Serial Peripheral Interface: SPI

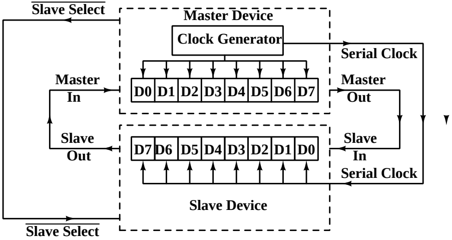
Dinesh Sharma

Department Of Electrical Engineering
Indian Institute Of Technology, Bombay

February 18, 2021

Serial Peripheral Interface: SPI

- The Serial Peripheral Interface (SPI) is a synchronous serial communication interface.
- The interface was developed by Motorola in the mid-1980s and is widely used by many peripheral devices and memories.
- The communicating devices are designated as master and slave.

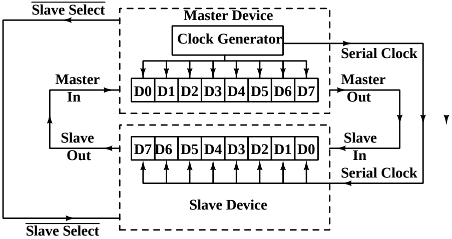


Output of the master shift register is connected to the input of the slave shift register, while the output of the slave shift register is connected to the input of the master shift register, thus forming a ring.

With clock pulses generated by the master device, there is simultaneous transfer of data between the two devices.

Serial Peripheral Interface: SPI

- There is no talker or listener – both devices provide outputs and receive inputs simultaneously.
- The master device generates the clock which is used by both devices.
- The master also provides the $\overline{\text{Slave select}}$ which enables the slave to take part in data transfer.



- The master can have multiple slave select lines – one for each slave.
- Slaves are connected in parallel to the data and clock signals.
- Only the selected device takes part in data transfer. Others do not drive their outputs.

Signals in SPI

The SPI interface defines the following signals:

MOSI: Master out, Slave in
: This is the output from the master and input to the slave for serial data.

MISO: Master in, Slave out
: This is the output from the slave and input to the master for serial data.

Sclk: Serial clock
: provided by the master. The master outputs a counted number of clock pulses to interchange the data in the two shift registers.

$\overline{SS_i}$: Slave select
: These are the slave select lines, one per slave. Typically, the microprocessor is the master device and multiple peripherals such as serial memory, ADC, DAC, real time clock chips etc. are the multiple slaves.

The master has to have a separate slave select line for each slave device. If the number of slaves is large and this many pins cannot be provided, a decoder can be used.

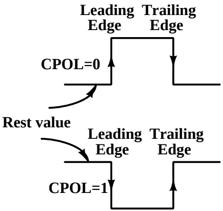
Data transfer in SPI

What if we don’t want two way communication?

- Data transfer in SPI is always bi-directional.
- While simultaneous data transfer is an advantage in some applications such as data streaming, simplex (one way) data transfers are much more common.
- If the master wants only to write the data to the peripheral, it places the data it wants to write in the shift register and ignores the data that arrives from the slave.
- If the master wants only to read, it just leaves the stale data in the shift register or loads a dummy command and carries out a transfer. After the transfer, the received data is read. The slave is supposed to ignore the data it received.

Clock Polarity and Phase

The master and the slave must agree on the clock polarity and phase for data transfer. Motorola has defined four modes based on clock polarity (CPOL) and phase (CPHA) combinations.



- The clock polarity decides the resting or inactive value on the Sclk line.
- CPOL = 0 defines that the clock will rest at 0 when idle and a clock pulse will have a leading edge from 0 → 1 and a trailing edge from 1 → 0.
- CPOL = 1 defines a clock signal which rests at 1 when idle. A clock pulse consists of a 1 → 0 transition as the leading edge and a 0 → 1 transition as the trailing edge.

It is possible to convert between the two choices with a single inverter in the Sclk line.

Clock Polarity and Phase

Clock phase decides the timing of data changes on the MOSI and MISO lines.

- CPHA=0 indicates that the MOSI and MISO lines will change data on the trailing edge while the inside circuit captures data at (or shortly after) the leading edge of the clock cycle.
- The driver for the data lines should hold the data steady till the next trailing edge.
- For the first cycle of the clock, the first bit must already be on the MOSI line before the leading edge of the clock arrives.
- Thus, for CPHA=0, there is half a cycle with clock at idle, followed by half a cycle with clock asserted.

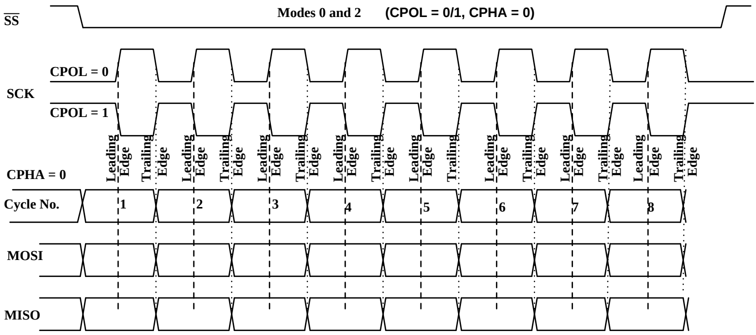
CPOL=0 and CPHA=0 are the most commonly used combination for SPI communication.

Clock Polarity and Phase

- CPHA=1 indicates that the “output” side will change data on the leading edge while the “input” side captures the data on (or shortly after) the trailing edge of the clock cycle.
- The “output” side should hold the data steady till the next leading edge.
- Thus, for CPHA=1, there is half a cycle with clock asserted, followed by half a cycle with clock idle.
- For the last cycle of the clock, drivers should hold the data steady till slave select is deasserted.

The four combinations of clock polarity and phase are often summarised as a mode number, with CPOL as the more significant bit and CPHA as the less significant bit.

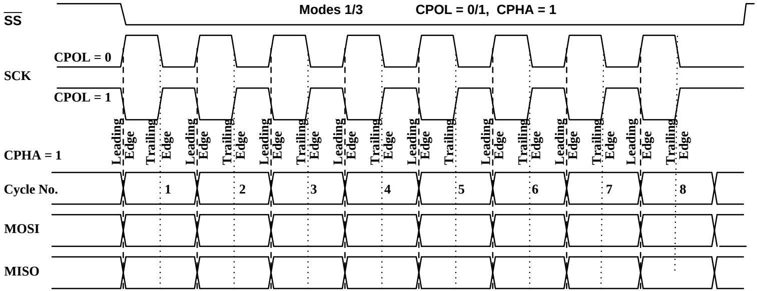
CPHA = 0 or Modes 0 and 2



Data sampled on leading edge, changed on trailing edge.
Half an idle cycle before first leading edge.

As long as the same mode is used by all devices, the exact clock frequency is immaterial.

CPHA = 1 or Modes 1 and 3



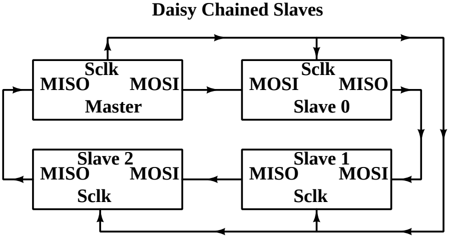
Notice that 9 clock cycles will be required for transferring 8 bits. This is because of the extra half clock cycle delay required to be inserted for ensuring that data change and sampling take place at different clock edges.

Data Size

- The most common size of data that is transferred at one go is a byte (8bits).
- However, other data sizes may be used by some devices.
- For example some audio codecs, such as the TSC2101 by Texas Instruments, use 16 bit Shift registers for transferring 16 bits of data at a time.
- Some other devices use twelve-bit shift registers – for example some 12 bit digital-to-analog or analog-to-digital converters.

Daisy chaining

One problem with SPI is that it needs a separate slave select signal for each slave. This may be a problem for many processors with a pin constraint.



- An alternative to connecting slave devices in parallel is to daisy chain these, with a single slave select used for all.
- Now we do not need multiple slave select signals.

The master can control where the data ends up by controlling the number of clock pulses to be supplied.