CLASSROOM-EXO MANUAL

A Comprehensive Guide to Classroom-Exo's Electronics Assembly

ABSTRACT

Classroom-Exo Manual is a step-by-step instruction to optimize the integration of essential components, from the PCB design to connecting sensors and servo motors. It gives detailed explanations and practical guidance, empowering users with a deep understanding of the system's capabilities.

Chapter 5

Electronics

In this comprehensive guide, we will walk you through the step-by-step process of assembling the electronics for your exoskeleton and applying essential improvements. Consider this manual as an informative substitute for Chapter 5 in the EduExo Pro Handbook. It serves as a detailed reference to ensure a seamless and enhanced assembly experience.

1. Overview and Functionality of the Components

The main electronic components of the Classroom Exo (Figure 1) are the PCB, the Power supply, the motor with an integrated angle sensor, the force sensor, an amplifier for the force sensor, the EMG sensors, and the microcontroller. The microcontroller is connected to a computer for programming.

1.1 Printed Circuit Board (PCB)

The heart of the Classroom Exo system is the PCB. The Classroom Exo's PCB enhances the exoskeleton's performance by upgrading to a 2-cell Li-Polymer Battery with Battery Management, ensuring optimal power for servo operation. With easily replaceable batteries, maintenance is simplified. The streamlined connection process and safety features make Classroom Exo more user-friendly, especially for educators and students. To understand how a PCB works, let's talk about some important components:

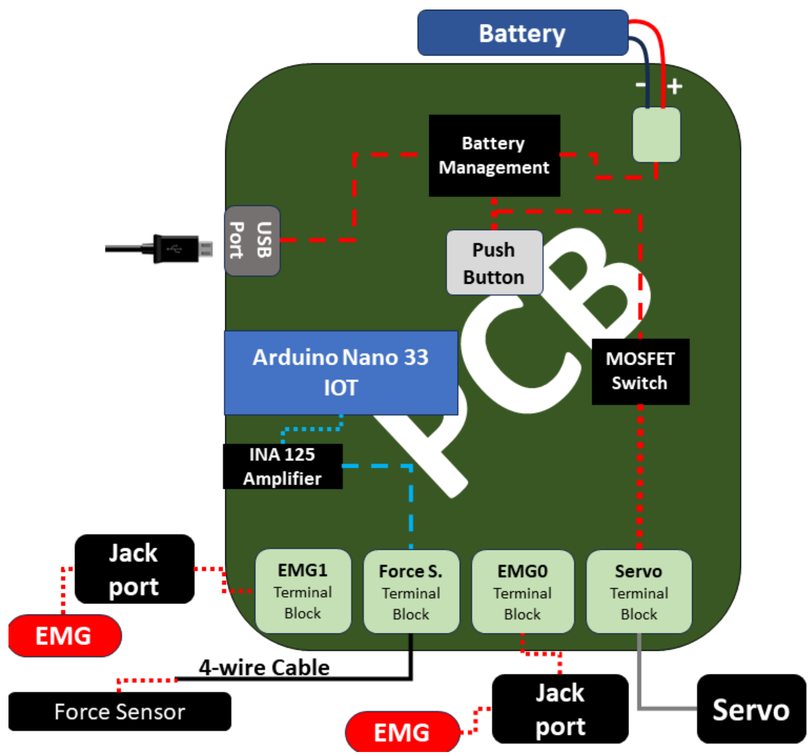

Fig. 1. The schematic illustrates the key components of the Classroom Exo PCB. Featured components include the BQ25886 battery management IC for efficient power regulation, Arduino Nano IoT 33 for control and communication, a USB port for battery charging, a MOSFET switch for servo control, and 5 terminal blocks facilitating connections for battery, EMG sensors, servo, and force sensor.

1.1.1 Resistors & Capacitors

Resistors control the flow of electricity in a circuit. They can limit the current and voltage or divide voltage in specific parts of the circuit. Imagine them as traffic controllers, directing the flow of electricity. They're also like volume knobs, adjusting the electric power in different areas. In circuits, resistors are often used to make sure devices like transistors work just right. They can even act as bodyguards, protecting components from too much electricity.

Additionally, Capacitors are like energy storage units in a circuit. They help smooth out bumps in electric power, making sure everything runs steadily. When there's a brief interruption in power, capacitors step in and provide a quick boost. They're also excellent at cleaning up messy signals. Think of them as noise-canceling headphones for electronic signals. In power supplies, capacitors are like stabilizers, keeping the voltage at a steady level. They're handy in amplifiers and other circuits, acting like traffic barriers to allow only certain signals to pass through.

1.1.2 Battery Management (BQ25886)

The BQ25886, integrated into the PCB design of the Classroom Exo, serves as a Battery Management IC with multifaceted functionalities. Primarily, it facilitates efficient charging of the 2-cell Li-Polymer battery, optimizing power delivery to the exoskeleton. Its Instant Charge feature ensures rapid activation, providing immediate power when needed. The BQ25886 contributes to the longevity of the battery by precisely controlling the charging process, preventing overcharging or undercharging.

Moreover, the IC includes safety mechanisms, indicated by the integrated LEDs on the exoskeleton. The blue LED signifies ongoing charging, low charge, or charge completion, while the red LED indicates a reliable input source. In case of faults, the blue LED blinks, alerting users to potential issues. The BQ25886 acts as a robust guardian for both the microcontroller and the battery, ensuring the reliable and safe operation of the Classroom Exo.

1.1.3 USB Port & Power Supply

We utilize a 2-cell Li-Polymer battery that supplies a voltage ranging from 7.4 to 8.4 volts, providing ample power to operate the servo and other components at their maximum capacity. Additionally, the *USB micro-B* connector is vital for charging the battery. Simply connect an external power source like a USB power adapter to start charging.

1.1.4 MOSFET Switch

The MOSFET (Metal-Oxide-Semiconductor Field-Effect Transistor) switch on the Classroom Exo's PCB is responsible for controlling the power supply to the servo motor. It acts as an electronic switch that can turn the power on or off based on the control signals received from the microcontroller.

In the Classroom Exo's PCB, we use two MOSFETs to handle power efficiently. The P-Channel MOSFET (Si3483CDV) acts as a switch between the battery management input (Source) and the servo (Drain). The microcontroller controls the gate of this MOSFET through an N-Channel MOSFET (BSS123), ensuring precise control of power distribution. This setup improves control and optimizes power usage in the exoskeleton system.

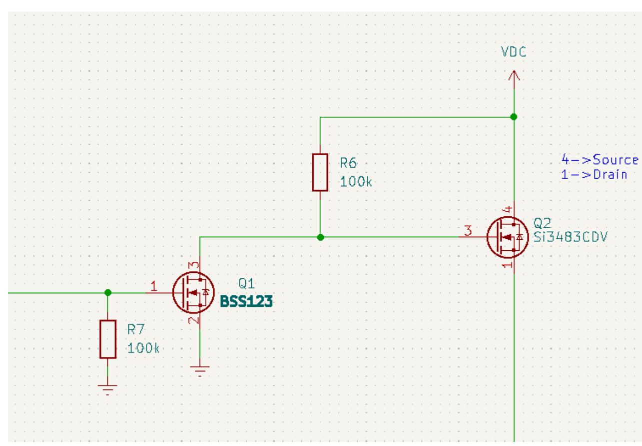

Fig. 2. MOSFET Switch (Si3483CDV) in Classroom Exo PCB: Controls power flow from the battery to the servo with microcontroller precision through N-Channel MOSFET (BSS123)

1.1.5 Pushbutton Switch (LP11)

The Lp11 is a Latching Non-Tactile RGB Pushbutton Switch designed for the microcontroller, sensors, and other components in the Classroom Exo's PCB. With RGB illumination options, you have the flexibility to change button colors and assign specific colors to different functionalities, enhancing user interaction and system feedback.

1.1.6 Test Points for Error System

In the Classroom Exo's PCB, test points are strategically placed locations that allow you to easily measure and monitor signals or voltages at specific points in the circuit. These points serve as convenient checkpoints for testing the functionality of different components and ensuring that the PCB is operating as intended.

1.1.7 Terminal Blocks

The Classroom Exo's PCB incorporates five terminal blocks, each serving a specific purpose:

- Servo Terminal Block (4 Positions): Connect the servomotor securely using the 4-position terminal block.
- Force Sensor Terminal Block (4 Positions): The 4-position terminal block is dedicated to the force sensor, ensuring a reliable connection.
- EMG Sensor Terminal Blocks (3 Positions Each): Utilize two 3-position terminal blocks for the EMG sensors, providing organized connections.
- Battery Terminal Block (2 Positions): The 2-position terminal block facilitates a secure connection to the battery.

Terminal blocks play a crucial role in the Classroom Exo system, offering a secure and organized method to connect and disconnect external wires. They ensure a reliable interface with the circuit, simplifying the overall wiring process. We used Screwless Terminal Blocks to make it easy to take out and insert in the wires.

1.2 Microcontroller

The microcontroller, specifically the Arduino Nano 33 IoT serves as the brain, orchestrating the various components within the exoskeleton. Additionally, it plays multiple roles by reading signals from sensors, controlling motor movements, and managing overall system operations. The microcontroller is easily programmable and modifiable, making it accessible to users with various levels of programming experience. it is also equipped with serial communication capabilities; the microcontroller communicates with other components in the system to ensure synchronization and smooth operation.

1.3 Motor and Angle Sensor

The Classroom Exo employs an electric servomotor, specifically the CYS-S0650 55KG model_ which is the same servo in EduExo Pro. This servomotor includes gear and control electronics housed within, providing reliable performance. The servomotor incorporates an integrated potentiometer to precisely measure the rotation angle (position) of the motor's shaft.

The servomotor simplifies wiring with four color-coded wires:

- Power Supply Inputs: Red and brown wires
- Control Signal Input: Orange wire

- Angle Signal Output: White wire

1.4 Force Sensor

The force sensor measures the interaction force between the exoskeleton and the user. It can be used to record the interaction and to control the exoskeleton. The force sensor is integrated into the exoskeleton so it is directly affected by interaction forces. When force is applied, the metal beam and attached strain gauges are deformed, changing the electric resistance of the strain gauges. These changes in resistance are measured in a Wheatstone bridge circuit: two of the wires of the sensor are for the supply voltage V_s, and the other two are the output voltage V_{out}, (the sensor signal). Because the output voltage is very small, it is connected to an amplifier chip. This chip is then connected to the Arduino microcontroller that reads the amplified signal.

1.5 Electromyography (EMG) Sensor

The Classroom Exo incorporates two EMG (Electromyogram) sensors to monitor muscle activity in the biceps and triceps. These sensors are seamlessly integrated into the system through dedicated plugs connected to the PCB via terminal blocks.

We've chosen the Myoware EMG sensor for its user-friendly features. Myoware comes equipped with built-in circuits that effectively filter muscle signals, ensuring precise readings.

To enhance user flexibility, a shield with a long external wire is added to the EMG sensor. This allows users to freely position the sensor for optimal comfort during use. The extended wire accommodates varied placements, providing convenience in setting up the electrodes on the skin.

2. Preparation

Through this chapter, we will explain how to get our PCB design, how to solider the components to the PCB, and how to 3d print the upper-arm cover that contains the PCB, Servo, and the sensors' connections.

2.1 PCB and Components

Our commitment to accessibility led us to choose the open-source application KiCad for designing the PCB of the Classroom Exo. This decision ensures that all students and educators can benefit from the improvements made to the Exoskeleton.

Accessing the PCB Design

1) Download PCB Design File:
   - Visit our GitHub repository at www.github.fabian.
   - Locate and download the Classroom Exo PCB design file.
2) Printing the PCB:
   - Send the downloaded PCB file to your preferred PCB printing company.
   - Most companies prefer the Gerber file format; you can find it in the "fabrication" folder of our design.

Customization Option

Feel free to tailor the PCB design to your preferences. If you choose to make modifications, create your own Gerber file based on the changes you implement.

Ordering Components:

In addition to ordering the PCB, ensure you order the necessary components. Refer to our Excel sheet[1], providing details on required components, quantities, and links to potential suppliers

2.2 Soldering Components to the PCB: A Step-by-Step Guide

This guide aims to simplify the soldering process, starting with Surface-Mount Device (SMD) components before moving on to Dual In-line Package (DIP) components.

Required Supplies:

- Solder wire
- Solder paste
- Flux
- Soldering station
- Reflow oven/Hot air gun (typically included with the soldering station)
- Multimeter (recommended)
- IPA (Isopropyl Alcohol) for cleaning (optional)

2.2.1 SMD and DIP Soldering Steps:

To simplify the process, adhere to this sequence and consult the provided table for the placement of components (Fig.3). Each component on the PCB is assigned a *Footprint Reference*, and this identifier is marked on the front

[1] www.github.fabian

side of the component's footprint. The Footprint Reference typically begins with a letter indicating the component type, such as R for resistors and C for capacitors.

Item	Qty	Reference(s)	Value	Footprint
3	5	C1, C11, C13, C15, C18	1uF	Capacitor_SMD:C_0805_2012Metric
4	1	C2	6.8uF	Capacitor_SMD:C_0805_2012Metric
5	3	C3, C6, C16	10uF	Capacitor_SMD:C_0805_2012Metric
6	1	C4	47nF	Capacitor_SMD:C_0805_2012Metric
7	1	C5	4.7uF	Capacitor_SMD:C_0805_2012Metric
8	2	C7, C8	22uF	Capacitor_SMD:C_0805_2012Metric
9	5	C9, C10, C12, C14, C17	100nF	Capacitor_SMD:C_0805_2012Metric
10	1	D1	BAT60B	Diode_SMD:D_SOD-323
11	1	D2	LED_Blue	LED_SMD:LED_0603_1608Metric
12	1	D3	LED_RED	LED_SMD:LED_0603_1608Metric
13	1	D4	D1213A	Diode_SMD:D_SOD-323
14	2	D5, D6	ESD321	Diode_SMD:D_0402_1005Metric
15	4	H1, H2, H3, H4	M3	MountingHole:MountingHole_3.2mm_M3
16	1	J1	EMG0	EDUEXO-custom-PCB:TE_1-2834011-3
17	1	J2	EMG1	EDUEXO-custom-PCB:TE_1-2834011-3
18	1	J3	SG0	EDUEXO-custom-PCB:TE_1-2834011-4
19	1	J4	SRV0	EDUEXO-custom-PCB:TE_1-2834011-4
20	1	J5	USB_B_Micro	EDUEXO-custom-PCB:USB_micro_B_10118192-0002LF
21	1	J6	BAT0	EDUEXO-custom-PCB:TE_1-2834011-2
22	1	L1	1uH	EDUEXO-custom-PCB:L_0630CDMCCDS-1R5MC
23	1	Q1	BSS123	Package_TO_SOT_SMD:SOT-23
24	1	Q2	Si3483CDV	Package_TO_SOT_SMD:TSOT-23-6
25	1	R1	60.4R	Resistor_SMD:R_0805_2012Metric
26	3	R2, R3, R4	10k	Resistor_SMD:R_0805_2012Metric
27	2	R6, R7	100k	Resistor_SMD:R_0805_2012Metric
28	1	R8	383R	Resistor_SMD:R_0805_2012Metric
29	1	R9	Vset	Resistor_SMD:R_0805_2012Metric
30	1	R10	Iset	Resistor_SMD:R_0805_2012Metric
31	1	R11	5.24k	Resistor_SMD:R_0805_2012Metric
32	1	R12	30.31k	Resistor_SMD:R_0805_2012Metric
33	1	R13	300	Resistor_SMD:R_0805_2012Metric
34	1	R14	170	Resistor_SMD:R_0805_2012Metric
35	1	TH1	Thermistor_NTC	Resistor_SMD:R_0805_2012Metric
36	1	U1	INA125P	DIP794W45P254L1969H508Q16
37	1	U2	BQ25886RGER	EDUEXO-custom-PCB:Texas_S-PVQFN-N24_EP2.1x2.1mm_ThermalVias
38	1	U3	LP11EE1NCSRGB	EDUEXO-custom-PCB:SW_LP11EE1NCSRGB

Fig. 3. Table of Components and Footprint References: A comprehensive guide to all PCB components, their respective locations, through their Footprint References for efficient assembly in the Classroom Exo project.

Stage 1:

1. Start with inductors, capacitors, and resistors.
2. Proceed to solder the Battery Management IC (BQ25886).
3. Solder the USB port and the (blue, and red) LED indicators.
4. Solder the 2-position terminal block (battery terminal block).
5. Solder the upper test points for voltage (red) and ground (black).
6. Test component functionality by connecting the battery to the terminal block.
7. Verify voltage at the upper test points using a multimeter.
8. Check USB port functionality by connecting a USB wire and ensuring LEDs light up.

9. After testing, disconnect the battery, avoiding contact with the board to prevent damage.

Notes:

- For any issues with the multimeter readings, particularly around the battery management IC, recheck soldering.
- Verify that the blue LED doesn't blink during USB charging; if it does, inspect the battery or charging adapter.
- Always disconnect the battery after testing to prevent potential damage to the board and components.

Stage 2:

1. Begin with MOSFET SWITCH and diodes, completing SMD soldering.
2. Solder Arduino's strip connectors socket.
3. Solder INA 125 amplifier socket.
4. Solder the lower terminal blocks.
5. Solder the lower test points: white for the EMG sensor, orange for the force sensor, yellow for servos, red for VDD voltage, and black for GND.
6. Solder the Lp11 pushbutton on the other side of the PCB. By completing these steps, all components on the PCB should be soldered.
7. Place Arduino and INA 125 amplifiers into their respective sockets.

2.2.2 General Tips

- Temperature Control: Adjust the soldering iron temperature based on the component and PCB requirements.
- Practice on Scrap PCBs: Practice soldering on scrap PCBs before working on your project.
- Steady Hands: Keep your hands steady, and take breaks to avoid fatigue.
- Safety: Use safety precautions, such as eye protection and proper ventilation.

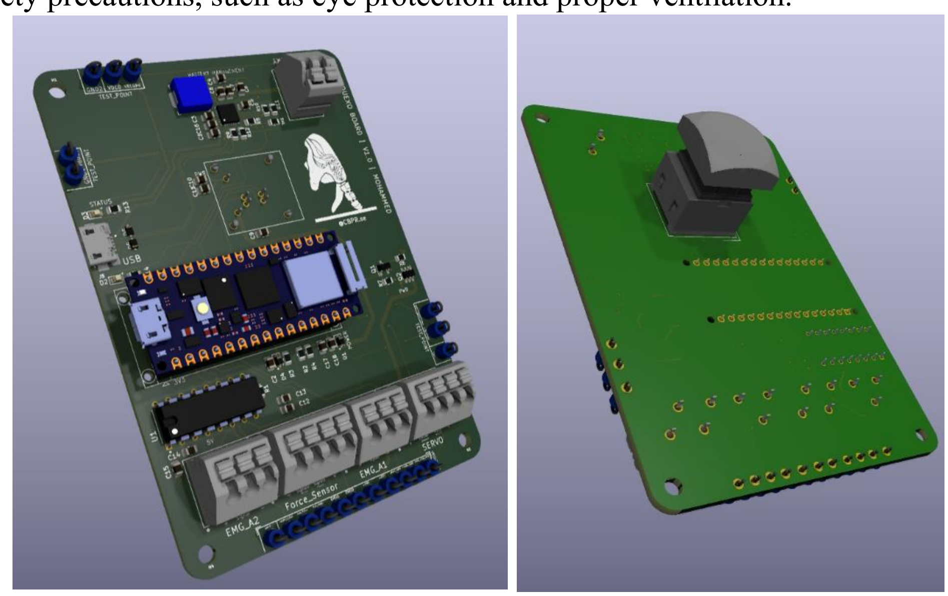

Fig. 4. Front and Back View of the Classroom Exo PCB: An illustration showcasing the meticulously soldered components on both the front and back sides of the PCB.

2.3 Upper-Arm Cover

The upper arm cover of the Classroom Exo houses all the crucial electronic components. Our focus on design refinement led to the development of a Printed Circuit Board (PCB) to replace the prototype board wire connection. In this section, we will guide you through the assembly process, dividing it into the sidewall and frontside parts.

2.3.1 Manufacturing Sidewall Part

3D Printing:

If you have a *FlashForge* printer, access the GX file here. Alternatively, obtain the STL file here. Use your preferred slicing application to prepare the part for printing, adjusting settings according to your 3D printer specifications.

2.3.2 Manufacturing Frontside Part:

To enhance the exoskeleton's aesthetics, the front side is crafted from acrylic material using a laser cutter. The laser cutter precisely cuts the material to the required dimensions, and logos and labels are engraved for a polished finish.

For laser-cutting, no worries if you don't have access to one. You can 3D print the front side using this file here.

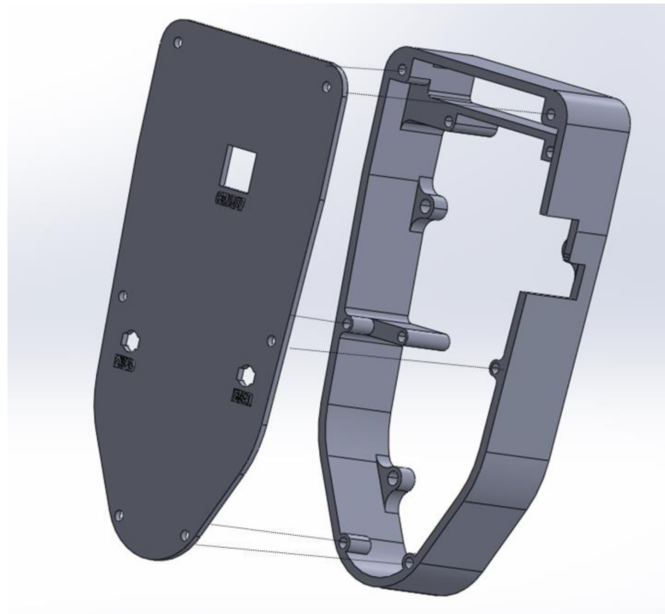

Fig. 5. In our development the Upper-arm Cover is contain of two main parts. The sidewall and The frontside part

NOTE: Ensure to follow the appropriate safety measures while operating 3D printers and laser cutters.

3. Components' Software Test

After finishing working with the PCB and the Upper-arm Cover, you should start testing the sensors using the Arduino in the PCB. Installing the Arduino IDE

Before you can start programming the microcontroller, you have to install the Arduino Integrated Development Environment (IDE). You can download it for several operating systems on the Arduino website here.

When you install and open the IDE (Figure), you will find everything you need to write code and upload it to the Arduino board. To program the Arduino microcontroller, you have to connect it to your computer with a USB A-B cable. You can refer to this link here to get the files to check the servo and sensor using Arduino.

3.1 Testing Servo Motor

3.1.1 Connecting Servo

- ❖ **Power Supply Connection:**
  - Connect the motor's red wire to the terminal block's pole labeled (+).
  - Connect the motor's brown wire to the terminal block's ground pole (-).
- ❖ **Angle Sensor Connection:**
  - Connect the white wire of the potentiometer angle sensor to the pole labeled "Position Sensor."

❖ **Position Control Connection:**

- Connect the control signal wire (orange) to the pole labeled "Signal."

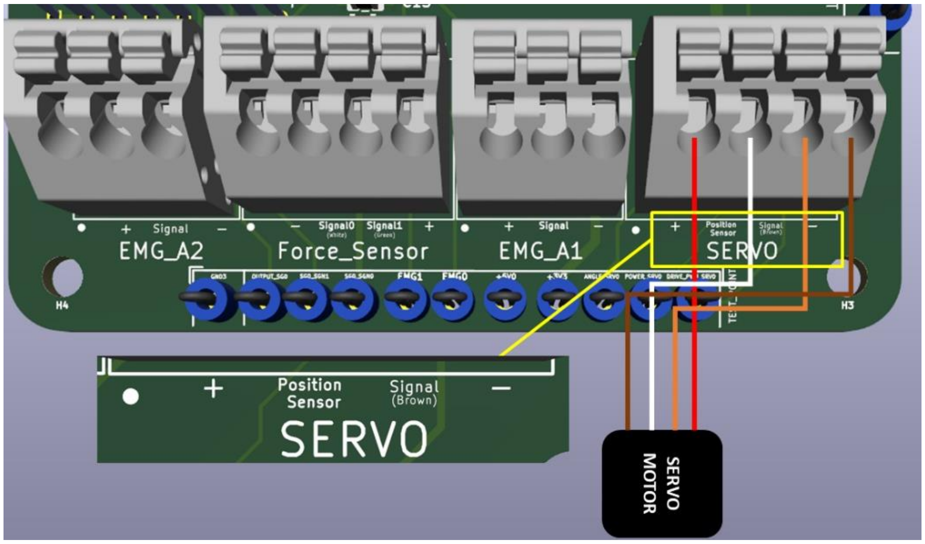

Fig. 6. Visual guide on connecting the servo motor to the terminal block

3.1.2 Controlling the Motor Angle

Now, it is time to write our first program and test the functionality of the PCB and Servo. An Arduino program is usually called a 'sketch'. Sketches contain one or several source code files that can be compiled, uploaded, and run on the Arduino. The program is designed to move the servo smoothly from 0 to 180 degrees. Refer to the accompanying figure for a visual representation of the code lines.

```
#include <Servo.h>
Servo myservo;        // create servo object to control a servo

int servo_control_pin = 5;
int pos = 0;          // variable to store the servo position

void setup() {
  pinMode(2, OUTPUT);
  digitalWrite(2, HIGH); // digitalWrite(2, LOW);    OFF means Servo is OFF // HIGH means Servo is ON
  delay(20);
  myservo.attach(servo_control_pin, 850, 2150);  // attaches the servo on pin 5 to the servo object
  myservo.write(180);

  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
    myservo.write(180 - pos);            // tell servo to go to position in variable 'pos'
    delay(15);                         // waits 15ms for the servo to reach the position
  }
  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(180-pos);              // tell servo to go to position in variable 'pos'
    delay(15);                         // waits 15ms for the servo to reach the position
  }
  myservo.write(180);
}
```

Fig. 7. Arduino program to move a motor position from 0 to 180 and back.

3.1.3 Reading the Motor Angle

this program (figure) will read the position from the analog feedback sensor to measure the exoskeleton's elbow joint angle.

```
#include <Servo.h>
Servo myservo;        // create servo object to control a servo
int servoAnalogInPin = A3;
int posIs;

void setup() {
  Serial.begin(9600);
  delay(10);
}

void loop() {
  posIs = analogRead(servoAnalogInPin);
  Serial.print("Position:");
  Serial.println(posIs);
  delay(10);
}
```

Fig. 8. Arduino program to read the position sensor

3.2 Testing Force Sensor

After testing the servo, proceed to connect the Force sensor to its designated terminal block on the PCB for testing. The wire of the Force sensor is delicate, so it's advisable to connect it to the 4-wire cable first. For detailed instructions, you can refer to Chapter 5 of the EduExo Pro manual, which provides a comprehensive guide on how to handle this process.

3.2.1 Connecting The Force Sensor to the PCB

The depicted colors in (Fig. 9), correspond to the force sensor wire colors. However, when connecting the force sensor to a 4-wire cable initially, the colors may differ. To ensure accuracy, trace each force sensor wire from the cable to its corresponding terminal based on the color code presented in the figure. This step ensures a proper match and secure connection. Now, I will explain how to connect the force sensor to the terminal using the force sensor wires' colors.

Connect the red wire to the pole labeled (+), the black wire to the (-) pin, the white wire to the pole labeled Singnal0 (white wire), and lastly the green wire to the pole labeled Singnal1 (green wire). You can refer to this (Fig. 10) for a visual guide.

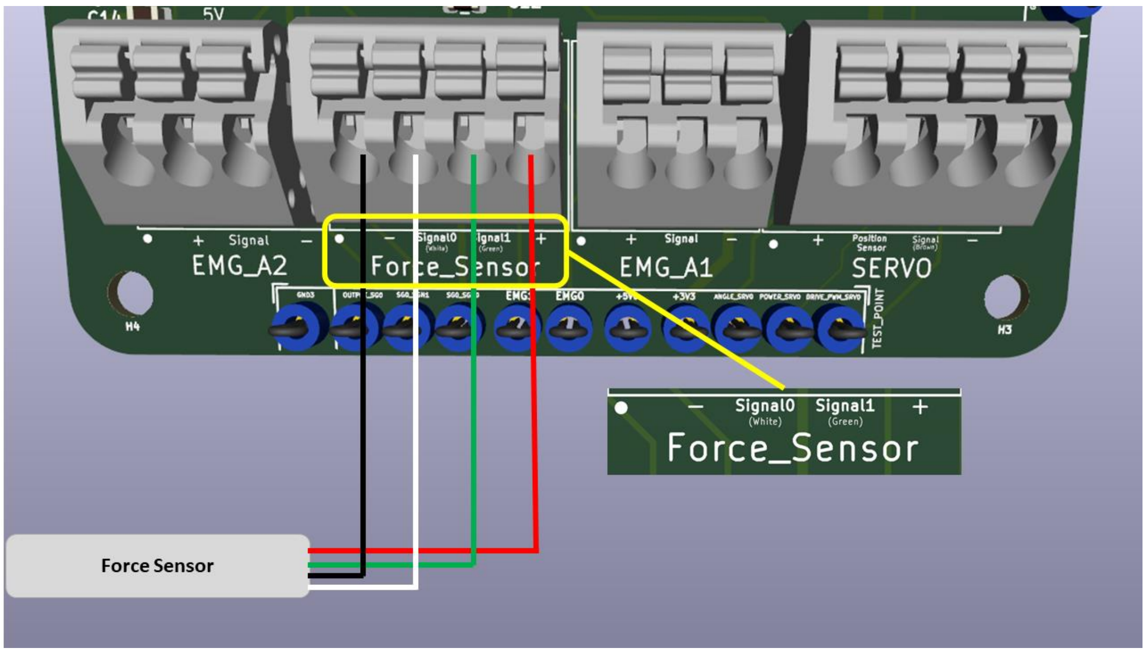

Fig. 11. Force Sensor Connection: Follow the color-coded traces to match force sensor wires from the 4-wire cable to their designated terminals for a secure and accurate connection.

3.2.2 Checking the Force Sensor

With the force sensor connected, we can now implement the program that will read the force sensor and display the measured value on the screen using the serial monitor. Reading the force data is similar to reading the motor angle. The amplified signal is connected to an analog input pin, which can be read by the following code (Fig. 12).

```
int forceAnalogInPin = A0;
float xn=0;

void setup() {
  Serial.begin(9600);
  delay(1000);
}

void loop() {
  xn=analogRead(forceAnalogInPin);
  Serial.print("Force=");
  Serial.println(xn);
  delay(10);
}
```

Fig. 13. Arduino code to check force sensor functionality

3.3 Testing EMG Sensor

As mentioned before, the Exoskeleton has two ports for two external EMG Sensors which are connected to the PCB's terminal blocks.

3.3.1 Preparing and Upgrade the EMG Sensors

One of the drawbacks of the EMG sensor's built-in electrode snaps is that it limits the user's range of movement. To address this issue, we have employed an external cable shield, which accompanies a 3-connector sensor cable linked to a Cable Shield. Proper stacking of the cable shield above the EMG sensor, as illustrated in Fig.11, is crucial. The fundamental rule of stacking involves connecting pinholes with identical labels; for instance, the "+" hole of the Sensor board should be linked to the "+" hole of other shields, and similarly, the "RAW" hole of the Sensor should be connected to the "RAW" hole of other shields, and so forth.

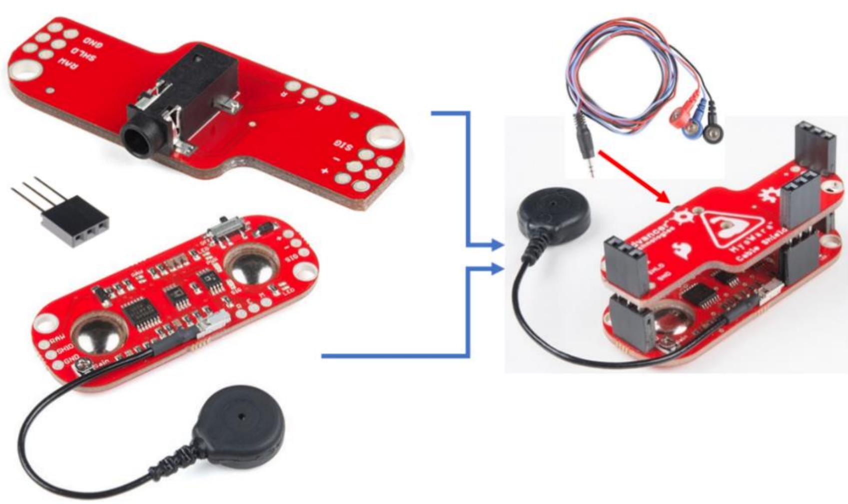

Fig. 14. EMG sensor stacked to a Cable Shield

3.3.2 Connecting The EMG Sensor to the PCB

Here, we will explain how to connect the jack socket to the terminal blocks. The jack socket has 3 pins, numbered from 1 to 3. Solder 3 wires to these jack socket pins: solder the red wire to PIN 1 (designated for power), solder the black wire to the longest pin (number 2, designated for ground), and solder the white wire to PIN 3 (designated for signals).

After soldering the wires to the jack socket, we can now connect them to the terminal block. Connect the white wire to the pole labeled "signal" in the terminal block. Connect the black wire to the ground pole in the terminal block. Connect the red wire to the pole labeled (+) in the terminal block.

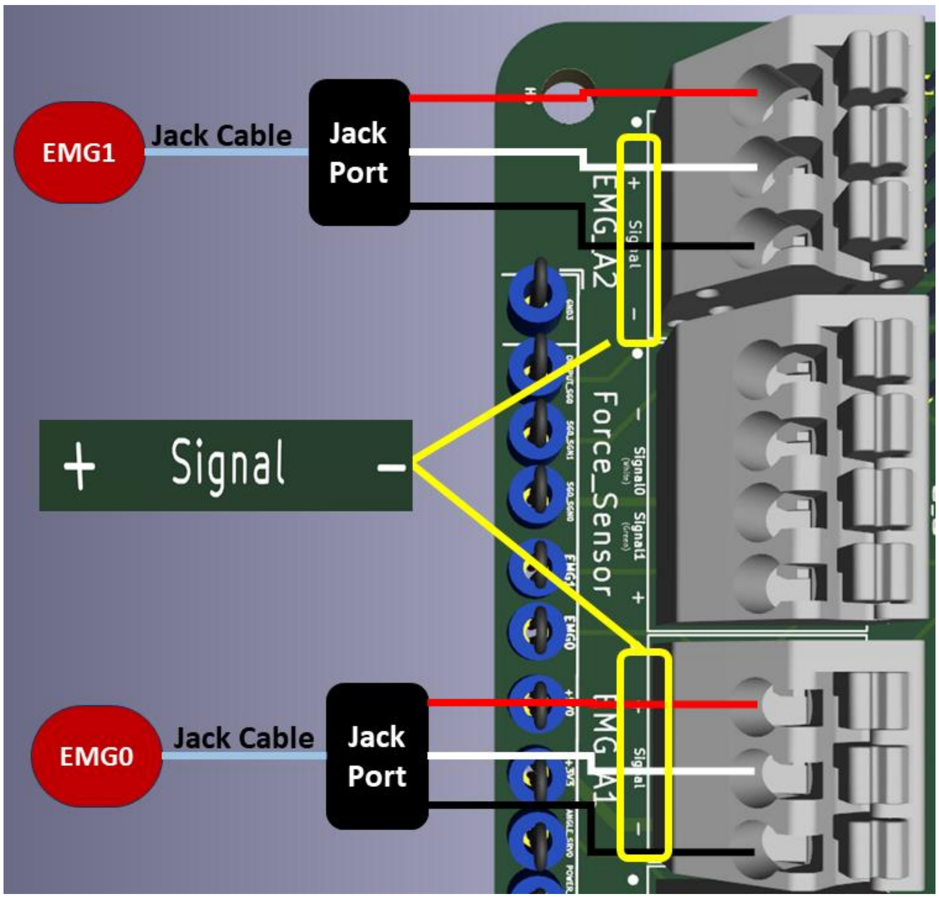

Fig. 15. EMG Sensor Wiring: Schematic illustrating the connection of EMG sensors to a jack socket, with clear guidance on linking the jack socket wires to the corresponding terminals on the PCB.

3.3.3 Checking the EMG Sensor

With the EMG Jack Sockets connected, we can now proceed to implement the program that reads the EMG sensor and displays the measured value on the screen using the serial monitor. Reading the EMG data is akin to reading the Force Sensor. The amplified signal is connected to an analog input pin, and the corresponding code for this operation is depicted in the following figure.

```
int emgAnalogInPin2 = A2;
int emgSignal2 = 0;
int emgAnalogInPin1= A1;
int emgSignal1 = 0;

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

void loop() {
  emgSignal2 = analogRead(emgAnalogInPin2);
  Serial.print("emgSignal2:");
  Serial.println(emgSignal2);
  delay(20);
  emgSignal1 = analogRead(emgAnalogInPin1);
  Serial.print(",");
  Serial.print("emgSignal1:");
  Serial.println(emgSignal1);
  delay(100);
}
```

Fig. 16. Arduino program to read EMG sensor

3.4 Testing the RGB Push Button (LP11)

A significant enhancement in the Exoskeleton is the integration of an RGB Push Button. This button serves as a connection point for the battery, Arduino, and sensors. When switched ON, it enables the Arduino and sensors to draw power directly from the battery. Conversely, when turned OFF, the Arduino remains inactive or draws power from the connected PC if one is linked.

To test the Push Button:

- Connect the battery to its terminal block. Ensure the (+) wire is correctly connected to the positive pole in the terminal block.
- Connect the Arduino to your PC and upload the code depicted in (Fig.14). The code is designed to change the color of the integrated RGB in the Push Button.

```
const int PIN_RED   = 3;
const int PIN_GREEN = 6;
const int PIN_BLUE  = 9;

void setup() {
  pinMode(PIN_RED,   OUTPUT);
  pinMode(PIN_GREEN, OUTPUT);
  pinMode(PIN_BLUE,  OUTPUT);
  analogWrite(PIN_RED,   0);
  analogWrite(PIN_GREEN, 255);
  analogWrite(PIN_BLUE,  255);
}

void loop() {
  // color code #00C9CC (R = 0,   G = 201, B = 204)
  analogWrite(PIN_RED,   0);
  analogWrite(PIN_GREEN, 201);
  analogWrite(PIN_BLUE,  204);

  delay(1000); // keep the color 1 second

  // color code #F7788A (R = 247, G = 120, B = 138)
  analogWrite(PIN_RED,   247);
  analogWrite(PIN_GREEN, 120);
  analogWrite(PIN_BLUE,  138);

  delay(1000); // keep the color 1 second

  // color code #34A853 (R = 52,  G = 168, B = 83)
  analogWrite(PIN_RED,   52);
  analogWrite(PIN_GREEN, 168);
  analogWrite(PIN_BLUE,  83);

  delay(1000); // keep the color 1 second
}
```

Fig. 17. Arduino code to test the push button's RGB functionality

4. Assembling and Connecting

Here we will explain how to assemble the Upper-arm Cover and connect the Sensors, Servo, and others to the PCB. Screw Thread Insert

4.1 Assembling the Upper-arm Cover

4.1.1 Assembling the Sidewall

With all the parts ready, you can now assemble the upper-arm cover of the Classroom Exo by inserting the screw thread inserts into the holes of the sidewall segment. You'll need 10 M3 screw thread inserts and 4 M4 screw thread inserts for this step. Insert a screw into the thread insert, then screw the thread insert into the hole using the tip of the inserted screw. Repeat this process for all the required inserts.

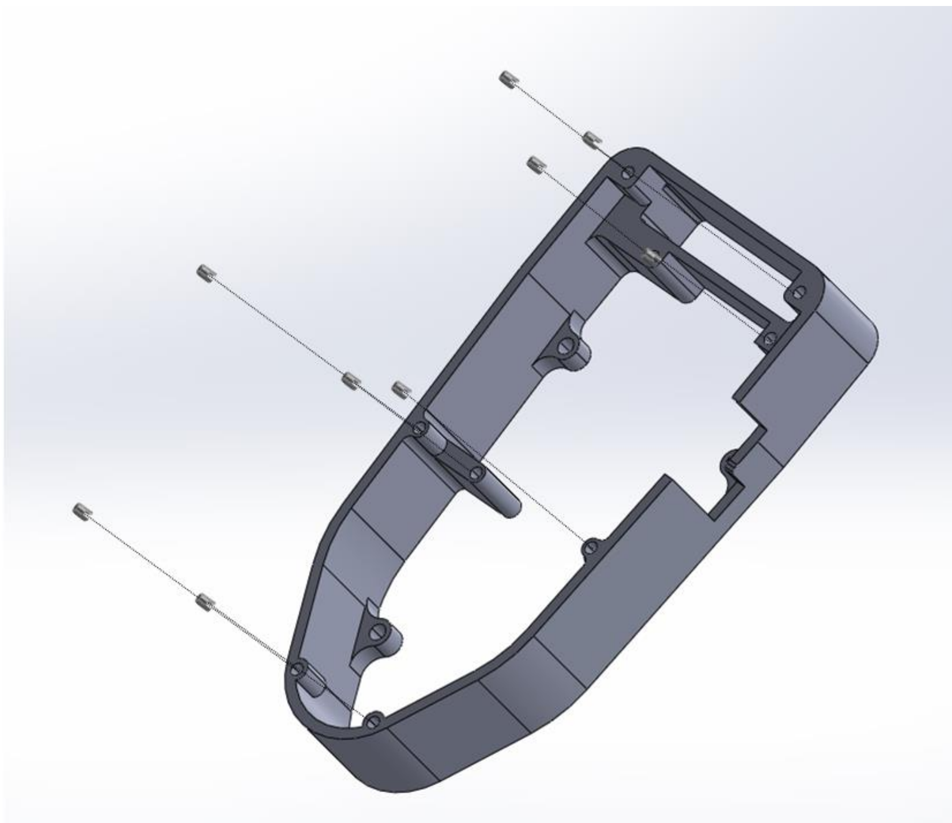

Fig. 18. The Sidewall segment has 10 "M3" holes and 4 "M4" holes.

4.1.2 Assembling the Frontside

Once the sidewall part is assembled, consider inserting the EMG sensor jack sockets into the frontside segment. The frontside part is designed to be connected to the sidewall segment using six M3 screws. However, this step comes after connecting the sensors and servo to the PCB, which will be discussed in the next section.

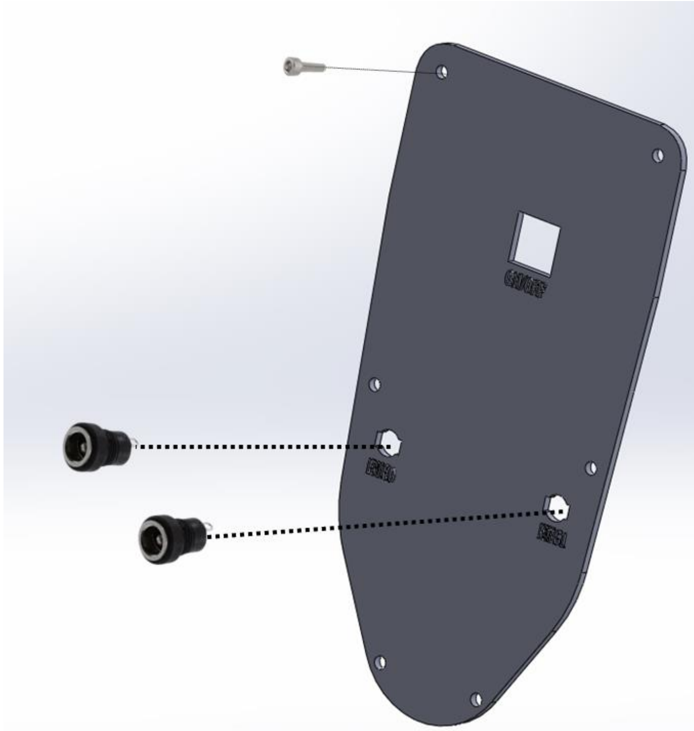

Fig. 19. Placing the EMG Plug socket on the Frontside segment of the Upper-arm Cover

4.1.3 Assembling the Sidewall Segment to the Metal Backside of the Upper-arm Cover

As previously mentioned, this manual serves as a partial or complete replacement for Chapter 5 in the EduExo Pro manual, focusing primarily on the electronics part of the assembly. At this point, you should have assembled most of the mechanical parts, including the metal segment with the attached servo, serving as the back side of the upper-arm cover, as depicted in Fig.17. Screw the Backside segment to the Sidewall segment using the M5 screw.

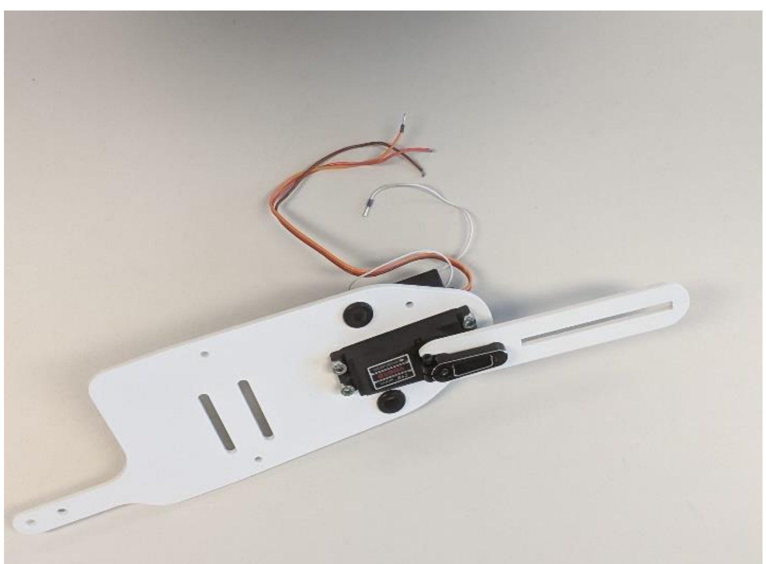

Fig. 20. Backside segment of the Upper-arm Cover

4.2 Connection

Through this section, we will discuss the connection of the Battery, Servo, and sensors. We recommend to follow our order when you connect.

4.2.1 Battery Connection

Connect the battery to its terminal block, ensuring that the red wire (+) is connected to the terminal block labeled (+) and the black wire is connected to the (-) pole, as illustrated in Fig.1.

4.2.2 Servo Connection

- Connect the motor's red wire to the terminal block's pole labeled (+),
- Connect the motor's brown wire to the terminal block's ground pole (-),
- Connect the white wire of the potentiometer angle sensor to the pole labeled "Position Sensor",
- Connect the control signal wire (orange) to the pole labeled "Signal", as illustrated in Fig.21.

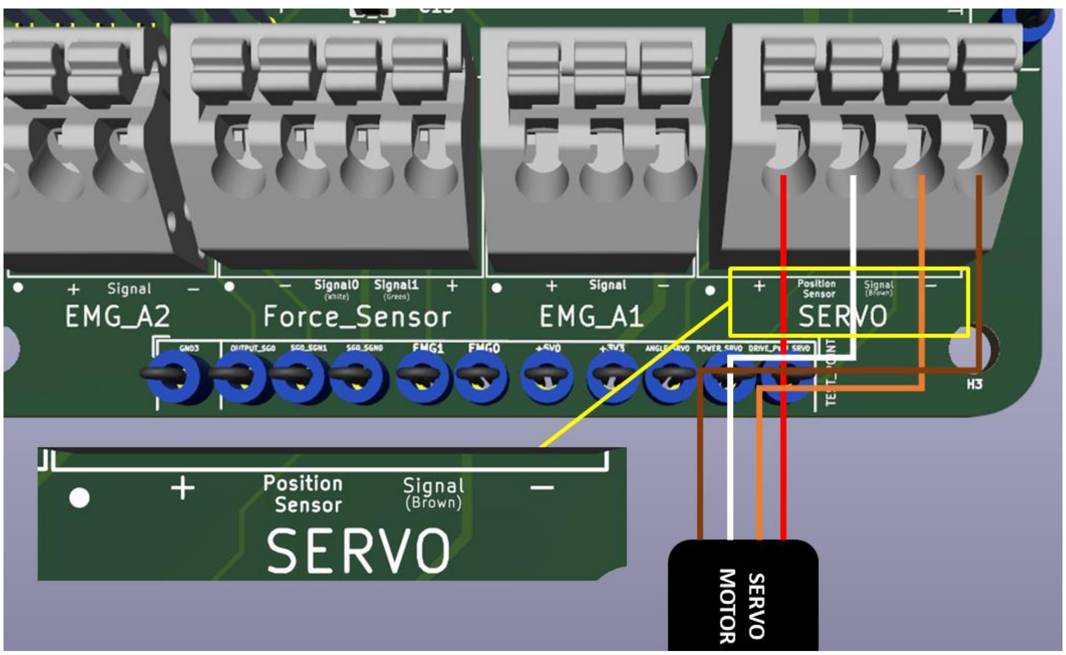

Fig. 21. Visual guide on connecting the servo motor to the terminal block

4.2.3 Force Sensor Connection

As mentioned earlier, solder the Force Sensor to a 4-wire cable before connecting it to the PCB terminal block. Given that the Force Sensor is situated in the Wrist segment of the Exoskeleton, the optimal way to route the wire inside the Upper-arm Cover is by threading it through one of the holes in the metal backside of the Upper-arm Cover.

- Note that wires' colors in the cable are not standardized. First, Identify the red wire, the black wire, the white wire, and the green wire according to the force Sensor wires' colors.
- Connect the red wire to the terminal labeled (+).
- Connect the black wire to the terminal labeled (-).
- Connect the white wire to the terminal labeled (white wire).

- Connect the green wire to the terminal labeled (green wire).

Refer to this (Fig.22) for a visual guide to ensure accurate connections.

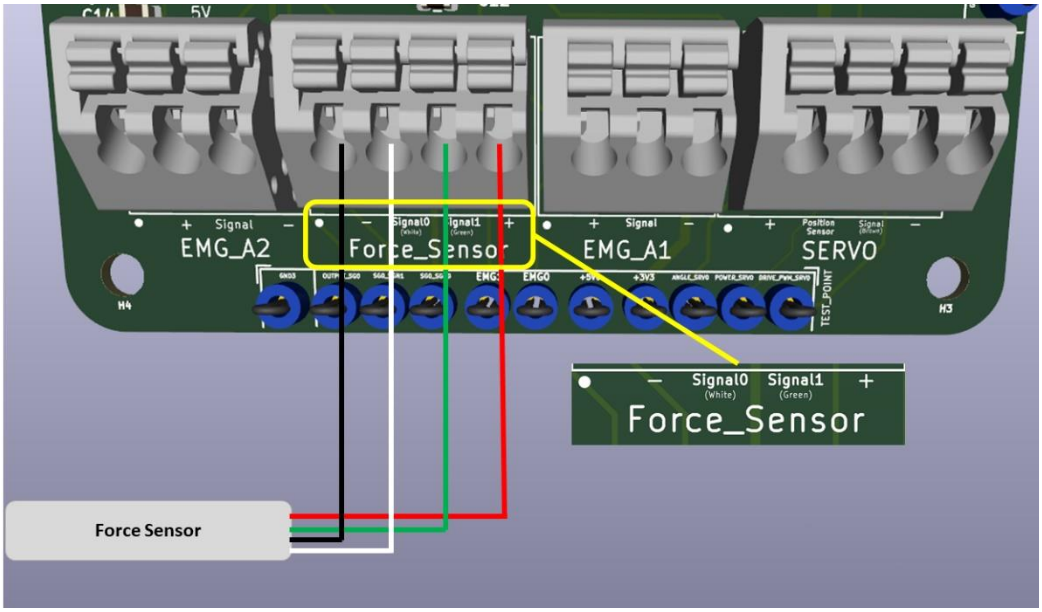

Fig. 22 Force Sensor Connection: Follow the color-coded traces to match force sensor wires from the 4-wire cable to their designated terminals for a secure and accurate connection.

4.2.4 EMG Sensor Connection

As the EMG sensors are connected through jack plugs, let's go through the process of connecting the jack sockets to the terminal block.

- Connect the white wire (PIN 3 in the socket) to the hole labeled "signal" in the terminal block.
- Connect the black wire (the longest PIN in the socket) to the ground hole in the terminal block.
- Connect the red wire (PIN 1 in the socket) to the (+) pin in the terminal block.

Refer to this (Fig.23) for a visual guide to ensure accurate connections.

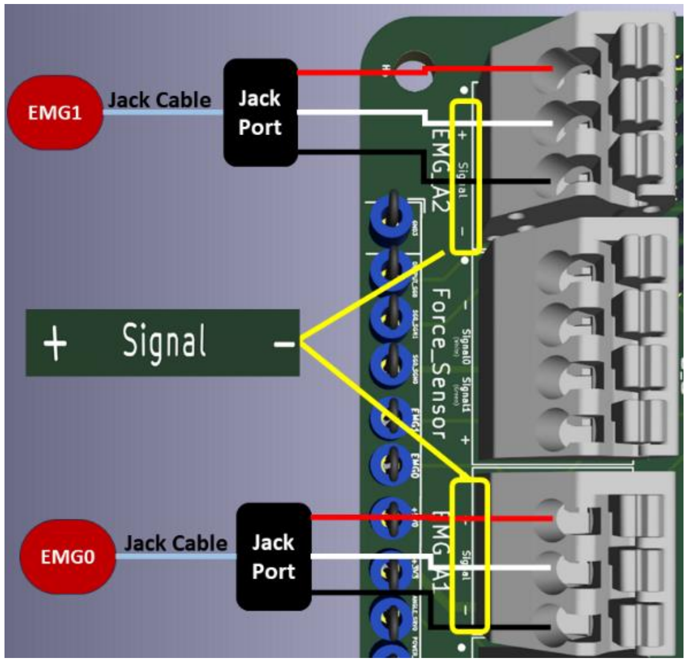

Fig. 23. EMG Sensor Wiring: Schematic illustrating the connection of EMG sensors to a jack socket, with clear guidance on linking the jack socket wires to the corresponding terminals on the PCB.

4.2.5 Placing the PCB and Closing the Upper-arm Cover

After finishing all the connections to the PCB, it is time to place the PCB on the top of the Sidewall segment UPSIDE DOWN so the Push Button should face up. During that, you should place the battery inside the upper section of the Sidewall segment. You can refer to (Fig.24) for a visual guide.

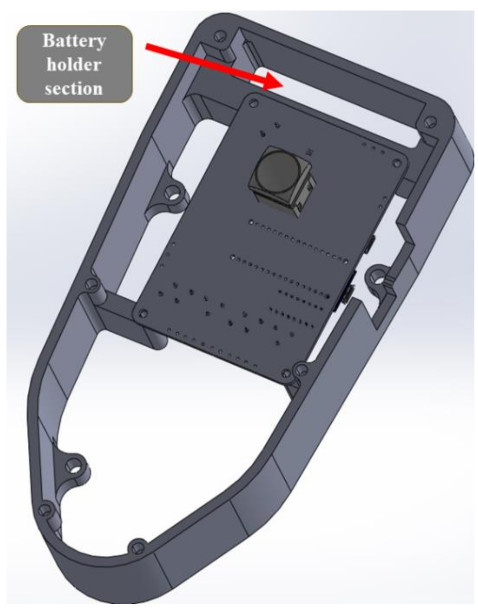

Fig. 24. Assembling the Upper-Arm Cover: Placement of the PCB and Battery within the sidewall segment, following the connection of sensors and servo to the PCB.

Gently position the Frontside part onto the Sidewall segment, securely fastening the two segments using 6 M3 screws, as depicted in Fig. 25. Ensure a snug fit for a cohesive and robust assembly.

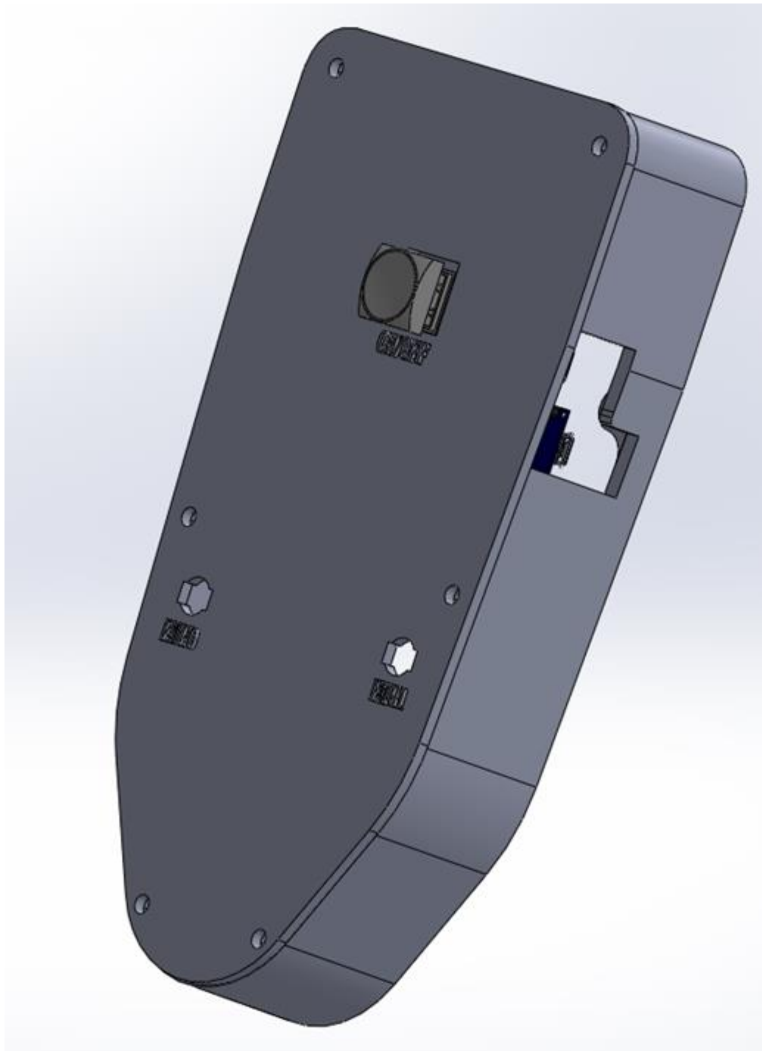

Fig. 25. Completed Upper-Arm Cover: The assembled upper-arm cover showcases the successful integration of the PCB, with the Frontside and Sidewall segments seamlessly joined. A finished look ready for further testing and application.

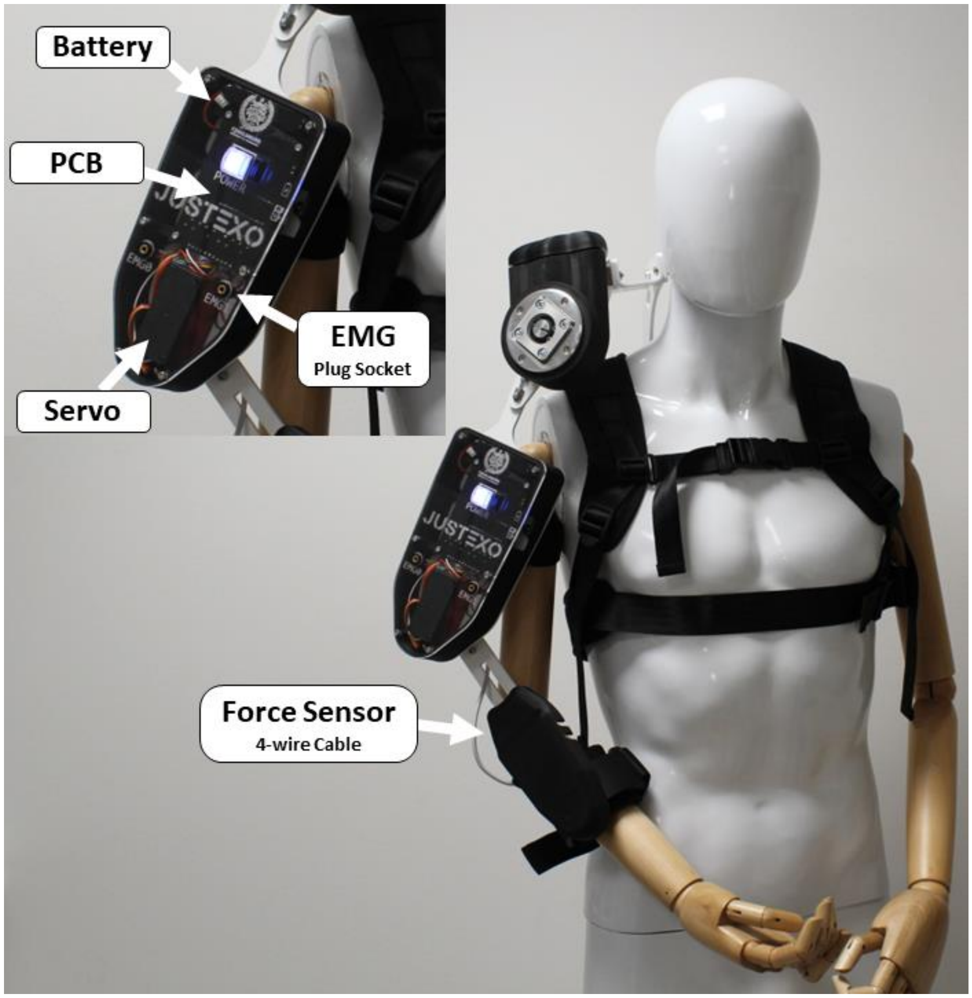

Fig. 26. Classroom Exo: A glimpse of the fully assembled Classroom Exo, embodying improved electronics and enhanced functionalities.