







Virtual Instrumentation Project

Modular Arduino – powered Fingerprint Door Lock

Students:

-  *Rebegea Victor*
-  *Pop Raul Adrian*
-  *Craciun Adina Stavrula*
-  *Goman Alexandru*

Coordinating professor:

-  *Ph.D. engineer Lascu Mihaela*

University: *Polytechnic University of Timisoara*

Faculty: *Electronics, Telecommunications and Information Technologies*


Specialization: *TST- Telecommunications and telecommunications systems*


Academic year: *3rd*




Table of contents

 Main idea of the project and what it consists of

 Required tools and components

 Designing and printing necessarily parts

 Block diagram and process of implementing

 Final overview of the project

 Annex

Main idea of the project and what it consists of

While spending some time thinking about what we could implement for our faculty project, we figured that it would be helpful to make and implement a project that could actually serve a good purpose, something that makes or at least tries to make a certain action easier, using technology and virtual instrumentation.

After some time, as we sat there and thought about possible ideas, a colleague of ours told us a story about how he lost his apartment keys and was not able to get inside after a day of work, this being a moment of sheer frustration, despite being somewhat amusing at the time.

After hearing his story, we thought “How can we solve such situations and prevent them from happening in the first place? Wouldn’t it be helpful if we actually implement something that could help in such a situation?”

And then, it happened, we thought about making a mechanism that used a fingerprint to unlock your door, even if you do not have the keys on you, or maybe you have forgotten them at home, maybe at work or the office.

But this idea required more than a fingerprint sensor, some mechanical and electronic components, it also required programming, a virtual instrumentation method in order to work as desired and actually help the cause through technology.

In the end, we all agreed that this was to project to make and further below in this document, you will see the ways we used to make it happen, as a team.

Required Tools and Components

In the making of this project, we had to use quite a few components and tools in order to create a pathway for each component to function with each other.

The main parts are:

Development board – Arduino Nano v3

The Arduino Nano is Arduino's classic breadboard friendly designed board with the smallest dimensions. The Arduino Nano comes with pin headers that allow for an easy attachment onto a breadboard and features a Mini-B USB connector.

It features the **ATmega328 microcontroller** CPU which runs with 16 MHz internal oscillator and features 32 KB of Flash Memory (of which 2 KB used by bootloader).

It also has a tiny footprint with a length of 45 mm and a width of 18 mm, the Nano is Arduino's smallest board and weighs only 7 grams.

Some of its technical aspects are listed below:

Power	I/O Voltage	5V
	Input voltage (nominal)	7-12V
	DC Current per I/O Pin	20 mA
Clock speed	Processor	ATmega328 16 MHz
Memory	ATmega328P	2KB SRAM, 32KB flash 1KB EEPROM
Dimensions	Weight	5gr
	Width	18 mm
	Length	45 mm

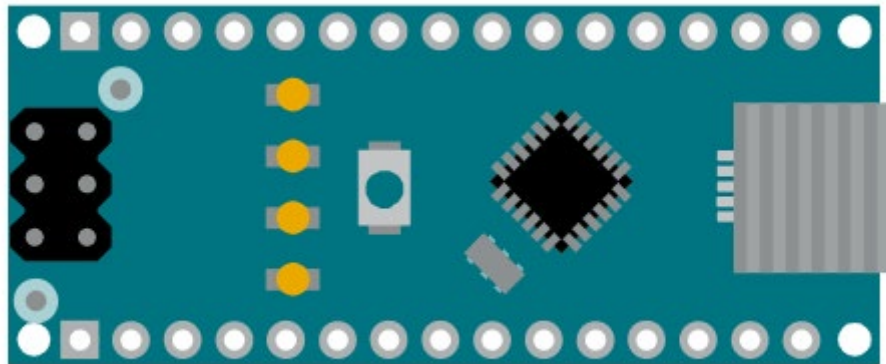


Figure.1 – General layout of the Arduino Nano Dev. Board

Servo motor – MG90S

This motor will be used to do rotation movements because it is able to rotate to a maximum of 180 degrees and it has added metal gears that offer better durability. Also, we can use any servo code, hardware or library to control these servo movements.

Some of its technical specifications are the following:

- Weight: 13.4 g
- Dimension: 22.5 x 12 x 35.5 mm approx.
- Stall torque: 1.8 kgf·cm (4.8V), 2.2 kgf·cm (6 V)
- Operating speed: 0.1 s/60 degree (4.8 V), 0.08 s/60 degree (6 V)
- Operating voltage: 4.8 V - 6.0 V
- Dead band width: 5 μ s

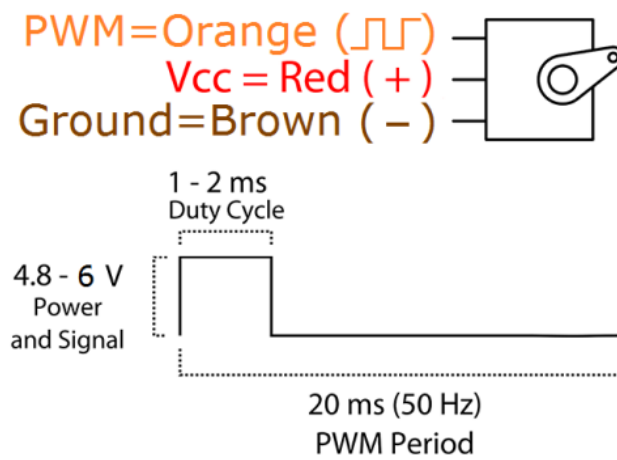


Figure.2 – PWM Signal controlling the motor

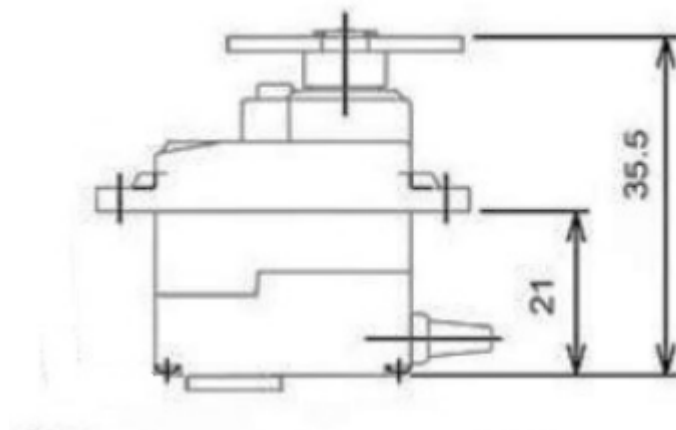


Figure.3 – Diagram of the MG90S Servo motor

Fingerprint sensor – FPM10A

The fingerprint module we used in this project is the FPM10A which is a fingerprint sensor compatible with Arduino.

This module comes with FLASH memory to store the fingerprints and work with any microcontroller or system with TTL serial. Also, it can be added to security systems, door locks, time attendance systems, and much more.

Some of its technical features are:

- Voltage supply: DC 3.6 to 6.0V
- Current supply: <120mA
- Backlight color: green
- Interface: UART
- Safety level: five (from low to high: 1,2,3,4,5)
- False Accept Rate (FAR): <0.001% (security level 3)
- False Reject Rate (FRR): <1.0% (security level 3)
- Able to store 127 different fingerprints

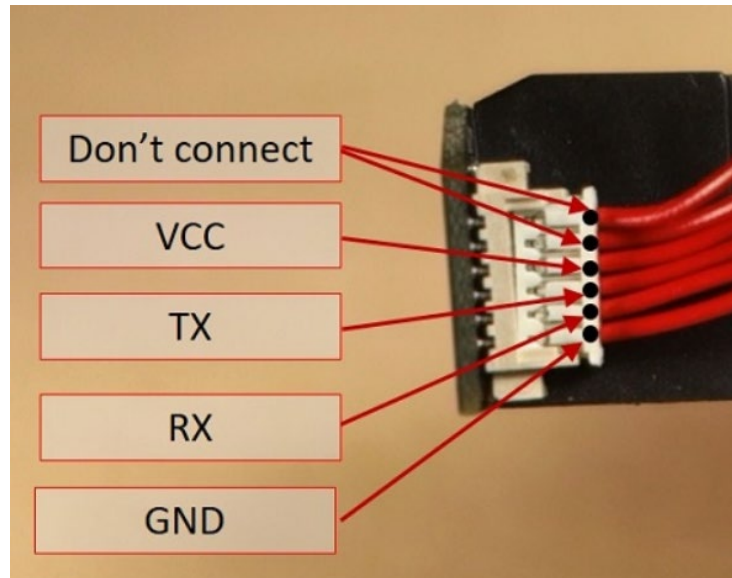


Figure.4 – Pinout of the FPM module

Magnetic

door reed switch

This part functions using the following principle:

The switching mechanism is comprised of two ferromagnetic blades, separated by only a few microns. When a magnet approaches these blades, the two blades pull toward one another. Once touching, the blades close the normally open (NO) contacts, allowing electricity to flow.



Figure.5 – Magnetic Door Reed Switch

- ✚ 2 x 1k Ω Resistors and 1 x Push Button
- ✚ Jumper cables for testing and prototyping
- ✚ 10 x M3 bolts and 10 x M3 nuts
- ✚ 3D printer for printing certain parts
- ✚ Soldering iron

✚ Designing and printing necessary parts

In this part, we needed a suitable design in order to actually make this door lock “modular”, which means that we would be able to attach it or take it off whenever we wanted, so we started to do some research and we found multiple ways we can design this door lock, but we looked for the most interesting way and came up with the following design, seen further below.

(Also, a very important mention is the fact that we have used a 3D printer in order to generate the parts, because they could not be found as singular items in technical shops.)

The following parts were needed:



Figure.6 – The base support for the modular door lock

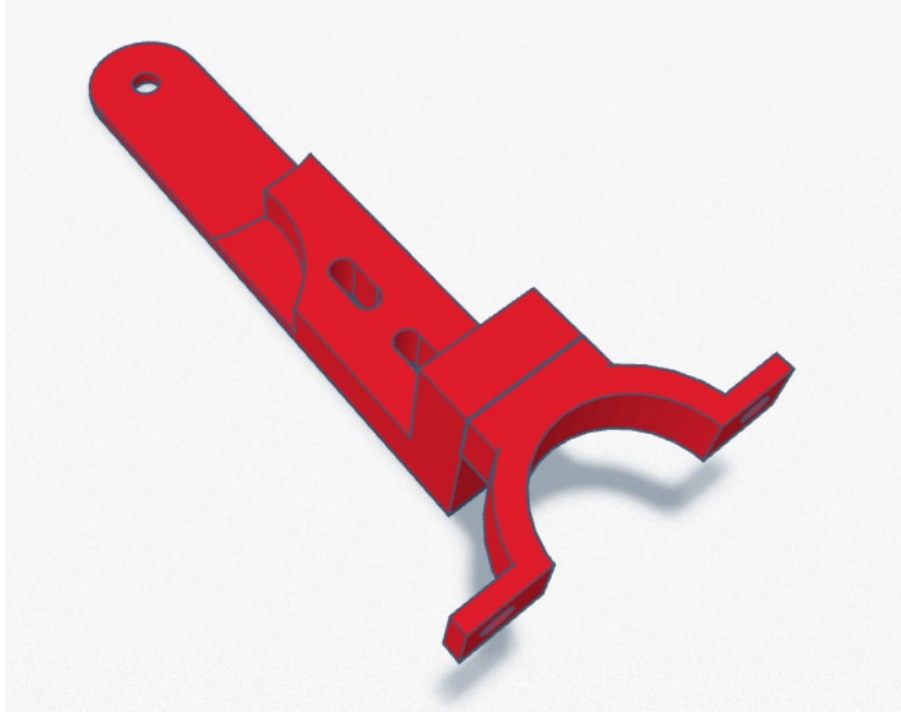


Figure.6.1 – Another view on the base support

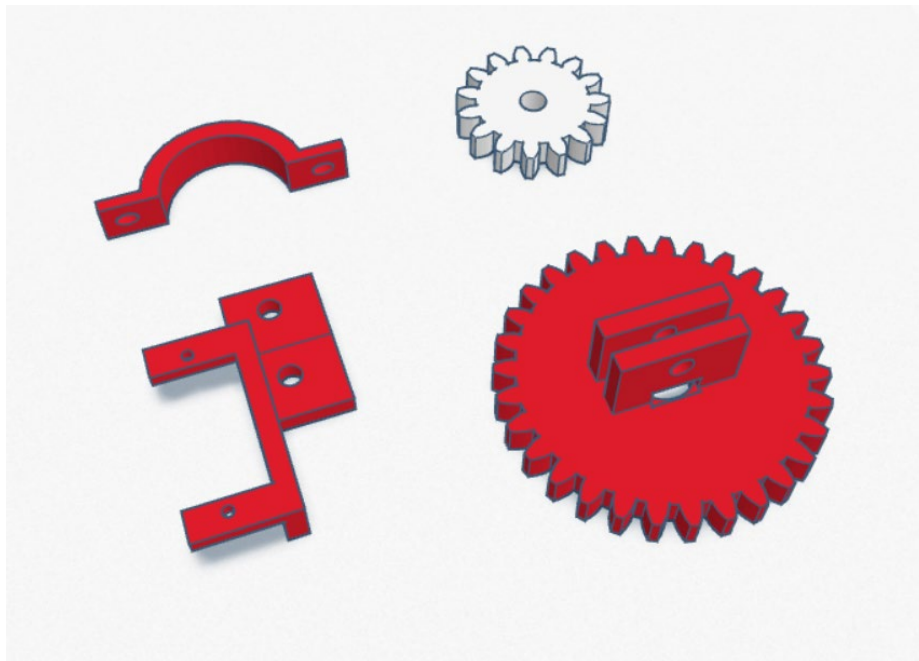


Figure.7 – The other parts used to implement the door lock

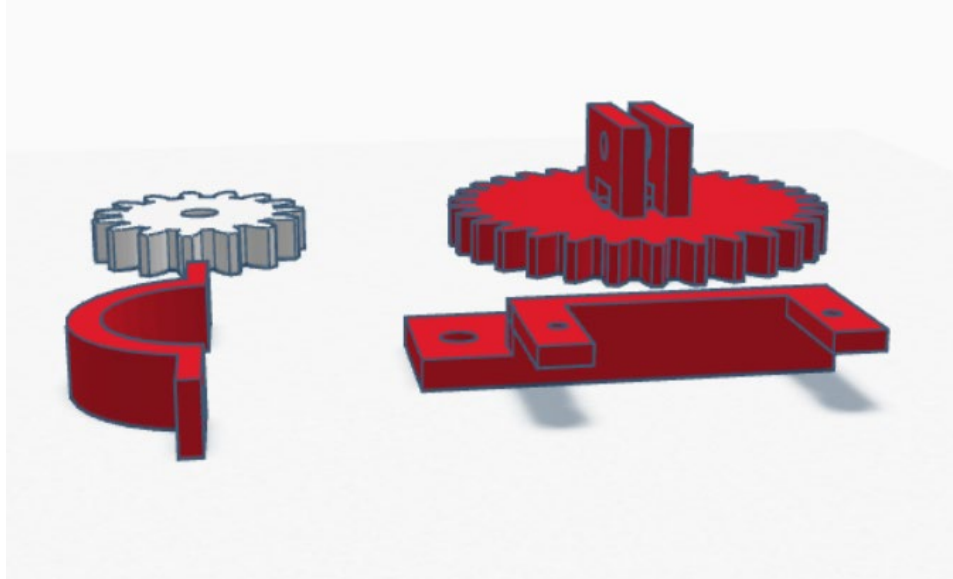


Figure.7.1 – Another view on the rest of the 3D printed parts

+ Block diagram and process of implementing

Now that our project was complete from the design and idea point of view, we had to come up with block schematic in order to do prototype testing and go through all necessary steps in order to implement physically an electronic circuit that would function correctly, alongside the programming part. For the prototype testing we have used a breadboard, batteries, jumper cables, a digital multimeter and Arduino's programming IDE to enroll fingerprints and create & debug the main code of the project.

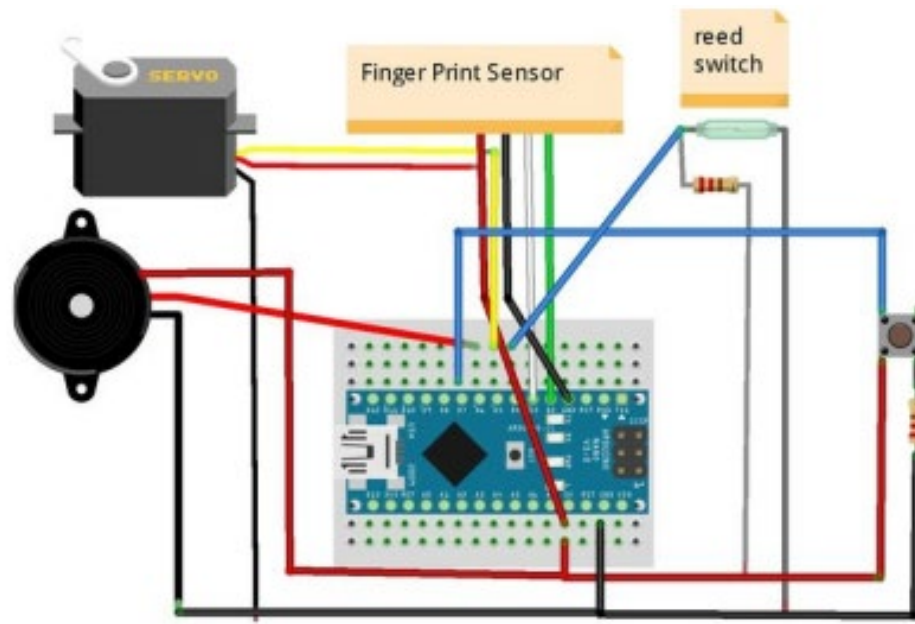
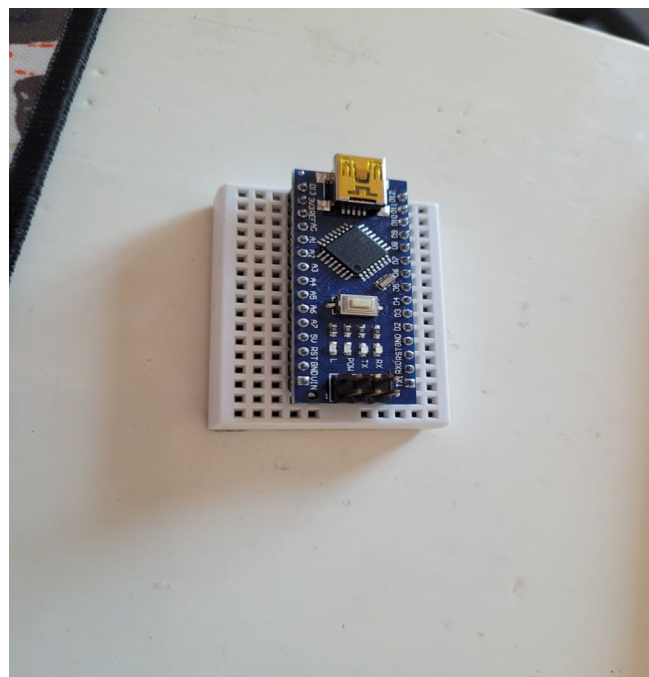
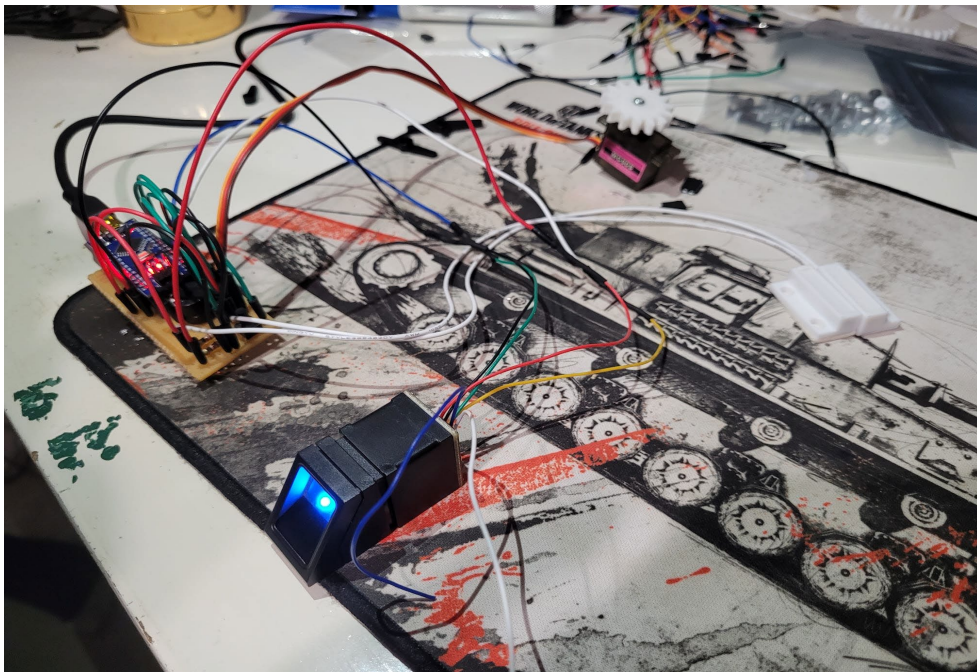
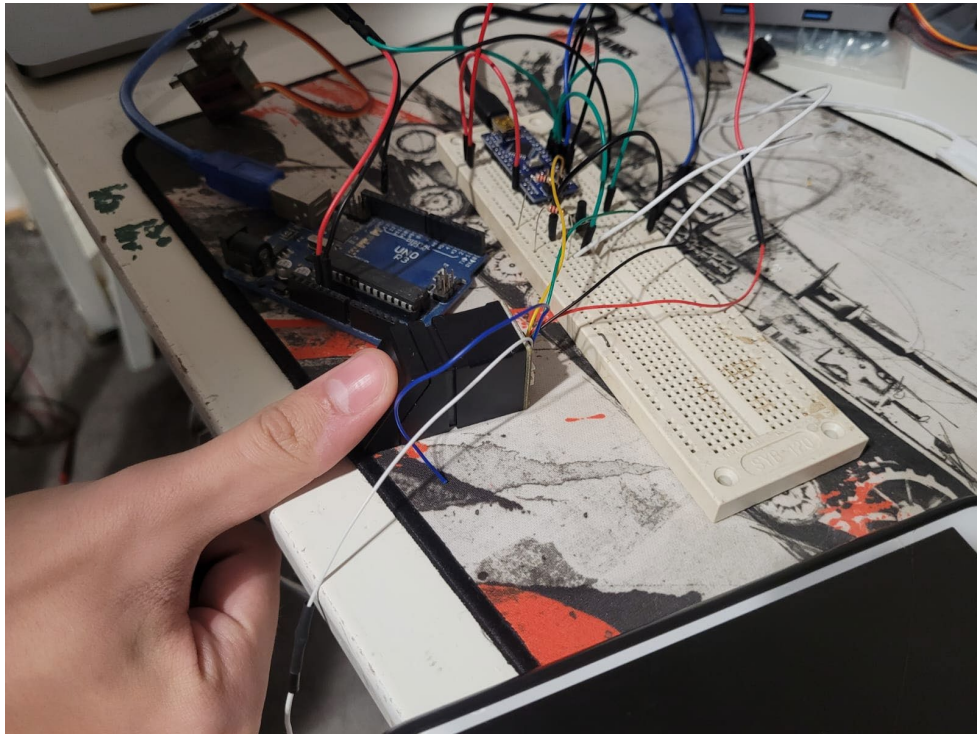
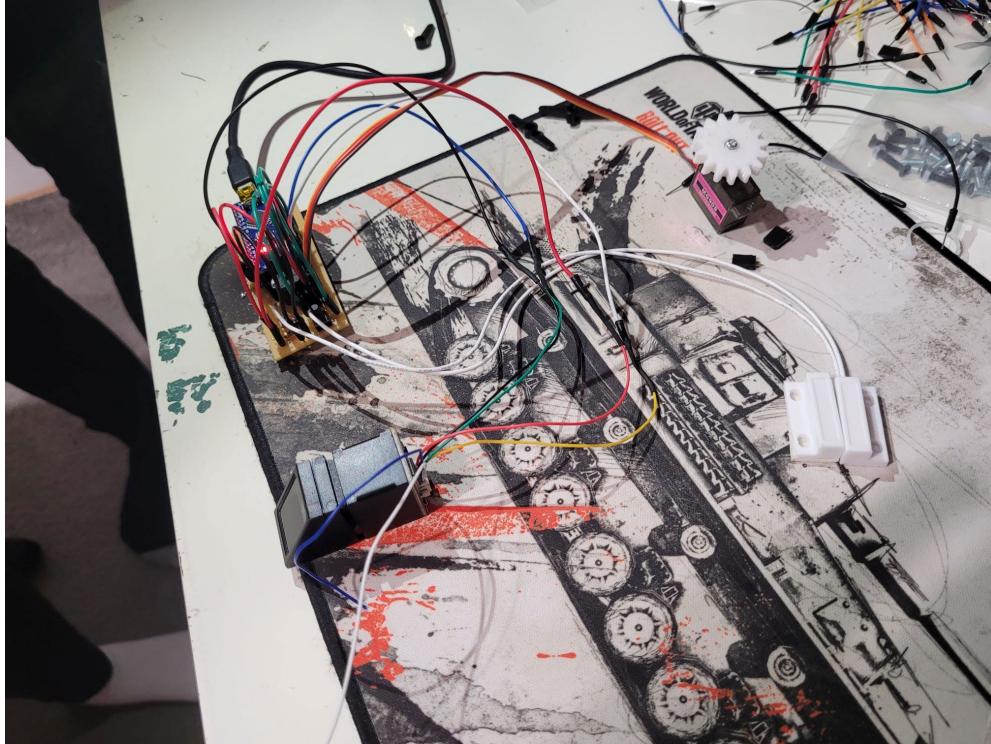


Figure.8 – Block Diagram of the physical implementation

Further below, there are some pictures of the testing & implementing process of our project:







✚ Final overview of the project

This project was created with the idea of displaying how technology is able to have a positive impact in every day related matters and also how it becomes a very useful tool if handled correctly.

We wanted solve a certain problem through technology and its capabilities, and we managed to do it, even though it required more time than we initially planned.

Although it took more time, we are glad that this project forced some of our “engineering instincts” out of us and we were able to connect as people, in order to achieve something that could be added to our experience, and ultimately making us more resilient and patient when dealing with difficulties.

Each and every single one of us had something to bring to table, and this project would not have been possible without everyone who participated. As a drawback to our project, we were not able to make a state of the art design and physical implementation, but we managed to display the functioning of such a system, and although it is not perfect, we look forward to upgrade and make even more interesting projects in the future.

In the end, we want to thank our professor, *Mihaela Lascu*, for the support and for allowing us to participate as a team in the making of this project, rather than having to do it individually, and for all the concepts we learned about Virtual Instrumentation and how it impacts every single thing related to technology, from codes & software to hardware & physical processes.

This project was created with research and education purposes, and it shall not be used in other matters that do not involve ethical and education based contents.



Annex

The code we have used for our project is the following:

```
#include <Servo.h>
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>

Servo myservo;
int getFingerprintIDez();
boolean lock = false;
int pos = 0;
int buzzer = 6;
int buttonPin = 7;
int magnet = 4;
boolean doorStatus = false;
// pin #2 is IN from sensor (GREEN wire)
// pin #3 is OUT from arduino (WHITE wire)
SoftwareSerial mySerial(2, 3);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);
// On Leonardo/Micro or others with hardware serial, use those! #0 is green
// wire, #1 is white
//Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial1);
void setup()
{
  while (!Serial); // For Yun/Leo/Micro/Zero/...
  pinMode(buzzer, OUTPUT);
  pinMode(buttonPin, INPUT);
  pinMode(magnet, INPUT);
  Serial.begin(9600);
  Serial.println("finger detect test");
  // set the data rate for the sensor serial port
```

```

finger.begin(57600);
if (finger.verifyPassword()) {
  Serial.println("Found fingerprint sensor!");
} else {
  Serial.println("Did not find fingerprint sensor :(");
  while (1);
}
Serial.println("Waiting for valid finger...");
}
void loop(){
  //Serial.println(digitalRead(magnet)); // test for magnet readings.
  if(doorStatus){
    if(digitalRead(magnet)==LOW){
      delay(1000);
      sweepc(); //rotate servo from 20 to 180 degrees.
      doorStatus=false;
    } else {
      sweepcc(); //rotate servo from 180 to 20 degrees.
    }
  }
  if(getFingerprintIDez()>=0||digitalRead(buttonPin)==LOW){
    digitalWrite(buzzer, HIGH);
    delay(200);
    digitalWrite(buzzer, LOW);
    doorStatus=true;
    sweepcc();
    delay(1000);
  }
  delay(50); //don't need to run this at full speed.
}
uint8_t getFingerprintID() {
  uint8_t p = finger.getImage();

```

```

switch (p) {
case FINGERPRINT_OK:
Serial.println("Image taken");
break;
case FINGERPRINT_NOFINGER:
Serial.println("No finger detected");
return p;
case FINGERPRINT_PACKETRECEIVEERR:
Serial.println("Communication error");
return p;
case FINGERPRINT_IMAGEFAIL:
Serial.println("Imaging error");
return p;
default:
Serial.println("Unknown error");
return p;
}
// OK success!
p = finger.image2Tz();
switch (p) {
case FINGERPRINT_OK:
Serial.println("Image converted");
break;
case FINGERPRINT_IMAGEMESS:
Serial.println("Image too messy");
return p;
case FINGERPRINT_PACKETRECEIVEERR:
Serial.println("Communication error");
return p;
case FINGERPRINT_FEATUREFAIL:
Serial.println("Could not find fingerprint features");
return p;

```

```

case FINGERPRINT_INVALIDIMAGE:
Serial.println("Could not find fingerprint features");
return p;
default:
Serial.println("Unknown error");
return p;
}
// OK converted!
p = finger.fingerFastSearch();
if (p == FINGERPRINT_OK) {
Serial.println("Found a print match!");
} else if (p == FINGERPRINT_PACKETRECEIVEERR) {
Serial.println("Communication error");
return p;
} else if (p == FINGERPRINT_NOTFOUND) {
Serial.println("Did not find a match");
return p;
} else {
Serial.println("Unknown error");
return p;
}
// found a match!
Serial.print("Found ID #"); Serial.print(finger.fingerID);
Serial.print(" with confidence of "); Serial.println(finger.confidence);
}
// returns -1 if failed, otherwise returns ID #
int getFingerprintIDez() {
uint8_t p = finger.getImage();
if (p != FINGERPRINT_OK){
return -1;
}
p = finger.image2Tz();

```

```

if (p != FINGERPRINT_OK) return -1;
p = finger.fingerFastSearch();
if (p != FINGERPRINT_OK) return -1;
// found a match!
Serial.print("Found ID #"); Serial.print(finger.fingerID);
Serial.print(" with confidence of "); Serial.println(finger.confidence);
return finger.fingerID;
}
void sweepc(){
if(lock==false){
myservo.attach(5);
for (pos = 20; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
// in steps of 1 degree
myservo.write(pos); // tell servo to go to position in variable 'pos'
delay(15); // waits 15ms for the servo to reach the position
}
lock=true;
myservo.detach();
}
}
void sweepcc(){
if(lock){
myservo.attach(5);
for (pos = 180; pos >= 20; pos -= 1) { // goes from 180 degrees to 20 degrees
myservo.write(pos); // tell servo to go to position in variable 'pos'
delay(15);
}
lock = false;
myservo.detach();
}
}

```

The bibliography used:

- 1) <https://docs.arduino.cc/hardware/nano> - Accessed on December 28th, 2022
- 2) <https://components101.com/motors/mg90s-metal-gear-servo-motor> - Accessed on December 28th, 2022
- 3) <https://www.adafruit.com/product/751> - Accessed on December 29th, 2022
- 4) <https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library> - Accessed on January 2nd, 2023
- 5) <https://www.arduino-libraries.info/libraries/adafruit-fingerprint-sensor-library> - Accessed on January 2nd, 2023
- 6) <https://www.instructables.com> – Accessed on January 3rd, 2023