

Internship Program

Jingchao Zeng

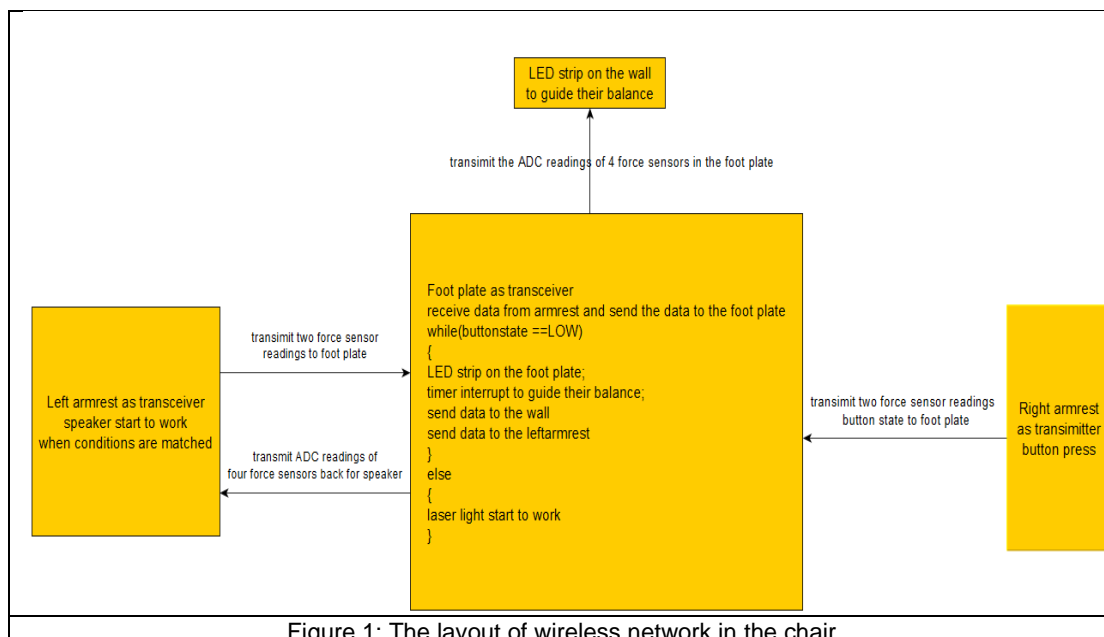
Requirement:

My aim for this internship is to develop the instrumented chair to guide the elderly people in the sit-to-stand exercise, including the data flow from the armrest to the foot plate and then finally to the wall. Hopefully, my design can guide their balance during the exercise and have fun when playing with them.

Wireless network:

In Figure 1, The SPI communication enables this wireless network to transmit ADC readings of force sensors within the chair.

The LED strips on the armrest seems not very tidy and reasonable so that RF transceiver module transmit all the data from the armrest to the foot plate. There are two force sensors in each armrest, which record the force in the upper arm and back arm. Importantly, I mounted the speaker in the left armrest and button in the right



armrest to enable more functions in the chair. The button is used to control the behavior in the foot plate. When button state is HIGH, only the laser light works to guide the elderly people how to keep balance when they stand on the foot plate. Otherwise, the timer interrupts on LED strips and speaker would work to guide their balance. The speaker would be used to generate tone when people are standing on the foot plate. In order to achieve that, the force sensor readings of the foot plate should be sent to the left armrest, which acts as the transceiver in SPI communication.

The foot plate acts as a master node, which receives the data from both armrests and transmit its force sensor readings to both the left armrest and wall. Most feedback is built in the foot plate, including the laser light and LED strips.

The wall acts as the child node, which receives force sensor readings from the foot plate and feedback the readings in terms of color in the LED strip. It would only work when the button state is LOW.

Hardware

I. Layout

In Figure 2 on the page 3, there are 3 LED strips on the top surface of the foot plate. In the front middle, there are 17 LEDs for the foot plate, where the middle is always white to divide the LED strip into left and right. In each side of foot plate, there are 5 LEDs respectively for left and right armrest. The laser light with servo motors is mounted with screws behind the LEDs strip. The hole nearby is for the wire of servo motors and laser light down to the Arduino Mega. All the LEDs strips are stabilized by the blue tape and screws. With this design, people have plenty of spaces to stand on the foot plate and see the lighting on LEDs clearly during the exercise.

In Figure 3 on the page 3, there are 4 casing of force sensors to support the base of foot plate. There are high enough to mount the Arduino and let the jumper wires to connect in the strip board. In the middle, there is an Arduino mega 2560, which

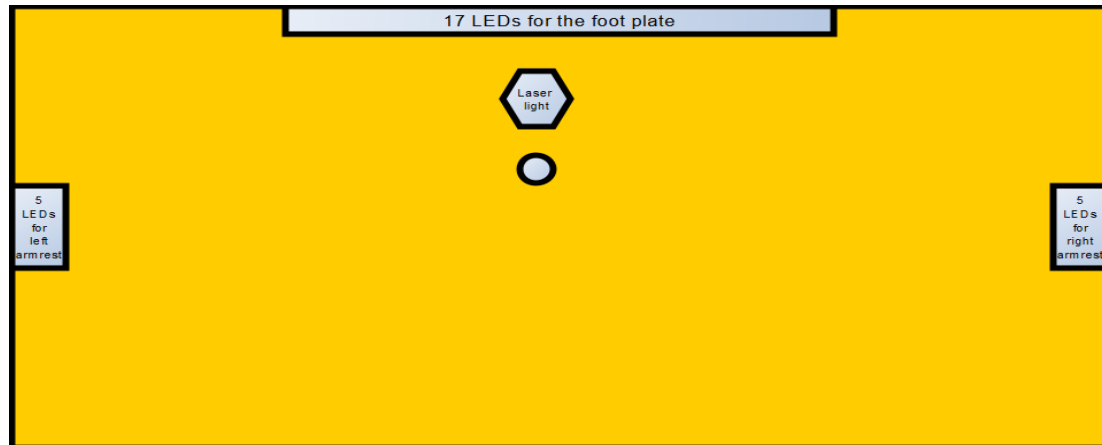


Figure 2: The top view of the foot plate

provides enough pins and up to six timers. On each side of the Arduino, there are four amplifier circuits for each force sensors. Besides, there is a strip board to share 5V pins and GND for amplifier circuits, servo motors, laser light and LED strips. Also, this strip board connect the signal pins of LED strips to 470 Ohm resistor before connecting to the microcontroller, which can display enough brightness and save power consumption. There is a nRF24L01 transceiver module below the controller, which builds up the wireless communication. Moreover, three connectors of LED strips are stabilized by the tape and screw, where the jumper wires are extended by joining together.

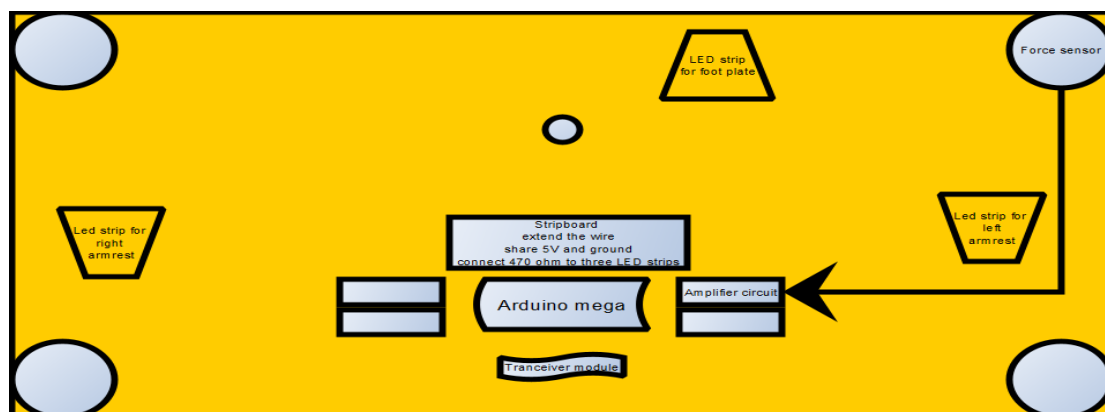
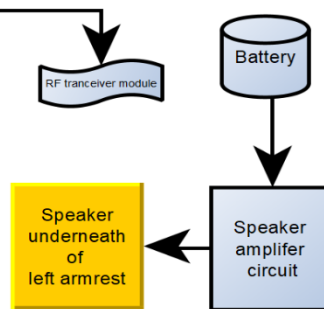
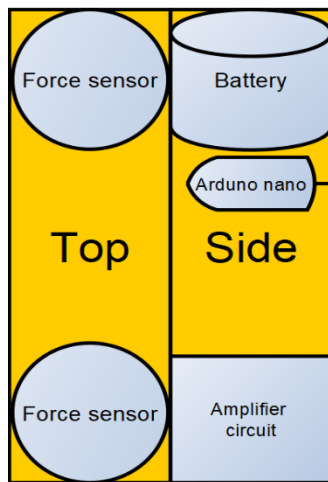
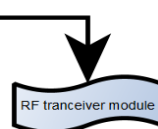
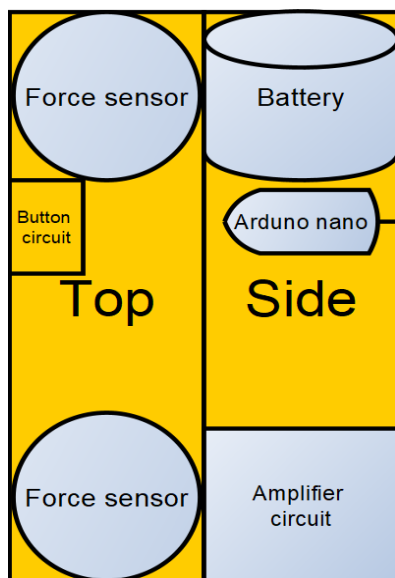


Figure 3: The bottom view of the foot plate



This is the top view and side view of the left armrest. There are two force sensors in the upper and back arm on the top surface. In each side of arm rest, there are two according amplifier circuits. To be more specific, in the inner side, there

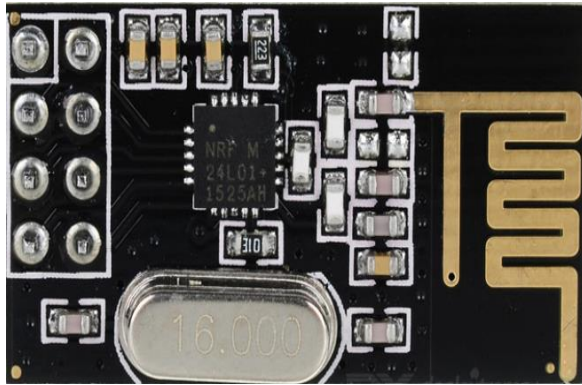
is an Arduino nano, which connects to the RF transceiver module. However, in the outer side, there is a 9V battery to power up the microcontroller. The speaker is mounted underneath of the arm rest, which is amplified by the LM386 audio amplifier in combination with specific capacitors and resistors supplied by 9V battery. Also, the speaker amplifier strip board has enough spaces to share 5V and GND with the amplifier circuit of the force sensors.



This is the top view and side view of right armrest. The design is mostly same as the left armrest for the top view. The only difference is that there is a button circuit near to the upper force sensor, which shares 5V and GND for the force sensors' amplifier circuits. The side view is exactly same as the left armrest, which has battery for Arduino nano

and amplifier circuits in each side.

II. Components



The wireless network is achieved by the nRF24L01 radio transceiver for the world wide 2.4 GHz ISM band, which can reach up to 100 meters. Also, it can communicate up to 6 other units at the same time with baud rates from 250 kbps up to 2 Mbps.

In Figure 4, PWM audio signal is connected to the 10k potentiometer, which can vary the voltage of audio going through the LM386 audio amplifier. Several capacitors and resistors connected to amplifier has different functions. To be more specific, a 100 nF capacitor between the positive input signal (2) and ground (4), which filters radio interference picked up by the audio input wires. A 470 pF capacitor between pins 4 and 6, for additional decoupling of the power supply to the chip. A 1.2K Ohm resistor

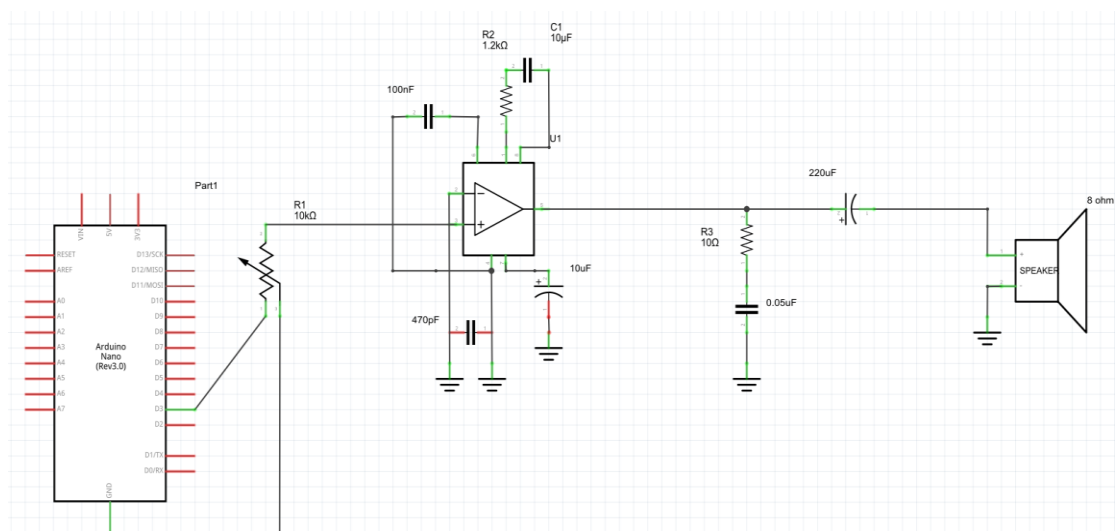
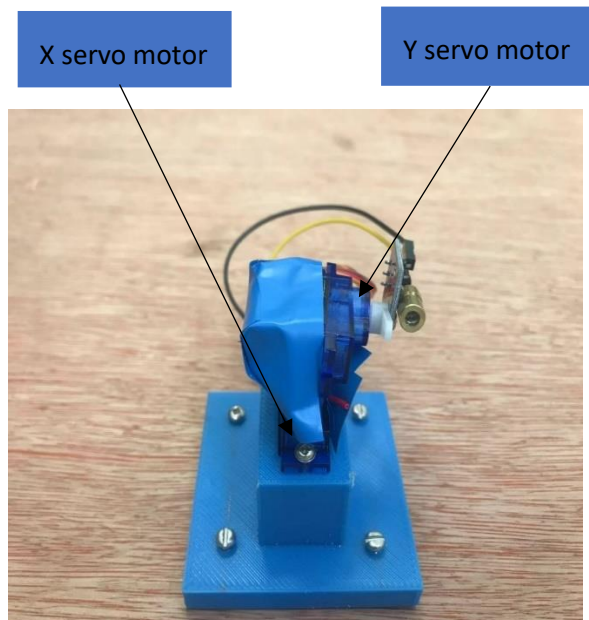


Figure 4: Schematic of speaker amplifier circuit

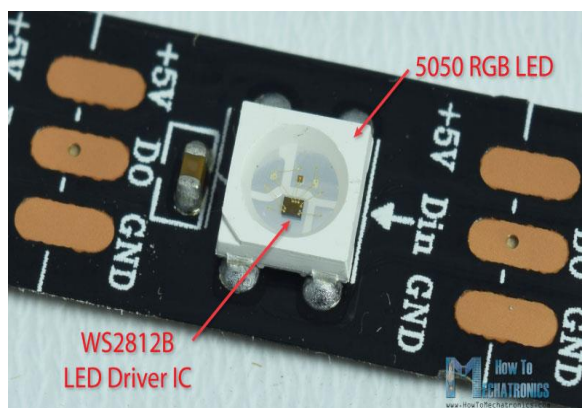
and a 10 μ F capacitor in series between pin 1 and pin 8 is to set the gain to 50 for acceptable volume of output. A 10 μ F capacitor in series between pin 7 and ground

is to decouple the audio input signal. The load impedance of speaker is 8 Ohm, which is in acceptable range between 4 Ohm and 32 Ohm. With this design, we can clearly hear tone from speaker clearly with acceptable noise interference.



The laser light can move in x and y direction with use of two servo motors in combination, which is stabilized by screw of the Y servo motor. The base is mounted on the top of foot plate in case it moves. This can be used to train people to keep balance on the foot plate just following the laser light. Also, it can turn left, right, up and down with

the use of motors, where people can capture and move with the defined patterns created by laser light.



WS2812B is an individually addressable digital LED strip. This type contains the LEDs, with three one-byte data members for each of the three Red, Green and Blue color channel. The data output pad of the previous LED is connected to the Data Input pad of the

next LED. We can cut the strip to any size we want, as well as distance the LEDs using some wires. Therefore, I cut the 17 LEDs for foot plate and 5 LEDs for two armrests, which enables 256 brightness and 16777216 full color display.

Software:

A simplified version of the system in the instrumented chair is shown in the flow chart below.

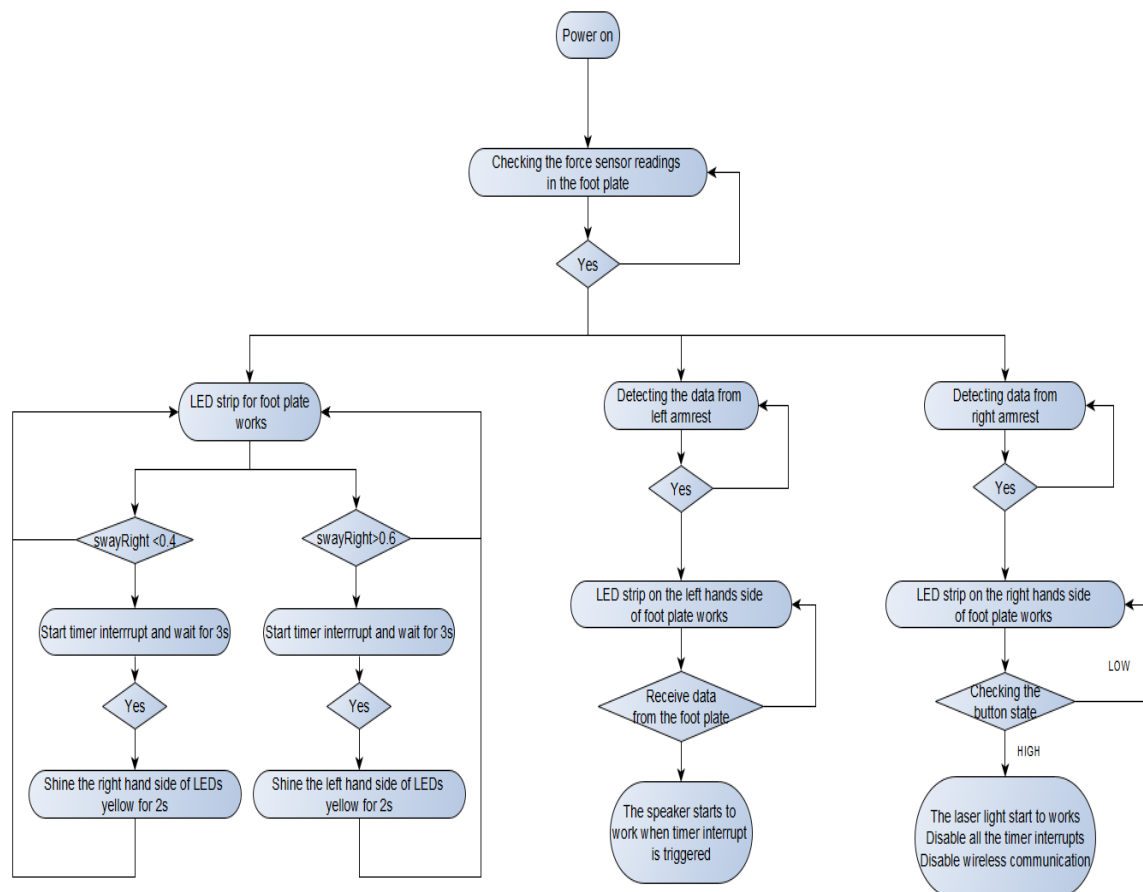


Figure 5: Software system flow chart

I. System explanation

The whole system runs in a big while loop. The first thing is to define all the useful libraries and initializes all the peripheral devices, including LED strips, RF24 transceiver modules, servo motors and a laser emitter module. After initialization, the program would start to check the force sensor readings on the

foot plate. Then, the foot plate starts to listen the readings from node 01 (left armrest) and node 02 (right armrest). In the meantime, it transmits its four force sensors' readings to the left armrest for audio feedback implementation. The LED strips would change its LEDs from red to green according to the force applied. Also, the threshold to turn up the green LEDs would be changed to fit different sort of people in the serial monitor.

The debounced button in the right armrest would be sent to the foot plate to change the mode of operation. When the button state is HIGH, only laser light and servo motors would work and disabled other functions. However, when the button state is LOW, the timer interrupt overflow is set to delay yellow LEDs 2 second when people sway over defined time. In the meantime, the speaker would work synchronously with the yellow LEDs to give the warning message.

II. Wireless network

```
//===== Udate the network =====//
network.update();
//=====Receive ADC readings from armrest=====//
while (network.available())
{
    threshold_arm = threshold / 8;
    RF24NetworkHeader header;
    receive_force forceVal;
    network.read(header, &forceVal, sizeof(forceVal));
    //=====Listening from the left armrest data=====//
    if (header.from_node == 01)
    {
        leftUpper_arm = forceVal.Upper;
        leftBack_arm = forceVal.Back;
        if (count_arm == 127)
        {
            leds_left_arm(leftUpper_arm, leftBack_arm);
            red_left_arm(leftUpper_arm, leftBack_arm);
        }
    }
    //=====Listening from the right armrest data=====//
    if (header.from_node == 02)
    {
        rightUpper_arm = forceVal.Upper;
        rightBack_arm = forceVal.Back;
        buttonstate = forceVal.button;
        if (count_arm == 127)
        {
            leds_right_arm(rightUpper_arm, rightBack_arm);
            red_right_arm(rightUpper_arm, rightBack_arm);
        }
    }
}
```

Figure 6: Data received from node 01 (left armrest) and node 02 (right armrest)

As shown in Figure 6, the network would be updated and checked each time in the beginning to receive and transmit data. When the network is available, it would start to turn the slave select LOW in SPI communication to hear the message. In order to hear the message, the recipient address, header would be needed to define. Then using the read () function we read the data and store it into the forceVal variable.

Header.from_node () function is used to hear message from different logical address in the different communication channels. Hence, in this case, you need to define two different received structures of payloads for left and right armrest.

In Figure 7, with the use of the millis () function, the force sensor readings can be transmitted every 100 milliseconds to the left armrest. In each defined interval, the according force sensor readings and threshold would be updated in the structure of payloads to the defined node.

The boolean variable ok2 can be used to debug in the serial monitor. If it is 1, it means the data have been transmitted to correct node.

```
//=====Sending the data to the left armrest=====//
unsigned long now_arm = millis();
unsigned long last_arm;
if (now_arm - last_arm >= 100)
{
    last_arm = now_arm;
    send_arm armVal = { upperLeft , backLeft , upperRight , backRight, threshold};
    RF24NetworkHeader header3(node01);
    bool ok2 = network.write(header3, &armVal, sizeof(armVal));
}
```

Figure 7: Data transmitted from master node (foot plate) to the node01 (left armrest)

III. Timer 2 interrupt overflow

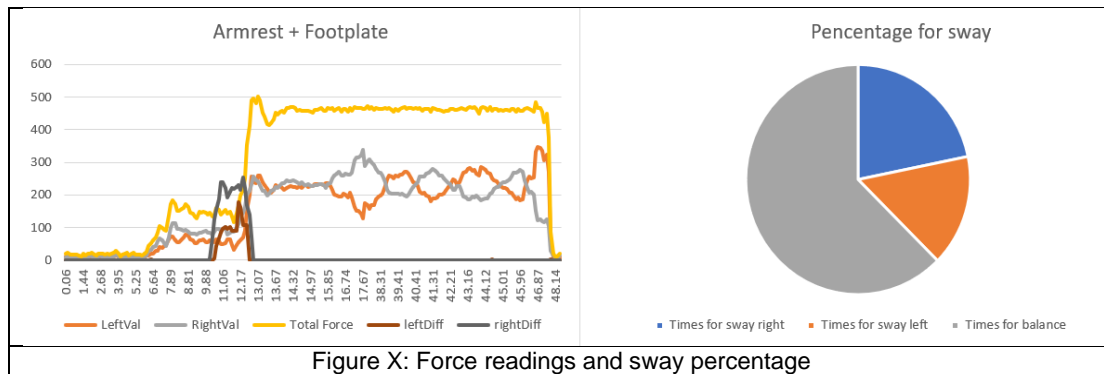
```
//===Define the Timer2 interrupt overflow ====  
TIMSK2 = (TIMSK2 & B11111110) | 0x01;  
TCCR2B = (TCCR2B & B11111000) | 0x07;
```

Figure 8: Set up Timer 2 overflow interrupt

As can be seen in Figure 8, to enable timer overflow, we must set the TOIE bit on TIMSK2 to use mask so that only the least significant bit is affected. To slow down the timer, we need to increase the divisor value so that I write 1024 maximum value as divisor to the TCCR2B register.

```
unsigned int count_arm = 127;  
unsigned int count_foot = 127;  
unsigned int count_motor = 127;  
ISR(TIMER2_OVF_vect)  
{  
    if (count_arm != 127)  
    {  
        if (count_arm > 62)  
        {  
            count_arm -= 1;  
        }  
        else  
        {  
            count_arm = 127;  
        }  
    }  
    if (count_foot != 127)  
    {  
        if (count_foot > 62)  
        {  
            count_foot -= 1;  
        }  
        else  
        {  
            count_foot = 127;  
        }  
    }  
}
```

Data analysis:



Normally, the exercise can be divided into three phases. In preparation phase, it begins with the decrease of vertical force by more than 2.5% of feet weight. The preparation phase lasts until the peak of vertical force was reached, which indicate seat-off. The rising phase starts with the peak vertical force. The end of rising phase was defined as the point when the vertical force reaches body weight after decreasing and increasing again. In stabilization phase, the vertical force oscillates around body weight. The end of stabilization phase was defined as the point when the vertical force oscillates inside the corridor plus/minus body weight.

As be seen in the Figure X, this is how the force readings changes during these three phases. The total force is around 180 when sitting on the chair and then reaches up to 500 when seat off, finally oscillates around 450 in the stabilizing phase. During the exercise, the comparison of LeftVal and RightVal can be seen clearly in the line chart. The LeftVal represents the left side of force sensors readings whereas the RightVal represents the right side of force sensors reading on the foot plate. With these data, we can measure how much force they need to stand up and whether they are using the force properly on their feet. Also, we can measure the percentage of sway during the exercise by looking at the time spent in pie chart.

Moreover, the leftDiff and rightDiff represent the difference in force between upper arm and back arm. When people sit in the chair, these two values would not change and only change in the beginning of rising phase. At that time, they would shift the force needed in feet to the arm. After that, they would totally leave the arm from the

chair until the upright position is reached, where people reach the vertical peak force on the foot plate. As you can see, the period for using the force at arm is very short.

Besides, we can measure the approximate weight with use of total force. In experiment, I ask a few of them to try my prototype. The total force is 7.2 times of people's weight during the stabilizing phase. This number could be more accurate when taking more samples.