Digital Communications

ESD_Cap6--/++ (from *Extra* folder)

Overview

- Main Issues
- Communication Protocols: Basic Concepts
- Parallel Communications
- Serial Communications

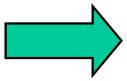
Main Issues

Main Issues

- Information $\Rightarrow$ Simbols $\Rightarrow$ Signals
  - Coding
    - Cost, Robustness, Efficiency
- Binary Simbols
  - *bit*
- Electrical (Optical) Signals
- Trasmission Media
  - Cables (Optical Fibers), ether
  - Equipments for generation, amplification, conversion, etc.

Main Issues

- Information to be sent
  - bit ⇒ character ⇒ message
- Synchronization and symbols identification
  - Agreement between sender and receiver on which temporal windows are available to represent symbols
    - They could be used auxiliary signals to indicate istants meaningful for synchronization
      - Parallel Communications (short distances)
        - » Normally they are used auxiliary signals on dedicated cables
      - Serial Communications (long distances)
        - » Normally it is used a single signal both for synchronization and data transfer



Communication Protocols

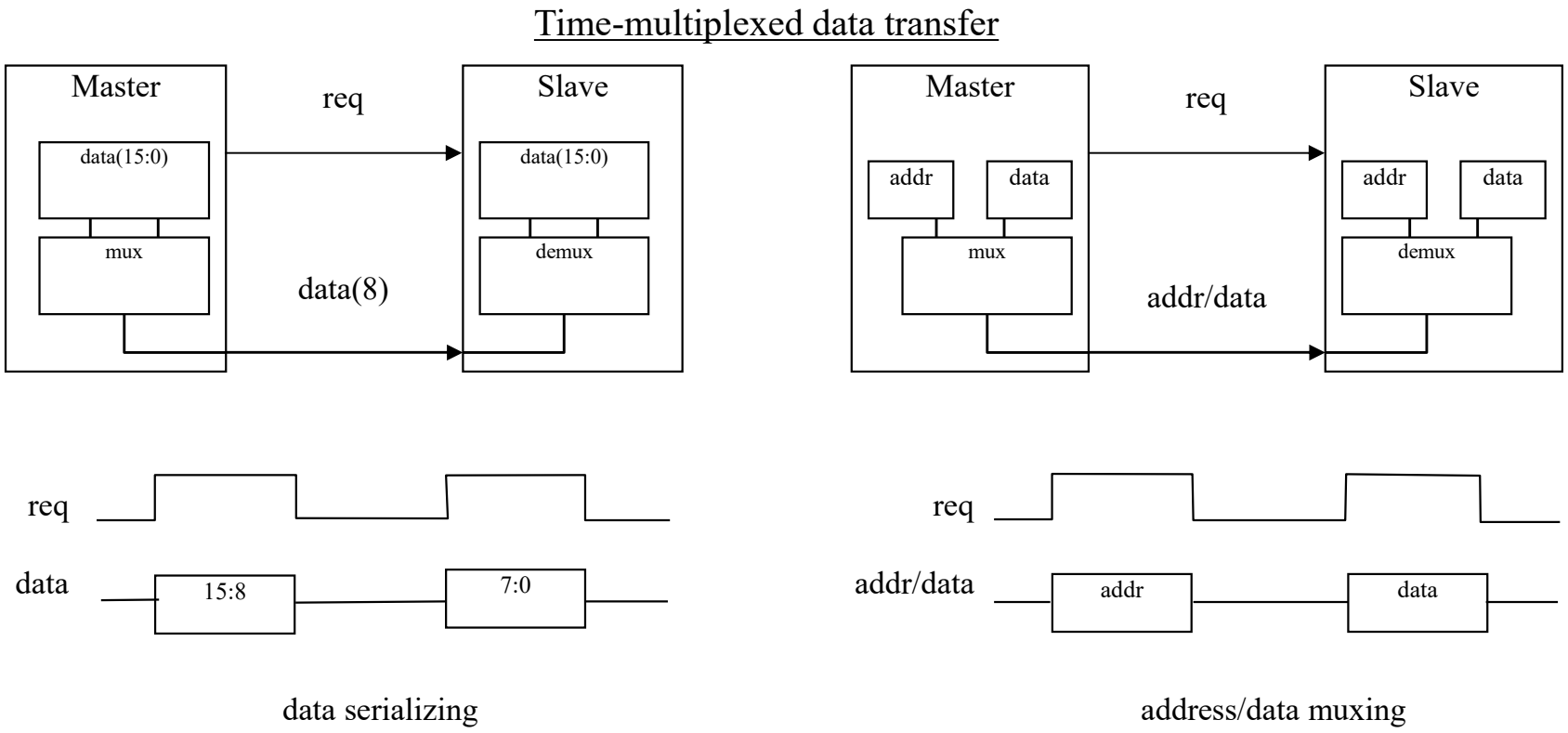
Comunication Protocols: Basic Concepts

Comunication Protocols: Basic Concepts

- Roles
  - Master/Slave
- Direction
  - Sender/Receiver
- Address
  - Data with a particular meaning
    - It specifies a memory location, a peripheral device or a peripheral device register

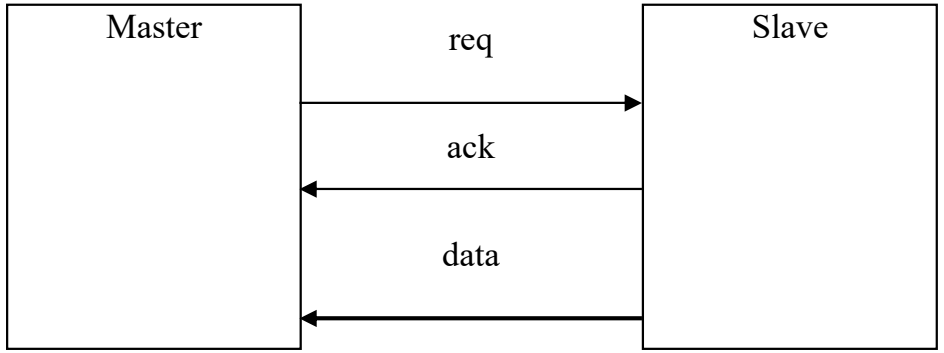
Comunication Protocols: Basic Concepts

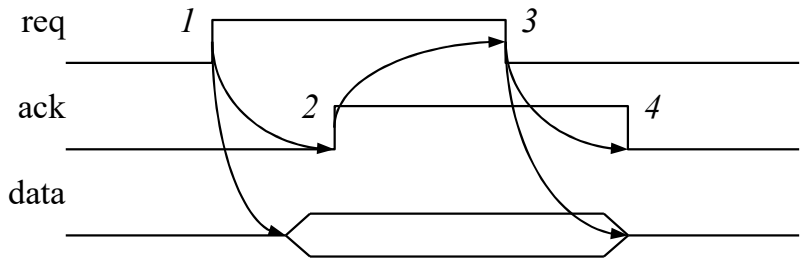
- Time multiplexing
  - Sharing of a set of wires to transfer more data
    - Less complexity but higher transmission time



Comunication Protocols: Basic Concepts

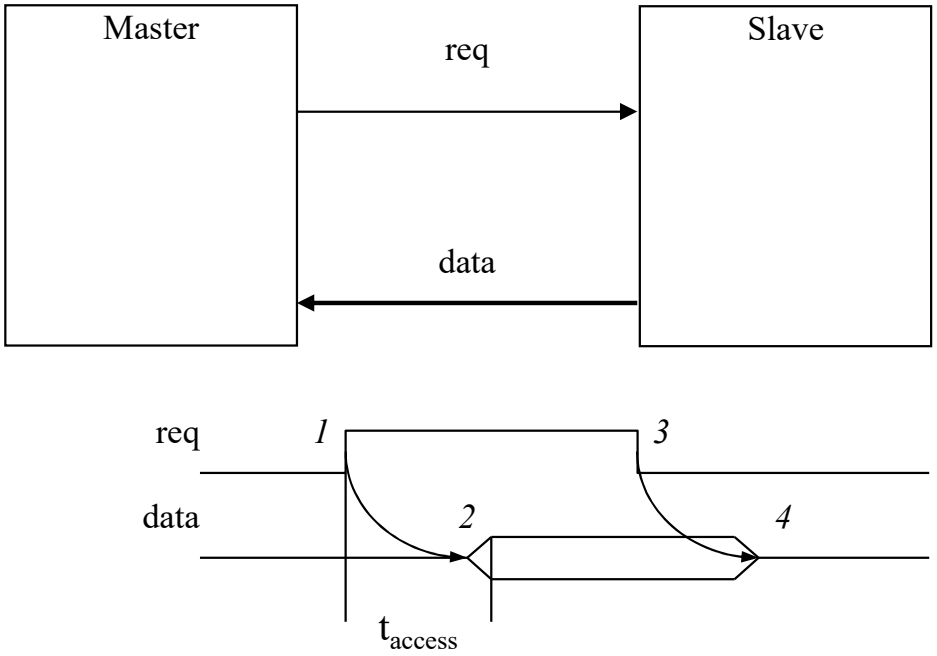
- Main synchronization techniques
  - *Handshake*
    - The most famous one
      - It allows to adapt sender and receiver velocity to avoid loosing data





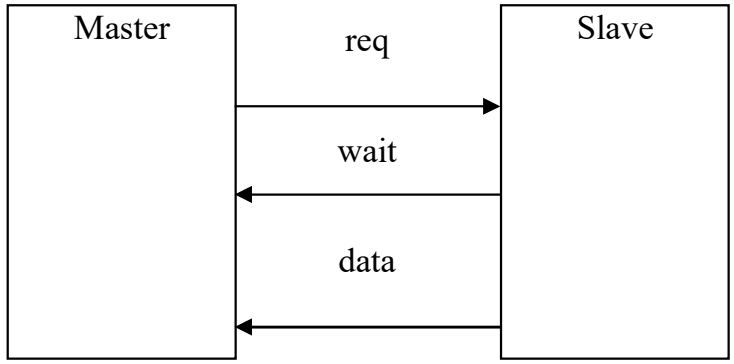
Comunication Protocols: Basic Concepts

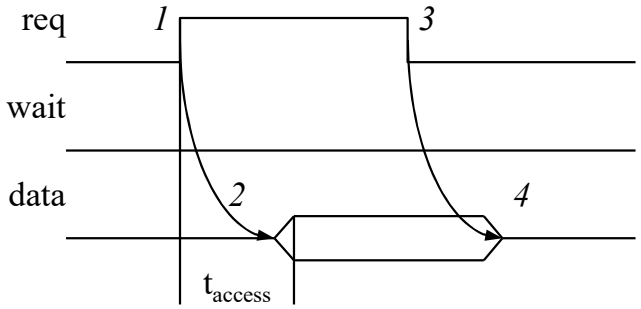
- Main synchronization techniques
  - *Strobe*
    - The simpler one

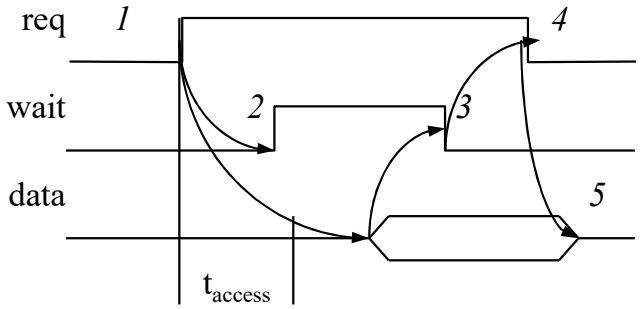


Comunication Protocols: Basic Concepts

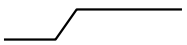
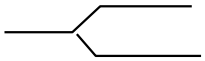
- Main synchronization techniques
  - *Strobe/Handshake*
    - Hybrid approach





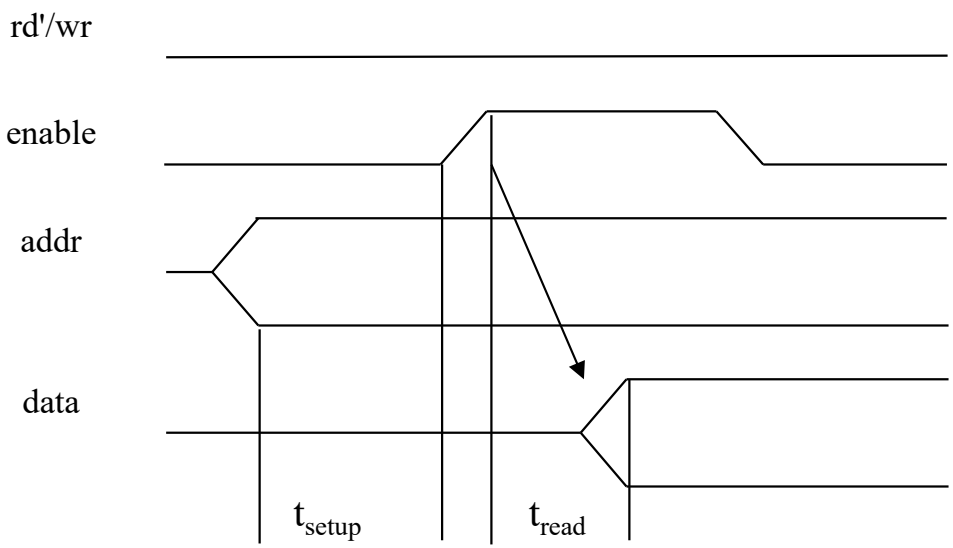


Comunication Protocols: Basic Concepts

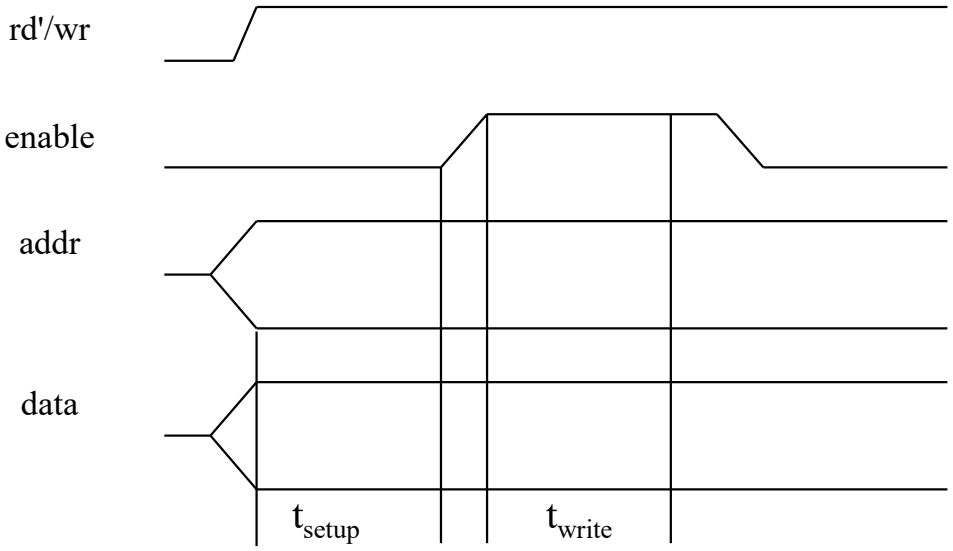
- Timing Diagrams
  - Formalism used to describe a communication protocol
    - Time over x axis
  - Control signals: low or high 
    - Low/High Active
    - Assert/Deassert
  - Data signals: valid or not valid 

Comunication Protocols: Basic Concepts

- Timing Diagrams
  - Example of reading/writing bus cycles



read protocol

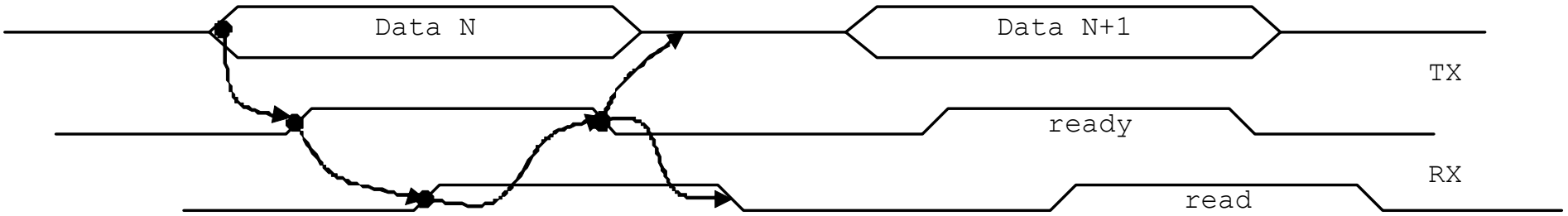


write protocol

Parallel Communications

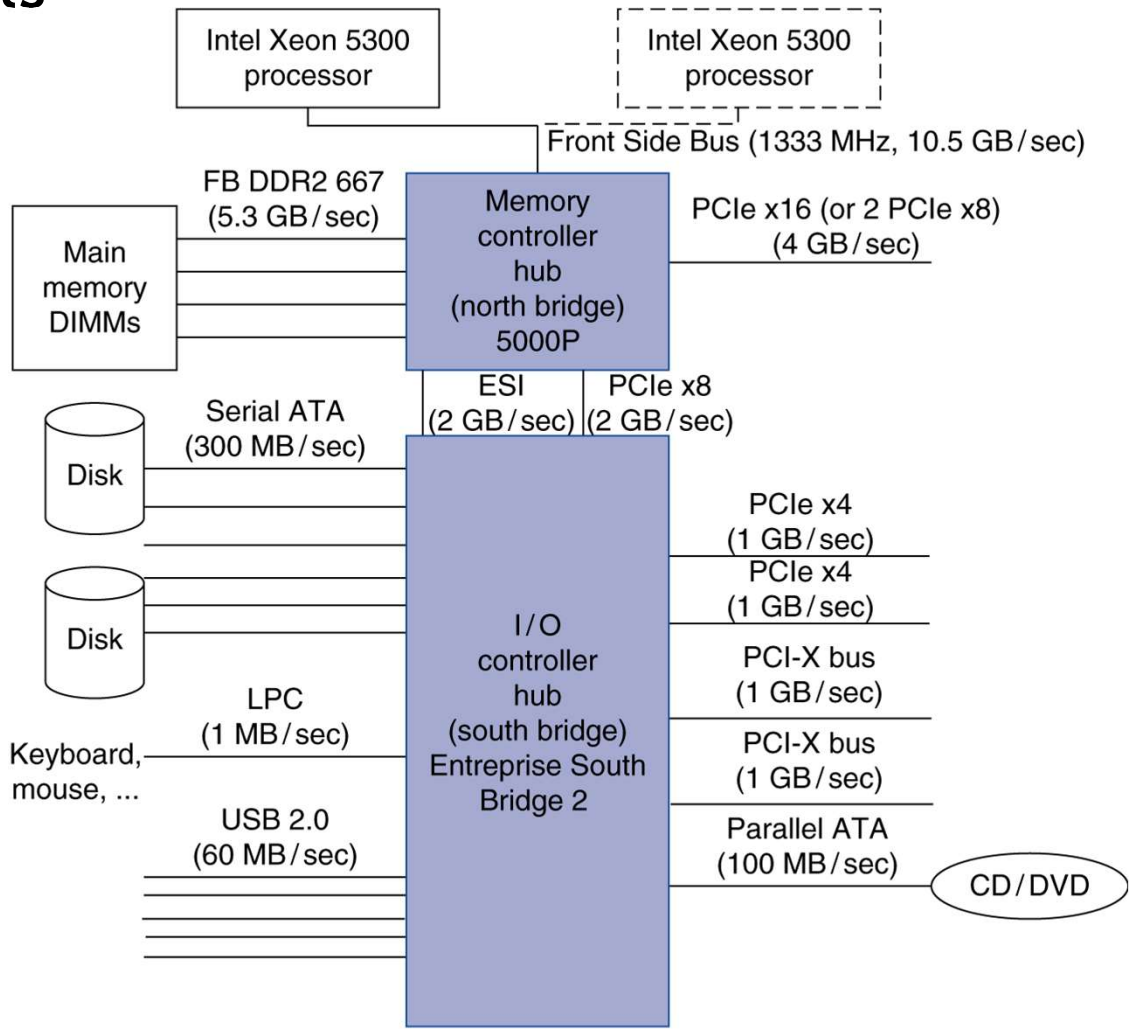
Parallel Communications

- Used to transmit concurrently whole groups of bit (8, 16, 32, etc.) in short distance connections (cm/m)
  - Synchronization is normally performed by means of auxiliary signals and *handshaking*



Parallel Communications

- Standard parallel protocols
  - Hystorical
    - Standard IEEE-488
    - Centronics
  - System/Pheriferal Bus
    - *FrontSide*
    - ISA/EISA
    - SCSI
    - ATA/IDE
    - PCI/PCI X/PCI Express
    - AMBA (AXI, AHB)



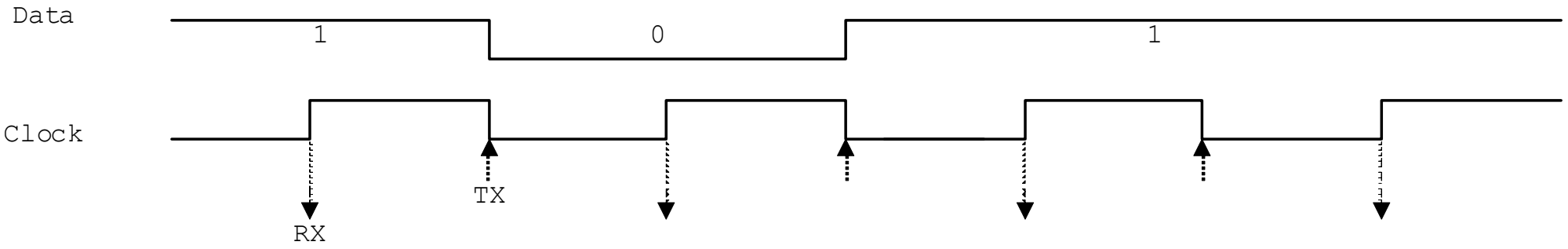
Serial Communications

Serial Communications

- Used to transmit a bit at a time by means of a single signal in short and long distance connections (m and over)
  - Syncronization techniques
    - They allow the receiver to identify the “data valid” window
      - Isochronous Transmission
        - » One ausiliary signal (clock) common to sender and receiver
      - Asynchronous Transmission
        - » Separated sender and receiver clock generators working at the same frequency and synchronized every character transmission
      - Synchronous Transmission
        - » Separated sender and receiver clock generators with “Phase Lock Loop”

Serial Communications

- Isochronous Transmission
  - Bit synchronization is based on a single clock signal common to sender and receiver
    - e.g. (rising/falling can be exchanged)
      - Clock falling edge
        - » The sender writes the next bit (commutation)
      - Clock rising edge
        - » The receiver reads the bit value (sampling)

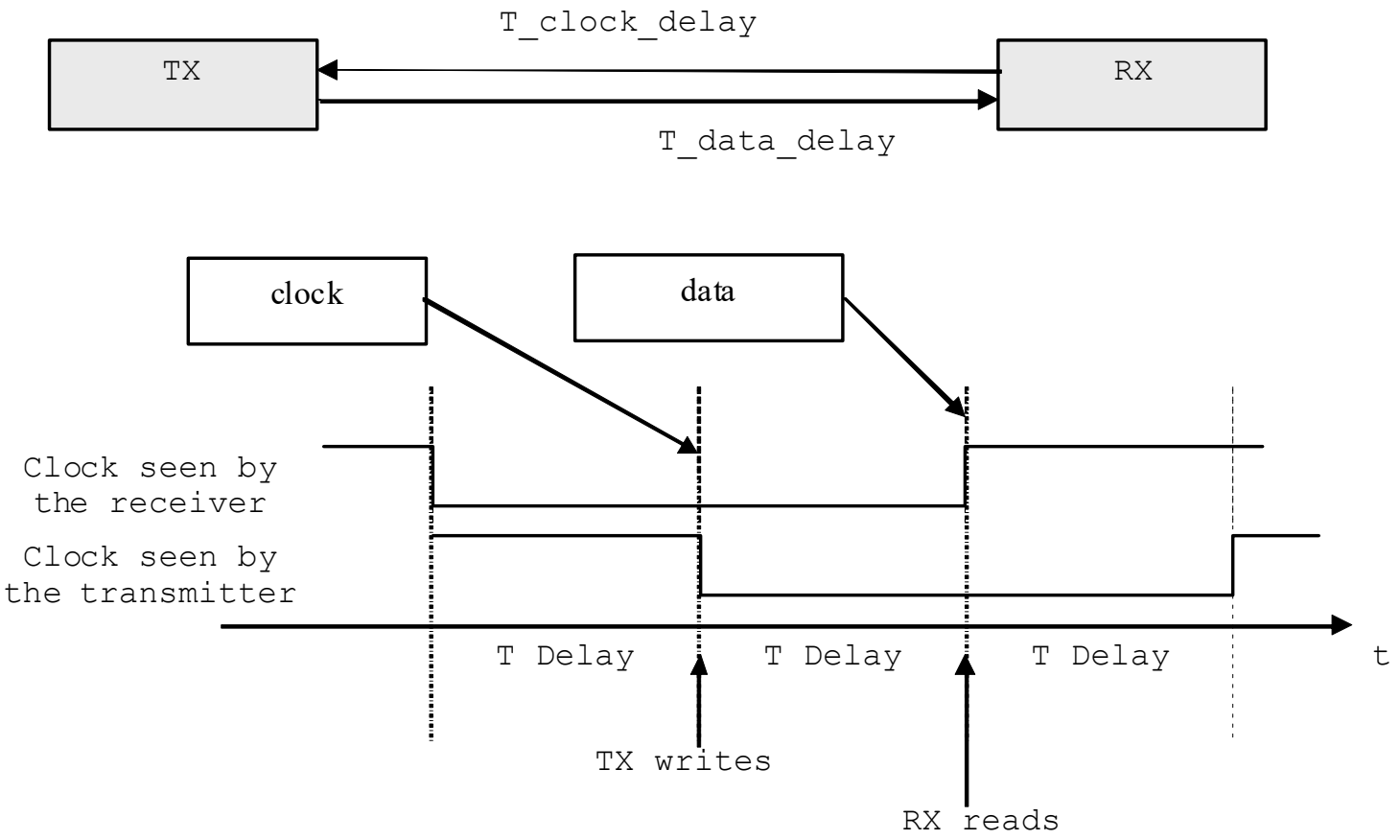


Serial Communications

- Isochronous Transmission
  - The clock is a signal on a dedicated wire
    - If generated by the sender
      - Clock and data are propagated in the same direction
        - » Correct phase alignment
    - If generated by the receiver
      - Clock and data are propagated in different directions
        - » The delay is `2*Propagation_Time`
    - The distance between rising and falling clock edges should be greater than `2*Propagation_Time`
      - It is a limit to the transmission rate!
  - Due to the limits imposed by propagation time and the possible disturbances on the separated clock line, this kind of transmission is used only for very short connections

Serial Communications

- Isochronous Transmission



Serial Communications

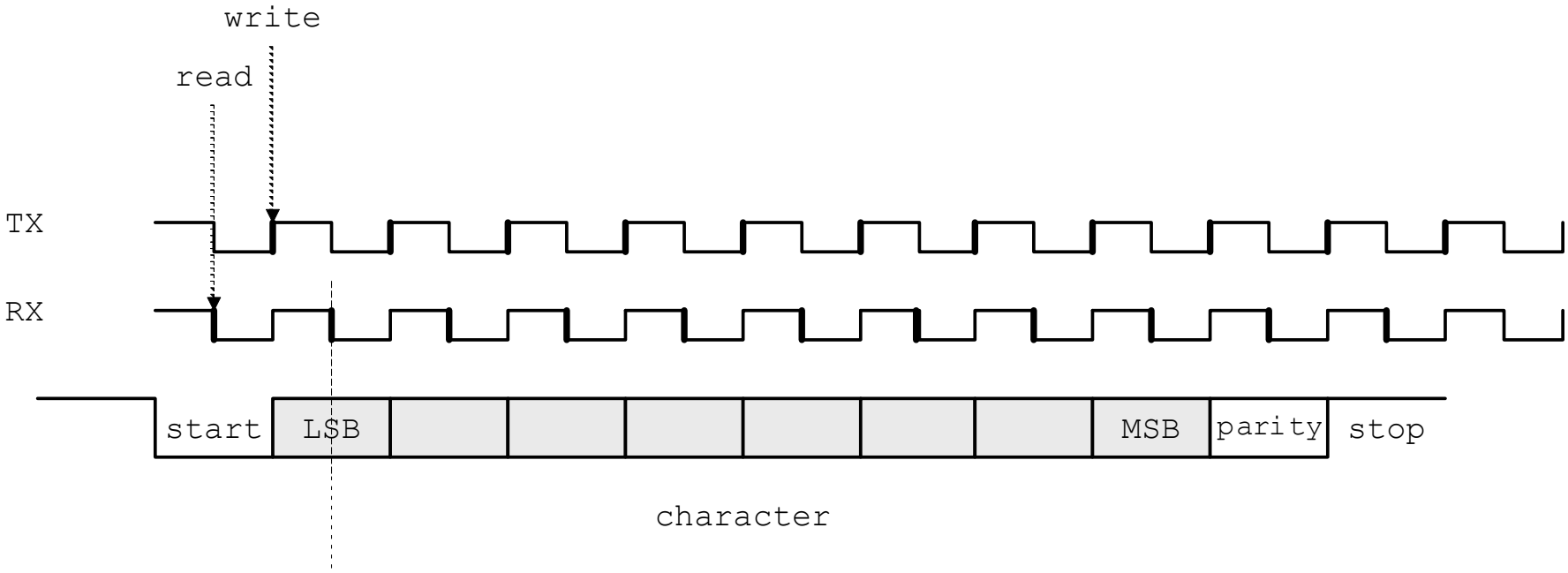
- Asinchronous Transmission
  - TX and RX have own clocks generators with the same frequency and stable enough to allow the correct transmission of at least 1 character
  - Character-level synchronization
    - A `start bit` is sent before each character
    - One or more `stop bit` are sent at the end of the character
    - The recevier synchronizes itself on the `start bit` edge and keeps synchronism for the whole character
      - The transmission rate between characters can change
  - This technique is very used for short/long point-to-poit connections

Serial Communications

- Asinchronous Transmission
  - Once clock generators are synchronized they looks like the ideal isochronous clock
    - Transmitter clock generator provides edge for commutation while receiver clock generator provides edge for sampling
      - For a correct acquisition they have to be inside the bit window validity
    - Similarity between clock frequencies shall be enough to assure that for each bit of the character the phase shift is contained in the interval of +/- half of the bit duration

Serial Communications

- Asinchronous Transmission (e.g. 8 bit with parity)

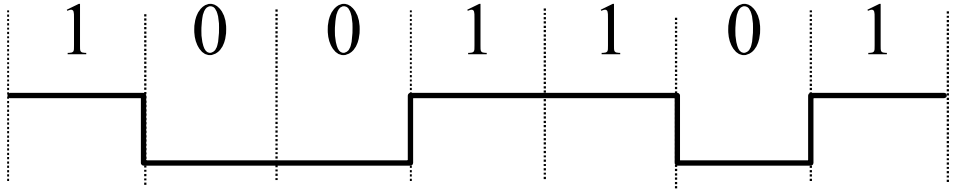


Serial Communications

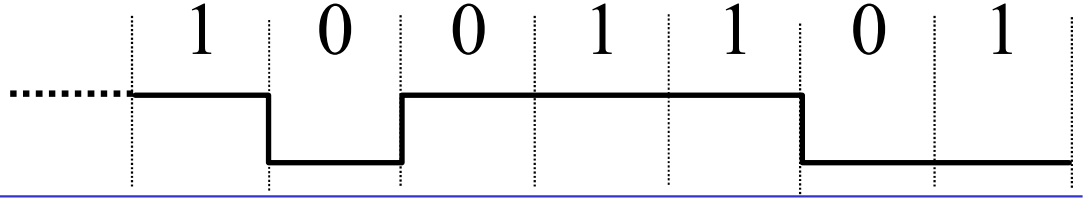
- Sinchronous Transmission
  - TX has a clock generator with stable frequency
  - RX has a clock generator with *Phase Lock Loop* (PLL) capability
    - Bit-level synchronization is obtained after some commutations that allow the phase lock
    - Character-level synchronization is obtained by means of some synchronism characters (3..5) sent before each **message**
      - The message bits are sent without delays also avoiding too long sequences without commutations
        - » To avoid loosing the phase lock could be also introduced some extra bits or could be adopted special modulation techniques
  - Very used for high-speed transmissions

Serial Communications

- Signals, coding and modulation
  - Two-value signals: used with cables for short-distance transmissions
    - NRZ (*Not Return to Zero*): each transition indicates a logic value change
      - The most used

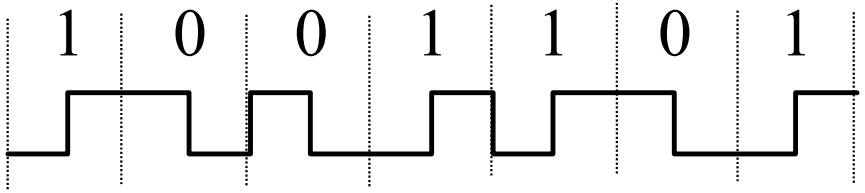


    - NRZI: the presence or the absence of a transition indicates a proper logic value
      - Less used but it requires less bandwidth

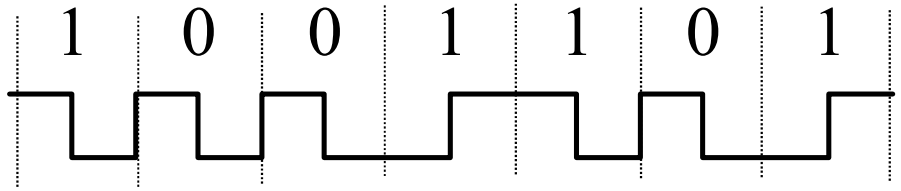


Serial Communications

- Signals, coding and modulation
  - Two-value signals
    - Bi-Phase Manchester: a specific edge in the middle indicates a proper logic value(e.g. (rising=1; falling=0)
      - Used in synchronous transmissions to facilitate the phase lock but they use more bandwidth



    - Differential Manchester: a logic 1 (0) is represented by an edge at the beginning while a 0 (1) is represented by an edge in the middle



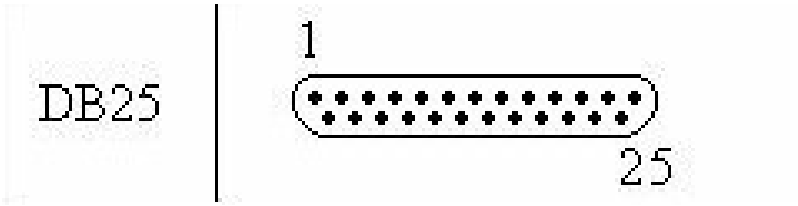
Serial Communications

- Errors Management
  - Character-level
    - Recognition: e.g. `parity bit`
    - Correction: e.g. `Hamming code`
  - Message-level
    - Recognition: extra characters function of the transmitted ones
      - Cyclic Redundancy Check
        - » Evaluate on a given numebr of bits
      - Checksum
        - » Sum with modulo
      - Longitudinal parity
    - Correction: re-transmission is normally adopted

Serial Communications

- Standard protocols
  - EIA RS-232-C (1969)
    - Proposed to connect DTE devices with DCE devices
      - DTE: *Data Terminal Equipment* (PC, printers, ecc.)
      - DCE = *Data Communications Equipment* (modem)

    - Electrical signals: 0 = ON $\Rightarrow$ V > +3 V; 1 = OFF $\Rightarrow$ V < -3 V
    - Distance: < 15 m
    - Velocity: < 20.000 bit/sec
    - Sequence: from LSB to MSB with possible parity
    - DTE connector: Cannon male with 25 contacts



Serial Communications

- Standard protocols
  - EIA RS-232-C: main signals

1	GND	ground
2	TXD	transmitted data
3	RXD	received data
4	RTS	request to send
5	CTS	consensum to send
6	DSR	data set ready
...		
15	TXC	TX clock
17	TXD	RX clock
20	DTR	data terminal ready

  - The standard is often used in a reduced version (9 pin connector) to directly connect DTE devices (i.e. without intermediate modems) by using a cable callled “**null modem**” where properly connections are crossed (2-3, 15-17, etc.)

Serial Communications

- Standard protocols
  - RS-485
    - It is widespread a standard related only to electrical issues
      - It allows connections by more TX (tristate) and RX on the same wire
    - It is very used to realize bus topologies for very noisy industrial plants

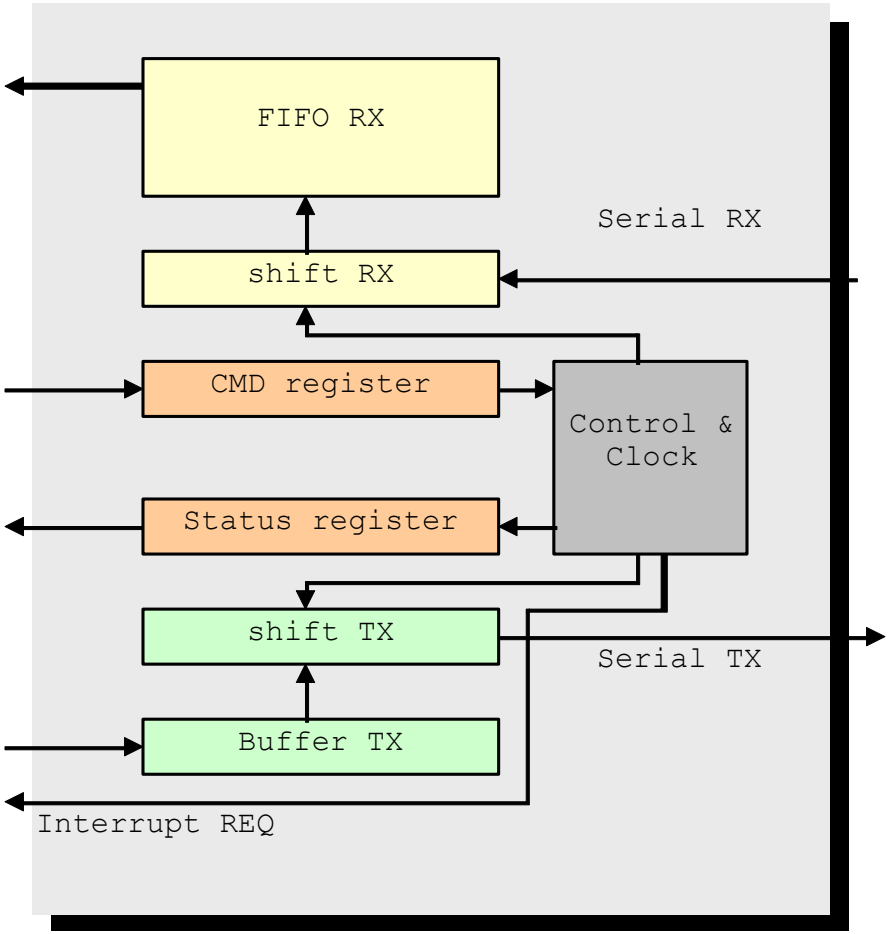
Serial Communications

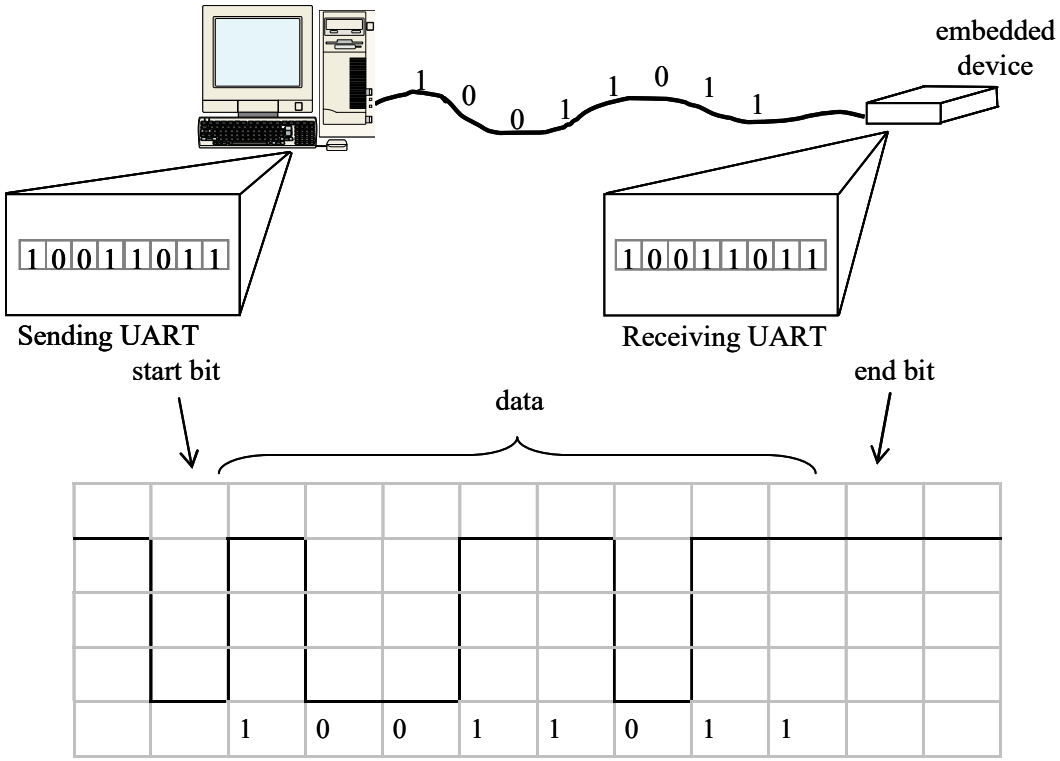
- Standard protocols
  - U(S)ART
    - Serial transmissions, syncronous and asynchronous, are so popular that it has been possible to produce a component called USART *(Universal Synchronous Asynchronous Receiver Transmitter*) that allows easily interfacing with microprocessors
      - *Standard Single Purpose Processor*
        - » Oftne they are directly integrated on *microcontroller*/SOC
    - They are IC that provide the basic functionality needed for ISO/OSI low-levels in serial transmissions
      - UART NS8250 e NS16550
        - » They are called UART when able to manage only asynchronous transmissions

Serial Communications

- Standard protocols
  - U(S)ART
    - Main functionality
      - Serial/parallel conversion
      - Parity bit management
      - Bit/character synchronization management
      - Buffering
      - Interrupt requests generation
    - Typical registers
      - Buffer and transmission shift
      - Buffer and reception shift
      - Mode register
      - Command register
      - Status register

Serial Communications (by using UART)





Serial Communications

- Standard protocols
  - USB (Universal Serial Bus)
    - Developed in 1996 to support connections between PC and peripherals (monitor, printer, modem, external memories, etc.)
      - Today there are differenet versions
        - » v1.x-v3.x - 1.5 Mbps – 5 Gbit/s
    - It allows star topologies by means of USB hubs connected to PC
      - The hub could be embedded into devices or stand-alone
        - » Different devices can be connected to an hub
    - USB host controller
      - Manages bandwidth and drivers needed for the different peripherals
      - Dynamically allocate power depending on connected devices

Serial Communications

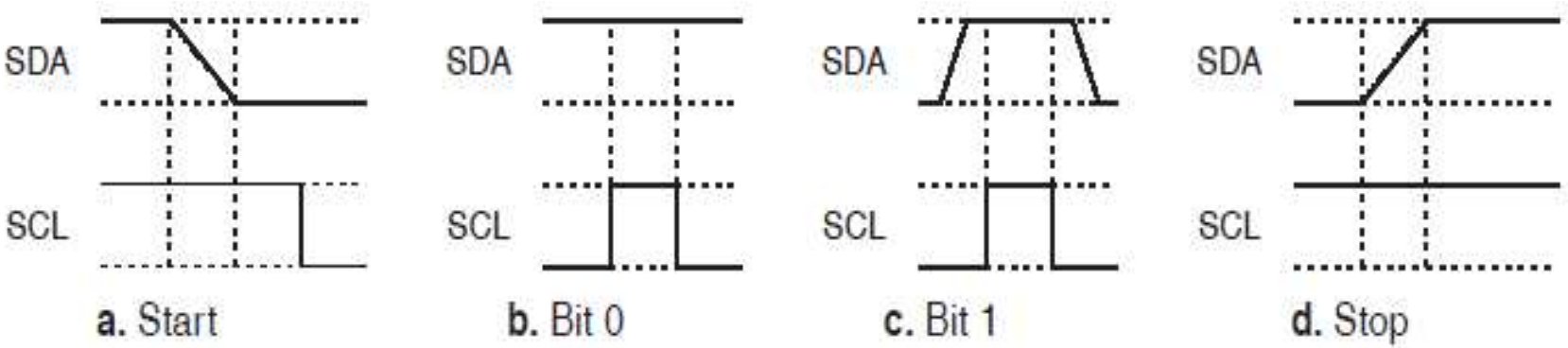
- Standard protocols
  - I^2C (*Inter IC*)
    - Developed by Philips at the beginning of ‘80
    - It has been designed to interconnect more devices limited distances
      - Variable data rate: from 100~Kbit/s with 7 bit addresses (original version) up to 3.4~Mbit/s with 10 bit addresses (recent versions)
    - It uses two lines: SDA (*Serial Data Line*) and SCL (*Serial Clock Line*) used to connect all the devices
      - Devices can be master or slave
        - » It is possible to have more than one master thanks to a deterministic and quite simple mechanism for bus arbitration

Serial Communications

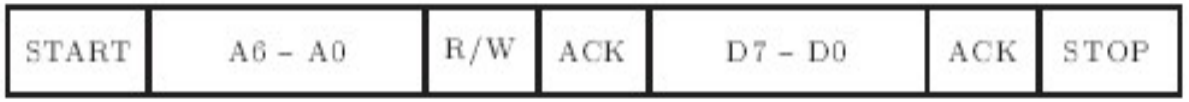
- Standard protocols
  - I^2C (*Inter IC*)
    - It is an isochronous/asynchronous bus since there is a dedicated clock line and synchronization is required for each transmitted character
    - Communications start with a *start condition* and teminates with a *stop condition*
      - A packet is composed of the following fields
        - » Start condition
        - » Address
        - » Requested operation
        - » Ack bit, written by the slave
        - » Data byte
        - » Another ack bit, written by the slave
        - » Stop condition

Serial Communications

- Standard protocols
  - I²C (*Inter IC*)
    - Start/stop conditions and 0/1 symbols



    - Packet logical structure

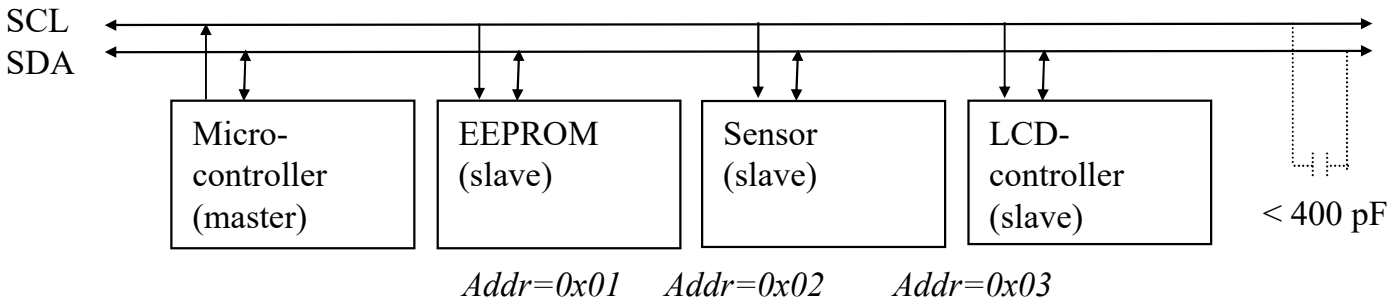


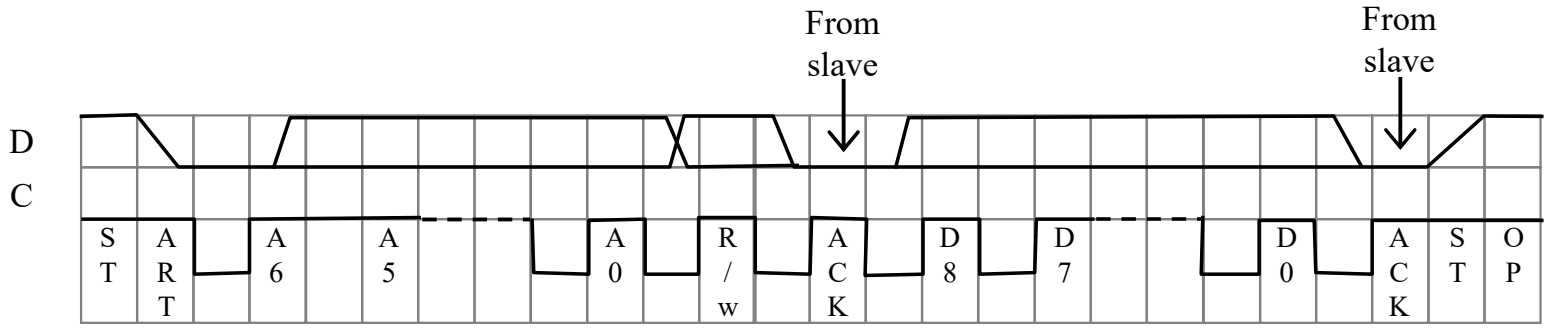
Serial Communications

- Standard protocols
  - I^2C (*Inter IC*)
    - Bits are ordered from the most to the less significant ones
      - The master has always the control of the lines apart from ack bits that are written by the slave
    - It is possible to connect more master devices
      - Arbitration mechanism that allows to a possible master to detect if it is really able to control the bus (i.e. no other master is already working)
        - » *Open-drain* connection that realize a *wired-and*
      - When a master want to control the bus it has to verify two conditions
        - » Check if the data line is 0
        - » Check if it is able to write a 1 on the clock line

Serial Communications

- Standard protocols
  - I²C (*Inter IC*)





Typical read/write cycle

Serial Communications

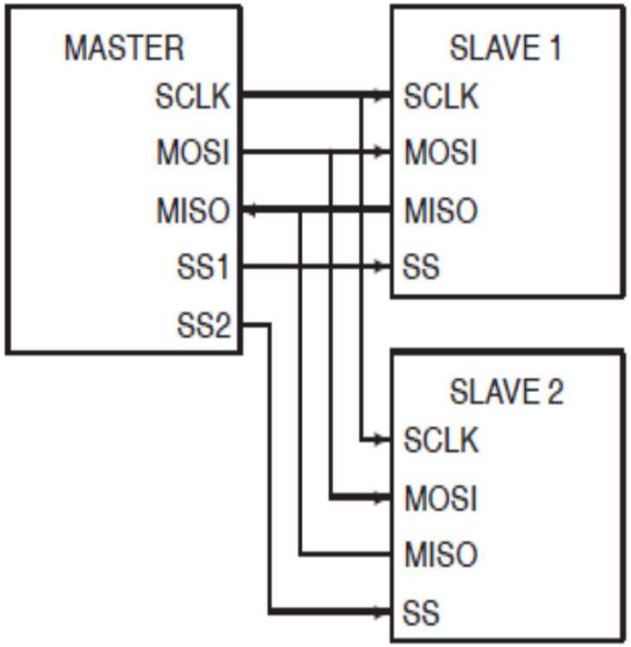
- Standard protocols
  - SPI (*Serial Peripheral Interface*)
    - It is based on a isochronous bidirectional connection with only one master
    - The protocol uses a bus with at least 4 lines for each slave
      - SCLK (*Serial Clock*)
      - MOSI (*Master Output Slave Input*)
      - MISO (*Master Input Slave Output*)
      - SS (*Slave Select*) for each slave
    - MISO signals are connected in *wired-or* on the same line so the not-active slaves (i.e. the one for which SS signal is high) have to keep MISO to Z

Serial Communications

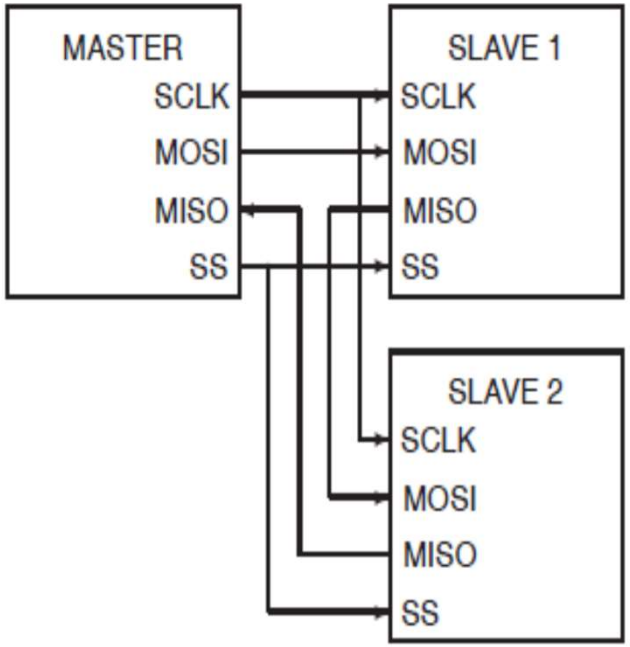
- Standard protocols
  - SPI (*Serial Peripheral Interface*)
    - To start a communication, the master uses the proper SS then generates a clok signal with frequency <= the max slave supported frequency by means of the SCLK line
      - Thank to this clock master and slave communicate in full-duplex by using MOSI and MISO lines
    - A drawback is that it is needed SS line for each slave
      - A daisy-chain schema alloes to remove this limitation

Serial Communications

- Standard protocols
  - SPI (*Serial Peripheral Interface*)



Parallel connection



Daisy-chain

Serial Communications

- Standard protocols
  - SPI (*Serial Peripheral Interface*)
    - SPI connections normally provide an higher throughput than I^2C
      - Full-duplex and No-Address
    - The main drawbacks are that it is not considered a physical-level ack mechanism and that it is allowed only one master device

Serial Communications

- Standard protocols
  - MIPI I3C (https://www.mipi.org/specifications/i3c-sensor-specification)
    - *Improved Inter-Integrated Circuit (aka SenseWire)*
    - The I3C standard has been developed in 2017 as a collaborative effort between electronics and computer related companies under auspices of the *Mobile Industry Processor Interface Alliance* (MIPI Alliance)
      - MIPI I3C is a specification for a two wires, shared, serial data bus that combines the advantage of the simple, two wire I^2C architecture with the higher communication speeds common to more complicated buses such as the Serial Peripheral Interface (SPI)
        - The I3C standard is designed to retain some backward compatibility with the I^2C system

Serial Communications

- Standard protocols
  - CAN (*Controller Area Network*)
    - CAN bus is based on a serial differential protocol, developed by Bosch at the beginning of ’80 to connect control devices in very noisy environments (from an electromagnetic point of view)
      - This bus is suitable to transmit little size messages (8 byte) that are associated to a CRC-15 code to protect them by errors
        - » It supports data-transfer up to 1~Mbit/s for <= 40~m distances but, by using lower frequencies, it can cover also greater distances (e.g. 500~m with 125~Kbit/s)
      - It belongs to the category of the so called *field-bus*

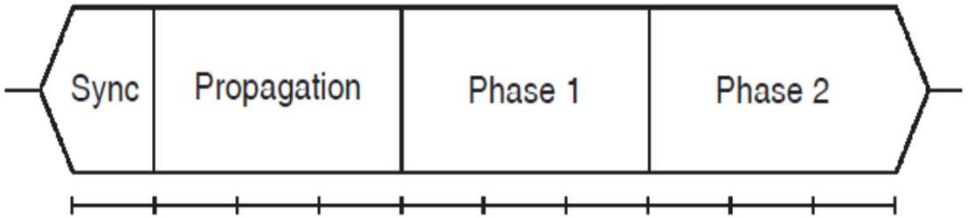
Serial Communications

- Standard protocols
  - CAN (*Controller Area Network*)
    - It is a multi-master bus that adopts an intrinsic priority-based arbitration mechanism
      - An higher-prority message is able to win the arbitration with respect to a lower-priority one and so it is transmitted without collisions
        - » This mechanism is realized by means of a model based on binary logic with *wired-or* and on the concepts of dominant and recessive bits

	dominant	**recessive**
dominant	*dominant*	*dominant*
recessive	*dominant*	*recessive*

Serial Communications

- Standard protocols
  - CAN (*Controller Area Network*)
    - It is an asynchronous bus that uses a whole structure for the transmission of each bit
      - It allows alignment of clock generators and sampling instants



    - Each bit is composed of 4 segments
      - Synchronization
      - Propagation
      - Aligment (Phase1 and Phase2)
        - » Data is sampled between the two phases

Serial Communications

- Standard protocols
  - CAN (*Controller Area Network*)
    - At an higher level the protocol uses *frames*
      - A frame is the basic transmission item and it can be of 4 types
        - » Data Frame, Remote Frame, Error Frame, Overload Frame
    - The most common one is the *data frame* that is used to exchange data, i.e. the major part of the traffic on the bus

