Barrel Racer's Calculator

By

Hollie Kucera

Senior Project

ELECTRICAL ENGINEERING DEPARTMENT

California Polytechnic State University

San Luis Obispo

2012

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

Acknowledgements

I would like to thank the following people for making this senior project possible: Professor Braun for being my advisor and guiding me through this entire senior project.

Dino and Michael Cocchi for building the beautiful wooden box to hold the components mounted on the saddle.

Lastly, to my mother and father who stayed with me for hours while testing the device.

Thank you to you all.

Abstract

Barrel racing is a rodeo event in which a horse and rider attempt to complete a clover-leaf pattern around preset barrels in the fastest time. This sport combines the horsemanship skills of a rider and the horse's athletic ability to safely and successfully maneuver around the three barrels (three fifty-five gallon metal drums) placed in an arena.

In barrel racing, the purpose is to make the run as fast as possible. An electric eye and timing system keep track of the time. The time starts when the horse and rider cross in between the electric eyes and ends when the horse and rider complete the pattern and cross back through them. The rider's time depends on how tightly the horse turns the barrels, how quickly the horse accelerates after turning the barrel, and how quickly the horse runs in between the barrels.

In this high speed event the time is recorded to an accuracy of milliseconds. One thousandths of a second can be the difference between first and second place. The horse and rider need to insure that they are running the quickest pattern possible. The quickest pattern requires the horse and rider cover the least amount of ground and accelerate the quickest after turning a barrel.

Presently the barrel racing industry has no technical device that allows the riders to accurately determine the inefficiencies in their run. Current analysis methods include: videotaping each run (from only one angle), placing a set of electric eyes in front of each barrel (the eyes are expensive), and running the pattern over and over again and comparing the times.

The Barrel Racer's Calculator allows the rider to analyze their runs and determine how to improve upon them. It records the distance and the time covered during every run, thus improving upon the methods previously mentioned. The Barrel Racer's Calculator consists of two apparatuses, an electric eye and a sensor mounted to the horse. The electric eye triggers the accelerometer on the horse to start and stop recording data through the use of an Arduino Uno, PIR motion sensor, and a wireless XBee transmitter. The XBee transmits a signal to the sensor box on the horse which contains an Arduino Uno, Accelerometer, SD Data Logger, and a wireless XBee receiver.

I. Introduction

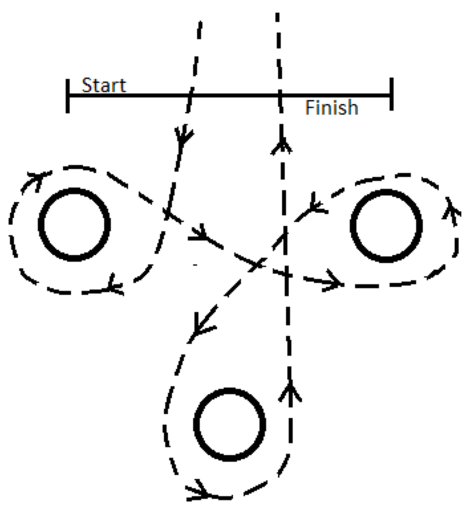

Figure 1: Clover-leaf Barrel Racing Pattern

In barrel racing, a horse and rider complete a clover-leaf pattern (figure 1) around three barrels as fast as possible, while clocked by a timer module and electric eye[5]. The time begins when the horse and rider cross the start line, and ends when the barrel pattern has been successfully executed. The rider's time depends on how tightly the horse turns the barrels, how fast the horse accelerates after turning each barrel, and how quickly the horse runs in between the barrels and across the finish line[4].

In this high speed event the time is recorded to an accuracy of milliseconds [6][2]. One thousandths of a second can be the difference between first and second place. The horse and rider need to insure that they run the quickest pattern possible in order to win. The quickest pattern requires the horse and rider cover the least amount of ground and accelerate the quickest after turning a barrel[8].

The barrel racing industry has no technical device that allows the riders to accurately determine where they are losing time during their run. Current analysis methods include: videotaping each run (from only one angle), placing a set of electric eyes in front of each barrel (the eyes are expensive), and running the pattern multiple time then comparing the runs. The Barrel Racer's Calculator allows the rider to analyze their runs and determine how to improve upon them. It records the distance and the time covered during every run. Since no device in the barrel racing industry calculates the distance traveled by the horse during a run this device can provide a great service to barrel racers.

II. Background

Barrel Racing has two methods of improving a run. These methods include timing the runs multiple times and videotaping the runs. Both of these methods are not very accurate, because a run depends on more than just the time and with a video we have human error.

Issues with the Timing Method

This timing method is a trial and error method. The rider runs the horse over and over again, changing little things, such as how they approach the barrels and how much they turn the horses head, trying to make the run quicker. The issues with this include repeated runs that tire the horse out. A tired horse cannot run as fast, therefore making the later trials useless. Also the more that a barrel horse runs a pattern they become 'sour' and will not want to run the pattern anymore[8].

Issues with the Videotaping Method

Videotaping the runs provides the rider with more useful information than the previous method. From the video the position of the horse and rider turning the barrels, and most speed changes can be determined. The videos are only taken from one angle, meaning that during some points of the run the barrel blocks the horse and rider from the camera. Barrels in front of the camera make viewing the exact pattern impossible.

A more expensive method of video tapping includes the use of multiple cameras from different angles. Even with multiple cameras, this method lacks efficiency.

Barrel Racer's Calculator

The timing and videotaping methods do not look at where your horse's footsteps lie in relation to the barrels. You can see the pattern visually but not the actual pattern that was run. The barrel racer's calculator will give the rider this capability. Both methods can be done at home but the Barrel Racer's Calculator provides the best method for seeing what the rider and horse actually do in a race situation.

III. Requirements and Specifications

1. The device has a sturdy and durable design, capable of withstanding the accidental impacts that would occur when attached to a large running horse

    I. A small, lightweight, chassis must contain the device for durability and safety.

3. The device does not cause shock to the horse or user.

    I. The device comes in contact with the horse's skin, so it must not cause any harm.

4. The device has a minimum of 5 hours of battery life.

    I. This allows the user to make multiple runs without having to charge the device.

5. The placement of this device on the horse cannot in anyway change the horse's performance

    I. The device, including the chassis, cannot exceed 3lbs., so that the horse is not slowed down by the addition of the device.

6. The device should meet all of the design stipulations stated below:

    I. The device must start and stop when the horse crosses a motion sensor.

    II. The device must operate at 2gs.

        a. This g-force was calculated assuming that the horse slows down to 12 MPH while turning the barrel.

IV. Design

The Barrel Racer's calculator consists of an Arduino Uno with an accelerometer communicating wirelessly with a PIR motion sensor through the use of XBee's. Other topologies considered included the use of a wireless receiver to detect the signal from the existing electronic eye to the scoreboard. This design was deemed impossible when an employee from FarmTech would not disclose the current message structure[6]. To meet the required size and weight constraints, the Arduino seems to be the best possible route. The Arduino is more than capable of processing and storing the data with the use of its ATmega328 processor. The data will be taken in through digital pin connected to the XBee receiver connected to the motion sensor. Once motion has been detected, the accelerometer will start recording data. I decided on a three axis accelerometer in order to accurately store the path of the horse. Once the data has been taken, the user can analyze the data by removing an SD card with the data of the run stored on it. The SD card will have the data stored as a comma separated value file (CSV) that can be opened with Microsoft excel. Each column will represent a different value of the accelerometer, x, y, z and time. Plugging these values into a few physics equations will result in an average velocity and distance measurement. The data can also be plotted for further analysis.

<u>Overall system block diagram</u>

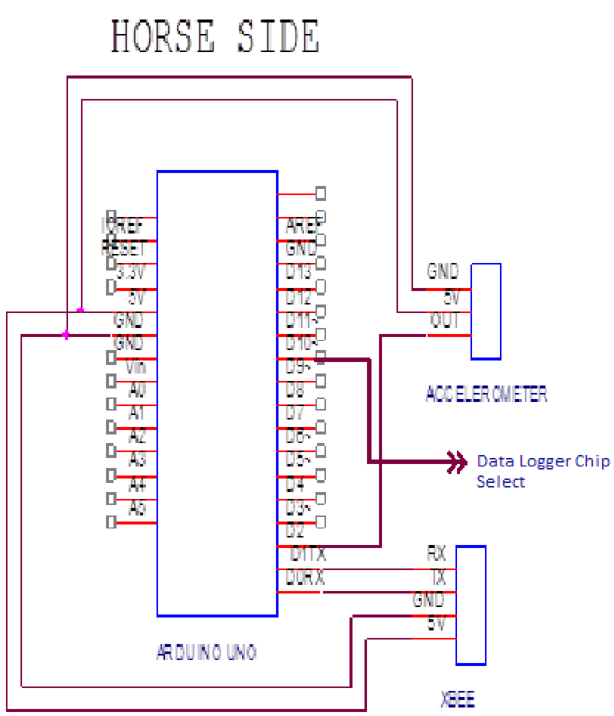

Figure 2: overall system block diagram

Arduino Uno

The Arduino Uno (figure 3) contains a microcontroller board based on the Atmega328. It has 14 digital input/output pins, 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, and ICSP header, and a reset button. Operating voltage is 5V, clock speed is 16 MHz, and it has a flash memory of 32 KB. The Arduino Uno power jack allows the system to use a 9V battery for power. Schematic displayed in appendix B.

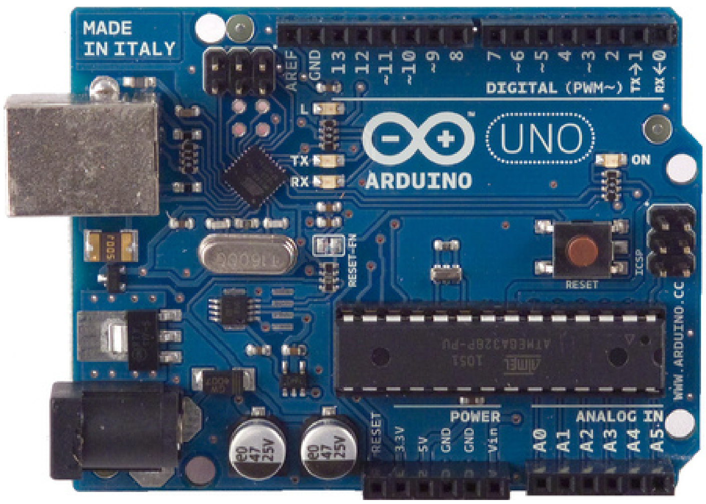

Figure 3: Arduino Uno

Power Supply

To power the Arduino Uno a 9 volt battery (figure 4) will be used. The battery will be connected to the board through the Arduino Uno's power jack.

Figure 4: 9V battery with container and power jack connector

PIR Motion Sensor

The PIR motion sensor (figure 5) detects motion by sensing changes in infrared levels emitted by surrounding objects. When motion is detected, the PIR sensor outputs a high signal on its output pin, which will be read by an Arduino Uno, then transmitted to a second Arduino Uno, which then triggers an accelerometer to start. The PIR sensor is operational at a temperature range of 0 →50˚C and has a power requirements of 3 →6 VDC. The Arduino Uno's operating voltage is 5V, so the two will be compatible. Schematic displayed in appendix B.

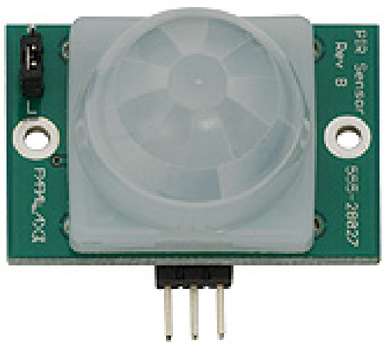

Figure 5: PIR Motion Sensor

XBee Wireless Kit

The XBee wireless kit (figure 6) is able to interface to a 5V supply or device up to 100ft away (outdoor range, indoor range is 300ft). The XBee is operational at a temperature range of -40˚C →70˚C and its power requirement is between 2.8V → 3.4VDC. It is compatible with 5V microcontrollers, so it will work with the Arduino Uno. Schematic displayed in appendix B.

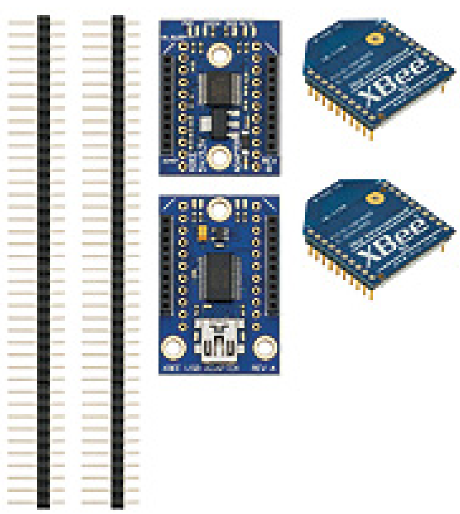

Figure 6: XBee Wireless Kit

ADXL335 Accelerometer

The ADXL335 is a 3-axis accelerometer (figure 7) with signal conditioned voltage outputs. It measures acceleration with a minimum full-scale range of +/- 3g [11]. The ADXL335 can measure the static acceleration of the gravity in tilt sensing applications, as well as dynamic acceleration resulting from motion, shock, or vibration. It is operational between 1.8V → 3.6V. The Arduino Uno has a voltage rail of 3.3V, so these two will be compatible. Schematic displayed in appendix B.

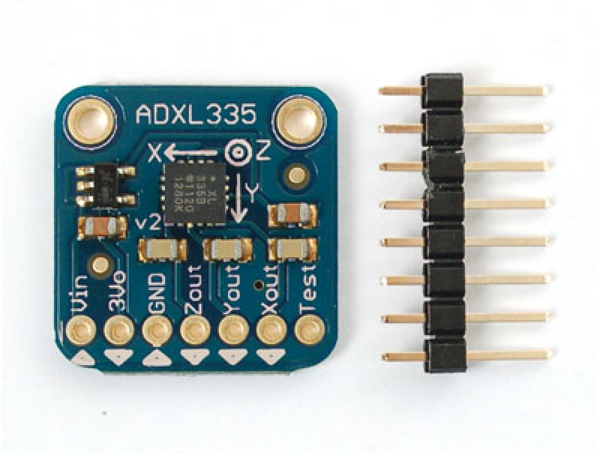

Figure 7: ADXL335 Accelerometer

Data logging shield for Arduino

The data logging shield (see figure 8) will store the values taken from the accelerometer on the SD card. The values will be formatted as a comma separated value file. The CSV file can be opened with Microsoft Excel to compute the average velocity and total distance of the runs. Schematic displayed in appendix B.

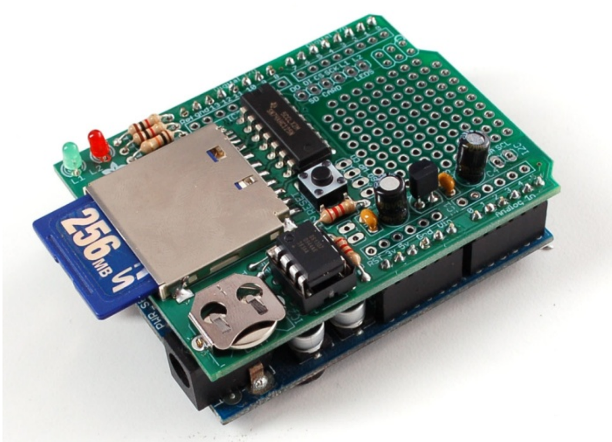

Figure 8: Data Logger for Arduino Uno

Packaging for electric eye

The electric eye with consist of an Arduino Uno, PIR motion sensor, XBee wireless transmitter, a 9V battery, and a 12in tripod. The Arduino Uno, PIR sensor, and battery will all be placed inside of a small plastic box with a hole in it for the PIR sensor to look out. The PIR sensor will have a ½ in piece of pipe placed around the lens, to focus the infrared detection beam to a straight line [7] [9]. This small box will then be attached on top of the tripod (see figure 9).

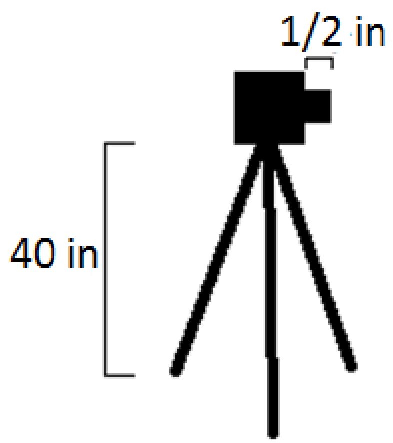

Figure 9: Tripod with electric eye packaging

Packaging for the device on the horse

The device that will be on the horse includes an Arduino Uno, XBee wireless receiver, ADXL335 accelerometer, and a 9V battery. All of these will be placed inside a small plastic box and secured to the gullet (figure 10) of the saddle by two straps (see figure 11).

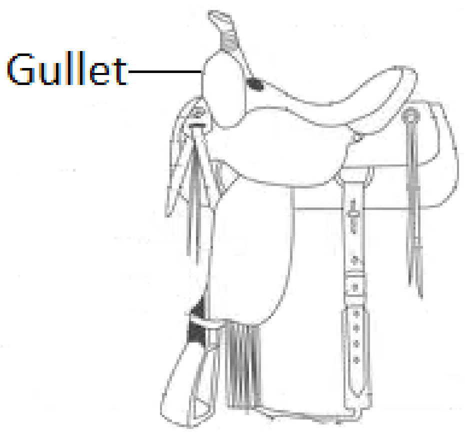

Figure 10: Gullet of a western saddle

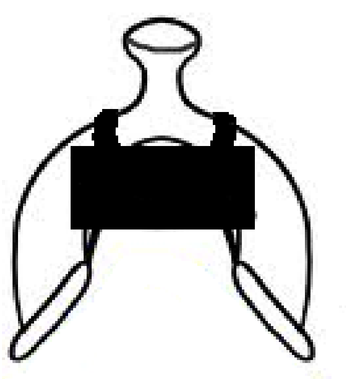

Figure 11: Packaging attached to gullet

V. Test Plans

1. The electric eye device must be tested.

    1.1 The PIR sensor connects to an Arduino Uno.

        1.1.1 To ensure that the PIR sensor works properly something will be moved in front of it. An LED on the Arduino board will turn on when it detects motion.

        A message will be sent to the serial port once motion is detected. The output will be displayed through the Arduino serial monitor.

        1.1.2 An enclosure will be placed around the lens of the sensor (to narrow its range) and then tested again. If the light on the sensor turns on only when an object is moved in front of the pipe, the test was successful.

        1.1.3 The sensor will be tested at multiple distances to see how far it reaches.

    1.2 The XBee transmitter will be placed on the Arduino Uno with the PIR sensor on it, and the XBee receiver will be placed on the second Arduino Uno.

        1.2.1 The PIR sensor will be triggered to see if the XBee transmitter is communicating with the XBee receiver.

        The message will be read to ensure proper operation.

        The delay time will be measured to ensure that the horse does not trip the sensor twice.

        The sensor will be tripped again to ensure that the stop message is sent.

        The message will be read to ensure proper operation.

2. The Receiving device must be tested.

    2.1 The Arduino Uno, which already has the XBee receiver on it, should be examined when the PIR sensor is set off to ensure that it is receiving information from the electric eye device. This will be done by programming the Arduino Uno have an LED turn on when the XBee receives a high signal from the transmitter.

    2.2 The ADXL335 accelerometer will be placed on the Arduino Uno.

        2.2.1 The Arduino Uno during this time will be connected to a computer through a USB connector.

2.2.2 The Arduino Uno and accelerometer will be moved around to see if the accelerometer is functioning properly.

2.2.2.1 If the accelerometer is functioning properly corresponding plots of its movement will show up on the serial monitor.

2.2.3 The accelerometer will be tested over long distances and at different speeds.

The data logger replaces the USB cable and the output saved on the card.

The data will be analyzed by excel and plotted.

3. The receiving device will then be placed in the enclosure on the horse and the devices will be tested.

The enclosure mates with the saddle and then jerked to ensure proper mounting.

Once the accelerometer and Arduino Uno are mounted in the box the rider will ask the horse to trot.

While trotting, the rider checks to make sure the device does not interfere with their ability to grab the horn.

The rider watches the front of the box to ensure it does not hit the horse in the neck.

The rider will let the horse takeoff to test the structural integrity of the box.

4. The XBees will be tested once the Arduino Uno is in the enclosure.

The horse will walk past the motion sensor to trigger the device.

The horse will turn around and walk through the sensor again.

With the SD card removed, Excel will plot the data.

5. A complete run will be accomplished.

VI. Development and Construction

A few designs were considered for the BRC. First a Garmin forerunner 110 gps enabled watch [1]. Using a built in GPS the watch can measure distance and the pace of the rider [3]. The plan was to mount the watch to the saddle and have it record the runs. It records distance and can tell the rider if their pace was to slow. A few problems arose that made this implementation path impossible. First the device needs to start when the IR beam of the electronic eye is tripped. This required that the device be taken apart and implementation of a circuit that would know when the eye is tripped. The circuit would start and stop the watch [2]. Without accurate documentation of the watch, this seems near impossible. The watch also does not save the data to a media accessible by a computer. In order for a barrel racer to improve, they need to see their path and correct it. This was deemed impossible with the Garmin.

After the Garmin, I considered the Arduino Uno. I have seen other students use them in CPE 329. After a little research, the Arduino Uno looked like the microcontroller board for the project. With a wide array of sensors and open source code, the Arduino Uno was chosen for the project [10]. Adafruit industries make a wide array of sensors and shields for the Arduino Uno. The triple axis accelerometer ADXL335 was chosen so that the acceleration could be measured forwards, laterally and vertically at a max bandwidth of 50 Hz [11]. These accelerations can be used to create velocity vectors. With velocity and time the distance of the horse can be plotted [12]. To get a straight line distance, the Pythagorean Theorem must be used.

After talking to a technician at Polaris, they were unwilling to disclose the message structure of the signal that they were sending to start and stop their timers [6]. To imitate this signal, a tripod with an IR motion sensor mounted on it, could start and stop the BRC. The sensor “relies on temperature differences between the target and background” to detect motion [9]. Parallax makes a PIR motion senor in the correct package for the project that can reliably detect motion up to 30 feet away. But it seemed too sensitive when it detected motion with almost a 180 degree field of view. This is unacceptable; the sensor needs to activate by the movement of the horse and not its surroundings. Focusing the beam with a cone decreased the number of false readings [7].

The signal from the motion sensor needs to trigger the accelerometer to start taking data. To send the signal, a pair of XBees was used to talk between the boards. The XBees send the data over a specified frequency. They are normally connected to the TX and RX lines of the board. The Arduino Uno however uses the TX and RX lines of the board when programming the flash [10]. This results in an error and the inability of the user to program the board. To work around this a software serial can be used to send the data. By setting on pin as an output and another as an input at the same baud rate, the pins can take the place of the TX and RX pins. To make sure that the signal is not send on a widely used frequency, the frequency and permissions of the XBee were changed.

The data from the accelerometer needs to be easily analyzed and accessible. Attaching a USB cable after every run seemed undesirable. After some research, the Adafruit data logger seemed like the shield for the job. The SD card would capture all of the data for the run. The size of the SD card would allow the user to use the card for multiple runs. After the run the SD card can be read by a computer and plotted in Excel. This would allow the user to look at different points in their run to determine where time was gained and lost.

Attaching the device to the horse seemed like a problem at first. The vibration of the horse could damage the device and make the readings indecipherable. The enclosure must also not interfere with the horse's ability to run and the rider's ability to perform the run. These constraints made all of the premade boxes at multiple store unusable. To remedy this problem, I constructed a box that when strapped to the horse could hold the electronics and not interfere with the run.

VII. Conclusion

This prototype was successfully able to meet most of the requirements established at the beginning of the project and therefore can be considered a success. The accelerometer on the horse could be started and stopped when an IR motion sensor was tripped. Velocity and distance can be calculated using the acceleration vectors, but the end result accumulates error due to the inability to have an infinite amount of sample points.

Even though the prototype shows a proof of concept the Barrel Racer's Calculator has many modifications that need to be implemented for later designs. Further modifications would make this device more useful to its user as well as more profitable to the manufacturer. The device currently cost $230 in parts alone, which is near the marketable range of the device. To make the device profitable enough for mass production other less costly components are required. One suggestion would be, a sensor change from an accelerometer to an inertial measurement unit. Doing this before production would make the device more useful to the consumer.

References

1. Garmin. (2011, October 17) Forerunner 110. [Online]. Available: https://buy.garmin.com/shop/shop.do?cID=142&pID=63511

2. Y. Kazuhiro and N. Michiaki. (2011, October 17) Electronic Stop Watch, United States Patent 4104861. [Online]. Available: http://www.freepatentsonline.com/4104861.html

3. K. Matsumoto and M. Terashima. (2011, October 17) GPS Receiver, United States Patent 7649494. [Online]. Available: http://www.freepatentsonline.com/7649494.html

4. J. G. Henderson. (2011, October 17) Rodeo: The Basics of Barrel Racing. [Online]. Available: http://www.helium.com/items/1910019-barrel-racing-rodeo-horse-rider-women-cloverleaf-pattern-timed

5. Dr. W. (2011, October 17) Barrel Racing Pattern. [Online]. Available: http://www.barrelracingclinic.com/BarrelPattern.htm

6. Polaris. (2011, October 17) Polaris Electronic Timer. [Online]. Available: http://www.tackstop.com/polaris.htm

7. J. J. García, J. Ureña, M. Mazo, C. Losada, Á. Hernández, and J. A. j, "Reliability Improvement of Obstacle Detection in an IR Barrier ," *IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference on* , 2006.

8. S. Camarillo, D. Arnold, R. Witte, *The A.R.T. of Barrel Racing*, Edition of book, Colorado Springs: Western Horseman, 2007.

9. 1st Initial. Navmetocprodev det, “Atmospheric Effects on EO Sensors and Systems," Naval Meteorology and Oceanography Professional Development Detachment Atlantic, Vol. , no. , pp. 8, June 2005. [PDF]. : www.deltagearinc.com/OpticsFacts/EO.pdf. [Accessed 3/16/12]

10. T. Tom igoe, "Arduino Uno,”, Vol. , no. R3, pp. all, September 18, 2010.[Web page]. :http://arduino.cc/en/Main/ArduinoBoardUno. [Accessed 3/16/12]

11. Analog devices, “Small, Low Power, 3-Axis ±3 g Accelerometer ADXL335," Rev. B, 2010. [PDF]. http://www.analog.com/static/imported-files/data_sheets/ADXL335.pdf. [Accessed 3/16/12]

12. Starlino, “A guide to using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications," , December 29, 2009.[Web page]. : http://www.starlino.com/imu_guide.html. [Accessed 3/16/12]

Appendixes A: Senior Project Analysis

Summary of Functional Requirements

- Records acceleration of each run
- Starts and stops by crossing an electric eye
- Records multiple runs for the user to compare

Primary Constraints

Two limiting factors that impacted my approach was the cost of the components and calculating distance and velocity from acceleration. I funded the project on my own, so I had to be choosey when it came to picking out components. Attempting to calculate distance and velocity from the data produced by the accelerometer proved to be near impossible. The results from the calculations were extremely inaccurate and eventually were removed from the project. This unfortunately makes the device more difficult for the owner to use.

Economic

The Barrel Racer's Calculator has the potential to pay for itself in the future. The overall cost of the device will be around $300. Winning first place at a barrel race pays out anywhere from $100 → $5,000. A person could easily win back the cost of the product in three or less races.

Parts

Table 1: Estimated and Actual Cost of Parts

Component(s)	Est. Cost	Actual Cost	Est. Quantity	Actual Cost	Est. Total Cost	Actual Total Cost
Microcontroller(Arduino Uno)	29.95	29.95	2	2	59.90	59.90
Power Supply (9V battery & Holder)	3.95	3.95	2	2	7.90	7.90
PIR Motion Sensor	9.99	9.99	1	1	9.99	9.99
XBee Wireless Kit	99.99	99.99	1	1	99.99	99.99
Accelerometer (ADXL335)	14.95	14.95	1	1	14.95	14.95
Data Logger	19.50	19.50	1	1	19.50	19.50
Packaging	10	7.50	2	2	20	15
					$232.23	**$227.23**

Labor

Table 2: Estimated and Actual Labor

Activity	Est. Hours	Actual Hours
Initial Design	25	30
Prototype Build	12	15
Prototype Test	10	6
Result Analysis	5	5
Designs Updates	8	10
Final Build	8	4
Acceptance Test	12	10
System Integration	20	25
Total Hours	**100**	**105**

Manufactured or Real Capital: Price charged for final product: ~ $300.00. This price takes into account the cost of the parts and a portion of the labor cost. The labor required to create more after the first one is built will decrease drastically.

Natural Capital: This product will not benefit the natural environment.

Cost benefits for this project accrue for the consumer after having used the device. This device will allow the racer to improve their times, therefore causing them to place higher at races and win more money.

The cost to build the overall device will be: Parts (~$227.23), Labor (~105 hrs. at $75 per hr. = $7,875). I will be financing this project myself.

The first prototype will be ready in June of 2012. The products life time is hoped to be around 3 years, but testing of the prototype will determine this number more accurately. The maintenance cost will vary depending on what is damaged. The operation costs are limited to only the price of charging it.

Gantt Charts: Estimated and Actual

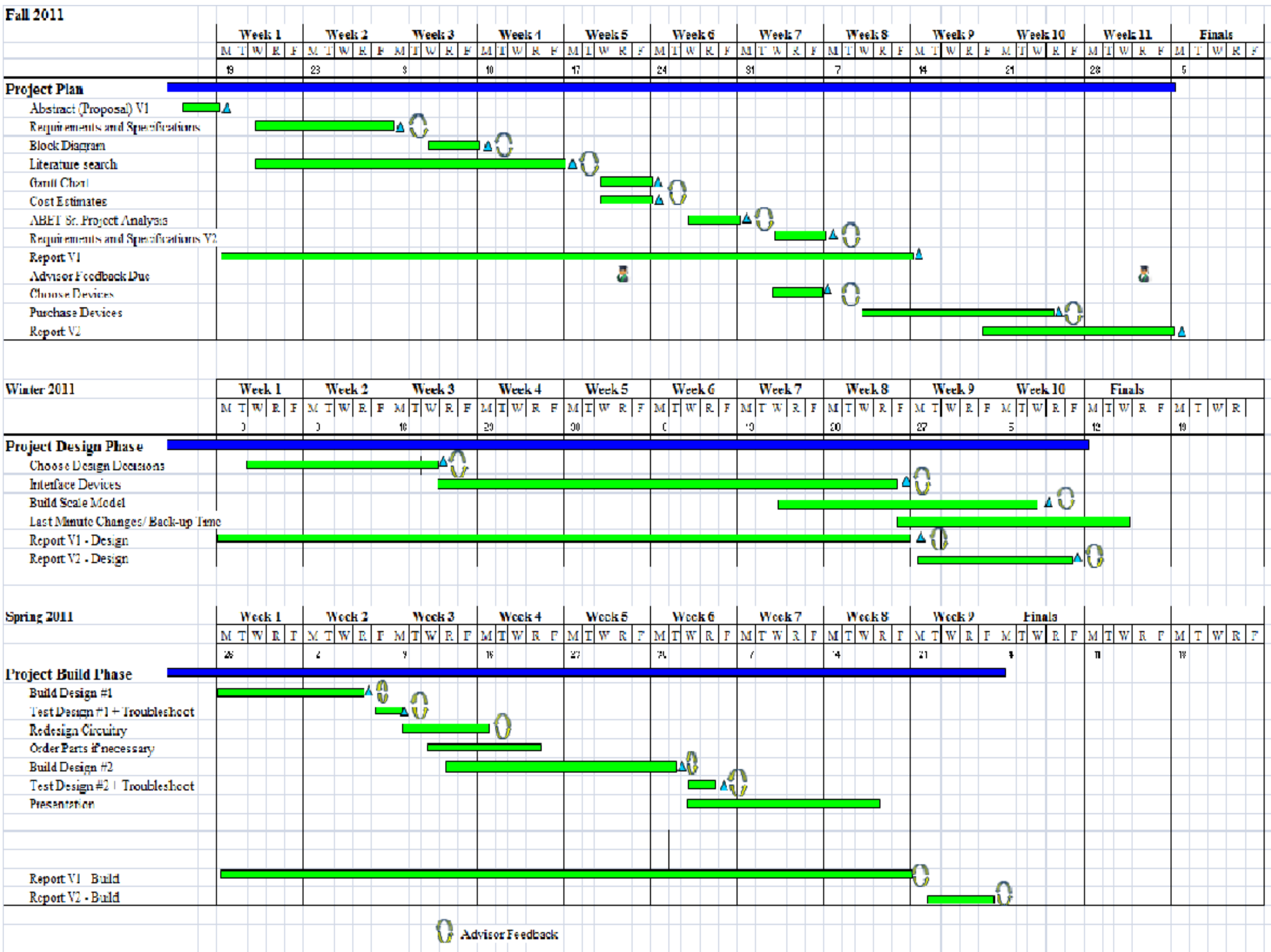

Figure 12: Estimated Gantt Chart

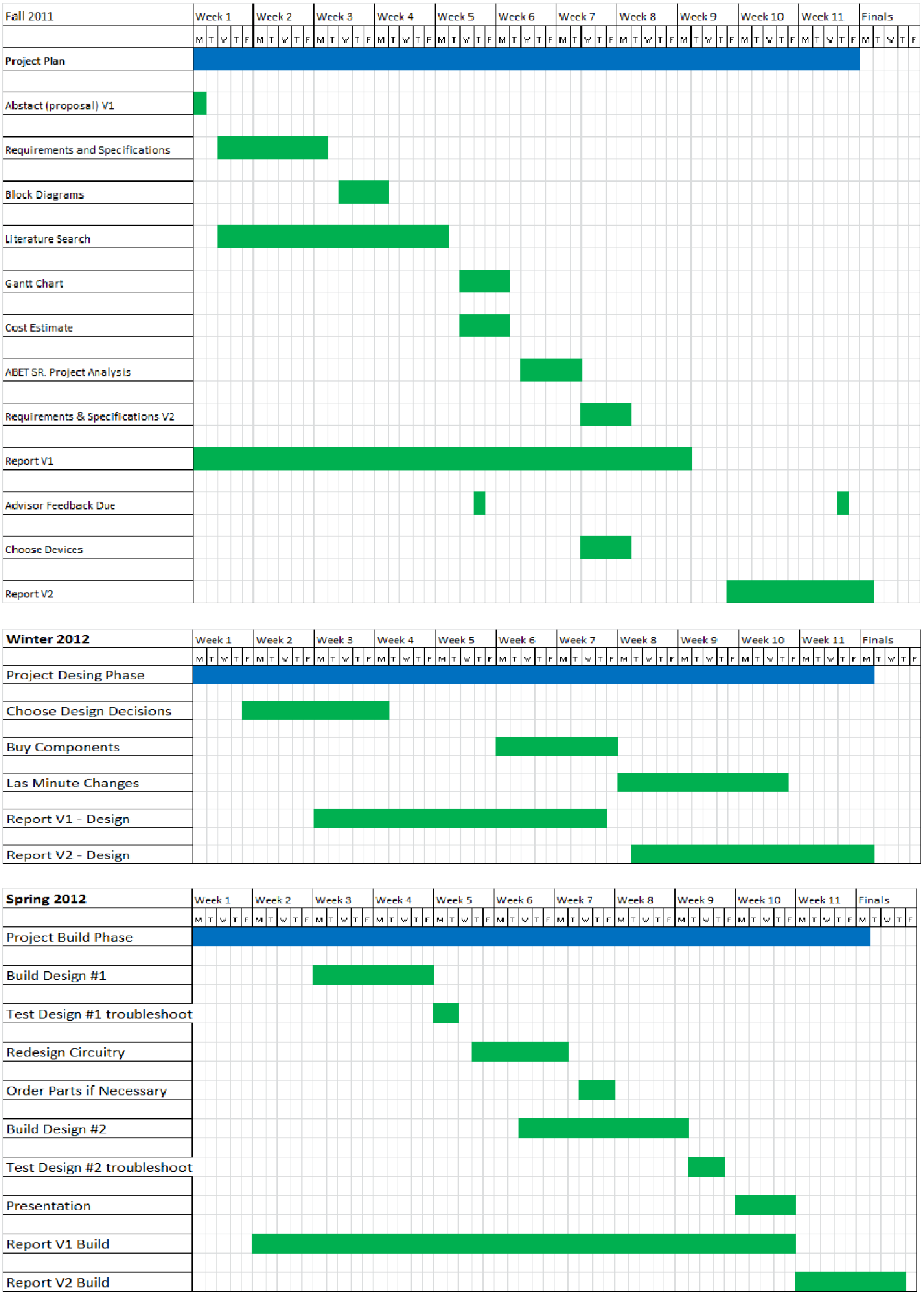

Figure 13: Actual Gantt Chart

After the prototype is completed it is hoped that the possibility to sell it to the public will become a reality.

If Manufactured on a Commercial Basis:

Number sold each year:

Year 1: ~ 100
Year 2: ~ 300
Year 3: ~1000

Manufacturing cost per device: ~$250

Purchase price per device: ~$300

Profit per year:

Year 1: ~$5,000
Year 2: ~ $15,000
Year 3: ~$50,000

Estimated cost to operate device: Cost to replace batteries is ~$4.10.

Environmental

The environmental impact of this project being created will be in the devices used to complete it. The devices are made out of plastics and metals. The building of the product will be marginally harmful to the environment, but after the lifetime of the device it will be able to be recycled.

Manufacturability

Calculating distance and velocity accurately from the accelerometer date will prove time consuming and difficult. Also the box on the horse needs to decrease in size meaning that new, more compact, components are required.

Sustainability

With any electronic device there will be challenges to ensure that it continues working properly. Proper maintenance will be required by the user. The device should not be dropped or abused in any way. The devices requires the use of plastic and metals to be made and also requires electricity to be charged. These are all precious resources.

Ethical

This device will help improve on the speed of a run. If the user runs the animal too many times in a row to try and improve on their last run, damage will be caused to the animal. The animal could be injured or be hurt mentally by excessive runs.

Health and Safety

Parts of this device are small and could be ingested by the animal causing serious harm or death. If device is damaged and then used, shocking could occur and cause harm to the animal.

Social and Political

This device could potentially put other similar devices out of the market, thus harming the people involved with making and selling that device.

Many people disagree with having any electronic device on an animal, because of the risk of shocking the animal.

This project will impact the people involved in creating it, the users, and the animals it is used on.

The direct stake holders are the people who created it and the people using the device. The indirect stakeholders are the people/companies who the components are purchased form in order to make the device.

The project will benefit the stakeholders financially, but it may harm the people who create similar products but decreasing their sales.

Development

Working in groups to determine the requirements of the product has been extremely successful.

Appendix B: Schematics

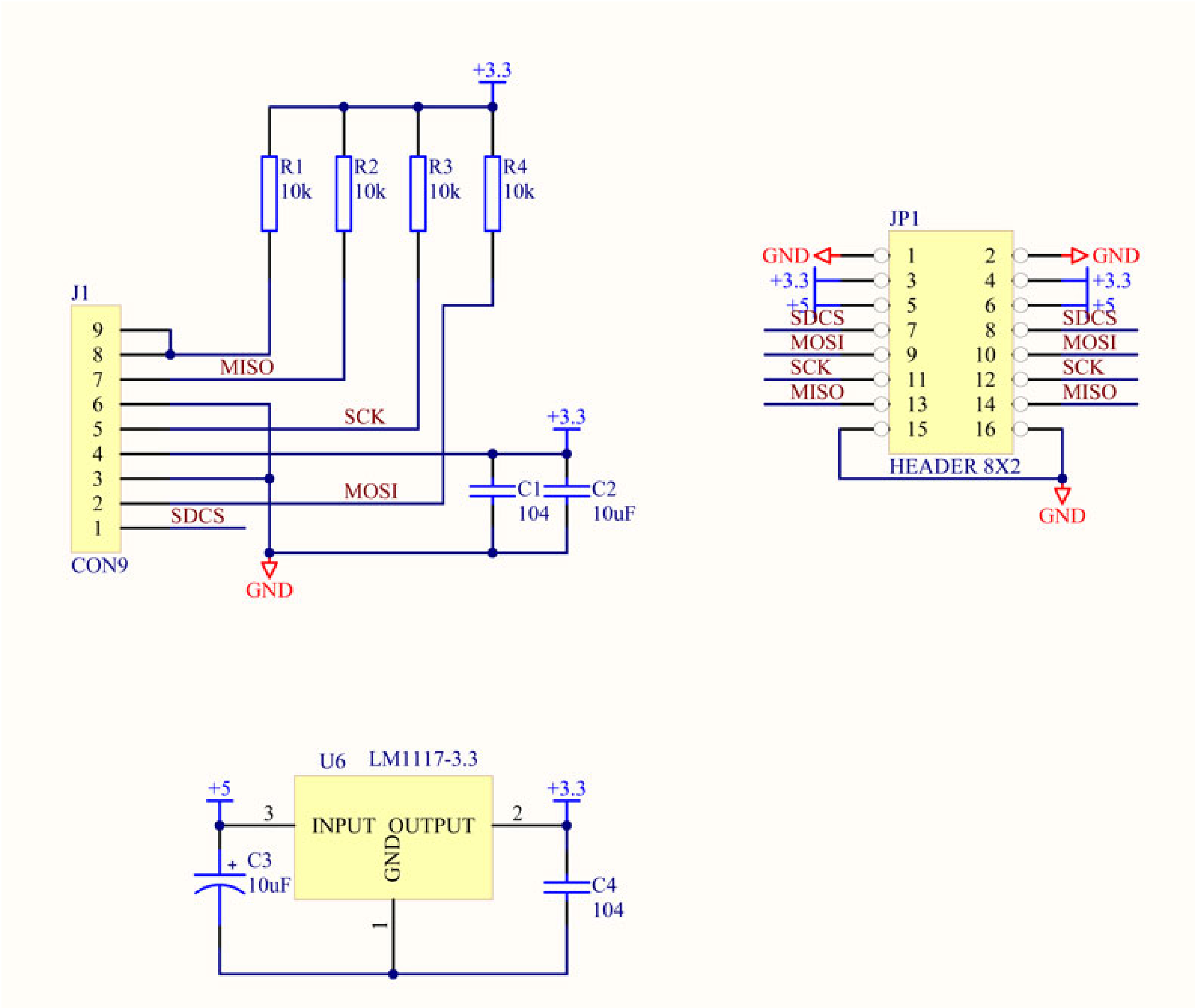

Figure 14: SD data logger

Schematic found at http://www.turtlesarehere.com/html/data_logger.html

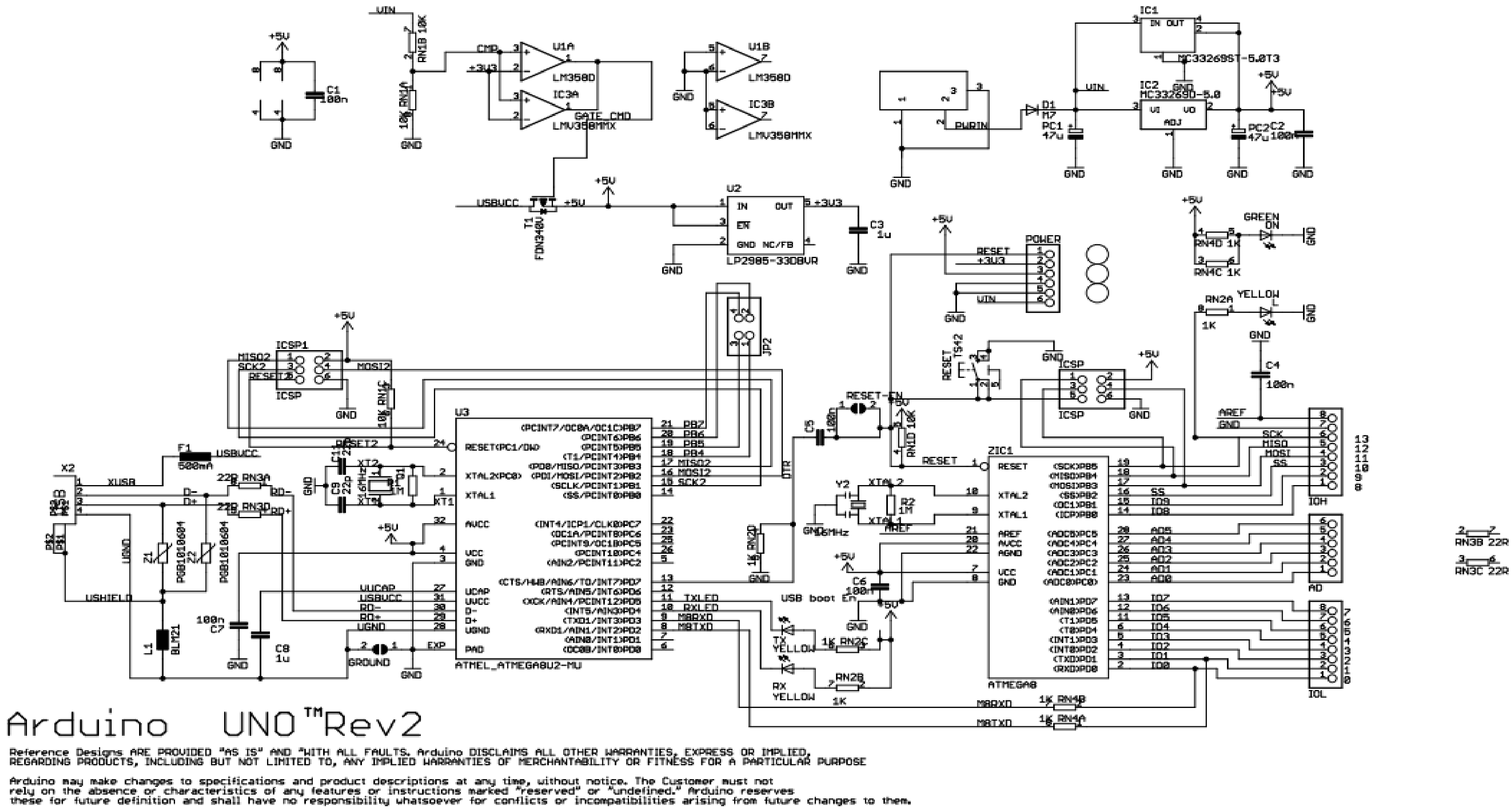

Figure 15: Arduino Uno Schematic

Schematic found at http://arduino.cc/en/uploads/Main/arduino-uno-rev2-schematic.pdf

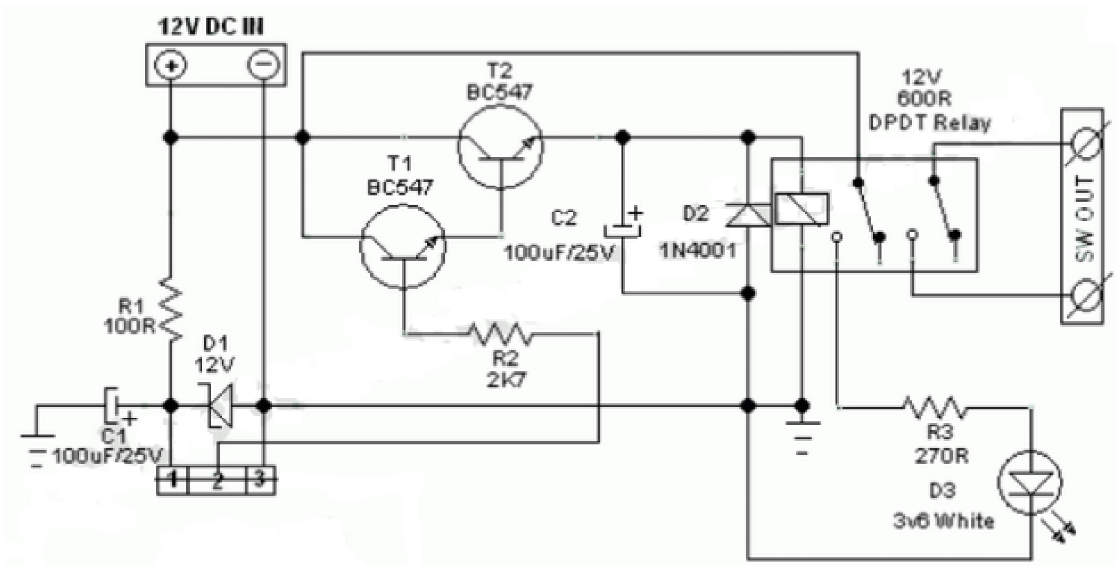

Figure 16: PIR Motion Sensor Schematic

Schematic found at http://electroschematics.com/6039/security-light-switch-with-pir-sensor/

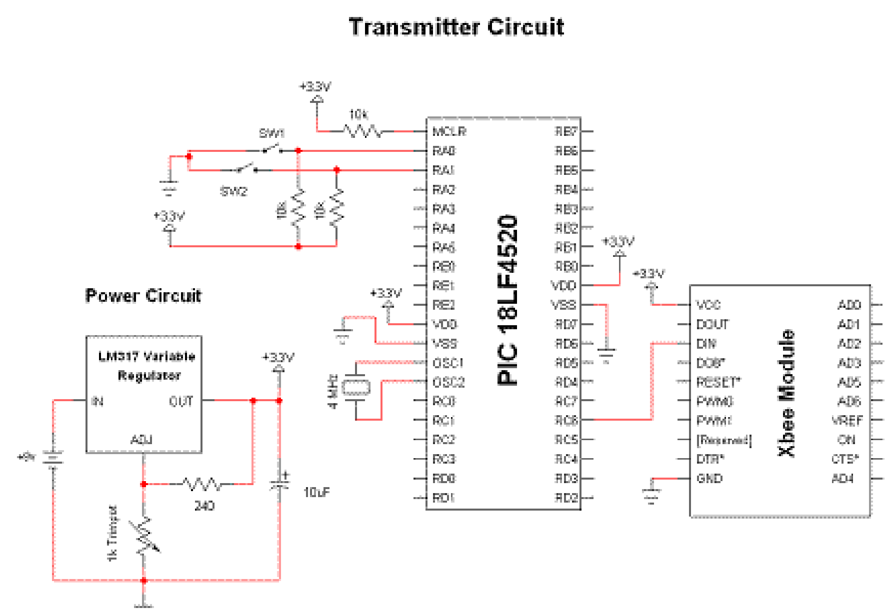

Figure 17: XBee transmitter schematic

Schematic found at http://www.pyroelectro.com/tutorials/xbee_wireless_interface/schematic.html

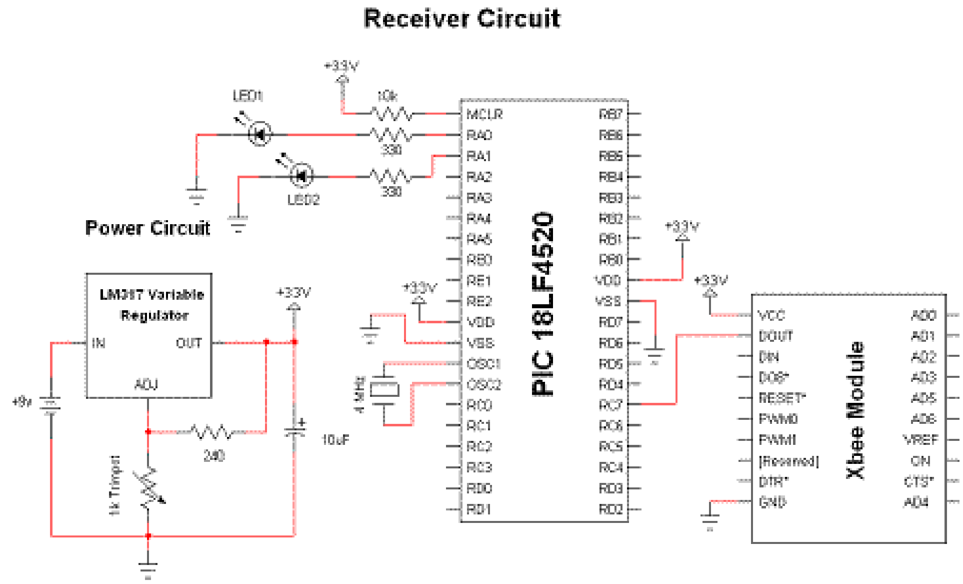

Figure 18: XBee receiver schematic

Schematic found at http://www.pyroelectro.com/tutorials/xbee_wireless_interface/schematic.html

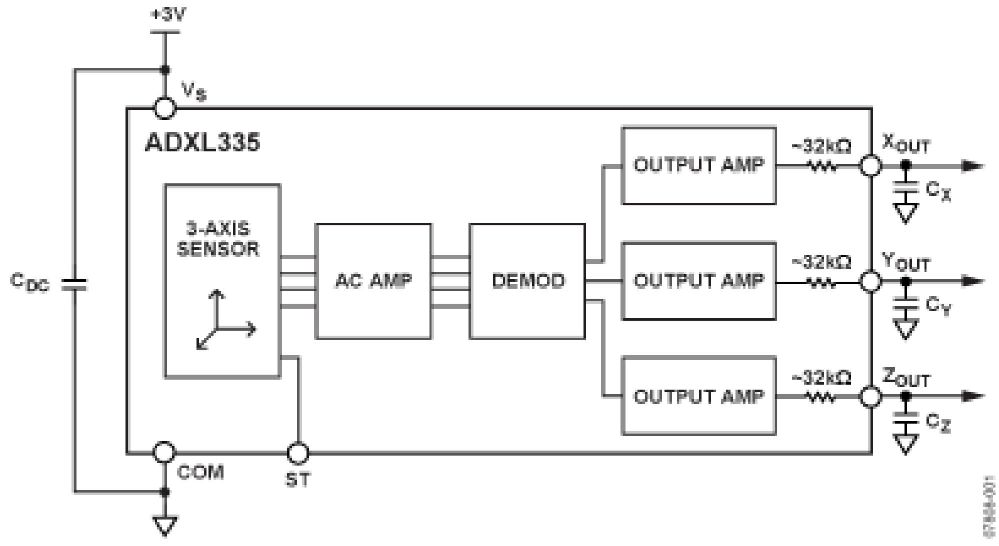

Figure 19: ADXL335 Accelerometer schematic

Schematic found at http://www.analog.com/en/mems-sensors/mems-inertial-sensors/adxl335/products/product.html

Appendix C: Code Used

Code for electric eye

```
#include <SoftwareSerial.h>//header file to call SoftwareSerial function

//setting output pins

#define xbeeRX 3

#define xbeeTX 4


const byte motionPin = 2; // motion detector input pin

int x = 0;              //used as internal count

byte senseMotion = 0;    // variable to hold current state of motion detector

SoftwareSerial xbee = SoftwareSerial(xbeeRX,xbeeTX); // setting up the RX, TX pins

void setup() {

  // set the digital pin directions

  pinMode(motionPin, INPUT);//pin used to detect motion

  Serial.begin(9600);//serial baud rate

  xbee.begin(9600);//Xbee baud rate

}


void loop()

{

   // Watching for motion

   senseMotion = digitalRead(motionPin);//read value on pin 2 to determine H or L

   if (senseMotion == HIGH & x==0) {    // motion sensor has been tripped

     Serial.print('H');

     xbee.print('H');

    x=1;
```

```
 delay (5000);
}

else if (senseMotion == HIGH & x==1)// motion sensor has been tripped for a second time
{
  Serial.print('L');
  xbee.print('L');
  x=0;
  delay (5000);
}
 senseMotion == LOW;
}
```

Code for on-horse device

```
#include <SoftwareSerial.h>//header files that allows us to call certain functions

#include "SD.h"//header file to write to SD card

#include <Wire.h>//header file to write to file


// pins used for XBee communication

#define xbeeRX 2

#define xbeeTX 3


//Analog read pins

const int xPin = 0;

const int yPin = 1;

const int zPin = 2;


// for the data logging shield, we use digital pin 10 for the SD cs line

const int chipSelect = 10;


// the logging file

File logfile;


//The minimum and maximum values that came from

//the accelerometer while standing still

//You very well may need to change these

//int minVal = 0;

//int maxVal = 1023;


//to hold the calculated values
```

```
double x;
double y;
double z;
int incomingByte;//from the Xbee reciever
int c = 0;//initialize the SD card only once

//setup software serial for XBee
SoftwareSerial xbee = SoftwareSerial(xbeeRX, xbeeTX); // RX, TX

void setup(){
  Serial.begin(9600);// baud rate for serial port to control speed of data output
  xbee.begin(9600);
}

void loop() {
  while(c < 1)
  {
    // initialize the SD card
  Serial.print("Initializing SD card...");
  // make sure that the default chip select pin is set to
  // output, even if you don't use it:
  pinMode(10, OUTPUT);

  // see if the card is present and can be initialized:
  if (!SD.begin(chipSelect))
  {
    Serial.println("Card failed, or not present");
```

```
  // don't do anything more:
  return;
}
Serial.println("card initialized.");

// create a new file
char filename[] = "LOGGER00.CSV";
for (uint8_t i = 0; i < 100; i++)
{
  filename[6] = i/10 + '0';//only increment every 10
  filename[7] = i%10 + '0';//prints remainder
  if (! SD.exists(filename))
  {
    // only open a new file if it doesn't exist
    logfile = SD.open(filename, FILE_WRITE);
    break;  // leave the loop!
  }
}
c = 1;
}
if (xbee.available() > 0) {//checks to see if info has been sent from other xbee
  incomingByte = xbee.read();//value sent by xbee transmitter
  if (incomingByte = 'H') {
      while(incomingByte != 'L') {
      //read the analog values from the accelerometer
      double xRead = analogRead(xPin);
      xRead = (xRead*3.3/1023-1.5)/.3;
```

```
        double yRead = analogRead(yPin);
        yRead = (yRead*3.3/1023-1.5)/.3;
        double zRead = analogRead(zPin);
        zRead = (zRead*3.3/1023-1.5)/.3;

        //Write to the SD card. comma separated values to place in excel
        logfile.print(xRead);
        logfile.print(",");
        logfile.print(yRead);
        logfile.print(",");
        logfile.println(zRead);
        logfile.println();

        //Output the calculations to the serial port
        Serial.print(xRead);
        Serial.print(",");
        Serial.print(yRead);
        Serial.print(",");
        Serial.println(zRead);
        Serial.println();
        delay(500); //just here to slow down the serial output - Easier to read
        incomingByte = xbee.read();//checking to see what is on the softwareSerial
      }
    }
    logfile.close();
  }
}
```

Appendix D: Example of Testing

1) Securely fasten the on – horse box to the gullet of the saddle as seen in figures 20 – 22.

Figure 20: Front view of device on saddle

Figure 21: Top view of device on saddle

Figure 22: Side view of device on saddle

2. Set up the electric eye as seen in figure 23.

Figure 23: Placement of the electric eye

3. After completing a run the devices are both shut off and the SD card is removed from the Data Logger. Placing the SD card into the computer it will automatically open up to the logger## file. The date will displays itself in excel as show below in table 3. X values are listed in the first column, Y values in the middle column, and Z values in the last column. According the positioning of the accelerometer inside the box the X values correspond to side to side motion, Y values correspond to forward and backward motion, and Z values correspond to up and down motion. Data points are taken approximately every 0.5 seconds.

Table 3: Data recorded from accelerometer

X(sideways)	Y(forward)	Z(up/down)
-1.51	-1.51	-0.47
-1.48	-1.51	-0.46
-1.49	-1.49	-0.48
-1.44	-1.49	-0.43
-1.53	-1.49	-0.48
-1.55	-1.49	-0.49
-1.54	-1.49	-0.48
-1.52	-1.51	-0.48
-1.53	-1.52	-0.49
-1.51	-1.51	-0.48
-1.51	-1.52	-0.48
-1.49	-1.51	-0.49
-1.48	-1.51	-0.48
-1.48	-1.51	-0.46
-1.47	-1.54	-0.49
-1.51	-1.49	-0.48
-1.48	-1.51	-0.48
-1.53	-1.49	-0.47
-1.46	-1.51	-0.48
-1.46	-1.52	-0.47
-1.46	-1.49	-0.46
-1.51	-1.53	-0.42
-1.51	-1.46	-0.43
-1.41	-1.51	-0.47
-1.39	-1.49	-0.46
-1.42	-1.49	-0.46
-1.41	-1.51	-0.47
-1.41	-1.49	-0.46
-1.39	-1.48	-0.46
-1.43	-1.48	-0.48

-1.43	-1.49	-0.47
-1.41	-1.49	-0.48
-1.43	-1.49	-0.47
-1.42	-1.48	-0.48
-1.42	-1.49	-0.48
-1.43	-1.48	-0.47
-1.42	-1.49	-0.48

4. The values are then graphed individually versus time to result in the graphs shown below (figures 24 - 26). The red lines are the sections when the horse is turning a barrel.

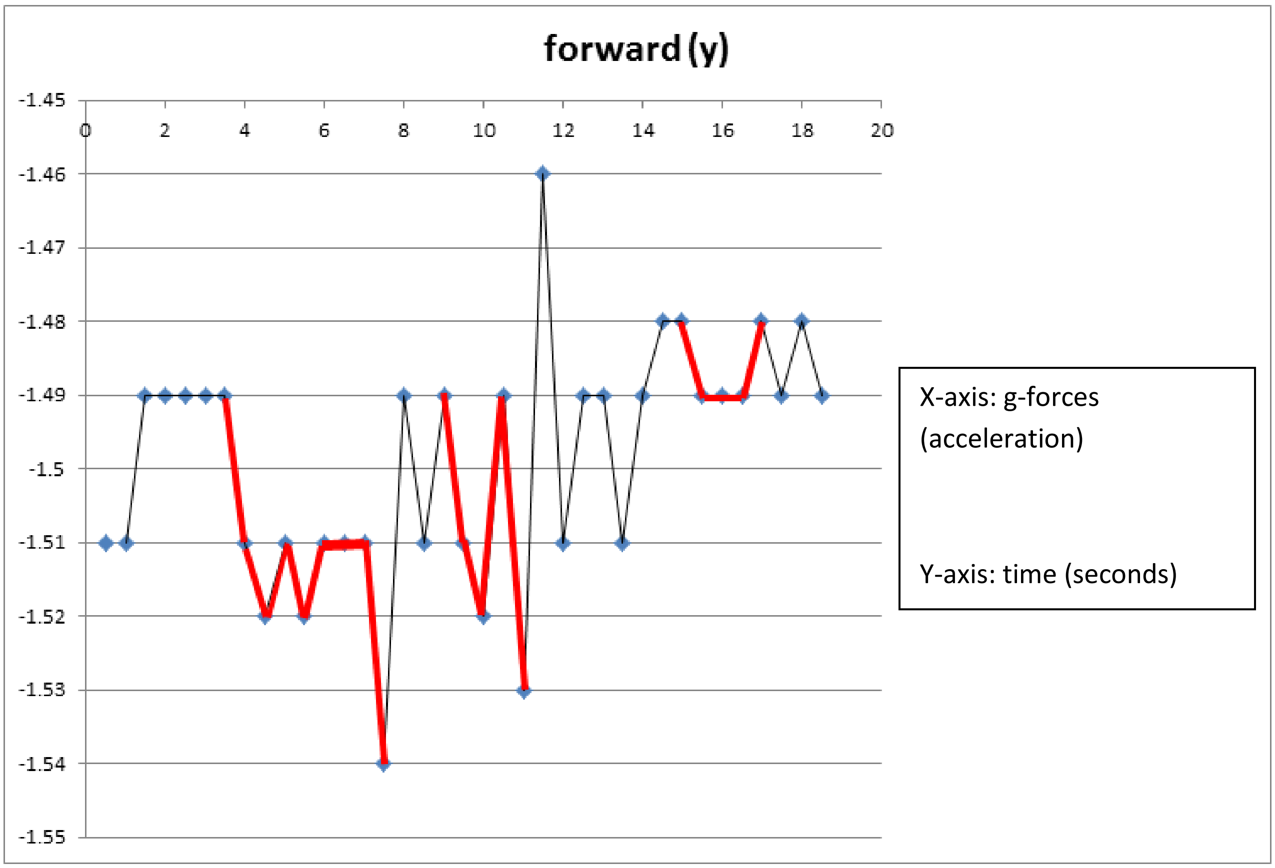

Figure 24: Forward motion (Y)

Figure 25: Sideways motion (x)

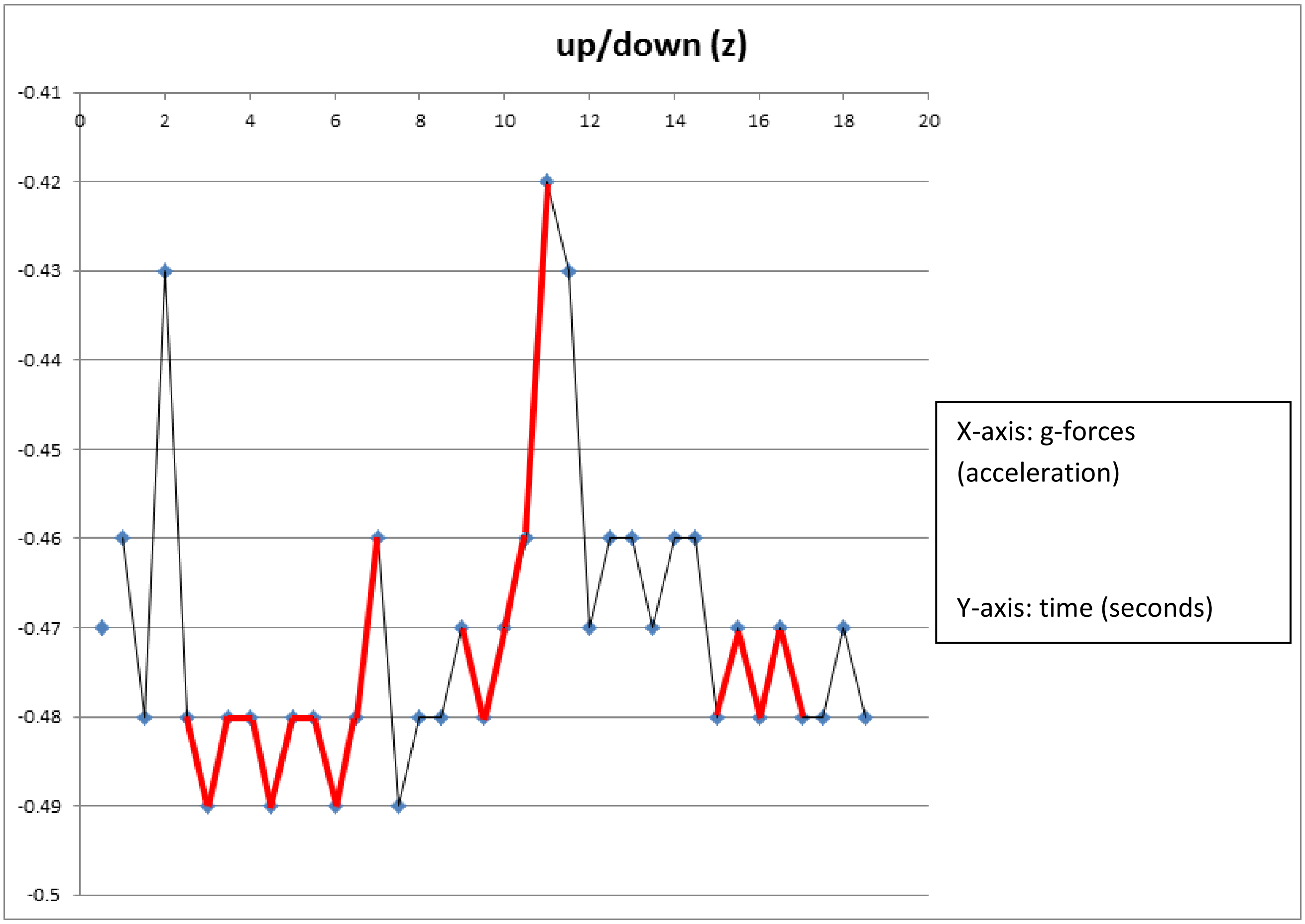

Figure 26: Up/Down motion (Z)

Analyzing this data is difficult and several improvements are necessary before this product becomes marketable. Decreasing the sensitivity of the accelerometer would help make the graphs smoother as would increasing the frequency.