Sensors & Buses

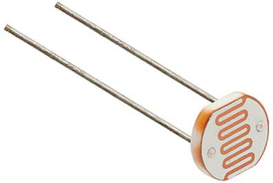
ELEC-D0301 Protopaja

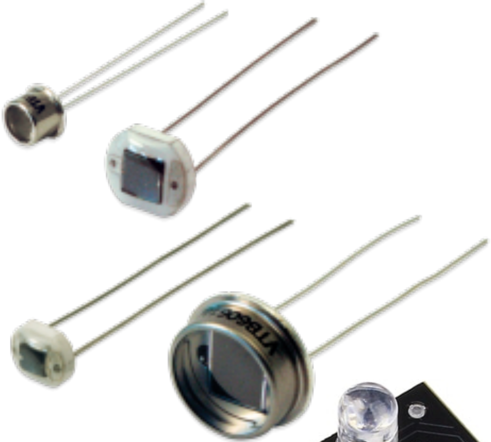
Aalto University
School of Electrical
Engineering

Aleksi Zubkovski
(Based on slides by Juha Biström & Mikko Simenius)

9.6.2021

Sensors

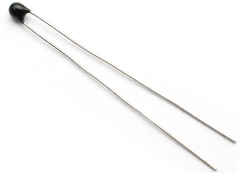


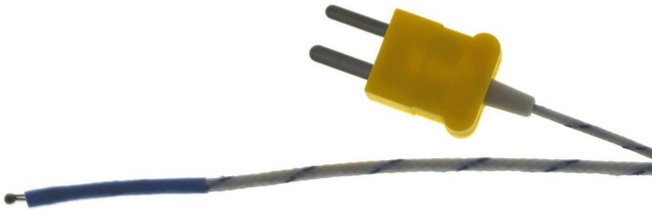


- **Light, Colour**
  - LDR (photoresistor), Phtodiodes, complex semiconductors

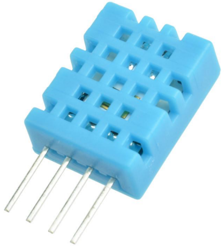
- **Sound**
  - Microphone, Piezo

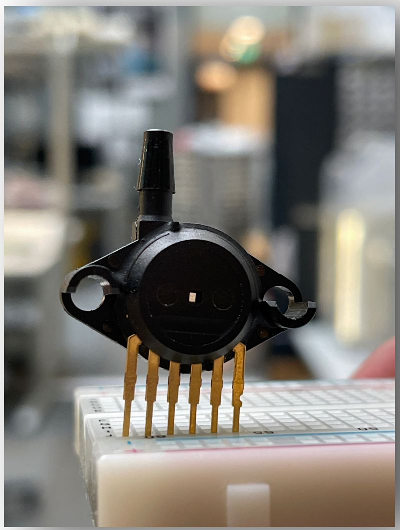
Sensors





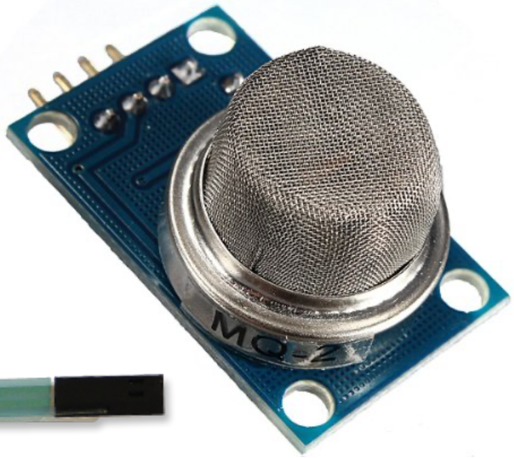
- **Temperature**
  - Thermistor, Thermocouple, PT100, LM35, DS18x20…
- **Humidity, Moisture**
  - E.G. DHT11
- **Pressure**
  - E.G. MPX5100AP

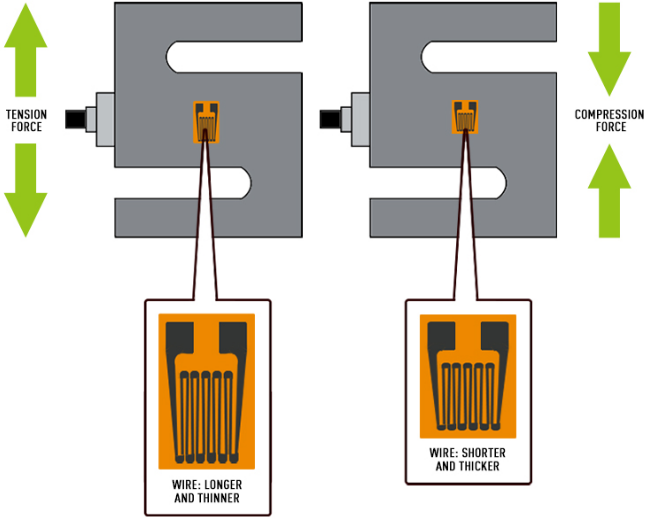


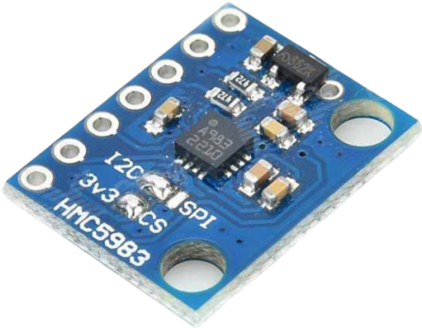


Sensors

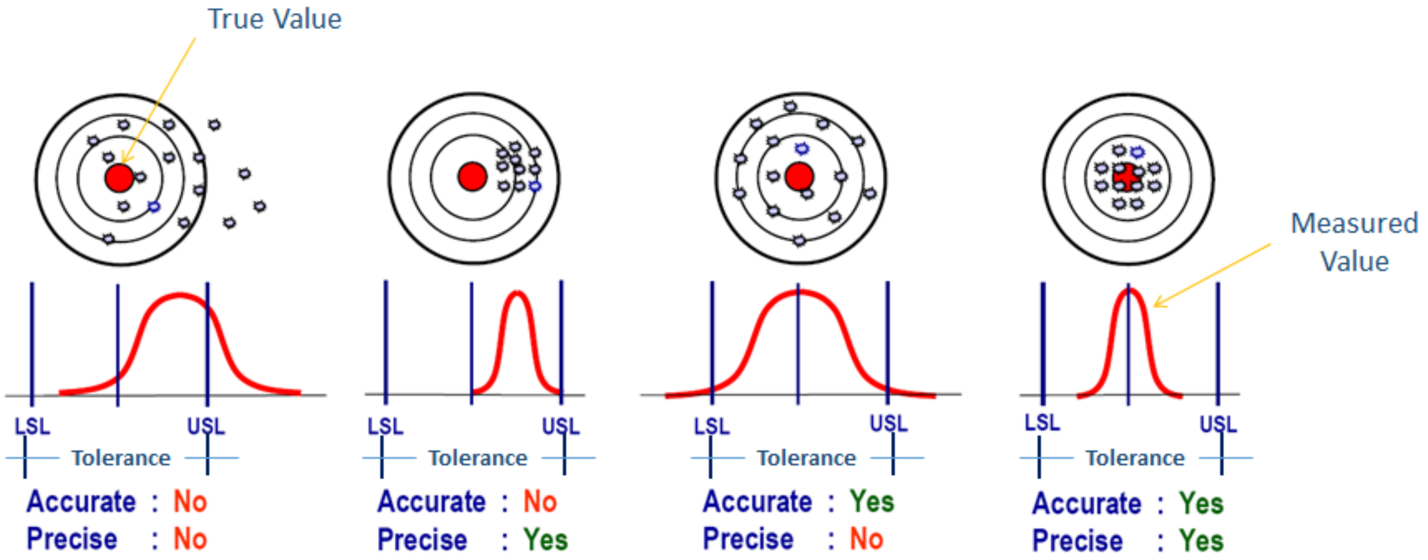
- **Gas Contents**
  - For example MQx for different gases
  - MQ3 - for ethanol
- **Force Sensors**
  - Force Sensing Resistor (Sensing Stretch)
  - Load Cell
- **Magnetic Field**
  - E.G. HMC5883







Calibration: Accurate vs Precise



Nothing can be done.

Sensor Calibration

Data Averaging

No action necessary

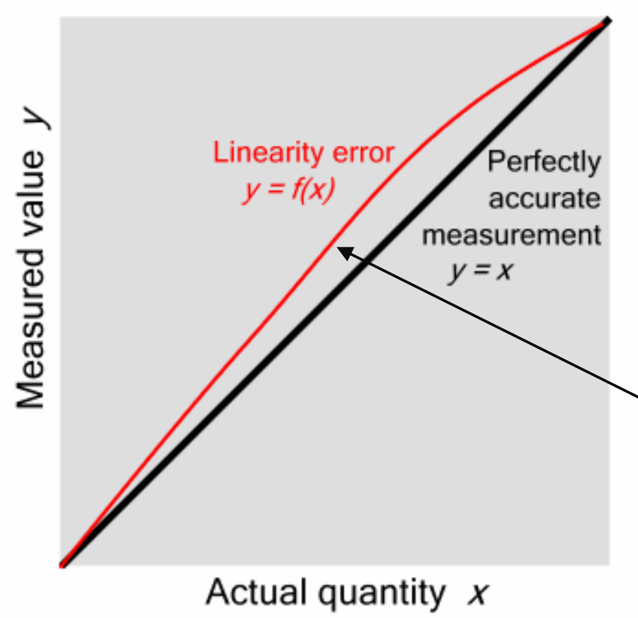
Aalto University
School of Electrical Engineering

Sensor Characteristics: Stability & Drift

- A sensor is stable if it is able to produce consistent measurements for constant environment
- Drift = differentiation around the constant

Important in process monitoring, medical equipment, etc.

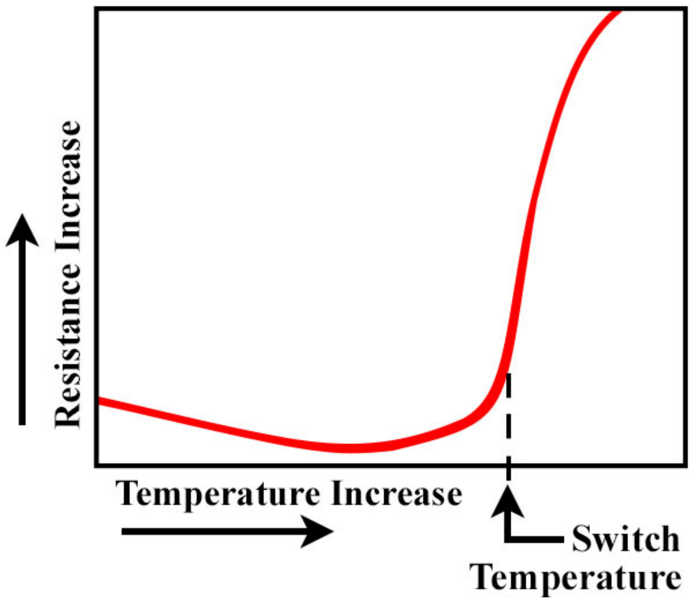
Sensor Characteristics: Linearity



Non-linearity

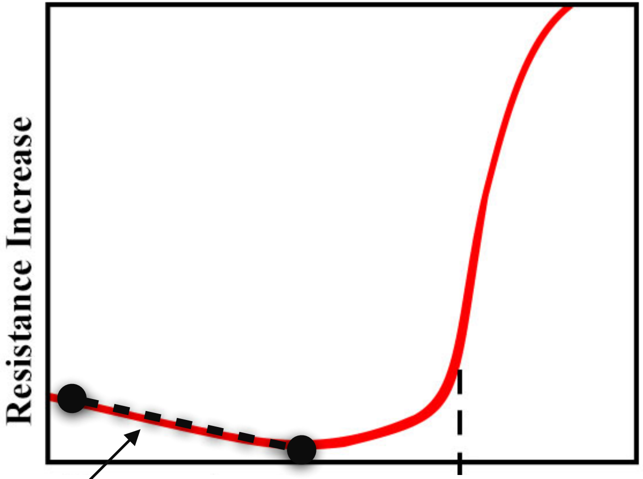
Sensor Characteristics: Linearity

E.G.Thermistor response curve:



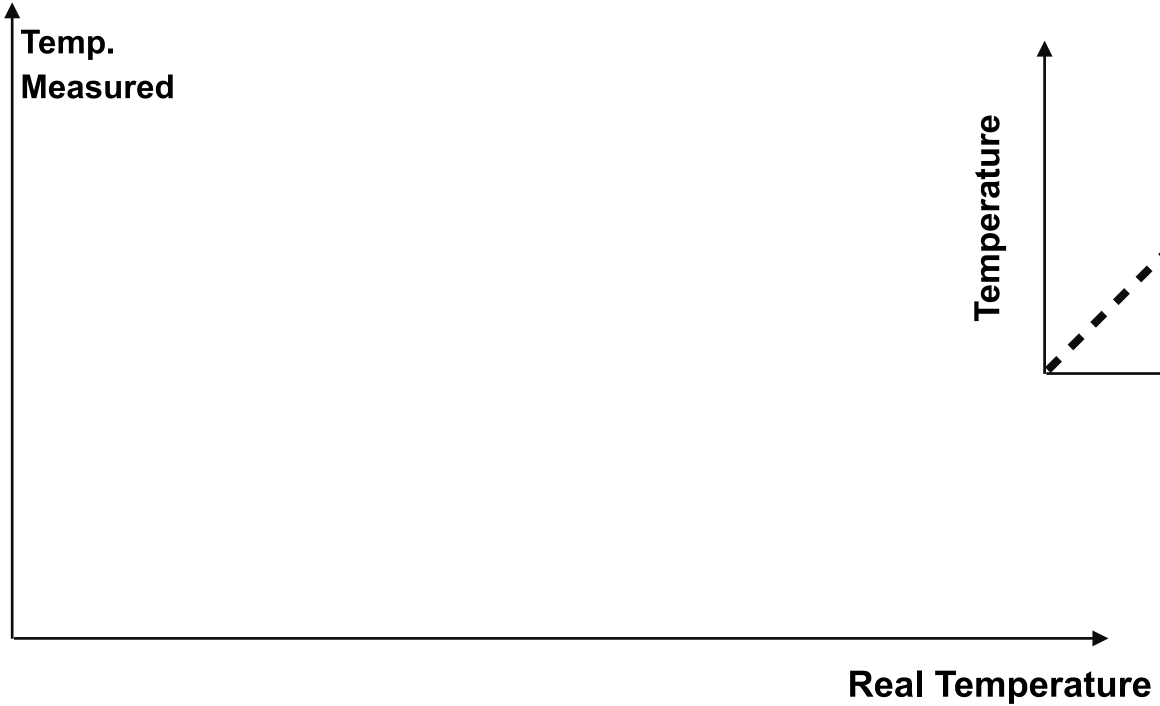
Sensor Characteristics: Linearity

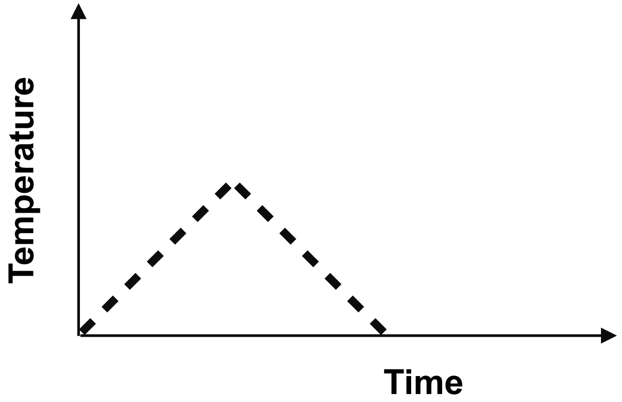
E.G.Thermistor response curve:



Chose sensors that are linear in range of use

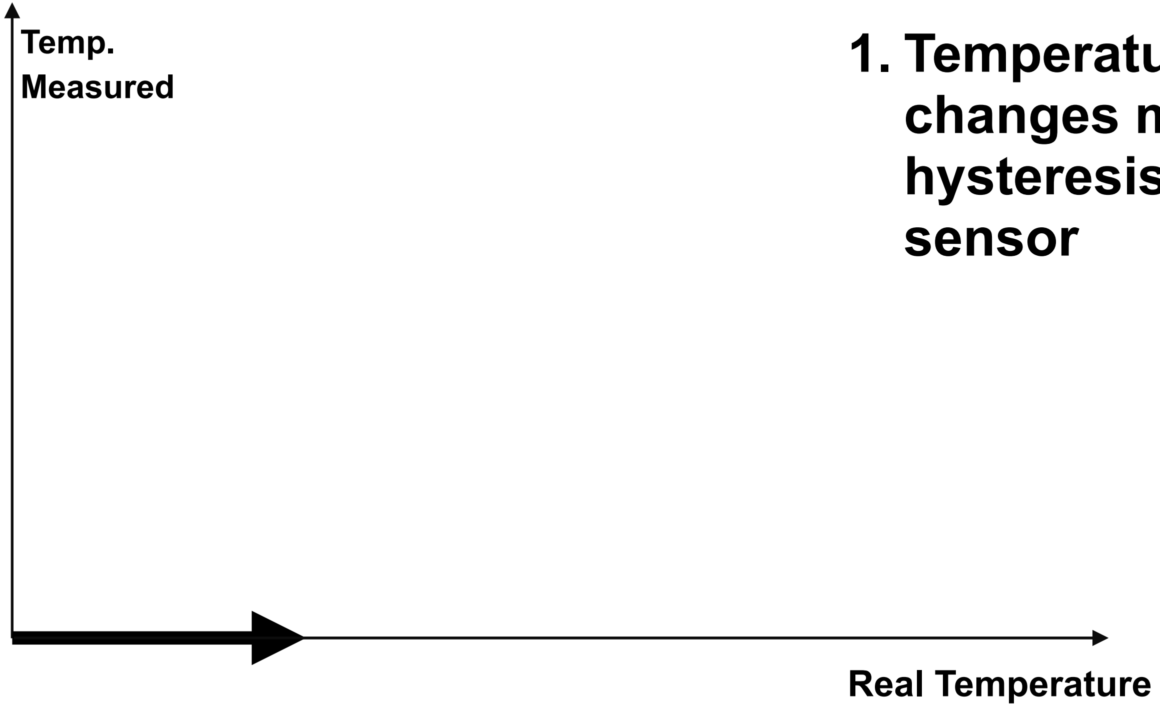
Sensor Characteristics: Hysteresis





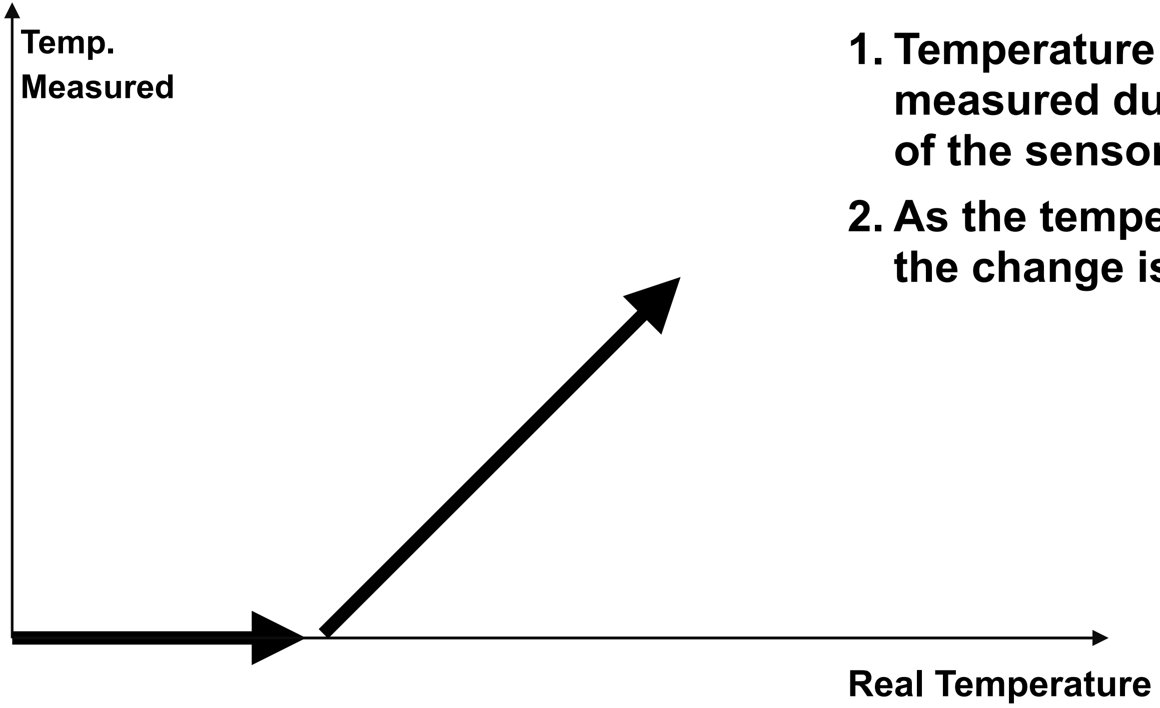
Aalto University
School of Electrical
Engineering

Sensor Characteristics: Hysteresis



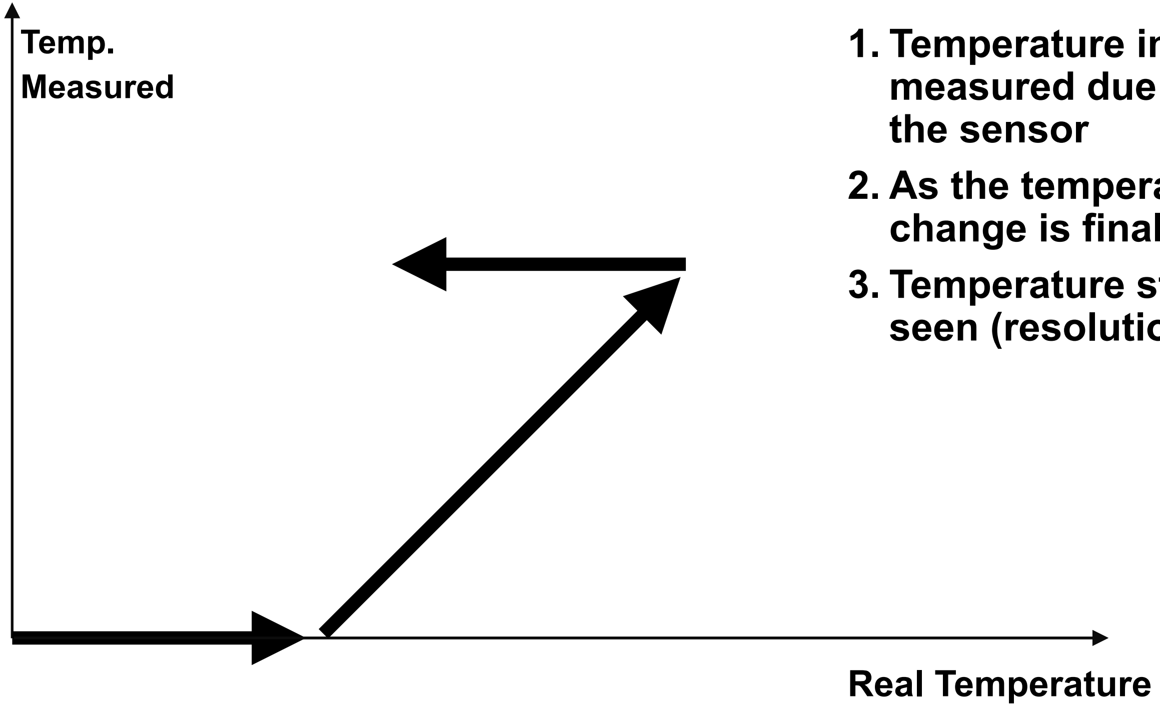
1. Temperature increases, but no changes measured due to hysteresis & resolution of the sensor

Sensor Characteristics: Hysteresis



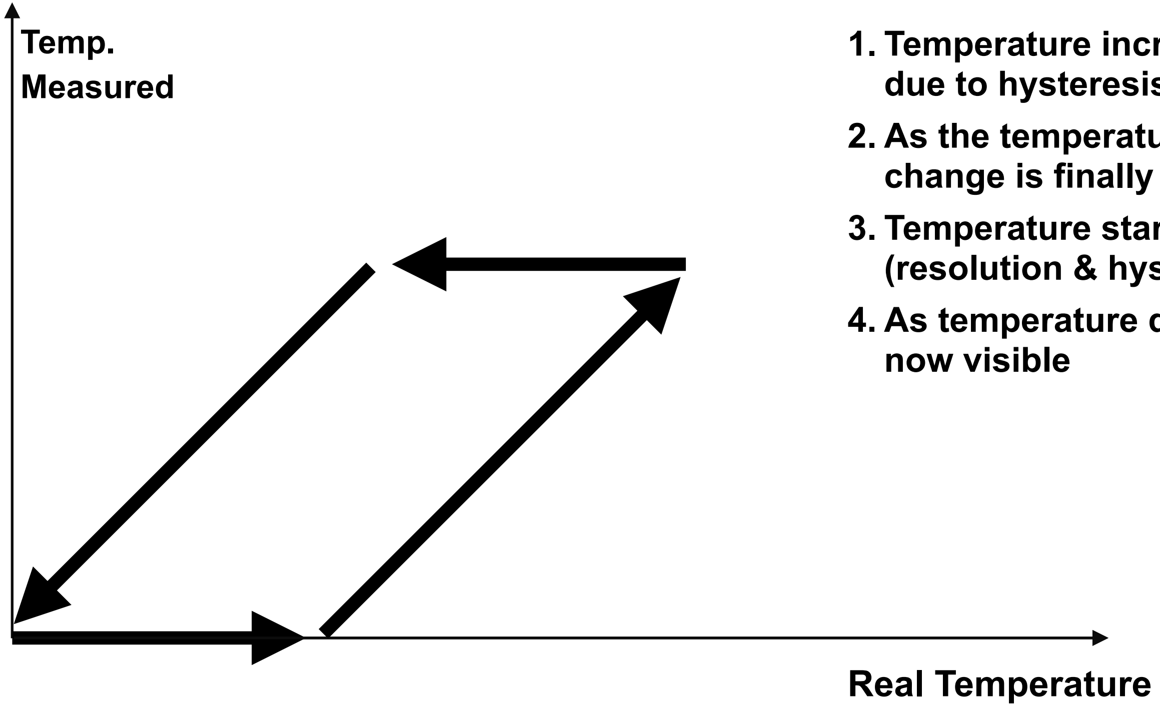
1. Temperature increases, but no changes measured due to hysteresis & resolution of the sensor
2. As the temperature continues increasing the change is finally seen

Sensor Characteristics: Hysteresis



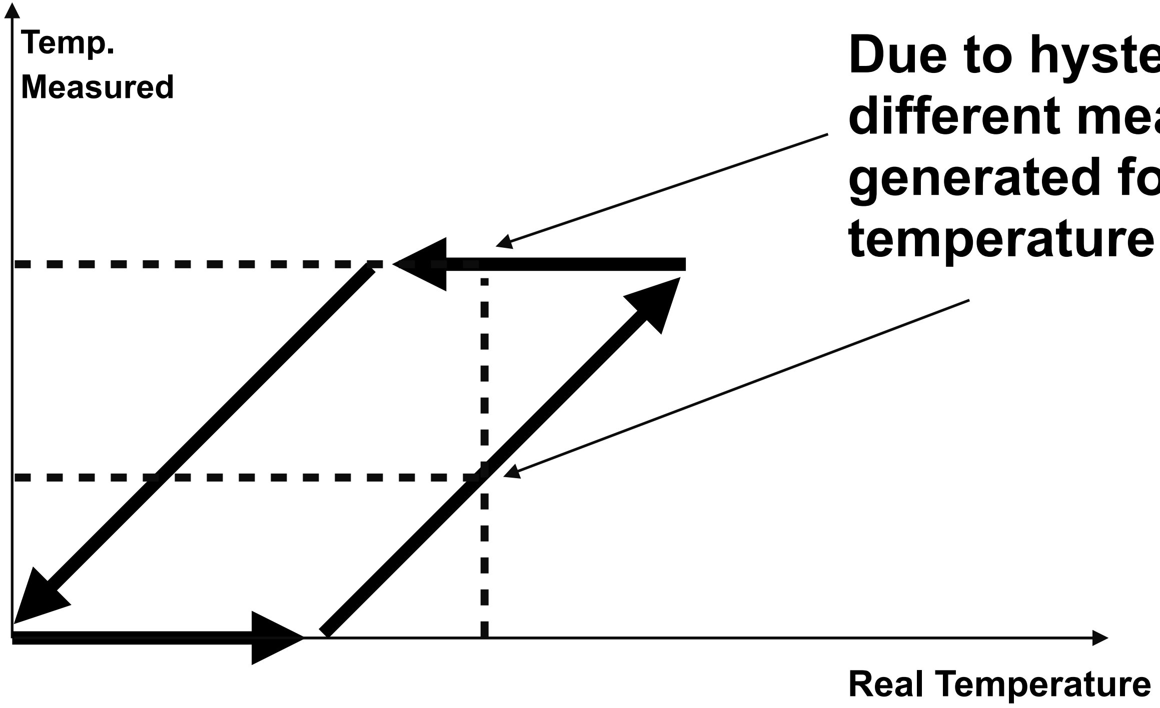
1. Temperature increases, but no changes measured due to hysteresis & resolution of the sensor
2. As the temperature continues increasing the change is finally seen
3. Temperature starts dropping, but no change seen (resolution & hysteresis)

Sensor Characteristics: Hysteresis



1. Temperature increases, but no changes measured due to hysteresis & resolution of the sensor
2. As the temperature continues increasing the change is finally seen
3. Temperature starts dropping, but no change seen (resolution & hysteresis)
4. As temperature decreases enough the change is now visible

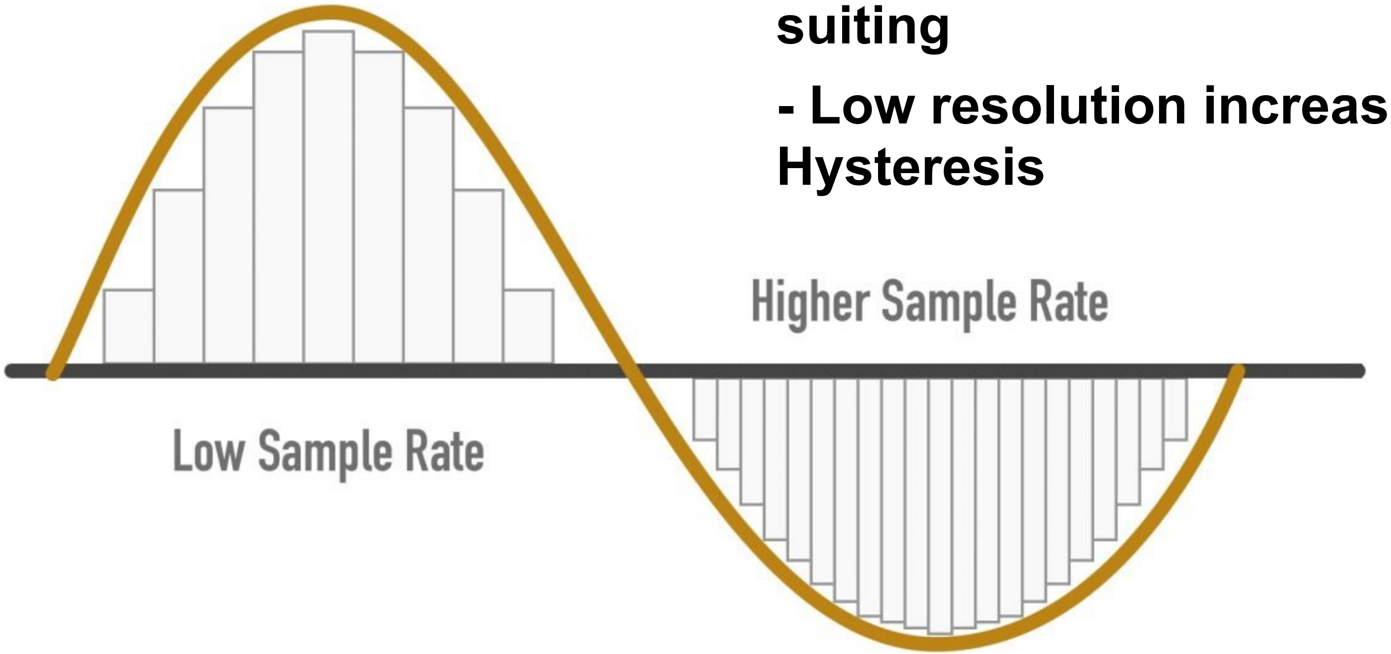
Sensor Characteristics: Hysteresis



Due to hysteresis we get two different measurements are generated for the same temperature!

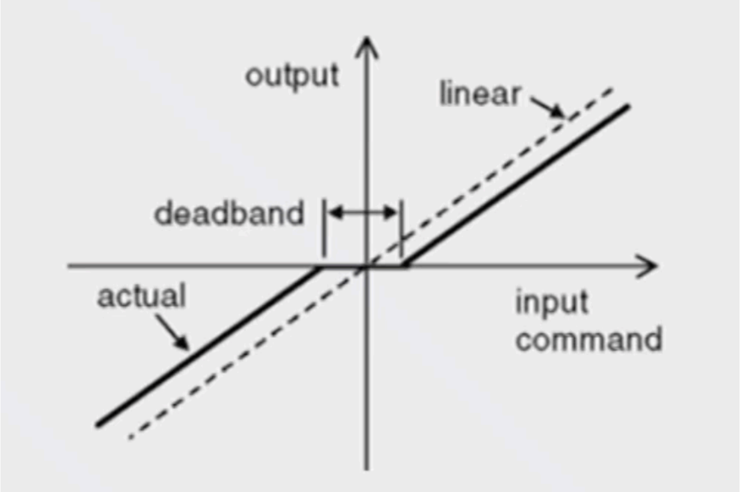
Sensor Characteristics: Resolution

- Sensor resolution should be suiting
- Low resolution increases Hysteresis

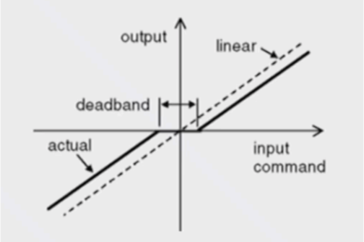


Sensor Characteristics: Dead Space

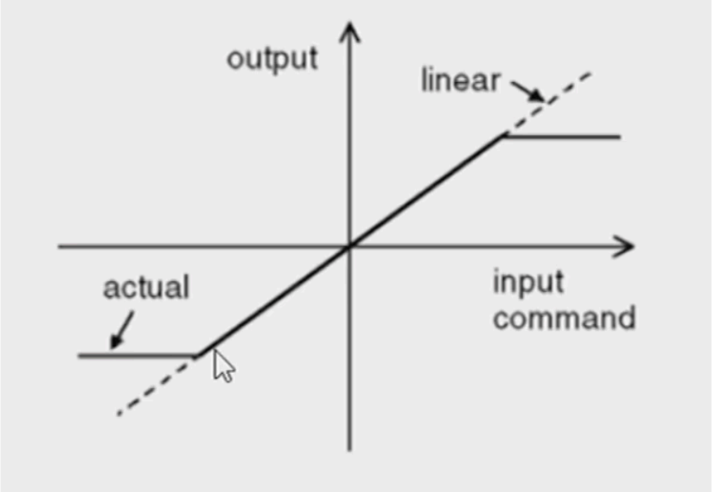
- Some sensors often have Dead Zones (usually near zero value)



Sensor Characteristics: Saturation

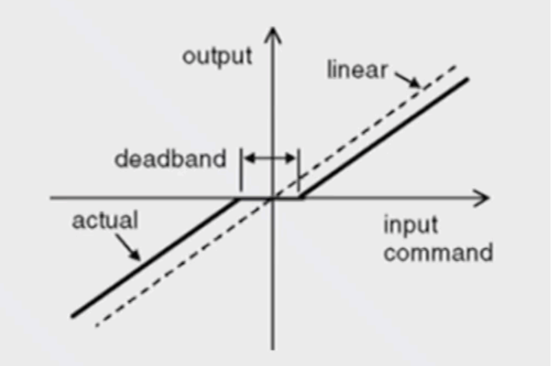


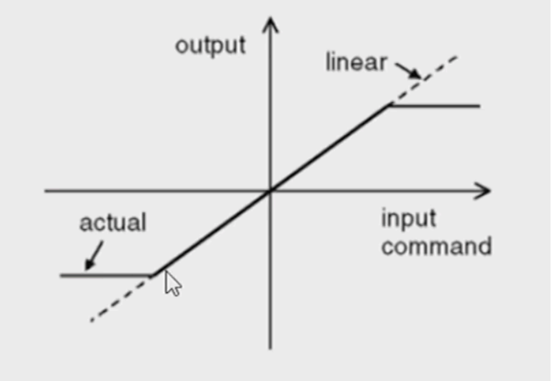
- Some sensors often have Dead Zones (usually near zero value)
- Saturation



Sensor Characteristics: Data-sheet

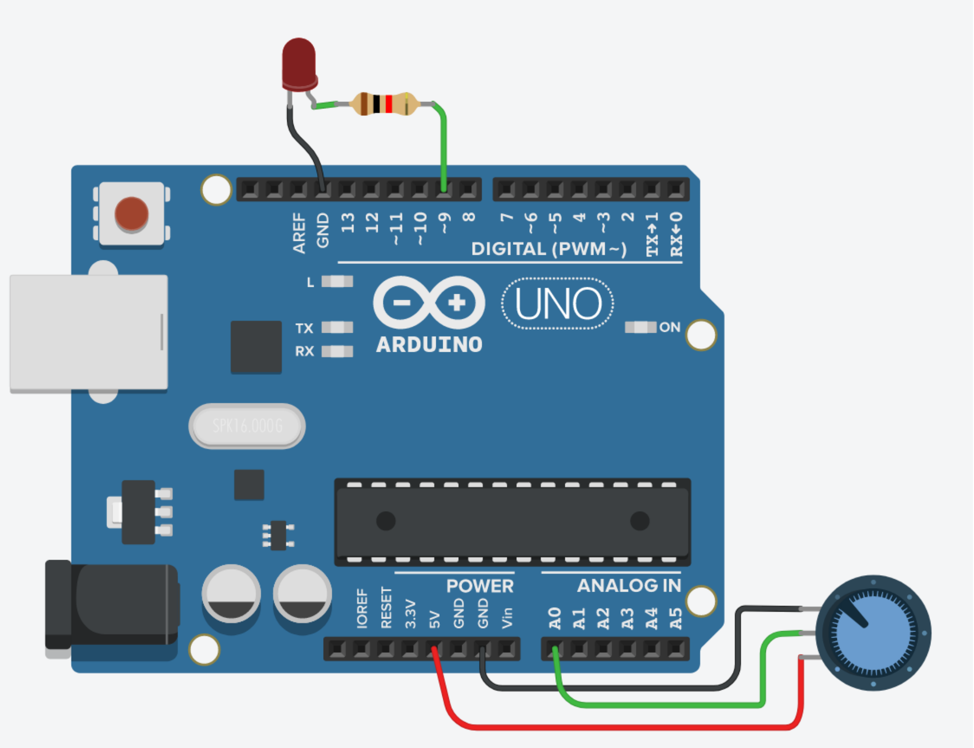
- Some sensors often have Dead Zones (usually near zero value)
- Saturation
- Characteristics are documented in sensor data-sheet





Analog & Digital Sensor Outputs

- If analog sensor data is processed by digital hardware it must be converted.



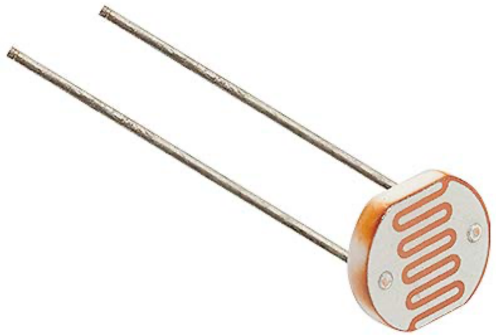
Aalto University
School of Electrical
Engineering

Analog & Digital Sensor Outputs

- If analog sensor data is processed by digital hardware it must be converted.
- Most sensors have A/D converters built in (easier to use)

Analog & Digital Sensor Outputs

- If analog sensor data is processed by digital hardware it must be converted.
- Most sensors have A/D converters built in (easier to use)
- High performance sensors usually have only analog inputs



A/D Converters

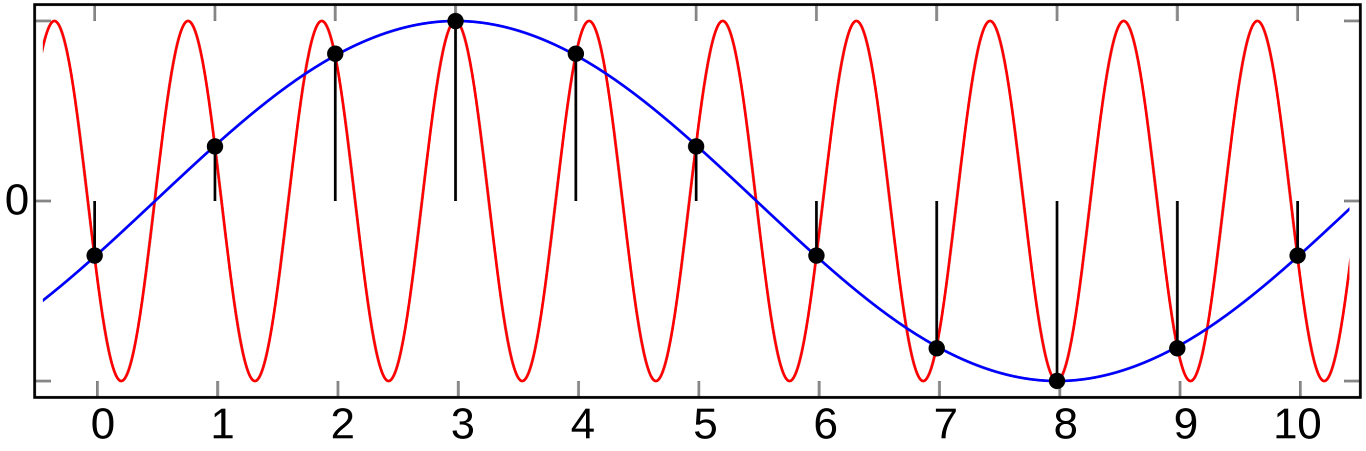
- Convert analog data to digital through quantisation
  - Leads to some data loss
- MCU’s usually have builtin A/D’s
  - E.g. in Arduino UNO whenever we use analog pins (*analog_read (___)*) MCU’s A/D unit is used.
  - External converters can be used too
- Noise considerations:
  - Component noise
  - Layout considerations
  - ADC saturation: apmlify signal before feeding it to the ADC.
    <u>Signal ~ ADC IN</u>

Signal Sampling

- Sampling frequency:
    - How frequently samples are taken (Value Differentiation Accuracy)
      E.g. Arduino UNO: almost 1kHz
- Bit depth (Resolution):
    - How precise values can be measured. (Value Accuracy)
    - E.g. Arduino UNO: 10 bit (= 1024)
- Nyquist Theorem:
  Sampling Frequency must be higher than 2 x maximum frequency being sampled.
    - E.g Audio freq. range is 20 Hz — 20 kHz, Sampling frequency must be 40 kHz

Signal Sampling

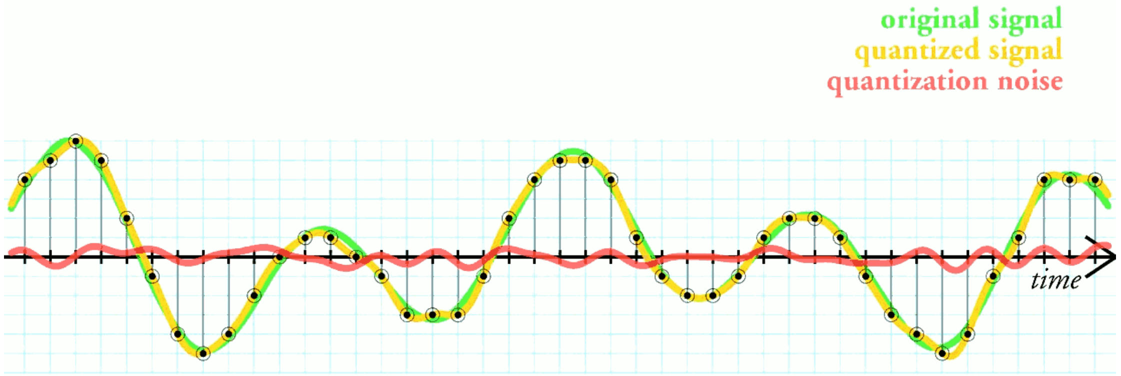
- **Aliasing**
  - **With low sampling frequency aliasing may occur**



- **Low pas filter may prevent aliasing, Aliasing filter**

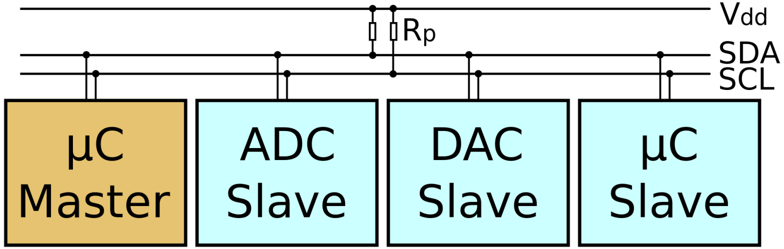
Signal Sampling

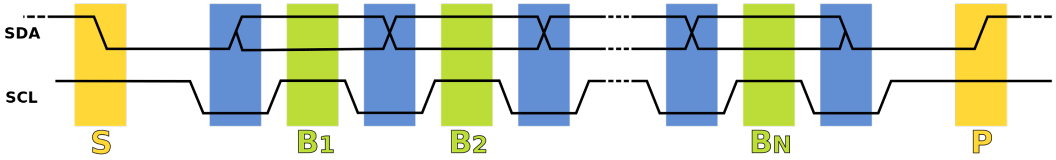
- Aliasing
  - With low sampling frequency aliasing may occur
  - Low pas filter may prevent aliasing, Aliasing filter
- Quantization Error
  - Quantization = Rounding/truncating signal to some resolution
  - Error in red



Buses

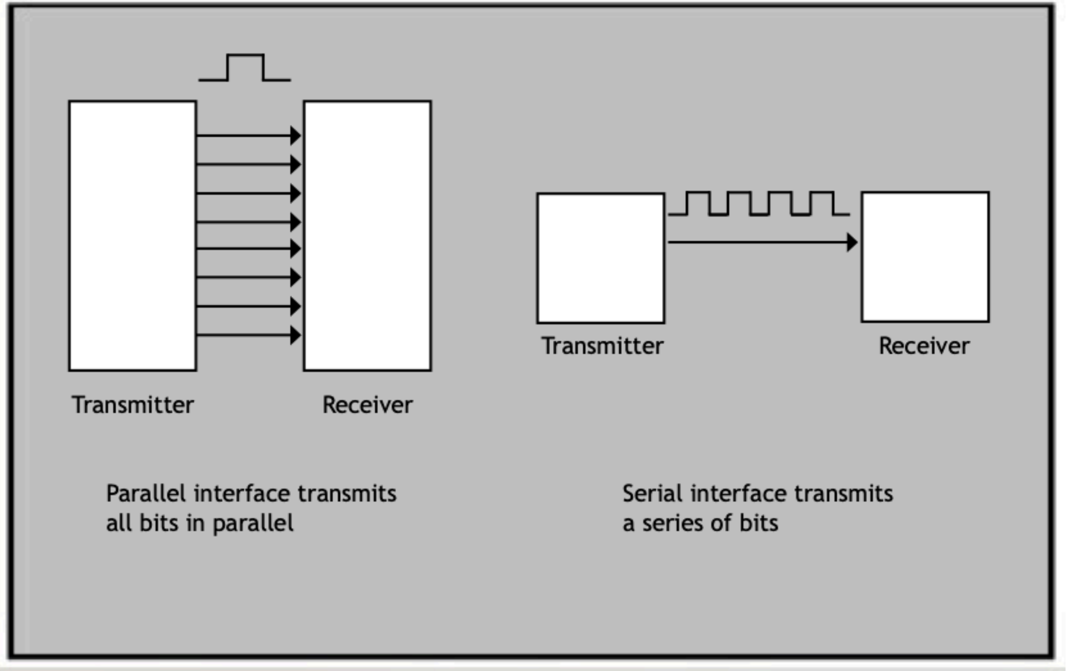
- Needed to communicate to sensors & other peripherals (& other devices)
- Implemented with hardware or software drivers
- Operate according to protocols
- Arduino libraries (Do not bitbang SPI)



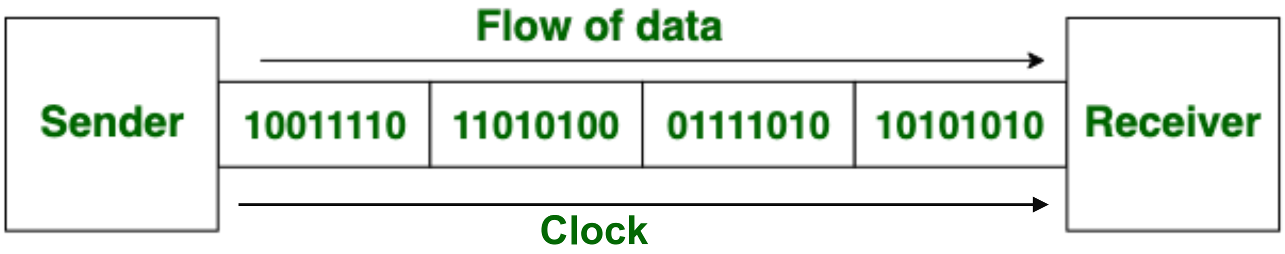


Buses

- Serial vs parallel

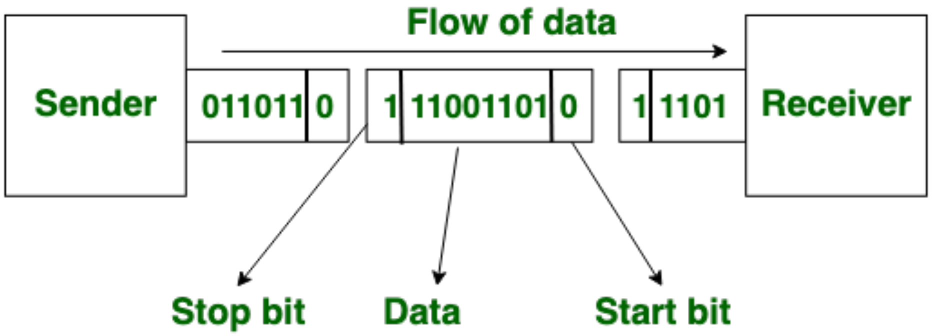


Buses



Synchronous Transmission

- Serial vs parallel
- Synchronous
vs
Asynchronous



Asynchronous Transmission

Buses

PCB Scale

- **SPI, I2C**/DDC/SMBus, I2S, 1-Wire, …

Device Scale

- **UART/serial,** RS232, RS422, RS485, CAN, LIN, **MIDI, USB, …**

Bit-Banging

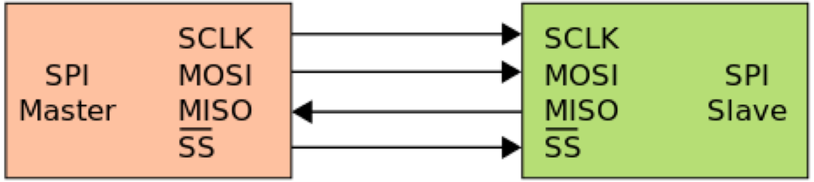
- **If suitable hardware peripheral is not available or free, many interfaces can be emulated with bitbanging.**
    - **Interface behaviour is emulated by swinging GPIO pins to proper state at proper timing, using MCU processing time to control pins**
    - **Essentially interface needs to be considerably slower than MCU/CPU speed**
        - *E.g. earlier versions of Arduino Software serial were able to run at max 19200 bps on 16MHz MCU (although now the library has been optimised for better speed)*
- **Many exotic interfaces need to be bit-banged, in lack of proper hardware peripherals, e.g. 1-Wire**

Bit-Banging SPI in C

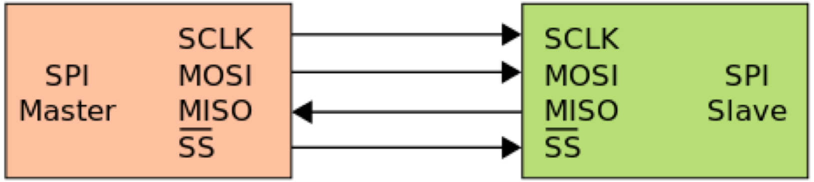
```c
// transmit byte serially, MSB first
void send_8bit_serial_data(unsigned char data)
{
   int i;

   // select device (active low)
   output_low(SD_CS);

   // send bits 7..0
   for (i = 0; i < 8; i++)
   {
       // consider leftmost bit
       // set line high if bit is 1, low if bit is 0
       if (data & 0x80)
           output_high(SD_DI);
       else
           output_low(SD_DI);

       // pulse the clock state to indicate that bit value should be read
       output_low(SD_CLK);
       delay();
       output_high(SD_CLK);

       // shift byte left so next bit will be leftmost
       data <<= 1;
   }

   // deselect device
   output_high(SD_CS);
}
```

SPI

- Serial Peripheral Interface Bus
  - High speed (up to > 10 MB/s), full duplex capable
- Master initiated, simultaneous bidirectional data transfer capable
  - MISO (master in slave out), MOSI (master out slave in), SCK (serial clock), SS/CS (slave / chip select)
- Easy to use with Arduino libraries (SPI Library)



SPI



Advantages of using SPI

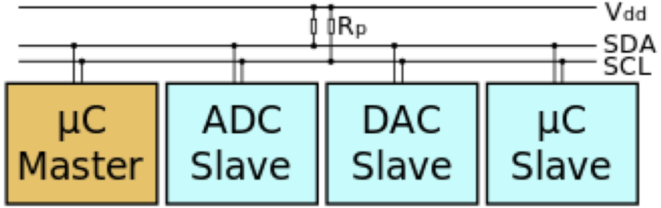
- The protocol is simple as there is no complicated slave addressing system like I2C.
- It is the fastest protocol compared to UART and I2C.
- No start and stop bits unlike UART which means data can be transmitted continuously without interruption
- Separate MISO and MOSI lines which means data can be transmitted and received at the same time

Disadvantages of using SPI

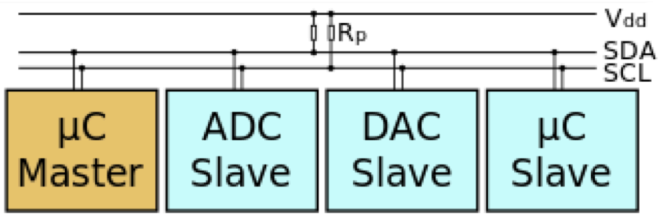
- More Pin ports are occupied, the practical limit to a number of devices.
- There is no flow control specified, and no acknowledgement mechanism confirms whether data is received unlike I2C
- Uses four lines – MOSI, MISO, NCLK, NSS
- No form of error check unlike in UART (using parity bit)
- Only 1 master

I2C

- Inter-Integrated Circuit, Display Data Channel, System Management Bus…
    - Low speed: 400 / 100 kHz usually, but higher speed devices available (>1 MHz)
    - Developed, Patented & Controlled by Philips Semiconductors
- Master initiated, half-duplex
    - SDA (SerialData), SCL (SerialClock)
    - Several devices can share same bus, (each has 7-bit unique address)
    - Devices interface open-collector/open-drain (pull-up resistors)
    - Available at VGA, DVI, HDMI-Connectors
    - Used in PCI, DIMM etc. for identification & configuration
- Easy to use with Arduino (Wire Library)



I2C



Advantages of using I2C

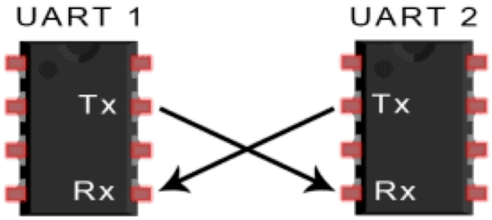
- Has a low pin/signal count even with numerous devices on the bus
- Flexible, as it supports multi-master and multi slave communication.
- Simple as it only uses 2 bidirectional wires to establish communication among multiple devices.
- Adaptable as it can adapt to the needs of various slave devices.
- Support multiple masters.

Disadvantages of using I2C

- Slower speed as it requires pull-up resistors rather than push-pull resistors used by SPI. It also has an open-drain design = limited speed.
- Requires more space as the resistors consume valuable PCB real estate.
- May become complex as the number of devices increases.

UART

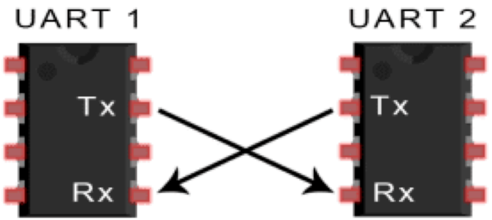
- Serial, Asynchronous, Bidirectional, half-/fullduplec
  - Only GND, TX and RX, no separate clock signal
  - Simple, Easy to use
- Protocol not defined, several standard electrical interfaces
- Usually used for specific peripherals, E.g. Bluetooth transmitters, GPS, GSM
- Arduino library: (Serial, SoftwareSerial)



UART

Advantages of using UART

- Simple to operate, well documented as it is a widely used method with a lot of resources online
- No clock needed
- Parity bit to allow for error checking



Disadvantages of using UART

- Size of the data frame is limited to only 9 bits
- Cannot use multiple master systems and slaves
- Baud rates of each UART must be within 10% of each other to prevent data loss.
- Low speed

Other

- **1-Wire: Low speed single datawire bus by Dallas/Maxim**
  - Several devices can share same data bus
  - E.g. used in DS18x20 digital interfave temperature sensors
  - Arduino library (OneWire)

- **MIPI: Camera & Display Serial interface, HD resolutions**
  - Requires driver to work
  - Found on Raspberry Pi platforms
- **USB: Hard, complicated protocol**
  - Always requires a driver, usually it is easier to use a general Serial-over-USP that emulates traditional serial port (UART)
  - Supplies power, max 500 mA