ASSIGNMENT – 2

Advanced Programming Lab

Name: Utsav Balar

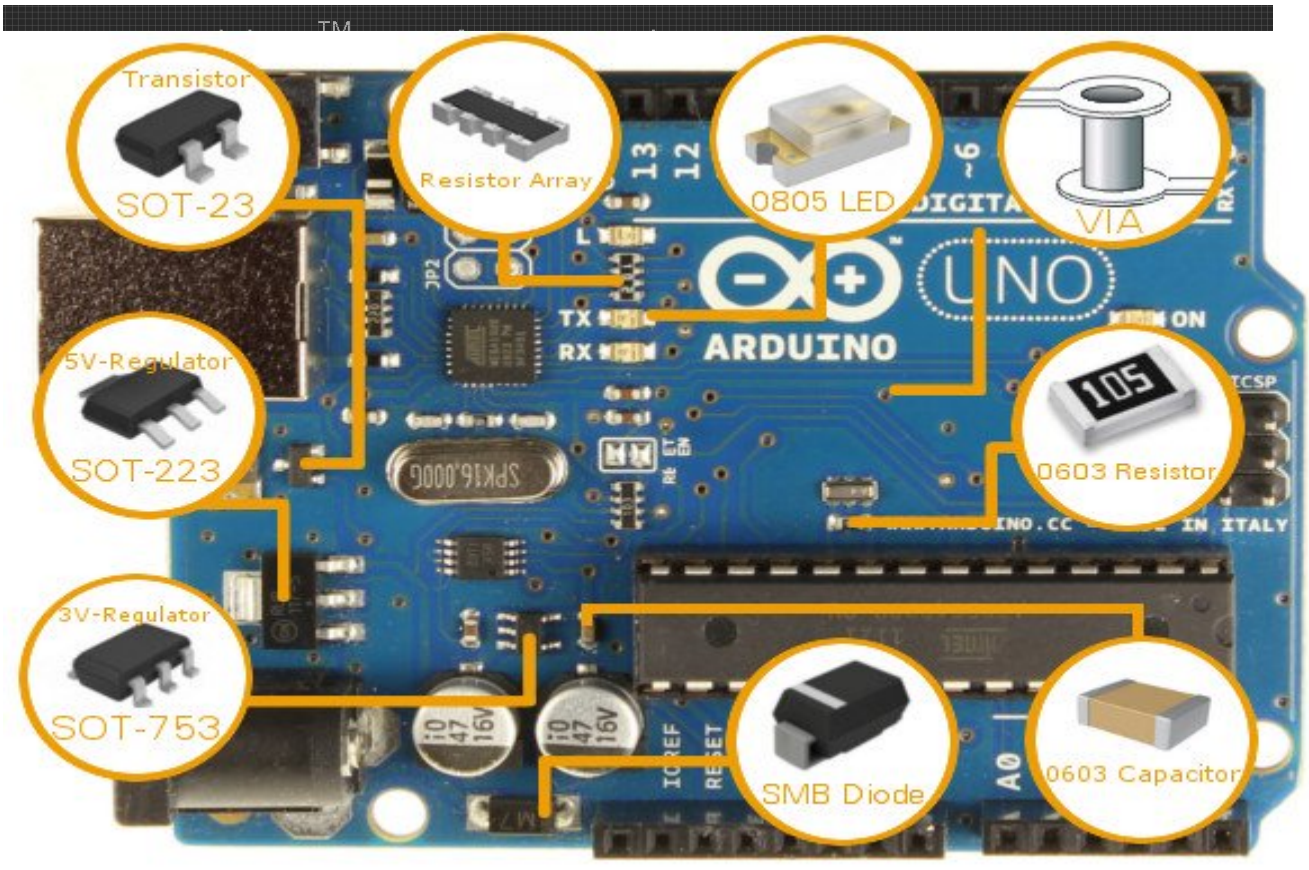
Roll.no. t24cs003

Objective:

Assignments on Familiarisation microcontroller board (Arduino/ Raspberry Pi) Assignment on sensors and how to interface them with the microcontroller board, data acquisition and processing of GSR data

Overview of the Arduino UNO:

The **Arduino UNO** is a microcontroller board built around the ATmega328P. It features 14 digital input/output pins, of which 6 support PWM outputs, as well as 6 analog inputs, a 16 MHz ceramic resonator, a USB port, a power jack, an ICSP header, and a reset button. The board is equipped to handle all the essentials for running the microcontroller; you can power it either through a USB connection to a computer or with an AC-to-DC adapter or battery. The **Arduino UNO** is an open-source hardware platform developed by Arduino.cc, initially released in 2010. It supports integration with a variety of expansion shields and circuits, thanks to its digital and analog I/O pins. It has a USB-B connector for programming with the **Arduino IDE**. The board can run on power supplied through USB or a barrel jack, accommodating input voltages from 7 to 20 volts. The design is open-source and distributed under the Creative Commons Attribution Share-Alike 2.5 license, with layout and production files available on the Arduino website.



Key Specifications (from UNO R1 to R3):

1. **Microcontroller**: Microchip ATmega328P (8-bit AVR core)
2. **Clock Speed**: 16 MHz on the board, with a max of 20 MHz at 5V for the IC
3. **Memory**:
4. **Flash**: 32 KB (0.5 KB used by the bootloader)
5. **SRAM**: 2 KB
6. **EEPROM**: 1 KB
7. **Peripherals**:
8. 1 **UART** (default configured as an 8N1 UART)
9. 1 **SPI**
10. 1 **I²C**
11. **Operating Voltage**: 5V
12. **Digital I/O Pins**: 14
13. **PWM Pins**: 6 (pins 3, 5, 6, 9, 10, 11)
14. **Analog Input Pins**: 6
15. **DC Current Limits**:
16. Per I/O Pin: 20 mA
17. 3.3V Pin: 50 mA
18. **Physical Characteristics**:
19. **Size**: 68.6 mm x 53.4 mm
20. **Weight**: 25 g
21. **ICSP Header**: Yes
22. **Power Options**:
23. **USB Connector**: Supplies power in the range of 4.75 to 5.25 volts. Official boards use a USB-B connector, though third-party variations might feature microUSB, miniUSB, or USB-C.
24. **Barrel Jack**: Supports 6 to 20 volts, with 7 to 12 volts recommended. Third-party boards may have different maximum voltages depending on the voltage regulator used.
25. **VIN Pin**: Accepts similar voltage as the barrel jack. However, it lacks reverse voltage protection, so an external diode is necessary if using both the VIN pin and barrel jack simultaneously.

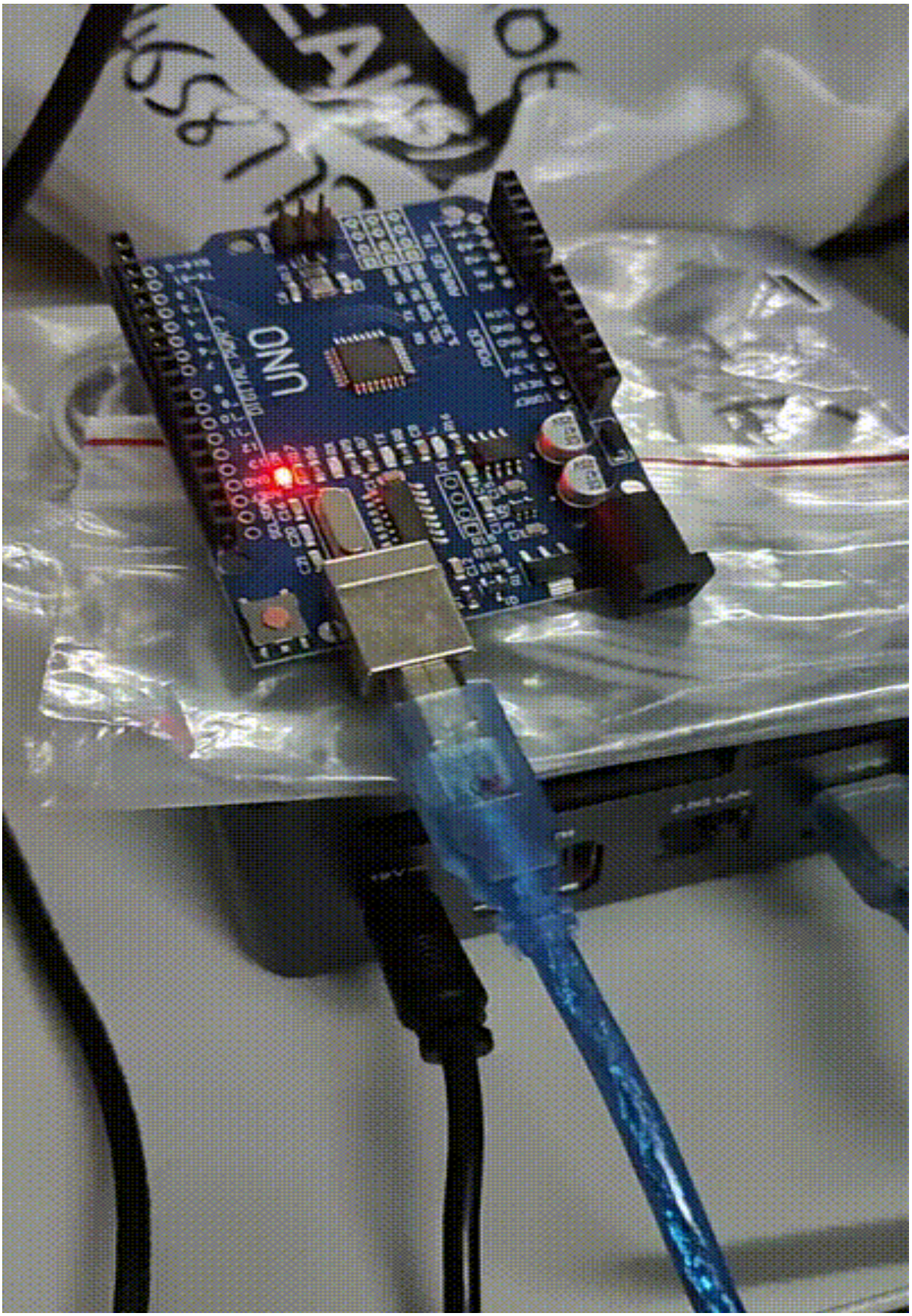
Applications of the Arduino UNO:

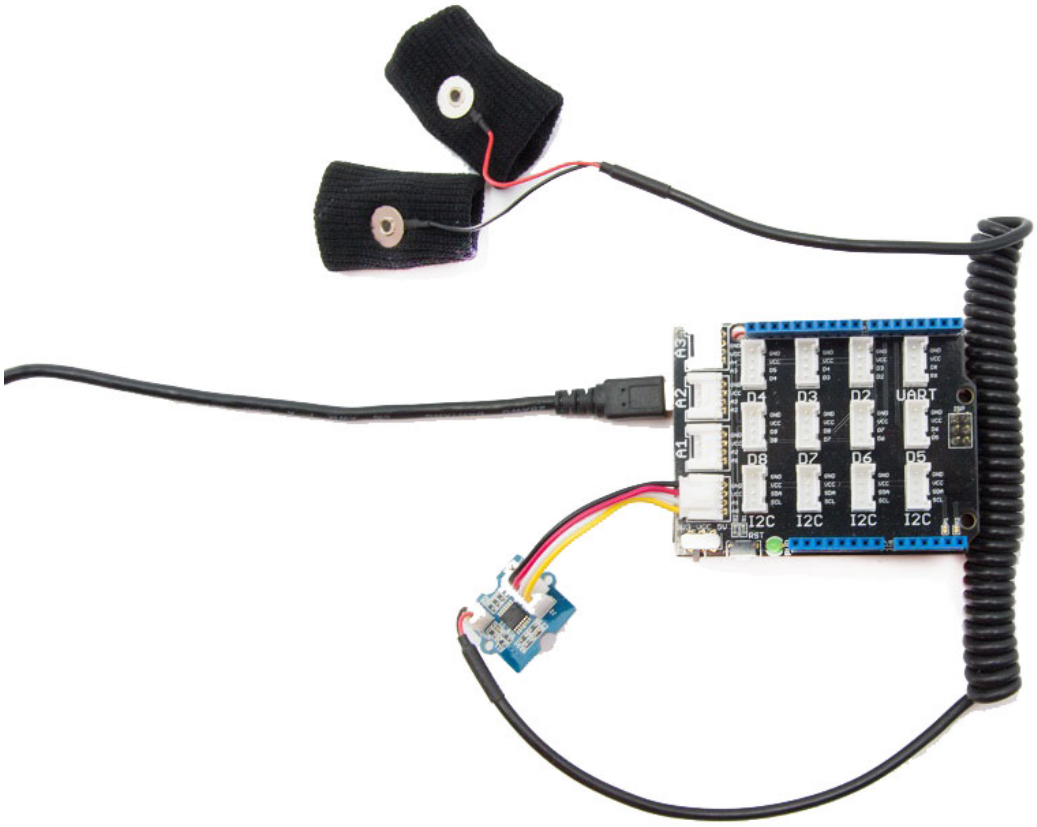
1. **Home Automation**: Automate lighting, temperature control, security systems, and energy management. For example, using IR sensors for automatic light control.
2. **Robotics**: Build autonomous robots equipped with sensors, motors, and controllers.
3. **Smart Gardening**: Develop systems for automated plant care, such as moisture-based irrigation.
4. **IoT Devices**: Create Internet of Things devices for various smart applications.
5. **Wearable Technology**: Design and implement wearable electronic gadgets.
6. **Control and Measurement**: Useful in advanced scenarios like Industry 4.0, industrial cyber-physical systems (ICPSs), and smart grid applications.
7. **Miscellaneous Uses**: Includes applications in security, embedded systems, digital electronics, weighing systems, parking counters, medical instrumentation, and countdown timers for traffic lights

Arduino code to blink a LED :

```
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}


// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the
voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the
voltage LOW
  delay(1000);                      // wait for a second
}
```


UNO


A3
A2
A1
D4
D3
D2
UART
D8
D7
D6
D5
I2C
I2C
I2C
I2C

*** Data acquisition and processing of GSR data**

code.ino:

```
// Pin configuration
constexpr uint8_t GSR_PIN = A0;

const int sensorValue = 0;
const int gsrAverage = 0;
const float V_supply = 5.0;
const float R_fixed = 10000;

void setup() {
    Serial.begin(9600);
}

void loop() {
    constexpr size_t NUM_READINGS = 10;
    long sum = 0;

    for (size_t i = 0; i < NUM_READINGS; ++i) {
        sensorValue = analogRead(GSR_PIN);
        sum += sensorValue;
        delay(5);
    }

    gsrAverage = static_cast<int>(sum / NUM_READINGS);

    long V_out = (sensorValue * V_supply) / 1023.0;
    long g_sensor_R = R_fixed * ((V_supply / V_out) - 1);
    Serial.println("Human Resistance: ");
    Serial.print(g_sensor_R);
    Serial.println("GSR Average: ")
    Serial.println(gsrAverage);

    delay(500);
}
```

Output after taking 100 samples:

```
0     449
1     471
2     462
3     458
4     462
      ...
57    483
58    494
59    487
60    494
61    492
Name: GSR Average, Length: 62, dtype: int64
```

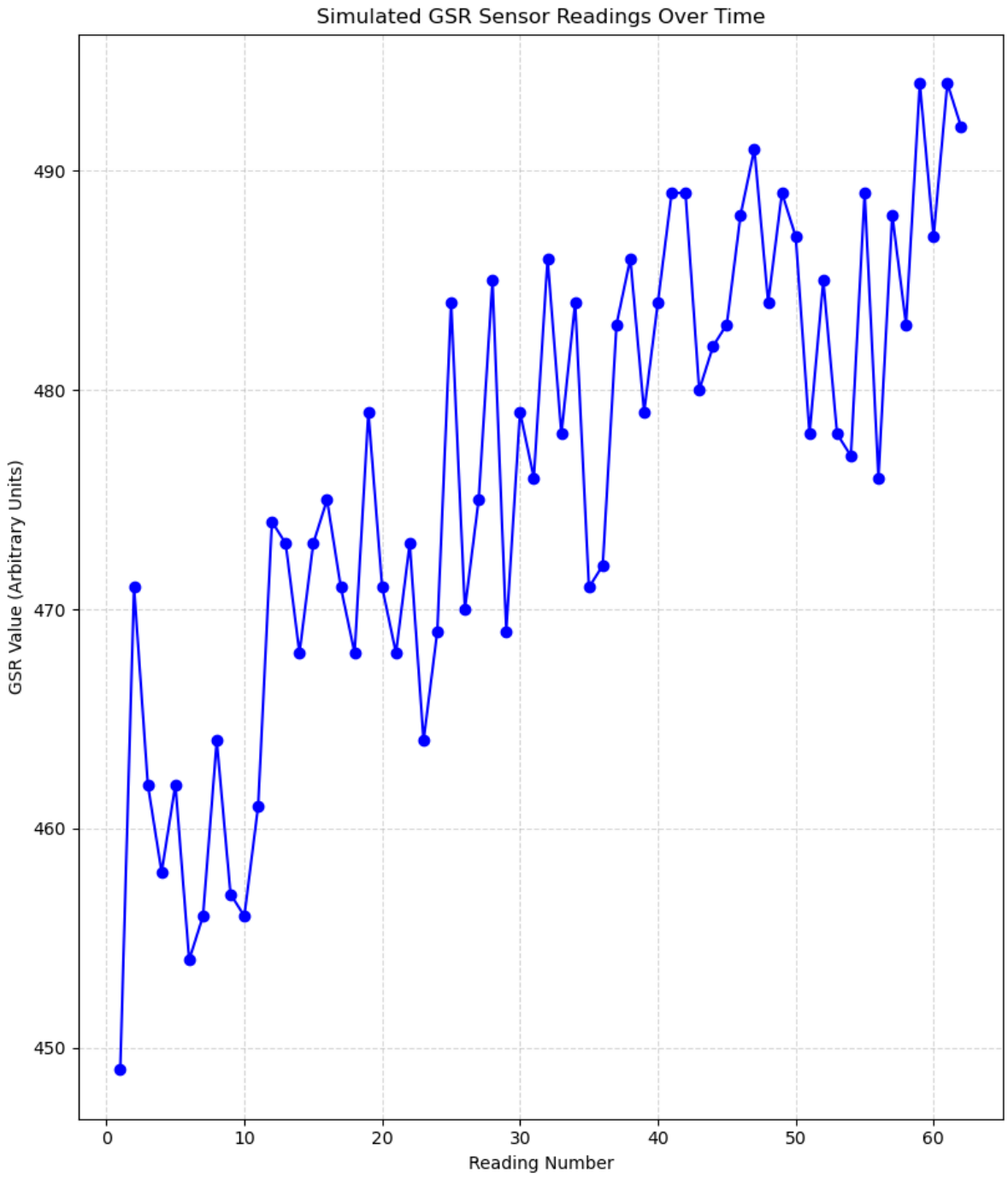


Fig: Simulated GSR readings over time

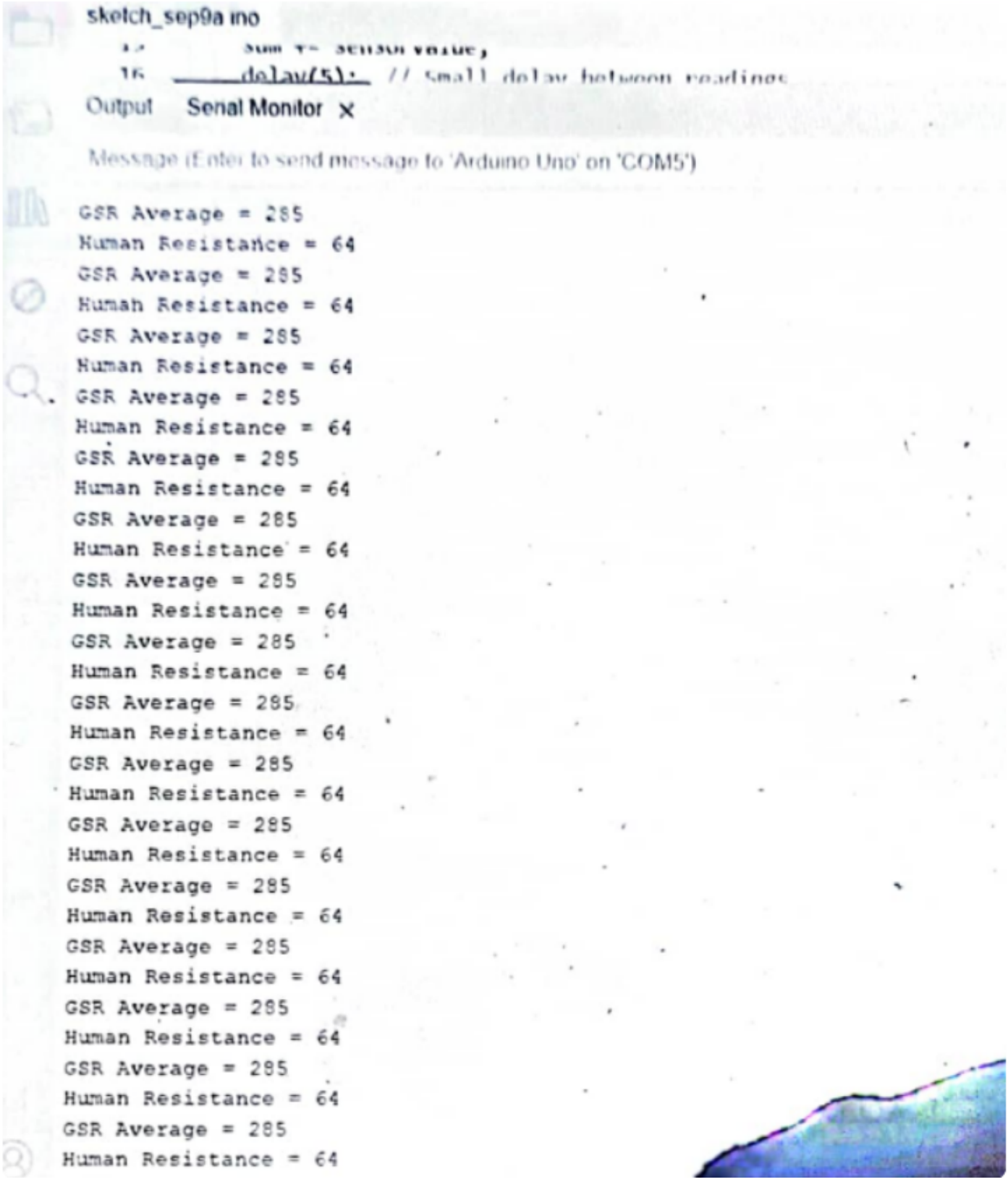


Fig: Output from the sensor