PD215 Mechatronics

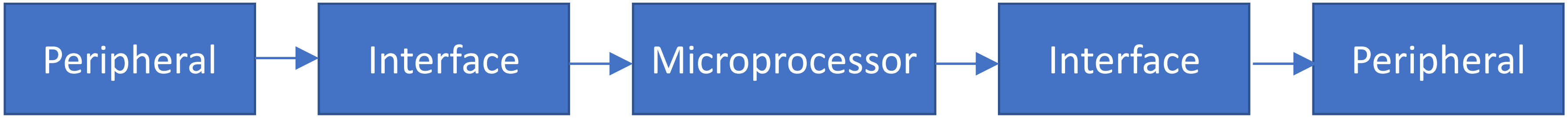
Week 3/4

Interfacing Hardware and Communication Systems

Interfacing with the physical world

A compute device (microprocessor) in mechatronic system needs to ***accept input information*** and ***respond with output signals*** to perform its action.

The term ***'Peripheral'*** is used for a device such as keyboard, sensor, actuator etc. which connects to the microprocessor.



Often interfacing hardware is required to address compatibility between peripheral signals and microprocessor.

Input/output addressing

Memory-mapped Input/output : The input/output interfaces are configured as fixed address for the microprocessor just like memory location and thus input or output operation can be considered as reading or writing to specific memory location.

Also, often it is possible to configure ports as input or output depending upon ***control register*** value. Such ports are called ***General Purpose I/O (GPIO)***

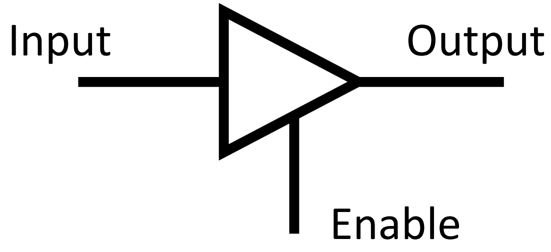
→***Find out how many GPIO does microcontroller in Arduino Uno, Mega and Pro-Mini.***

Functions of interfacing hardware

- Electrical buffering/isolation
- Timing control
- Changing number of lines
- Serial-to-parallel, vice versa and data transfer
- Analog to digital conversion and vice versa
- Code conversion

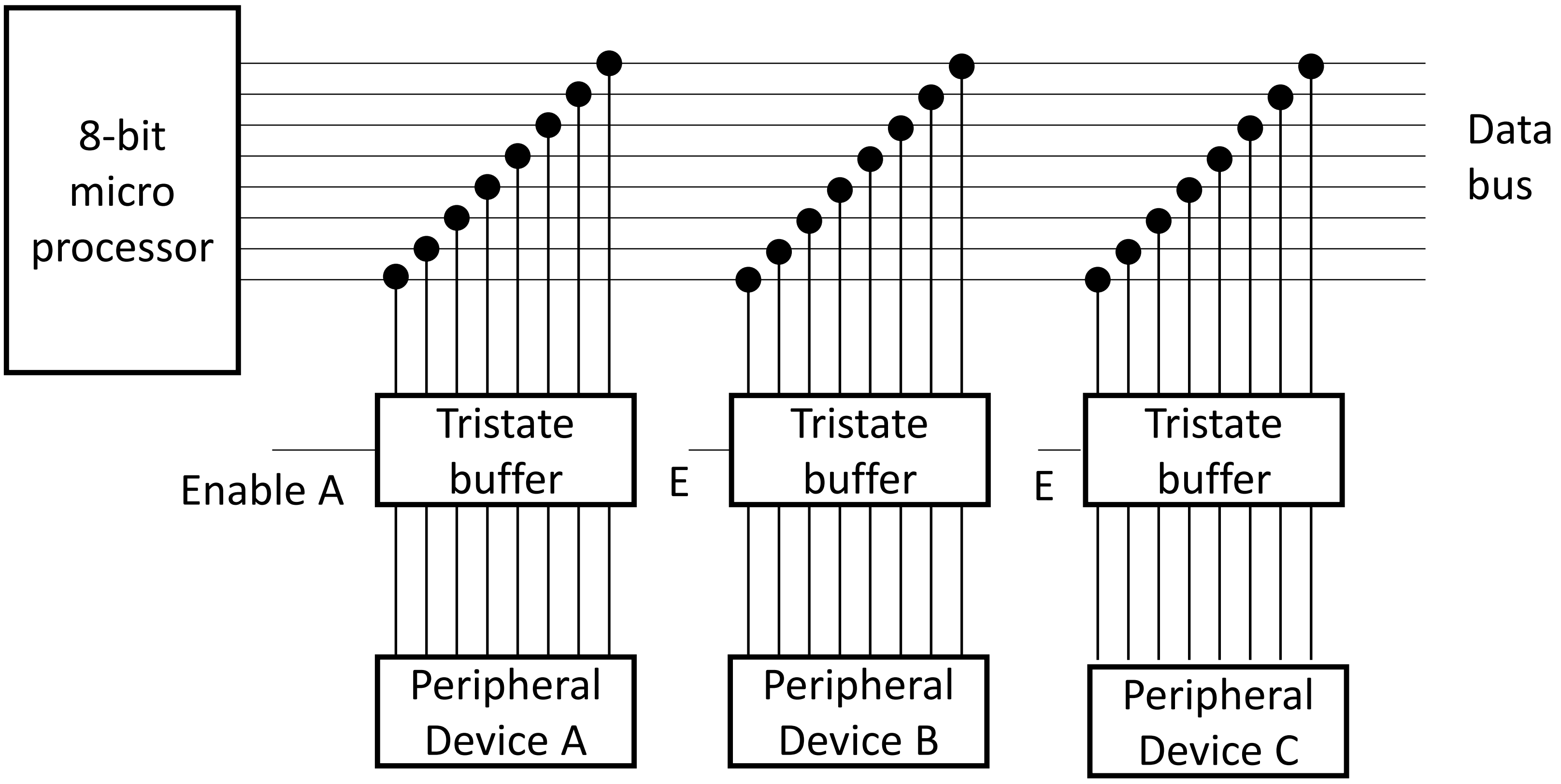
Buffers

- Electrical buffers connect two parts of the system and prevent unwanted interference between the two parts
- On input side buffers isolate input port from microprocessor data bus until microprocessor request it



Enable	Input	Output
0	0	High Impedance
0	1	High Impedance
1	0	0
1	1	1

Tristate buffer used to connect multiple peripheral devices with same data bus



Handshaking

- Timing control is needed when data transfer rates of the peripheral and micro processor are different.
- Special lines (handshake lines) are used for controlling the data transfer.

Use case of peripheral sending data:

- DATA READY from peripheral to CPU
- CPU determines DATA READY signal is active and read input data
- CPU sends INPUT ACKNOWLEDGE signal

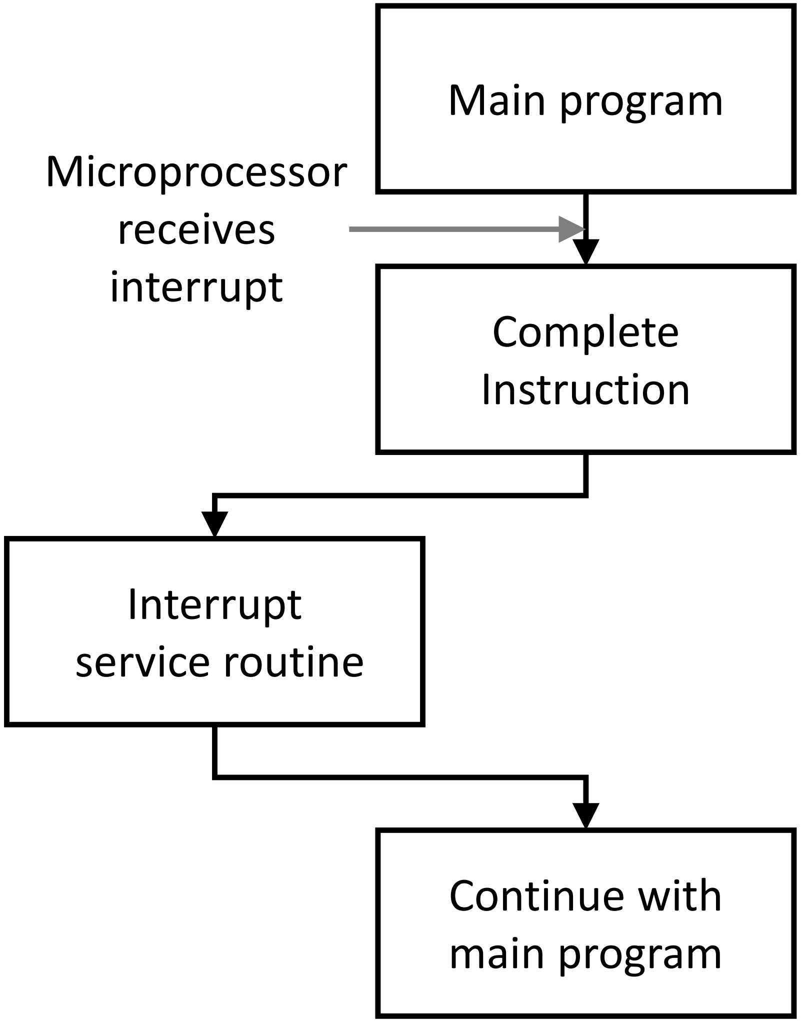
Polling and Interrupts

- Process of repeatedly checking peripheral device to see if it is ready to send data or accept new data is a called ***polling***
- In ***Interrupt control*** peripheral device activates a separate interrupt request line.
- The interrupt routine must not lead to loss of data hence upon reception following steps are executed:
  1. CPU waits till end of instruction currently executing
  2. All current CPU registers are pushed on to the stack and bit is set to stop any further interrupts

Cont.

Interrupts control

3. CPU determines address of interrupt service routine (ISR)
4. The CPU branches to ISR
5. After completion of ISR, the CPU registers are returned from the stack and main program continues from the point it left off.


Main program
Microprocessor receives interrupt
Complete Instruction
Interrupt service routine
Continue with main program

Unlike a subroutine call interrupt can be called from any point in the program

Interrupts control

Microcontroller can have multiple types of interrupts

- Some Interrupt signals can be '***masked***' , i.e. they do not lead to ISR being executed is particular bit is set in registers. While other interrupts may be ***non-maskable***
- ***Reset interrupts*** stops all activity and starting address of main program is loaded and startup sequence is executed.
- Computer operating properly (COP) ***watchdog timer*** can reset the system if CPU is not executing sections of the code within allotted time.

Serial Interface

With **Parallel data** transmission of data, one line is used for each bit.

In **Serial data** transmission single line is used to transmit data in sequential bits.

Asynchronous transmission:

Receiver and transmitter use there own clock

Each transmitted data has its own start bit and stop bit

Synchronous transmission:

Transmitter and receiver have common clock

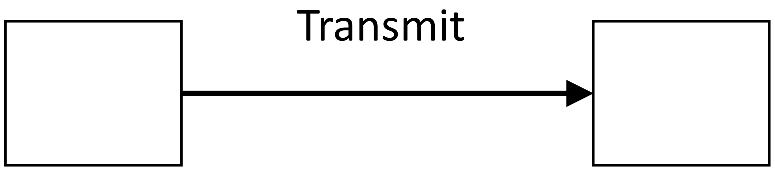
Many Microcontrollers have built in **UART (universal asynchronous receiver/transmitter)**

Serial Peripheral Interface (SPI) is a synchronous interface

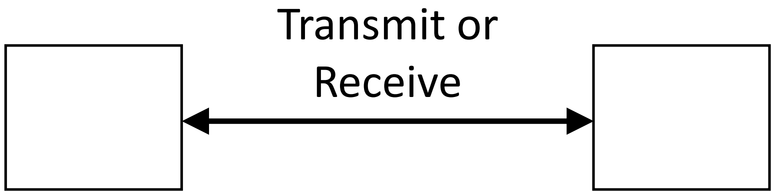
Serial Communication Interface (SCI) for an asynchronous interface

Serial Data communication modes

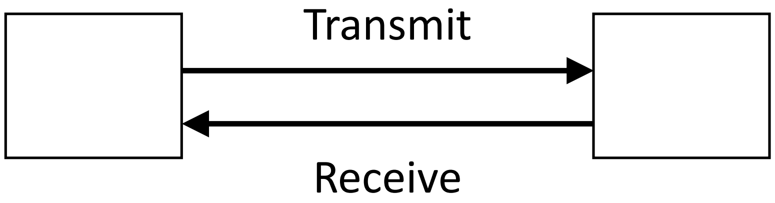
Simplex mode



Half duplex mode

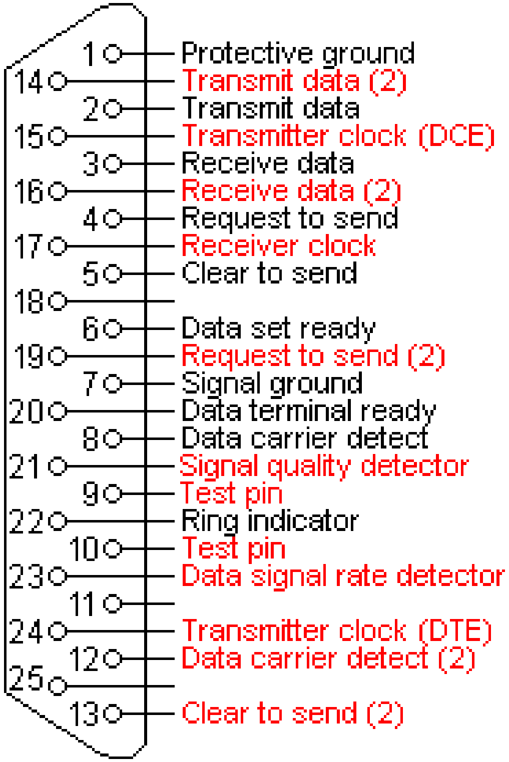


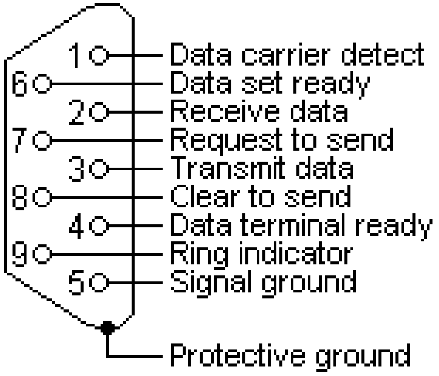
Full duplex mode

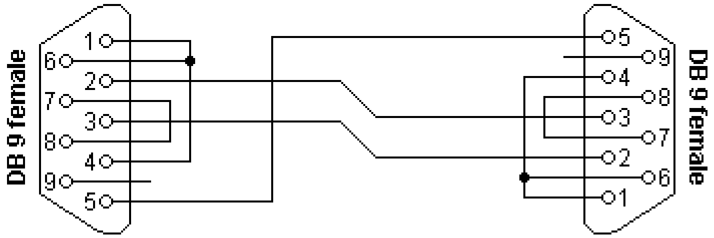


Serial interfaces

RS-232 is a popular serial interface standard developed in 1960s







Minimum configuration

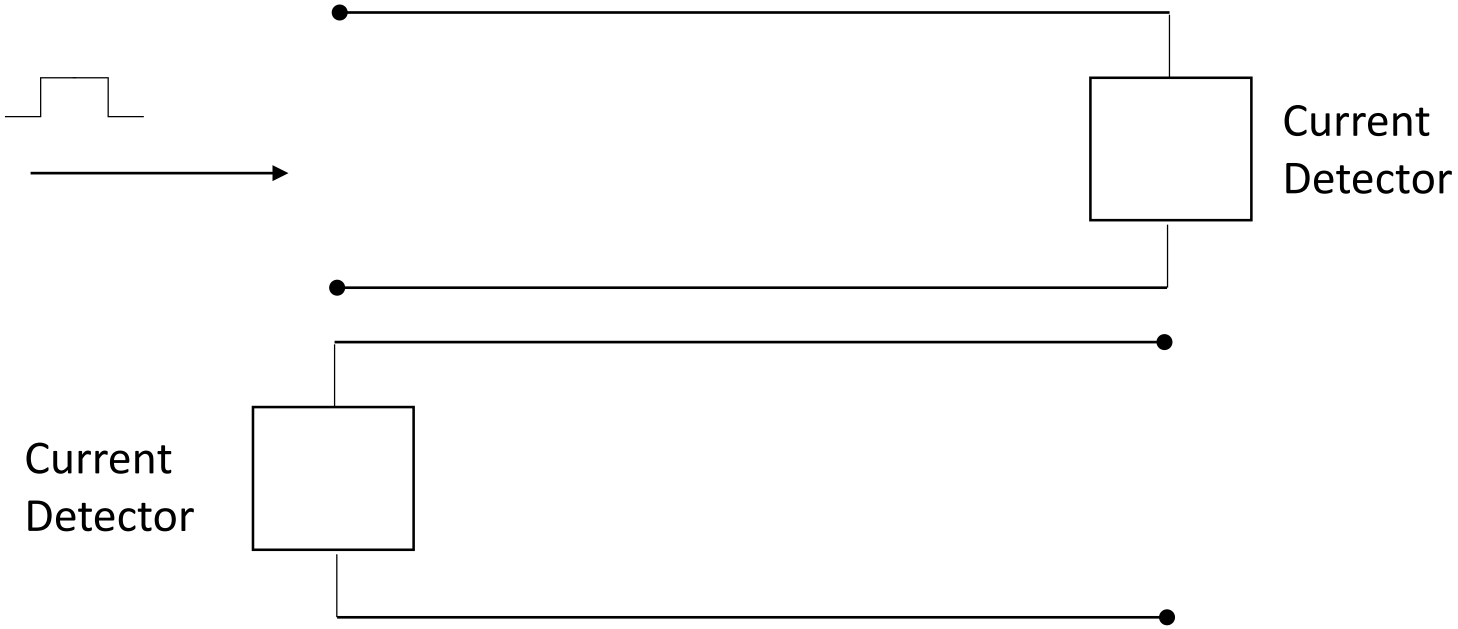
Limited to 20kbits/s and 15m distance

RS-422, RS-485 are similar to RS-232 but improve upon the distance and data rate

Serial Interfaces

- 20mA current loop
  - Current signal is used rather than voltage

  20mA = logic level 1   0mA = logic level 0

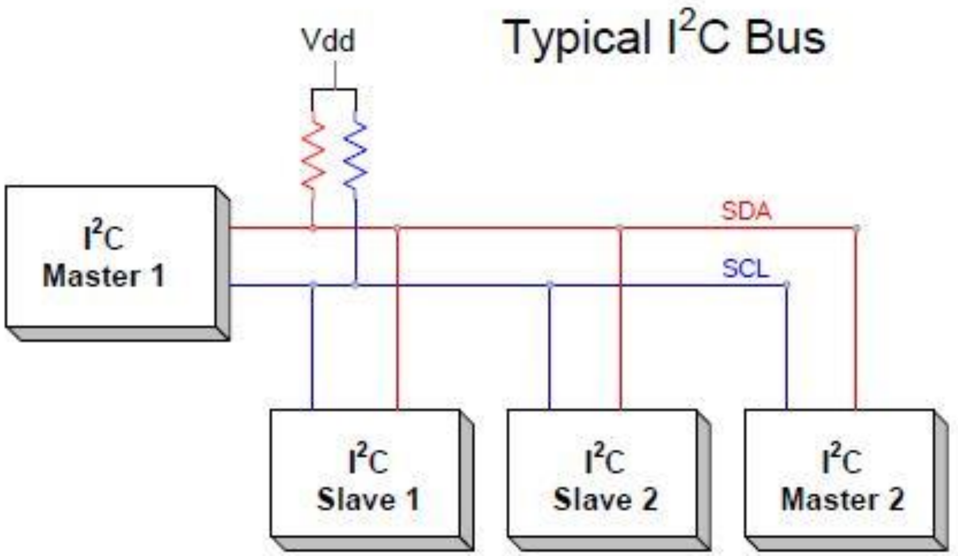


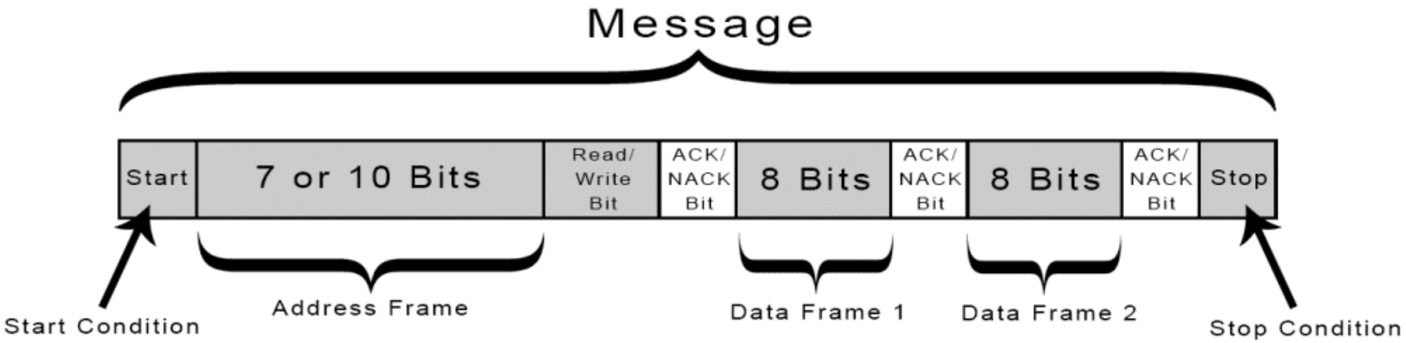
Can reach up to few kilometer, used in industrial setting

Serial Interfaces

- I2C

  Developed by Philips and uses just two wires





Serial Interfaces

- CAN
  - Controller Area Network
  - Develop by Bosch for automotive management
- USB
  - Universal Serial Bus
  - Developed by industry consortia
  - Uses star topology – thus one device connects to the computer and other connect via that device
- Firewire
  - Develop by Apple with specification given by IEEE 1394 standard

Networked Communications Systems

Large complex systems e.g. a whole car manufacturing plant, can be interconnected to be part of a single systems

In **Centralised control** a single computer will control entire system.

This central computer then becomes ***single point of failure***.

Hierarchical/distributed Systems

- In **Hierarchical system** there is hierarchy of computers where work is divided into computers according to functions involved.
- In **Distributed systems**, the each computer carries of similar task to all the other computers and in event of failure of one computers others in network keep it systems operations.

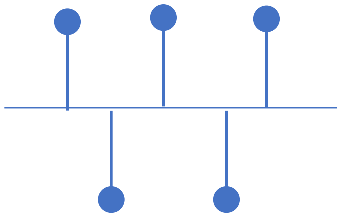
Modern complex system are mixture of hierarchical and distributed computers where compute devices at each level handle different task.

  - Level 1 Measurement and Actuators
  - Level 2 Direct Digital and sequence control
  - Level 3 Supervisory control
  - Level 4 Management control and design

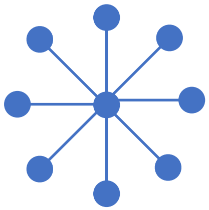
Network topologies

Network allows connection between two or more microprocessors to interchange data.

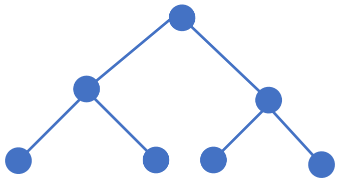
Topology is refers to logical linking of ***nodes***



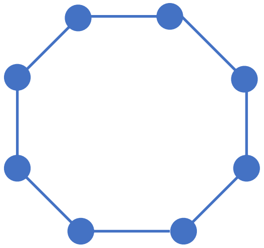
Databus



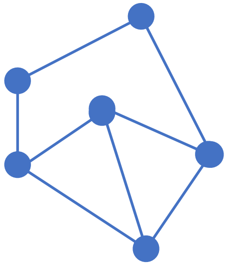
Star



Hierarchy or tree



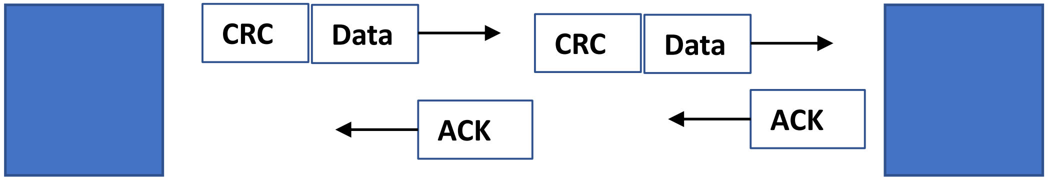
Ring



Mesh

Protocols

- Information other data need to exchanged to ensure data transmission is happens reliably
- ***Protocol data*** refers to bits of information ofther than data being transmitted that are part of the protocol
- Protocol specifies
  - Syntax
  - Semantics
  - Timing



Open Systems Interconnection (OSI) Model

1. Physical Layer
2. Data Link Layer
3. Network Layer
4. Transport Layer
5. Session Layer
6. Presentation Layer
7. Application Layer