# Communications

Anurag Dwivedi
Rudra Pratap Suman

# Scope of Communication

Telephones and Cell Phones

# Scope of Communication

Internet

# Scope of Communication

Wireless networks

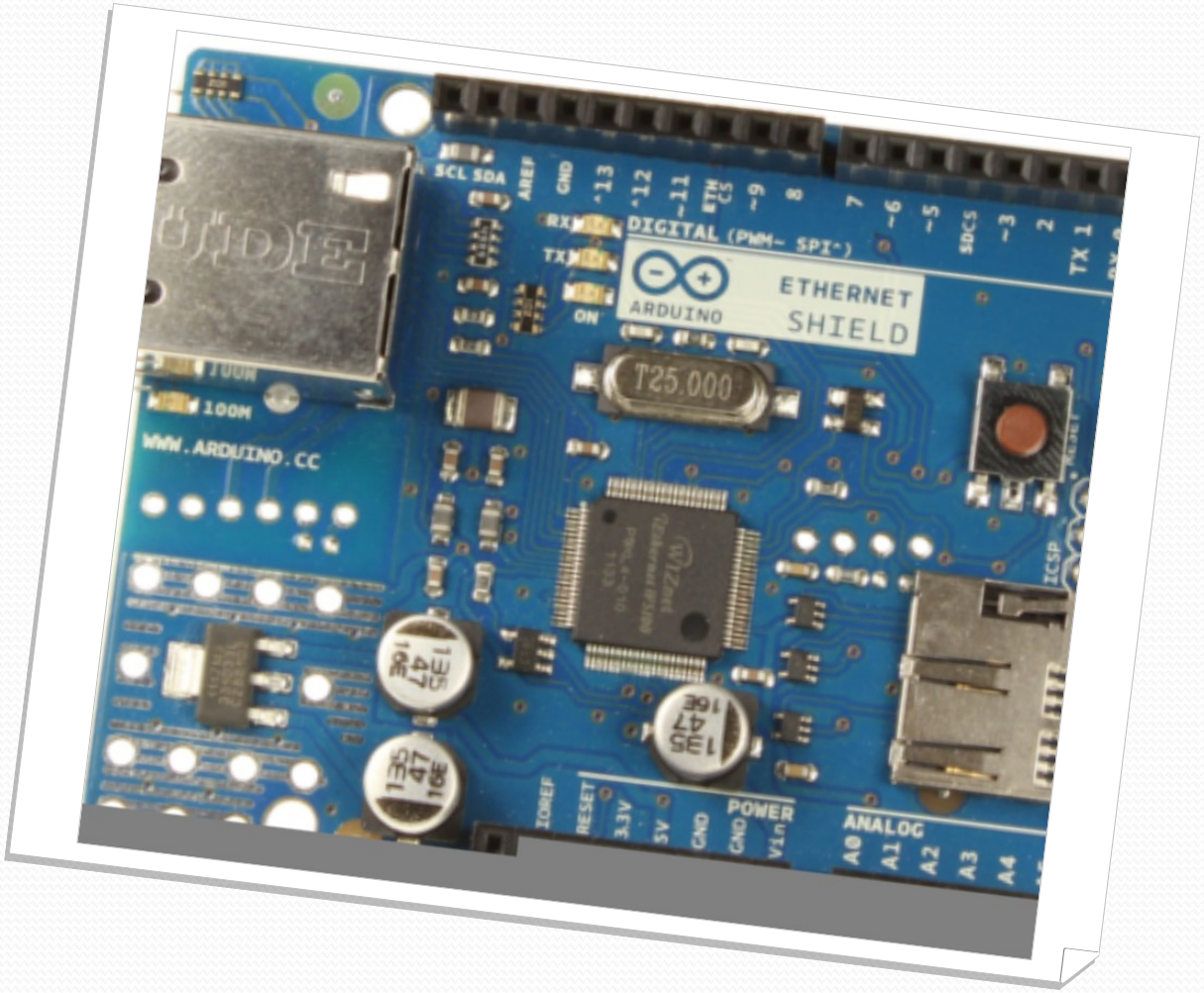
# Scope of Communication

Satellite Networks

# Our Interests !!

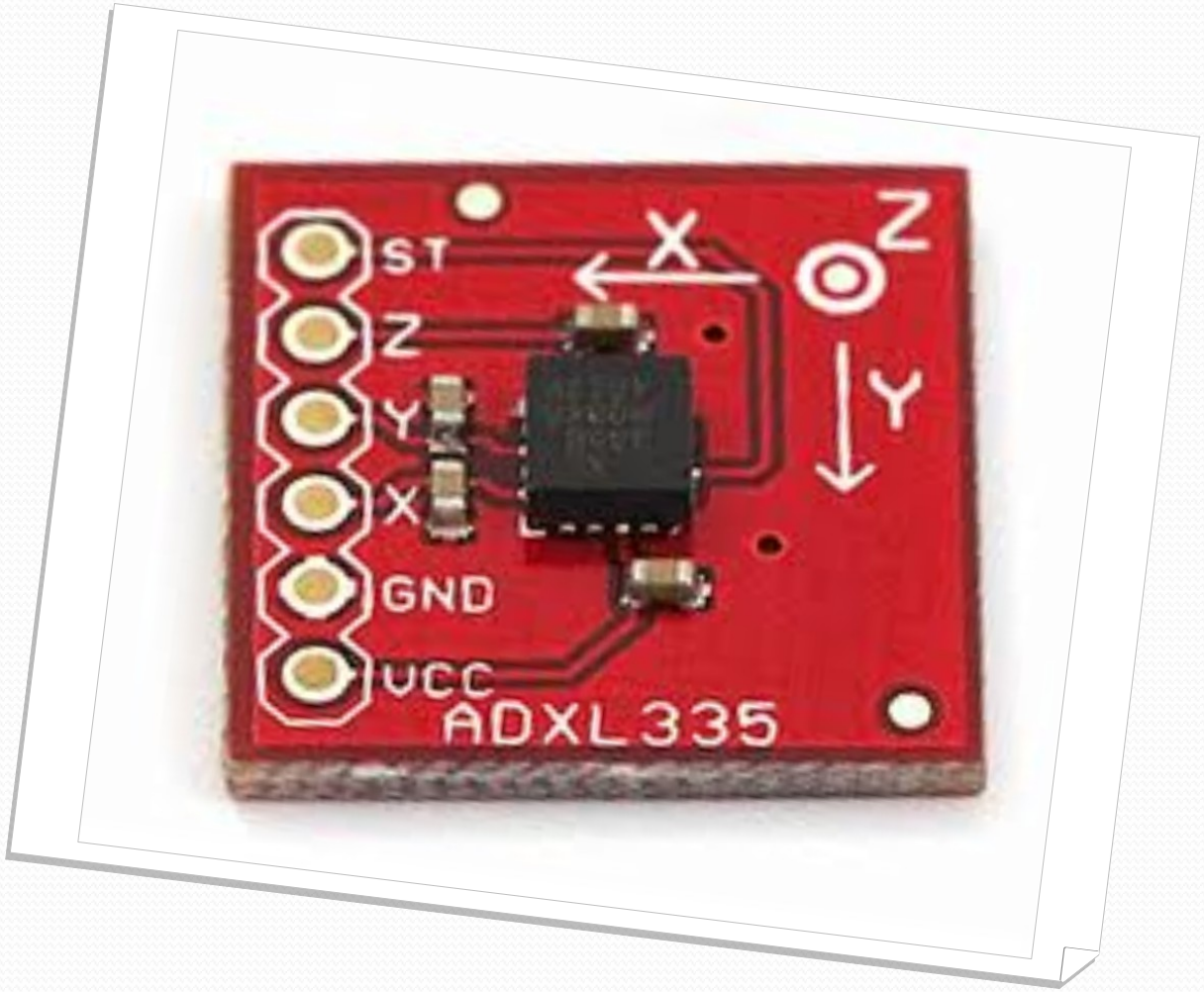
# Arduino Ethernet Shield

Uses SPI to communicate with Master board



# Accelero meters

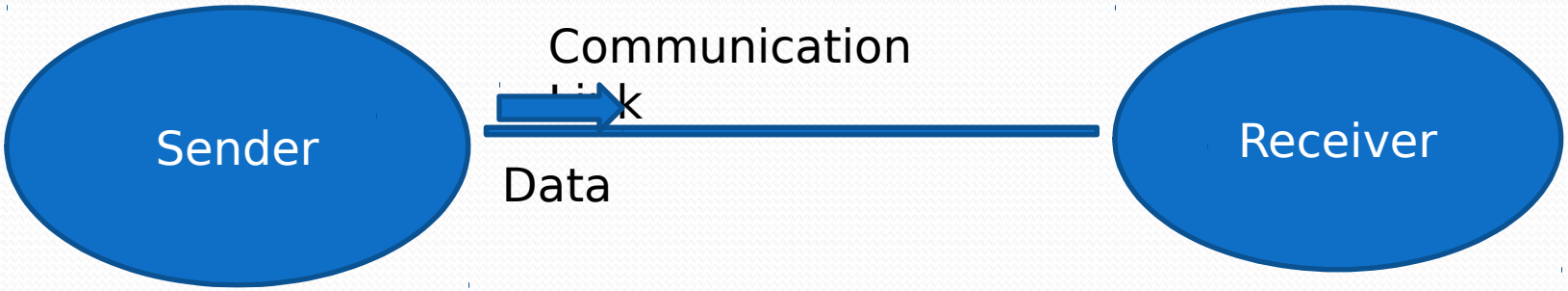
Communic ation through SPI



# JPG Color Camera

Uses UART to communicate with Master board

# Essentials of Communication




But this simple model requires many guarantees.

# Guarantees in Communications

- The communication link exists.
- The communication link is sound.
- The sender and receiver are the correct nodes.
- The sender is sending the correct data.
- The receiver is able to correctly interpret the incoming data.

# Protocols in Communication

- In order to have robust communication, the guarantees needs to be realized.
- To do so, we need an elaborate  and standardized mechanism.
- These standard rules that defines the parameters of communications and ensures these guarantees are called protocol.

# Advantages of Protocols

- Standardized, so interoperability is ensured.
- Usually include error-detection and error-correction mechanisms.
- Are available as implemented chips that can be directly used.

# Types of Protocols

- There are different ways of categorizing protocols
- First Categorization :

Serial Mode Transfer

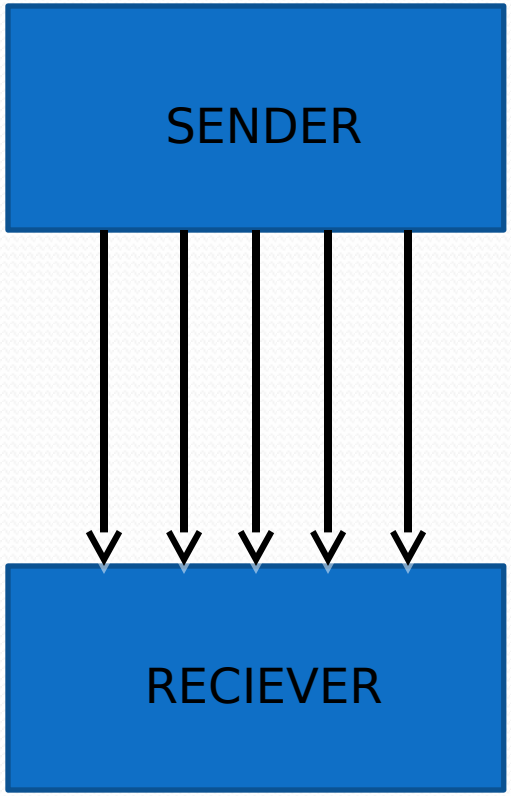
Parallel Mode Transfer

- Second Categorization :

Synchronous Mode Transfer

Asynchronous Mode Transfer

# Serial and Parallel Mode




SERIAL MODE
PARALLEL MODE

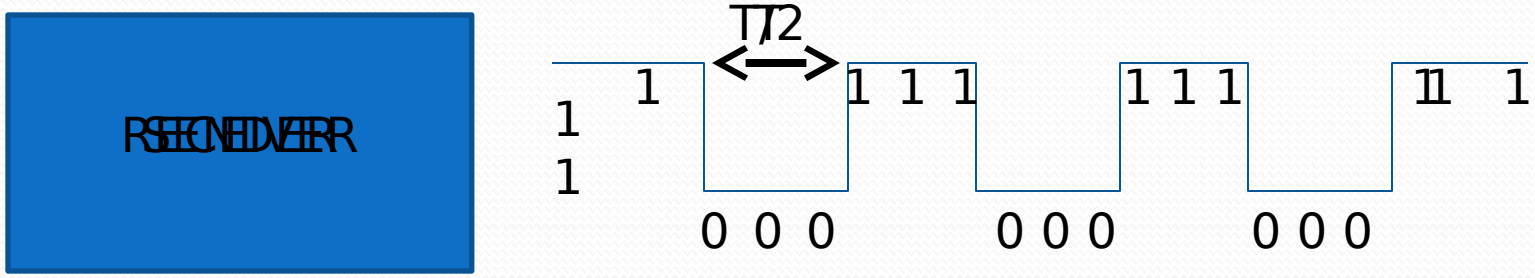
# Serial Vs Parallel Mode

| Parameter | Serial Mode | Parallel Mode |
|---|---|---|
| Reliability | Reliable | Unreliable |
| Speed | Slow | Fast |
| Power | Low | High |
| Cost | Low | High |
| Complexity | High | Low |
| Range | Long | Short |

# Synchronous and Asynchronous Mode

- Pertains to sender-receiver synchronization.
- Sender sends data at a certain speed. For flexibility, protocols allow for multiple speeds.

# Need of Synchronization



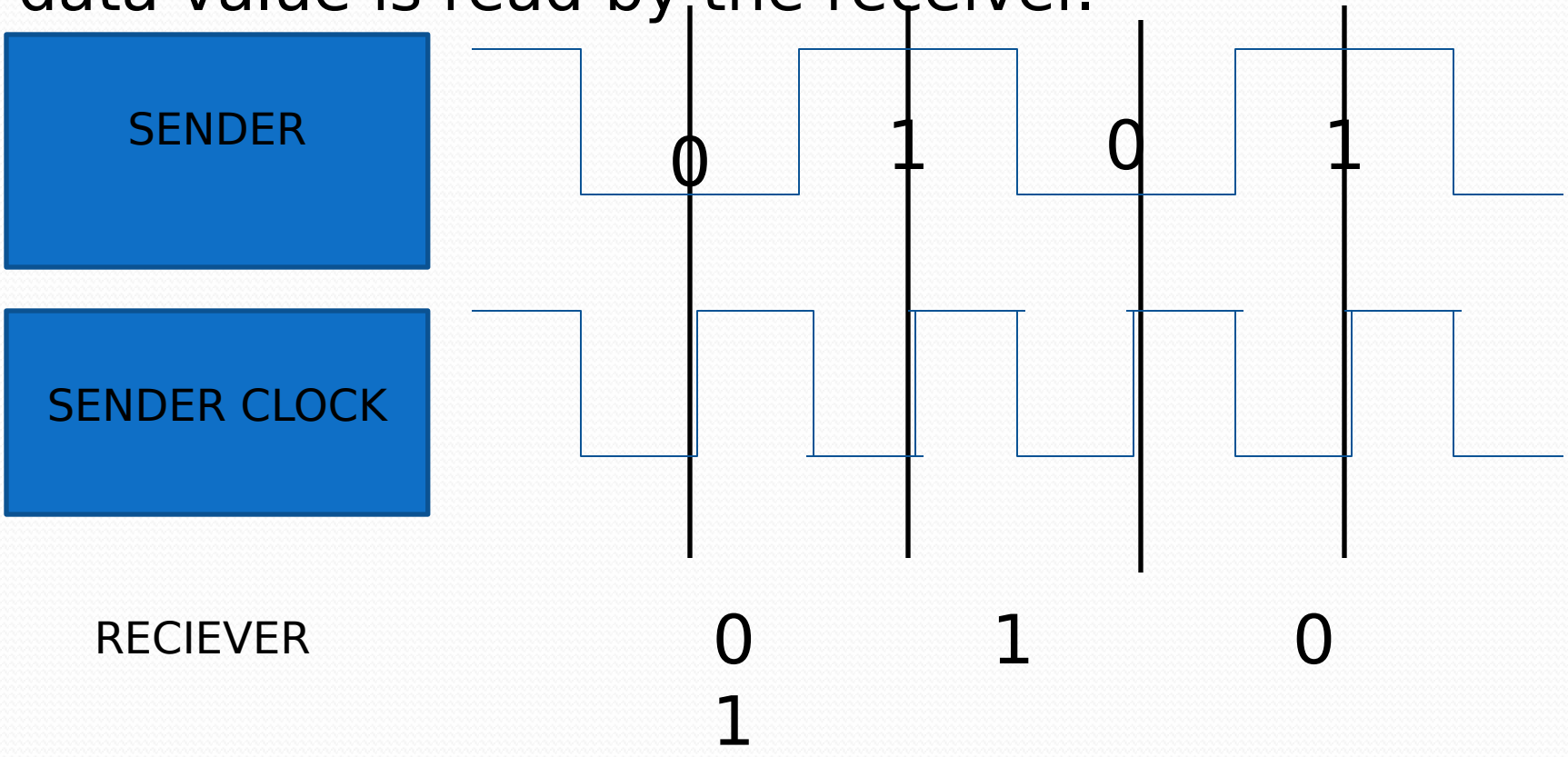

Suppose Sender sends data with a Time Period of T

What if Receiver doesn't know the speed and assume it to be say T/2

The Data received will be

# Synchronous Mode

- Sender sends a clock signal along with data at every rising / falling edge of the clock, the data value is read by the receiver.



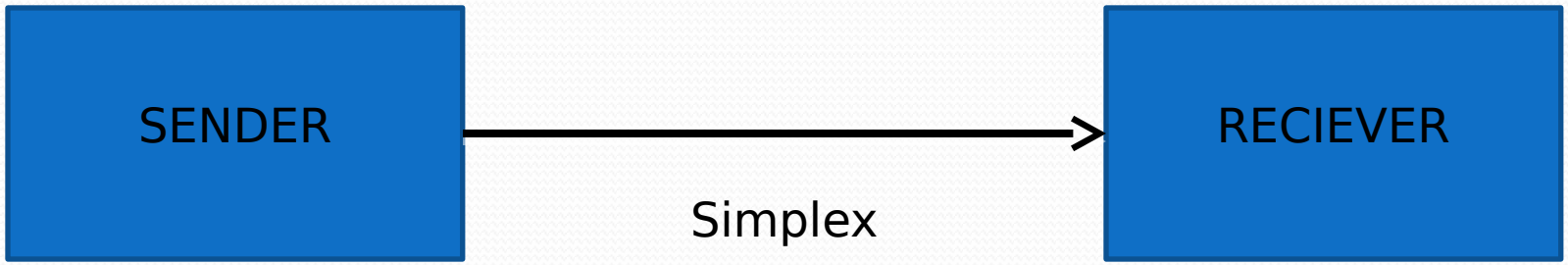
# Asynchronous Mode

- There is no clock signal.
- The receiver and the sender communicate at a predetermined speed (bauds or bits per second).
- Baud Rate : Baud Rate is a measurement of transmission speed in asynchronous communication. The devices that allows communication must all agree on a single speed of information - 'bits per second'.

# Synchronous Vs Asynchronous Mode

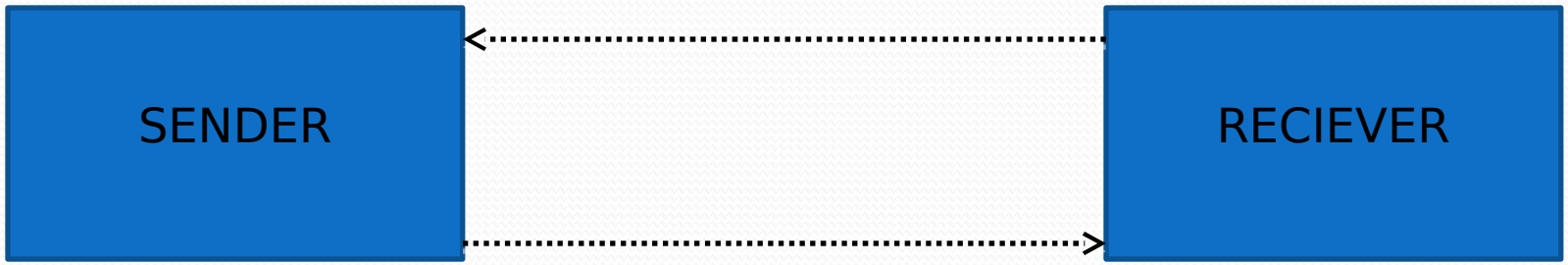
| Parameter | Synchronous | Asynchronous |
| --- | --- | --- |
| Reliability | ✓ Reliable | ✗ Error Prone |
| Cost | ✗ Expensive | ✓ Inexpensive |
| Complexity | ✗ Complicated | ✓ Simple |

# Transmission Modes




Only one way transmission takes place

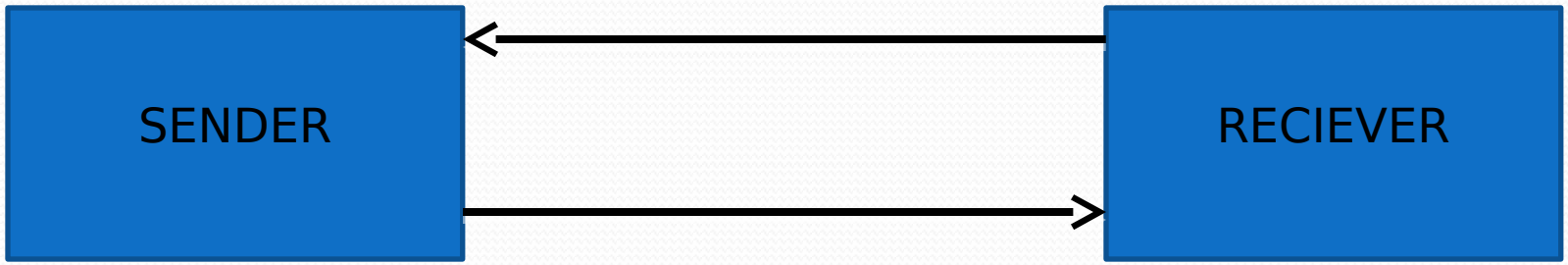
# Transmission Modes




Half-Duplex

Two way transmission takes place but only one end can communicate at a time

# Transmission Modes




Full-Duplex

Two way transmission takes place and both end can communicate simultaneously

# SPI – Serial Peripheral Interface

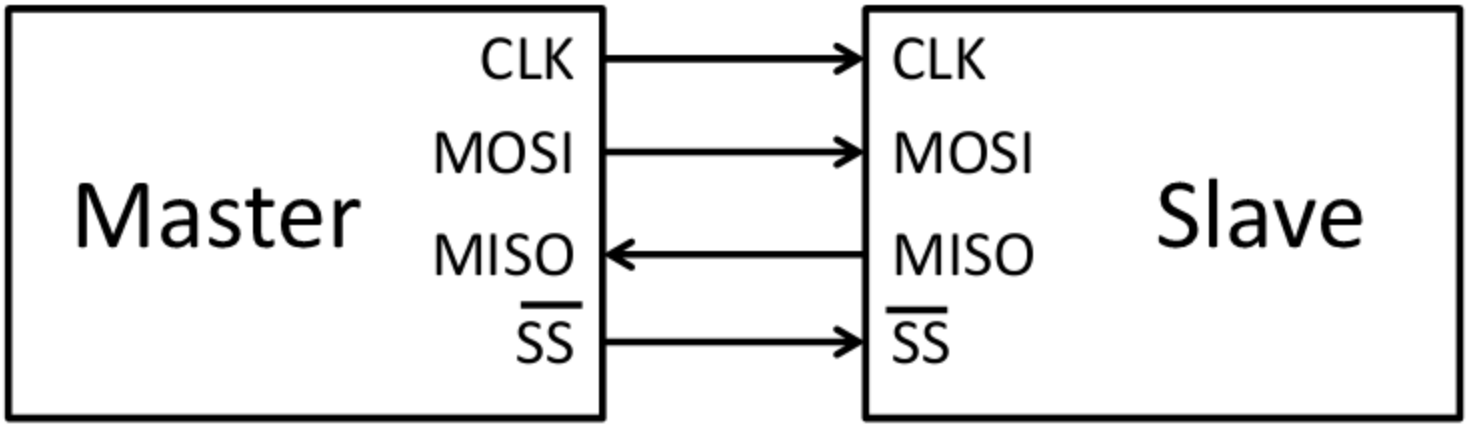
# SPI

- Serial ??
- Because it works on serial mode of transfer. It is also synchronous and full duplex.
- Peripheral Interface.
- Because it has the capability of communicate with many nodes.
- How?? Let us see.

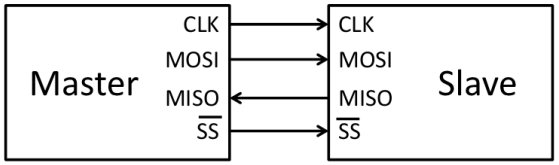
# SPI

- In SPI, the sender and receiver follows a master-slave relationship.
- There may be multiple nodes in the network.
- One node is master, the rest are slaves.
- The communication is always initiated by the master.
- The slaves can communicate only with the master.
- How do master selects the slave??

# SPI Schematics: Single Slave

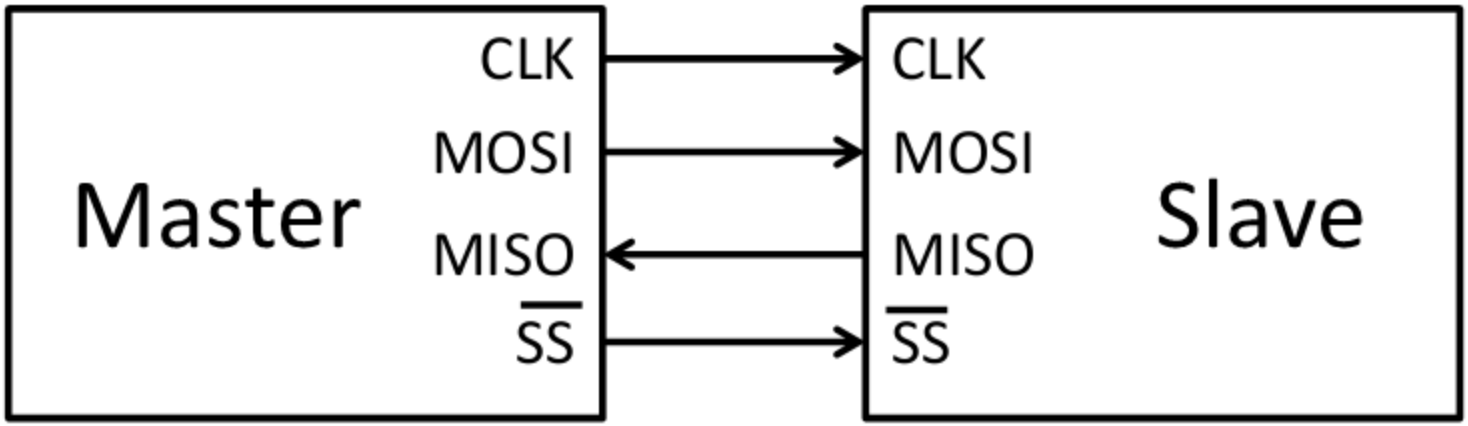


# SPI Pins

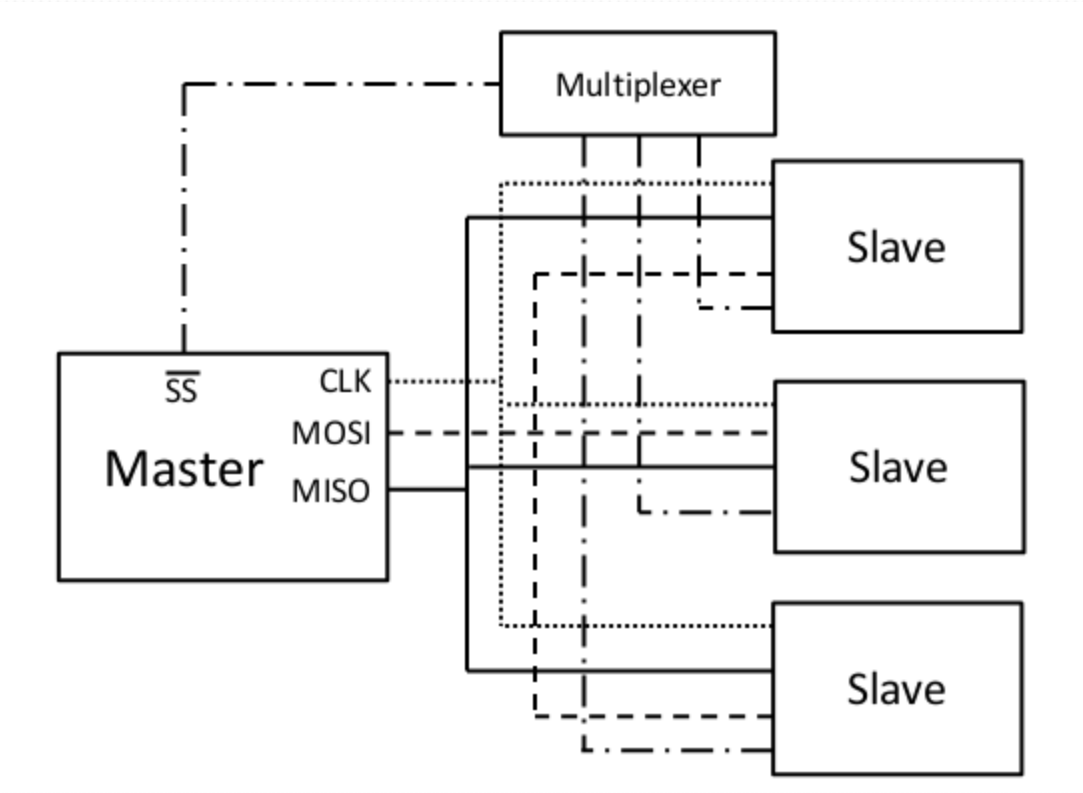



- CLK is generated by Master and is used as the mode is synchronous.
- MOSI is Master Out Slave In: Data sent by Master to Slave.
- MISO is Master In Slave Out: Data sent by Slave to Master.
- $\overline{S}\overline{S}$ is slave select: Slave communicates with Master only if this pin’s value is set as LOW.

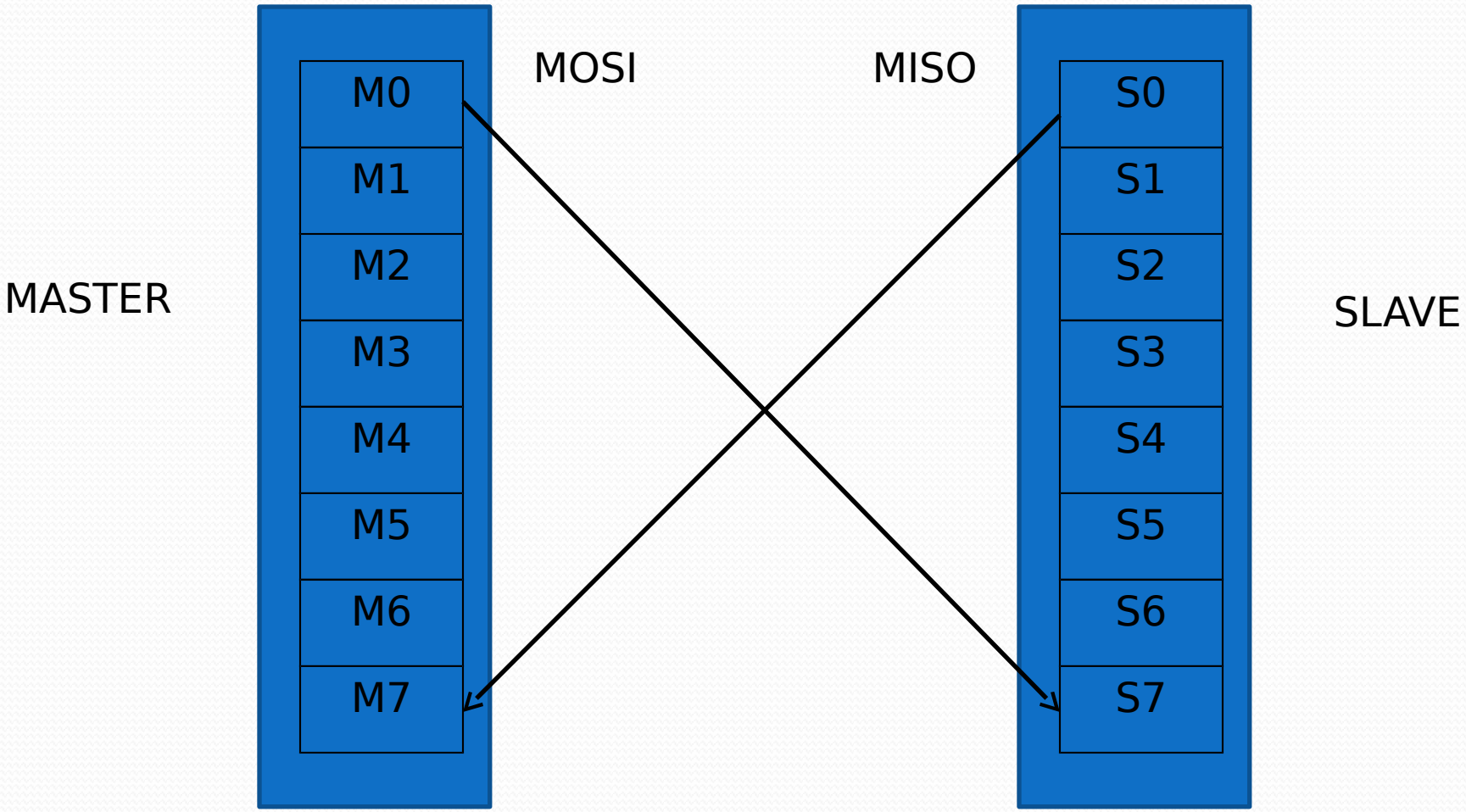
# SPI Schematics: Single Slave



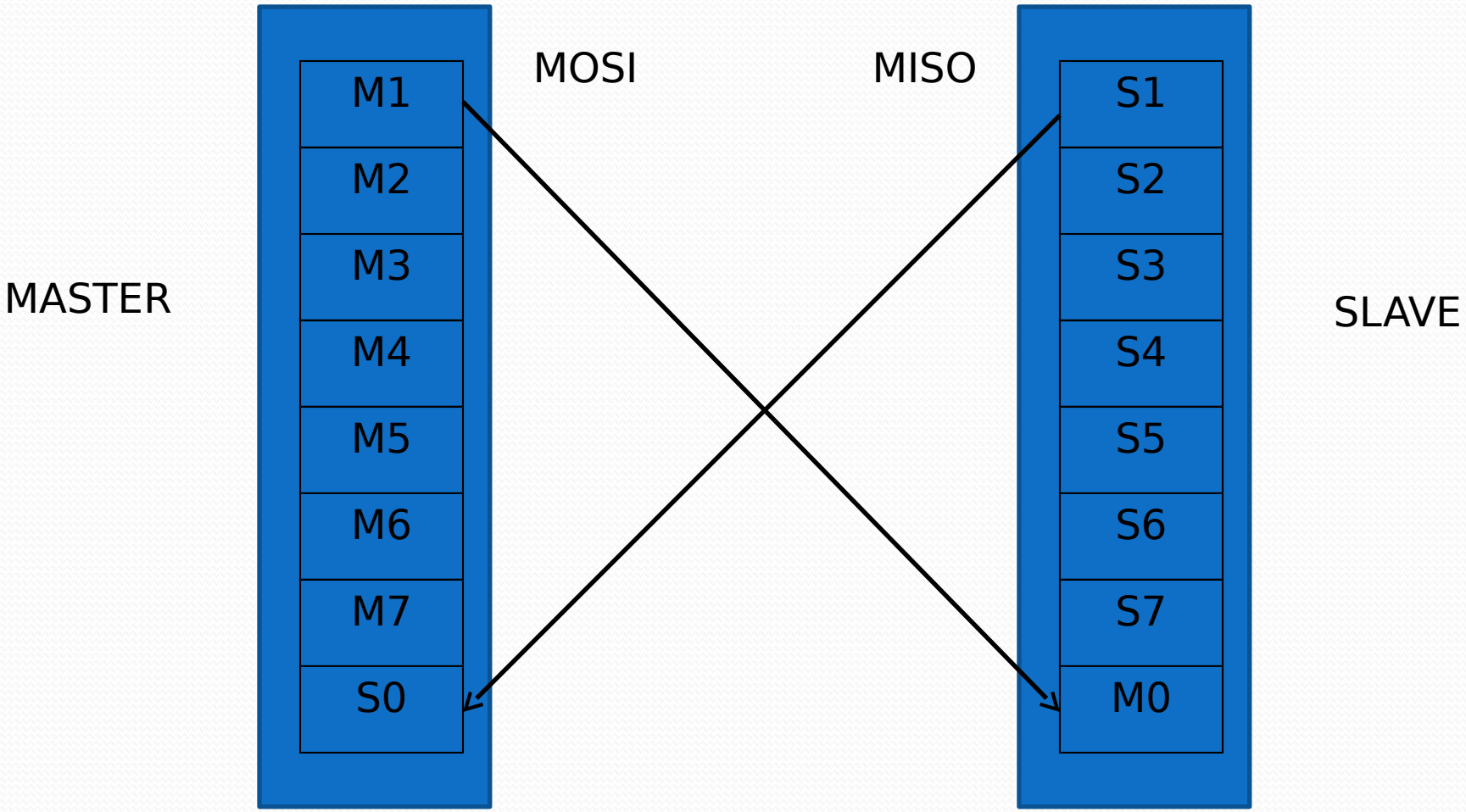
# SPI Schematics: Multiple Slaves



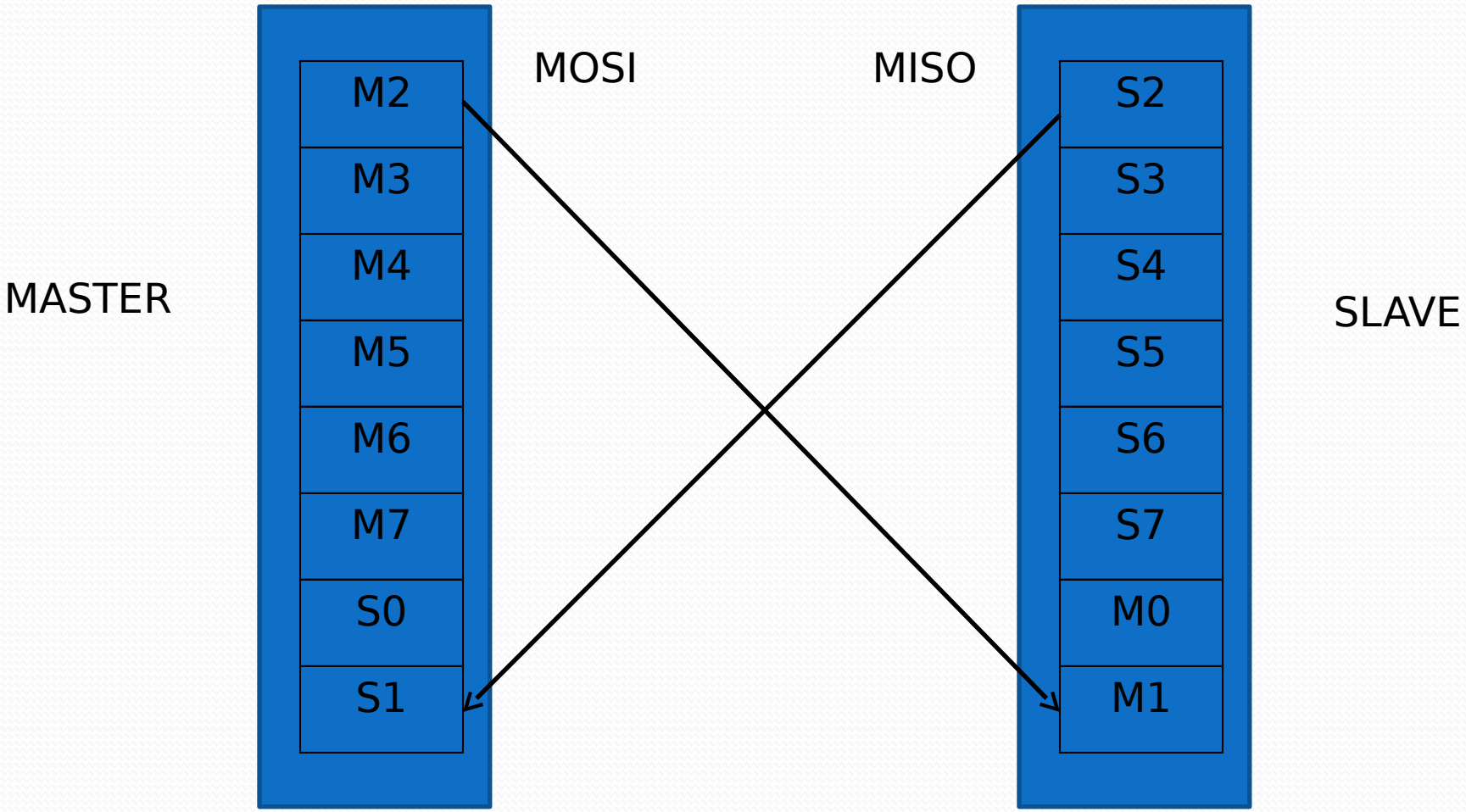
# Data Transfer in SPI



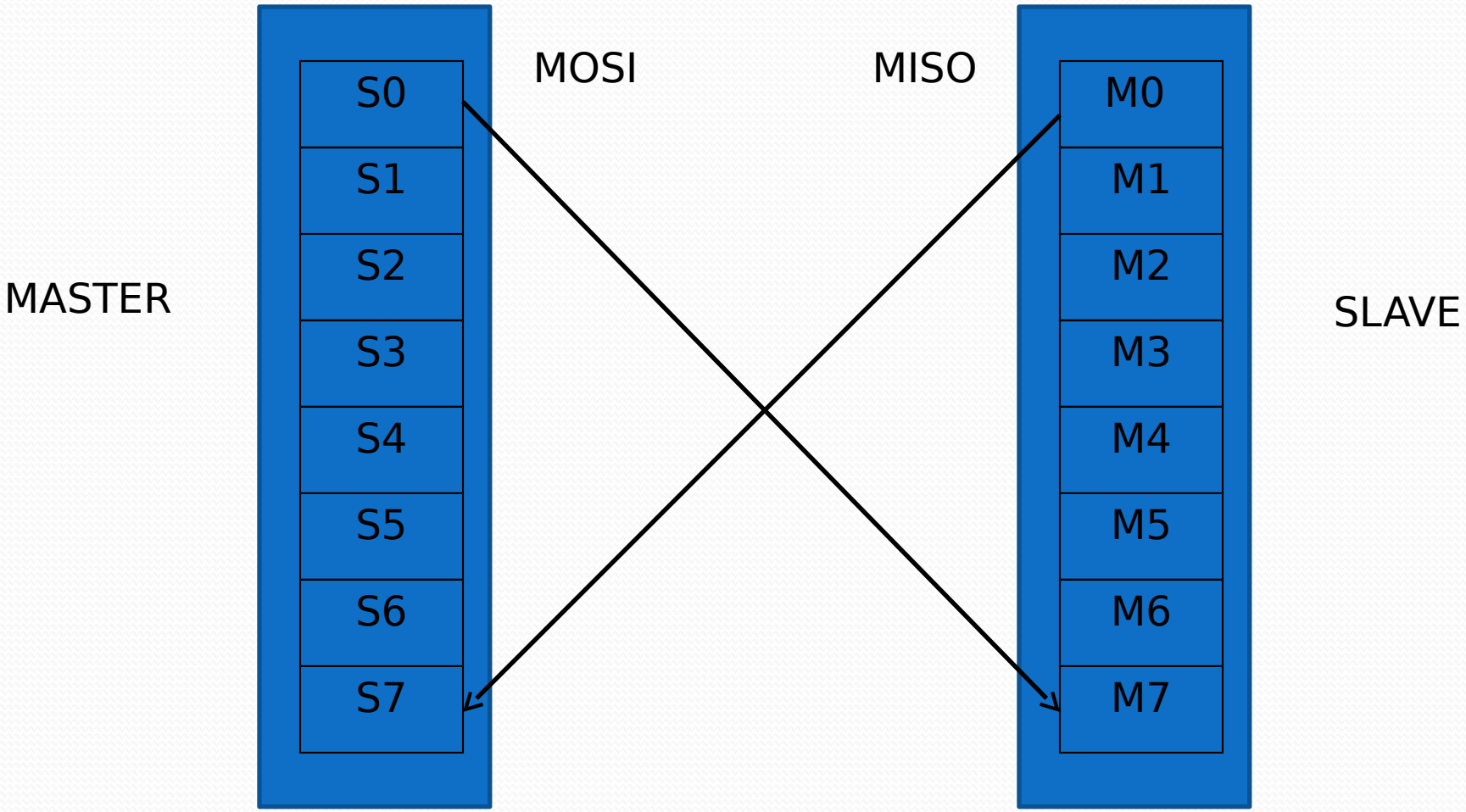
# Data Transfer in SPI



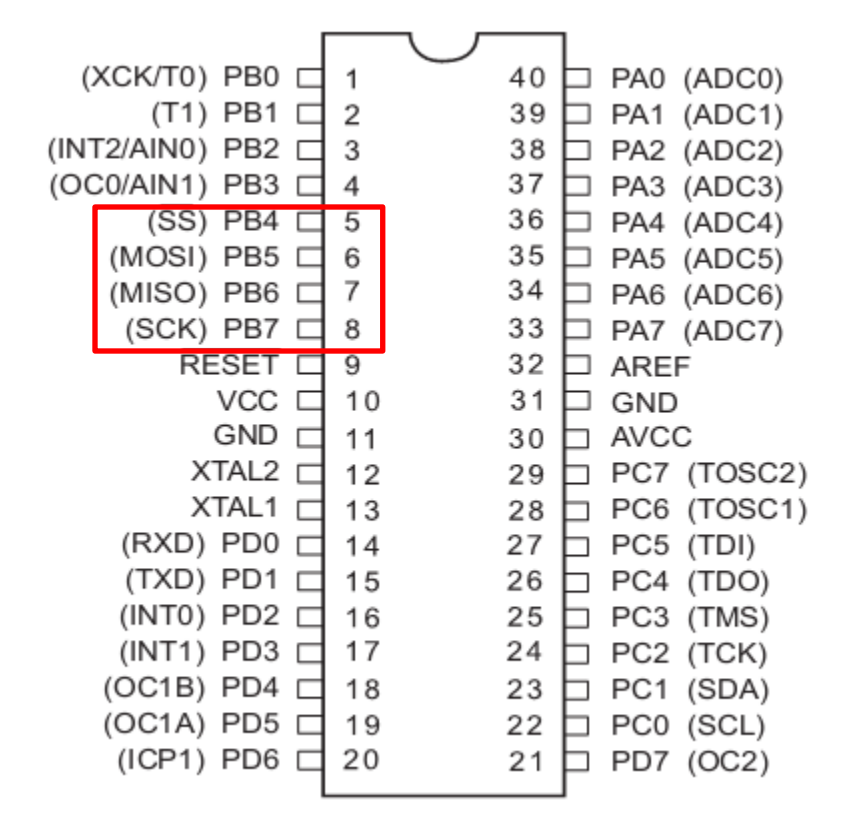
# Data Transfer in SPI



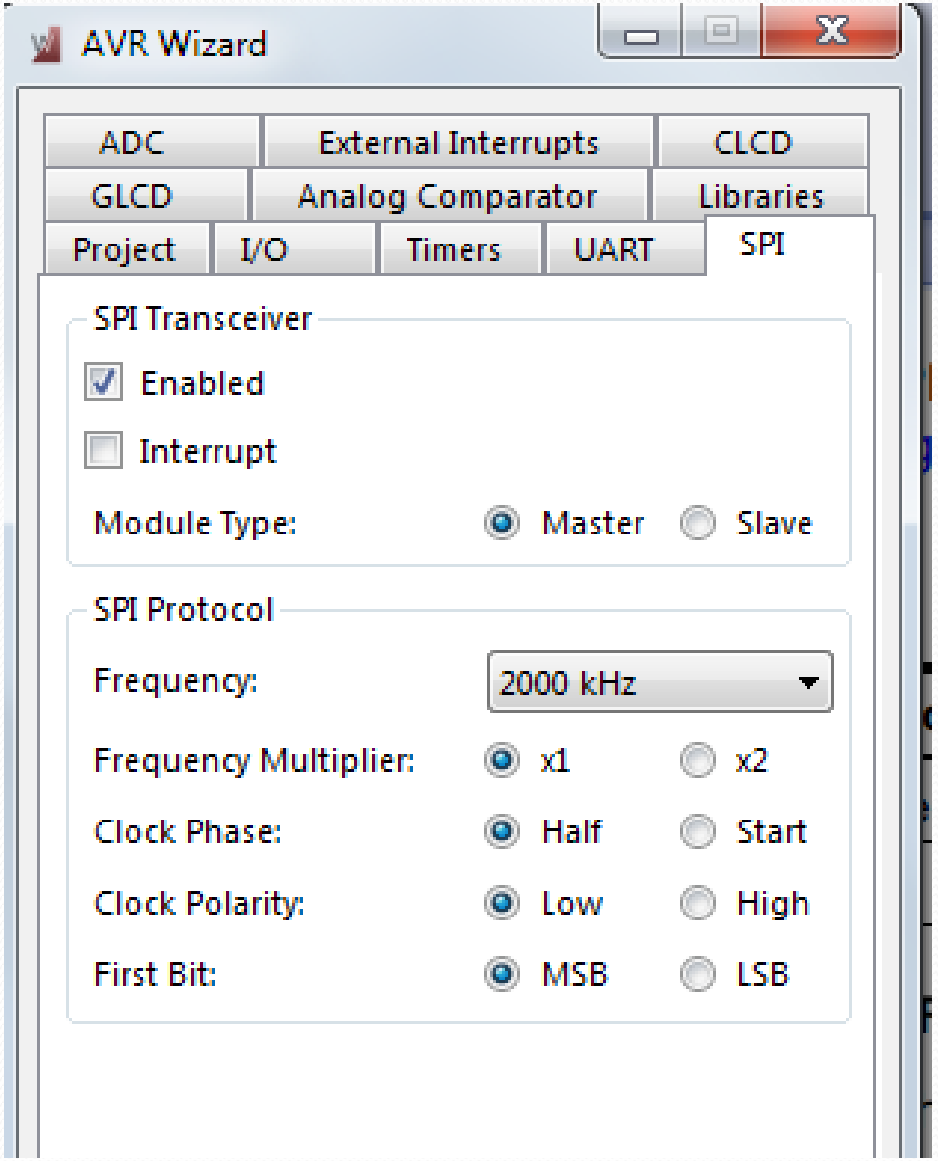
# Data Transfer in SPI



# SPI in Atmega 16



# SPI Coding


AVR Wizard
ADC
External Interrupts
CLCD
GLCD
Analog Comparator
Libraries
Project
I/O
Timers
UART
SPI
SPI Transceiver
Enabled
Interrupt
Module Type:
Master
Slave
SPI Protocol
Frequency:
2000 kHz
Frequency Multiplier:
x1
x2
Clock Phase:
Half
Start
Clock Polarity:
Low
High
First Bit:
MSB
LSB

# Clock Polarity (CPOL)

- The value of CPOL bit decides the value of Clock (SCK) in its idle state.
- When CPOL = 1 , SCK is 5V in idle state.
- When CPOl = 0 , SCK is 0V in idle state.

| CPOL | Leading (First) Edge | Trailing (Last) Edge |
|---|---|---|
| 0 (low) | Rising | Falling |
| 1 (high) | Falling | Rising |

# Clock Phase ( CPHA)

- The settings of the Clock Phase bit (CPHA) determine if data is sampled on the leading (first) or trailing (last) edge of SCK

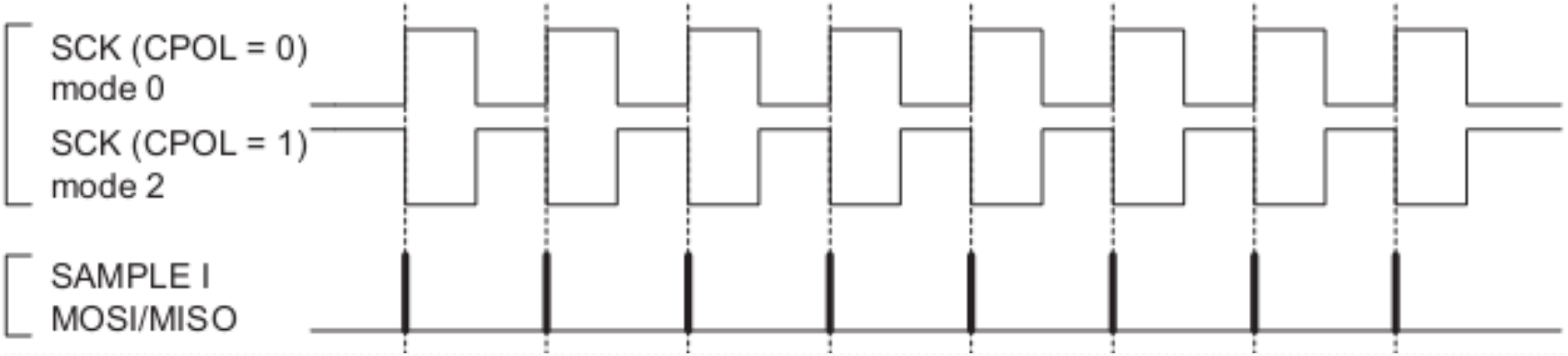
| CPHA | Sample |
|---|---|
| 0 (half) | Leading Edge |
| 1 (start) | Trailing Edge |

# Modes of SPI

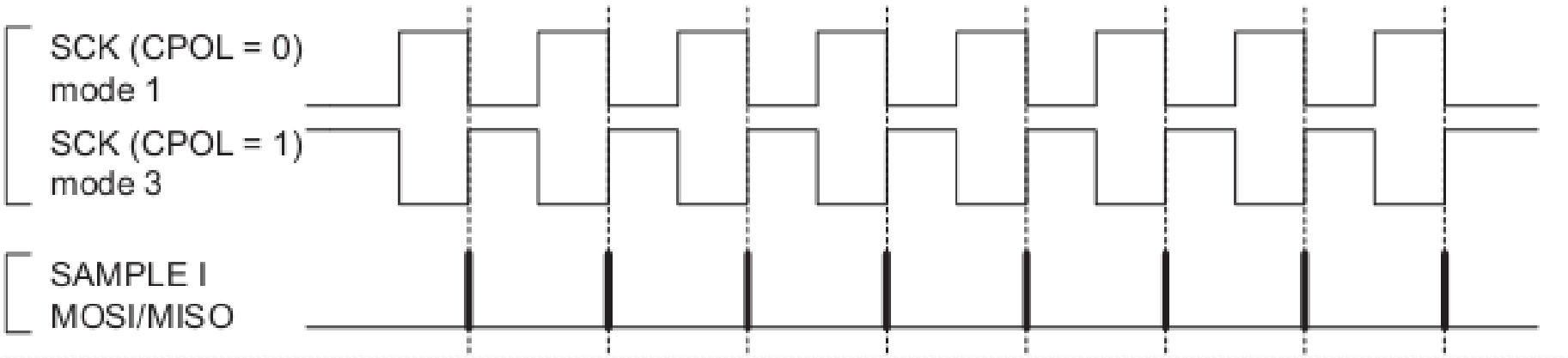
- Two  - Two possible values of CPOL and CPHA bits gives rise to 4 modes of SPI

| Mode | Clock Polarity (CPOL) | Clock Phase (CPHA) |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 2 | 1 | 0 |
| 3 | 1 | 1 |

# SPI Transfer Format with CPHA = 0



# SPI Transfer Format with CPHA = 1



# Simple SPI Code

```
char  data = SPITransmit('a');
```

In case of master, the data is written on the register and send to the slave.

In case of slave the data is written on the register and it waits for the master to transmit the data, when it also transmits its own data.

# Master Code

```
DDRB = 0b10110000;   // configure SPI Master Pins


ISR(INT0_vect)
{
    //  External Interrupt 0

   data = SPITransmit(0x01);  // when switch is
   pushed
        // send data
}
```

# Slave Code

```
DDRB = 0b10000000;   // configure SPI Slave Pins


ISR(SPI_STC_vect)
{
   //  SPI Transceiver Interrupt
   data = SPDR ;     // read the data
  if(data == 0x01){
   PORTA = ~PORTA;  // if data is correct toggle Led
   }
}
```

Thank You
Question??