

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
|  | | Smart Wireless System for Real-Time Posture and Leg Lift Detection | | | | |  | |
|  |  | | | | | | |  |
|  | | | |  |  | | | |
|  | | | | Arnav Jagtap |  | | | |
|  | | | | April 21, 2025—Project Course EE 299—Prof. Uttama Lahiri**A logo with text and a star  AI-generated content may be incorrect.** |  | | | |
|  | | |  | | |  | | |

# Introduction:

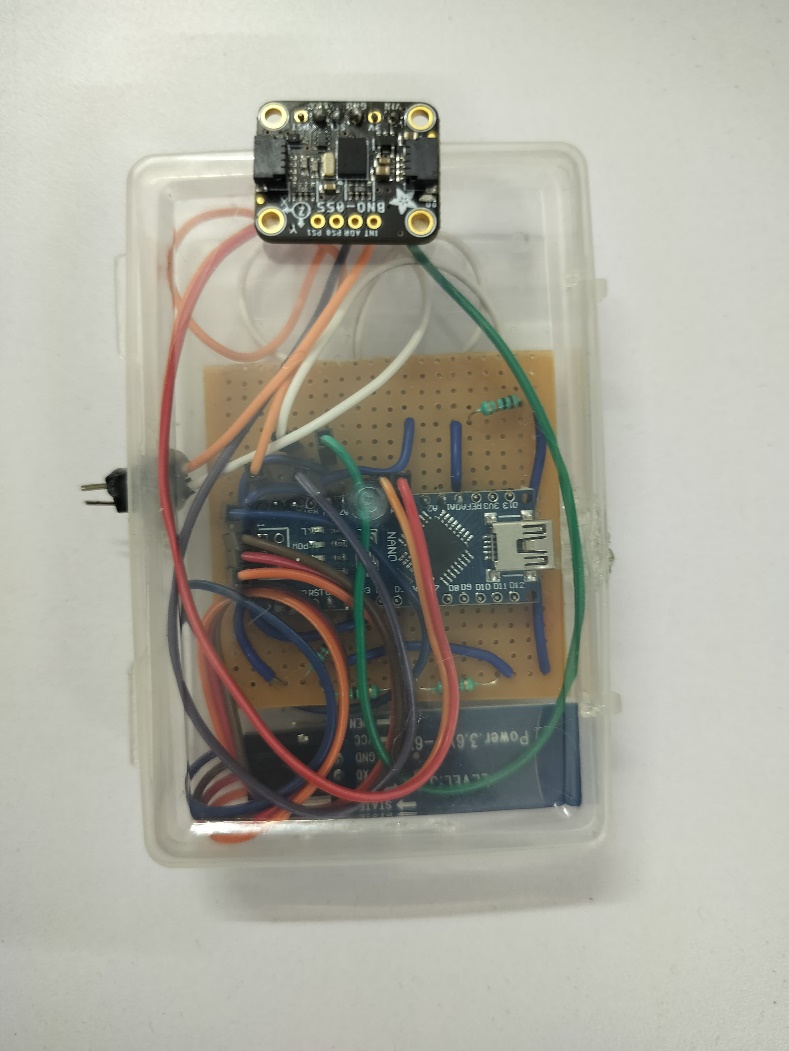
# We have created two practical smart wireless systems. The first one is designed for real-time detection and monitoring of human posture. It uses sensor-based feedback to detect and monitor the user's body inclination during balance-related tasks. The system uses an IMU (Inertial Measurement Unit) sensor BNO 055 and send feedback to the Virtual Environment. The second system is designed for the real-time detection of the leg lift. It uses FSR (Force Sensing Resistor) sensor-based feedback to detect incorrect posture or leg lift during balance-based tasks. The system triggers a buzzer when a user’s foot unintentionally lifts off the balance board.

# Experimental Setup:

Equipment and Tools:

* 2 × Arduino nano
* 2 × Arduino UNO
* Adafruit BNO 055 Absolute Orientation Sensor
* 2 × Tekscan FlexiForce A201/1 Flex Sensor
* 4 × HC-05 Bluetooth modules
* 14 × Resistors
* 5 V Battery
* Zero PCB
* Jumper wires
* Vizard Virtual Environment
* Arduino IDE

# Methodology:

 A plastic box with wires

AI-generated content may be incorrect.

Figure 1 : wireless posture detection system (left) and wireless leg lift detection system (right)

## Circuit diagrams and Codes

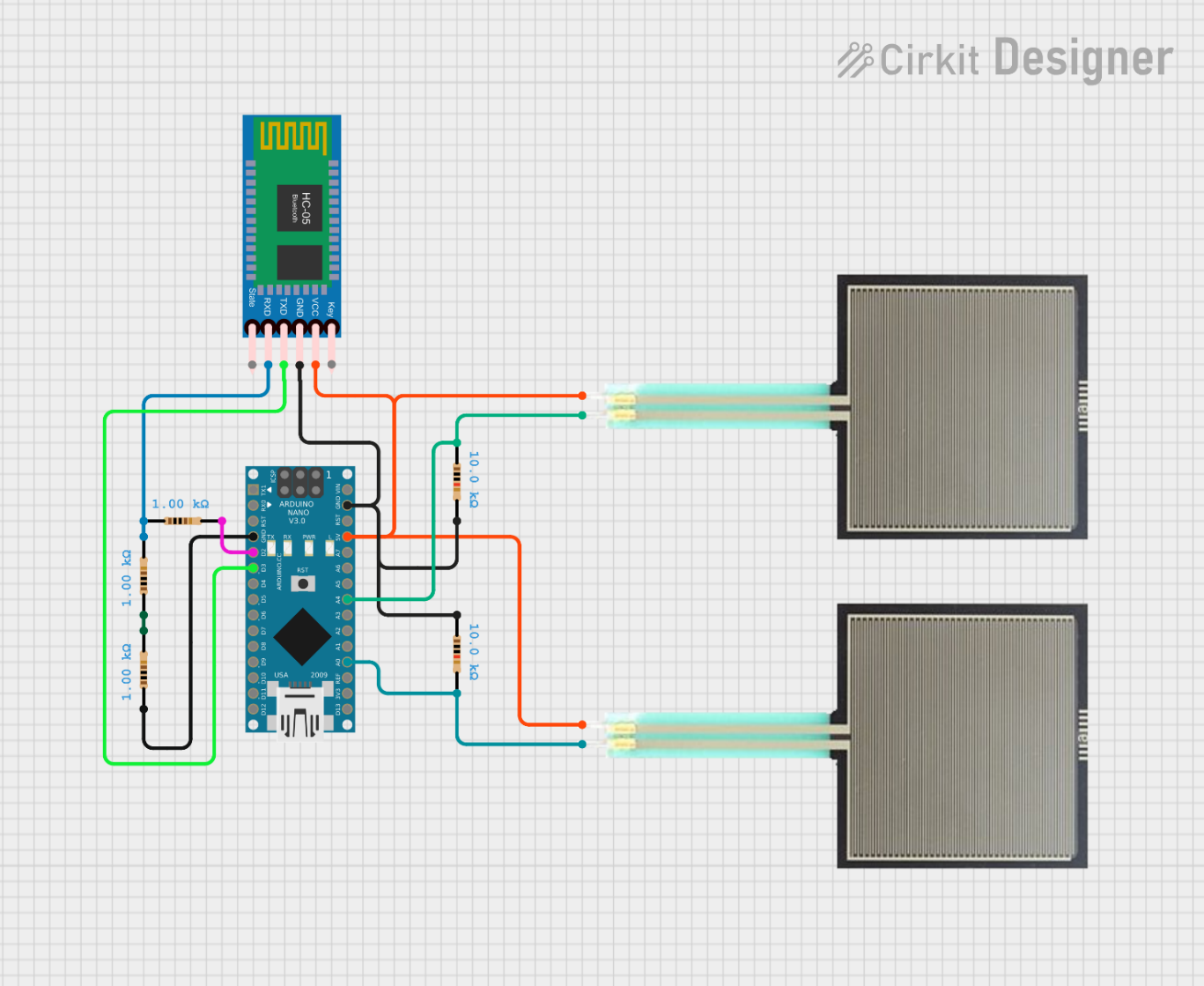


Figure 2: Master(sender) Arduino nano for the fsr sensor

1. Master of the fsr sensor (To send the data of the fsr sensor in digital values)

#include <SoftwareSerial.h>

SoftwareSerial BTSerial(3, 2);  // HC-05 TX → Pin 2, RX → Pin 3

void setup() {

  Serial.begin(38400);  // For debugging

  BTSerial.begin(38400);  // For HC-05

}

void loop() {

  int sensorValue1 = analogRead(A0);  // Read from A0

  int sensorValue2 = analogRead(A4);  // Read from A4

  // Send values in CSV format: "value1,value2"

  BTSerial.print(sensorValue1);

  BTSerial.print(",");

  BTSerial.println(sensorValue2);

  Serial.print("Sent: ");

  Serial.print(sensorValue1);

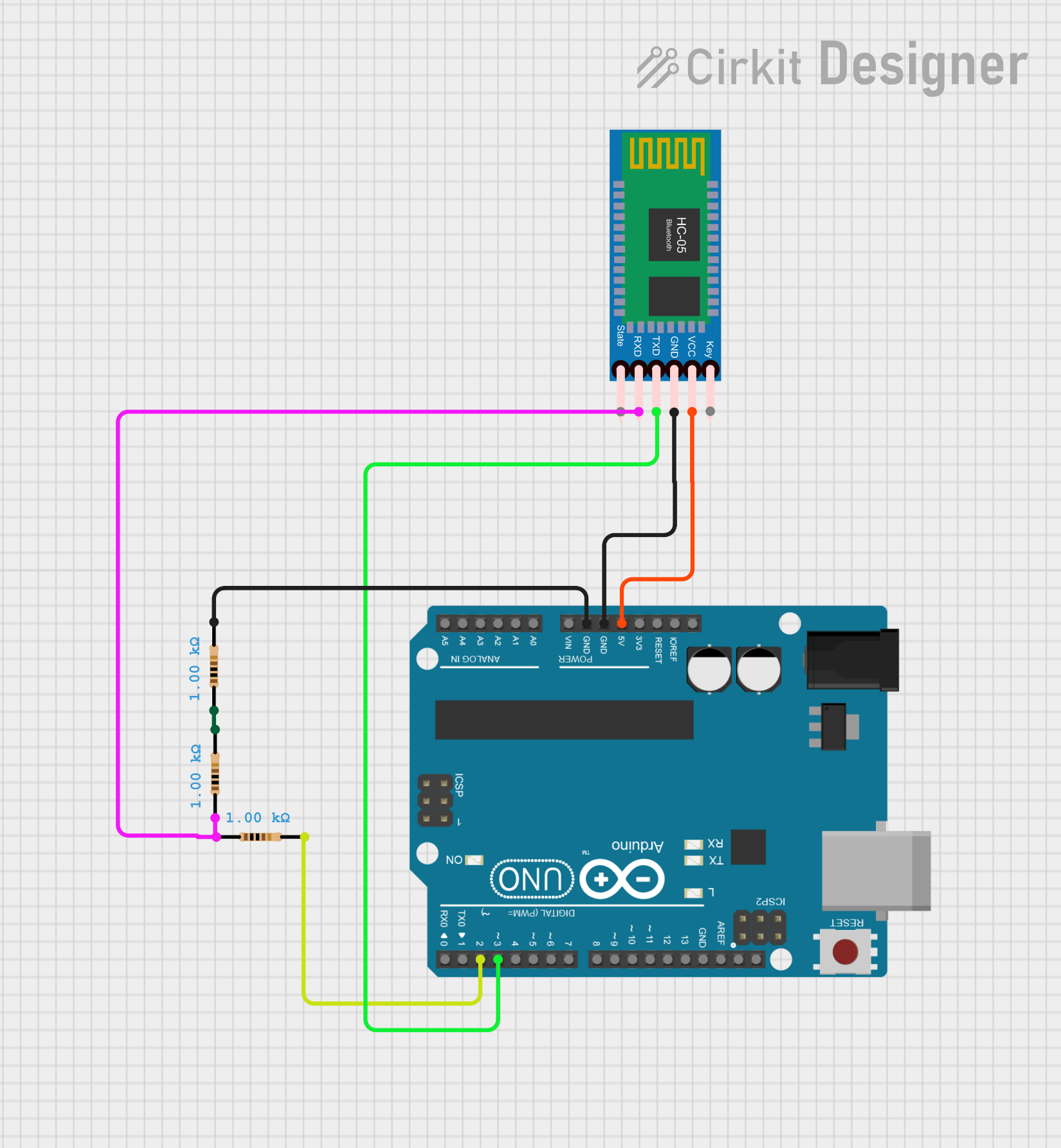
  Serial.print(", ");

  Serial.println(sensorValue2);

  delay(400);

}

1. First of all We connected this circuit as shown in the figure 2.
2. We used here 3 resistor in the Arduino nano because Hc05 Bluetooth module operates on the 3.3 volt but the Arduino nano supplies 5 volt.
3. To set modes of the Hc05 we need to connect the USB 2.0 to TTL (time to live serial convertor).
4. To connect master slave Hc05 we first go into the AT command mode of the Hc05.
   1. AT+ROLE=1 (To set it as master)
   2. AT+CMODE=0 (To connect the module to the specified Bluetooth address and this Bluetooth address can be specified by the binding command)
   3. AT+ADDR? (It shows the address of the master which is [+ADDR:0022:03:011976]).
   4. AT+BIND=0020,10,0860F1 (It connects with the respective address of the slave.)
   5. AT+UART=38400,0,0 (To fix the baud rate at 38400)
5. After all this the master (sender) send the 2 values of the analog pins via bluetooth module.
6. This two analog values are representation of the force values from the 0 to 1023 digital units.



*Figure 3: Slave (Reciever) Arduino nano for the fsr sensor*

1. Slave of the fsr sensor (To collect the data from the master)

#include <SoftwareSerial.h>

SoftwareSerial BTSerial(9, 8); // HC-05 TX → Pin 8, RX → Pin 9

void setup() {

  Serial.begin(38400);  // Debugging

  BTSerial.begin(38400); // HC-05 on pins 9,8

}

void loop() {

  if (BTSerial.available()) {

    String receivedData = BTSerial.readStringUntil('\n'); // Read full data line

    Serial.print("Received: ");

    Serial.println(receivedData);

  }

}

1. First of all We connected this circuit as shown in the figure 3.
2. We used here 3 resistor in the Arduino nano because Hc05 Bluetooth module operates on the 3.3 volt but the Arduino nano supplies 5 volt.
3. To set modes of the Hc05 we need to connect the USB 2.0 to TTL (time to live serial convertor).
4. To connect master slave Hc05 we first go into the AT command mode of the Hc05.
   1. AT+ROLE=0 (To set it as slave)
   2. AT+CMODE=0 (To connect the module to the specified Bluetooth address and this Bluetooth address can be specified by the binding command)
   3. AT+ADDR? (It shows the address of the master which is [+ADDR: 0020:10:0860F1]).
   4. AT+BIND=0022,03,011976 (It connects with the respective address of the Master.)
   5. AT+UART=38400,0,0 (To fix the baud rate at 38400)
5. After all this the master (sender) send the 2 values of the analog pins via bluetooth module.
6. We get the data received from Hc05 through the digital pin 3 and 2.

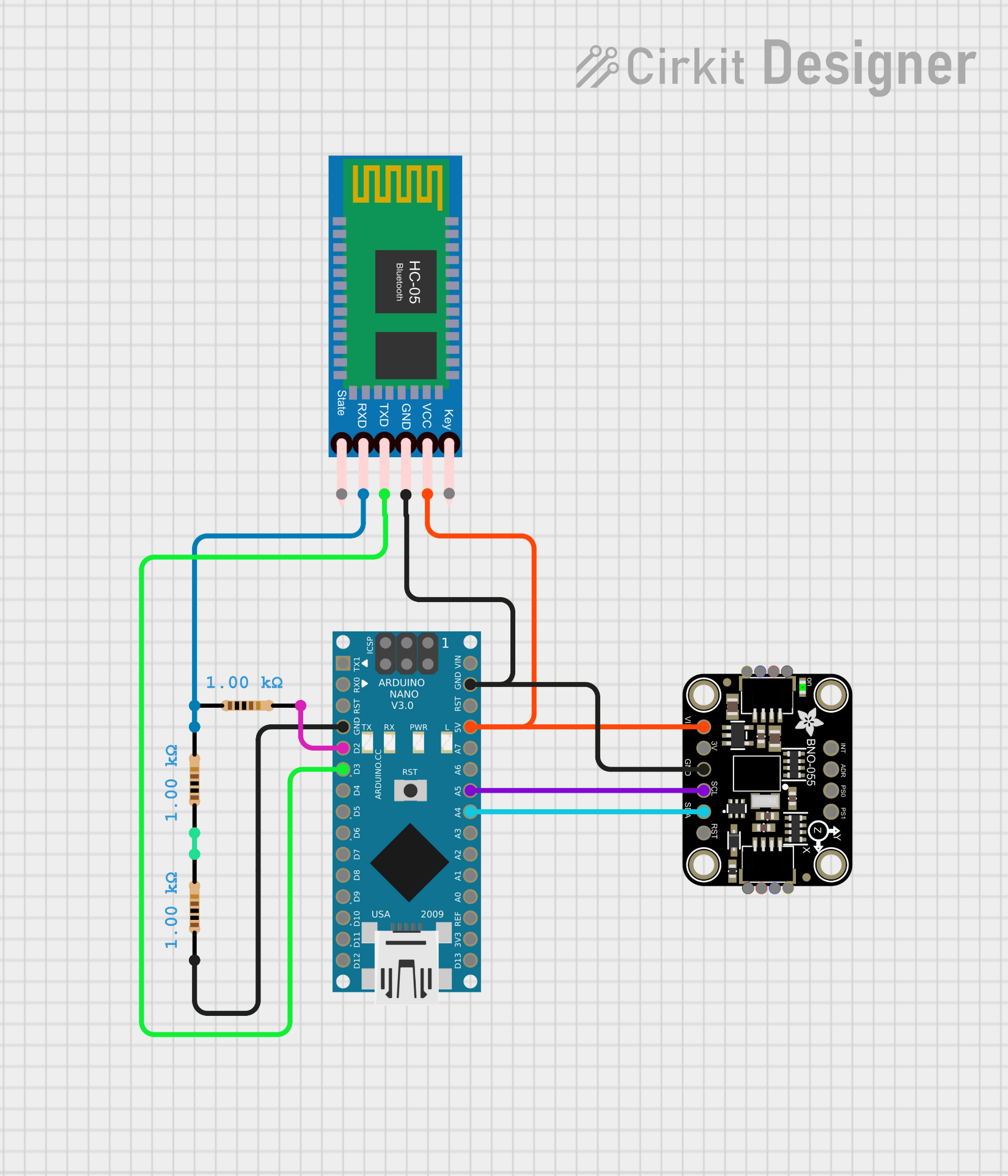


Figure 4: Master(sender) Arduino nano for the IMU sensor

1. Master of the BNO005 IMU sensor (To send the data of the IMU sensor)

#include <Wire.h>

#include <Adafruit\_Sensor.h>

#include <Adafruit\_BNO055.h>

#include <SoftwareSerial.h>

SoftwareSerial BTSerial(3, 2);  // RX, TX (Connected to HC-05 Master)

Adafruit\_BNO055 bno = Adafruit\_BNO055(55);

void setup() {

    Serial.begin(38400);       // For debugging

    BTSerial.begin(38400);     // For HC-05 communication

    Wire.begin();

    if (!bno.begin()) {

        Serial.println("No BNO055 detected!");

        while (1);

    }

    delay(400);

    bno.setExtCrystalUse(true);

}

void loop() {

    sensors\_event\_t event;

    bno.getEvent(&event);

    float x = event.orientation.x;

    float y = event.orientation.y;

    float z = event.orientation.z;

    Serial.print("X: "); Serial.print(x);

    Serial.print(" Y: "); Serial.print(y);

    Serial.print(" Z: "); Serial.println(z);

    // Send data to Slave HC-05

    BTSerial.print(x); BTSerial.print(",");

    BTSerial.print(y); BTSerial.print(",");

    BTSerial.println(z);

    delay(400);

}

1. First of all We connected this circuit as shown in the figure 4.
2. We used here 3 resistor in the Arduino nano because Hc05 Bluetooth module operates on the 3.3 volt but the Arduino nano supplies 5 volt.
3. To set modes of the Hc05 we need to connect the USB 2.0 to TTL (time to live serial convertor).
4. To connect master slave Hc05 we first go into the AT command mode of the Hc05.
   1. AT+ROLE=1 (To set it as master)
   2. AT+CMODE=0 (To connect the module to the specified Bluetooth address and this Bluetooth address can be specified by the binding command)
   3. AT+ADDR? (It shows the address of the master which is [+ADDR:0020:10:08428E]).
   4. AT+BIND= 0019,10,091A07 (It connects with the respective address of the slave.)
   5. AT+UART=38400,0,0 (To fix the baud rate at 38400)
5. SDA pin of the BNO 055 is connected to A4 pin of the Arduino nano. SCL pin of the BNO 055 is connected to the A5 pin of the Arduino nano. We can not connect these pins to any other analog pin because these pins are designed on the basis of I2C (Inter-Integrated Communication) protocol.
6. After all this the master (sender) send the 2 values of the analog pins via bluetooth module.
7. These two analog values are representation of the force values from the 0 to 1023 digital units.

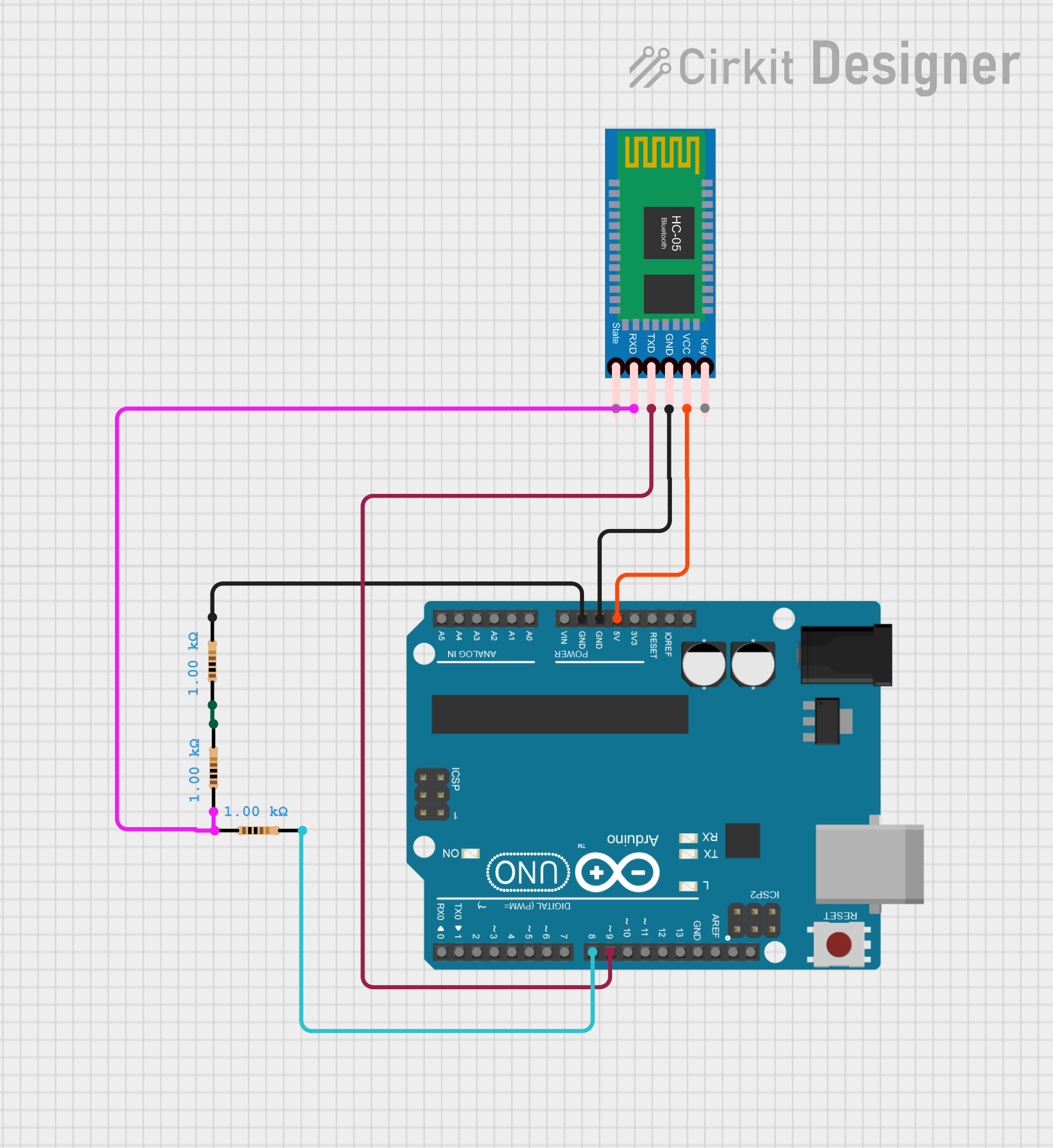


Figure 5: Slave (Reciever) Arduino nano for the IMU sensor

1. Slave of the BNO005 IMU sensor (To collect the data from the IMU sensor)

#include <SoftwareSerial.h>

SoftwareSerial BTSerial(3, 2);  // RX, TX (Connected to HC-05 Slave)

void setup() {

    Serial.begin(38400);  // Debugging

    BTSerial.begin(38400); // HC-05 communication

}

void loop() {

    if (BTSerial.available()) {

        String receivedData = BTSerial.readStringUntil('\n');

        Serial.println("Received: " + receivedData);

    }

}

1. First of all We connected this circuit as shown in the figure 3.
2. We used here 3 resistor in the Arduino nano because Hc05 Bluetooth module operates on the 3.3 volt but the Arduino nano supplies 5 volt.
3. To set modes of the Hc05 we need to connect the USB 2.0 to TTL (time to live serial convertor).
4. To connect master slave Hc05 we first go into the AT command mode of the Hc05.
   1. AT+ROLE=0 (To set it as slave)
   2. AT+CMODE=0 (To connect the module to the specified Bluetooth address and this Bluetooth address can be specified by the binding command)
   3. AT+ADDR? (It shows the address of the master which is [+ADDR: 0019:10:091A07]).
   4. AT+BIND=0020,10,08428E (It connects with the respective address of the Master.)
   5. AT+UART=38400,0,0 (To fix the baud rate at 38400)
5. After all this the master (sender) send the 2 values of the analog pins via bluetooth module.
6. We get the data received from Hc05 through the digital pin 3 and 2.

# Results and Analysis:

All the individual systems were successfully integrated into a combined setup for acquiring muscle synergy data. The system effectively detects leg lifts and measures posture in real-time.

I encountered several challenges while designing the circuits and developing the virtual environment, but these experiences significantly contributed to my learning and problem-solving skills.

In the future, the posture measurement system can be extended to full-body movement analysis. This would allow for mapping complete body movements within the virtual environment, enabling more immersive assessment.

This system can also be utilized as a balance training tool to enhance motor abilities in post-stroke patients, aiding in their rehabilitation process through interactive and engaging feedback.