UNIVERSIDAD
COMPLUTENSE
MADRID

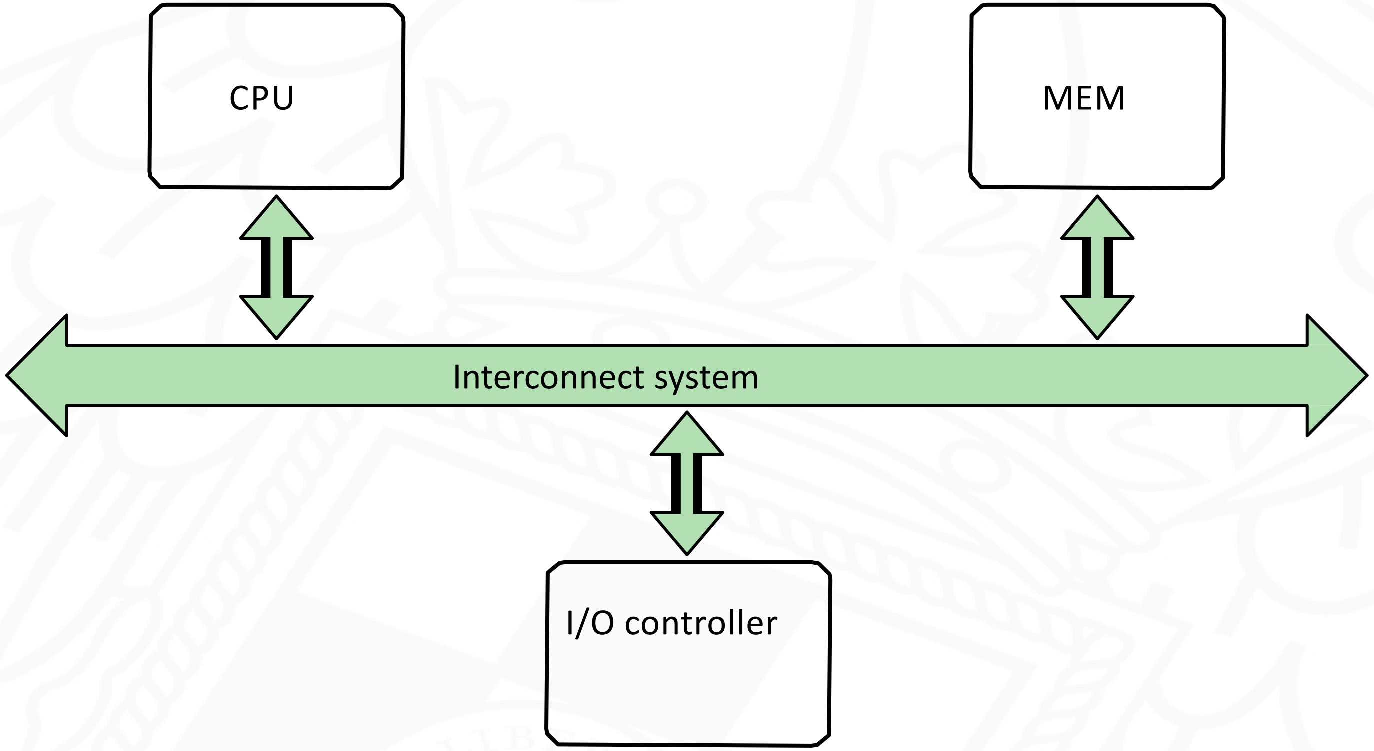
Serial buses.

IoT Node Architecture

Interconnect system and serial buses

- Interconnect system
  - Introduction
  - Classification of communication lines
  - Parallel communication
  - Serial communication
  - Synchronous and asynchronous communication
- Serial buses
  - UART
  - IIC
  - SPI

Interconnect system



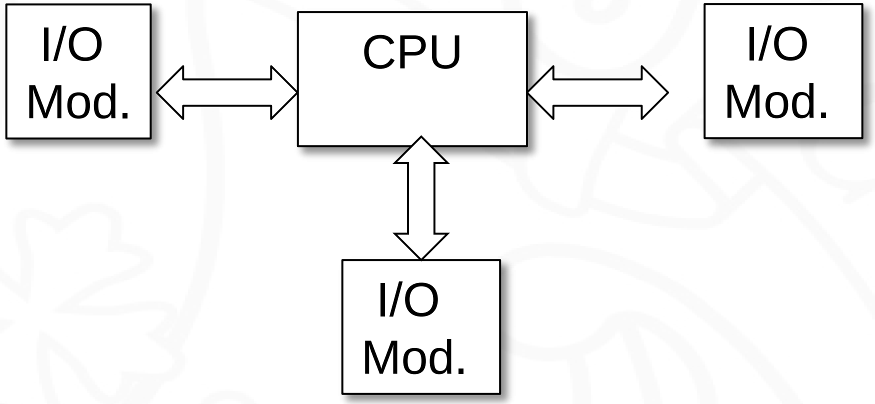
Classification of communication lines

- Depending on the following features:
  - Topology
    - Shared
    - Point-to-point interconnect
  - Width (number of lines)
    - Parallel
    - Serial
  - Synchronization mechanism
    - Synchronous
    - Asynchronous

Point to point vs shared

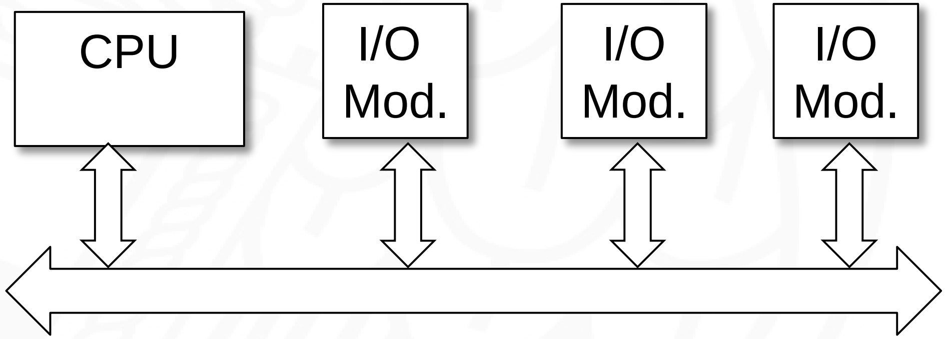
Point to Point Communication

- ✓ It can provide high performance
- ✗ It requires a large number of interfaces



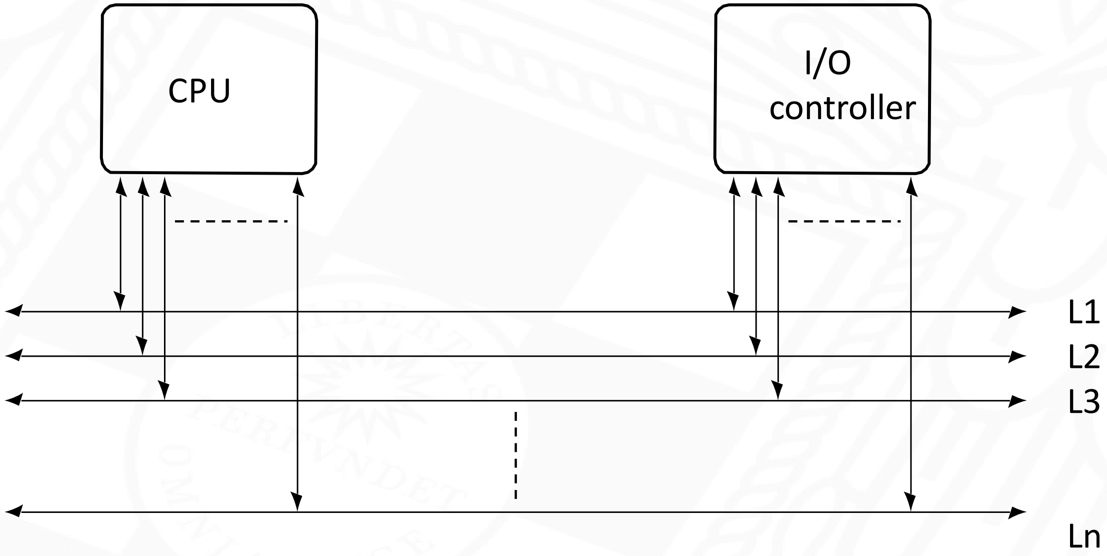
Shared: bus

- ✓ More versatile and lower cost
- ✗ It can create a communication bottleneck
- ✗ It requires synchronization between devices to prevent concurrent writes
- ✗ It may require arbitration



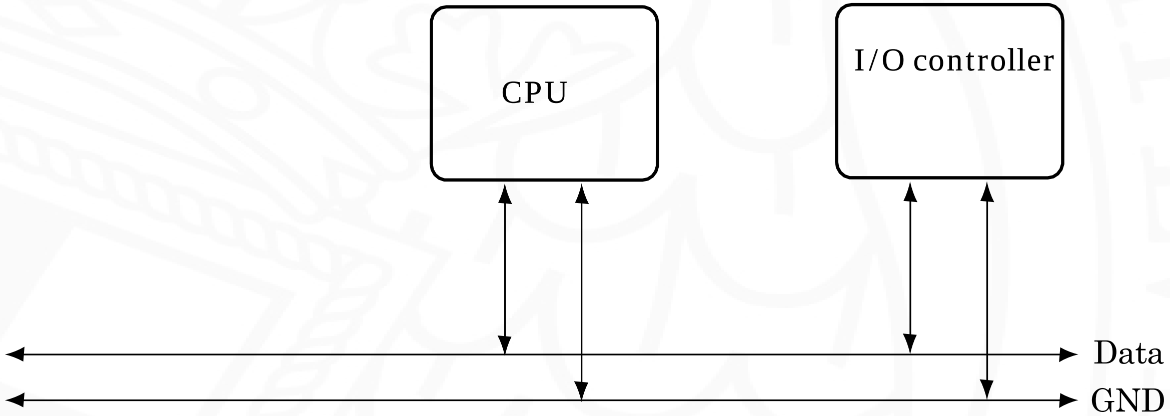
Parallel communication

- ❑ There are several lines (wires) that allow sending several data bits at the same time
  - ✓ High bandwidth (theoretically)
  - ✗ It is very complex to connect devices from medium to long distances
  - ✗ Operating frequency is limited due to physical factors
  - ✗ The low-speed devices do not use the potential speed of the parallel transmission (mouse, modem ...)



Serial communication

- Data bits are transferred sequentially through one unique data line
  - There can exist other lines for control (i.e. clk, select,. . .)
  - Some time ago, it was only used for long distance communication with external components
  - It is increasingly being used for short distances, even in internal buses (i.e. PCI Express)
- Advantages:
  - Cheaper
  - Higher frequencies (without *clock skew*)
- Drawbacks:
  - Lower bandwidth (for the same frequency)
  - Increasing bandwidth→ several serial connections



All the information (data and control) is sent serially through one line, while the second line is used as a reference

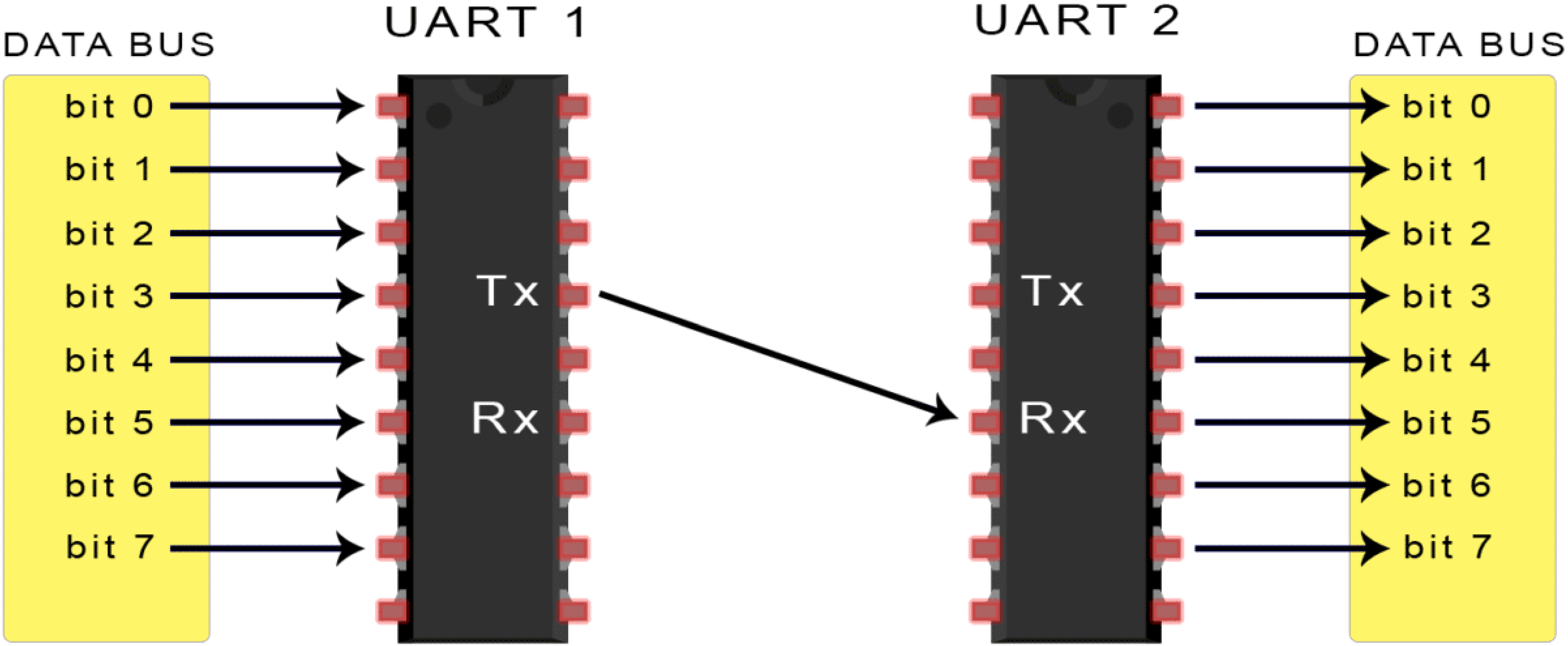
Synchronous vs asynchronous

- When does the transmission of data start/finish?
- Asynchronous
  - The clock signal is NOT sent through the communication line
  - The transmitter and the receiver use their own clock signals
  - It is necessary to establish a synchronization protocol among them
  - Character oriented communication
  - Example: UART (RS-232, . . .)
- Synchronous
  - The clock signal is sent along with the other signals
  - Block oriented communication
  - Examples: I^{2}C, SPI, 1-Wire, . . .

Serial buses: 1) UART

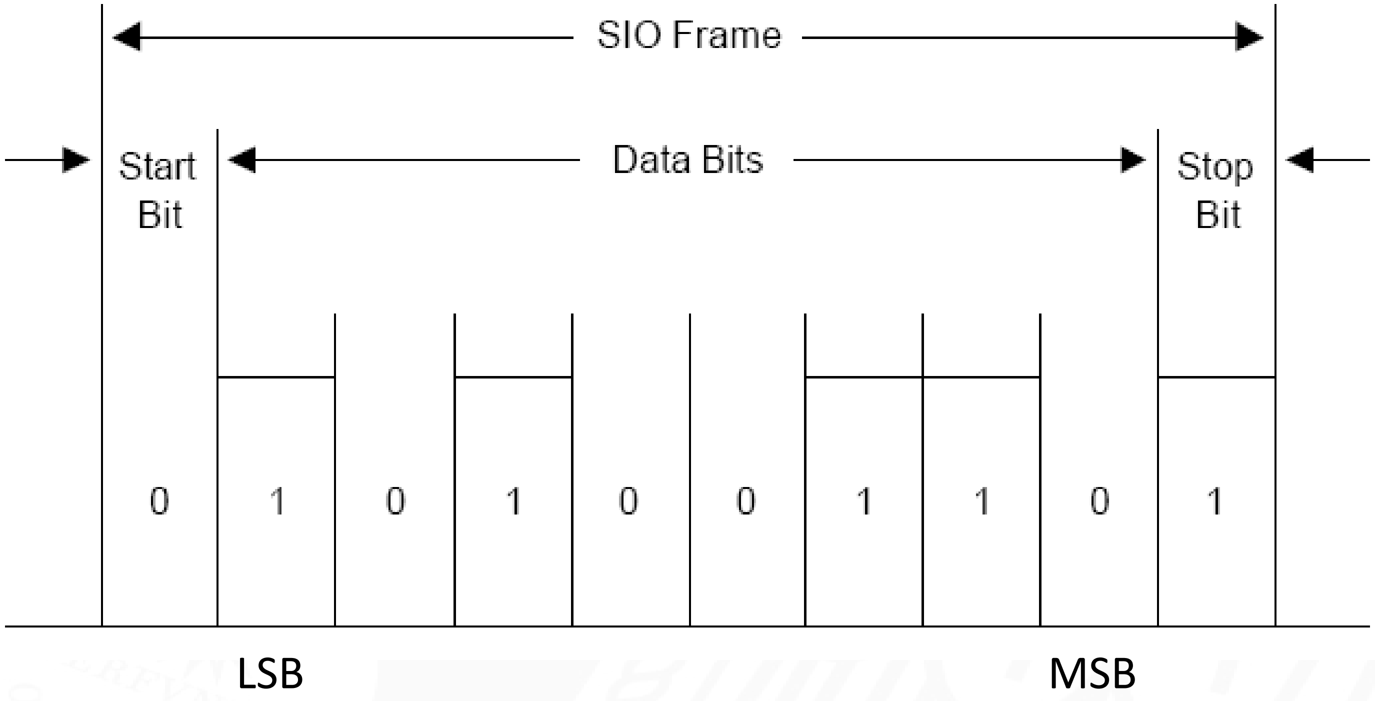
- The Universal Asynchronous Receiver/Transmiter is an asynchronous serial bus.
  - Used to make parallel/serial conversion and transmit data through a serial port
  - Configurable speed and data format
- Simple method of exchanging data using minimal system resources
  - Unidirectional (Half-duplex) or bidirectional (Full-duplex)
- Two common signal levels are RS-232, a 12V system, and RS-485, a 5V system, but there are other options.
- Since there are no clock signals sent, the sender and receiver need to work at the same frequency.

Connecting two devices using UART



UART: data frame

- Transfers are characted oriented: 5/6/5/8 bits per character
- It sends data bits one by one, from the least significant to the most significant, framed by start and stop bits.
- For each char:
  - Start bit (logic 0)
  - Data bits (5 to 8)
  - Parity bit (optional)
  - 1 or 2 Stop bits (logic 1)



Serial buses: 2) IIC

- The Inter-Integrated Circuit Bus (IIC, I2C, I^{2}C) is a synchronous serial bus developed by *Philips Semiconductor* in the early 80s.
- Multi-master and multi-slave
- Although the lines are bidirectional, the connection is unidireccional:

  master → slave or slave → master (halfduplex)

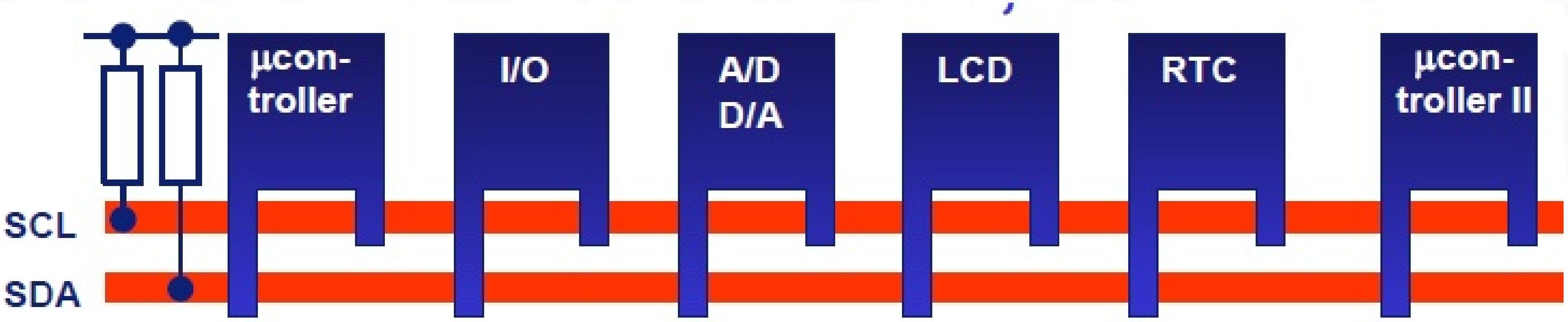
- Goal → Connecting integrated circuits with minimal number of pins
  - Low cost and simplicity
  - But also low speed
- Only three lines
  - Serial Data (SDA): all data
  - Serial Clock (SCL): clock signal, generated by the master
  - Ground (GND)

Inter-Integrated Circuit Bus (IIC, I2C, I^{2}C)

- The IIC reference design has a 7-bit address space.
  - Every device connected to the bus must have a different one.
  - Up to 112 nodes can be connected using 7-bit addresses
  - There is an extension with 10-bit addresses
- There are different communication modes that work with different bit rates:
  - Standard: 100Kb/s,
  - Fast: 400Kb/s,
  - Fast Plus:1Mb/s,
  - High-speed: 3.4Mb/s,
  - Ultra-fast: 5Mb/s

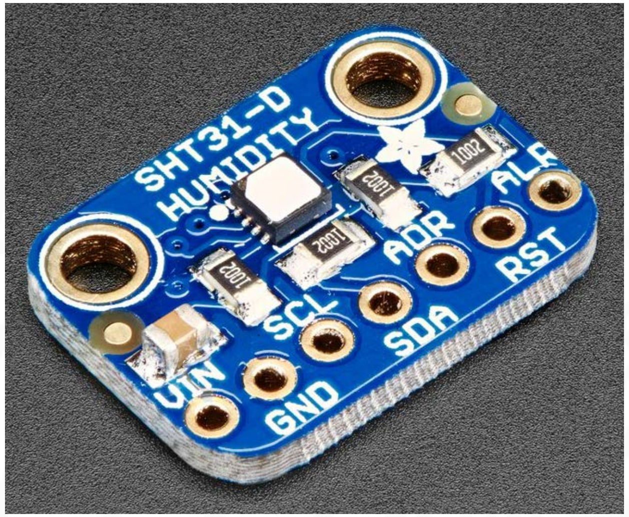
IIC: Physical properties

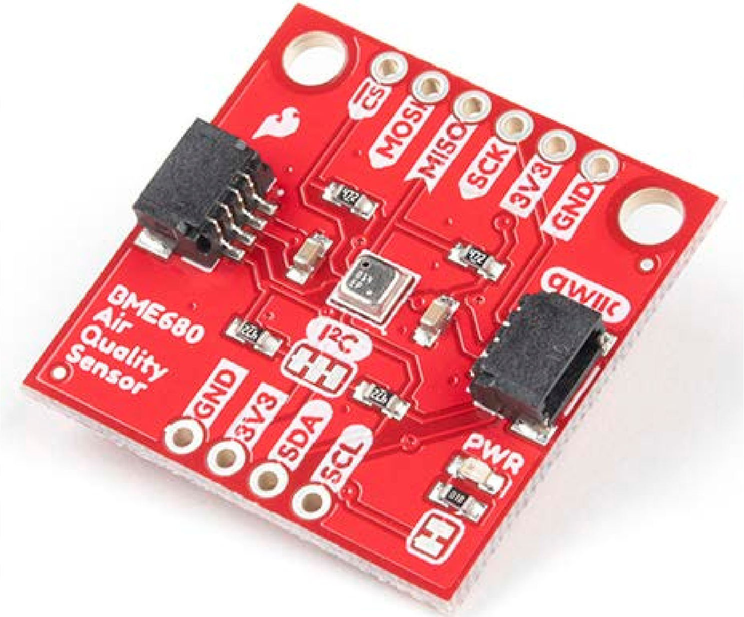
- IIC bus uses 2 bidirectional open-collector/open-drain lines pulled-up with resistors.
  - Connected to Vdd (logic 1) if not being used.
  - If any device puts a line to 0, it will be 0 (wired-AND)



IIC: Physical properties

- There are not standard connectors



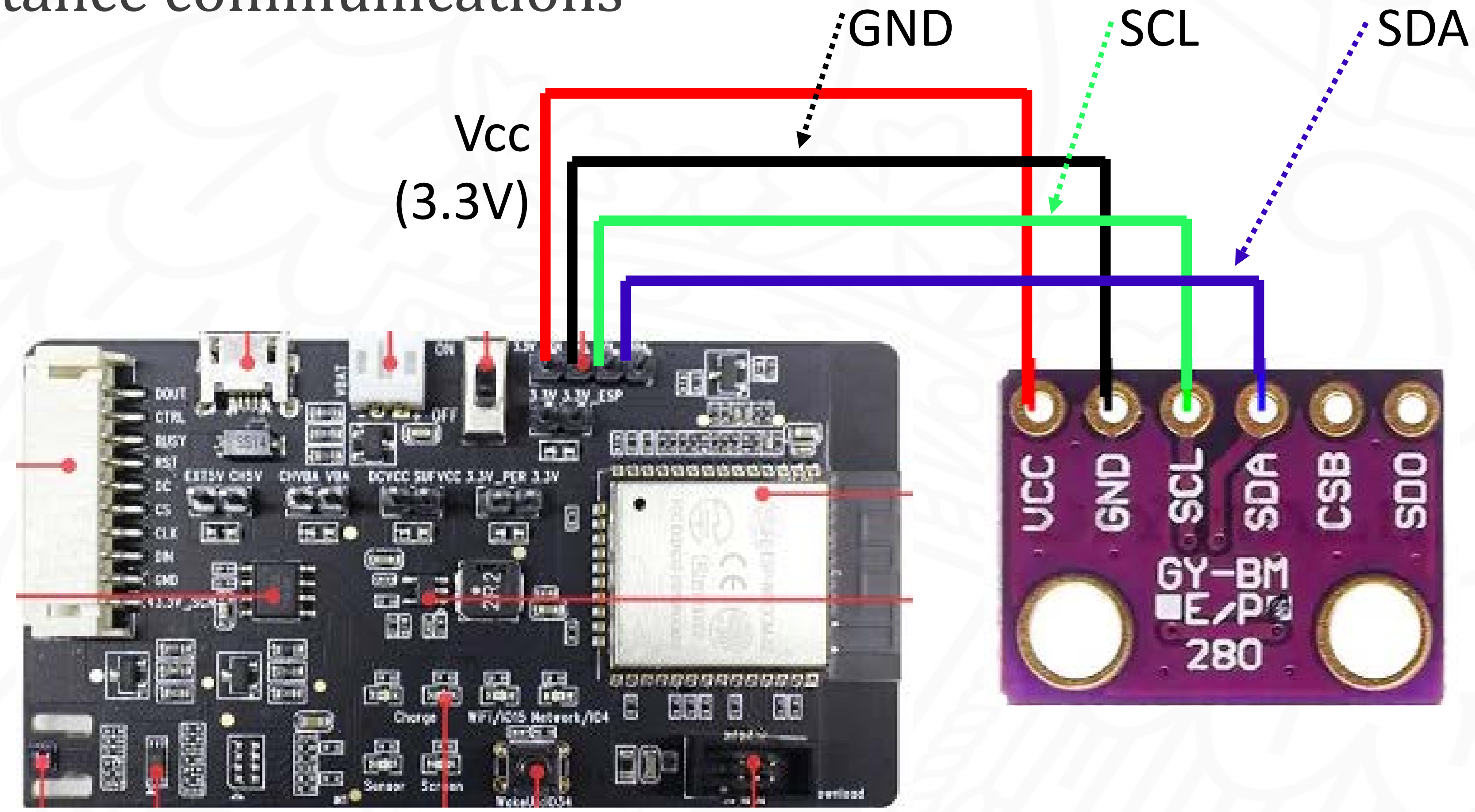


Adafruit Sensirion SHT31-D - Temperature & Humidity Sensor

SparkFun Environmental Sensor Breakout - BME680

Connecting with a sensor using IIC

- It is widely used for attaching low-speed peripheral ICs to microcontrollers in short distance communications



ESP32-MeshKit-Sense + 4 wires + any IIC sensor (BMP280)

IIC: Types of nodes

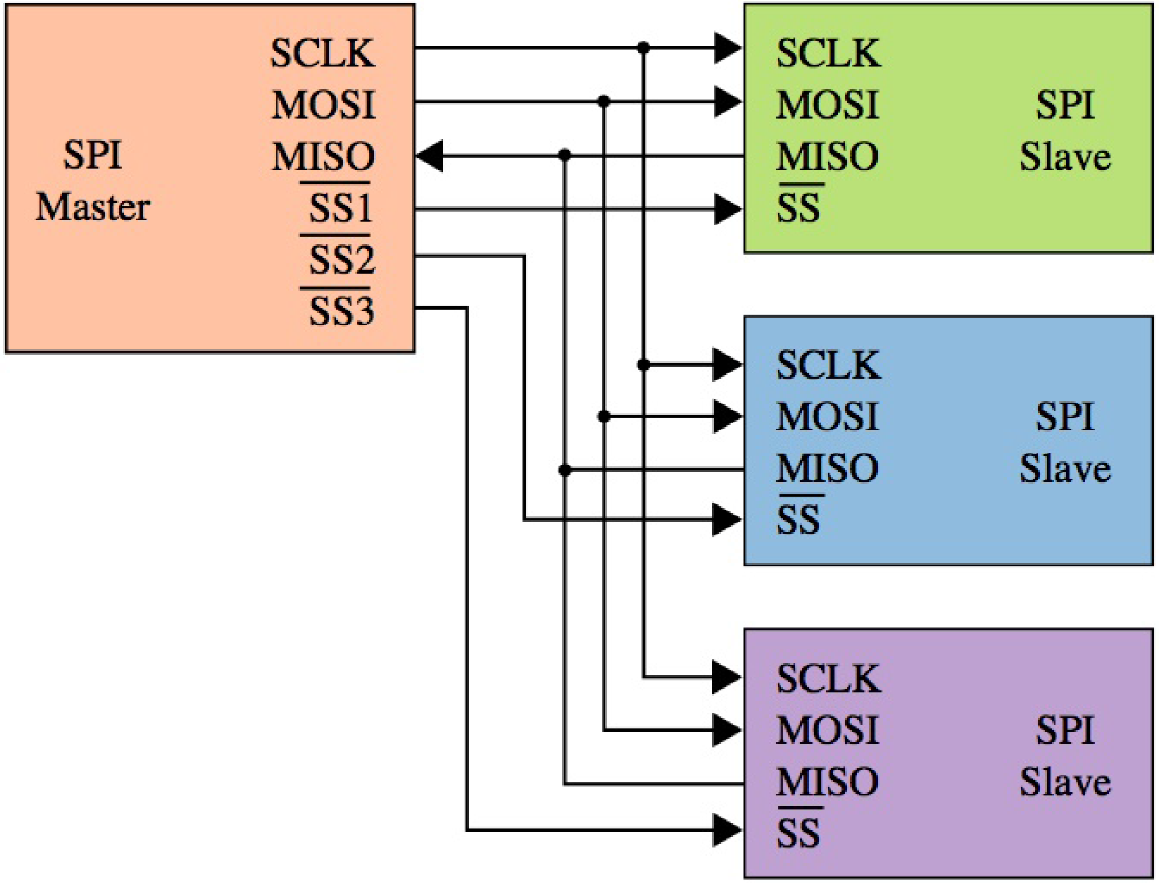
- There are two roles for nodes:
  - Master (controller) → Generates the clock and initiates communication.
    - Starts and stops the transfer
    - Sends the address of the target
  - Slave (target) → Receives the clock and responds when addressed by the controller.
- Controller and target roles may be exchanged between messages
- 4 potential modes of operation:
  - Master transmit/ receive
  - Target transmit/ receive
- Multi-slave and multi-master → Arbitration and conflict detection mechanisms needed

Serial buses: 3) SPI

- The Serial Peripheral Interface (SPI) bus is a synchronous serial connection.
- Single-master
- It is bidirectional (full duplex)
- Widely used for connecting a microcontroller with peripherals
  - Fast: 1-20MHz
- Four lines:
  - Master out/Slave in (MOSI, SDI)
  - Master in/Slave out (MISO, SDO)
  - Serial clock (SCK)
  - Slave Select (nSS, SS): active low

Connecting with external devices using SPI

- When used as a master, each SPI controller can drive multiple SS signals to activate multiple slaves.


SPI
Master
SCLK
MOSI
MISO
$\overline{SS1}$
$\overline{SS2}$
$\overline{SS3}$
SCLK
MOSI
MISO
$\overline{SS}$
SPI
Slave
SCLK
MOSI
MISO
$\overline{SS}$
SPI
Slave
SCLK
MOSI
MISO
$\overline{SS}$
SPI
Slave