Project description

1. Feedback system overview

Providing an efficient feedback system (figure 1) is undoubtedly a big challenge, which is still lacking in prostheses currently available on the market. Consequently, it has become important to develop advanced decoding/encoding algorithms offering precise feedback. The encoding/decoding algorithms should translate the captured information from the sensitized prosthetic hand into a feedback signal to the amputee through different feedback modalities, such as vibrotactile stimulation. To englobe, information coming from sensors over Arduino pass through data processing with the aim of predicting object nature (sharpness, stiffness or texture). The prediction operation exploits models already prepared to predict new data. In our project, data out of Arduino, which composed of 150 samples, proceed over I2C (figure 2) to meet the model reposed in Raspberry Pi card. At this juncture, the predicted information transmitted to the feedback device located in the amputee's biceps by way of Bluetooth module. A unique combination of components forms the feedback device, including Arduino Nano, a Bluetooth module, a couple of coin type vibration motors and heating gadget that impersonate heat danger. Afterwards, the measured pressure on the fingertips was translated into vibrotactile feedback using vibration motors placed on the subject's arm. However, thermistor captures temperature variation to warn prosthetic's user by the means of a heating device. Based on the received sensory feedback and the elicited sensations, the amputee can identify the stiffness, sharpness, texture and warm danger.

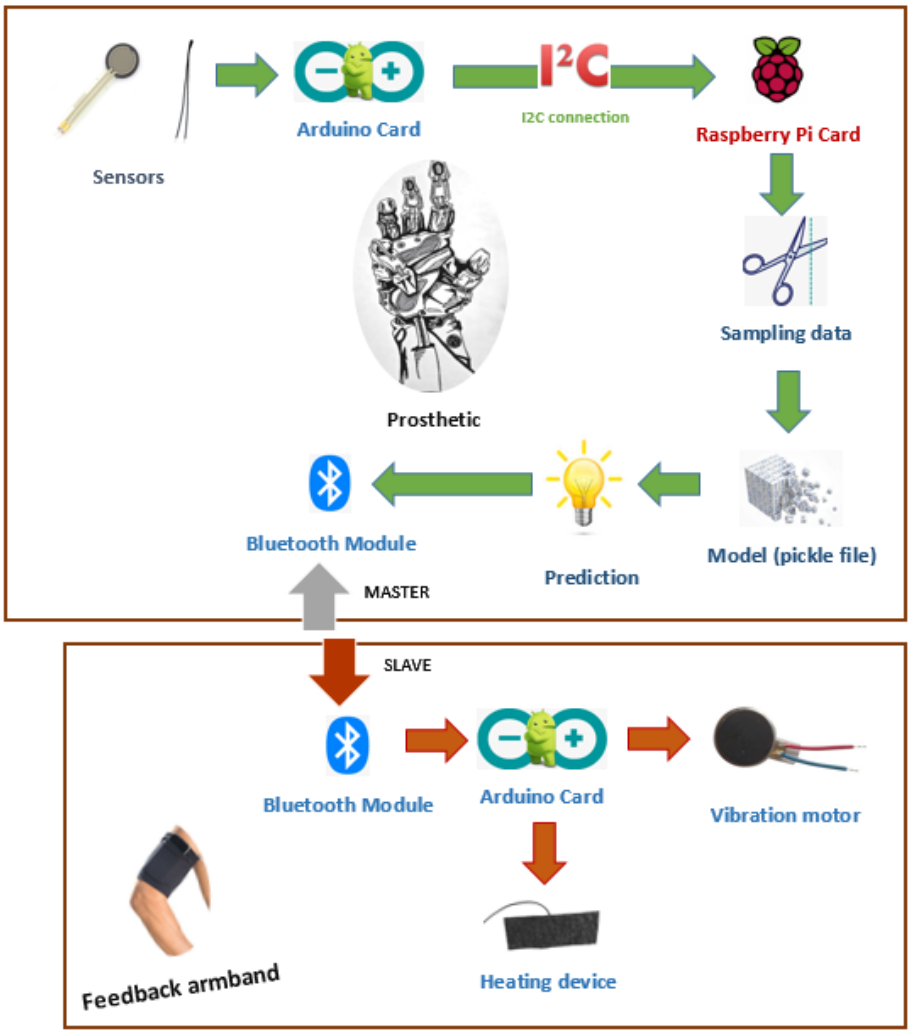


Figure 1: Feedback system overview

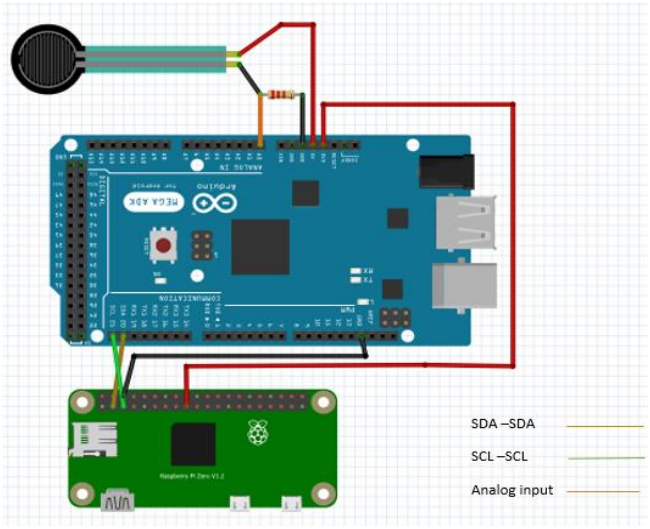


Figure 2: I2C communication

2. Glove design

We embedded an FSR sensor on all fingers to guarantee stiffness perception. In the other side, face-to-face FSR implanted in the thumb to achieve sharpness sensation. As well as for texture perception which is provided with the index's sensor. However, the index finger contains also a thermistor giving temperature feedback. The figure 3 below describe Arduino mega connection with sensors.

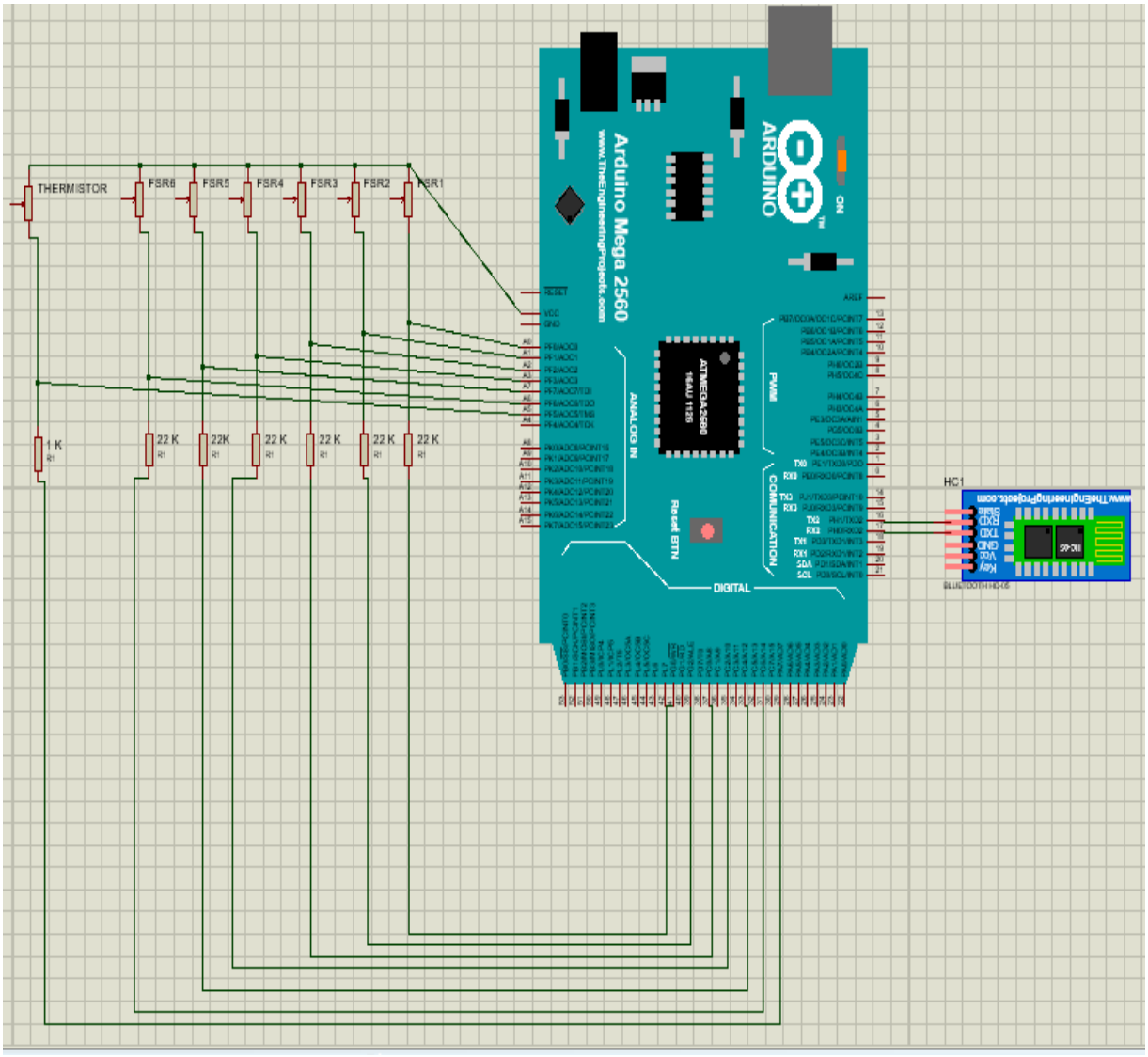


Figure 3: Sensors scheme

3. Decoding script algorithm

The Raspberry Pi catch data arriving from Arduino through I2C communication in order to make a prediction. However, Arduino sends info as array form after mapping value range between 0 and 255 since I2C bus can only transmit data byte per byte. With this being said, Pi earn information to convert it to a wider range (0 to 5000) in the interest of facilitating distinction between classes. Afterwards, we continue getting data and join it to an array until we reach 150 samples as describe the figure 4. Each array stores data value for every single sensor. Owning to the transmission speed, Pi could have empty data. As a result, we fix all missing details then merge all arrays in Panda Data. Every single perception has its own data-frame. Afterwards, we cipher this matrix to one row matrix with a fixed number of the column. This number refers to the features' number. Prediction operation (figure 5) is now feasible after loading the prepared model for each perception. Finally, the predicted info conveyed to the feedback armband over the HC-05 Bluetooth module. At this point and after experimenting system with an online review, the mechanism show latency when sending predicted value through Bluetooth. The Zero Pi's ram is immersed with processing the ML script. In sequence, we shift the HC-05 module to the Arduino Mega and send the predicted value over I2C. In the other end of the spectrum, the Arduino Mega algorithm (figure 6) consists of reading in an infinite loop from sensors and wait for order coming from the master (Raspberry Pi Zero). Ultimately, it dispatches heat danger to the armband if analog value passes 120.

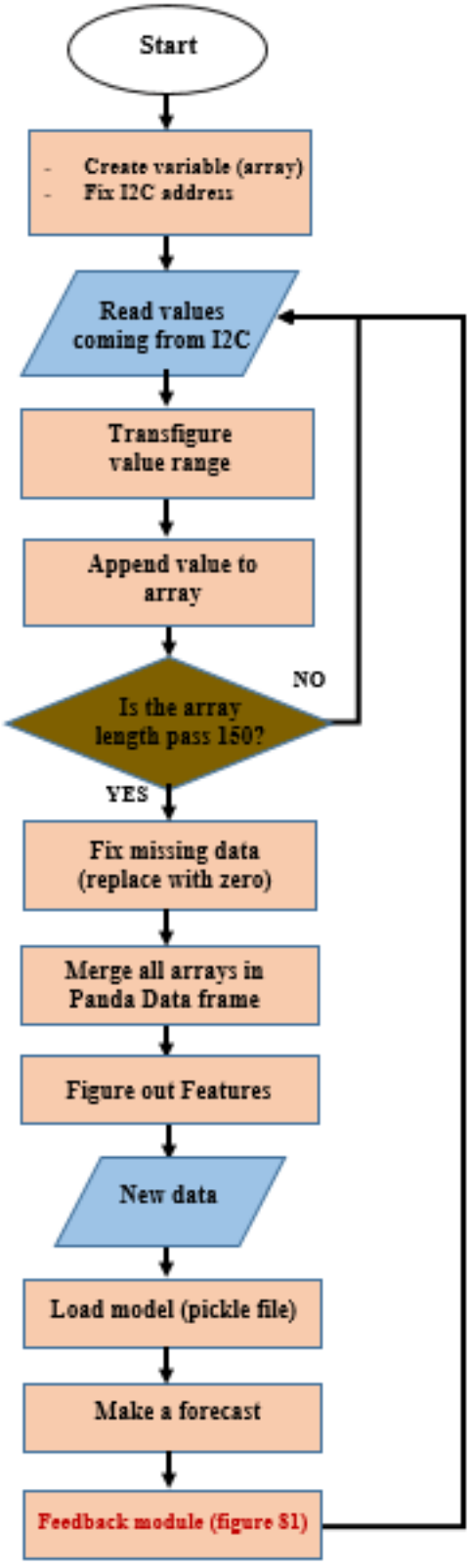


Figure 4: decoding algorithm

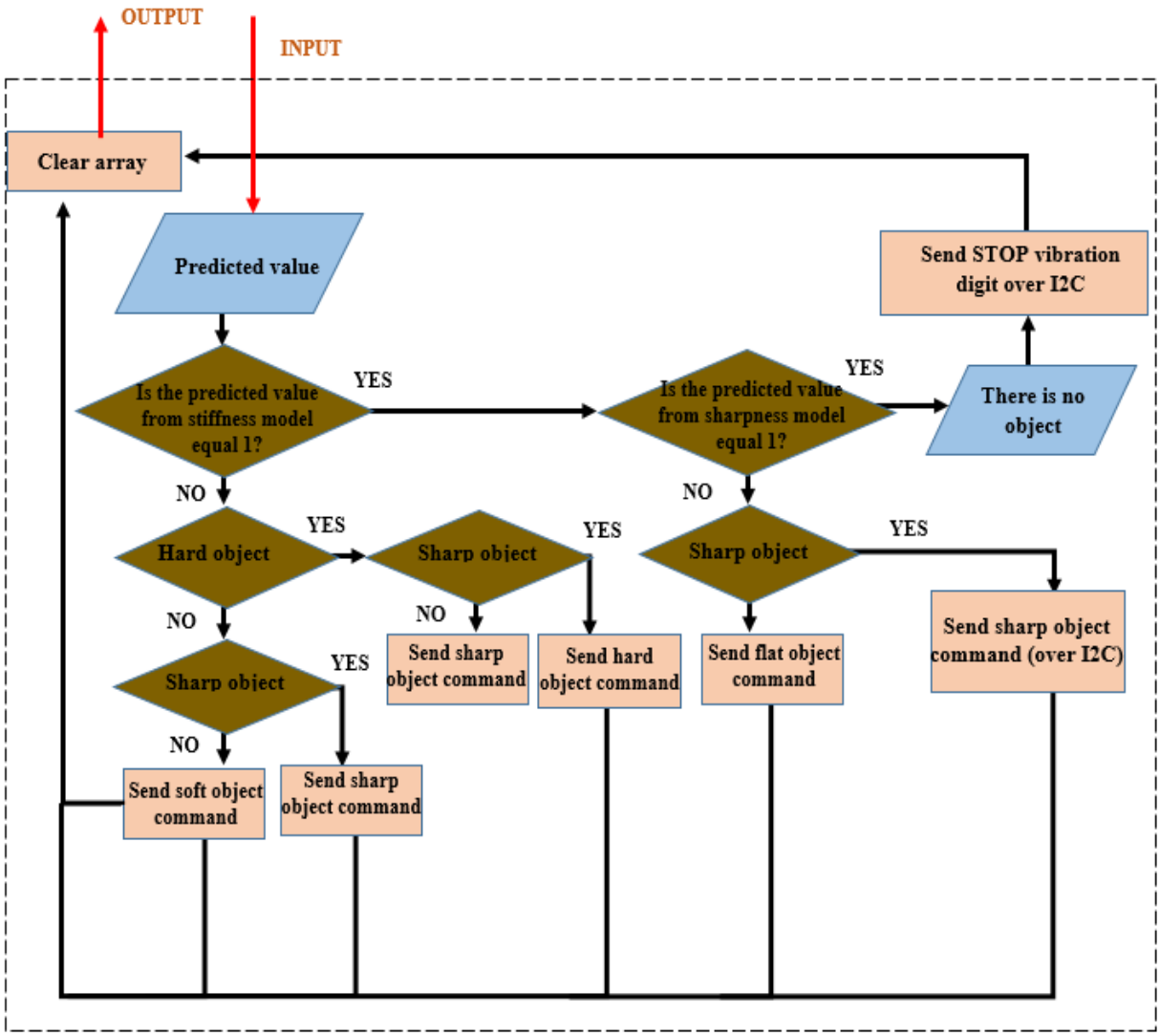


Figure 5: feedback module

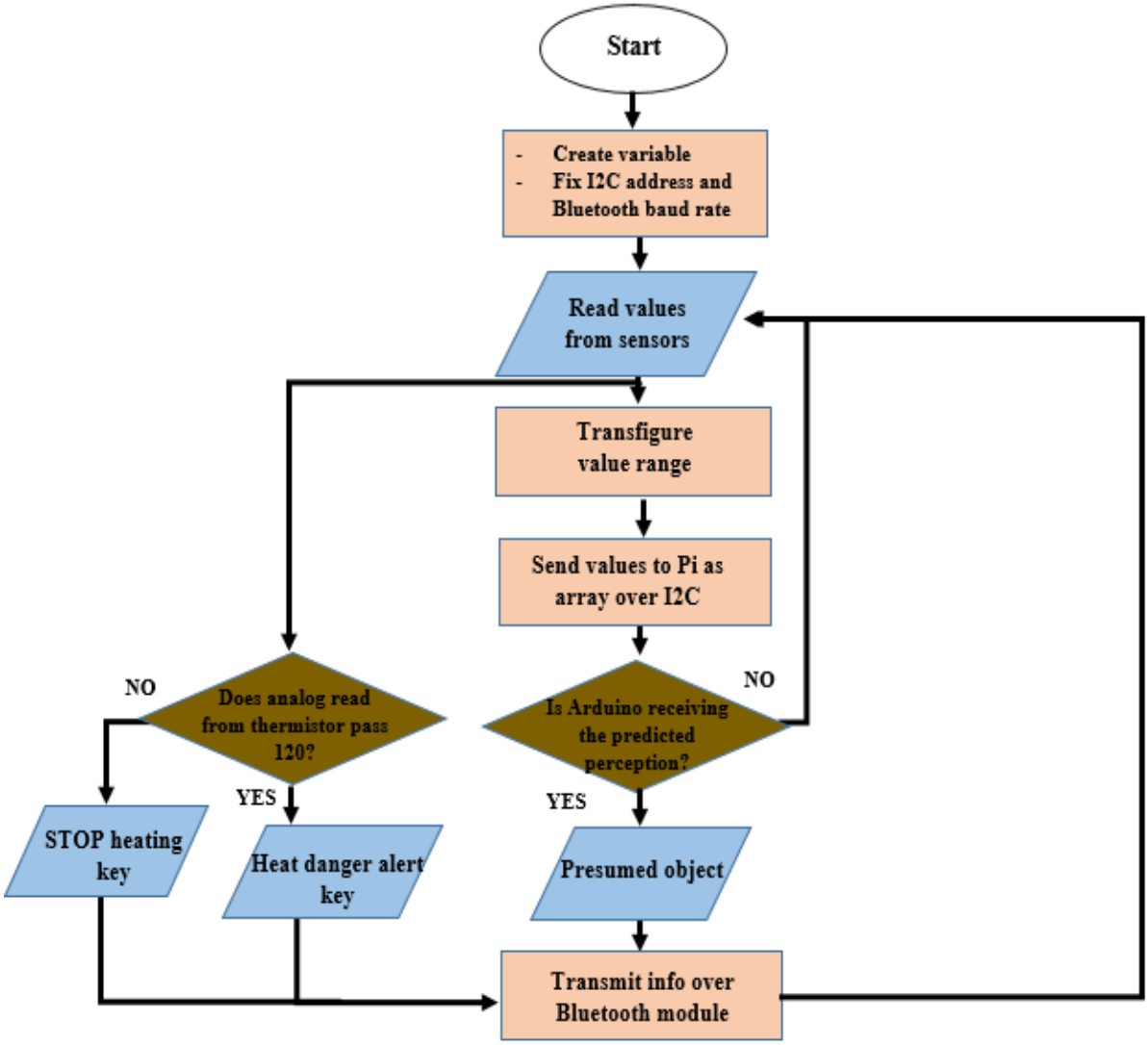


Figure 6: Arduino Mega algorithm

4. Armband design

Vibration motor connected with the pulse width modulation (PWM) pin of the Nano board, which control the vibration rate. Referring to Arduino Nano's pinout datasheet (figure 69), the PWM pins are D3, D5, D6, D9, D10 and D11. A Li-ion battery with 7.4 Volts power on our Nano through VIN and ground pins. Afterwards, we draw current through carbon fiber to warm the amputee's arm. This current is ruled over the digital pin D13. Lastly, in order to catch info coming from master (Arduino Mega), the slave module linked as shown in figure 7. Unlike master communication which RX of the Arduino to TX of the module (HC-05) and TX to RX, the slave connection is totally the inverse.

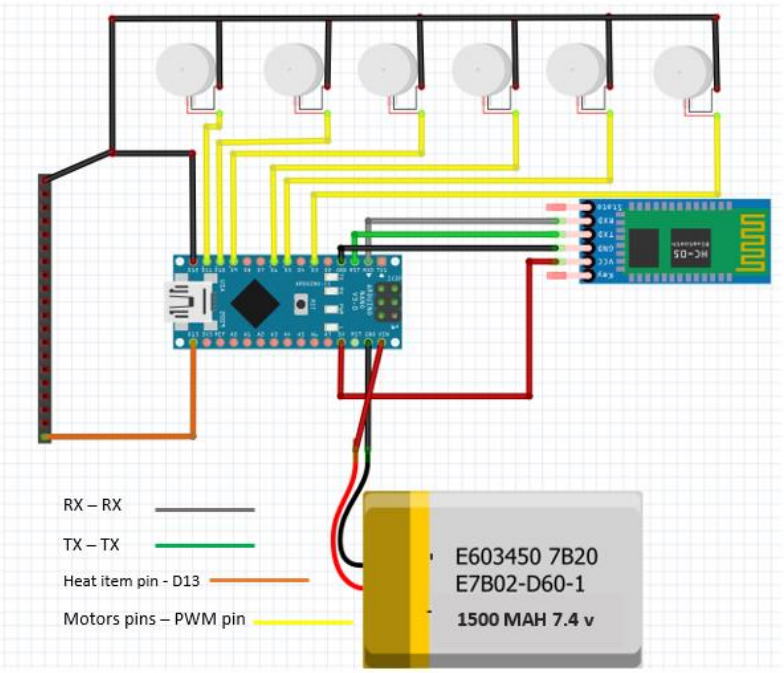


Figure 7: armband design

5. Encoding script algorithm

Decoding script ensures the transformation of the predicted perception of the appropriate object nature through coin vibration motors association. Vibration position differentiates a sharp (flat) object from hard (soft) while vibration speed distinguishes between the flat and soft object. However, the Bluetooth slave starts listening to the master until it sends an order. According to this request, the Nano board actuate a specific vibration motor combination with a particular vibration rate for each perception. Eventually, figure 8 describe each combination and how the warming element behaves. The three coins, which covers biceps, operate with a pulse rate of 255 (PWM) for a hard object while it works with 100 for soft object notification. Nevertheless, the three-vibration motor on triceps ensures sharp object alert with a rate of 255 and 100 for flat items. Further, the heating element takes even high or low state, which means even 5 or 0 Volts.

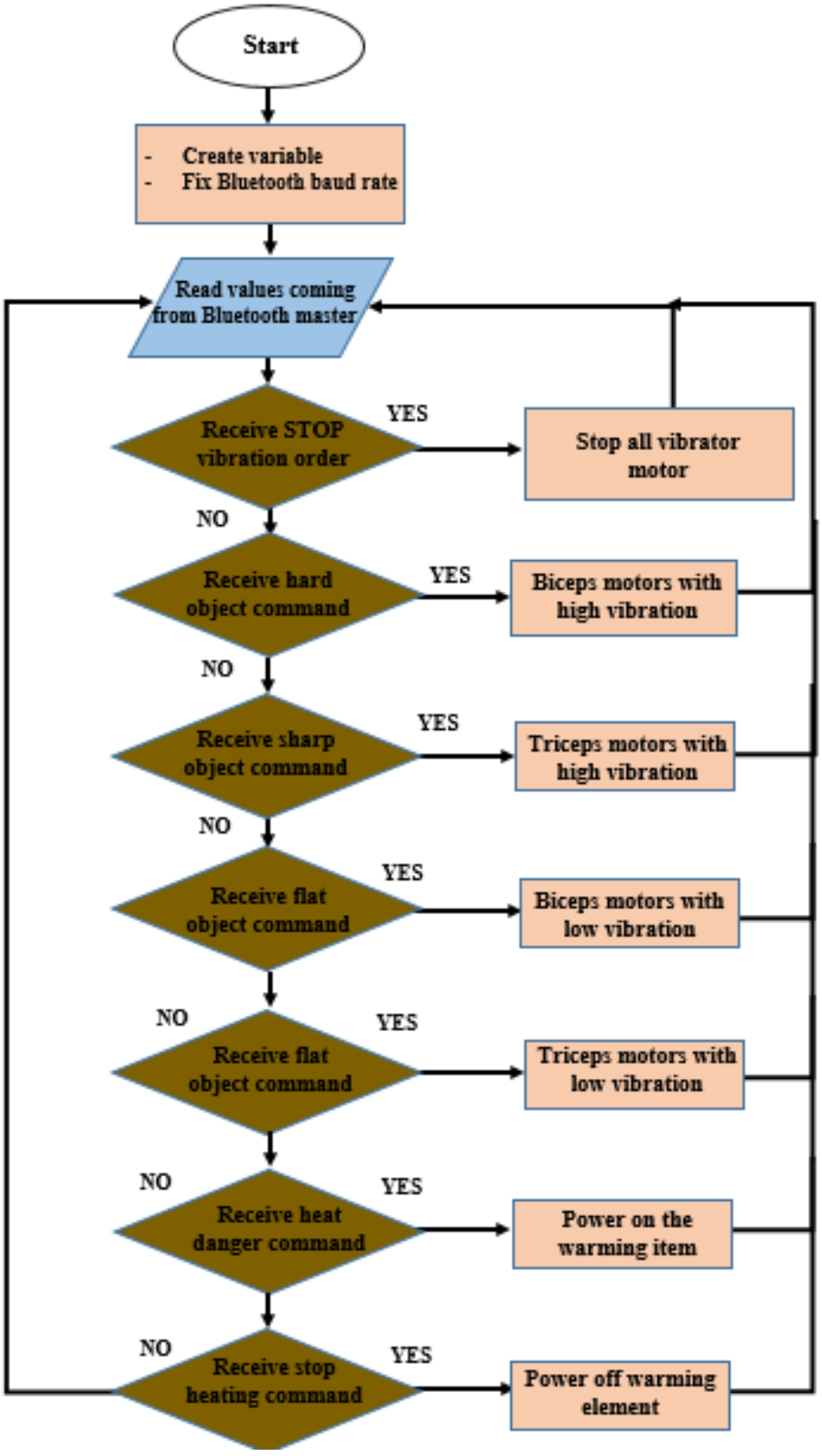
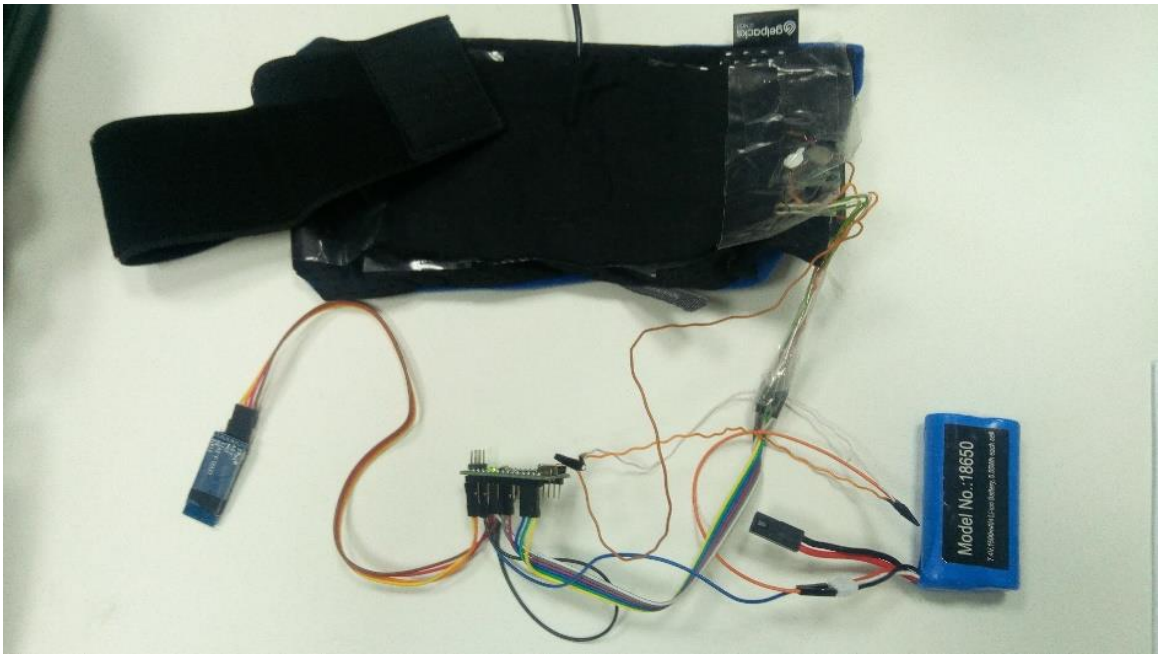


Figure 7: stimulation algorithm


CAUTION


Model No.:18650