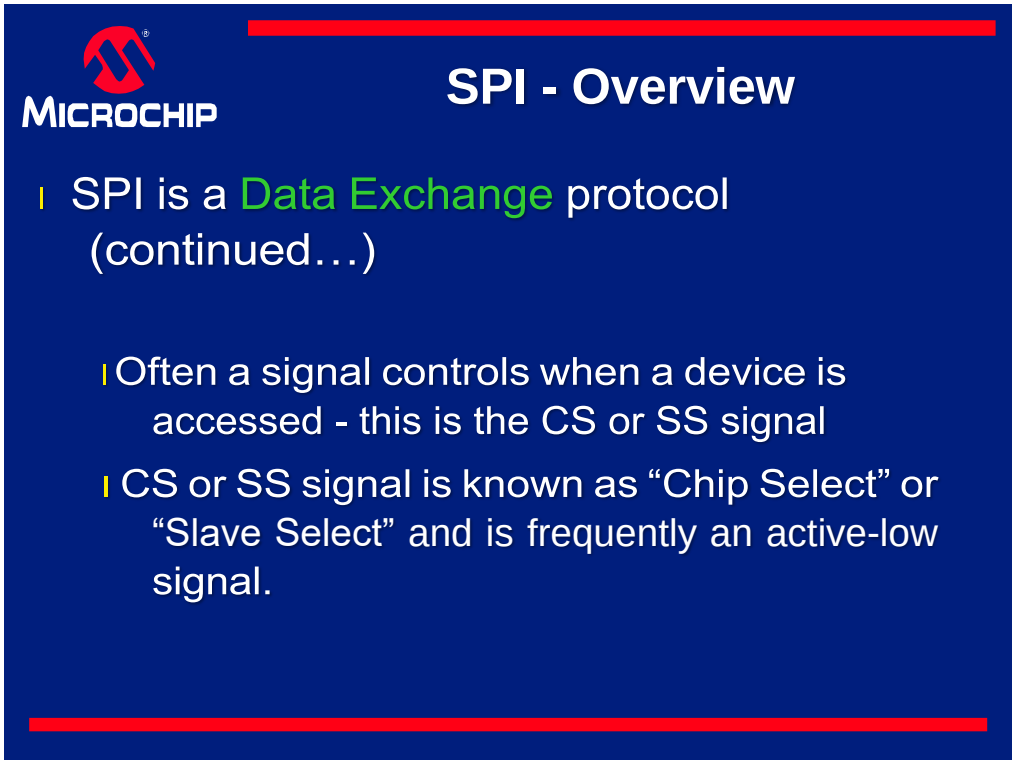


SPI is a Data Exchange protocol. As data is being clocked out, new data is also being clocked in.

When one "transmits" data, the incoming data must be read before attempting to transmit again. If the incoming data is not read, then the data will be lost and the SPI module may become disabled as a result. Always *read* the data after a transfer has taken place, even if the data has no use in your application.

Data is always "exchanged" between devices. No device can just be a "transmitter" or just a "receiver" in SPI. However, each device has two data lines, one for input and one for output.

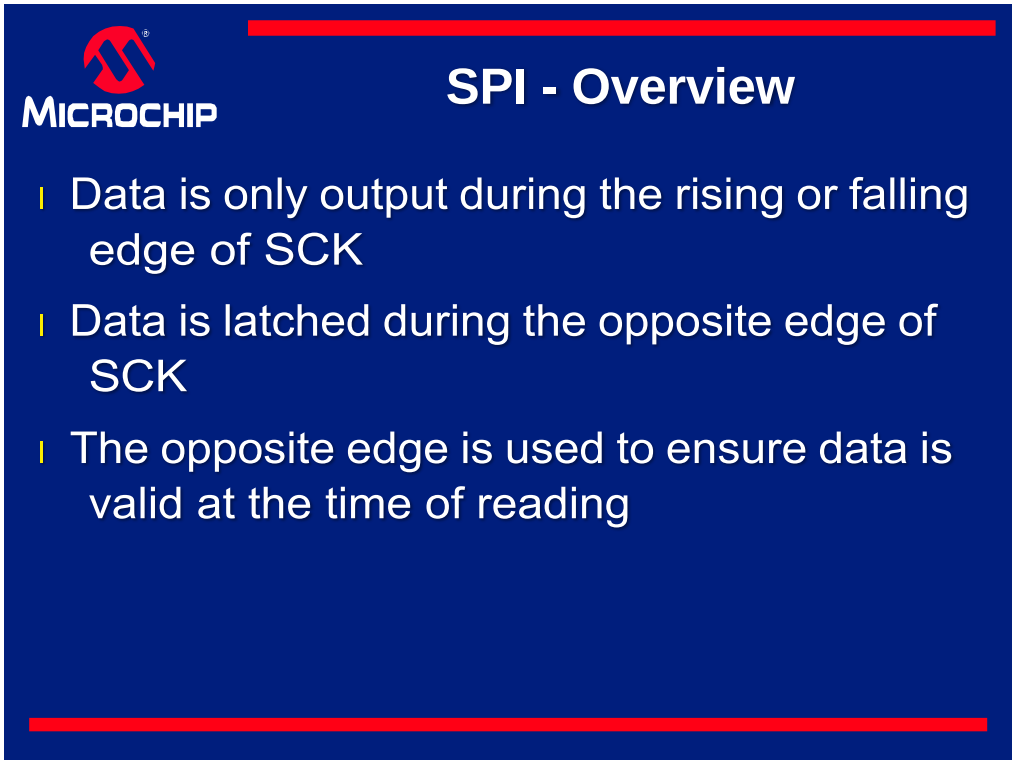
These data exchanges are controlled by the clock line, SCK, which is controlled by the master device.



Often a slave select signal will control when a device is accessed. This signal must be used for when more than one slave exists in a system, but can be optional when only one slave exists in the circuit. As a general rule, it should be used.

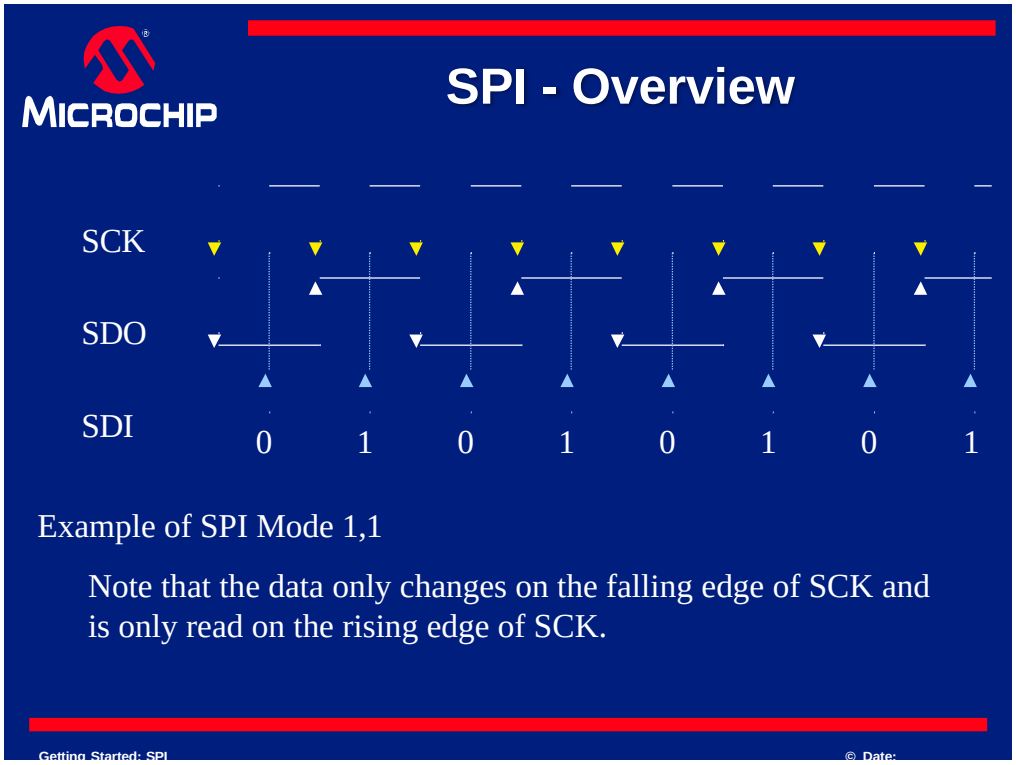
This signal is known as the SS signal and stands for “Slave Select.” It indicates to a slave that the master wishes to start an SPI data exchange between that slave device and itself. The signal is most often active low, so a low on this line will indicate the SPI is active, while a high will signal inactivity.

It is often used to improve noise immunity of the system. Its function is to reset the SPI slave so that it is ready to receive the next byte.



In SPI, data typically changes during the rising or falling edge of SCK. This is how the data is synchronized with the clock signal.

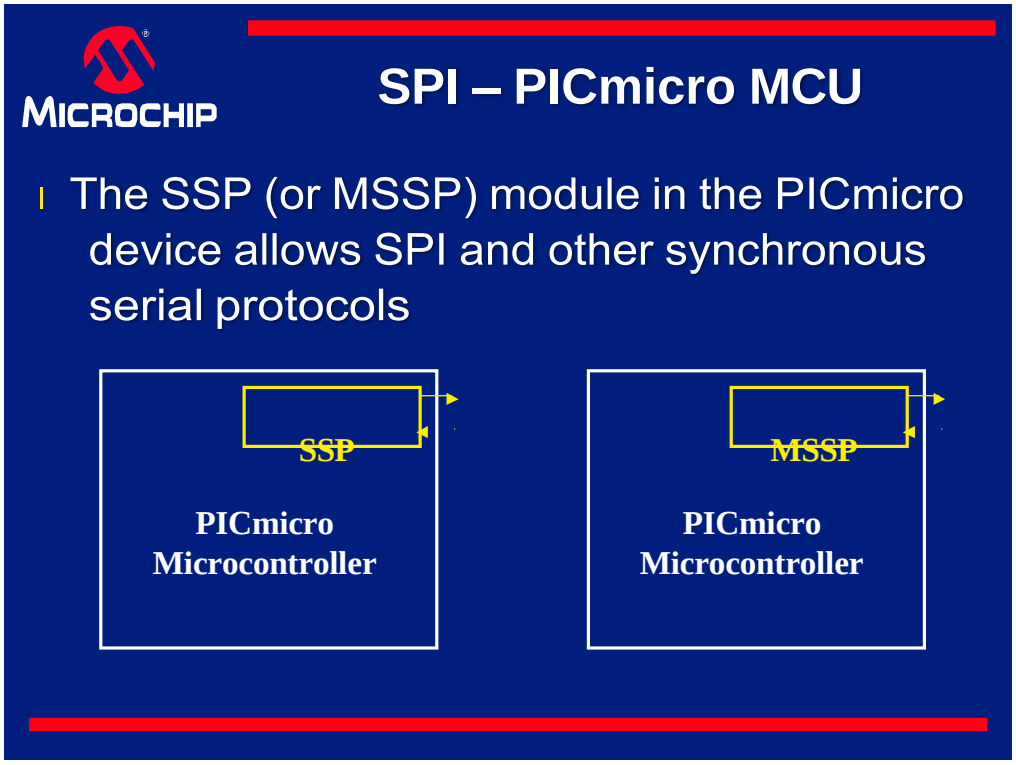
Logically, the point at which data is read is opposite from when it changes. The data is valid at the point of reading.



Here is an example of SPI communication. The arrows indicate whether the signal is experiencing a rising or falling edge at the time. SDI shows when the data is sampled.

As can be clearly seen, the sampling is done on the opposite clock edge of when the data changes.

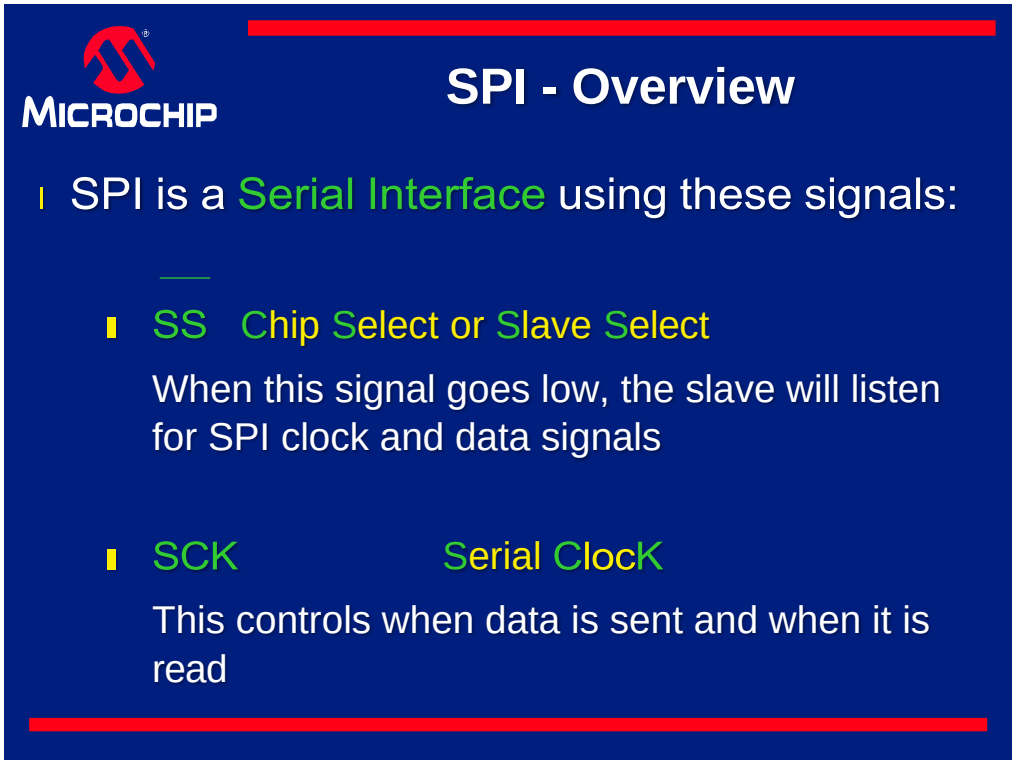
The term “Mode 1,1” refers to the different modes of SPI which will be briefly discussed later and can be found in detail in the device data sheet.



In the PICmicro device, a module is used for the SPI protocol. This module is named the SSP or MSSP module and allows SPI or I^2C to be implemented.

SPI and I^2C are both synchronous serial protocols, and hence the name of the SSP module. SSP stands for "Synchronous Serial Port". If you want to use SPI ensure your PICmicro device has this port. Check the product line card or the device datasheet to ensure it has an SSP or MSSP module.

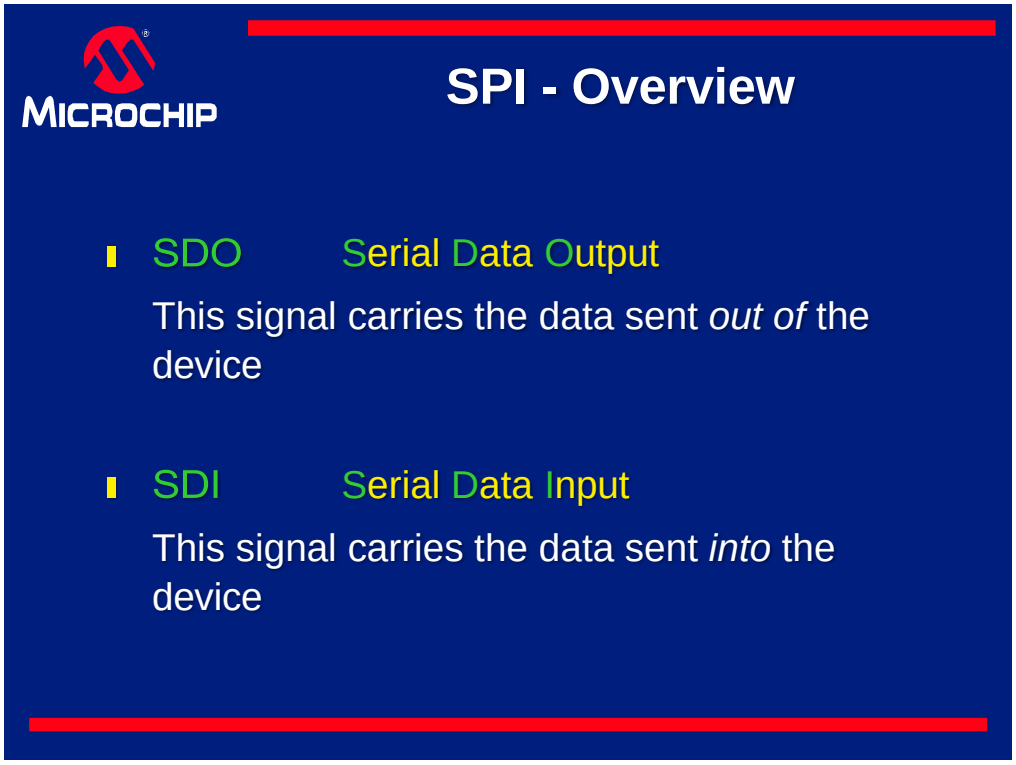
The "M" is MSSP stands for "Master" and relates to how it handles I^2C data. It does not affect its SPI performance, so either a MSSP or SSP module can be used for SPI.



SPI is a Serial Interface and uses the following signals to serially exchange data with another device:

SS - This signal is known as Slave Select. When it goes low, the slave device will listen for SPI clock and data signals.

SCK - This is the serial clock signal. It is generated by the master device and controls when data is sent and when it is read.



SDO - This is the Serial Data Output signal. SDO carries data out of a device.

SDI - SDI is the Serial Data Input line. It carries data into a device.