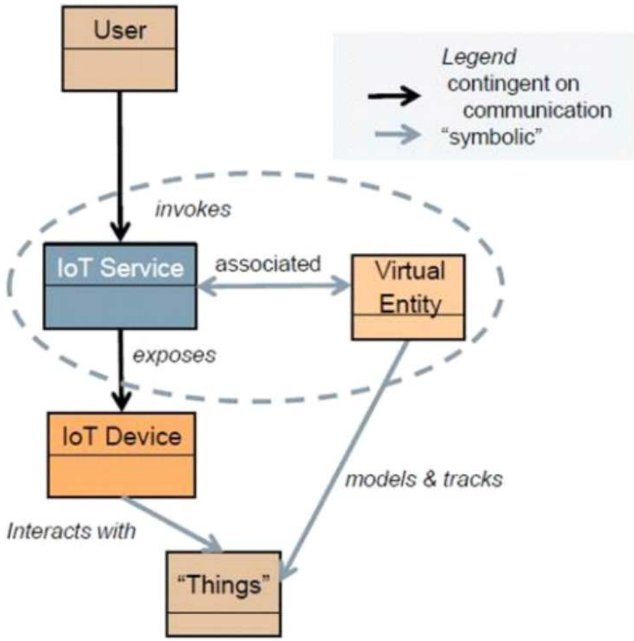
EIOT

Lecture 11: Sensors and actuators

Aleksander Pruszkowski

Instytut Telekomunikacji Politechniki Warszawskiej



THIS LECTURE

- Conversion by generic sensor
- ADC converters
- Sensors
- Actuators

SENSORS

- Conversion by a generic sensor

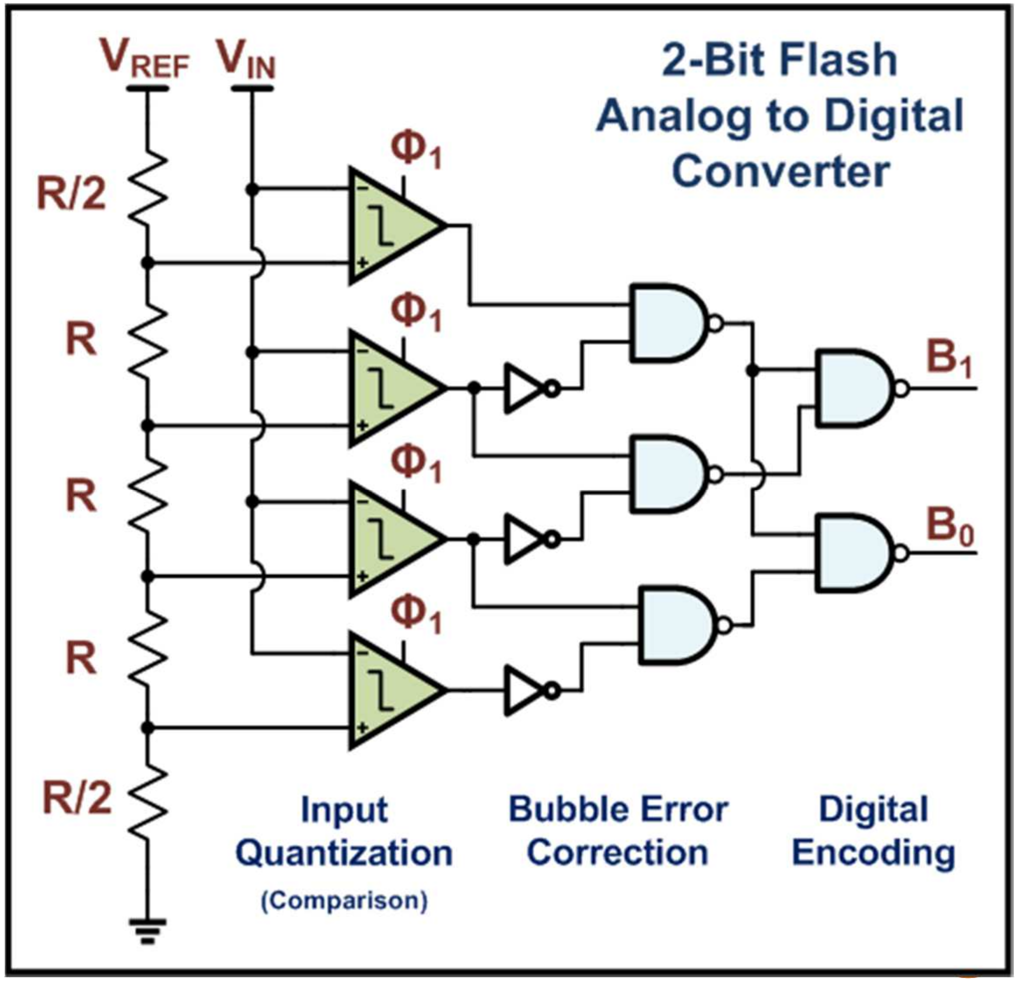
External stimulus

- Convert to electric quantity :
  - voltage
  - current
  - frequency
  - waveform
    - duty cycle
    - slope
  - ...

- Convert from analog to digital:
  - ADC
    - Direct-conversion (flash)
    - Successive approximation
    - ...
  - counters
    - time
    - no. of pulses
    - duty cycle
  - ...

SENSORS

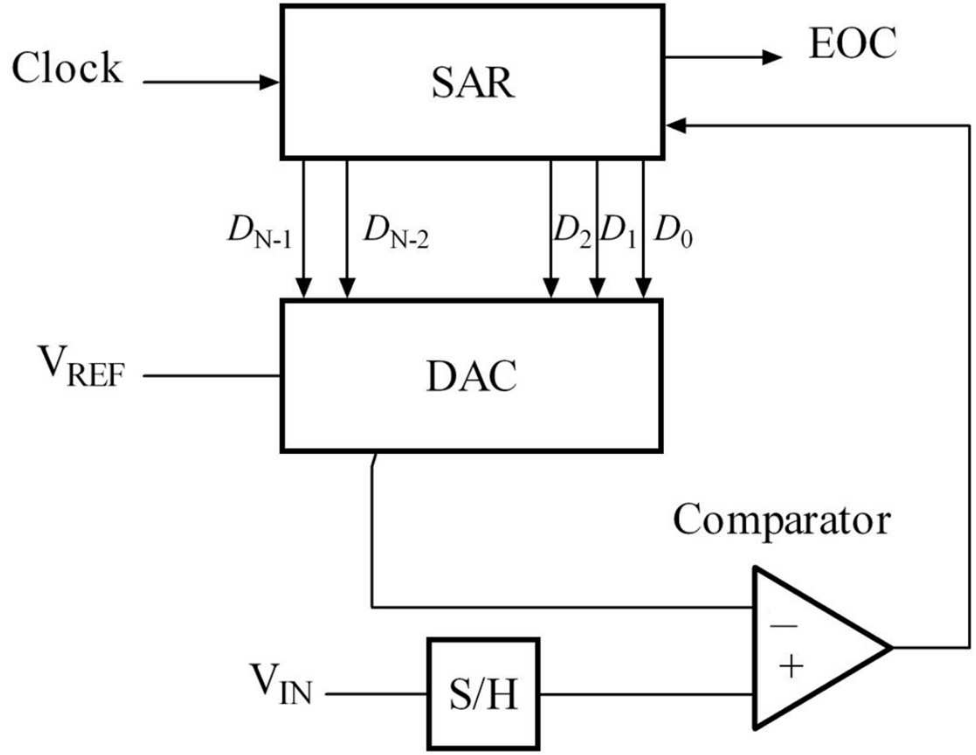
- ADC converters
  - Direct-conversion converter - Flash ADC
  - Pro: speed
  - Con: complexity
    - 2^n-1 comparators in an n-bit converter



Źródło: https://en.wikipedia.org/wiki/Flash_ADC

SENSORS

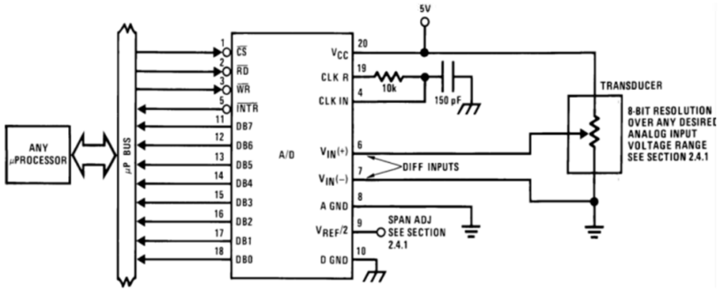
- ADC converters, cont.
  - Successive approximation ADC
    - In each clock cycle, the next bit in SAR (D_{n-1}, then D_{n-2} , ..., then D_o) is changed, so as to make the DAC output voltage equal to Vin
  - Pro: medium speed, low complexity
  - Con: non-linearity

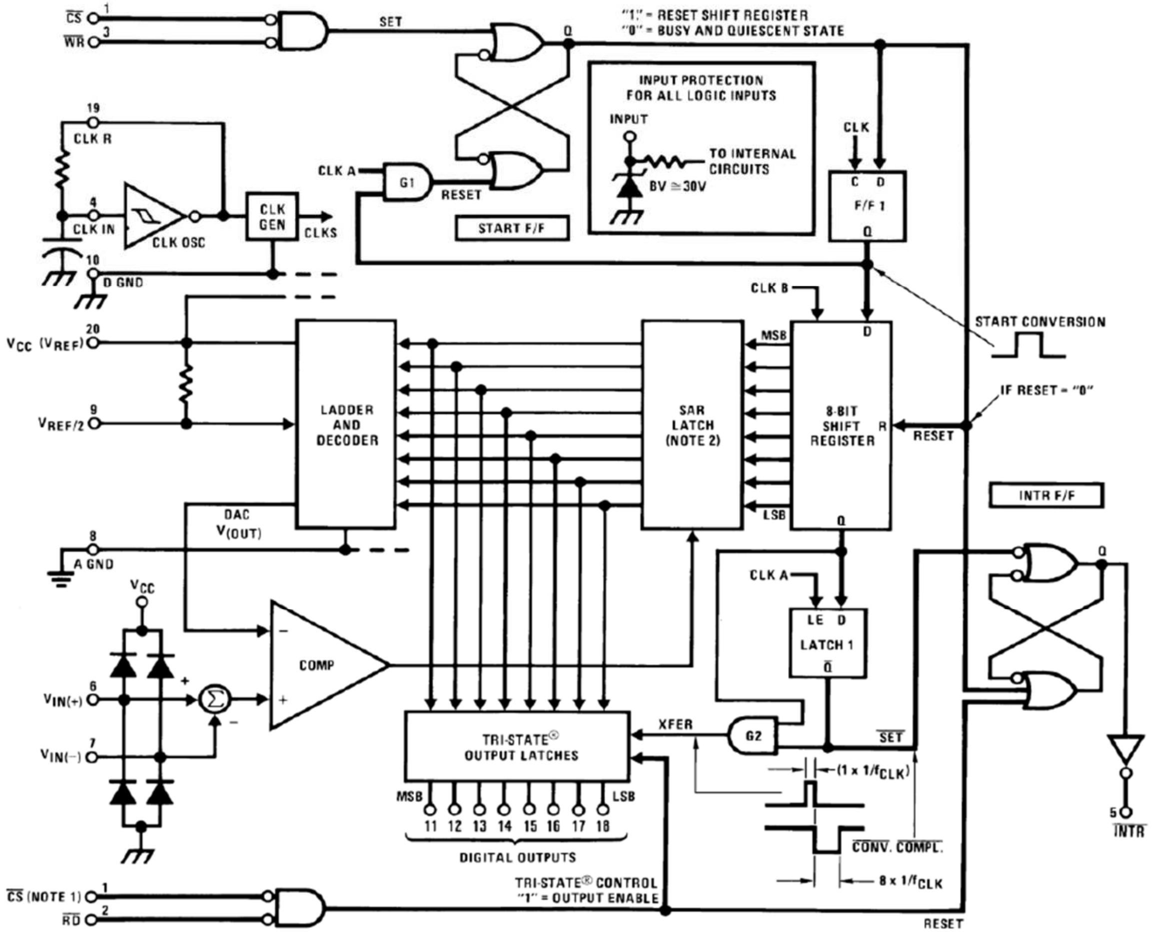


Źródło: https://en.wikipedia.org/wiki/Successive_approximation_ADC

SENSORS

- ADC converter example: **ADC0804**
  - parallel bus interface (compatible with 8080)
  - 8bits, +/- 1LSB
  - conversion time: 100us
  - one channel



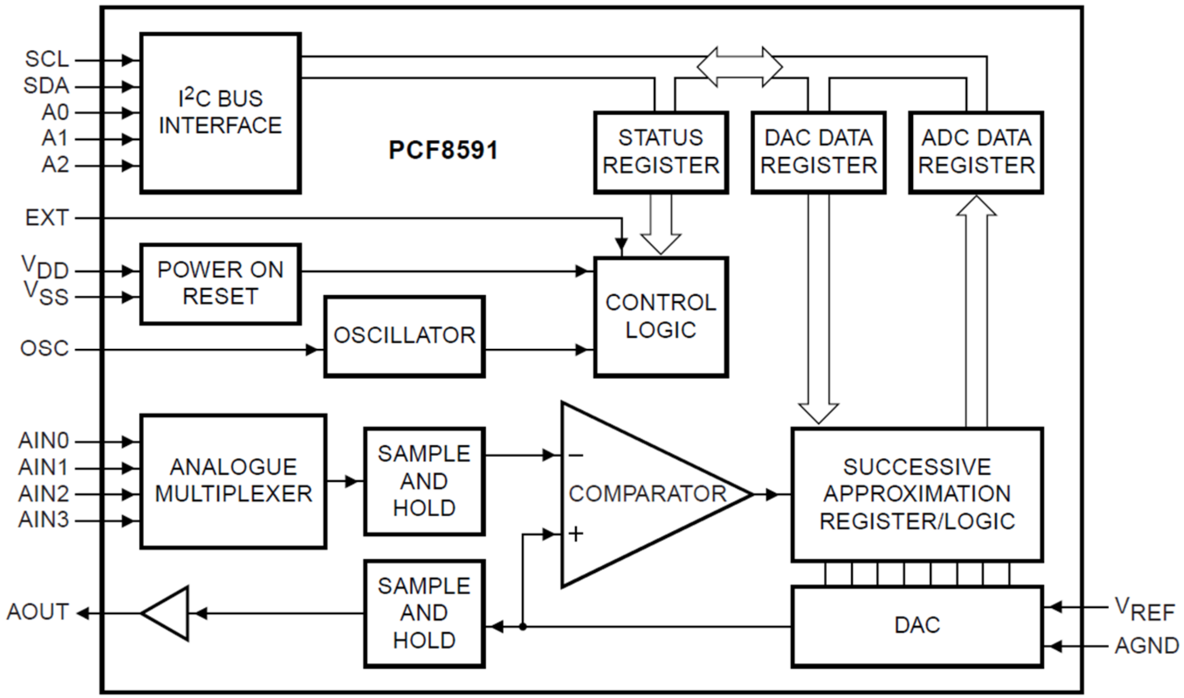


Źródło: https://www.ti.com

SENSORS

- ADC converter example: **PCF8591**
  - simple ADC
  - serial bus interface (I2C)
  - 8 bits, +/- 1.5LSB
  - conversion time 90us
  - four channels
  - internal DAC

    can be used for user-level digital-to-analog conversion



Źródło: https://www.nxp.com

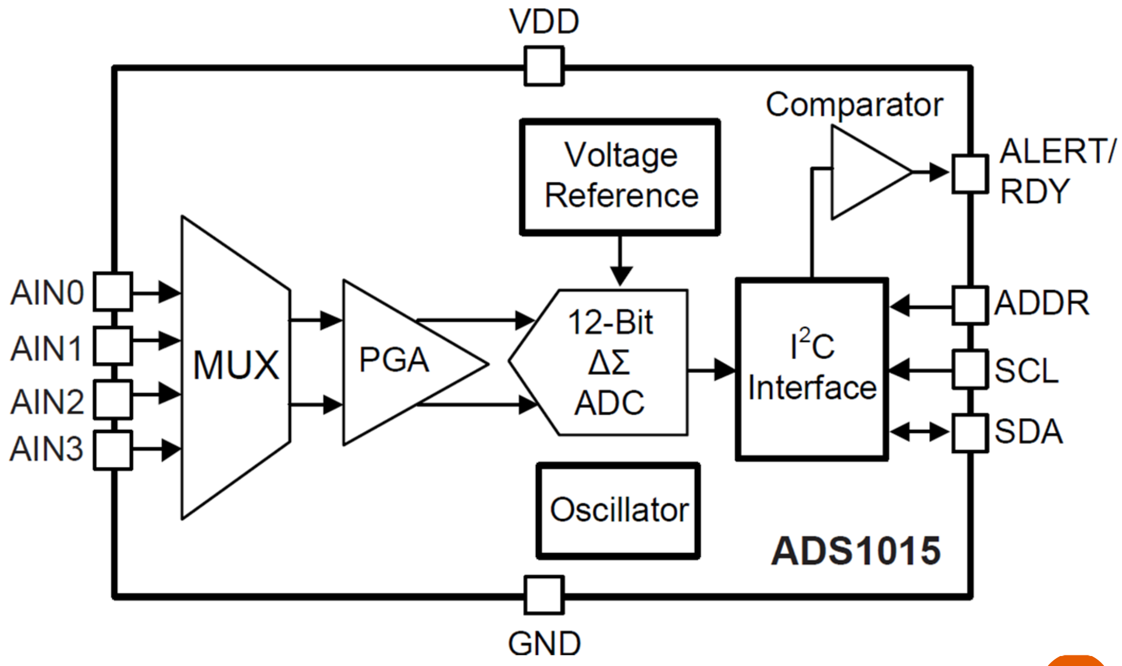
SENSORS

- Handling ADC's in software
  - **PCF8591** + Arduino API (source: tronixstuff.com)

```
#include "Wire.h"
#define PCF8591 (0x90 >> 1)
Wire.begin();

Wire.beginTransmission(PCF8591);
Wire.write(0x04);          //conf.: read ADC0 then auto-increment
Wire.endTransmission();
Wire.requestFrom(PCF8591, 4);
value0=Wire.read();        //input no. 1
value1=Wire.read();        //input no. 2
value2=Wire.read();        //input no. 3
value3=Wire.read();        //input no. 4
```

SENSORS

- ADC converter example: **ADS1015**
  - advanced ADC
  - serial bus interface (I2C)
  - 12 bits, +/- 0,5LSB
  - conversion time: 303us
  - four channels
  - features an amplifier (PGA) and comparator



Źródło: https://www.ti.com

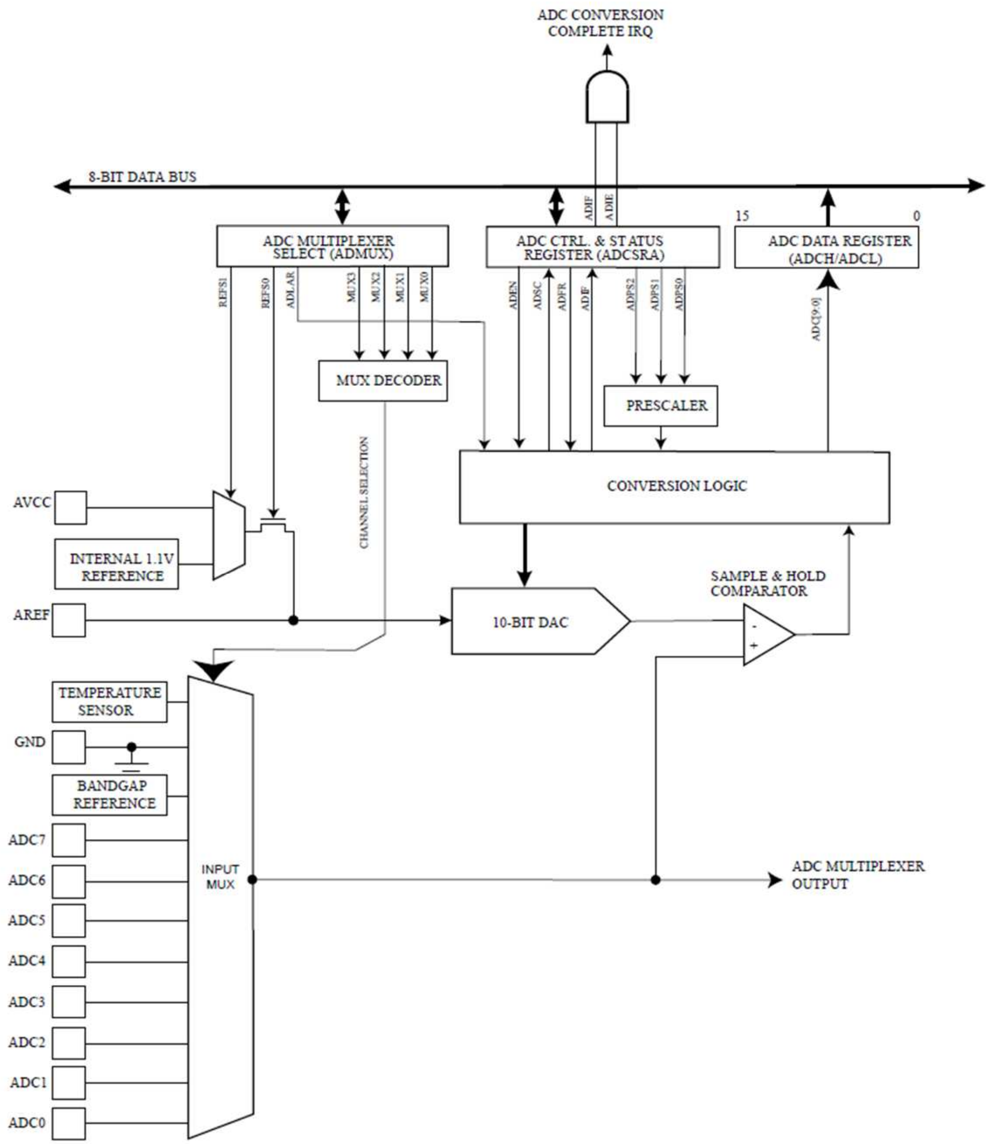
SENSORS

- Handling ADC's in software
  - **ADS1015** + Arduino API (Adafruit library)

```
#include <Adafruit_ADS1015.h>
Adafruit_ADS1015 ads;
ads.setGain(GAIN_ONE);
ads.begin();

adc0=ads.readADC_SingleEnded(0);
```

SENSORS

- ADC converter example: ADC in **ATMega328P** MCU
  - 10 bits, +/- 2LSB
  - conversion time: 13us
  - six externals channels
  - reference voltage:
    - internal or external (for bigger accuracy)
    - integrated with MCU interrupts



Źródło: https://www.atmel.com

SENSORS

- Handling ADC's in software
  - ADC in **ATMega328P** (raw usage) + Arduino API

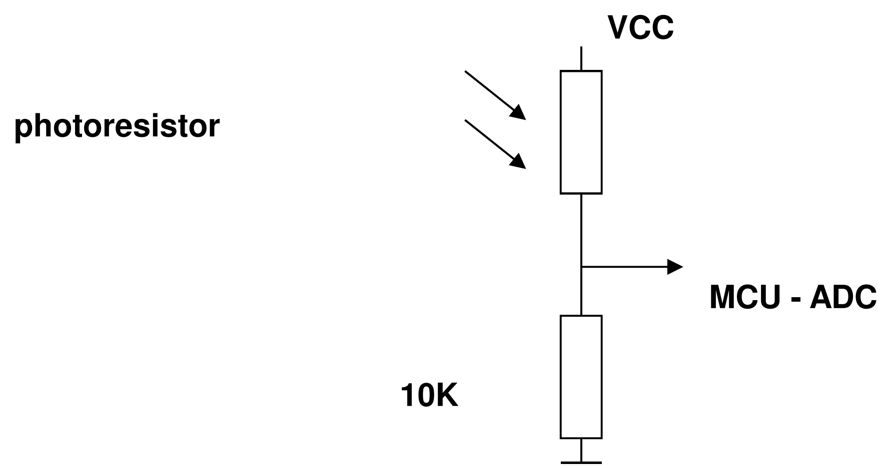
```
sensorValue=analogRead(A0);
```

SENSORS

- Sensing physical quantities: light
  - photoelectric cell (of historical importance)
    - change of the current flowing through an electron tube (in vacuum)
  - photoresistor (LDR)
    - change of the resistance (pro: simple, con: limited sensitivity)
  - photodiode, phototransistor
    - change of the current flowing through a silicon junction (within a single silicon structure, one can amplify, filter, and do preliminary processing)
  - sensors based on CCD/CMOS matrices
    - a general approach (the sensing element can additionally detect colors, shapes, etc.)

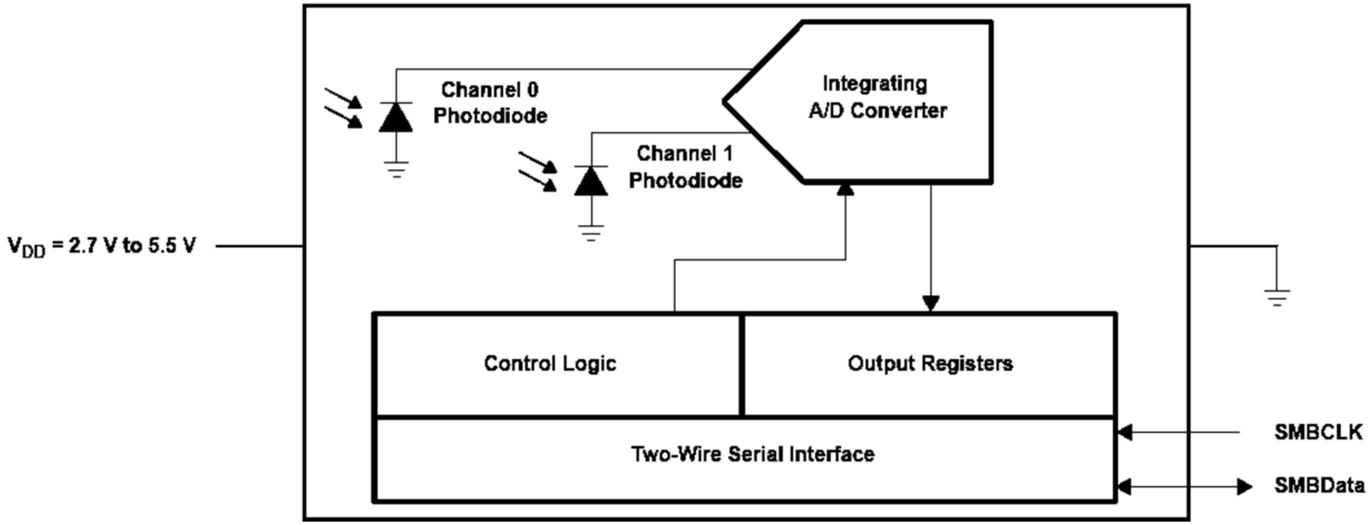
SENSORS

- Photoresistor example: Clairex **CL94L**
  - photosensitive element based on CdSe 690nm
  - range of resistance: full lighting – 2K, in the dark – 520K
  - peculiarity: at low lighting levels, the resistance may rise with time (reaction time)



SENSORS

- Photodiode example: TAOS **TSL2561**
  - intelligent photodiode
  - 400-1000nm
  - internal logic converts light into a digital format (16-bit ADC)
  - the module supports SMB/I2C/TWI interfaces
  - supports commands: read ADC 0|1 /.../ power down / power up



SENSORS

- Handling light sensors in software
  - **TSL2561** (Adafruit library for Arduino)

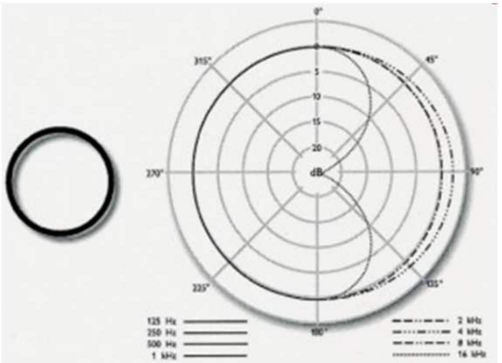
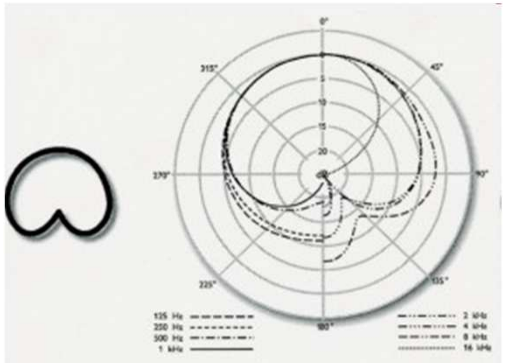
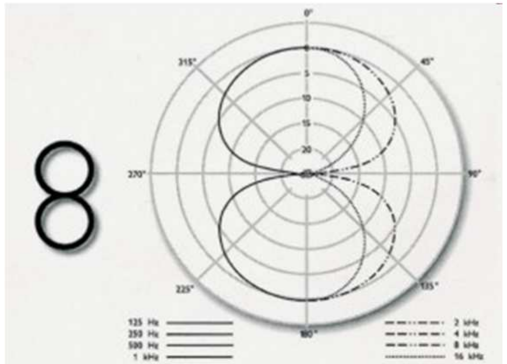
```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2561_U.h>
const int address=TSL2561_ADDR_FLOAT;           //real adr.: 0x39
Adafruit_TSL2561_Unified tsl =
                         Adafruit_TSL2561_Unified(address, 12345);
tsl.setGain(TSL2561_GAIN_16X);        //internal amplification 16x
tsl.setIntegrationTime(TSL2561_INTEGRATIONTIME_402MS);    //402ms

sensors_event_t event;
tsl.getEvent(&event);
if (event.light){
   Serial.Println(event.light);Serial.Println(" lux");
}
```

SENSORS

- Sensing physical quantities: sound
  - microphone
    - converts sound , i.e., changes of air pressure, into changes of an electrical signal
  - microphone making technologies
    - carbon microphone: the membrane presses carbon granules, which changes resistance
    - cons: poor dynamic range, noise, “crackling”, ...
    - electret microphone: the membrane changes the capacitance of a capacitor
      - pro: no needed polarity voltage, con: requires an external amplifier
    - piezoelectric microphone: a change of the piezoelectric membrane’s shape produces an electrical signal
    - dynamic microphone: the membrane’s movement in constant magnetic field produces electrical current (electromagnetic induction)
      - con: complexity, pro: best results
    - MEMS microphone: made as an integrated circuit
      - pro: integrated amplifiers, filters, ADC converters

SENSORS

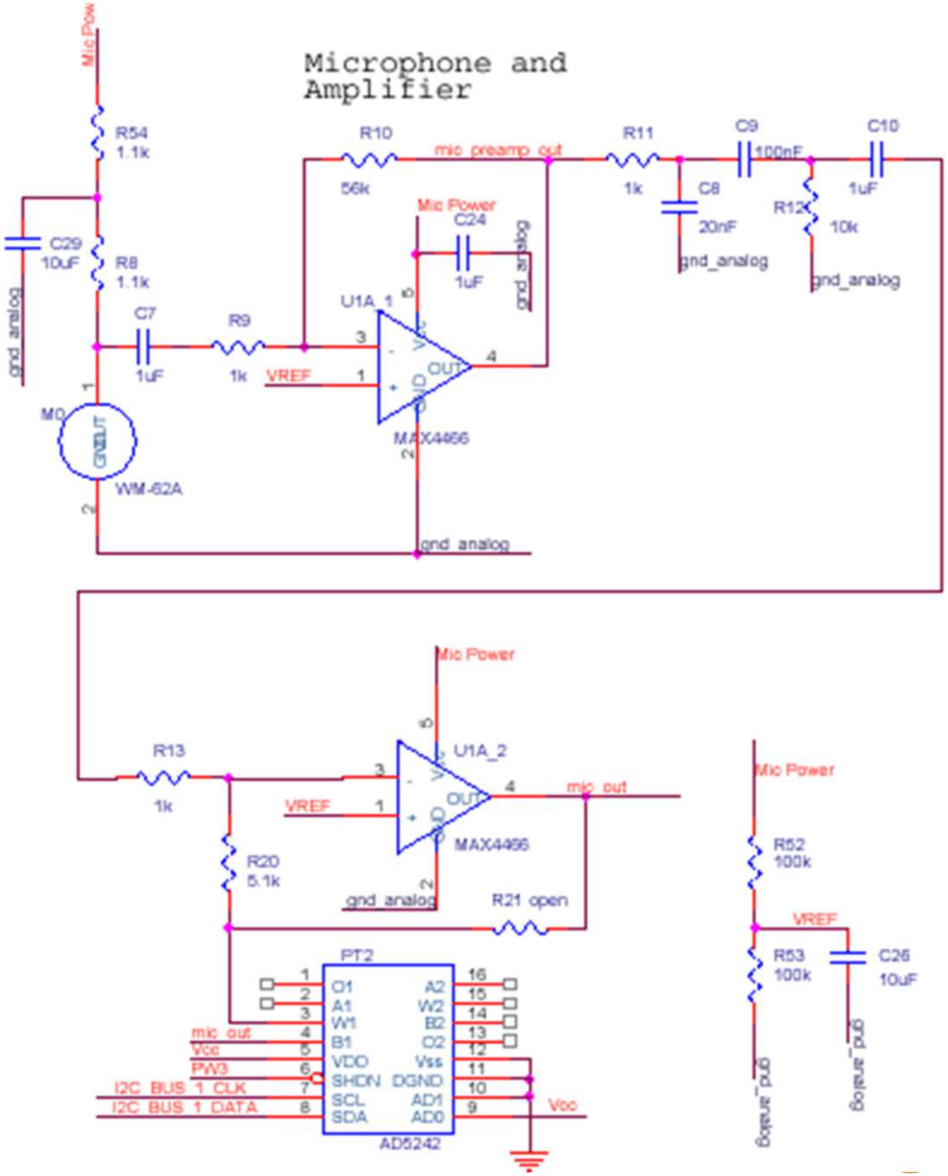
- Sensing physical quantities: sound, cont.
  - microphone polar patterns
    - sensitivity to sound coming from different directions: omnidirectional, cardioid, bi-directional or figure of digit 8

Źródło: http://www.altra.net.pl

SENSORS

- Microphone example
  - converter **WM-62A**
  - low-noise operational amplifier MAX4466 (preliminary amplification and filtering)
  - ADC converter ADC5242
  - I²C bus to connect to MCU



Źródło: www.xbow.com

SENSORS

- Sensing physical quantities: temperature
  - Bi-metal
    - a mechanical element consisting of two layers made of different metals
    - change of temperature affects the shape of the element (due to different thermal expansion of the metals)
    - used almost exclusively in thermostats
  - Thermocouple
    - made of two different electrical conductors, made of alloys: nickel+chromium and nickel+aluminum
    - voltage at the junction: 40uV per 1^{o}C
  - Thermistor
    - made of material that changes its resistance as a result of temperature change
    - may have Positive Temperature Coefficient (PTC) or Negative Temperature Coefficient (NTC)

SENSORS

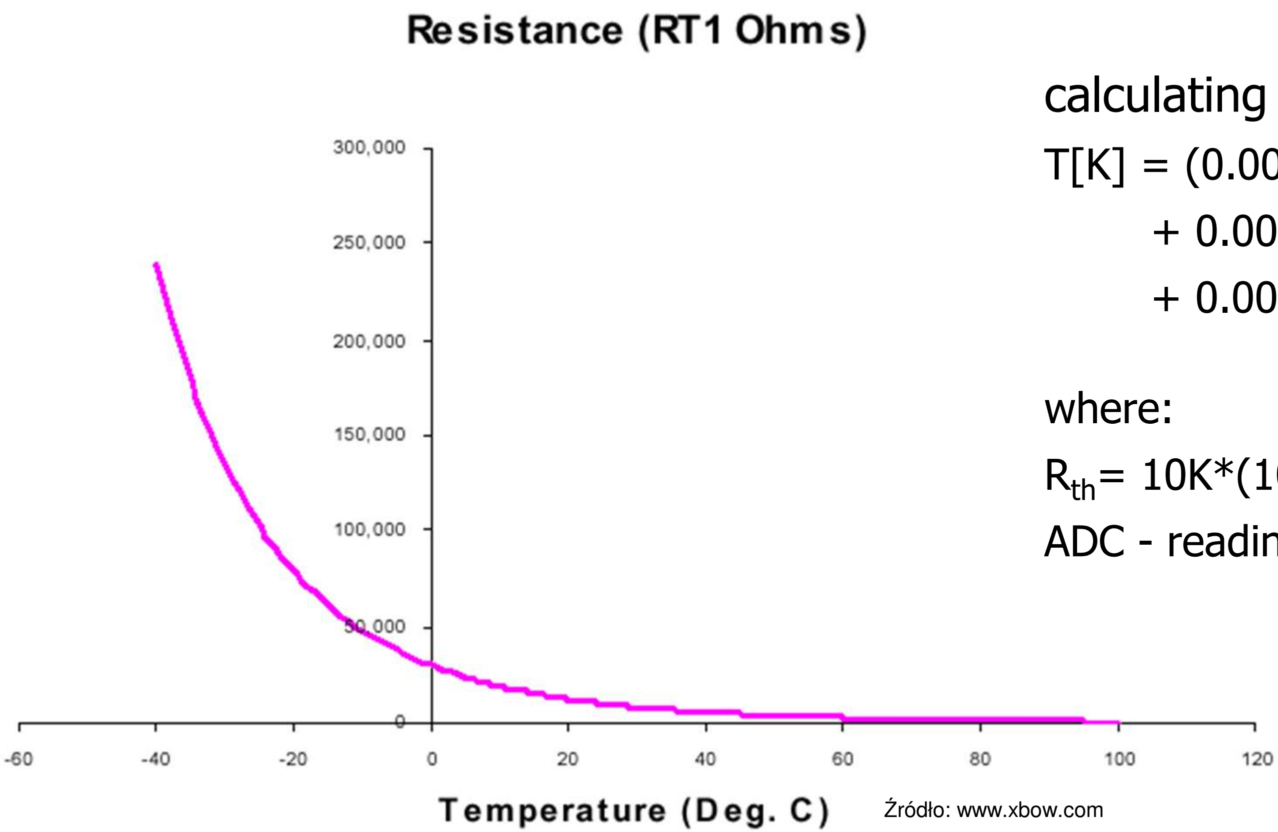
- Sensing physical quantities: temperature , cont.
  - Semiconductor temperature sensors
    - temperature change at the semiconductor junction leads the voltage change: +2mV per +1^{o}C
    - can be integrated, within one semiconductor structure, with voltage processing circuitry
      - con: circuit needs temperature compensation for internal voltage sources of reference levels

SENSORS

- Sensing physical quantities: temperature, cont.
  - Problems with temperature sensors
    - zero calibration
      - exists both at manufacturing time and while in use
    - aging
      - need to calibrate periodically
    - non-linearity
      - to some extent may be compensates via digital processing
    - impact of sensor electronics on the temperature of the sensing element itself

SENSORS

- Temperature sensor example: **YSI 44006** thermistor
  - resistance at 25°C: 10K
  - highly nonlinear
  - accuracy after calibration: 0.2°C



Źródło: www.xbow.com

calculating temperature

$$T[K] = (0.001010024 + 0.000242127 * \ln(R_{th}) + 0.000000146 * [\ln(R_{th})]^3)^{-1}$$

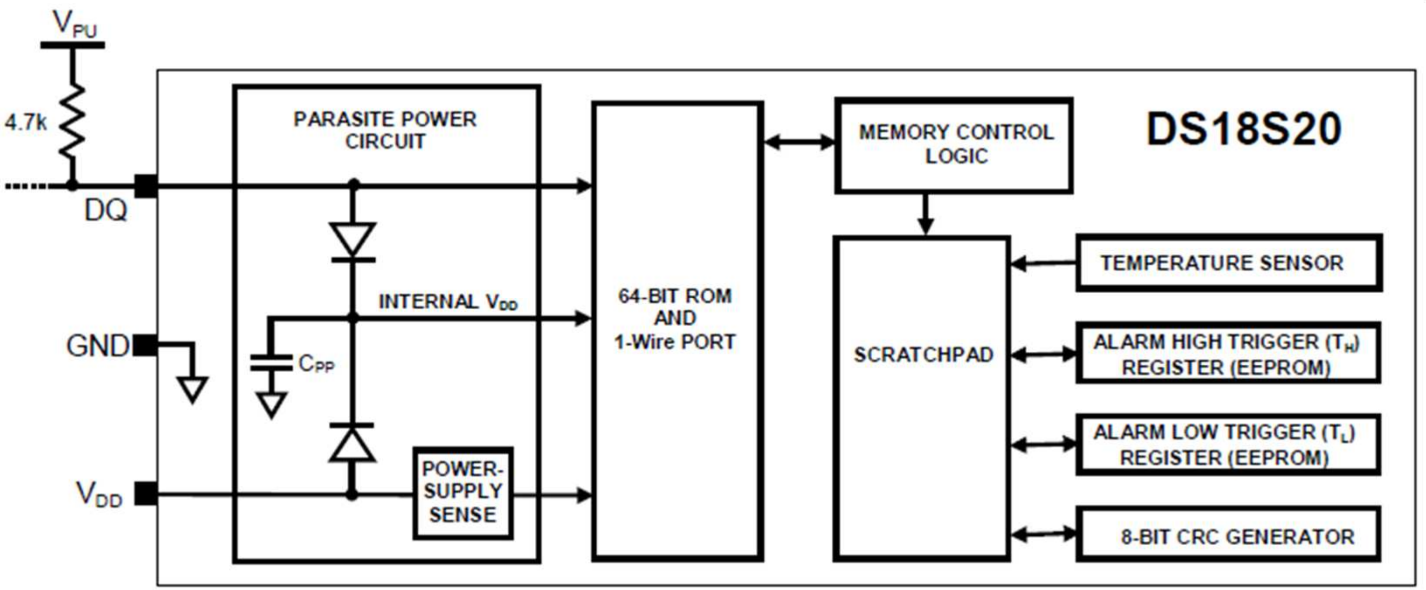
where:

$R_{th} = 10K*(1023\text{-}ADC)/ADC$

ADC - reading from A/C converter

SENSORS

- Temperature sensor example: **DS18S20**
  - semiconductor thermometer with digital output,
  - 9-bit, accuracy: ±0.5°C
  - 1-Wire interface (developed by the company named Dallas)
  - temperature range: -55°C to +125°C,
  - each piece has its own unique id(!)



Źródło: www.maxim-ic.com

SENSORS

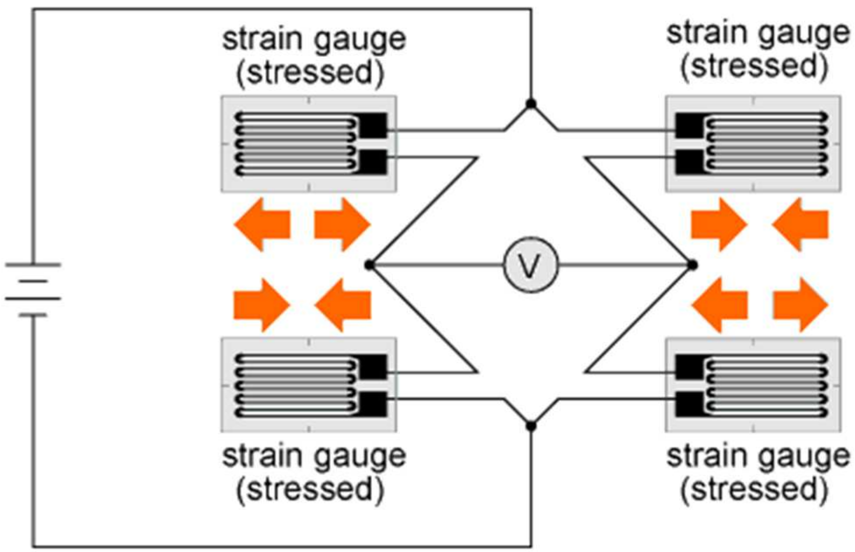
- Handling temperature sensors in software
  - **DS18S20** (PJRC library for Arduino)

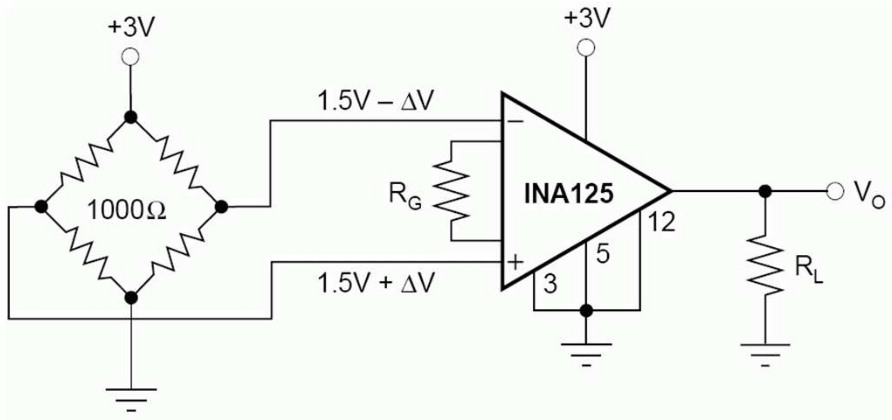
```
#include <OneWire.h>
OneWire ds(2);
byte data[12],addr[8];
if(ds.search(addr)){                              //are there any devices
  if(OneWire::crc8(addr,7)==addr[7]){      //is ”CRC” OK.?
     ds.reset();
     ds.select(addr);
     ds.write(0x44, 1);                       //start converting
     delay(1000);
     ds.reset();
     ds.select(addr);
     ds.write(0xBE);             // read from memory of DS18B20(9B)
     for(i=0;i<9;i++)
       data[i] = ds.read();
     Serial.print((float)((data[1] << 8) | data[0]) / 16.0);
  }
}
```

SENSORS

- Sensing physical quantities: strain
  - strain gauge
    - change of size or shape causes resistance change
    - popular in electronic weigh scales

Full-bridge strain gauge circuit



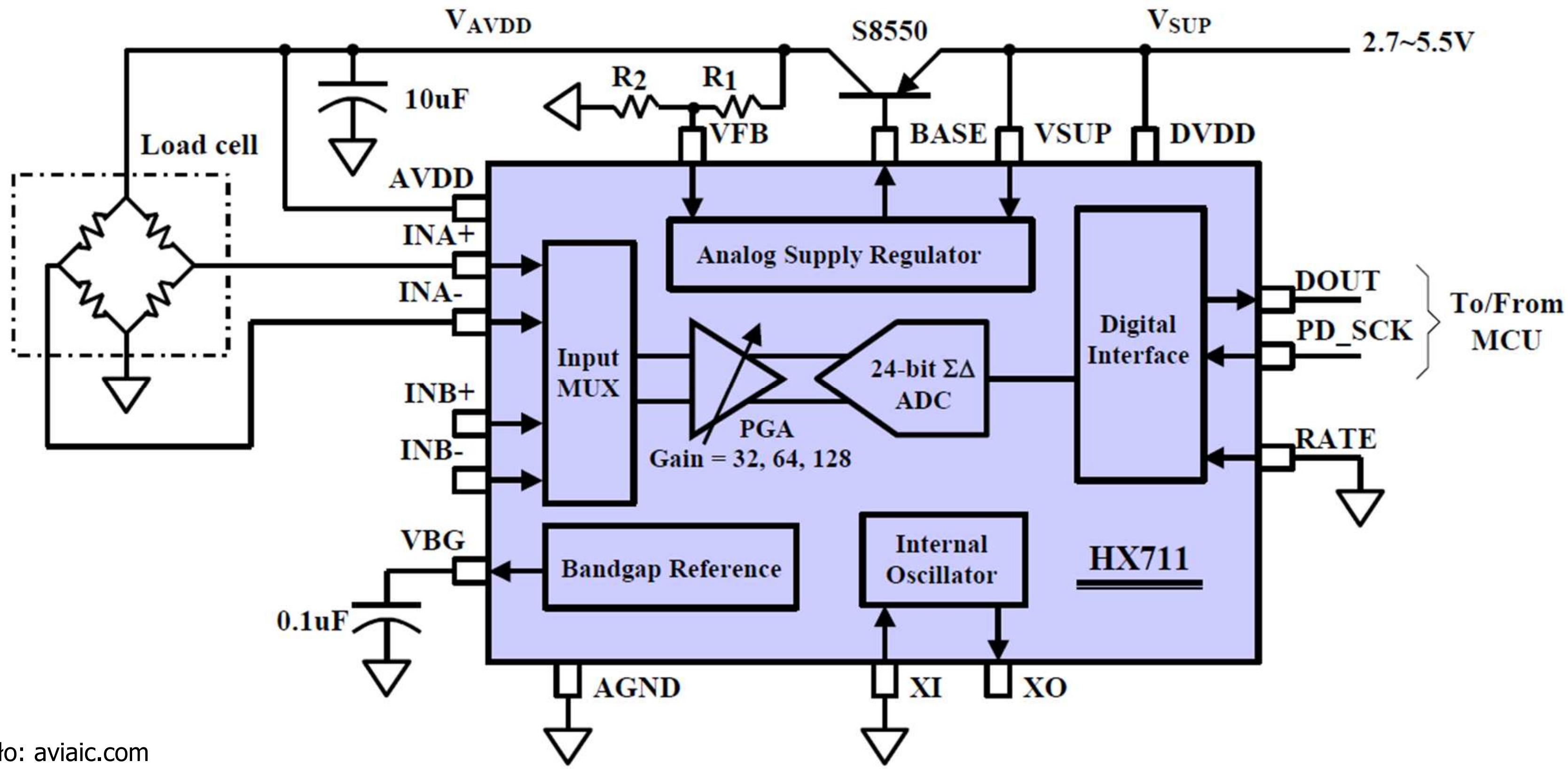


Źródło: http://cerulean.dk/words/?page_id=42

Źródło: ti.com

SENSORS

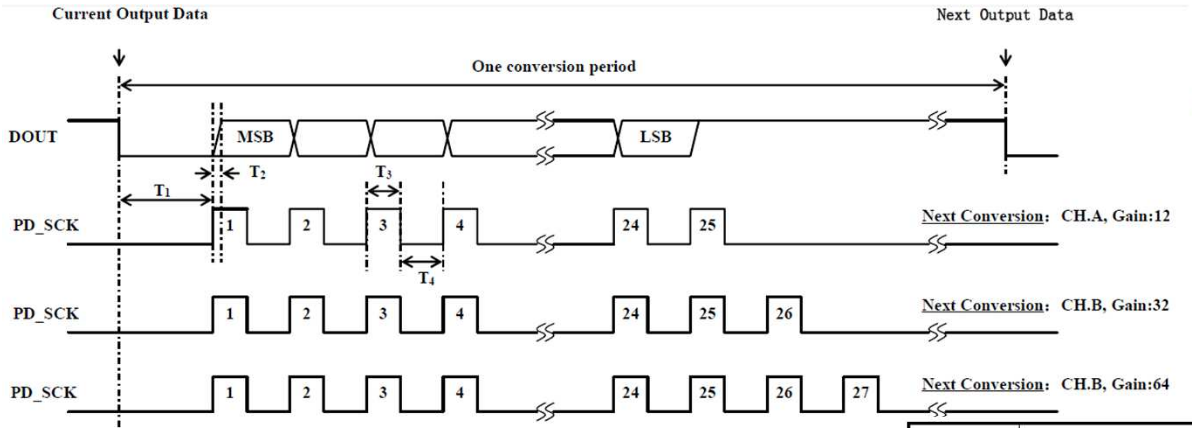
- Strain gauge example: **HX711**
  - instrumentation amplifier, power supply, and ADC converter in a single package

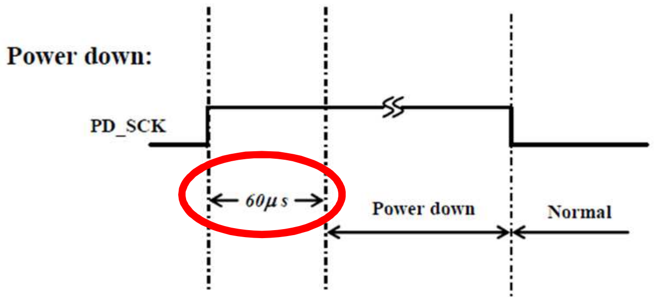


Źródło: aviaic.com

SENSORS

- Strain gauge example: **HX711**, cont.
  - practical tip: the timing of HX711 auto-power down mode makes it impossible to correctly communicate with it through the GPIO supported in Linux
    - from HX711's PDF: *„...when PD_SCK pin changes from low to high and stays at high for longer than 60us, HX711 enters power down mode...”*





Symbol	Note	MIN	TYP	MAX	Unit
T_1	DOUT falling edge to PD_SCK rising edge	0.1			µs
T_2	PD_SCK rising edge to DOUT data ready			0.1	µs
T_3	PD_SCK high time	0.2	1	50	µs
T_4	PD_SCK low time	0.2	1		µs

Źródło: aviaic.com

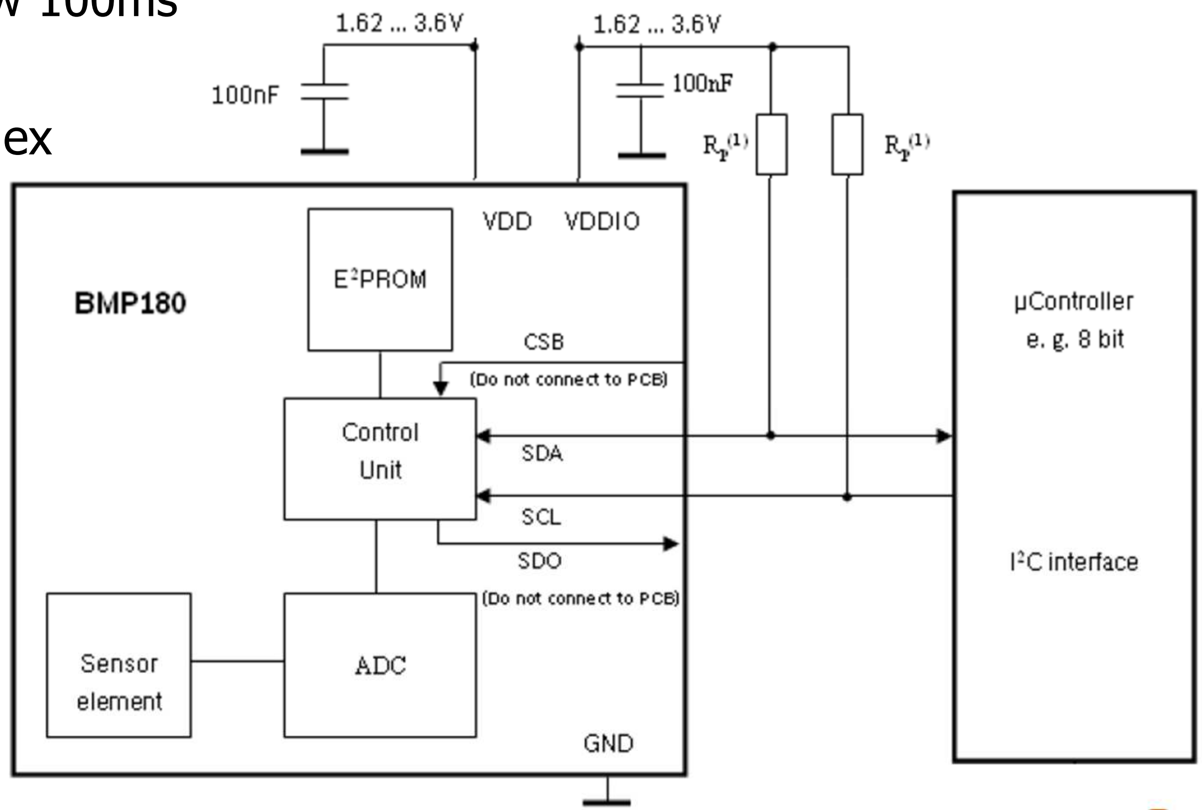
SENSORS

- Handling strain gauges in software
  - **HX711** (SparkFun library for Arduino)

```
#include "HX711.h”
#define cal_factor -7050.0     //HX711 treats the zero mass as ...
#define LBS2KG         0.45359237       //”funt to Kg” conv.
  const

HX711 scale(3, 2);                                       //DOUT=D3, CLK=D2
scale.set_scale(cal_factor);
scale.tare();                                            //zero calibration

Serial.print(scale.get_units()/LBS2KG);
Serial.println(”Kg”);
```

SENSORS

- Sensing physical quantities: pressure
  - pressure sensor example: **BMP180**
  - targets home applications
  - drawing less than 50uA at 3.3V
  - conversion time below 100ms
  - I^{2}C interface
  - requires rather complex calculations



Źródło: www.bosch-sensortec.com

SENSORS

- Pressure sensor example: **BMP180**

  Before your start sensing, you should read correlation coefficients embedded into BMP180 at manufacturing time:

  AC1, AC2, AC3, AC4, AC5, AC6

  VB1, VB2,

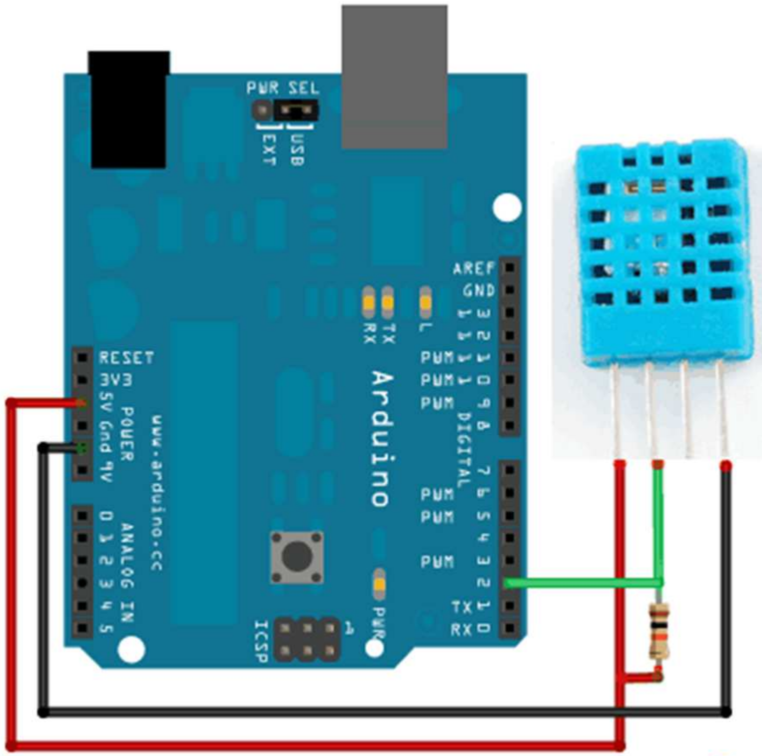
  MB, MC, MD

```
c3 = 160.0 * pow(2,-15) * (double)AC3;
c4 = pow(10,-3) * pow(2,-15) * AC4;
b1 = pow(160,2) * pow(2,-30) * VB1;
c5 = (pow(2,-15) / 160) * AC5;
c6 = (double)AC6;
mc = (pow(2,11) / pow(160,2)) * MC;
md = MD / 160.0;
x0 = AC1;
x1 = 160.0 * pow(2,-13) * AC2;
x2 = pow(160,2) * pow(2,-25) * VB2;
y0 = c4 * 32768.0;
y1 = c4 * c3;
y2 = c4 * b1;
p0 = (3791.0 - 8.0) / 1600.0;
p1 = 1.0 - 7357.0 * pow(2,-20);
p2 = 3038.0 * 100.0 * pow(2,-36);

tu=...;   //raw data read from the sensor
pu=...;   //raw data read from the sensor

a = c5 * (tu - c6);

x = (x2 * pow(s,2.0)) + (x1 * s) + x0;
y = (y2 * pow(s,2.0)) + (y1 * s) + y0;
z = (pu - x) / y;

P = (p2 * pow(z,2.0)) + (p1 * z) + p0;
T = a + (mc / (a + md));
```

Źródło: http://wmrx00.sourceforge.net/Arduino/BMP085-Calcs.pdf

SENSORS

- Sensing physical quantities: humidity
  - humidity sensor example: **DHT22** (correct name: **AM2302**)
  - good for home applications (low cost)
  - "almost 1-Wire", wire length up to 100m(!)
  - power supply voltage in the range of 3.3V...5V
  - draws less than 2.5mA
  - humidity range: 0...100%RH
  - temperature range: -40...+125^{o}C
  - resolution: 16bits (!?)
  - accuracy: +/- 0.5%
  - sensing time: 2s



Źródło: https://learn.adafruit.com

SENSORS

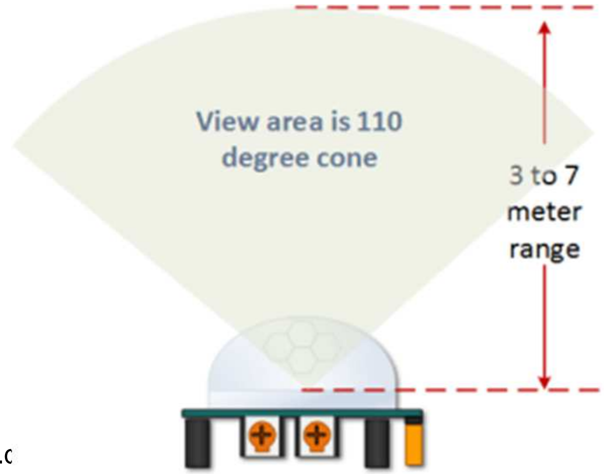
- Handling humidity sensors in software
  - **DHT22** (Adafruit library for Arduino)

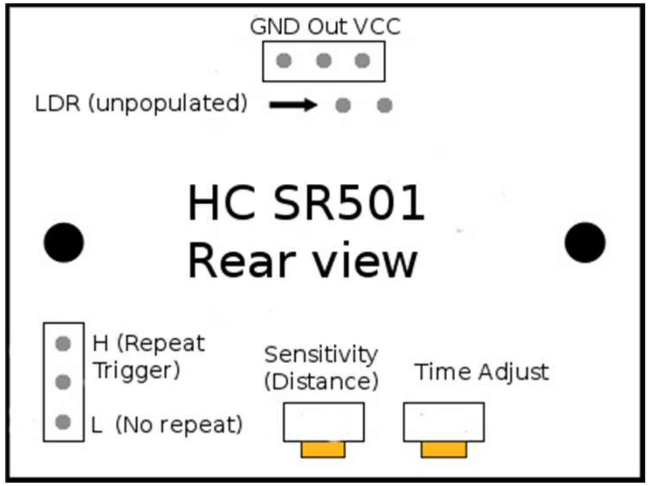
```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>
DHT_Unified dht(D2, DHT22);
dht.begin();

sensors_event_t event;
dht.temperature().getEvent(&event);
if( ! isnan(event.temperature)){
   Serial.print(event.temperature);Serial.println("C");
}
dht.humidity().getEvent(&event);
if( ! isnan(event.relative_humidity)){
   Serial.print(event.relative_humidity);Serial.println("%");
}
```

SENSORS

- Sensing physical quantities: PIR
  - consists of an infrared detector (pyroelectric material) and a Fresnel lens
  - example: **HC-SR501** module
    - additional light sensor (LDR) helps in auto-calibration
    - potentiometers allow one to set the sensing distance and the duration of an alarm
  - working modes
    - single trigger mode: when triggered, generates an alarm that lasts for the "delay time" (an adjustable parameter)
    - repeatable trigger: when triggered, generates an alarm that lasts for as long as there is some activity

Źródło: http://henrysbench.capnfatz.c



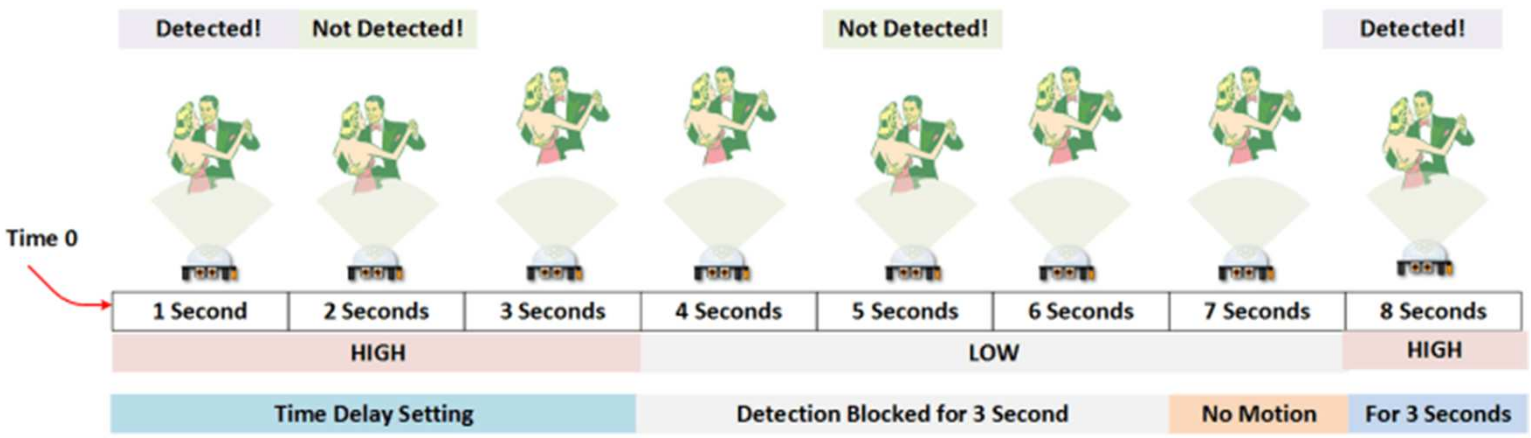


SENSORS

- Handling PIR sensors in software
  - HC-SR501 (Arduino API)

```
const int pirPin = 7;
pinMode(pirPin, INPUT);
Serial.println(digitalRead(pirPin));
```

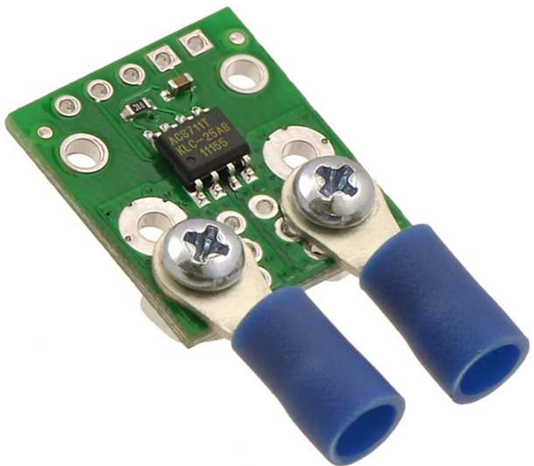
- How the PIR sensor really works:
  - settings: „single trigger mode”, delay time: 3 s



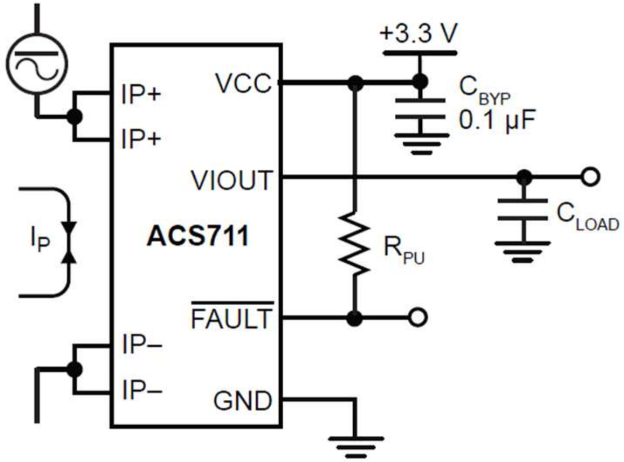
Źródło: http://henrysbench.capnfatz.com

SENSORS

- Sensing physical quantities: current
  - Measurement of supply current, example: **ACS711**
  - Internal measuring resistor (R) helps convert the current I_p to voltage (analog) proportional to I_p
  - range: +/- 12.5A
  - power supply voltage: 3V..5V (<5.5mA)



Źródło: https://www.pololu.com



Źródło: www.allegromicro.com

VIOUT is the voltage proportional to I_p

3,3V/2 - the current equals to 0A (ACSoff)

<3,3V/2 - “negative” current
>3,3V/2 - “positive” current

SENSORS

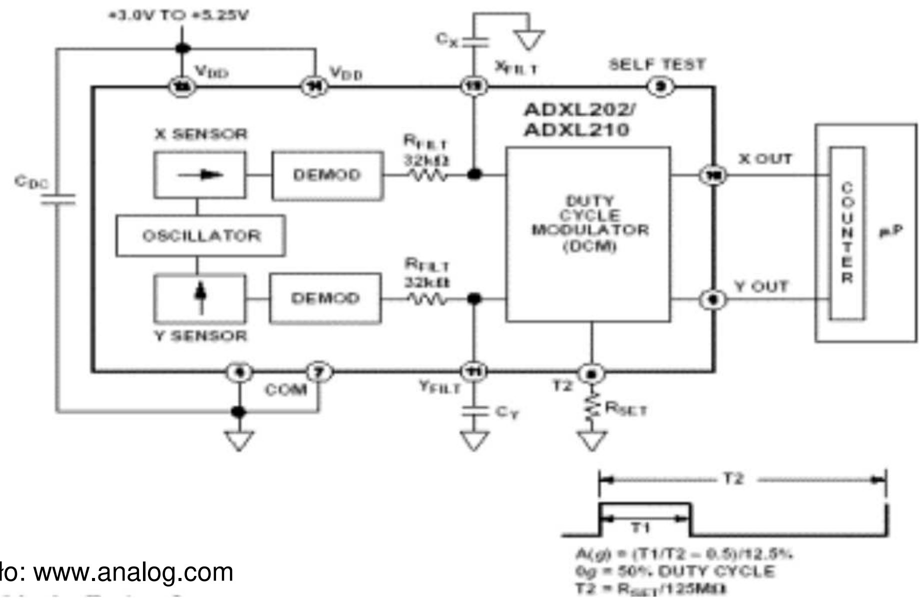
- Handling current sensors in software
  - **ACS711** (Arduino API)

```
const int analogIn=A0;
const int mVperAmp=185;      //const. for 20A Module (related to „R”)
const int Vref=3300;//ADC works with 3.3V external reference voltage
                    //(e.g. On Arduino Uno board the 3.3V voltage is
                    //present on one of the connectors)
const int ACSoff=Vref/2;
const int ADC_levels=1024;
int RawValue=0;
double Voltage=0;
analogReference(EXTERNAL);                              //3.3V VREF!

RawValue=analogRead(analogIn);
Voltage=((double)RawValue/(double)ADC_levels)*(double)Vref;
Serial.print((Voltage-ACSoff)/mVperAmp);Serial.println("A");
```

Źródło: http://henrysbench.capnfatz.com

SENSORS

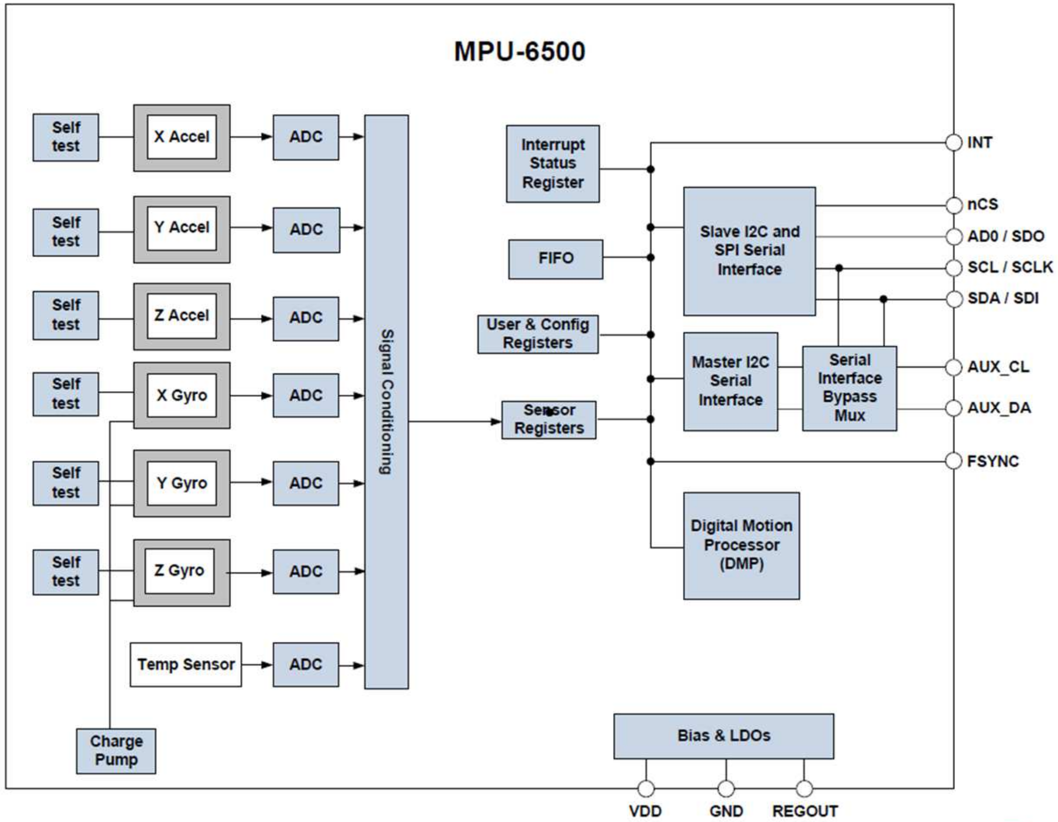
- Sensing physical quantities: motion
  - MEMS acceleration sensors
    - an integrated circuit features two sets of electrodes forming a capacitor
      - made in silicon, comb-shaped
      - one set of electrodes is attached to a frame and motionless
      - the other set moves inertly according to the movements of the element
    - motion/acceleration causes a change of the capacitor's capacitance
    - a number of varieties:
      - 2D, 3D, ..., 6D, ...
      - analog and digital
      - sensors detecting free fall
      - may include loggers and filters
    - example: **ADXL202** (2D)
      - output: PWM
      - acceleration range: +/-2g



Źródło: www.analog.com

SENSORS

- Sensing physical quantities: motion, cont.
  - example: **MPU6500/MPU6050** (6D)
  - accelerometer and gyroscope in a single package
  - supports I^2C i SPI buses
  - power supply currents: 0.5mA (gyroscope) and 3.2mA (accelerometer)
  - can generate interrupts ones selected thresholds are exceeded
  - built-in 16 bit converters
  - ranges:
    - +/- 2000°/sec
    - +/-16g



Źródło: www.invensense.com

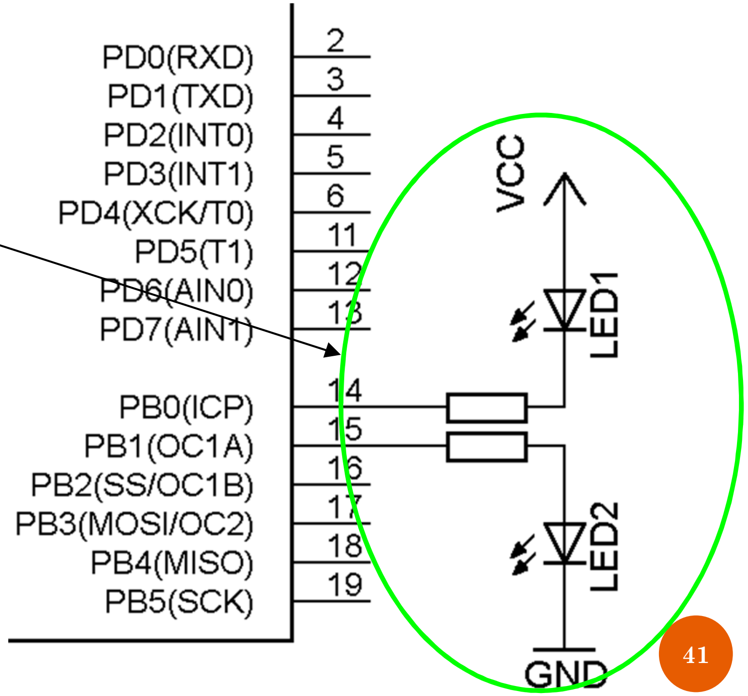
SENSORS

- How to handle motion sensor in software
  - **MPU6500/MPU6050** (Korneliusz Jarzebski library for Arduino)

```
#include <Wire.h>
#include <MPU6050.h>
MPU6050 mpu;

while(!mpu.begin(MPU6050_SCALE_2000DPS, MPU6050_RANGE_2G))
     delay(500);                    //searching of the sensor

Vector normAccel = mpu.readNormalizeAccel();
Serial.print(" Xnorm = ");Serial.print(normAccel.XAxis);
Serial.print(" Ynorm = ");Serial.print(normAccel.YAxis);
Serial.print(" Znorm = ");Serial.println(normAccel.ZAxis);
```

ACTUATORS

- How to connect an actuator: LED diode
  - Typical LED consumes from 5 to 20mA, but super bright LED can works with 1mA only (!)
  - how much current can you draw?
    - GPIO (per pin) current
      - „DC Current per I/O Pin” <40mA
      - note: in data sheets, sometimes, the allowed current is provided separately for each direction
    - total current for all peripherals
      - „DC Current VCC and GND Pins” <200mA
  - typically, both approaches are allowed



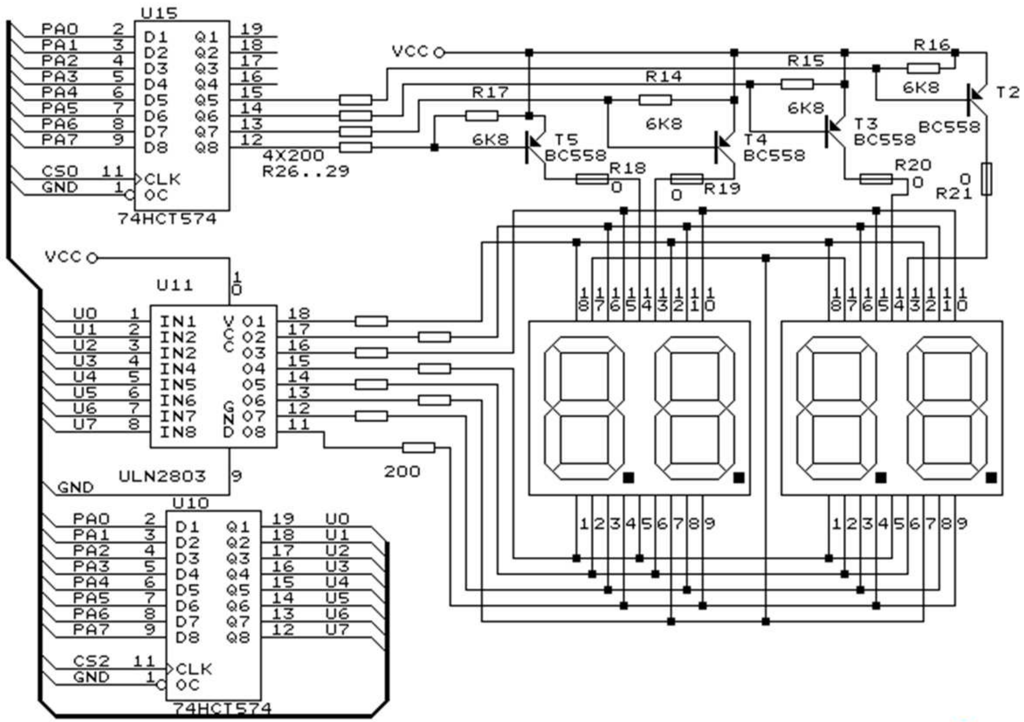
ACTUATORS

- How to handle GPIO in software
  - Arduino API (from Arduino IDE example)

```
void setup() {
  pinMode(13, OUTPUT); // init. digital pin 13 as an output.
}
void loop() {
  digitalWrite(13, HIGH); // turn LED on (HIGH is the voltage level)
  delay(1000);            // wait for a second
  digitalWrite(13, LOW);  // turn LED off by making the voltage LOW
  delay(1000);            // wait for a second
}
```

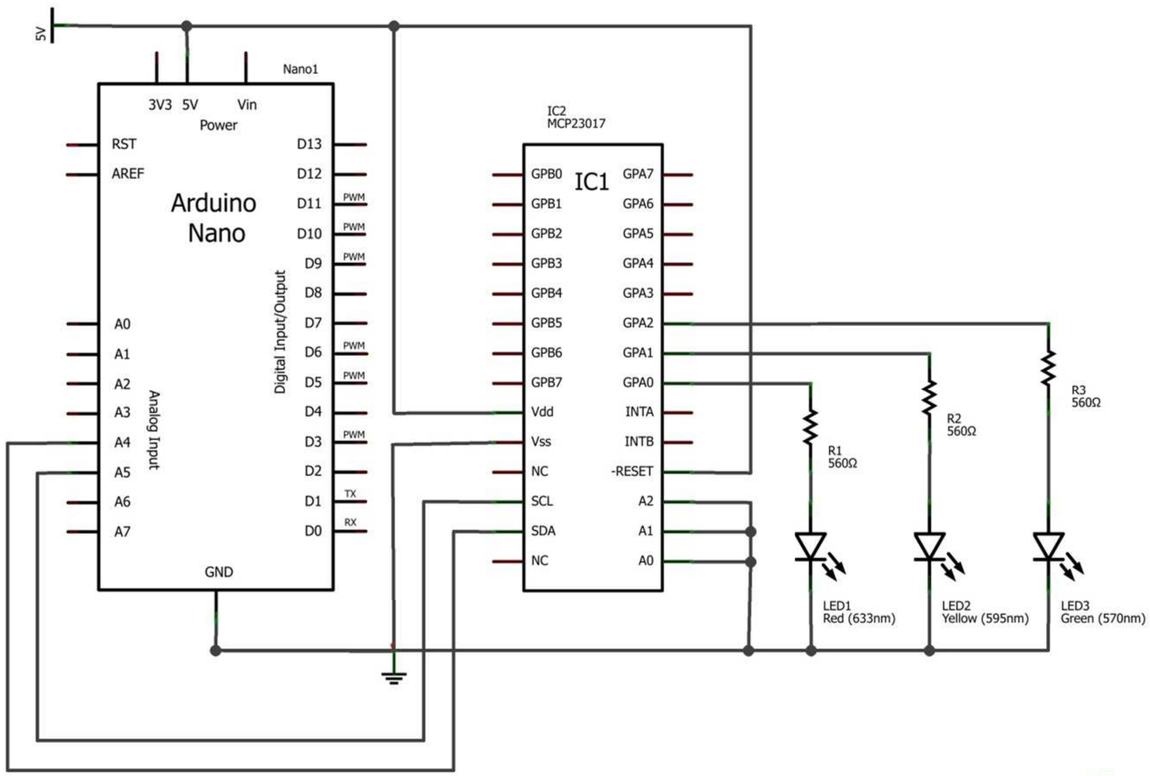
ACTUATORS

- How to connect an actuator: LED display (multiple LEDs)
  - limitations:
    - total available current
    - the number of available GPIO pins
  - solution I: use and CPU/MCU with more GPIO pins (more expensive)
  - solution II: apply segment multiplexing:



ACTUATORS

- How to connect an actuator: LED display (multiple LEDs), cont.
  - solution III: apply I/O expander, **MCP23017**
    - features 16 GPIO pins
    - can generate an interrupt when the state of a selected GPIO pin changes
    - communicates with CPU/MCU via the I^{2}C bus



Źródło: http://blog.jacobean.net

ACTUATORS

- How to handle multiple GPIO pins in software
  - **MCP23017** as a source of GPIO's (Adafruit library for Arduino)

```
#include <Wire.h>
#include "Adafruit_MCP23017.h"
Adafruit_MCP23017 mcp;
mcp.begin();

mcp.pinMode(0, INPUT);
mcp.pullUp(0, HIGH);        // turn on a pullup internally

Serial.println(digitalWrite(13, mcp.digitalRead(0)));
```

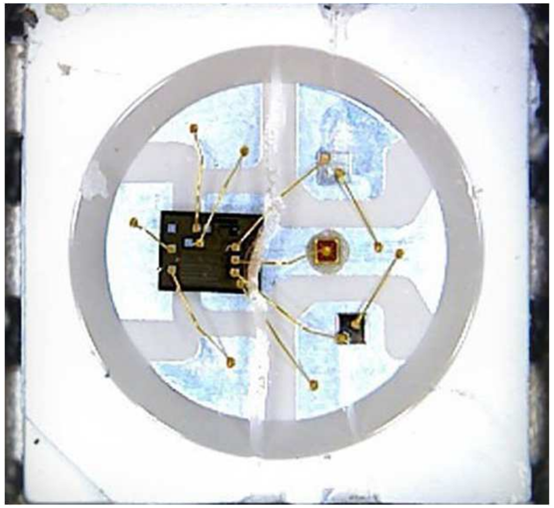
ACTUATORS

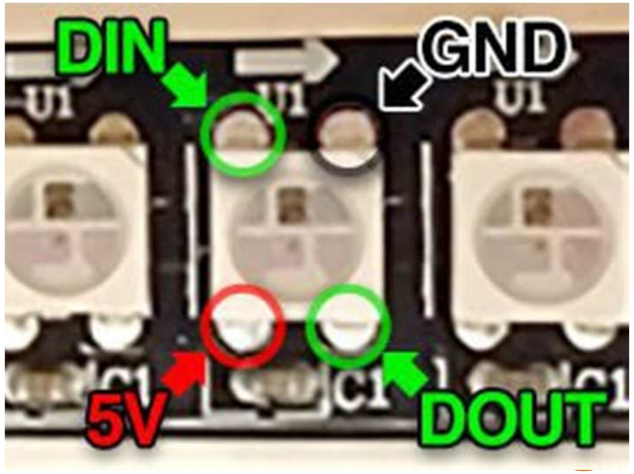
- How to handle multiple GPIO pins in software, cont.
  - **MCP23017** as a source of interrupt-producing GPIO’s (Adafruit library for Arduino)

```
#include <Wire.h>
#include "Adafruit_MCP23017.h”
Adafruit_MCP23017 mcp;
mcp.begin();

mcp.setupInterrupts(true, false, LOW);      //how AVR treats the INT
mcp.pinMode(mcpPinA, INPUT);
mcp.pullUp(mcpPinA, HIGH);            //turn on a pull-up internally
mcp.setupInterruptPin(mcpPinA, FALLING);      //falling edge trigger
attachInterrupt(AVR_VECTOR, handleInterrupt, FALLING);

void handleInterrupt(){
  Serial.println(”INT”);
  EIFR=0x01;                                   //clean Interrupt
}
```

ACTUATORS

- How to connect an actuator: LED strip (multiple LED's connected in series)
  - direct control
    - impractical (long cables, a thick cable bundle)
  - multiplexing, Charlieplexing
    - can be done, troublesome assembly
  - "telecom" approach
    - diodes are connected in series
    - each diode has an address
    - each diode accepts only messages addressed that are addressed to it
    - use **WS8211** (or **WS8212**) LED strips

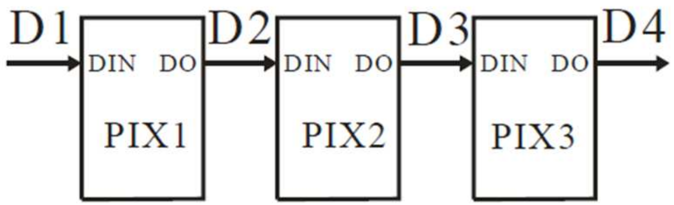


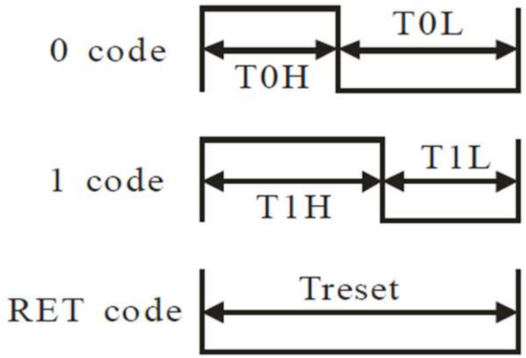


Źródło: https://learn.adafruit.com

ACTUATORS

- How to connect an actuator: **WS8211** LED strip
  - up to 1024 diodes in a single strip
  - each diode can glow in all three basic colors (R,G,B),
  - each diode can glow with intensity in the range of 0...255
  - each diode conditions the signal for the next one
  - transmission bit rate <800Kbps
  - con: the communication process is time-critical





T0H	0 code ,high voltage time	0.35us	±150ns
T1H	1 code ,high voltage time	0.9us	±150ns
T0L	0 code , low voltage time	0.9us	±150ns
T1L	1 code ,low voltage time	0.35us	±150ns
RES	low voltage time	Above 50µs	

Źródło: http://www.world-semi.com

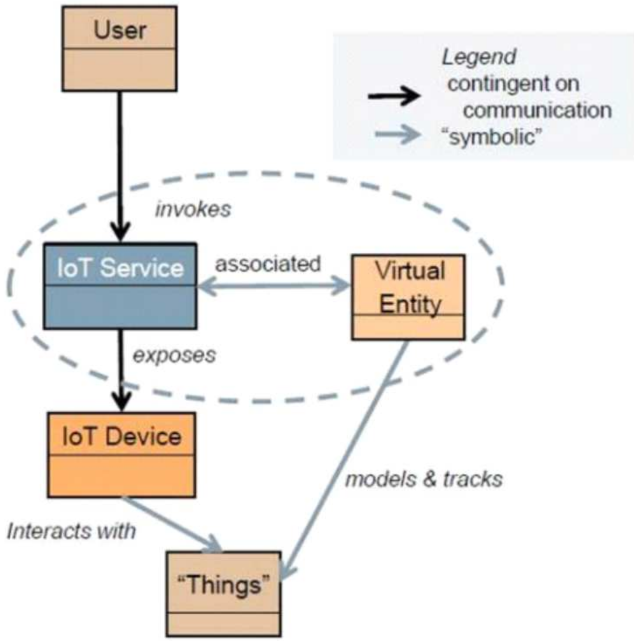
ACTUATORS

- Handling a LED strip in software
  - **WS8211** (NeoPixel Adafruit library for Arduino)

```
#include <Adafruit_NeoPixel.h>
#define PIN 6
#define NUMPIXELS 16
Adafruit_NeoPixel p=Adafruit_NeoPixel(NUMPIXELS, PIN,
                                               NEO_GRB + NEO_KHZ800);
p.begin();

// set led no. 0 (from one of 16 LEDs) on white color
p.setPixelColor(0, pixels.Color(0xFF, 0xFF, 0xFF));
p.show();                                    //blocking function(!)
```

- The implementation of show() method is trimmed (in ASM) to does transmission on time!!!

Thank you!


User
Legend
contingent on communication
"symbolic"
invokes
IoT Service
associated
Virtual Entity
exposes
IoT Device
models & tracks
Interacts with
"Things"