ECE 445 – Final Report

Rep-Counter for Weightlifting

TA: Ryan May

Fahim Kadhi, Andrew Mast, Ben Rosborough
Group 31

Table of Contents

1 Introduction

1.1 Motivation

Losing track of repetitions is a common problem with weightlifters. The problem arises when all the attention is focused towards achieving correct form and not towards the number of reps that have been completed. This project plans to solve this problem, as well as eliminate the notepad and paper most serious weightlifters use to track their repetitions and the weights.

1.2 Objectives

The goal of the project is to develop a portable electronic device that will count the number of reps for a weightlifter, determine the number of correct and incorrect reps and the log the workout data for future reference. Currently there are mobile applications for smartphones that calculate the number of reps for certain exercises. However, these applications are not able to differentiate between a correct and an incorrect rep. Moreover, these applications are not very accurate and record false positives for a rep-count.

This device will allow the user to choose from a set of a few given of the most common exercises (squats, bench press, etc.). The workout data will be stored and the user will be able to import the data onto his or her computer.

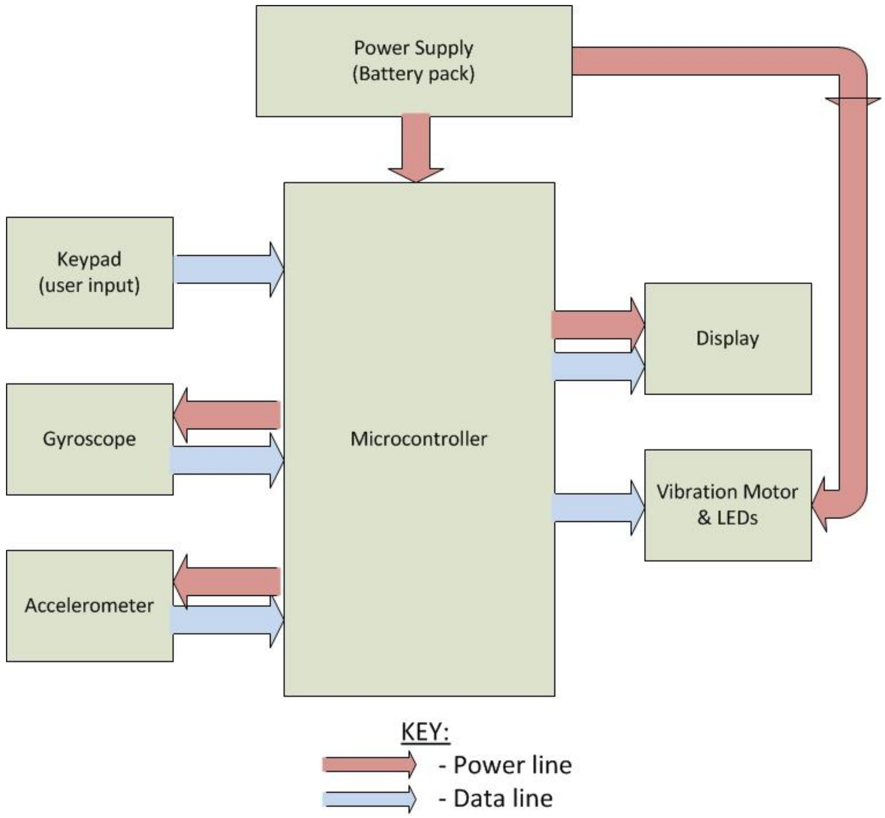
1.2.1 Benefits

- Accurately keeps track of the lifters workout data by logging the reps, weight and sets
- Provides a library of the most common exercises to choose from
- Keeps track of good form and notifies the user if bad form is implemented
- Interfaces with the computer to export workout data from the device into an Excel document

1.2.2 Features

- Good form is maintained by monitoring 5 degrees of freedom of movement by means of an accelerometer and a gyroscope
- Uploads workout data to track progress on an organized spreadsheet
- User notified when desired number of repetitions is achieved via vibration and light
- User notified of bad form via vibration and light
- Streamlined interface to save the number of reps and the weight used using minimal push buttons: up, down, select, back/menu and power

1.3 Block Diagram



2 Design

2.1 General Design Alternatives

There were some things to consider with respect to the design of the device. First, it needed to be small enough to be packaged on the wrist. Initially, the design included a wristband that would house the batteries instead of the project box holding the batteries. The intended design was altered since the available project box could hold the batteries as well as the rest of the unit in one small package. This also eliminated our need to protect external batteries from sweat and conserved the ease of replacing them. At the beginning of the design phase the Arduino Uno seemed like a proper fit for the application at hand. It was initially thought that the Uno had enough I/O pins for the project. It became apparent though that the bigger Arduino Mega was needed due to a large amount of I/O pins as well as a larger memory.

2.2 Circuit Layout

Shown below in Figure 1 is the circuit layout for the design of the system. This shows the connections between all of the devices and how all of the individual components are connected to the microcontroller.

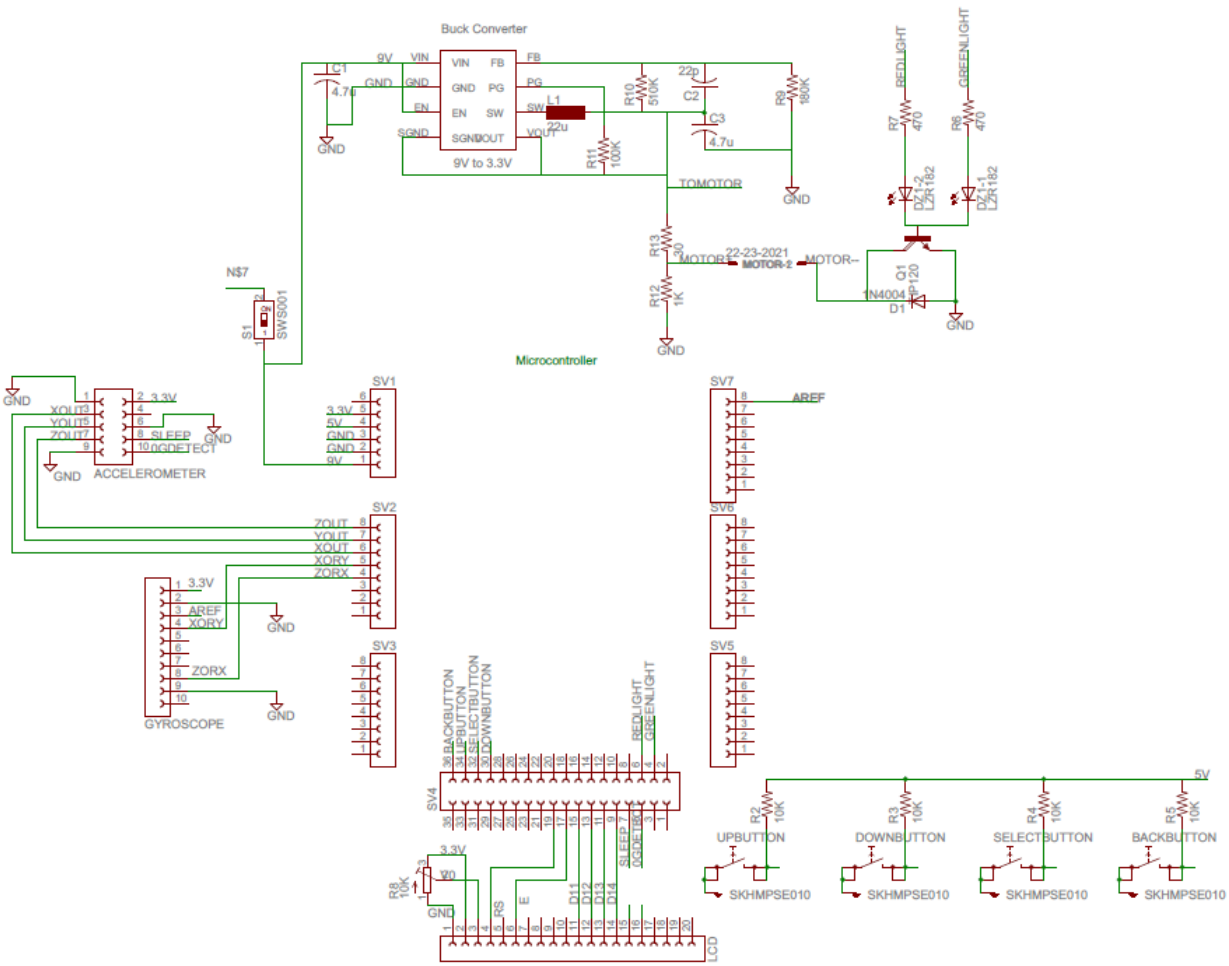


Figure 1 – Circuit Layout

2.3 Design Breakdown

This design consisted of numerous subcomponents. The first design issue of concern was how the device was going to be powered. Secondly, the computing power of the device needed to be taken into consideration. The next thing that needed to be taken into account was the sensors that will be used to track. Finally, the components that involved user interaction were considered: the buttons, LCD screen, LEDs, and vibration motor. Each of these subcomponents has been described in detail in their respective subsections.

2.3.1 Power Supply

The power supply consists of three 3-volt lithium ion batteries connected in series, which supply a nominal voltage of 9 volts. The negative terminal is grounded and the positive terminal is connected to the V_{in} port of the microcontroller. The microcontroller itself contains a voltage regulator that outputs 3.3 volts. This is used to power the accelerometer and the gyroscope. To power the other components an external 3.3 volt regulator is used. This was chosen because the microcontroller safely accepts a range of voltages between 7 and 12 volts and can output 3.3 volts to power the accelerometer and the gyroscope. The vibration motor could have drawn too much current from the microcontroller, so it is powered with the voltage regulator. The lithium ion batteries were selected over 6 AAA batteries because the have a significantly greater energy density and a longer lifetime. After it was determined that the microcontroller could not be used to power the vibration motor, it needed to be determined if the motor could be powered straight from the battery or use a voltage regulator. The vibration motor draws a maximum of 74 mA and needs 1.3 volts to operate. If the battery were connected straight to the motor, the following would result:

$$R = \frac{V}{I} = \frac{7.1}{0.074} = 95.6 \approx 100\ \Omega \qquad P = V \times I = 0.52\ W$$

The power dissipated in this case is too high because the resistors are rated at quarter-watt. If the voltage regulator that outputs 3.3 volts were used, the following would result:

$$R = \frac{2}{0.074} = 27.02 \approx 30\ \Omega \qquad P = 0.148\ W$$

It can be seen here that a 30 Ω resistor results in a safer amount of power dissipated.

Please refer to Table A for the power budget. It is a more detailed analysis of power consumption of each component in the system.

Part	Voltage (V)	Min Current (mA)	Max Current (mA)	Min Power (mW)	Max Power (mW)	%age Part is On	Average Current (mA)	Average Power (mW)
Accelerometer	3.3	0.003	0.4	0.0099	1.32	65%	0.26105	0.861465
Arduino Mega	8.4	1	200	8.4	1680	100%	200	1680
Buck Converter	8.4	0.15	7.5	1.26	63	20%	1.62	13.608
Display	3.3	17.5	18.5	57.75	61.05	100%	18.5	61.05
Gyroscope	3.3	6.8	0.005	22.44	0.0165	65%	2.38325	7.864725
LEDs	5	0	12	0	60	20%	2.4	12
Switch Resistors	3.3	0	1.32	0	4.356	20%	0.264	0.8712
Vibration Motor	1.3	0	74	0	96.2	20%	14.8	19.24
Total		25.453	313.725	89.8599	1965.9		240.228	1795.495

Table A – Power Budget

2.3.2 Microcontroller

The microcontroller that is being used is the Arduino Mega 2560. This microcontroller operates at 5 volts, while accepting anywhere between 7 and 12 volts. It includes an internal voltage regulator that outputs 3.3 volts, which is used to power the accelerometer and the gyroscope. It has 54 digital input/output pins, and 16 analog input pins. The accelerometer and the gyroscope are connected to the analog pins, while the LEDs, vibration motor, LCD screen, and the input buttons are connected to the digital I/O pins. The user selects which lift he or she wishes to perform, how many sets of the lift they will do, and how many reps are in each set. Data from the sensors is processed and the microcontroller keeps track of both correct and incorrect reps. The determination as to what constitutes a correct or incorrect rep is based on comparing the sensor data to known requirements for the lift. The user is notified with the LEDs and the vibration motor of both incorrect reps and when the preset number of reps has been performed. The microcontroller also allows for a small delay before the accelerometer and the gyroscope begin taking data so as to let the lifter get into position. The LEDs and vibration motor notify the user when they can begin. The Arduino Mega contains 256 KB of flash memory, along with 8 KB of SRAM and 4 KB of EEPROM. This memory holds both the program and the data that is stored from the workout. The data to be stored is what kind of lift was performed, what weight it was at, how many sets were performed, and how many correct reps were performed. The main reason for choosing this microcontroller was of its combination of a large memory and many digital I/O pins. The microcontroller that was initially selected did not have enough memory to store both the program and the data, and it did not have enough pins to add external memory. Additionally, this microcontroller has a USB port built in, which makes data transfer much easier.

2.3.3 Accelerometer

The accelerometer used in this project is a 3-axis analog accelerometer. The sensor is a low profile capacitive accelerometer. The mechanical structure, which is made of semiconductor materials, can be modeled as a set of beams attached to a movable central mass that move between fixed beams. These movable beams move from their stationary position when it experiences acceleration. A simplified model of the sensor is shown in Figure 2. Essentially, there are 2 variable capacitances, one on each side of the moving beam. The changes in distance between the beams (increasing area for a capacitor) cause changes in the voltage output, which is then used to calculate the acceleration.

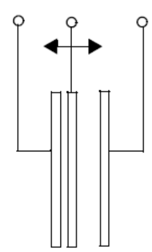


Figure 2 – Simplified model of accelerometer.

This specific unit is already packaged on a carrier board. It has a selectable sensitivity range of either ±1.5g or ±6g sensitivity. Included on the chip is a 0-g detect feature and a sleep pin. The 0-g detect will output a logically high signal when the accelerometer experiences no acceleration in any direction. The sleep pin is internally pulled low to default to a low-power mode. This pin is controlled by an I/O pin from the micro controller and will be used in conjunction with the other outputs to provide accurate data. The accelerometer itself is packaged on a carrier board, which is powered with an input voltage of between 2.2 and 3.6 volts. The 3.3 volt output of the microcontroller powers the accelerometer. The acceleration data from the accelerometer does not drift an appreciable amount. The accelerometer is used to provide data about the tilt angle. The main reasons this accelerometer was chosen were its small size, ease of use, and the fact that it is already packaged on a breakout board.

In order to obtain an accurate g-force from the sensor readings, the data obtained from the sensor is plugged into specific equations. The voltage from the accelerometer is interpreted as a value between 0 and 1023 by the ADC (analog-to-digital converter) in the Arduino microcontroller. This value is then plugged into equation 1 (including equations 2 and 3 for both the other axes) shown below to obtain the g-force.

In reference to equation 1, ‘accelX’ is the value from ADC. ‘Vref’ is the reference voltage, which is basically the scale to which we want to confine the readings to. The microcontroller is given this 3.3V Vref externally from the sensor board’s 3.3V supply. ‘accelXzeroVoltage’ is the nominal voltage that it gives off when it is stationary. Lastly, the ‘accelSensitivity’, given by the part manufacturer, is the voltage response of the unit per g-force (in mV/g). The equations for the other axes follow the same variable assignment.

$$accelerationX = \frac{\frac{accelX * Vref}{1023} - accelXzeroVoltage}{accelSensitivity} \quad \text{-- Equation 1}$$

$$accelerationY = \frac{\frac{accelY * Vref}{1023} - accelYzeroVoltage}{accelSensitivity} \quad \text{-- Equation 2}$$

$$accelerationZ = \frac{\frac{accelZ * Vref}{1023} - accelZzeroVoltage}{accelSensitivity} \quad \text{-- Equation 3}$$

For the purpose of this project, the accelerometer is also used to obtain tilt angle measurements. The X, Y and Z acceleration components are used in equations 4 through 6 to obtain the pitch, yaw and roll angles. Using basic trigonometric formulas, the angles can be calculated in radians. This value is then converted to degrees (the part b of equations 4 to 6). To calculate the pitch and yaw angles, the magnitude of the acceleration was required. This is calculated using the formula shown in equation 5c.

$$accelAngleZrad = \tan^{-1}\frac{accelerationY}{accelerationX} \quad \text{-- Equation 4a}$$

$$accelAngleZdeg = accelAngleZrad * \frac{180}{\pi} \quad \text{-- Equation 4b}$$

$$accelAngleXrad = \sin^{-1}\frac{accelerationX}{acceleration} \quad \text{-- Equation 5a}$$

$$accelAngleXdeg = accelAngleXrad * \frac{180}{\pi} \quad \text{-- Equation 5b}$$

$$acceleration = \sqrt{(accelerationX)^2 + (accelerationY)^2 + (accelerationZ)^2} \quad \text{-- Equation 5c}$$

$$accelAngleYrad = \sin^{-1}\frac{accelerationY}{acceleration} \quad \text{-- Equation 6a}$$

$$accelAngleYdeg = accelAngleYrad * \frac{180}{\pi}$$ -- Equation 6b

One of the disadvantages of calculating angles using the accelerometer is the similarity between the pitch and the yaw angles. Since the accelerometer's reference is earth's gravity pull, any rotation about the Z-axis would result in the same angle change. In order to differentiate between these two angles, the gyroscope readings were used to determine whether the device is moving over the pitch or yaw angles.

At different stages in the code, since the sampling rate is relatively high, most of the data obtained and calculated is averaged out frequently in order to obtain a more stable response.

2.3.4 Gyroscope

The gyroscope used in this project is a dual axis (pitch and yaw) analog gyroscope. The functionality of this sensor is based on the Coriolis effect- a vibrating object tends to continue vibrating in the same plane as its support rotates.

This specific unit is already packaged on a carrier board. It has a selectable sensitivity range of either ±100°/s or ±400°/s. The two axes are pitch and yaw, which are rotation about the X and Z axes. Also included on the chip is a power-down pin for use in low-power applications. This will not be utilized in the course of this project and it will be shorted to ground. The gyroscope itself is packaged onto a carrier board that is powered with an input voltage of between 2.7 and 3.6 volts. Again, the 3.3 volt output of the microcontroller is used to power the gyroscope. The reasons this gyroscope is being used are the same as the accelerometer- small size, ease of use, and the carrier board.

In order to obtain an accurate angular rate from the sensor readings, the data obtained from the sensor is plugged into specific equations. The voltage from the gyroscope is interpreted as a value between 0 and 1023 by the ADC (analog-to-digital converter) in the Arduino microcontroller. This value is then plugged into equation 7 (including equation 8 for the other axis) shown below to obtain the angular rate.

In reference to equation 7, 'gyroX' is the value from ADC. 'Vref' is the reference voltage, which is basically the scale to which we want to confine the readings to. The microcontroller is given this 3.3V Vref externally from the sensor board's 3.3V supply. 'gyroXzeroVoltage' is the nominal voltage that it gives off when it is stationary. Lastly, the 'gyroSensitivity', given by the part manufacturer, is the voltage response of the unit per angular rate (in mV/deg/s). The equations for the other axes follow the same variable assignment.

$$gyroXrate = \frac{\frac{gyroX * Vref}{1023} - gyroXzeroVoltage}{gyroSensitivity}$$ -- Equation 7

$$gyroZrate = \frac{\frac{gyroZ * Vref}{1023} - gyroZzeroVoltage}{gyroSensitivity}$$ -- Equation 8

Due to the inherent drift of the gyroscope, the accelerometer's data will be used to stabilize the gyroscope's data to more accurately obtain the angular rate. If the accelerometer is showing data that indicates the unit is stationary, the gyroscope will be re-zeroed.

Initially, the goal was to obtain the absolute angular measurement from the gyroscope and the accelerometer. However, after using the trapezoid method to integrate the data from the gyroscope, the results obtained had a strong drift component. Therefore, only the angular rates were used to determine good and bad form.

At different stages in the code, since the sampling rate is relatively high, most of the data obtained and calculated is averaged out frequently in order to obtain a more stable response.

2.3.5 Keypad (user input)

User input consists of five switches. The power button is a single pole, single throw, normally open toggle switch. The other four input buttons are single pole, single throw, normally open momentary switches. The power switch is connected in series with the battery. When the switch is closed, the battery is connected to the microcontroller, the vibration motor, and the LCD screen. This button is a rocker switch, so once it is pushed remains closed until it is pushed back the other way. The other four input buttons are connected to four digital inputs. The inputs are pulled high when the switches are open. When the switches are closed, the digital inputs read low because current flows to ground instead of the microcontroller. The microcontroller recognizes when the input reads low and respond with the appropriate action. The input pins are debounced by writing a subroutine into the microcontroller. The reason for choosing the specific switches is that it fulfills the simple mechanical requirement of the project. A normally open toggle switch is needed for the power button and normally open momentary switches for the input buttons. A large amount of switches are rated high enough for our design, so that was not of as much concern.

2.3.6 LEDs & Vibration Motor

The LEDs and the vibration motor notify the user of when to begin lifting, if an incorrect rep has been performed, and when the preset number of correct reps has been performed. There are two LEDs driven by digital output pins of the microcontroller. The vibration motor is powered by a 3.3 volt output from an external voltage regulator. An NPN transistor is used to regulate the current flow through the motor. When one of the digital outputs to the LEDs is high the transistor is switched on, allowing current to flow through the motor. Two different color LEDs are used to indicate different things to the user. When an incorrect rep has been performed and identified, a red LED is pulsed once, along with the vibration motor. When the preset number of reps has been performed, the green LED and the vibration motor are pulsed three times. Additionally, after the user has selected their lift, selected the number of reps, hit the start button, and the small time delay to allow the user to get into position has passed, the green LED and the vibration motor are pulsed three times to indicate they can now begin the lift. The only decision in choosing the LEDs was the color. The main reason for choosing the specific vibration motor was its small size.

2.3.7 LCD Screen

The LCD output text information to the user. It shows two lines of sixteen black characters against a white background. The display is used to show the menu options. The LCD logic is powered from the microcontroller, as is the backlight. A 10k Ω trimpot used to control the contrast of the backlight. The R'/W signal is pulled high because text is always being written to the LCD. The register select signal comes from a digital output of the microcontroller. This signal is used for various functions of the display. Pins DB4 through DB7 come from digital outputs of the microcontroller and are used to display different characters on the screen. A library is being utilized that will allow us to implement the user interface with only the four input pins. This LCD screen was chosen for a variety of reasons. The most important ones are that it can be powered with a 3.3 volt input and that its small size fits in with the physical scope of our project. Additionally, it can utilize a library to allow us to easily display text.

2.3.8 PCB

The printed circuit board is implemented as a shield that mounts on top of the microcontroller. The design was created using Eagle. On the PCB the rest of the circuitry will be mounted to keep the design as small as possible. The accelerometer and gyroscope will be oriented in such a way that both of them will be able to detect the proper axis of motion that is desired for the lifts. The PCB will provide all interconnects for the device to function properly. The PCB layout is located in Figure 1 of the appendix.

2.3.9 User Interface

The user interface combines the functionality of the microcontroller, LCD, and the keypad. The four buttons available to the user are up, down, select, and back. The up and down buttons are used to scroll through the menu. The select button selects an option and advances to the next menu level. The back button will return to the previous menu level. The interface allows the user to select the initial function, be it performing a new workout, uploading a saved workout to a computer, or clearing the memory of the device. If the user decided to perform a new workout they are taken to the next level of the user interface where they select which lift they will perform. The five choices are squat, bench press, deadlift, bicep curls, or tricep extensions. The selection of a lift will set the value of a variable called lift. Lift will equal 1 for squat, 2 for bench press and so on. After the lift has been selected the user then selects the number of sets they will perform. The maximum is ten due to constraints on writing to and reading from the memory. Pressing the up and down buttons will change the character displayed on the screen as well as the variable to store the number of sets and setgoal. When the select button is pressed the menu advances to the next level where the user selects the number of reps.

To select the number of reps the code first checks to see if setgoal=1. If this is true the user is immediately asked to input the number of reps. If setgoal is greater than 1, the user is asked if the number of reps is the same for each set. If it is, the user can select the number of reps once and a variable called repcheck is set to 0. If the reps are different for different sets, repcheck is set to 1, and the user is asked to select the number of reps for set 1. After a set is completed, if repcheck is equal to 1 the user returns to this menu level to select the number of reps for the next set. The actual process for selecting the number of reps is the same as for sets except reps can go up to 99.

After selecting the number of reps the user then selects the amount of weight he or she will be lifting. The procedure for selecting the amount of weight being lifted is the same as for selecting the number of reps, except there are three digits to select as opposed to just two.

After selecting the amount of weight, the user is prompted to hit a button to begin their lift. There is a five second countdown to allow them to get into position and then the device begins taking data. Depending on which lift the user is performing (the value of the “lift” variable) there are different conditions that need to be met to indicate that a rep has performed. Once a rep has been detected the device checks to see if it was performed with good or bad form. The number of reps performed is checked against the number of reps we wish to perform for the set. If we have not finished the set, we begin taking data again. If we have finished the set, we write the number of correct reps performed into the memory. We then check to see if we have completed the total number of sets. If we have, we are done with the exercise and we return to the beginning of the menu. If we are not, we check the values of the repcheck and weightcheck variables. If repcheck and weightcheck both equal 0, we wait for the user to press a button to begin the next set. If either repcheck or weightcheck are equal to 1, we need to go back and select a new number of reps and/or weight for the upcoming set.

The memory write scheme is explained in a bit more detail below. Data for squats is stored in memory locations 1-30. These 30 locations are allocated for the three categories of data we need to store, up to ten locations for each category. The number of correct reps performed is stored in locations 1-10, the number of total reps performed is stored in locations 11-20, and the weight is stored in locations 21-30. If we perform less than ten sets we will only utilize a number of memory locations corresponding to the number of sets we performed. The data for the other lifts is stored in locations 101-130, 201-230, 301-330, and 401-430 for the other four lifts respectively.

The second option at the top level of the menu is to upload the workout to a computer. This simple subroutine allows the computer program called Gobetwino to read from the EEPROM and write it to a text file on the computer.

The third option at the top level of the menu is to clear the EEPROM to get ready for the next workout. The only thing this subroutine does is loop through memory locations 1-500 and reset the value in memory to zero.

2.4 Schematic & Flowchart

Highlighted below is the schematic layout as well as the menu flow for the user. Also a flowchart for the entire system is shown below.

2.4.1 Schematic

The schematic below in Figure 3 is a basic overview of how the circuitry was implemented between the major components.

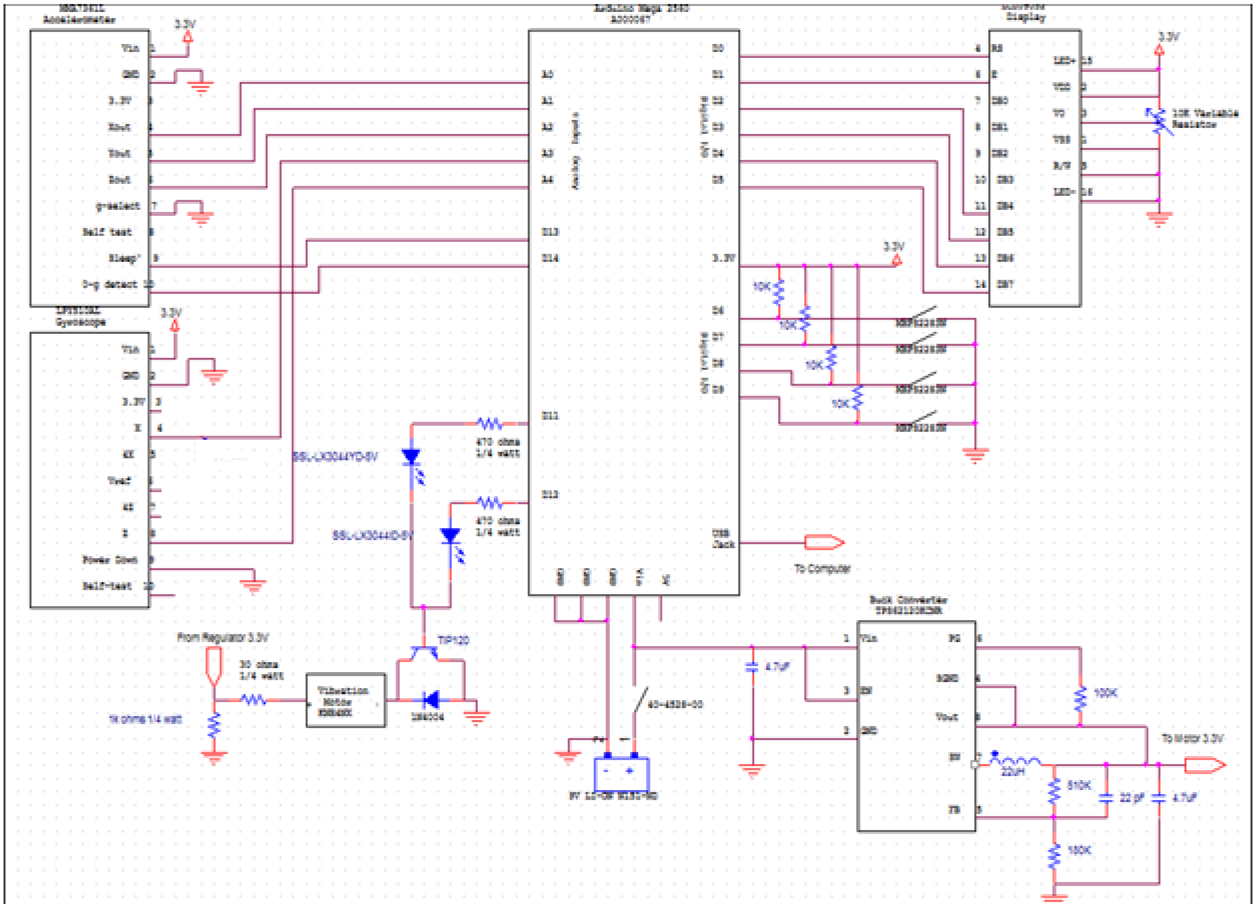


Figure 3 – System Schematic

2.4.2 System Flowchart

The system flowchart can be viewed in Figures 2-7 in the Appendix.

3 Verification

3.1 Testing & Verification

Functional tests for all of our components are shown in the Appendix in Table 1. These tests were performed on all of our parts. On top of these initial tests that were in the design review, other tests that were deemed relevant during the course of building our project are included. These mainly consisted of testing the PCB and the functionality of the user interface. The PCB had to be constantly tested to ensure that the solder joints were mechanically sound and the appropriate connections were being made on the board. The continuity function of a multimeter was utilized for this purpose.

For the user interface, the tests involved making sure all the variables were reset when necessary and incremented/decremented at the appropriate times. To do this, variables that represented the workout data were printed to the serial monitor and compared to the variables that were printed on the LCD screen. Additional tests for the user interface were to make sure that the microcontroller was properly writing to and reading from the EEPROM.

An additional test that we had to perform was testing if the switches were debounced. This was performed simply by pressing the button and seeing how much the number on the display was incremented. If the display incremented by more than a single count, the delay used to debounce the switch needed to be longer.

3.2 Results & Discussion

3.2.1 Accelerometer Results

Initial testing results indicated that the accelerometer was functioning properly within the confined performance criteria stipulated by the part manufacturer. The following were the results obtained:

Zero-rate level: The zero-rate level on the X-axis was 1.625V. This was almost the same as the expected 1.65V zero-rate voltage. The Z-axis is on channel 2 and the X-axis is on channel 1 of the oscilloscope. Zero-rate voltage on the Z-axis is 0.59V which is not exactly the expected reading (expected voltage reading shown in equation 9). The percentage difference is quite significant (approximately 30%). However, for the purpose of this project, so long as this zero voltage doesn't deviate much and stays constant, it wouldn't cause any problems in the calculations as we're dealing with relative changes in acceleration. Please refer to Figures 3a and 3b in the Appendix.

$$1.625V - \left(1[g] * 0.8 \left[\frac{V}{g}\right] \right) = 0.825V \qquad \text{-- Equation 9}$$

Jerk-response test: The accelerometer is jerked back and forth at a high speed over a short distance and the results obtained are as expected. The unit was jerked at an approximate rate of 1.10g (a little more the gravitational force). This yields a sensitivity of 0.786mV/g (1.10g/1.4V), which is almost the same as the expected sensitivity of 800mV/g. Possible causes of the difference could be due to the temperature because the same tests were carried out after long period of time; this test however yielded a sensitivity of approximately 0.82mV/g. The same result is obtained when the accelerometer is jerked at a slow speed over a longer distance (half a meter). The following are the results for the Z axes only. Please refer to Figures 4a and 4b in the Appendix.

Tolerance test: The same tests were carried out at different input voltages. The accepted voltage range for this unit according to the part manufacturer is 2.2V to 3.6V. The figures show one test at 2.2V and the second one at 3.6V. This time, the unit was jerked alternatively on both axes. The device performed correctly without any change in output values or noise level. Please refer to Figures 5a and 5b in the Appendix.

3.2.2 Gyroscope Results

Initial testing results indicated that the gyroscope was functioning properly (with differences in zero-rate voltages) within the confined performance criteria stipulated by the part manufacturer. The following were the results obtained:

Zero-rate level: The zero-rate level on the X-axis was 1.219V. This differed quite significantly from the expected response of 1.65V (note that the Z-axis is on channel 2 and the X-axis is on channel 1 of the oscilloscope). The zero-rate voltage on the Z-axis is the same and has the same percentage difference (approximately 26%). However, just as for the accelerometer, for the purpose of this project so long as this zero voltage doesn't deviate much and stays constant, it wouldn't cause any problems in the calculations as we're dealing with relative changes in acceleration. Please refer to Figures 6a and 6b in the Appendix.

Performance test (fast rotations): The gyroscope is rotated back and forth at a high speed and the results obtained are as expected. The unit was rotated at an approximate rate of 400º/s (due to extremely fast fluctuations). This yields a sensitivity of approximately 2.5mV/deg/s ($\frac{1V}{400º/s}$). The rotation speed estimated is a very crude approximation as it's based on human movements. More tests would indicate slightly different sensitivities. Please refer to Figures 7a and 7b in the Appendix.

Performance test (slow rotations): The gyroscope is rotated back and forth at a low speed and the results obtained are as expected. The unit was rotated at an approximate rate of 250º/s. This yields a sensitivity of approximately 2.5mV/deg/s ($\frac{0.65V}{250º/s}$). The rotation speed estimated is a very crude approximation as it's based on human movements. More tests would indicate slightly different sensitivities. Please refer to Figures 8a and 8b in the Appendix.

Tolerance test: The same tests were carried out at different input voltages. The accepted voltage range for this unit according to the part manufacturer is 2.7V to 3.6V. The figures show one test at 2.7V and the second one at 3.6V. This time, the unit was jerked alternatively on both axes. The device performed correctly without any change in output values or noise level. Please refer to Figures 9a and 9b in the Appendix.

3.2.3 LCD Results

Test for the LCD indicated proper operation with 3.1, 3.3, and 3.5 volts at the power pin for the LCD logic, and 3.3 volts at the backlight power pin. There was no flickering on the screen when characters were displayed. The only problem was a contrast defect. When scrolling through a certain menu level most of the options would be displayed fine, except for the last few. For these the display went dark and we had to adjust the contrast manually with the trimpot. Two different trimpots were tested, as well as connecting the contrast pin to both power and ground. Neither of them solved problem, and the conclusion was made that this was an inherent problem with the LCD screen.

3.2.4 Keypad (User Input) Results

Testing of the buttons indicated proper operation in all the different scenarios, for the push-buttons as well as the power button. When debouncing the switches, a few different delay times were tested until one was found one that did not cause the microcontroller to register multiple button presses. This was approximately 400 ms.

3.2.5 Vibration Motor & LED Results

Testing of the motor and the LEDs indicated that they were all functioning correctly. A simple test program was run to blink the LEDs individually and at the same time. All three components worked properly. After this both LEDs and the vibration motor were turned on for 30 minutes straight. During the course of this test the brightness of the LEDs did not diminish and the strength of the vibration of the motor was constant.

3.2.6 User Interface Results

Verification of the operation of the user interface was somewhat tedious but it did pass all the tests. The variables that stored workout data were printed to a serial monitor and compared to the characters that were displayed on screen. The number of sets, reps, and weight was scrolled through on the LCD and compared to the values being printed to the serial monitor. In all cases the numbers on the LCD screen matched the numbers on the serial monitor. The variables were also checked when the user back through the menu to ensure that they were reset when needed to be. Again, in all cases the variables on the screen and the serial monitor matched each other and were reset at the proper places.

Initial testing for proper writing to and reading from memory indicated that much of the code for that was not written properly. Sample data that we hard coded into the memory was read correctly, but when an actual workout was performed neither of the memory functions worked properly. A few flaws were found in the coded to write to and read from the memory and were fixed. Another test workout that was performed indicated that both the memory functions were working properly.

3.2.7 Microcontroller & Data Analysis Results

Verification of the operation of the microcontroller was the most extensive part of our project. The lower level tests all passed with no trouble. Testing of interfacing between the microcontroller and the LCD screen, the vibration motor, the LEDs, and the keypad is explained in the above sections. Verification of proper communication with the sensors was a bit more involved. Test programs to print acceleration, angular rate, and tilt angles to the serial monitor were run. Analysis of the data collected indicated that the microcontroller was interpreting the voltage values from the sensors correctly and outputting proper acceleration and angle values.

After verifying correct data from the sensors, data for specific lifts was taken. The program that was run constantly took in data and printed a certain variable to the serial monitor. Multiple good and bad reps were performed for all five exercises to determine what kind of acceleration, angular rate, and angle values would occur when the user was actually performing the lifts. For squats the total acceleration was monitored. For bench press and deadlift the ADC voltage values of the acceleration in the y-direction were monitored. The ADC values were used instead of the actual acceleration because it gave slightly better resolution. Finally, for bicep curls and tricep extensions the tilt angle about the z-axis was monitored. Examples of the data that was recorded for all five lifts is shown in Figures 15, 16, and 17 in the appendix.

The sensors were able to detect when a rep occurred for all five lifts. Analysis of the data revealed that when a good or bad rep was performed, there was a significant spike in the data that was being observed. This made it rather easy to detect the actual rep. Differentiation between good and bad form was a bit more difficult for some of the lifts however. For bicep curls and tricep extensions the angular rates provided by the gyroscope were monitored. It was decided that if too quick of a rep was detected, that would indicate bad form. Bad form detection for these two lifts was fairly consistent. For both squats and deadlift the angle of the device was monitored. If the angle was over a certain threshold when a rep was performed, this indicated that the user was tilting or leaning too much in a bad direction and that a bad rep needed to be counted. Bad rep detection for squat and bench press was not as reliable as for bicep curls and tricep extensions, but bad form was able to be detected. Deadlifts were a slightly more difficult problem. Bad form on a deadlift involves the angle of the back and the hips more so than anything to do with the arms or wrists. Because of this it was determined that no bad form detection was possible with deadlifts.

4 Cost Analysis

4.1 Parts

Part Description	Model Number	Quantity	Price Per Unit($)	Total
100KΩ Resistor	ERD-S1TJ104V	1	0.01	0.01
10KΩ Pot Core Resistor	3352K-1-103LF	1	0.83	0.83
10KΩ Resistor	CF14JT10K	4	0.005	0.02
180KΩ Resistor	ERD-S1TJ184V	1	0.01	0.01
30Ω Resistor	CF14JT30R0	1	0.005	0.005
470Ω Resistor	CF14JT470R	2	0.005	0.01
510KΩ Resistor	ERD-S2TJ514V	1	0.06	0.06
Accelerometer	MMA7361L	1	11.95	11.95
Arduino Mega 2560	A000067	1	38.95	38.95
Battery	N151-ND	3	3.5	10.5
Battery Holder	708-1412-ND	3	1.64	4.92
BJT	TIP120	1	0.68	0.68
Buck Converter	TPS62120DCNR	1	2.8	2.8
Capacitor 22pF	ECC-NVS220JG	1	0.15	0.15
Capacitor 4.7uF	ECC-NVS047JG	1	0.18	0.18
Diode	1N4004	1	0.016	0.016
Gyroscope	LPY510AL	1	19.95	19.95
Inductor 22uH	TSL0808RA-220K1R7-PF	1	0.25	0.25

LCD Display	LCD09052	1	14.95	14.95
LED Red	SSL-LX3044ID	1	0.63	0.63
Power Switch	40-4528-00	1	0.73	0.73
Push Button Switches	MHPS2283N	4	0.31	1.24
Vibration Motor	KHN4NX	1	4.7	4.7
Zener diode	1N5819	1	0.07	0.07
			Total:	106.88

4.2 Labor

Name	Rate (per hour)	Hours	Total	Multiplier (x2.5)
Andrew Mast	$40	200	$8,000	$20,000
Ben Rosborough	$40	200	$8,000	$20,000
M. Fahim Kadhi	$40	200	$8,000	$20,000
			Total:	**$60,000**

4.3 Grand Total

Entity	Total
Labor	$60,000
Parts	$106.88
Grand Total:	**$60,106.88**

5 Conclusion

5.1 Accomplishments

There were plenty of obstacles encountered during the course of this project. The first obstacle was obtaining meaningful data from the gyroscope. Obtaining steady and accurate values without significant drift posed to be a major challenge. We were able to acquire accurate angular rate data.

The next major milestone was determining threshold data to detect whether a rep was performed, and if it was with good form or bad. This involved much testing with a barbell and the device constantly taking data. After much testing we were able to determine exactly what kind of data we should reasonably expect to obtain during the course of a lift and program our device to detect reps accordingly.

A third accomplishment was expanding the exercise library to five different lifts. Some of the lifts, such as bicep curls and tricep extensions involved very similar movements and offered distinct spikes in data when a rep was performed. The other three lifts proved to be much more of a challenge in determining how exactly to interpret the data.

5.2 Uncertainties

Since the device is dependent on human movements, there could be cases that we have not considered. If the user does something that the program isn't set up for, the device could count unexpectedly.

Some lifts involved movements that proved to be very difficult to determine reps and bad form accurately. Sudden erratic spikes in the sensor outputs could be interpreted as incorrect reps. Moreover, inherent drift in the gyroscope could surpass conditional boundaries set for each specific exercise.

5.3 Ethical Considerations

The following are the codes that are concerned with this project:

Code 1. to accept responsibility in making decisions consistent with the safety, health and welfare of the public, and to disclose promptly factors that might endanger the public or the environment;

Code 9. to avoid injuring others, their property, reputation, or employment by false or malicious action;

This product needs to be safe for the user. Therefore all circuitry needs to be contained in such a manner that no sweat or moisture will be able to access the circuitry that could shock and harm the user. It also needs to be small enough so not to distract or throw the user off balance because of device weight and size.

5.4 Future Developments

This senior design project had many successes, and with these successes come new opportunities to make the rep-counter better. The device could be packaged into a much smaller case. The microcontroller's size was the main reason behind the bulk of the device. This could be easily taken care of by downsizing to a less complex and smaller microcontroller; the Arduino Uno is a possible replacement. The size of the unit could also be reduced if a rechargeable battery pack was used instead of the cylindrical CR 123s. This battery would be much slimmer and would reduce the height of the case.

The Gobetwino program turned out to be sufficient for the purpose of this project's application. It could be taken a step further by organizing the data obtained into a file that is compatible with Microsoft Excel or another spreadsheet program.

The most important future development in this project would most likely be to incorporate different types of sensors, as well as more accurate ones. We did not utilize a magnetometer at all in this project, which could provide us with three additional degrees of freedom to track our motion. In addition to this, more accurate sensors would help as well. They gyroscope contained a large amount of inherent drift that we were not able to overcome. While most gyroscopes contain some amount of drift, higher quality ones have less and thus are more suitable for our application.

6 References

Starlino (2009, Dec.). A Guide To using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications. Starlino Electronics [Online]. Available: http://www.starlino.com/imu_guide.html

Starlino (2010, Jan.). Arduino code for IMU Guide algorithm. Using a 5DOF IMU (accelerometer and gyroscope combo). Starlino Electronics [Online]. Available: http://www.starlino.com/imu_kalman_arduino.html

Vanzati (2011, Dec.). Integrating the gyro to obtain the angular position. Arduino Education [Online]. Available: http://scuola.arduino.cc/en/content/integrating-gyro-obtain-angular-position

Arduino. Language Reference [Online]. Available: http://arduino.cc/it/Reference/HomePage

7 Appendix

Figure 1: PCB Layout

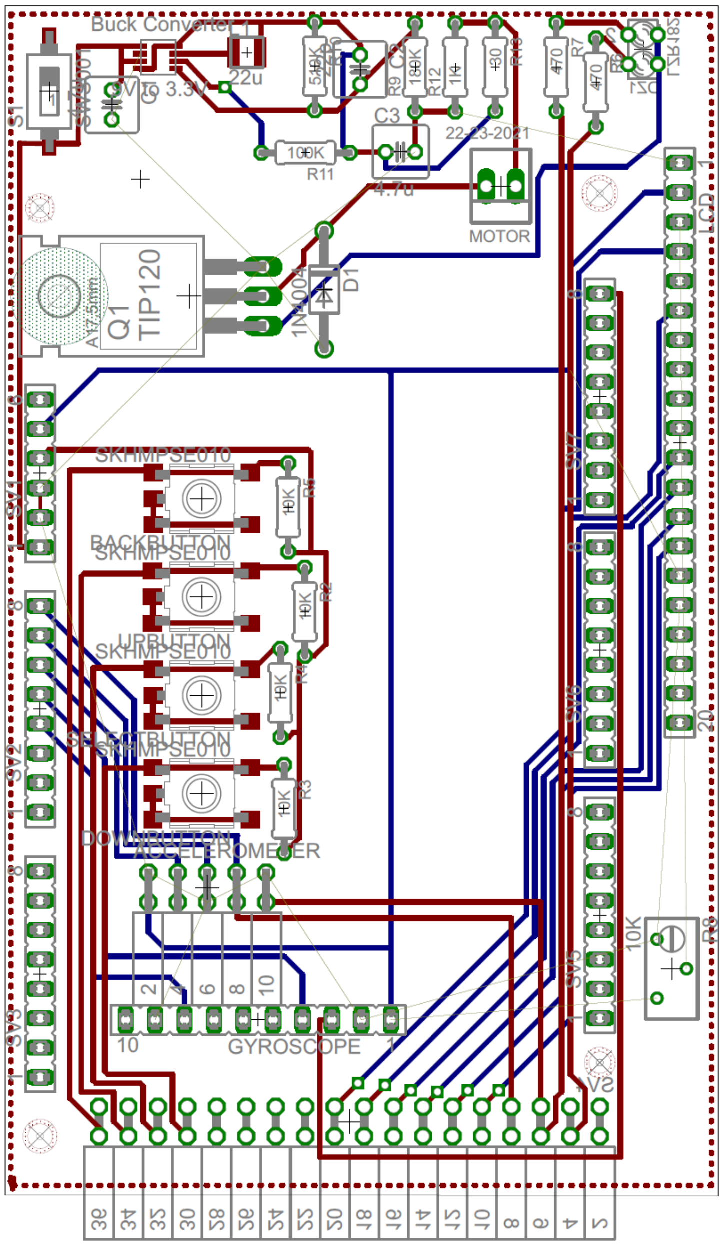


Figure 2: System Flowchart 1

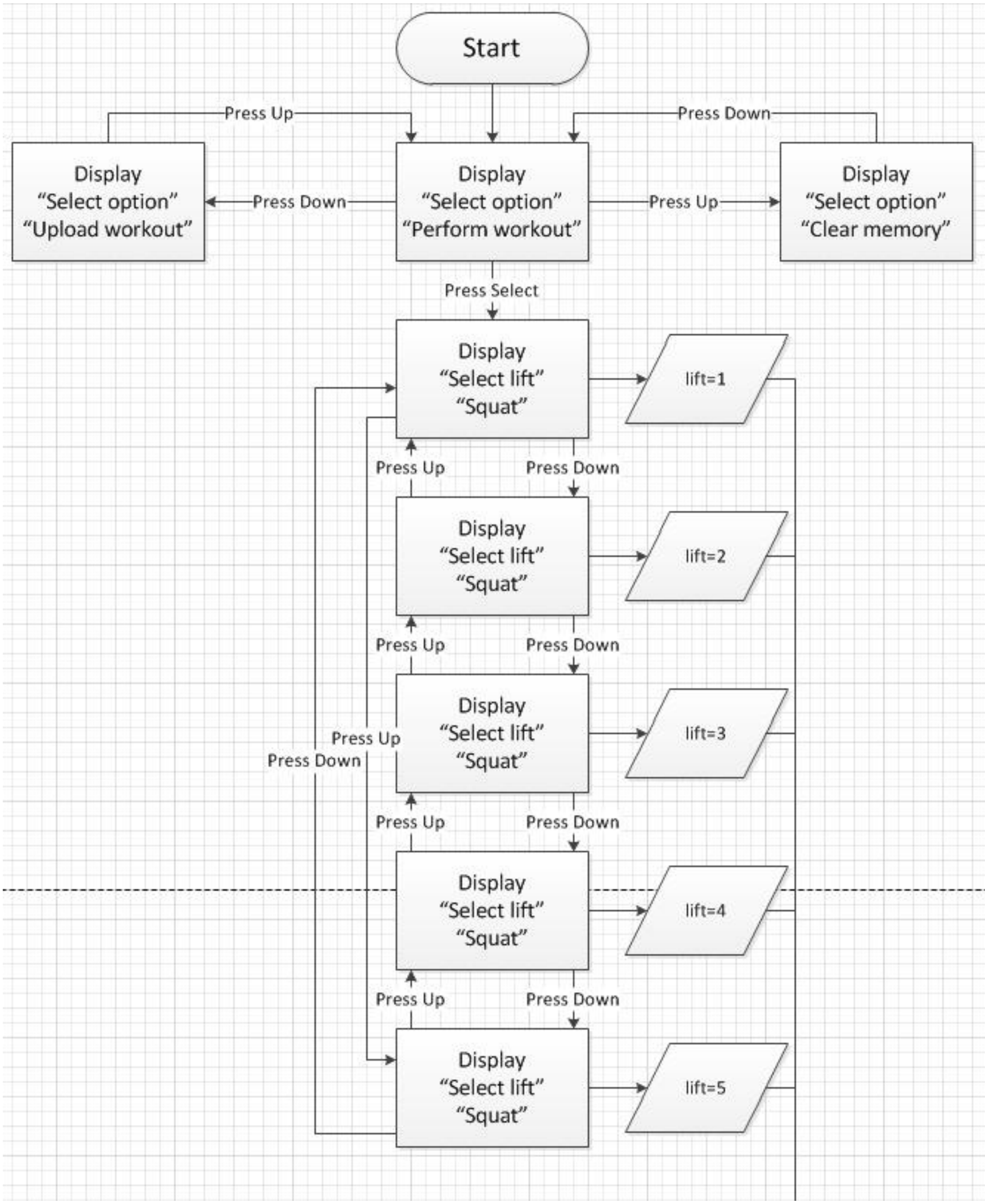


Figure 3: System Flowchart 2

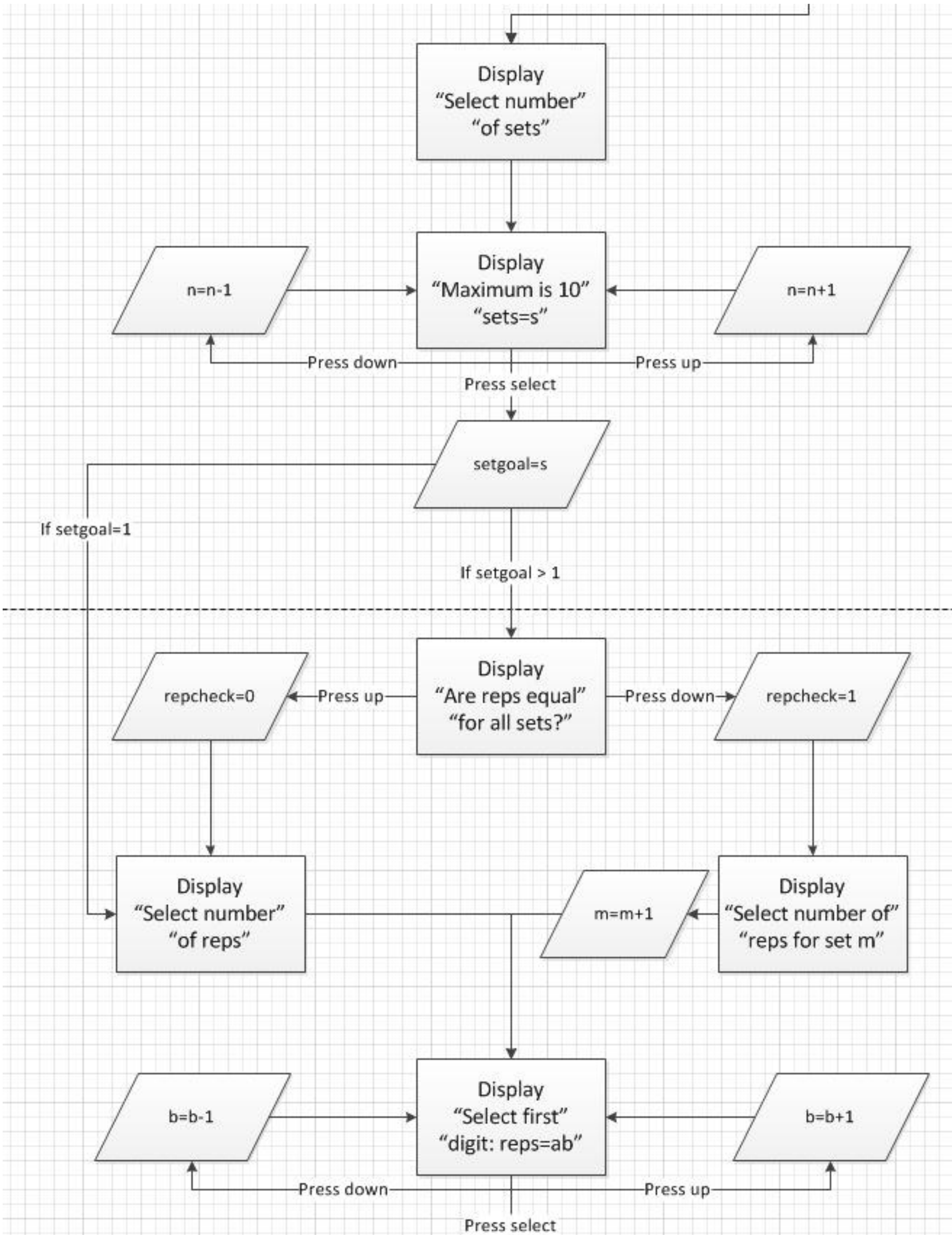


Figure 4: System Flowchart 3

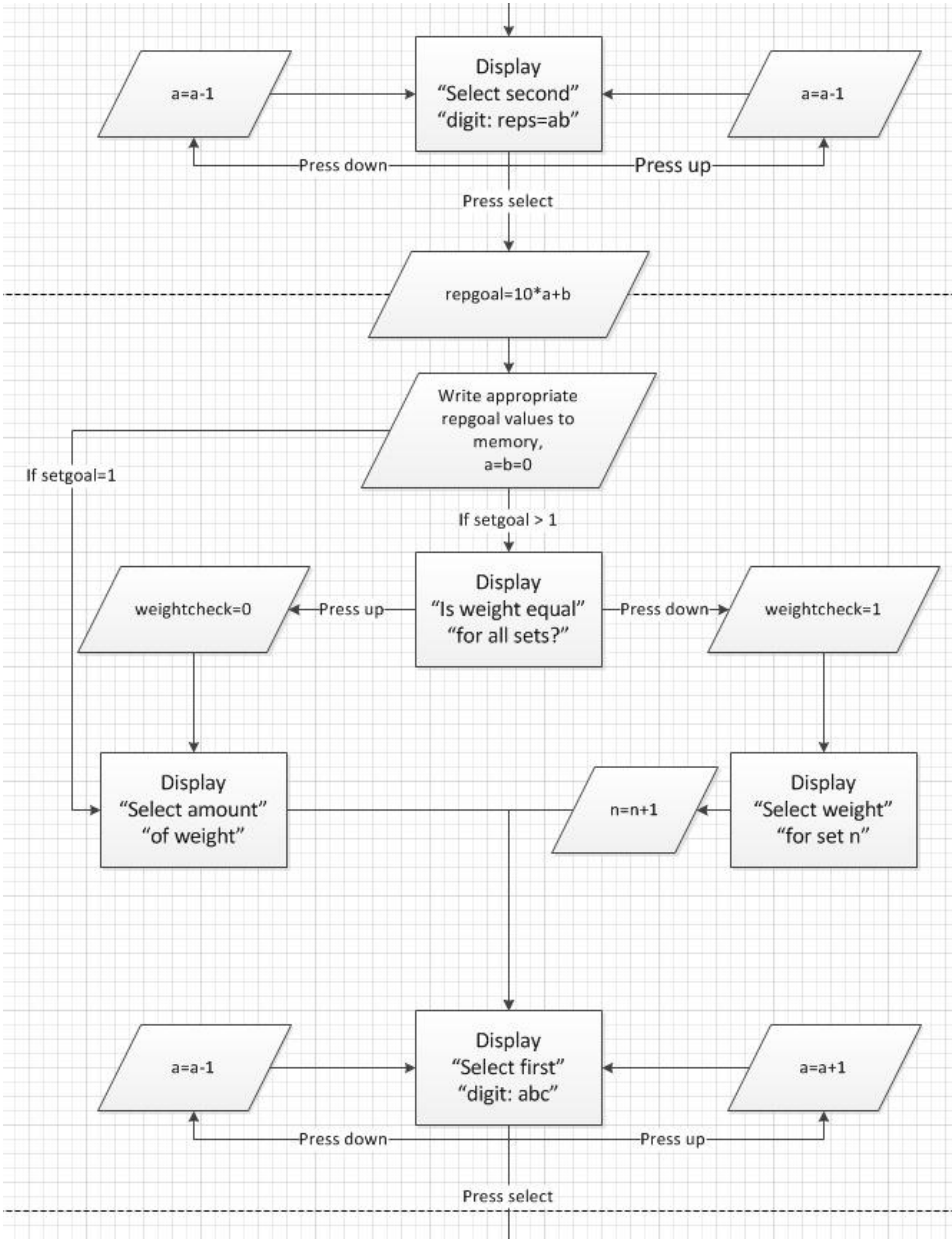


Figure 5: System Flowchart 4

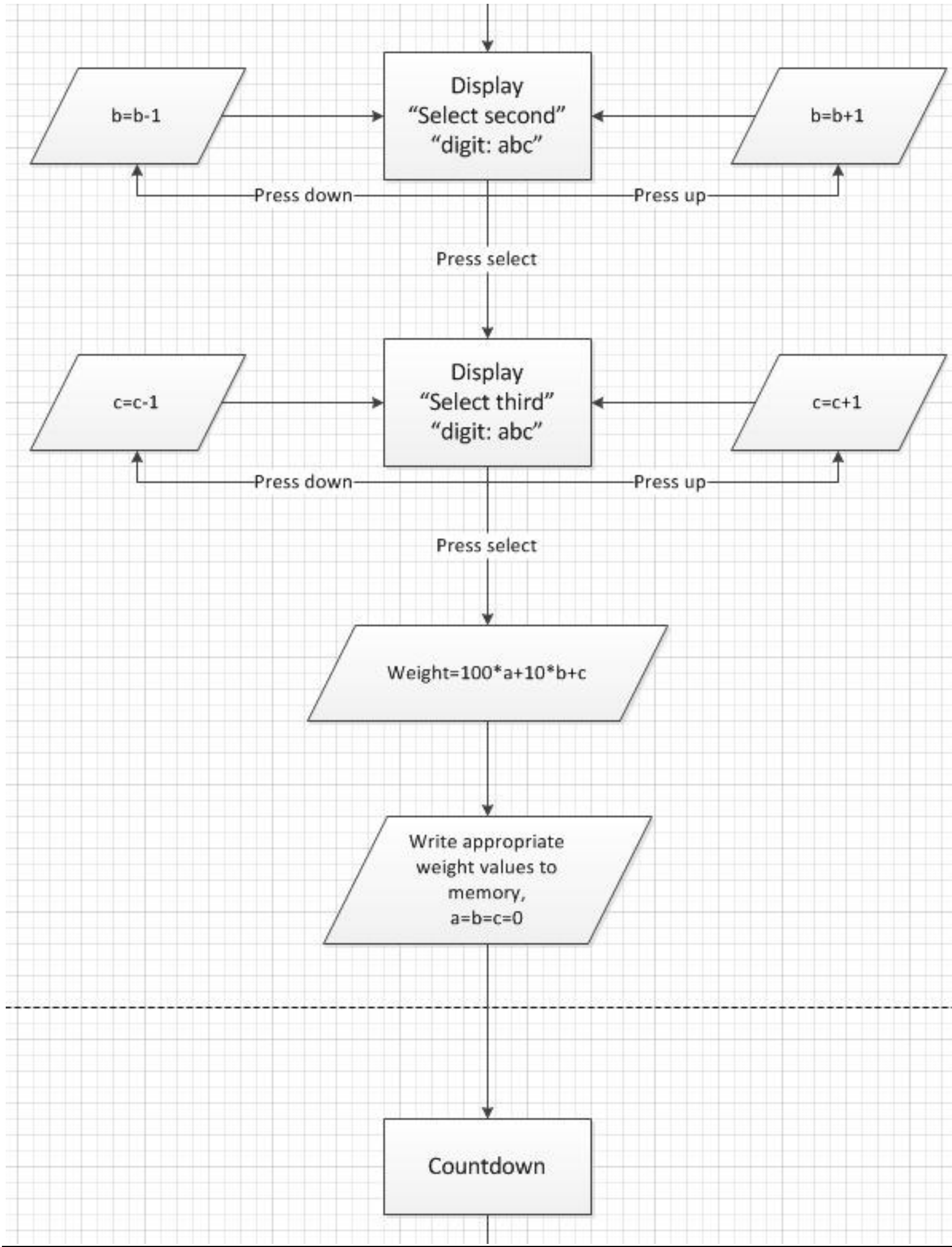


Figure 6: System Flowchart 5

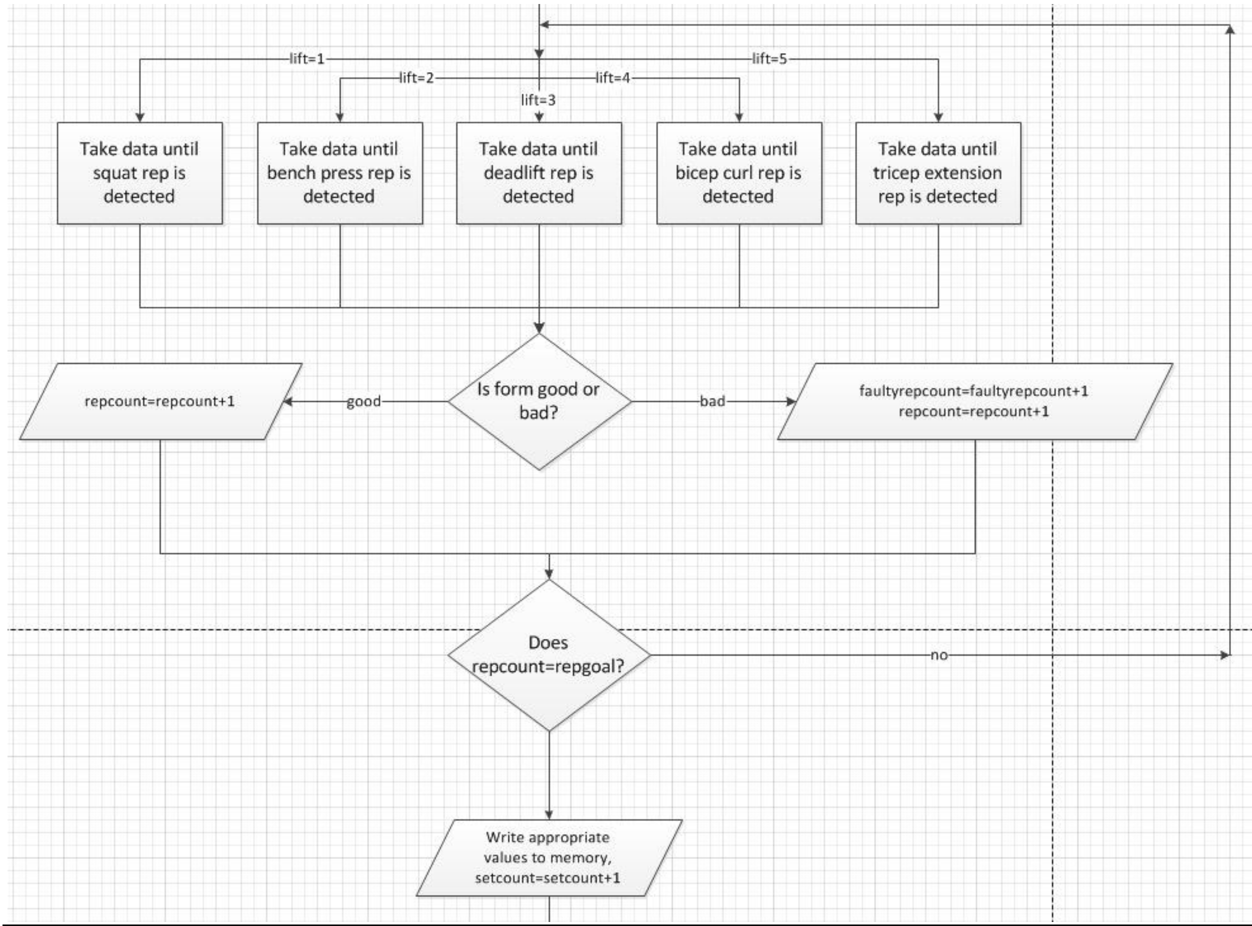
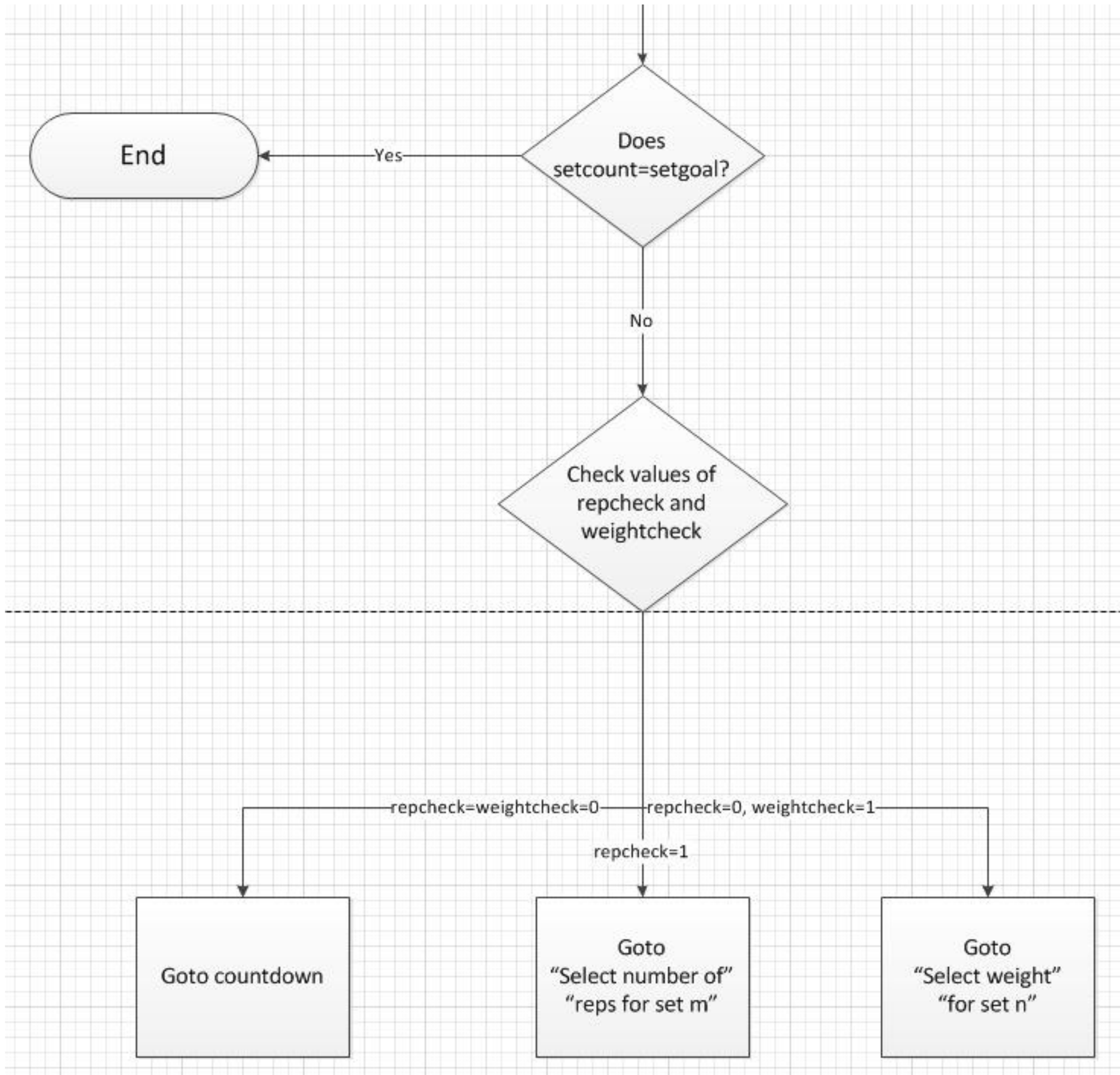


Figure 7: System Flowchart 6



Accelerometer Results:

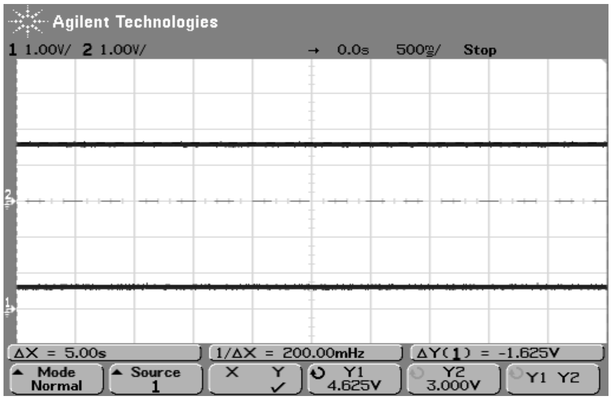


Figure 8a: Zero-rate on X-axis

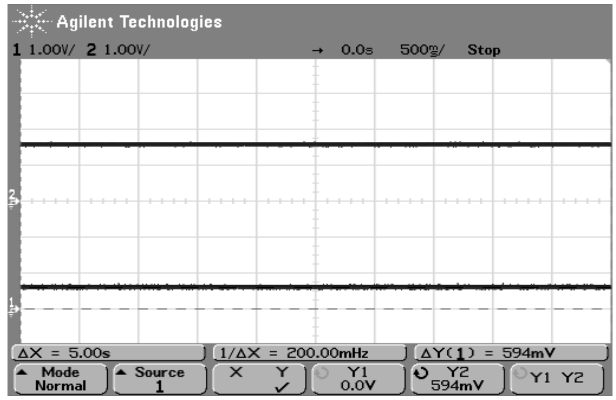


Figure 8b: Zero-rate on Z-axis

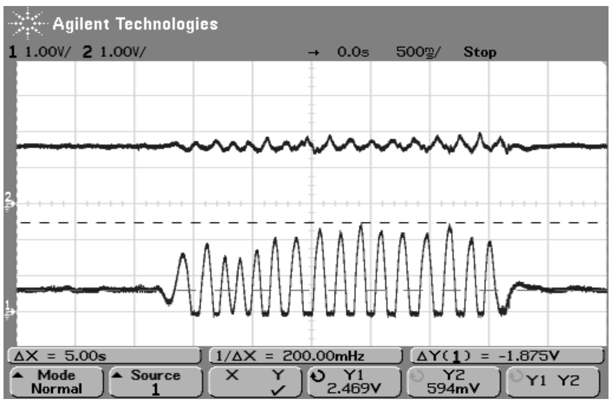


Figure 9a: Z-axis jerk test 1

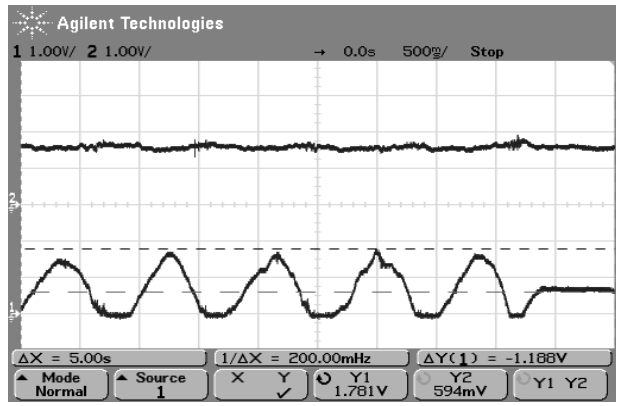


Figure 9b: Z-axis jerk test 2

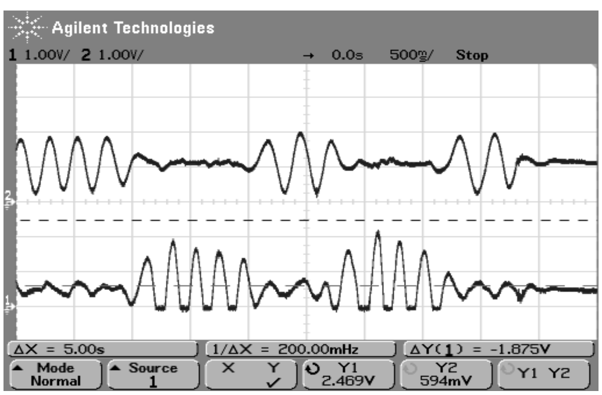


Figure 10a: Tolerance test 1

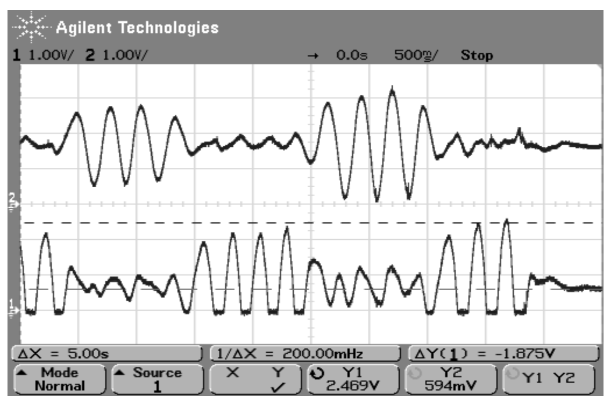


Figure 10b: Tolerance test 2

Gyroscope Results:

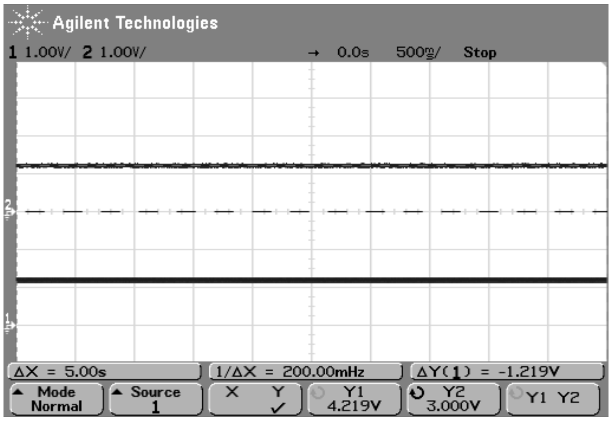


Figure11a: Zero-rate on X-axis

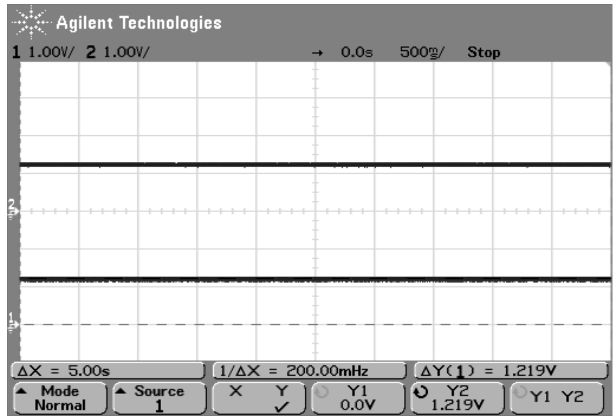


Figure 11b: Zero-rate on Z-axis

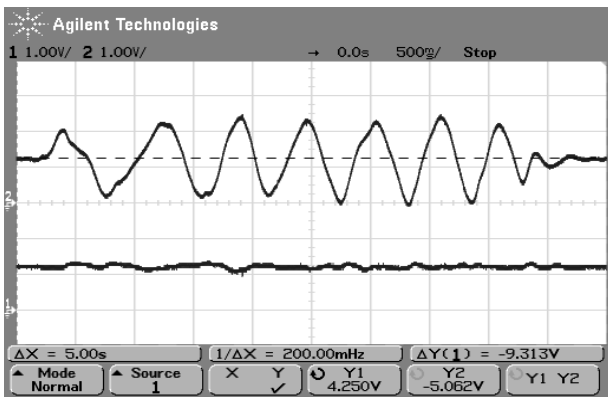


Figure 12a: Performance test (fast) 1

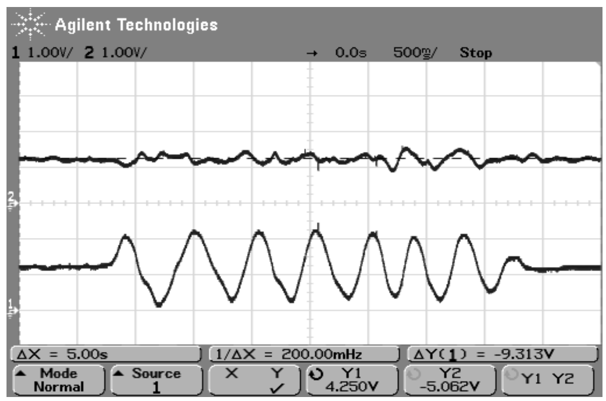


Figure 12b: Performance test (fast) 2

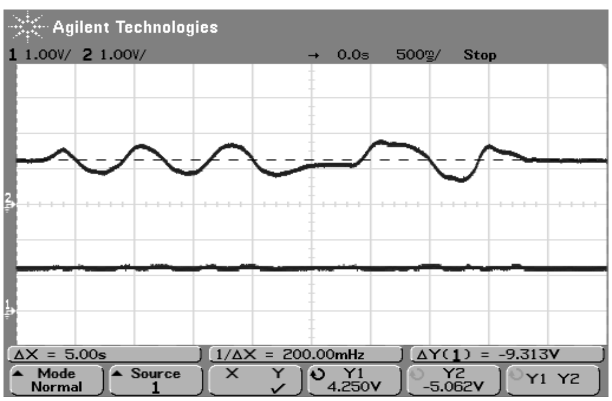


Figure 13a: Performance test (slow) 1

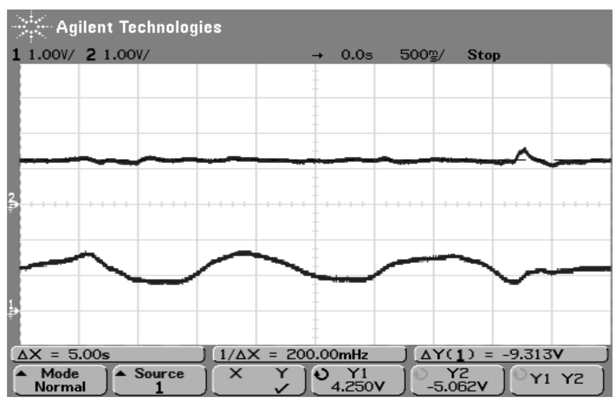


Figure 13b: Performance test (slow) 2

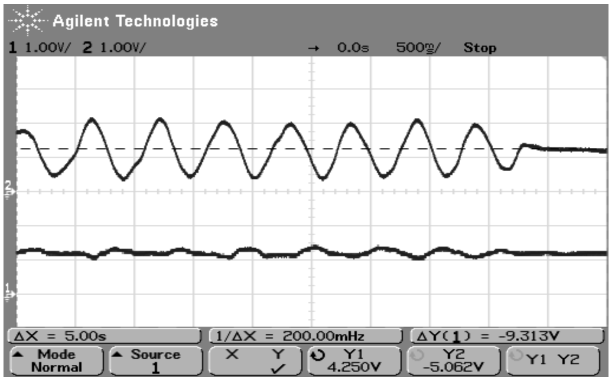


Figure 14a: Tolerance test 1

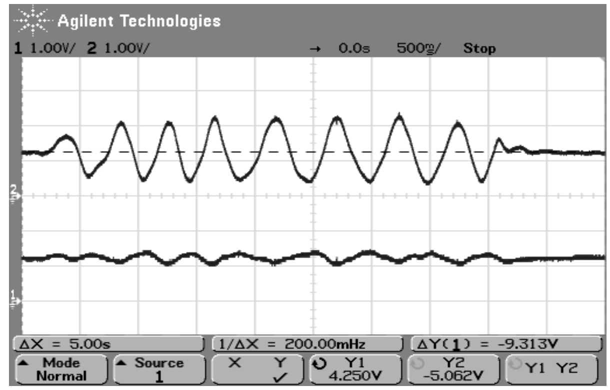


Figure 14b: Tolerance test 2

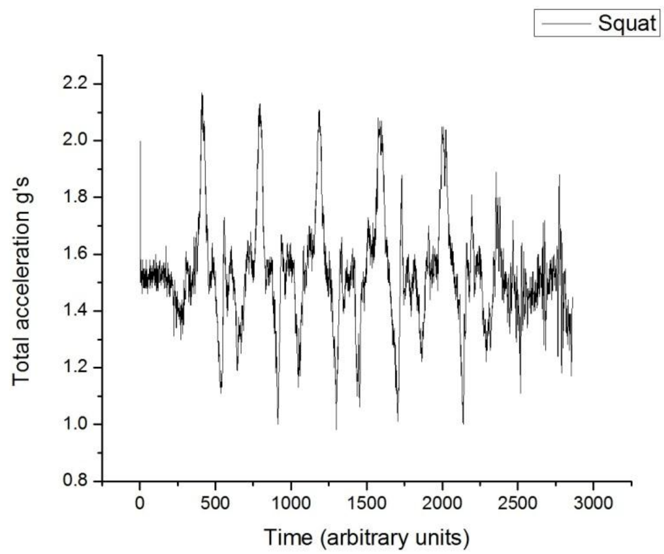


Figure 15: Squats Test

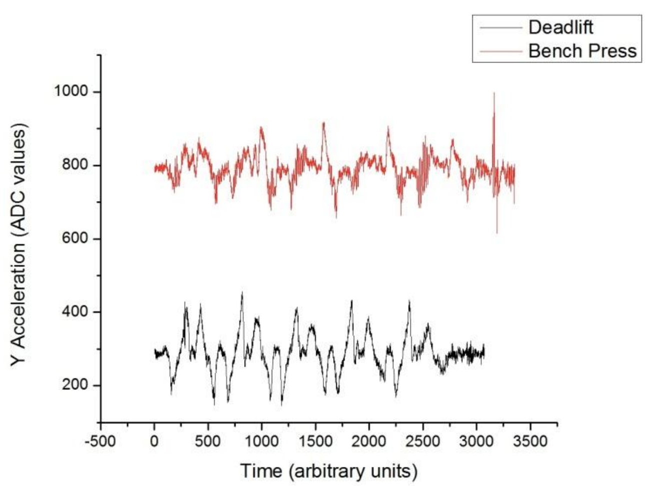


Figure 16: Deadlift and Bench Press Tests

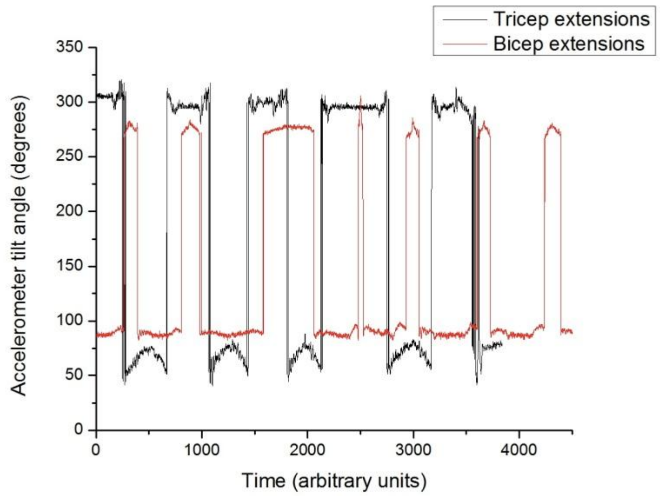


Figure 17: Tricep Extensions and Bicep Extensions Tests

Table 1: Testing and verification

<u>Requirement</u>	<u>Testing & Verification Procedure</u>
Power Supply Unit: <u>1.</u> The battery should be able to supply a nominal voltage of 9V (±1V) for approximately 3 hours. <u>2.</u> The battery will be able to supply a nominal current rating of 143mA and a max. current rating of 321mA for a period of 3 hours. <u>3.</u> The buck converter will be able to accept a nominal voltage of 9V (±1V) (Note: maximum acceptable range is 4.75V to 40V) <u>4.</u> The buck converter will be able to output a voltage of 3.3V (±2.5%) and a maximum current output of 75mA.	**Test:** <u>1.</u> The battery will be tested for 3 hours at maximum load (28Ω resistor based on maximum current and voltage). A DMM will be used to monitor the voltage across the battery and to measure the current drawn. Alternatively, for a more in depth analysis of the battery discharge rate, LabVIEW can be used to obtain a discharge graph for the 3 hours. <u>2.</u> The buck converter will be supplied a 9V input using a bench DC power supply, and a load of 63Ω will be connected across its output. The current through the load will be monitored using a DMM over a period of 3 hours. Testing will be done to obtain the duration of the initial current draw of 74mA by the motor. A 44Ω load will be used to test the buck converter's ability to withstand this high current draw. The duration of this test will depend on the results obtained from the initial current draw test for the motor. **Positive Result:** <u>1.</u> The battery is able to run the course of the test on one complete discharge cycle.

	2. The current through the load stays within a 5% range of 52mA for the nominal current test; the current through the load stays within a 5% range of 74mA for the startup current test. **Negative Result:** 1. The battery is completely depleted before the stipulated time of 3 hours. 2. The current drops significantly over the period of 3 hours for either of the tests.
Accelerometer: 1. The sensor will function with an input ranging from 2.2V to 3.6V (with a nominal input of 3.3V and 0.5mA). 2. It will output a voltage ranging from 0V to 3.3V to indicate acceleration ranging from 1.5g to +1.5g (0g corresponding to a 1.65V output). The scale factor of the output will be 800mV/g (±7.5%). This applies to all 3 axes.	**Test:** 1. A bench DC supply will be used to power the sensor with a 2.2V, 3.3V and 3.6V input. The test (mentioned below) will be performed at these 3 input voltages. At each of the above given voltages the differences in the output of the sensor will be measured. 2. The accelerometer will be jerked from left to right (or top to bottom) over a variety of distances (max = 1.5m) and accelerations (max = 9.8m/s^2). The same test is done on all 3 axes. For testing purposes, the output from the accelerometer will be analyzed using an oscilloscope or using NI-DAQ and LabVIEW. **Positive Result:** The accelerometer gives the readings that correspond to the correct axis on which it was jerked with a scale factor of 800mV/g (±7.5%). Faster jerks correspond to voltage responses closer to ±3.3V and slower jerks to 0V. **Negative Result:** The readings don't correspond to the right axis; relative changes in acceleration don't correspond to the right calibrated voltage; the sensor doesn't function with a scale factor 800mV/g (±7.5%).
Keypad (User-Input): 1. The push-buttons will open and close properly for all 3 scenarios of operation: momentary push, constant push and off. 2. The power switch will close when switched on and open when switched off.	**Test:** 1.The push-buttons will be pressed one at a time and a DMM will be used to check continuity. 2. The power switch will be toggled on and off and a DMM will be used to check continuity. **Positive Result:** The continuity check is positive when the respective buttons are

	pushed. **Negative Result:** The continuity check is negative when the respective buttons are pushed.
Gyroscope: 1. The sensor will function with an input ranging from 2.7V to 3.6V (with a nominal input of 3V). 2. It will output a voltage ranging from 0V to 3.3V to indicate the rate of change of angle ranging from 170 °/s to -170 °/s. The zero-rate level being 1.23V (not ratiometric to supply voltage). The scale factor of the device will be 10mV/ °/s (±10%).	**Test:** 1. A bench DC supply will be used to power the sensor with a 2.7V, 3.3V and 3.6V input. The test (mentioned below) will be performed at these 3 input voltages. At each of the above given voltages the differences in the output of the sensor will be measured. 2. The sensor will be rotated clockwise and counterclockwise along the pitch and yaw axes over a variety of angular rates (max = 170º/s). For testing purposes, the output from the gyroscope will be analyzed using an oscilloscope or using NI-DAQ and LabVIEW. **Positive Result:** The gyroscope gives the readings that correspond to the correct axis on which it was rotated with a scale factor of 10mV/ °/s (±10%); the sensor outputs a zero-rate level of 1.23V. **Negative Result:** The readings don't correspond to the right axis; relative changes in angular rates don't correspond to the expected voltage; the sensor doesn't function with a scale factor of 10mV/ °/s (±10%); the zero-rate level is not 1.23V.
Microcontroller: *1. Requirements related to interfacing with peripherals:* 1.1 The µC functions properly at voltages ranging from 8.25V to 9V with current draw not exceeding 200mA. 1.2 It's able to accept the voltage output from the accelerometer and convert it into g-force with a scale factor of 800mV/g (±7.5%) with a current input not exceeding 40mA. 1.3 It's able to accept the voltage output from	**Test:** *1. Testing related to interfacing with peripherals:* 1.1 A bench DC supply will be used to power the sensor with an 8.25V, 8.5V and 9V input. The test (mentioned below) will be performed at these 3 input voltages. At each of these voltages the differences in the behavior (current and voltage ratings at I/O pins) of the system will be measured. 1.2 The same tests stated for the accelerometer will be performed here, however this time, the sensor will be talking to the µC. A test sketch will be developed for the Arduino to output the

the gyroscope and convert it into an angular rate with a scale factor of 10mV/ °/s (±10%) and with a current input not exceeding 40mA. 1.4 The µC will detect when the push buttons have been turned on and off (including the power button). It distinguishes each button with its respective functionality (power button, navigation buttons, select button and the menu button). 1.5 The µC will output characters onto the display by using only 4 data pins. 1.6 It lights up the LEDs when the program demands it to. 1.7 It will run the vibration motor by sending pulses that last for a second each; it will be able to send out a string of pulses for a maximum of 4 seconds (1 second pulses with 0.5 seconds delay); the maximum current drawn from the I/O pin won't exceed 40mA. *2. Requirements for higher level program:* 2.1 The µC is able to tell if a rep performed is either a correct or an incorrect one by crosschecking the real-time data with the existing database of parameters. 2.2 The vibration motor and LEDs are activated when an incorrect rep is detected (based on data from accelerometer and gyroscope) or when a set is completed. 2.3 The µC interprets the keypad inputs correctly to determine state of program.	data received from the accelerometer in terms of g-force. 1.3 The same tests stated for the gyroscope will be performed here, however this time, the gyroscope would be talking to the µC. A test sketch will be developed for the Arduino to output the data received from the sensor in terms of angular rates measurements. 1.4 A custom Arduino sketch will be developed just to signal to the programmer when the respective keypad buttons are turned on and off. 1.5 A test program will be developed to interface with the display alone; the program will output a variety of strings of characters and the display will be checked to see if the strings match; the display will also be examined to see if it flickers. 1.6 An Arduino sketch will be developed to activate the LEDs at a variety of delays; the outputs on the I/O pins for the LEDs will be monitored using an oscilloscope. 1.7 A custom program will be written for the µC to test the vibration motor. The code will send pulses of high signals to the I/O pin for the vibration motor. The pulses will be 1 second long with delays 0.5 seconds; a maximum of 3 pulses will be sent to the digital output pin. This pin will be probed using an oscilloscope. *2. Testing for the higher level program:* 2.1 Incorrect reps (parameters of which will be determined after performing exercises with the system in place) will be performed and the output pins for the motor and the LEDs will be probed using an oscilloscope and a DMM. 2.2 The same test mentioned above is performed, except this time the vibration motor and LEDs are connected to the µC. 2.3 The user interface will be simulated by using a bench DC power unit to supply the data input ports with high and low signals (low corresponding to ON, since these button inputs will be active-low). A test sketch will be developed that will signal to the programmer

	which button was pressed. **Positive Result:** 1.1 There is a very little change in current and voltage ratings at I/O pins when operating at all 3 given voltages. 1.2 The resulting accelerometer data corresponds exactly to the g-force put in with a scale factor of 800mV/g (±7.5%). 1.3 The resulting gyroscope data corresponds exactly to the angular rate input with a scale factor of 10mV/ °/s (±10%). 1.4 The program correctly processes the keypad inputs (interprets the select button as the select button, interprets the menu button as the menu button and so on); the program is able to tell when each switch is on and when each switch is off. 1.5 The strings on the display match the test program's strings; the display doesn't demonstrate any flickering. 1.6 The output graph on the oscilloscope corresponds exactly with the program's duration of delay between activation signals. 1.7 The output on the oscilloscope corresponds exactly to the pulse width and delay commanded by the program. 2.1 The program recognizes an incorrect rep when an incorrect rep is performed. 2.2 The vibration motor and LEDs are activated when they are supposed to (as demanded by the program). 2.3 The program interprets each push-button correctly (what the button corresponds to: select, menu, etc.) **Negative Result:** 1.1 There is a significant change in current and voltage ratings (±20%) at the I/O pins when operating at the 3 given voltages. 1.2 The resulting accelerometer data doesn't correspond exactly to the g-force. In other words, the absolute error in scale factor is greater 7.5%. 1.3 The resulting gyroscope data doesn't

	correspond exactly to the input angular rate. In other words, the absolute error in scale factor is greater 10%. 1.4 The program incorrectly processes the keypad inputs (doesn't interpret the select button as the select button, doesn't interpret the menu button as the menu button and so on); the program is not able to tell when each switch is on and when each switch is off. 1.5 The strings on the display don't match the test program's strings; the display demonstrates flickering. 1.6 The output graph on the oscilloscope does not correspond with the program's duration of delay between activation signals. 1.7 The output on the oscilloscope does not correspond correctly to the pulse width and delay commanded by the program. 2.1 The program doesn't indicate an incorrect rep when an incorrect rep is performed. 2.2 The vibration motor and LEDs are not activated when they are supposed to (as demanded by the program). 2.3 The program does not interpret each push-button correctly (this includes false positives).
Display: 1. The display will function with an input ranging from 3.1V to 3.5V (with a nominal input of 3.3V). 2. The backlight of the display will light up properly at a nominal voltage of 3.3V (in-built resistor drops it down to 3V). 3. It will display numbers, letters and symbols correctly without any flickering or contrast defects.	**Test:** 1. A bench DC supply will be used to power the display with a 3.1V, 3.3V and 3.5V input. The test (mentioned below) will be performed at these 3 input voltages. At each of the above given voltages the differences in the quality of the display will be noted. 2. The backlight pin inputs (LED±) will be supplied a 3.3V input using a bench DC supply. 3. The display will be tested with the microcontroller connected to it. A custom sketch for the Arduino will make the display output a variety of character strings and the display will be checked to see if the strings match; the display will also be examined for flickering and contrast defects. **Positive Result:** 1. The backlight of the display lights up with

	the 3.3V supply. 2. The characters are displayed clearly on the screen with no flickering or contrast defects; the strings match the strings coded into the microcontroller. **Negative Result:** 1. The backlight of the display does not light up with the 3.3V supply or the backlight is very dim at this voltage. 2. The characters are not displayed clearly on the screen; the display exhibits flickering and there are contrast defects; the strings don't match the strings coded into the microcontroller.
Vibration Motor & LEDs: 1. The vibration motor will function properly at an input voltage of 3.3V (dropped to 1.3V for the motor). It will operate at a nominal current draw of 52mA and an initial start-up current draw of 74mA (duration of which will be determined during initial testing). The functionality of the motor won't deteriorate over a 30-minute period. 2. The LEDs light up with a 2V supply (maximum of 2.5V) and nominal current of 30mA.	**Test:** *Note: An initial test will be conducted to determine the duration of the high initial current draw of the motor (of 74mA).* 1. A bench DC supply will be used to power the motor with a 3.3V input. And a current limitation of 75mA. The test will be conducted for 30 minutes. The performance of the motor will be monitored over this period of time. 2. The LEDs will be supplied a voltage of 2V using a bench DC power supply. A switch will be used to turn it on and off manually. **Positive Result:** 1. The motor vibrates properly for 30 minutes without spinning off. 2. The LEDs light up exactly when the switch is turned on; there is no significant change in brightness over the course of the test. **Negative Result:** 1. The motor does not vibrate properly for 30 minutes; it spun off during the test. 2. The LEDs don't light up exactly when the switch is turned on; there is a significant change in brightness over the course of the test.