Monitoring Helium storage with a LV-MaxSOnar-EZ1 sonar sensor

From Sensor to Report - 1FA349

UPPSALA
UNIVERSITET

Xabier García Andrade

December 12, 2018

UPPSALA UNIVERSITET

Purpose

Accelerator development and light generation by charged particles.

How are particles accelerated?

RF cavities and magnets to deflect their trajectories.

Cooling down the equipment.

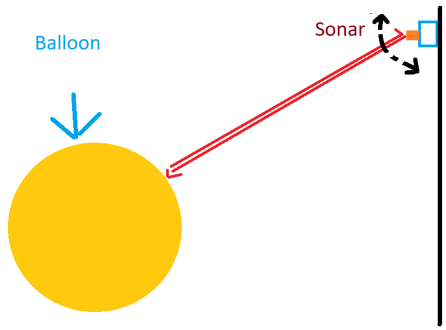
- Helium Liquifier.
- 2 K cryostats.

Motivation of the Project

UPPSALA UNIVERSITET

Scope

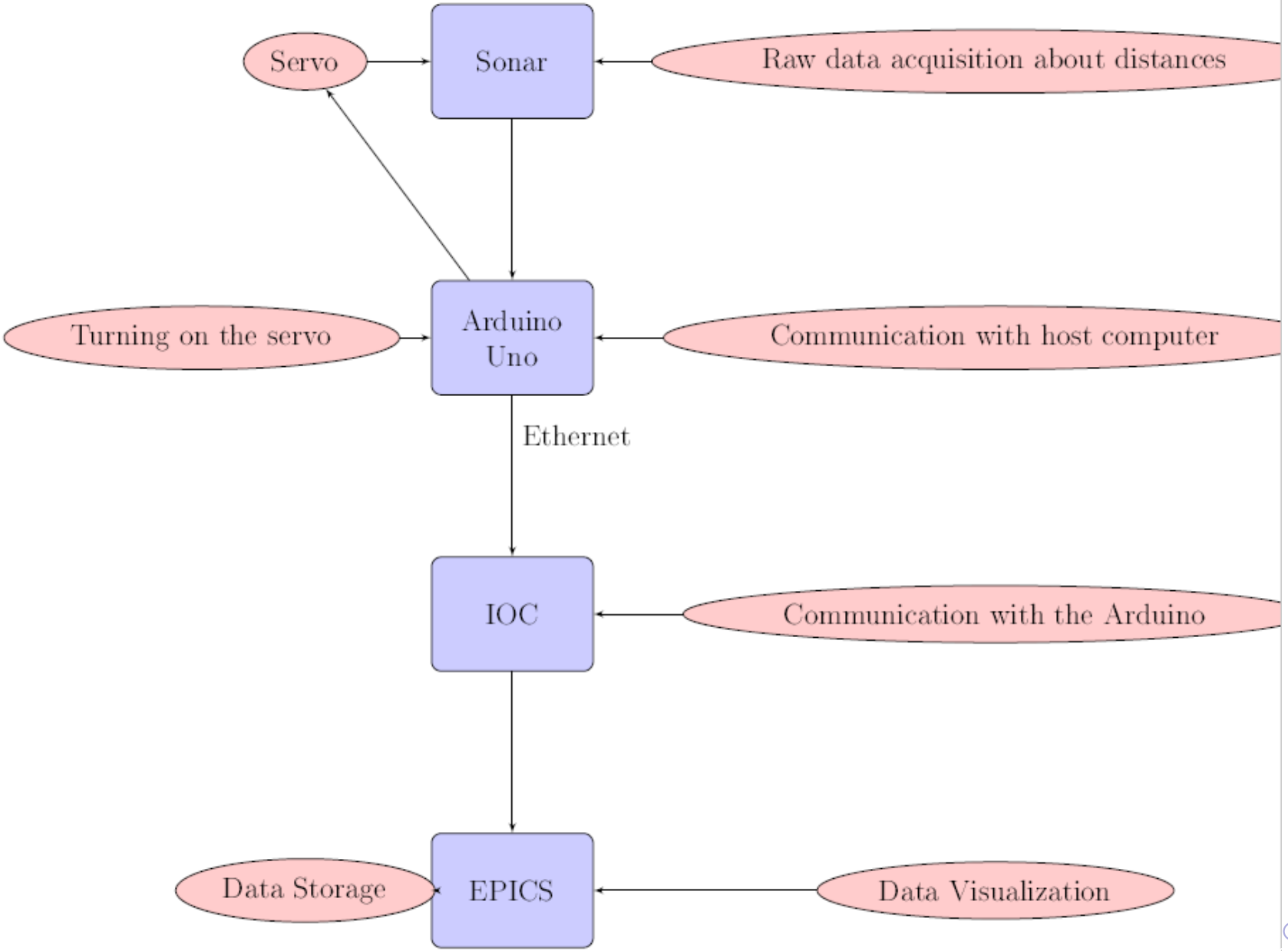
Monitoring the amount of helium contained in the gas balloon.



How do we achieve this?

Using ultrasounds to measure the distance to the surface from different angles.

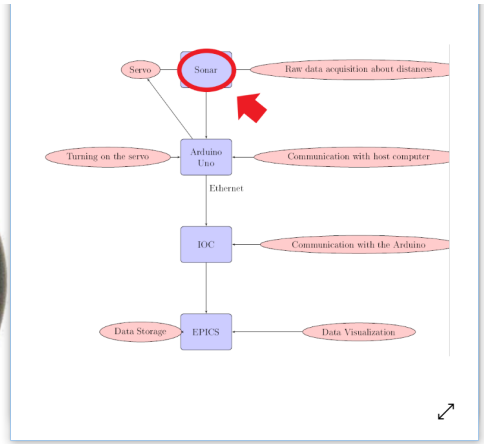
Overview


Servo
Sonar
Raw data acquisition about distances
Turning on the servo
Arduino Uno
Communication with host computer
Ethernet
IOC
Communication with the Arduino
Data Storage
EPICS
Data Visualization

UPPSALA UNIVERSITET

Sonar

Emits a short ultrasonic sound burst and records the duration until the echo arrives.

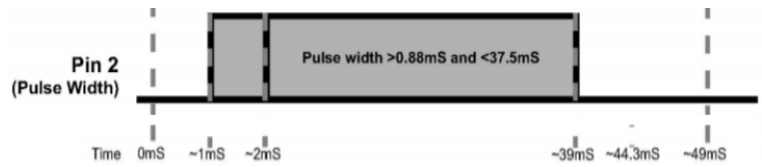


Several Output Modes

We use Pulse Width.

Specifications.

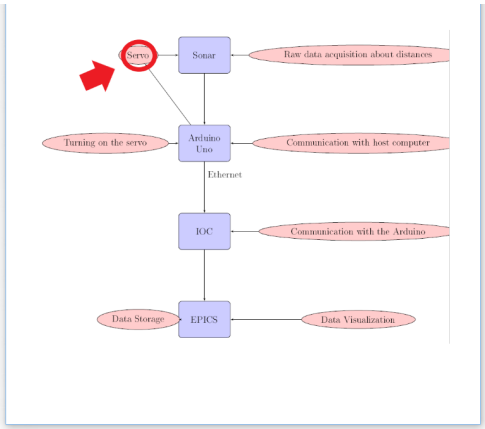
- Resolution of 1 in.
- Maximun Range of 254 in.
- Operates from 2.5 V to 5.5 V and draws 2 mA.



UPPSALA UNIVERSITET

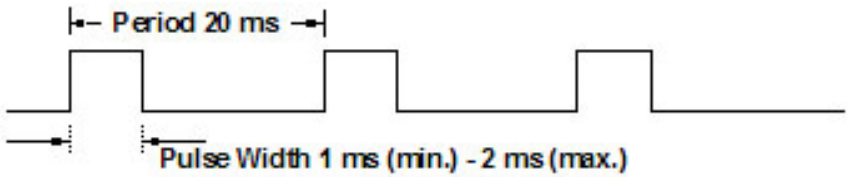
Servo

Motor with a position encoder.

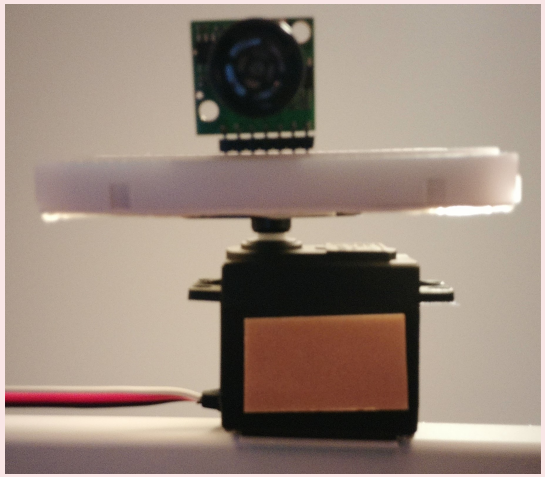


How to control it.

By sending a PWM signal, the width of the pulse determines the position of the servo.

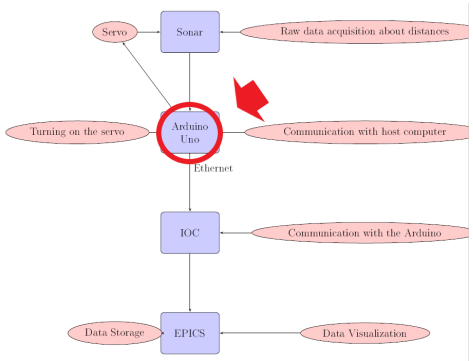


Mechanical Attachment.



UPPSALA UNIVERSITET

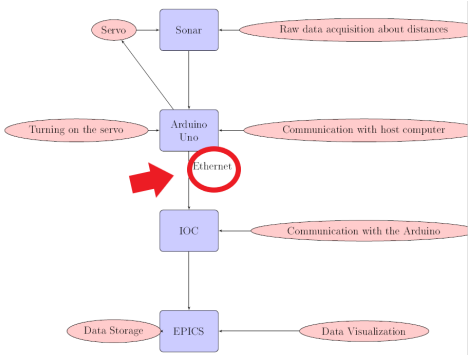
- Microcontroller board
- 14 digital input/output pins
- USB connection
- 16 MHz quartz crystal



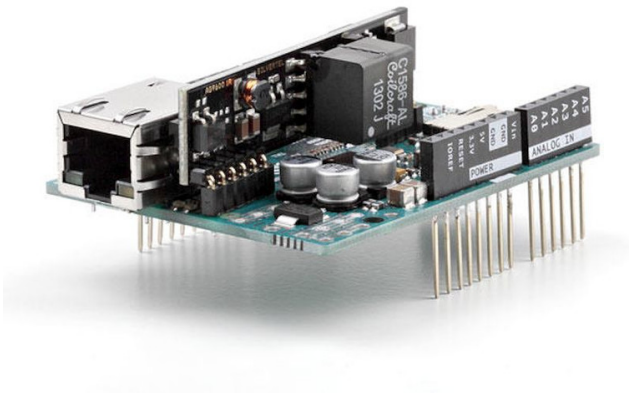
Reads the raw signal from the sensor and transforms it into a relevant physical quantity.

Interface

- Ethernet shield allows the arduino to communicate via Ethernet.
- More reliable and faster than Bluetooth.




```
#include <SPI.h>
#include <Ethernet.h>
EthernetServer server = EthernetServer(1137);
byte mac[] = {0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x02};
byte ip[] = { 192, 168, 1, 59};
byte gateway[] = { 192, 168, 1, 1 };
byte subnet[] = { 255, 255, 255, 0 };
```



Epics Control System

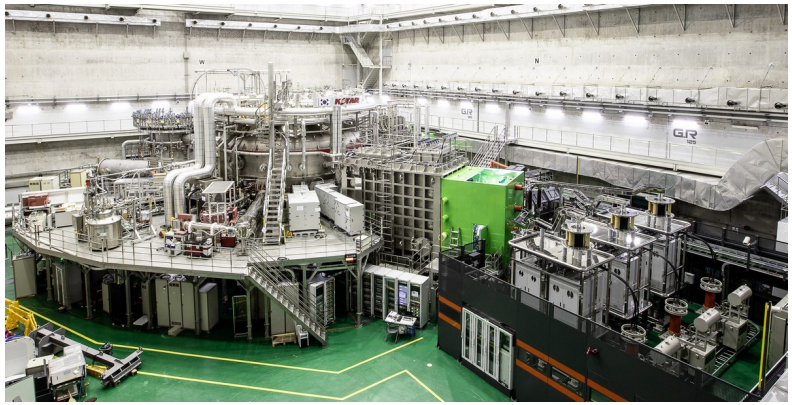
UPPSALA UNIVERSITET

What is it?

Software tools used to create distributed soft real-time control systems for scientific instruments such as a particle accelerators.

Specifications

- Communication: Client/Server and Publish/Subscribe techniques.
- Servers: IOCs (Input/Output Controllers).



- Publish: Using Channel Access (CA) network protocol.

UPPSALA UNIVERSITET

Setup:

At the beginning we need to specify the pins that we will be using, as well as initialize the Ethernet server

```
void setup() {
  //initialize our setup.
  myServo.attach(9);
  pinMode(pwpin, HIGH);
  Ethernet.begin(mac, ip, gateway, subnet);
  delay(2000);
  server.begin();
}
```

UPPSALA UNIVERSITET

Function where the distance is measured:

```
float measure(int pwpin) {
  float distance;
  float pulse;
  float inches;
  float cm ;
  pinMode(pwpin, INPUT);
  pulse = pulseIn(pwpin , HIGH);
  inches = pulse / 147.0;
  cm = inches * 2.54;

  return cm;
```

Three different commands:

"Scan"

Servo rotates while measuring distance in every position

```
else if (strstr(line , "SCAN?") == line) {
  int servo_pos = myServo.read();
  if (servo_pos<=90){
    int  count = 0;
    //servo sweeps while scanning, returns a matrix
     for (int pos = 0; pos <= 60 ; pos+=12) {
      myServo.write(pos);
      delay(100);
      struct rms_mean medidas = measure_mean(pwpin);
      matrix[count][0] = medidas.mean;
      matrix[count][1] = medidas.rms;
      matrix[count][2] = pos;
      count++;
    }
  }
```

UPPSALA UNIVERSITET

"Go to"

Servo moves to the selected position

```
if (strstr(line , "GOTO") == line) {
  uint16_t pos = (int)atof(&line[5]);
  //forces the servo to rotate
  if (pos <= 180) {
    myServo.write(pos);
    client.println(pos);
  }
```

"Measure"

Measures distance without moving

```
else if (strstr(line , "MEASURE?") == line) {
  //soanr measures distances without rotating.
  int can = 0;
  struct rms_mean medidas = measure_mean(pwpin);
  delay(100);
  client.print(medidas.mean);
  client.print(" , ");
  client.print(medidas.rms);
  client.print(" , ");
  client.println(can);

}
```

UPPSALA UNIVERSITET

Waveforms

After scaning, all the values will be stored in a matrix. Since we want to assign these values to a process variable in epics, we print them as a waveform.

```
else if (strstr(line , "WF RMS?") == line){
    for ( int i = 0 ; i<=5 ; i++){
      if (i<5){
        client.print(matrix[1][i]);
        client.print(",");
      }
    }
    client.println(matrix[1][5]);
  }
 else if (strstr(line , "WF POS?") == line){
    for ( int i = 0 ; i<=5 ; i++){
      if (i<5){
        client.print(matrix[2][i]);
        client.print(",");
      }
    }
   client.println(matrix[2][5]);
  }
```

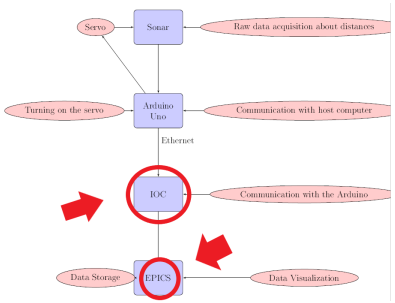
UPPSALA UNIVERSITET

Database File

Establishes how every variable is stored.

```
# .../Db/sonar_scan_pos.db

record(waveform , "$(USER):wfpos") {
        field(DESC, "SCAN POS waveform")
        field(DTYP , "stream")
        field(SCAN , "1 second")
        field(NELM, "6")
        field(FTVL, "FLOAT")
        field(INP , "@sonar_array.proto get_dis_pos $(PORT)")
}
```

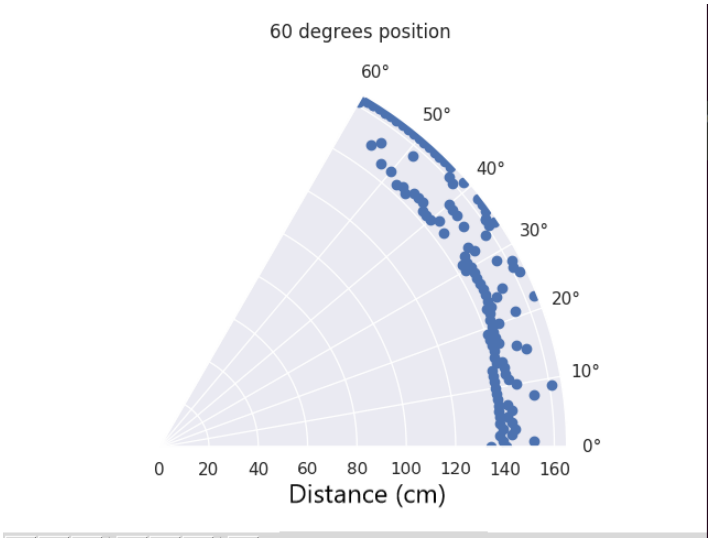


Protocol File

Defines functions that will be used to obtain the variables:

```
get_dis_scan {
        out "WF DIS?";
        separator =",";
        in "%f";
        ExtraInput = Ignore;
}
```

Prototype




```
plt.ion()

ser = serial.Serial('/dev/ttyACM0' , 9600, timeout = 1)

distance = []
degrees = []
rmeans = []

def plotear():
    ax = plt.subplot(111, projection = 'polar')
    aux = [2*np.pi*x/360.0 for x in degrees]
    ax.plot(aux, distance , 'bo')
    plt.title('60 degrees position ')
    plt.grid(True)
    plt.xlabel("Degrees")
    plt.ylabel("Distance (cm)")
    #plt.xlim(0,180)
    ax.set_thetamin(0)
    ax.set_thetamax(60)
```

```
while True:

    command = input()

    data = command.split(' ')
    if (command == 'SCAN?'):
        ser.write(command.encode('utf-8'))
        while (ser.inWaiting()==0):
            pass
        pos = 0
        while (pos!=180):
            valor = ser.readline()
            string = valor.decode('utf-8')
            linea = string.split(',')
            try:
                dis = float(linea[0])
                rms = float(linea[1])
                pos = float(linea[2])

                distance.append(dis)
                degrees.append(pos)
                rmeans.append(rms)
            except ValueError:
                print('check sensor')
            except IndexError:
                pass
            dnow.drawnow(plotear)
            print(pos)
```

Future Enhancements

UPPSALA UNIVERSITET

In the next future...

- Array of Sonars.
- Sonar with better resolution.
- Kinect for 3D reconstruction.
- Guide Rail.