

Faculty of Health, Engineering and Sciences

# Mapping rollercoaster forces using acceleration and GNSS data



A dissertation submitted by

**Mr. Zachary Montgomery**

In fulfillment of the requirements of

**Bachelor of Engineering (Honours)**

November 2020

---

# Abstract

Acceleration testing has been an integral part of amusement device safety for years. It is often included during the commissioning and design verification of new rides and as a preventative maintenance tool throughout their operating life cycle. However, there are limitations to current testing methods as the acceleration forces are measured and displayed as a function of time and have no direct connection with the fixed track upon which they are enacting.

This study has endeavored to correlate three-dimensional displacement with tri-axial acceleration data. The desired outcome was to produce a 'force map' of the amusement device that mirrors the fixed track in its dimensions. The method of achieving this was to couple precise MEMS accelerometer signals that are prone to drift with less precise but highly accurate GNSS receiver signals by aligning these datasets with wheel rotation counts as the train moves around the track.

Established and respected standards were used to determine feasibility requirements and the testing methods needed for acceleration sensors to produce accurate and repeatable data sets. The high vibration and acceleration forces of roller coasters proved to make this a challenging endeavor but a proof of concept was established. The learnings from this project can be used to build on the application of correlating acceleration data with GPS track location using revolution of wheel counts.

University of Southern Queensland  
Faculty of Health, Engineering and Sciences

## ENG4111 & ENG4112 Research Project

### **Limitations of Use**

The Council of the University of Southern Queensland, its Faculty of Health, Engineering and Sciences, and the staff of the University of Southern Queensland, do not accept any responsibility for the truth, accuracy or completeness of material contained within or associated with this dissertation.

Persons using all or any part of this material do so at their own risk, and not at the risk of the Council of the University of Southern Queensland, its Faculty of Health, Engineering and Sciences or the staff of the University of Southern Queensland.

This dissertation reports an educational exercise and has no purpose or validity beyond this exercise. The sole purpose of the course pair entitled “Research Project” is to contribute to the overall education within the student’s chosen degree program. This document, the associated hardware, software, drawings, and any other material set out in the associated appendices should not be used for any other purpose: if they are so used, it is entirely at the risk of the user.

The purpose of the project is to determine whether reliable data can be gathered in the manner described herein, not to gather actual data that relates to the safety of the amusement device itself.

## Certification

I certify that the ideas, designs and experimental work, results, analyses and conclusions set out in this dissertation are entirely my own effort, except where otherwise indicated and acknowledged.

I further certify that the work is original and has not been previously submitted for assessment in any other course or institution, except where specifically stated.

Zachary Montgomery

Student Number: XXXXXXXXXX

# Acknowledgements

*For my darling wife, Cynthia who has encouraged and supported me every step of the way.*

My studies at USQ have been transformative and I'd like to thank the faculty as a whole, with special mention to Dr. Jason Brown whose generosity and teaching helped me understand some of the more difficult engineering subjects. Also, my supervisor Prof. John Billingsley who has challenged me and made this project enjoyable and fun.

Special thanks to those at Village Roadshow Theme Parks who have supported me throughout my early career. Most notably Peter Henderson for your guidance and leadership, Adrian Summers for introducing me to acceleration testing which was the inspiration for this project and John Donaldson for your confidence in me and ongoing support.

Finally, thank you to all the electrical and maintenance technicians who have helped with the fabrication and testing for this project. It has been a pleasure to work alongside you and share our stories in Australia's best and biggest theme parks.

# Table of Contents

<b>ABSTRACT</b> .....	<b>I</b>
<b>LIMITATIONS OF USE</b> .....	<b>II</b>
<b>CERTIFICATION</b> .....	<b>III</b>
<b>ACKNOWLEDGEMENTS</b> .....	<b>IV</b>
<b>1. INTRODUCTION</b> .....	<b>- 1 -</b>
1.1. BACKGROUND AND IDEA INITIATION.....	- 1 -
1.2. CURRENT INDUSTRY STANDARDS, PRACTICE AND REGULATIONS.....	- 1 -
<b>2. LITERATURE REVIEW</b> .....	<b>- 3 -</b>
2.1. TECHNOLOGICAL COMPONENTRY.....	- 3 -
2.1.1. <i>Accelerometers</i> .....	- 3 -
2.1.2. <i>Gyroscopes</i> .....	- 6 -
2.1.3. <i>Global Navigation Satellite System (GNSS)</i> .....	- 9 -
2.1.4. <i>Hall Effect Sensors</i> .....	- 13 -
2.2. PROCESSING AND INTEGRATION.....	- 14 -
2.2.1. <i>A Complementary Approach</i> .....	- 14 -
2.2.2. <i>Digital Filters</i> .....	- 15 -
2.2.3. <i>Integrated Development Environment (IDE)</i> .....	- 16 -
2.3. PROJECT FEASIBILITY ANALYSIS AND STUDY JUSTIFICATION.....	- 16 -
<b>3. METHODOLOGY</b> .....	<b>- 18 -</b>
3.1. PROJECT DEVELOPMENT.....	- 18 -
3.1.1. <i>Aims, Objectives and Scope</i> .....	- 18 -
3.1.2. <i>Outcomes, Benefits and Applications</i> .....	- 18 -
3.2. PROJECT PLANNING.....	- 19 -
3.2.1. <i>Resources Required</i> .....	- 19 -
3.2.2. <i>Interpretation and Analysis of Results</i> .....	- 24 -
3.2.3. <i>Project Schedule</i> .....	- 25 -
3.2.4. <i>Ethical Considerations</i> .....	- 27 -
3.2.5. <i>Risk Assessment</i> .....	- 27 -
3.2.6. <i>Quality Assurance Plan</i> .....	- 27 -
<b>4. EQUIPMENT DESIGN</b> .....	<b>- 29 -</b>
4.1. TEST DUMMY.....	- 29 -
4.1.1. <i>Patron Coordinate System</i> .....	- 29 -
4.1.2. <i>Final Model</i> .....	- 31 -
4.2. HARDWARE.....	- 33 -
4.2.1. <i>Electronics</i> .....	- 33 -
4.2.2. <i>Packaging and Mounting</i> .....	- 37 -
4.3. SOFTWARE.....	- 39 -
4.3.1. <i>General Overview</i> .....	- 39 -
4.3.2. <i>Gathering Data</i> .....	- 40 -
4.3.3. <i>Digital filtering</i> .....	- 42 -

<b>5. EXPERIMENTAL DESIGN</b> .....	<b>- 43 -</b>
5.1. COLLECTION OF DATA.....	- 43 -
5.2. TESTING PROCEDURE .....	- 43 -
5.3. INSPECTION TEMPLATE.....	- 45 -
<b>6. EXPERIMENTAL PROGRAMME</b> .....	<b>- 48 -</b>
6.1. PROGRAMME OVERVIEW .....	- 48 -
6.2. TEST 1 – 24/9/2020 .....	- 48 -
6.3. TEST 2 – 25/9/2020.....	- 48 -
6.4. TEST 3 – 4/10/2020.....	- 49 -
6.5. TEST 4 – 6/10/2020.....	- 51 -
6.6. TEST 5 – 7/10/2020.....	- 51 -
6.7. TEST 6 – 8/10/2020.....	- 51 -
<b>7. RESULTS AND ANALYSIS</b> .....	<b>- 54 -</b>
7.1. 3-D ACCELERATION VERSUS TIME .....	- 54 -
7.2. 3-D ACCELERATION VERSUS DISTANCE .....	- 56 -
7.3. SPATIAL POSITION MAP VERSUS DISTANCE .....	- 58 -
<b>8. CONCLUSIONS</b> .....	<b>- 60 -</b>
<b>9. IDEAS FOR POTENTIAL FUTURE WORKS</b> .....	<b>- 61 -</b>
<b>REFERENCES</b> .....	<b>- 62 -</b>
<b>APPENDIX A – PROJECT SPECIFICATION</b> .....	<b>- 66 -</b>
<b>APPENDIX B – RISK ASSESSMENT</b> .....	<b>- 68 -</b>
<b>APPENDIX C – DATASHEETS</b> .....	<b>- 70 -</b>
<b>APPENDIX D – PROGRAM CODE</b> .....	<b>- 84 -</b>

# List of Figures

FIGURE 1: COORDINATE AXES REFERENCE FOR ACCELERATION DIAGRAM, (AS3533.1 2009) .....	1 -
FIGURE 2: ACCELERATION DIAGRAM SHOWING RESTRAINT TYPE AREAS (AS3533.1, 2009) .....	2 -
FIGURE 3: THE DELETERIOUS EFFECTS OF SENSOR DRIFT IN NAVIGATION, (KAI-WEI, CHENG-AN & THANH-TRUNG) .....	4 -
FIGURE 4: SIMULATED DRIFT OF MPU-6050 USING 0.5% BIAS ERROR.....	4 -
FIGURE 5: MECHANICAL FORCE-FEEDBACK PENDULOUS ACCELEROMETER, (GROVES 2013).....	5 -
FIGURE 6: BLOCK DIAGRAM OF MPU-6050, (MPU6050 2019).....	6 -
FIGURE 7: ORIENTATION OF AXES OF SENSITIVITY AND POLARITY OF ROTATION, (MPU6050 2019).....	7 -
FIGURE 8: CORIOLIS EFFECT CAUSES STORMS TO SWIRL, (NATIONAL GEOGRAPHIC 2020).....	8 -
FIGURE 9: AXES OF A VIBRATING GYRO, (GROVES 2013) .....	8 -
FIGURE 10: SATELLITE SYSTEMS IN ORBIT, (SPARKFUN 2020).....	9 -
FIGURE 11: INTERSECTING SIGNAL WAVES PINPOINT PRECISE LOCATION ON EARTH'S SURFACE, (GPS TRILATERATION 2019) .....	10 -
FIGURE 12: FOUR SATELLITES ARE REQUIRED FOR ACCURATE POSITIONING, (SPARKFUN 2020).....	10 -
FIGURE 13: PRINCIPLE SOURCES OF GNSS ERROR, (GROVES 2013).....	11 -
FIGURE 14: NMEA GPS CODE AS READ ON ARDUINO SERIAL MONITOR .....	12 -
FIGURE 15: "CROSSED" HALL EFFECT ANGLE SENSOR, (BILLINGSLEY 2006) .....	13 -
FIGURE 16: HALL EFFECT SENSOR, SHOWN IN A WHEEL, (BILLINGSLEY 2006).....	13 -
FIGURE 17: EXAMPLES OF LOW PASS FILTERS, (BILLINGSLEY 2006).....	15 -
FIGURE 18: ACCELERATION VS TIME PLOTS .....	17 -
FIGURE 19: PROJECT SCHEDULE.....	26 -
FIGURE 20: PATRON COORDINATE SYSTEM, (AS 3533.1 2009).....	29 -
FIGURE 21: MODIFIED TEST DUMMY (FRONT VIEW) .....	31 -
FIGURE 22: MODIFIED TEST DUMMY (SIDE VIEW).....	31 -
FIGURE 23: MODIFIED TEST DUMMY (TOP VIEW).....	32 -
FIGURE 24: MODIFIED TEST DUMMY (SENSOR MOUNTING PLATE).....	32 -
FIGURE 25: ELECTRONIC LAYOUT OF TEST DEVICE - INITIAL DESIGN .....	33 -



FIGURE 26: LM317 VOLTAGE REGULATOR CIRCUIT, (LM317 2002).....	- 34 -
FIGURE 27: PROTOTYPE TESTING PACKAGE (TOP VIEW).....	- 34 -
FIGURE 28: STREAMLINED ELECTRONICS LAYOUT (MK2).....	- 35 -
FIGURE 29: ELECTRONIC TESTING SETUP – NO GPS (MK3).....	- 36 -
FIGURE 30: ELECTRONIC TESTING SETUP W/ GPS (MK4).....	- 36 -
FIGURE 31: HALL EFFECT SENSOR MOUNTED TO WHEEL CARRIER.....	- 37 -
FIGURE 31: TEST DUMMY MOUNTED DURING TESTING.....	- 38 -
FIGURE 32: SD CARD PIN CONNECTIONS FOR ARDUINO MEGA, (HOW TO MECHATRONICS 2020).....	- 40 -
FIGURE 33: GPS PLOT TAKEN DURING TEST 2.....	- 49 -
FIGURE 34: TIGHT CLEARANCE BETWEEN ROAD WHEEL AND WHEEL CARRIER.....	- 50 -
FIGURE 35: MAGNETS TO SIT WITHIN WHEEL HUB RECESS.....	- 50 -
FIGURE 36: UBLOX U-CENTER GNSS GRAPHICAL INTERFACE.....	- 52 -
FIGURE 37: GPS MESSAGE ACCURACY – NMEA (TOP) VS UBX (BOTTOM).....	- 53 -
FIGURE 38: PROJECT ACCELERATION DATA X-AXIS 7/10/2020.....	- 54 -
FIGURE 39: HISTORICAL ACCELERATION DATA X-AXIS.....	- 54 -
FIGURE 40: PROJECT ACCELERATION DATA Z-AXIS 7/10/2020.....	- 55 -
FIGURE 41: HISTORICAL ACCELERATION DATA Z-AXIS.....	- 55 -
FIGURE 42: ACCELERATION VS WHEEL COUNT USING 4 MAGNETS RECORDED 6/10/2020.....	- 56 -
FIGURE 43: ACCELERATION VS WHEEL COUNT USING 1 MAGNET RECORDED 6/10/2020.....	- 57 -
FIGURE 44: ACCELERATION VS WHEEL COUNT USING NEW CODE 7/10/2020.....	- 57 -
FIGURE 45: GPS TRACE LAT & LON ONLY 8/10/2020.....	- 58 -
FIGURE 46: GPS TRACE WITH HAND DRAWN CONNECTIONS 8/10/2020.....	- 59 -

# List of Tables

TABLE 1: MPU-6050 ACCELEROMETER SENSITIVITY (MPU6050 2019).....	- 5 -
TABLE 2: NMEA RMC MESSAGE FORMAT (MATHWORKS 2020) .....	- 12 -
TABLE 3: THE COMPLIMENTARY FEATURES OF GPS AND ACCELEROMETER TECHNOLOGIES, (BENEDETTI, DERMANIS & CRESPI 2017) .....	- 14 -
TABLE 4: SOFTWARE AND HARDWARE EVALUATION.....	- 20 -
TABLE 5: PROJECT TASK DESCRIPTIONS.....	- 25 -
TABLE 6: INSPECTION TEMPLATE FOR RECORDING ACCELERATION DATA.....	- 45 -

# 1. Introduction

## 1.1. Background and idea initiation

There is a current market gap in the Amusement industry for analysing forces experienced on an amusement ride as it moves through space. Current testing methods used in industry measure acceleration over time only. For a fixed amusement device, it would be far more constructive to have the exact location in three-dimensional space of where the accelerations are occurring as opposed to estimating the location based on event timings.

## 1.2. Current Industry Standards, Practice and Regulations

Acceleration forces experienced on an Amusement ride will determine both the ride classification (1-5) and restraint type (1-5). With '1' being the lowest, e.g. intrinsically safe children's ride and '5' being the highest, e.g. high speed, inverting rollercoaster.

Verification of the ride classification and restraint type is performed during the design registration. The illustration shown in Figure 1 shows the positive and negative directions of the x and z axes, the values of which determine the restraint type as shown in Figure 2. AS3533.1 (2009).

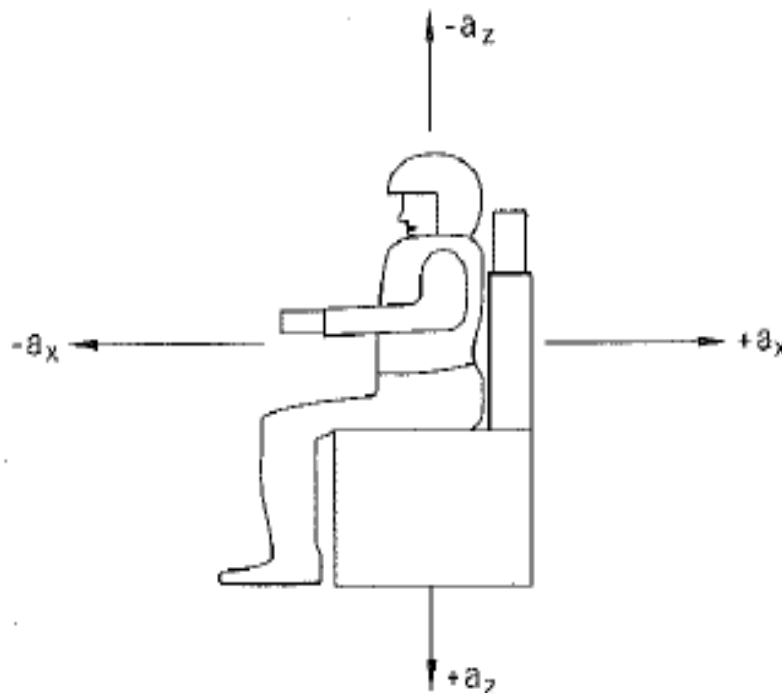


FIGURE 1: COORDINATE AXES REFERENCE FOR ACCELERATION DIAGRAM, (AS3533.1 2009)

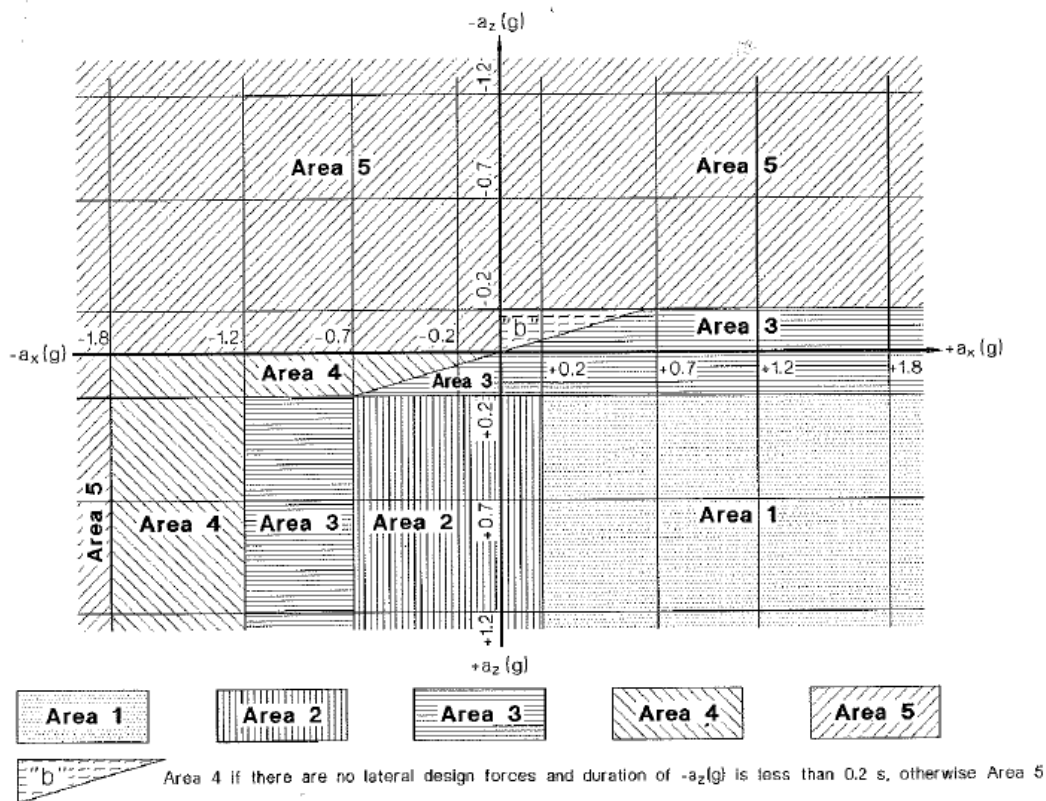


FIGURE 2: ACCELERATION DIAGRAM SHOWING RESTRAINT TYPE AREAS (AS3533.1, 2009)

Current methods for acquiring data related to the dynamic characteristics of amusement rides and devices are prescribed in ASTM F2137. Section 3.1.18 specifies a transducer as a device that converts physical phenomenon into a calibrated, electrical signal. This can include a number of sensors which are used to record a variety of physical data. The standard advocates specified testing, calibration, transducer performance, recorder performance, transducer mounting, and location criteria used to achieve standardized and repeatable results. These techniques will be incorporated into the project methodology. (ASTM International 2016).

Currently, accelerometers are the industry standard for measuring forces on Amusement devices and are often included as part of the preventative maintenance requirements from ride manufacturers. They are an accurate and effective tool used by both the project engineers representing the manufacturer and local engineers verifying the design.

The only requirements, regulatory or prescribed by the manufacturer regarding the measurement of forces experienced on an Amusement ride are to determine the maximum value of acceleration in each direction of each axis. (AS3533.1 2009). After scanning the literature, it became clear that neither real time or targeted monitoring have been adopted by manufacturers or industry and so there is potential to improve upon existing processes.

## 2. Literature Review

### 2.1. Technological Componentry

#### 2.1.1. Accelerometers

Fernandez & Dang (2013) explain that accelerometer sensors measure the acceleration exerted upon the sensor in up to three axis-vector components (x, y, z). They typically yield static force applied through gravity which can be used for tilt/orientation detection and dynamic force applied through movement. The term 'force' is used to describe acceleration in this report as they are directly proportional to one another. It should be noted that while it is the accelerations that are being measured, the forces experienced can be derived easily.

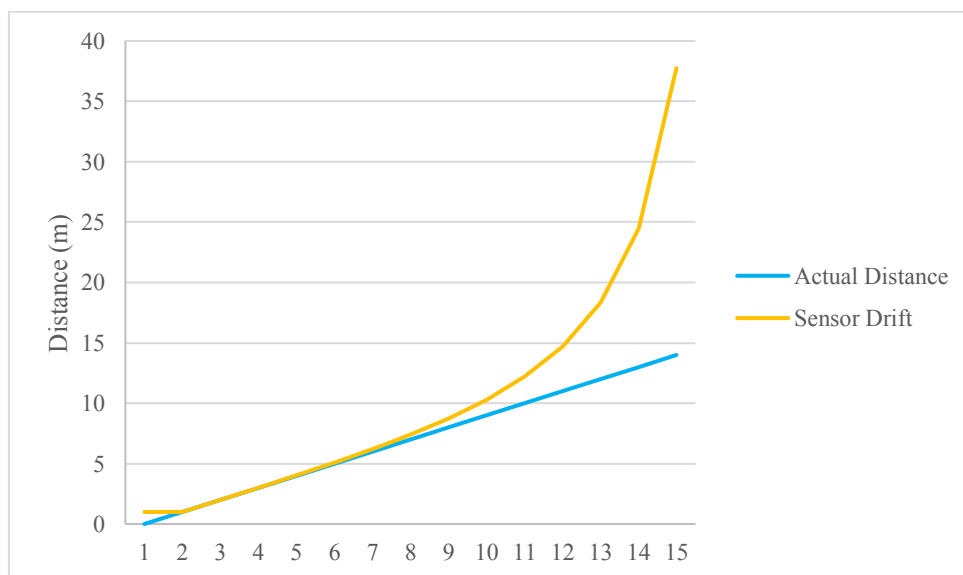
According to Corke (2017) accelerometers measure acceleration along a single axis and typically have three individual accelerometers packaged together and oriented orthogonally to mirror classical x, y, z axes. Once seated on a stable platform a simple integration of the acceleration measurement yields the velocity of the platform, a second integration allows the position to be obtained.

To obtain accurate position estimates acceleration sensors need to be extremely precise as any error present in the acceleration values will be compounded with the double integration. Low cost sensors used in phones, drones and the MPU-6050 used in this report's experiments are far from perfect and so suffer from bias. Bias varies from device to device and varies with both time and temperature. Consider that with a positive bias reading, the output will be bigger than it should be. At each recorded time-step the calculated offset in position will continue to grow. This phenomenon is known as sensor drift and is illustrated in Figure 3. (Corke 2017).



*FIGURE 3: THE DELETERIOUS EFFECTS OF SENSOR DRIFT IN NAVIGATION, (KAI-WEI, CHENG-AN & THANH-TRUNG)*

As outlined in the MPU6050 2019 datasheet, typical errors in non-linearity for this device would introduce a 0.5% offset. While this seems quite small at first glance, the compounding error of the double integration method limits quickly distorts the recorded position as shown in Figure 4.



*FIGURE 4: SIMULATED DRIFT OF MPU-6050 USING 0.5% BIAS ERROR*

TABLE 1: MPU-6050 ACCELEROMETER SENSITIVITY (MPU6050 2019)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
<b>ACCELEROMETER SENSITIVITY</b>						
Full-Scale Range	AFS_SEL=0		±2		g	
	AFS_SEL=1		±4		g	
	AFS_SEL=2		±8		g	
	AFS_SEL=3		±16		g	
ADC Word Length	Output in two's complement format		16		bits	
Sensitivity Scale Factor	AFS_SEL=0		16,384		LSB/g	
	AFS_SEL=1		8,192		LSB/g	
	AFS_SEL=2		4,096		LSB/g	
	AFS_SEL=3		2,048		LSB/g	
Initial Calibration Tolerance			±3		%	
Sensitivity Change vs. Temperature	AFS_SEL=0, -40°C to +85°C		±0.02		%/°C	
Nonlinearity	Best Fit Straight Line		0.5		%	
Cross-Axis Sensitivity			±2		%	

The inertial sensors used in this study are microelectromechanical systems (MEMS) technology. This involves using mass-produced, small and light quartz and silicone sensors that have been etched on a single wafer. The downside to these chips is their relatively poor performance due to their error behavior. Most accelerometers are either pendulous or use vibrating beams. Both technologies use the same basic principle, a mechanical force-feedback pendulous accelerometer found in closed-loop pendulous MEMS devices is illustrated in Figure 5. The torquer comprises an electromagnet mounted on the pendulum and a pair of permanent magnets of opposite polarity mounted on either side of the case. A capacitive pickoff is mounted so that as the pendulum moves the capacitance of one pair of plates increases as the other decreases. This capacitive value is then sent to an ADC on the IMU Sensor Registers as shown in Figure 6. (Groves 2013).

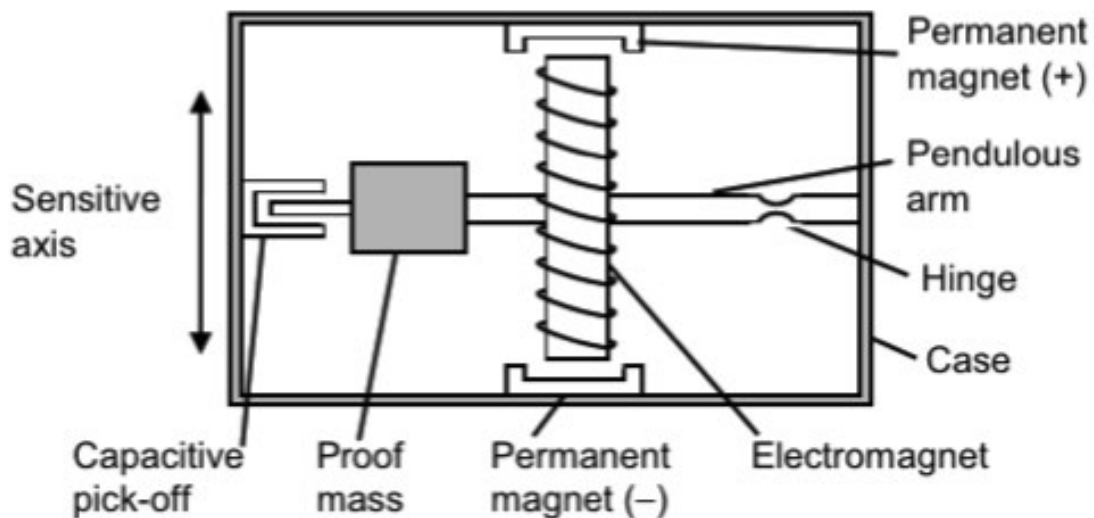


FIGURE 5: MECHANICAL FORCE-FEEDBACK PENDULOUS ACCELEROMETER, (GROVES 2013)

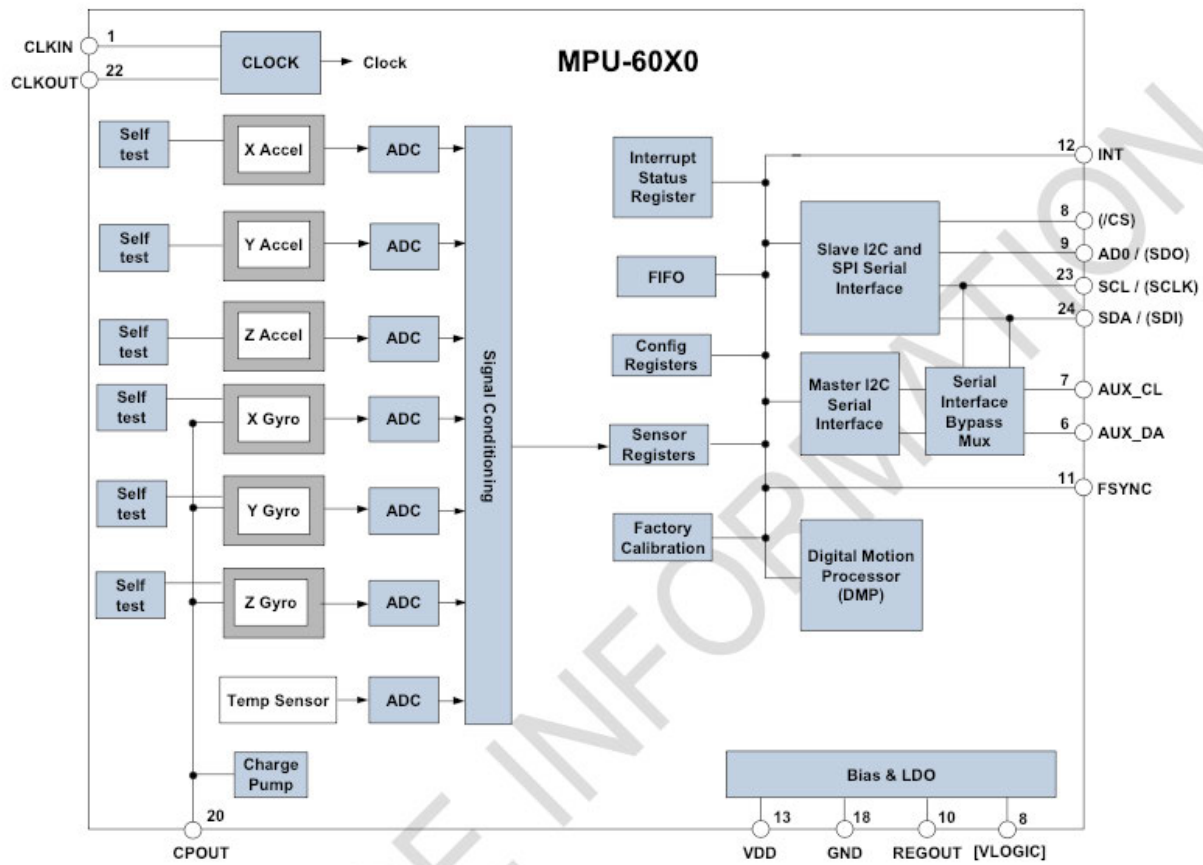


FIGURE 6: BLOCK DIAGRAM OF MPU-6050, (MPU6050 2019)

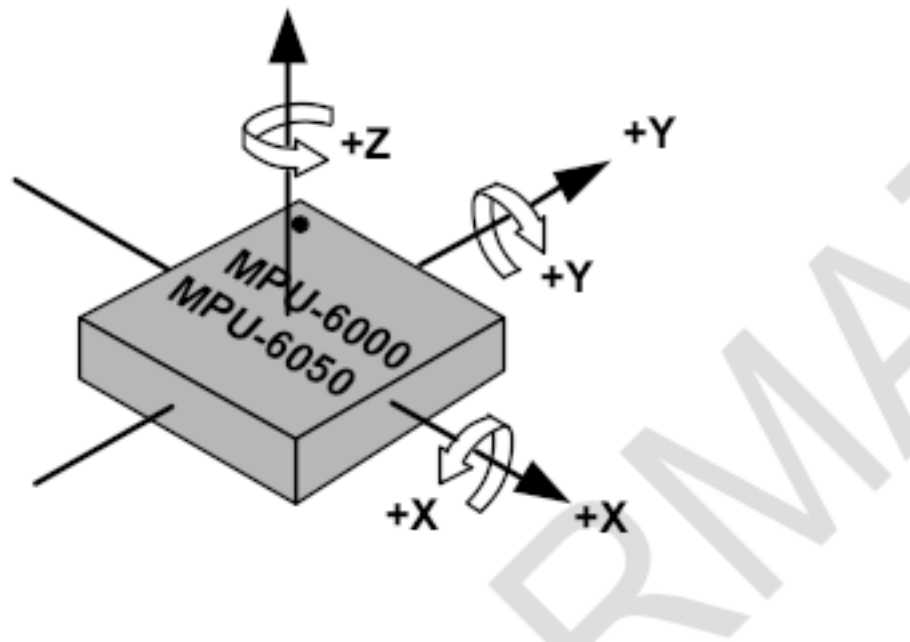
ASTM International (2016) states that the raw data obtained from an accelerometer must be processed so that it better reflects the actual forces experienced on the ride. A convenient and effective way of achieving this is through digital signal processing software. Oshana (ed.) (2012) explains that used correctly system designers can use ordinary languages such as C/C++ to produce an algorithm that digitally processes signals with high levels of efficiency. The flexibility and portability of this solution allows developers to process their data on desktop computers. (Oshana (ed.) 2012).

### 2.1.2. Gyroscopes

While accelerometers measure force along a specific axis, gyroscopes measure angular rate. Similar to accelerometers, three gyros are commonly set up orthogonally to measure along x, y and z axes. An *inertial measurement unit* (IMU) combines multiple accelerometers and gyros to produce three-dimensional measurement of specific force



and angular rate. The MPU-6050 IMU used in this report's experiments is such a device using three of each inertial sensor as illustrated below in Figure 7. (Groves 2013).



*FIGURE 7: ORIENTATION OF AXES OF SENSITIVITY AND POLARITY OF ROTATION, (MPU6050 2019)*

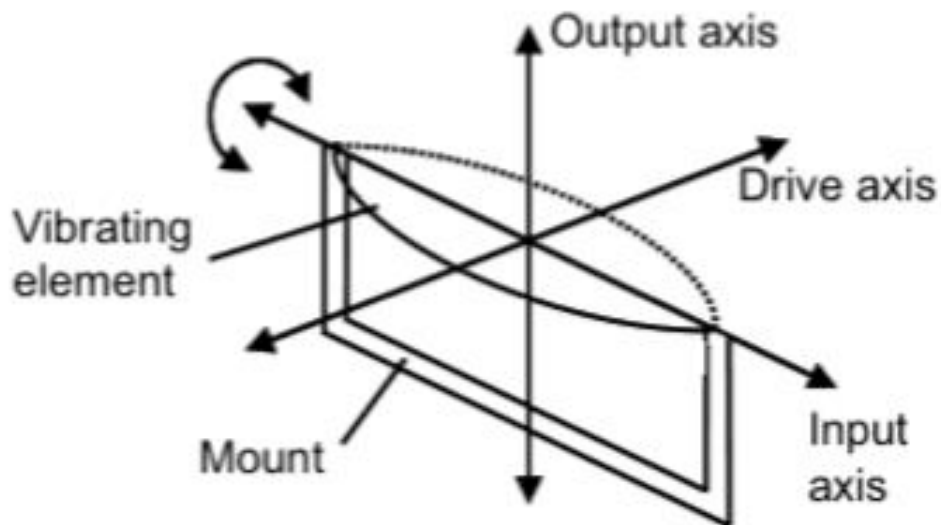
Corke (2017) explains that the gyros used in MEMS technology vary somewhat with different designs, but all contain a mass vibrating at a high frequency typically over 10kHz in a plane. Rotation about an axis normal to that plane causes an orthogonal displacement which is measured capacitively.

This displacement is caused by the Coriolis acceleration which instigates simple harmonic motion along the axis of the vibrating gyro. As explained by National Geographic (2020) the Coriolis effect is caused by the Earth rotating faster at the equator than it does at its poles, naturally causing a deflection to all objects not firmly connecting to the ground. This is a global phenomenon best illustrated by large scale weather systems as shown in Figure 8. (Groves 2013).



*FIGURE 8: CORIOLIS EFFECT CAUSES STORMS TO SWIRL, (NATIONAL GEOGRAPHIC 2020)*

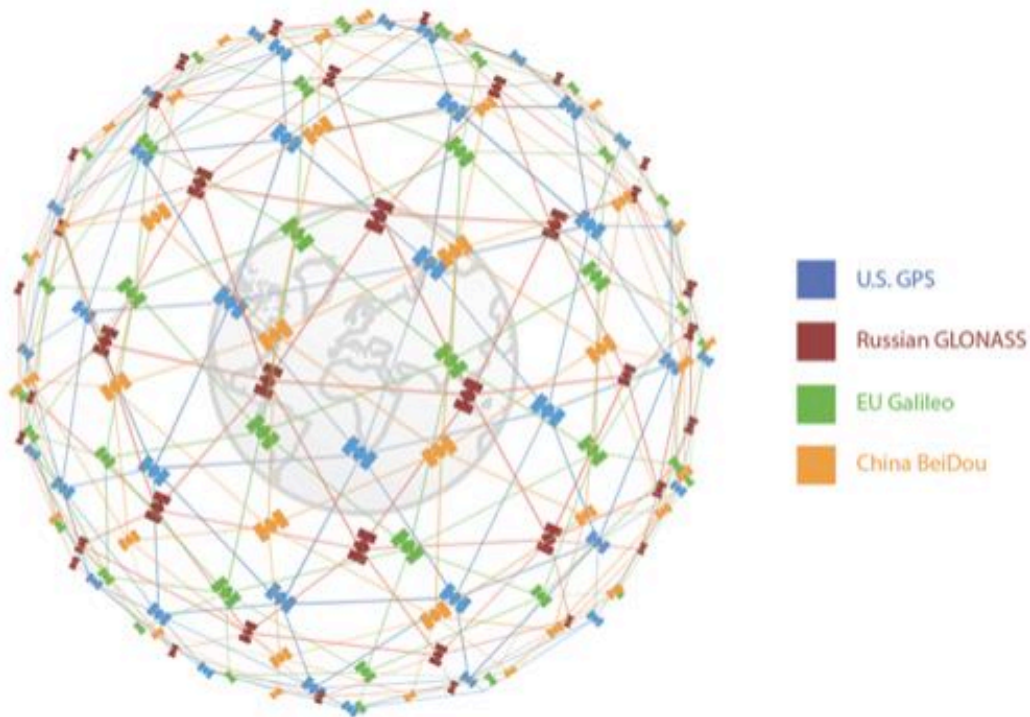
The amplitude of this motion is proportional to the angular rate and so can be measured. As shown in Figure 9, any rotation about the vibration axis won't result in a Coriolis acceleration which is why three sensors must be set up orthogonally to capture three-dimensional position. (Groves 2013).



*FIGURE 9: AXES OF A VIBRATING GYRO, (GROVES 2013)*

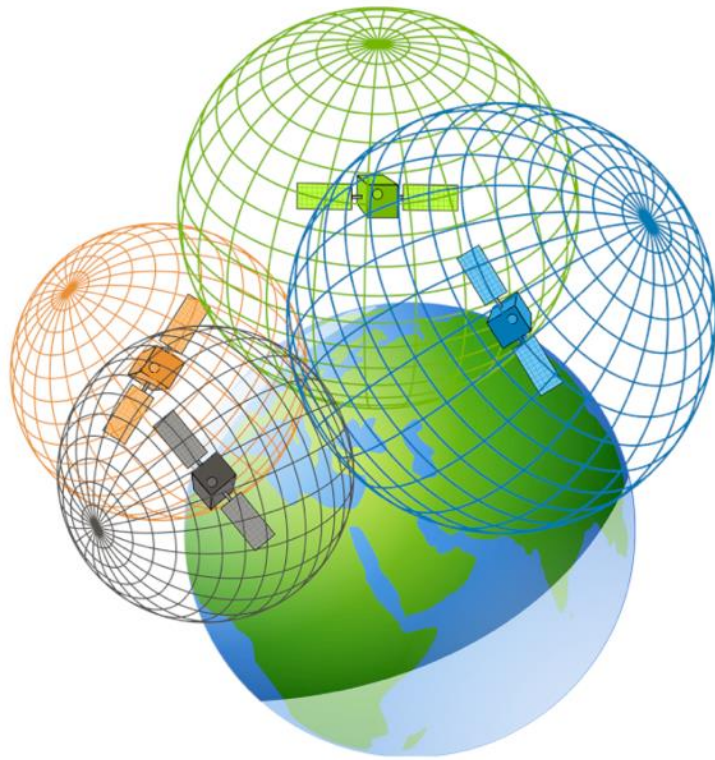
### 2.1.3. Global Navigation Satellite System (GNSS)

As explained by Sparkfun (2020) Global Navigation Satellite Systems (GNSS) are multiple constellations of satellites that orbit roughly 20 000km above the Earth's surface. Four of the main constellations are illustrated in Figure 10. Each satellite uses an incredibly accurate atomic clock to send out a unique, timestamped signal wave that can be picked up by a receiver.



*FIGURE 10: SATELLITE SYSTEMS IN ORBIT, (SPARKFUN 2020)*

The distance from a satellite to a receiver on the Earth can be pinpointed by knowing the precise time a signal wave left the satellite and multiplying this by the rate of the signal (speed of light). As illustrated in Figure 11 and Figure 12, the nature of the signal wave means that this distance could in fact lie anywhere along the edge of a sphere meaning that with only one satellite the exact position of the receiver on the Earth's surface would be impossible to extrapolate. In fact, to determine a location accurately in three dimensions at least four satellites are required. Three for x, y and z coordinates (latitude, longitude, altitude) and one more to determine the time for the signal to travel from the satellites to the receiver. (Sparkfun 2020).

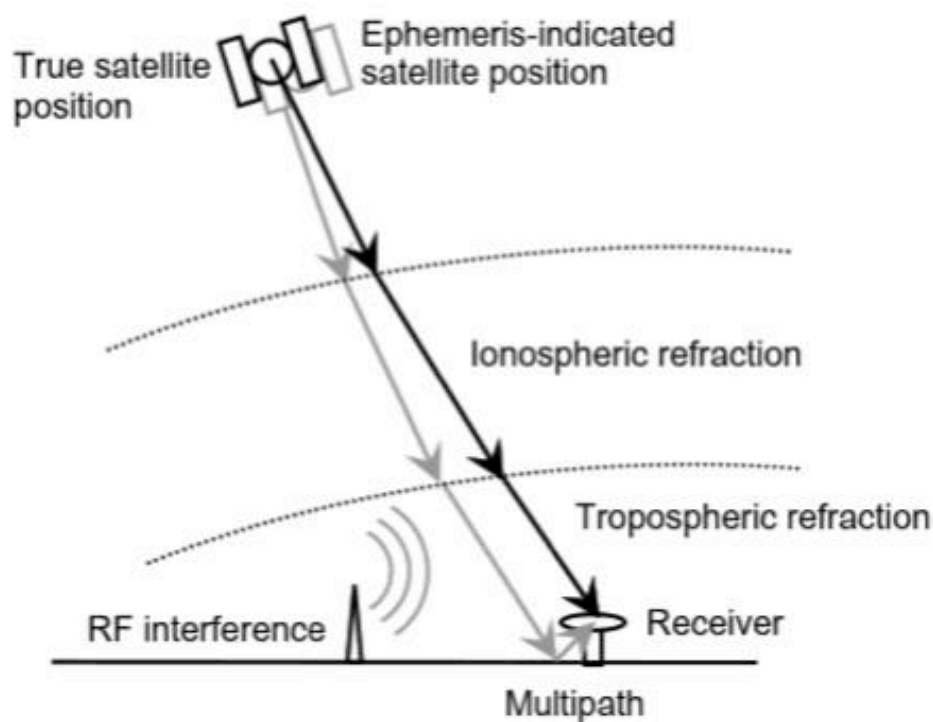


*FIGURE 11: INTERSECTING SIGNAL WAVES PINPOINT PRECISE LOCATION ON EARTH'S SURFACE, (GPS TRILATERATION 2019)*



*FIGURE 12: FOUR SATELLITES ARE REQUIRED FOR ACCURATE POSITIONING, (SPARKFUN 2020)*

There are a number of GNSS error sources that can distort the true positioning of a receiver as shown in Figure 13. The ephemeris-indicated satellite position (which is sent to the receiver) may not match the true satellite position and the signal may be refracted in both the Ionosphere and the Troposphere as it travels through the Earth's atmosphere. As it nears the Earth's surface tall buildings or natural formations like mountains may block the signal or it may encounter RF interference or multipath interference as it bounces off the ground and the receiver is subjected to more than one path. Despite this, most GNSS receivers are accurate to within a few meters depending on weather conditions and satellite availability.



*FIGURE 13: PRINCIPLE SOURCES OF GNSS ERROR, (GROVES 2013)*

Turning the signal waves sent from multiple satellites into useful information is done through various GPS communication protocols including binary messages and the NMEA-0183 standard (NMEA) which breaks down these signals into readable sentences of information as shown in Figure 14. When using NMEA there are multiple sentences sent with each signal wave, each with varying packets of information that start with easily identifiable protocol headers, e.g. \$GPRMC and \$GPGGA. (Sparkfun 2020).

```

/dev/cu.usbmodem14101
Send
22:09:04.184 -> $GPGGA,120904.00,2755.28776,S,15324.31612,E,1,07,2.90,23.1,M,37.5,M,,*7D
22:09:04.251 -> $GPGSA,A,3,12,02,13,05,29,25,15,,,,,5.63,2.90,4.82*09
22:09:04.318 -> $GPGSV,3,1,12,02,14,120,34,05,44,132,38,12,40,006,32,13,13,068,21*78
22:09:04.385 -> $GPGSV,3,2,12,15,21,036,32,18,35,263,19,20,19,324,,23,13,329,10*7D
22:09:04.452 -> $GPGSV,3,3,12,25,61,317,22,26,06,218,,29,56,189,25,31,03,257,*78
22:09:04.521 -> $GPGLL,2755.28776,S,15324.31612,E,120904.00,A,A*75
22:09:05.069 -> $GPRMC,120905.00,A,2755.28777,S,15324.31612,E,0.149,,170820,,A*6C
22:09:05.137 -> $GPVTG,,T,,M,0.149,N,0.275,K,A*2F
22:09:05.174 -> $GPGGA,120905.00,2755.28777,S,15324.31612,E,1,07,2.90,23.0,M,37.5,M,,*7C
22:09:05.246 -> $GPGSA,A,3,12,02,13,05,29,25,15,,,,,5.63,2.90,4.82*09
22:09:05.314 -> $GPGSV,3,1,12,02,14,120,34,05,44,132,38,12,40,006,32,13,13,068,21*78
22:09:05.384 -> $GPGSV,3,2,12,15,21,036,32,18,35,263,18,20,19,324,,23,13,329,10*7C
22:09:05.452 -> $GPGSV,3,3,12,25,61,317,22,26,06,218,,29,56,189,25,31,03,257,*78
22:09:05.519 -> $GPGLL,2755.28777,S,15324.31612,E,120905.00,A,A*75
22:09:06.077 -> $GPRMC,120906.00,A,2755.28775,S,15324.31621,E,0.029,,170820,,A*6A
Autoscroll Show timestamp Carriage return 9600 baud Clear output

```

FIGURE 14: NMEA GPS CODE AS READ ON ARDUINO SERIAL MONITOR

GPRMC and GPGGA sentences were of particular interest to this project as they contain not only time of position fix, latitude and longitude but also speed and course over ground as shown in Table 2.

TABLE 2: NMEA RMC MESSAGE FORMAT (MATHWORKS 2020)

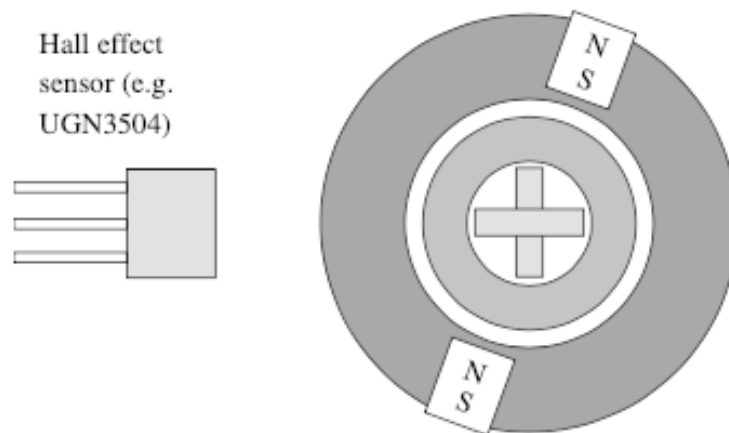
\$GPRMC,hhmmss,Status,Latitude,N,Longitude,E,SOG,COG,ddmmyy,MV,MVE,Mode\*CS<CR><LF>

\$GPRMC,083559.00,A,4717.11437,N,00833.91522,E,0.004,77.52,091202,,,A\*57

Field No	Name	Example	Description
1	\$GPRMC	\$GPRMC	Message ID, RMC protocol header
2	hhmmss.ss	083559.00	UTC Time, Time of position fix, 08:35:59.00 UTC
3	status	A	Status, V – Navigation receiver warning, A – Data valid
4	Latitude	4717.11437, N	Latitude 47 deg 17.11437' North
5	Longitude	00833.91522, E	Longitude 8 deg 33.91522' East
6	SOG	0.004	Speed over Ground in Knots
7	COG	77.52	Course over ground in degrees
8	ddmmyy	091202	Date, 09th December 2002
9	MV, MVE	Receiver did not output these values	Magnetic Variation and Magnetic Variation E/W indicator
10	Mode	A	Mode indicator, (Only for NMEA version above 2.2) N – No Fix A – Autonomous GNSS Fix D – Differential GNSS Fix E – Estimated/Dead Reckoning Fix
11	CS	57	Checksum
12	CR LF		Carriage Return and Line Feed

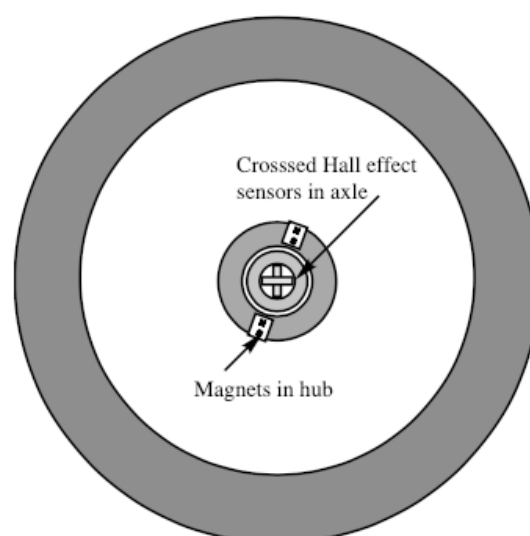
## 2.1.4. Hall Effect Sensors

As stated by Billingsley (2006) analog Hall Effect sensors such as the one shown in Figure 15 are capable of detecting a magnetic field perpendicular to their sensing plane. They are advantageous due to their “non-contact” operation and high reliability.



*FIGURE 15: “CROSSED” HALL EFFECT ANGLE SENSOR, (BILLINGSLEY 2006)*

By positioning magnets in the hub of a road wheel for instance and counting the revolutions of each wheel using a single or pair of crossed hall effect sensors as shown in Figure 16, an accurate measurement of the total length of the track can be achieved. This is aided by the closed loop nature of the track and the precision parking system already in place on most roller coasters which ensures the train starts and stops in basically the same position.



*FIGURE 16: HALL EFFECT SENSOR, SHOWN IN A WHEEL, (BILLINGSLEY 2006)*

## 2.2. Processing and Integration

### 2.2.1. A Complementary Approach

Much research has been conducted into the merging of GNSS and Micro-Electro-Mechanical System (MEMS) accelerometer technologies. Their inherent properties complement each of their strengths and weaknesses as outlined in Table 3. (Benedetti, Dermanis & Crespi 2017) assessed the feasibility of loosely integrating low cost GNSS and MEMS accelerometers by separately processing the kinematics (displacement and velocity) of each sensor using numerical integration with promising results.

*TABLE 3: THE COMPLIMENTARY FEATURES OF GPS AND ACCELEROMETER TECHNOLOGIES, (BENEDETTI, DERMANIS & CRESPI 2017)*

	<b>GPS</b>	<b>Accelerometer</b>
Dynamic Displacement	Measures directly, real-time	Requires double integration
Freq. bandwidth	Currently up to 100 Hz	More than 1000 Hz
Portability and Deployment	Portable, outdoor only, requires sky visibility	Portable, indoor and outdoor
Position Stability	High stability over time, no bias, no drift	Poor stability over time, bias and drift
Position Accuracy	mm to cm	millimeter level
Tolerance	All weather system	Low temperatures



(Kai-Wei, Cheng-An & Thanh-Trung 2014) successfully tested the integration of high frequency GPS, non-GPS derived references including digital maps and terrain models, and tactical inertial navigators such as accelerometers. They tested on land and then in the air using an unmanned aerial vehicle and the results showed a vast improvement over a standalone inertial navigation system with sustained results in the meter-level accuracy range.

### 2.2.2. Digital Filters

To remove unwanted high-frequency noise from the accelerometer signal, it is best to filter it out using a simple lowpass RC circuit as shown in Figure 17. Digitized filtering is possible but is prone to aliasing, which produces an unrepresentative signal output. (Billingsley 2006).

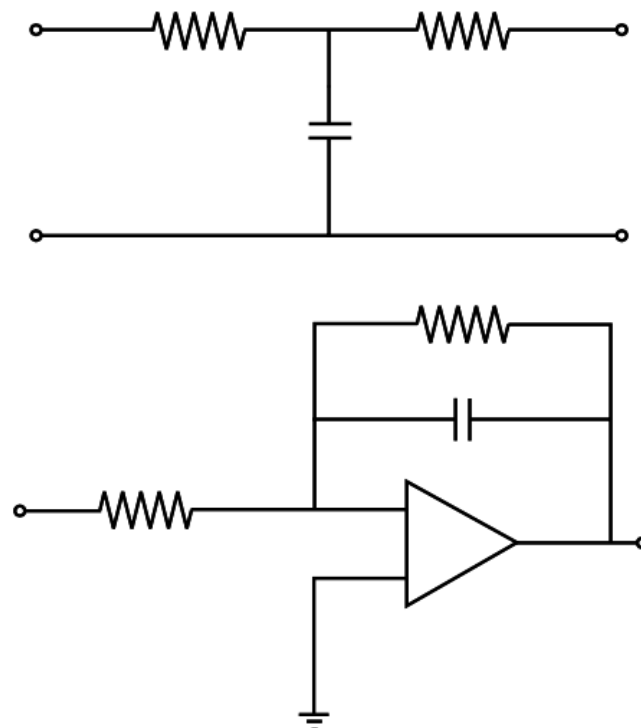


FIGURE 17: EXAMPLES OF LOW PASS FILTERS, (BILLINGSLEY 2006)

### 2.2.3. Integrated Development Environment (IDE)

An integrated Development Environment (IDE) is a comprehensive software package that allows programmers to build, edit, compile and debug source code and then send that code to a microcontroller to be used remotely. Three popular IDEs were investigated to be used for this project: Arduino, Mu editor and Android Studio.

Arduino is an open-source electronics platform developed for use with Arduino boards. The programming language is based on C/C++ and is easy to use, it is extendable and has the support of many experienced programmers. (What is Arduino? 2019).

Mu-editor is a Python code editor for beginner programmers using only the most essential features so that users are not baffled or intimidated by a complex interface. It has been developed to work with Micro: bit controllers. (About Mu 2019).

Android Studio is the official IDE for Android app development used to build projects on Android devices. It boasts a fast and feature rich emulator, extensive testing tools and frameworks and supports multiple languages including C++ and Java. (Meet Android Studio 2019).

## 2.3. Project feasibility analysis and study justification

Presently amusement rides rely on preventative maintenance performed by skilled workers and testing has been limited to an analysis of tri-axial acceleration vs time as illustrated in Figure 18. Current methods effectively portray accelerations and can be interpreted for both in-service monitoring and design verification as long as the equipment is adequate and used correctly. (ASTM International 2016).

However, there is definite scope to accurately measure accelerations against three-dimensional space. An advantage of this method would be having accelerations correlated to specific track members rather than having to work out where the vehicle was positioned at a given point in time. This opens up the possibility for rapid improvements in fault finding, condition monitoring, track gauging, design and verification.

The cost of componentry will be minimal, whether using individual components i.e. accelerometer, GPS and running them through an Arduino type microprocessor or as highlighted by Eriksson & Pendrill (2019) using a smartphone which includes a built-in accelerometer, GPS and video recorder. Software for both cases is free to develop and

access to amusement devices for testing was easily organised with VRTP making this a financially viable project.

No record was found in the literature for signal data being used to model accelerations in space. There were many theoretical or mathematical research articles but none that targeted live testing methods and especially not for amusement devices which justified completing the project.

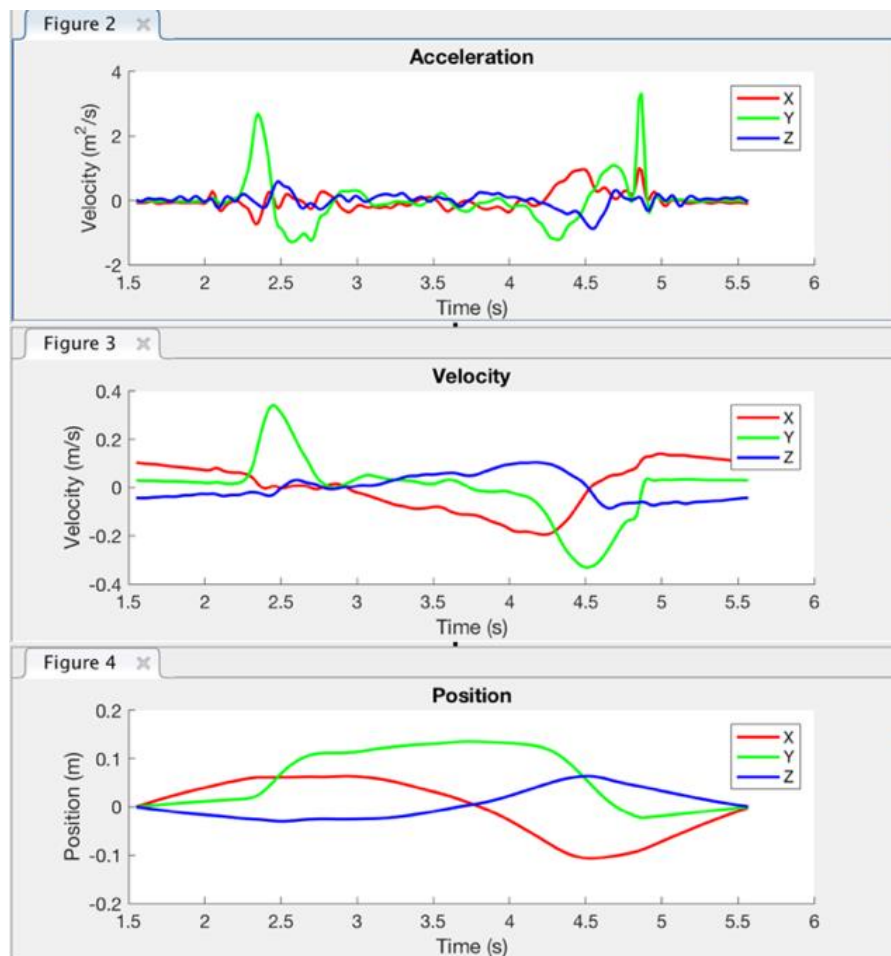


FIGURE 18: ACCELERATION VS TIME PLOTS

## 3. Methodology

### 3.1. Project Development

#### 3.1.1. Aims, Objectives and Scope

The aim of this project was to develop a useful tool for analysing the forces experienced on a fixed track amusement device. This project has been built on existing testing methods to match accelerations with specific track locations rather than at a given point in time. The intent is to increase accuracy of results and provide a more complete understanding of the forces involved. These aims will be fulfilled with the following objectives:

- Choose an accurate and repeatable way to record GPS and acceleration data that complies with ASTM F2137.
- Build a test dummy and sensor package to mount to a rollercoaster.
- Develop software to read, process and present acceleration and displacement data in an accurate and readable way.
- Correlate this data so that accelerations (forces) can be shown as a function of displacement rather than a function of time.
- Compare these results to another accepted testing method such as previous, accredited acceleration reports.
- Analyse and post filter recorded results to increase accuracy and create a force map of the rollercoaster.

The scope of this project is limited to a single, fixed track roller coaster owned and operated by VRTP. The device chosen was a Class 5 design registered ride of significant size and complexity which allowed for a thorough testing analysis. A traditional gravity roller coaster was chosen as they are the most popular style of ride and their layouts are better suited to measuring accelerations at varying points in space. Having a single train move through space in three planes (not two only) without crossing the same position more than once per cycle (no pendulum or rotary style rides) assisted in analysing the results.

#### 3.1.2. Outcomes, Benefits and Applications

This project will develop the understanding of forces/accelerations experienced on an Amusement device. Each measurement will coincide with a specific point along the

continuous track of the ride which will improve the accuracy of the model and allow for an easier interpretation of the results.

The expected outcomes, benefits and applications of this include:

- Increased safety by improving upon existing testing methods and adding another layer of control to mitigate the risk of a hazardous situation.
- Useful condition monitoring tool that can be implemented as a stand-alone test or as an autonomous device that monitors accelerations for each lap cycle.
- Increased accuracy and targeted results will provide a better understanding of the ride during the critical phase of the design verification.
- Baseline modelling during commissioning will produce a force map of the track that can be used to monitor the ride over its life cycle. This could be used as a reference to compare the data sent from the autonomous device every ride cycle to ensure accelerations are kept within a defined tolerance.
- Extend Asset life by confirming irregularities quickly and locally to specific track members or sections of track. This will ensure issues are identified before advanced damage to plant and equipment occurs.
- Reduced operational downtime by identifying issues early and planning maintenance to rectify them outside of park hours.
- Powerful tool for fault diagnosis which can help technicians and engineers understand the ride dynamics.

## 3.2. Project Planning

### 3.2.1. Resources Required

The project required the acquisition of data from multiple sensors to measure forces in space both precisely and accurately. As no one sensor can achieve this different hardware and software options were evaluated in Table 4 to develop an integrated solution.

These options included:

- GNSS/GPS
- Accelerometer
- Gyroscope
- Hall effect sensor
- Rotary encoder
- Microcontroller
- Software (IDE)

TABLE 4: SOFTWARE AND HARDWARE EVALUATION

ITEM	MODEL	SOURCE	COST	SPECIFICATION
<b>Laptop</b>	Apple Macbook Air	Student	\$0	MacOS Catalina Capable of running most interfacing software including Arduino, Python, JavaScript, etc.
<b>Microcontroller</b>	Arduino Uno	Student	\$0	ATMega328 14 Digital IO 6 Analog inputs 1 Serial Ports 16MHz clock 32KB Flash Memory 2KB SRAM 1KB EEPROM USB Arduino IDE (Arduino Uno Year Unknown).
<b>Microcontroller</b>	Arduino Mega 2560 R3	Student	\$0	ATMega2560 54 Digital IO 16 Analog Inputs 4 Serial Ports 16MHz clock 256KB Flash memory 8KB SRAM 4KB EEPROM USB Arduino IDE (Arduino Mega Year Unknown).
<b>Microcontroller</b>	BBC Micro: bit	Jaycar	\$35	ARM Cortex-M0 32-bit processor 256KB Flash ROM 16KB RAM 16MHz Clock 19 Digital IO including, 6 Analog Inputs 1 Serial Port Accelerometer / Magnetometer: 12bit, 16g range Temperature Sensing Bluetooth: 4.1 low energy, 2.4GHz band LED Matrix Micro USB MicroPython / JavaScript using Mu Editor IDE (Micro: bit 2019).
<b>Accelerometer / Gyroscope</b>	MPU-6050	Student	\$0	3.3V/5V 16bit AD converter 16bit data output Gyro $\pm$ 250 500 1000 2000 °/ s Acc $\pm$ 2, 4, 8, 16g (MPU-6050 2019).

ITEM	MODEL	SOURCE	COST	SPECIFICATION
GPS	NEO6MV2	Core Electronics	\$25	3.3V UART USB 50 Channel receiver 2.5m position accuracy 0.1m/s speed accuracy 5Hz data output (Neo6 2019).
GPS	NEO8MN	Ebay	\$15 (Not available due to COVID-19)	3.3V UART USB 72 Channel receiver 0.05m/s Velocity accuracy 0.3° Heading accuracy 2.5m Horizontal position accuracy 10Hz navigation update rate (NEO-M8 2020).
Hall effect sensor and magnet	XC4434 and LM1626	Jaycar	\$5 (sensor)  \$10 (each magnet)	5V Digital Output Active Low Schmitt Trigger 30 Gauss Activation threshold 10 Gauss Deactivation threshold (XC-4434 Year Unknown)
Two Phase Rotary Encoder	SR1230	Jaycar	\$10	5V SPST Pole Throw Type 7 poles 30 detents / 15 pulses per revolution (Rotary Encoder 2020)

### 3.2.1.1. Hardware Selection

#### Microcontroller: Arduino UNO vs Arduino MEGA vs BBC Micro: bit

*Cost: UNO (1st Tie), MEGA (1st Tie), Micro: Bit (3rd)*

Although the micro: bit is by no means an expensive microcontroller, having the Arduinos in possession meant that they were a simple option unless the micro: bit was able to provide better functionality.

*Functionality: MEGA (1st), Micro: Bit (2nd), UNO (3rd)*

As this project required acceleration data, GPS data, and rotary position (wheel revolution count) the Arduino Mega is the clear choice with 4 x serial ports and being of genuine

make, i.e. not a clone version. This allowed for the main serial Tx/Rx to be left open for the USB interface which was convenient. The Micro: bit has a diverse range of sensors including an accelerometer however, this was negated by the MPU-6050 accelerometer sensor I already had in possession. None of its other sensors had any application within this project.

*Component Used: Arduino MEGA*

### **Accelerometer / Gyroscope: MPU6050 vs BBC Micro: Bit**

*Cost: MPU6050 (1<sup>st</sup>), Micro: Bit (2<sup>nd</sup>)*

Having chosen the Arduino Mega as the microcontroller and having the MPU6050 already in possession, purchasing the Micro: Bit as an accelerometer was an added expense.

*Functionality: MPU6050 (1<sup>st</sup>), Micro: Bit (2<sup>nd</sup>)*

While both devices could achieve accurate recordings, the MPU6050 boasts a 16-bit resolution whereas the Micro: Bit is 12-bit. Both devices are capable of 16g measurement which is ample.

*Component Used: MPU6050*

### **GPS / GNSS Module: NEO6MV2 vs NEO8MN**

*Cost: NEO8MN (1<sup>st</sup>), NEO6MV2 (2<sup>nd</sup>)*

While very marginal the price for the newer model NEOM8 was cheaper than the locally sourced NEOM6 however, due to market interruptions from the COVID-19 crisis the stock was not able to be shipped in time. However, upon arrival the M6 was swapped out for the M8 which performed better in testing.

*Functionality: NEO8MN (1<sup>st</sup>), NEO6MV2 (2<sup>nd</sup>)*

The NEOM8 is an evolution of the NEOM6 with more channels, similar accuracy and a faster navigational update rate of 10Hz versus 5Hz. Unfortunately, the NEOM8 was not able to be sourced locally but the NEOM6 was deemed to be sufficiently accurate in its place.

*Component Used: Both*



## **Rotary Wheel Count: XC-4434 Hall Effect Sensor vs SR1230 Rotary Encoder**

*Cost: SR1230 (1<sup>st</sup>), XC-4434 (2<sup>nd</sup>)*

This is quite close but factor in spare / replacement magnets and the Hall Effect sensor seems less attractive.

*Functionality: XC-4434 (1<sup>st</sup>), SR1230 (2<sup>nd</sup>)*

While the XC-4434 is a simple option, the set up would need to be rigid to ensure the magnet is detected and no counts are missed. This was a concern but also even with a solid set up only one input is being counted per revolution. Even though this should be enough the SR1230 offers 15 inputs per revolution and with the 90° offset of the two-phase encoder, it is an intrinsically better design.

Even so, it was discovered that there was no way to mount the encoder onto a rotating wheel on the majority of rollercoasters which use wheel carriers and static pins to mount their running wheels. A magnet (or two) attached to the wheel hub and a hall effect sensor rigidly mounted clear of any moving gear was deemed to be the most practicable solution.

*Component Used: XC-4434*

### **3.2.1.2. Software**

As the microcontroller chosen in Section 2.2.1.1 was the Arduino Mega, it was decided to use the Arduino 1.8.12 IDE as the corresponding software platform. As explained by Arduino 1.8.12 (Year Unknown), “The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board.” The Arduino IDE proved to be a suitable choice for the testing although it did suffer from nuisance crashing. The simple nature of the interface and the multiple libraries and rich community support were invaluable in developing a complete system from scratch.

### *3.2.1.3. Machinery and Operators (during experiments / testing)*

Access to plant (roller coaster) and competent persons able to operate the plant (myself and other VRTP staff) required organization. The Road Runner Roller Coaster was chosen for its availability, relatively short ride cycle and simple, one-man operation. Other rides that could also have sufficed were ruled out due to scheduled maintenance. Testing was completed outside of park operational hours at night which allowed the maintenance team to free up resources to help with the testing and ensured there was no risk of impinging upon ride availability for guests at the park.

### 3.2.2. Interpretation and Analysis of Results

Data from testing was compared with historical acceleration data provided by VRTP to establish baseline authenticity. The existing acceleration tests have been documented and were completed by Vipac, a reputable and NATA accredited engineering firm who work within the requirements of AS 3533.1. The GPS data was compared with existing drawings of the ride and a 3D-rendered model on Google Earth.

As the path of travel is a fixed track, the intent of the captured GPS data set was to reproduce what looks like a discretized model of the track. However, the raw GPS signal tends to drift over time, though only by a few meters. (Billingsley 2020). Errors inherent in the accelerometer also needed to be accounted for which were largely negated by zeroing the data sets. Post filtering using software algorithms was performed on acceleration vs time data and could be applied to future programs.

The data from the accelerometer and GPS was recorded according to the data acquisition rates of the devices and associated program functions. The Arduino program was run in a continuous loop which output the values from each sensor to a file at set frequency or wheel count. This correlated the acceleration values in the x, y and z planes at a given point on the track with the intention of matching those points to a 3D GPS map via a lookup table. The file was coded in .csv format and then analysed on a personal computer which allowed for graphical reproduction of the results.

### 3.2.3. Project Schedule

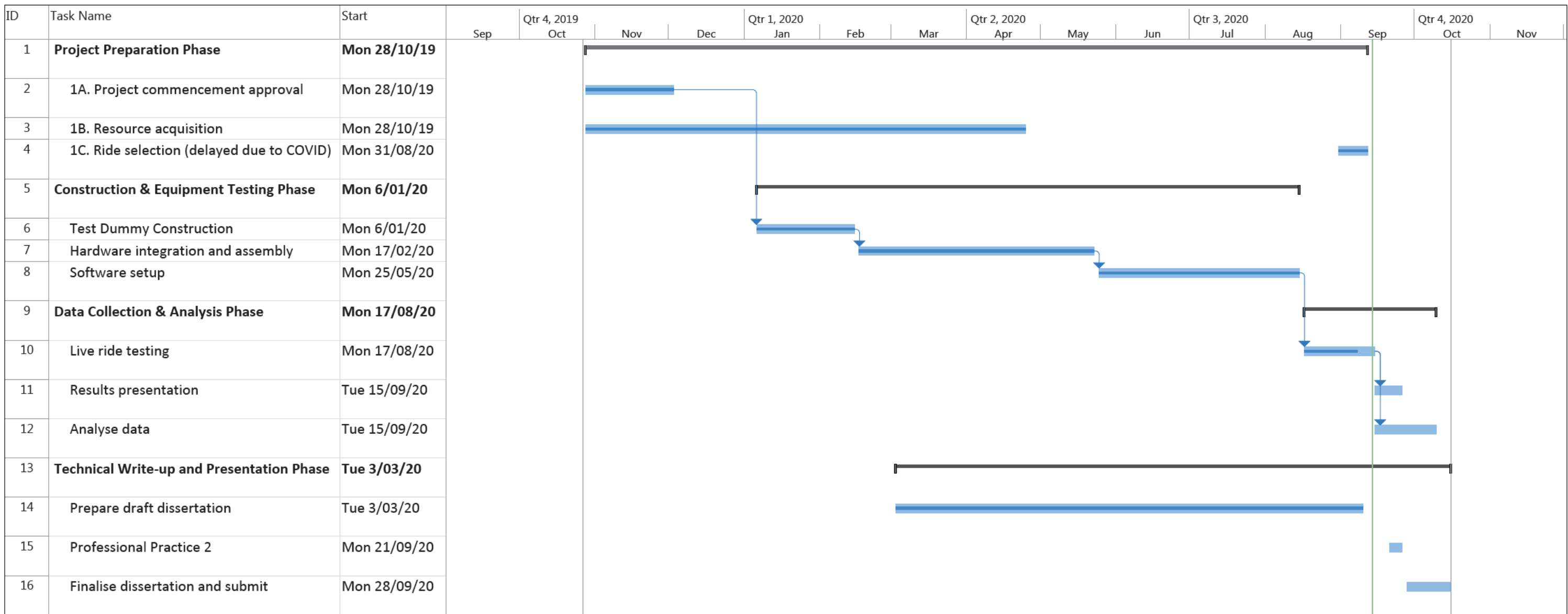
Due to the unforeseen interruption from COVID-19 this timeline has been adjusted to accommodate the availability of VRTP's Amusement Parks. These amendments include:

*1C – Ride Selection moved to Phase 3 due to uncertainty of available Amusement Devices.*

*2B – Test Dummy construction brought forward to 2A*

TABLE 5: PROJECT TASK DESCRIPTIONS

Phase 1	
<b>Project Preparation Phase</b>	
<b>1A</b>	<u>Project commencement approval</u> – Obtain official approval to commence project from USQ and VRTP.
<b>1B</b>	<u>Resource acquisition</u> – Decide on an option listed in Section 2.2.1.1 and procure necessary hardware and software.
<b>Phase 2</b>	
<b>Construction and Equipment Testing Phase</b>	
<b>2A</b>	<u>Test Dummy construction</u> – With VRTP's permission, modify a test dummy to be able to house the hardware assembly. Refer to Section 2.1.3.3.
<b>2B</b>	<u>Hardware integration and assembly</u> – Once all equipment has been purchased assemble components into a mountable container that can be attached to the test dummy.
<b>2C</b>	<u>Software setup</u> – IDE will be determined from the choice of hardware. Refer to Section 2.1.2.2. Initialise chosen microcontroller and test sensors for functionality in a controlled environment i.e. not in the field.
<b>Phase 3</b>	
<b>Data Collection &amp; Analysis Phase</b>	
<b>3A</b>	<u>Ride selection</u> – Finalize decision on which ride will be used for testing. Undertake audit of ride to ensure historical data is accurate and available and ensure project plan will not be affected by VRTP maintenance schedule.
<b>3B</b>	<u>Live ride testing</u> – Data is to be gathered using the hardware assembly attached to the test dummy mounted to the ride vehicle.
<b>3C</b>	<u>Analyse data</u> – post filter and review data. If not conclusive, troubleshoot and perform Step 3A again.
<b>3D</b>	<u>Results presentation</u> – Once data is deemed credible, export into a program to correlate GPS and acceleration values and present them as a 3D model.
<b>Phase 4</b>	
<b>Technical Write-up and Presentation Phase</b>	
<b>4A</b>	<u>Prepare draft dissertation</u> – The draft dissertation will be written and submitted for review by USQ Supervisor.
<b>4B</b>	<u>Professional Practice 2</u> – Present results and findings at the Residential School for ENG4903 online.
<b>4C</b>	<u>Finalise dissertation and submit</u> – After feedback from Supervisor the dissertation will be finalized, edited, reviewed and then submitted for grading.



Project: project timeline Date: Mon 14/09/20	Task		Project Summary		Manual Task		Start-only		Deadline
	Split		Inactive Task		Duration-only		Finish-only		Progress
	Milestone		Inactive Milestone		Manual Summary Rollup		External Tasks		Manual Progress
	Summary		Inactive Summary		Manual Summary		External Milestone		

### 3.2.4. Ethical Considerations

Permission from VRTP's Executive Officer – OHS&E was granted before any testing was undertaken. All data gathered will not be considered as safety critical due to the experimental nature in the way the tests were conducted. The results are therefore only to establish a proof of concept and may not accurately reflect the dynamics of the ride.

Although the quality of measurement equipment did not comply with ASTM requirements all data was reviewed to verify that the forces measured were within the design specifications of the ride. If any data had displayed out of parameter operation of the equipment, i.e. higher than expected forces this would have been reported to the Engineering Manager and investigated further.

### 3.2.5. Risk Assessment

The risk assessment tool used in this report is a similar version of that used by Village Roadshow Theme Parks. This tool was chosen to ensure that all activities undertaken on VRTP property would be in accordance with their policies and procedures. The Risk Assessment for the testing of equipment is included in Appendix B.

### 3.2.6. Quality Assurance Plan

To ensure that the project was completed to a high standard the following steps were taken.

- All hardware used was documented with particular attention to technical specifications.
- All software used was documented and any changes made recorded in a version control history.
- Live ride testing followed the procedure highlighted in Section 2.1.3.1.
- Each live ride test was documented using the template in Section 2.1.3.2.
- Original copies of the raw data were kept in a repository for reference.
- Copies of code used to filter the raw data was kept with the post filtered data itself.
- All data was presented using the same format to ensure like for like comparison of data sets.

Prior to submission, the results were submitted for review by Professor John Billingsley. Guidance from Prof. Billingsley was sought and given for delicate processes such as data acquisition and post filtering. The intent of the above controls was to ensure the final submission entailed reflective and meaningful results of a high standard.

## 4. Equipment Design

### 4.1. Test Dummy

#### 4.1.1. Patron Coordinate System

The patron coordinate system is used interchangeably in both ASTM F2137 and AS 3533.1. A test dummy was needed made to ensure accelerations are measured in reference to the accepted center of mass for a typical patron riding the amusement device. It was assumed that an unloaded train will give satisfactory results as the logistics of loading an entire train with 66-77kg weighted dummies in each patron location was not feasible for repeat testing. The sensor was mounted to the dummy in a way that represents a typical adult, not a child. (ASTM International 2016 & Australian Standards 2009).

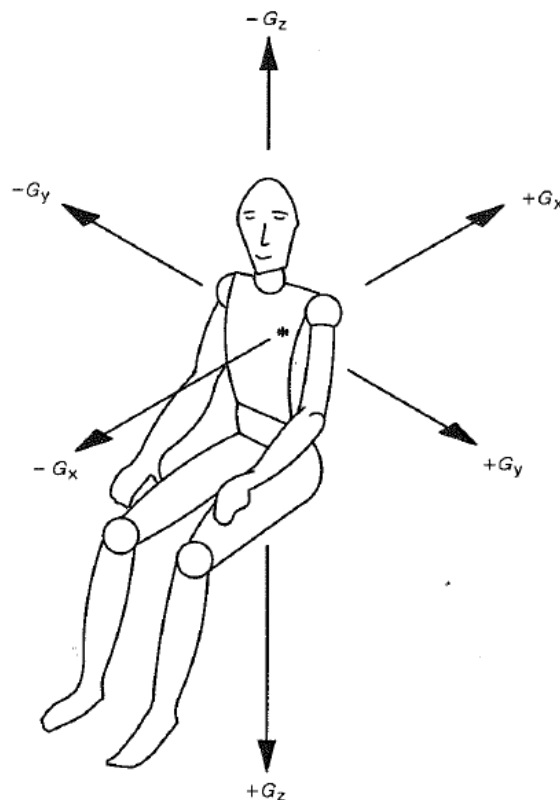


FIGURE 20: PATRON COORDINATE SYSTEM, (AS 3533.1 2009)

According to ASTM International (2016) the mounting location for the transducer must adhere to the following:

- Adults
  - Seated:  $600 \pm 50$ mm above seat level,  $100 \pm 30$ mm in front of back rest
  - Standing:  $1200 \pm 50$ mm above floor level,  $100 \pm 30$ mm in front of back rest
  - Prone:  $1200 \pm 50$ mm above foot contact level,  $100 \pm 30$ mm in front of back rest
  
- Children
  - Seated:  $500 \pm 50$ mm above seat level,  $50 \pm 20$ mm in front of back rest
  - Standing:  $760 \pm 50$ mm above floor level,  $50 \pm 20$ mm in front of back rest
  - Prone:  $760 \pm 50$ mm above foot contact level,  $50 \pm 20$ mm in front of back rest
  
- Mounting should be accurate to within  $5^\circ$  of the Patron Coordinate System (Figure 20).



#### 4.1.2. Final Model



*FIGURE 21: MODIFIED TEST DUMMY (FRONT VIEW)*



*FIGURE 22: MODIFIED TEST DUMMY (SIDE VIEW)*



*FIGURE 23: MODIFIED TEST DUMMY (TOP VIEW)*



*FIGURE 24: MODIFIED TEST DUMMY (SENSOR MOUNTING PLATE)*

## 4.2. Hardware

### 4.2.1. Electronics

The initial design focused on having adequate 3.3V and 5V power supply to the sensors as based on various data sheets it was unclear if the Arduino would be able to supply adequate current.

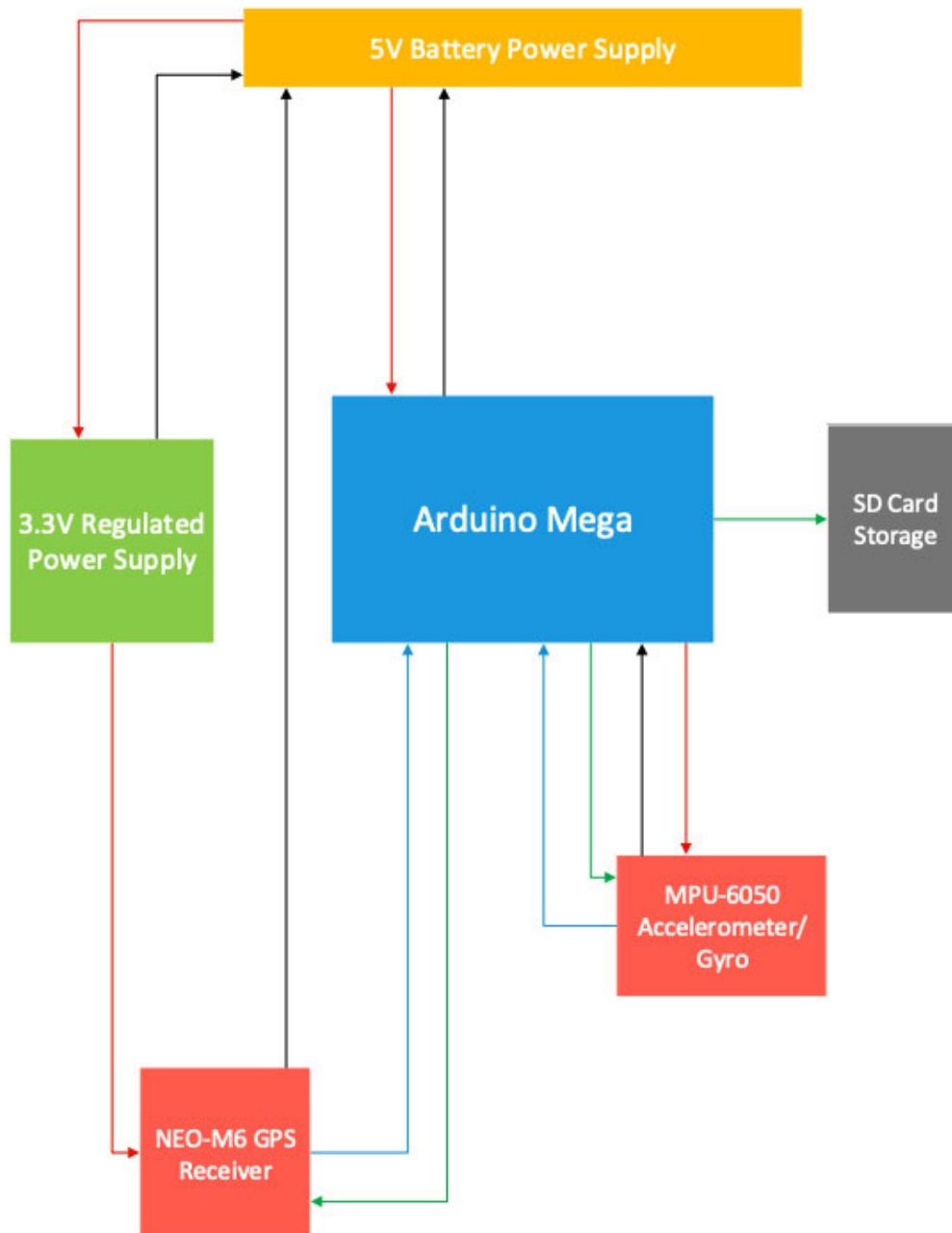


FIGURE 25: ELECTRONIC LAYOUT OF TEST DEVICE - INITIAL DESIGN

A LM317 Voltage regulator was able to regulate the 5V battery pack with one side of the breadboard set up as a 5V rail and the other a 3.3V rail as shown in Figure 27.

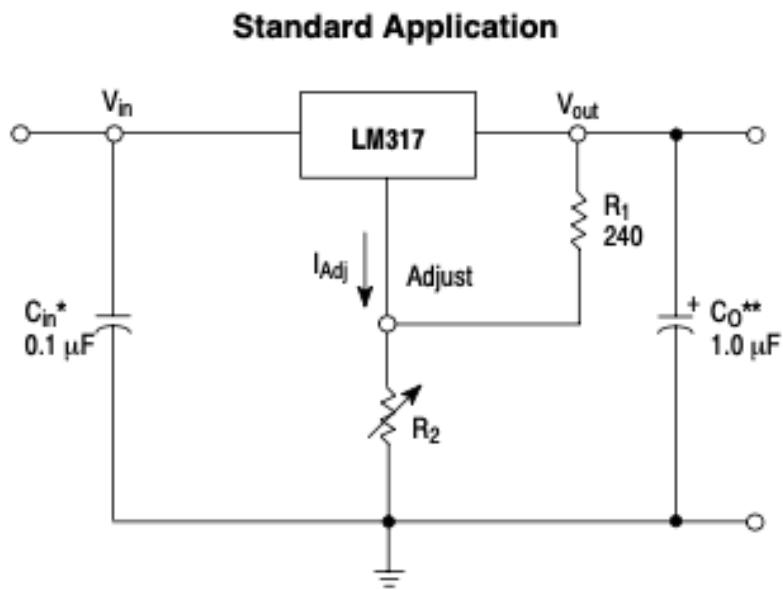


FIGURE 26: LM317 VOLTAGE REGULATOR CIRCUIT, (LM317 2002)

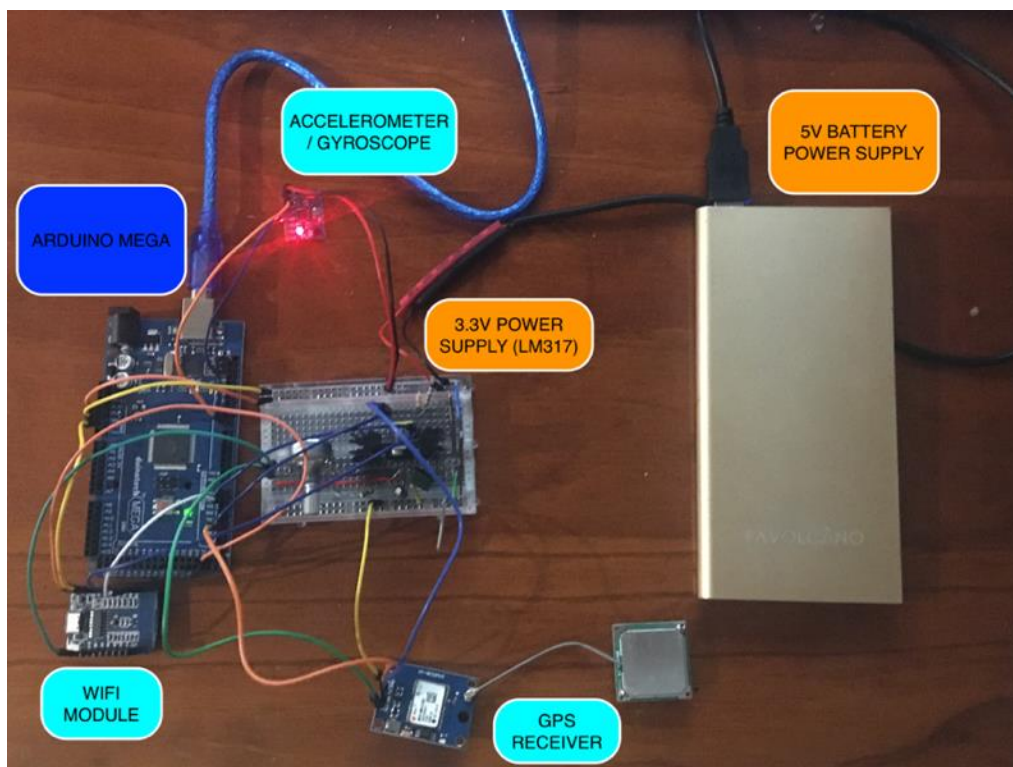
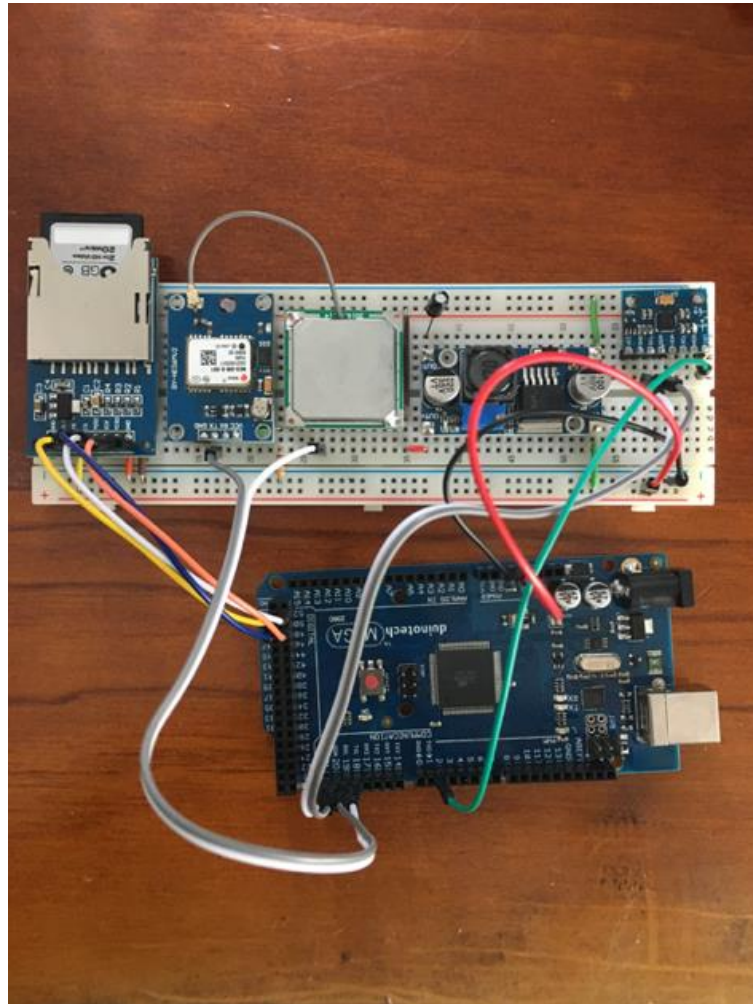


FIGURE 27: PROTOTYPE TESTING PACKAGE (TOP VIEW)

The LM317 was later replaced with an inexpensive chipset voltage regulator, the Wifi module was removed and an SD Card integrated into the design as shown below in Figure 28.



*FIGURE 28: STREAMLINED ELECTRONICS LAYOUT (MK2)*

The testing program split the data gathering into two parts: tracking GPS and then measuring wheel counts and acceleration. This again modified the hardware design with the input sensors being powered directly from the Arduino and the introduction of a hall effect sensor module as shown in Figure 29. Connections in the MK3 and MK4 versions were soldered as required and individual boards were secured to a wafer board so that they could not move.

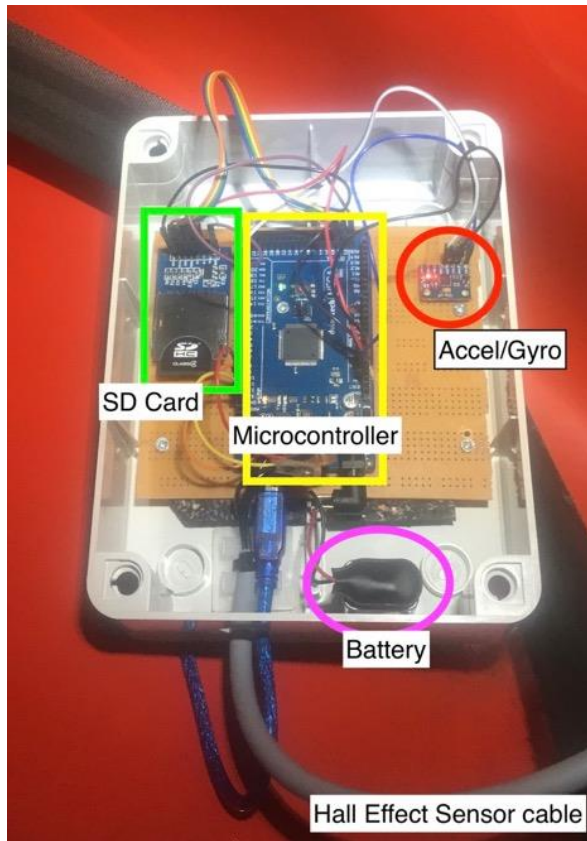


FIGURE 29: ELECTRONIC TESTING SETUP – NO GPS (MK3)



FIGURE 30: ELECTRONIC TESTING SETUP W/ GPS (MK4)

A magnet was glued to a shelled section of the wheel hub and a hall effect sensor was mounted on a bent, Aluminium bar secured to the road wheel pin's castellated nut as shown below in Figure 31.



*FIGURE 31: HALL EFFECT SENSOR MOUNTED TO WHEEL CARRIER*

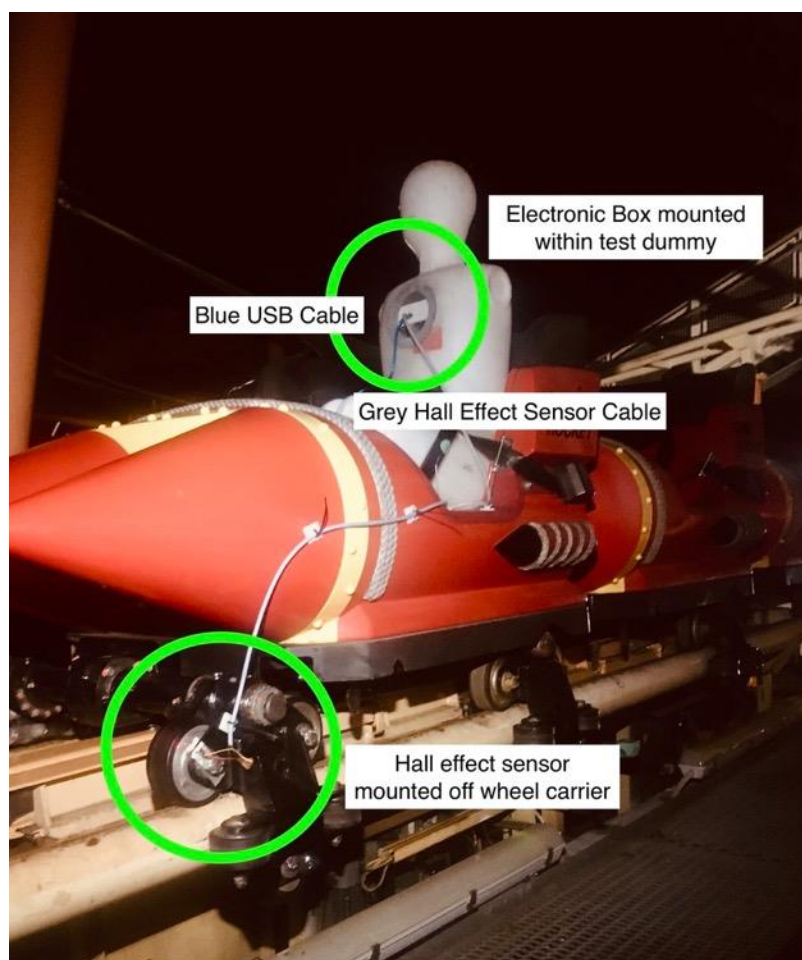
This particular mounting was quite challenging due to the small clearances between the road wheel and the wheel carrier and the reading range of the hall effect sensor being constrained to <math><12\text{mm}</math>.

#### 4.2.2. Packaging and Mounting

As the electronic equipment would be subjected to considerable g-forces for each cycle and realizing that a significant number of cycles would need to be completed to gather sufficient data, the test box needed to be both shock absorbent and secure. As shown in Figure 30 a hard, plastic electrical box was used and lined with thick, rubber matting. The

wafer mounting that held all the electronic equipment was screwed down onto this matting at four points.

A 9V battery was glued to the side wall and a hole was cut in the front of the box to create an opening for the hall effect sensor cable and the Serial USB cable to pass through shown below in Figure 31. Also shown is the test dummy properly restrained in the vehicle a seat belt and a lap bar, this set up targets the effects of acceleration on the human body. The electrical box was placed on the steel angle plate shown previously in Figure 25 and secured with plastic cable ties. A large hole was cut into the back of the dummy to accommodate this.



*FIGURE 31: TEST DUMMY MOUNTED DURING TESTING*



## 4.3. Software

### 4.3.1. General Overview

The software was developed using the Arduino IDE, it was tested for functionality in my home workshop and then reviewed continuously under the guidance of Prof. John Billingsley. It was further adapted after each night of testing as findings from the data sets revealed inherent weaknesses in the code which was able to be manipulated to better serve the purpose of the project. An example of this is the changing of the code structure for hall effect count vs acceleration to allow more acceleration readings without disturbing the count readings.

Two different programs were used to gather data, the first focused on plotting acceleration/gyro data against both time a hall effect sensor which read wheel count revolutions, effectively segmenting the track into equal length sections. The second captured the GPS location of the vehicle against time, unfortunately an effective code that matched GPS data with wheel counts was unable to be produced.

#### 4.3.1.1. Setup

Various pre-existing, open-source libraries were used to supplement the framework of the code and allow the various sensors to easily integrate with the Arduino. These included:

- Software Serial.h – Serial communication from sensors to Arduino using IO pins
- SD.h – Writing data to SD Card
- TinyGPS++ - Reading NMEA sentences from u-blox modules
- I2Cdev.h – communication with accelerometer/gyro
- MPU6050.h – accelerometer/gyro library with specific functions
- Wire.h - communication with accelerometer/gyro

Serial baud rate was set to 115200 and GPS baud rate set to 9600 (default for u-blox sensors). The program checked to see if a “ZMData##” File existed and if so, incremented a counter until a new one was created. This was especially useful completing multiple cycles that needed to be recorded and analysed separately. The programs would first check that an SD card was present and the GPS or accelerometer was active before creating a file, writing headers to the file and closing it before starting the main loop.

### 4.3.1.2. Storage

Upon each reset or upload of the program a new file would be created on a SD Card connected to the Arduino. This was achieved using the SD.h library and ensuring the pins were wired correctly as per Figure 32. A 3.3V or 5V supply from the Arduino was needed and both GND pins had to be connected for stability (only 1 GND pin shown in figure). The remaining four SPI pins were connected specifically to suit the Arduino Mega.

To write to the file it first had to be opened at the start of the loop before having each sensor print data to it and then finally closed at the end of the sequence before looping back to the start of the main program.

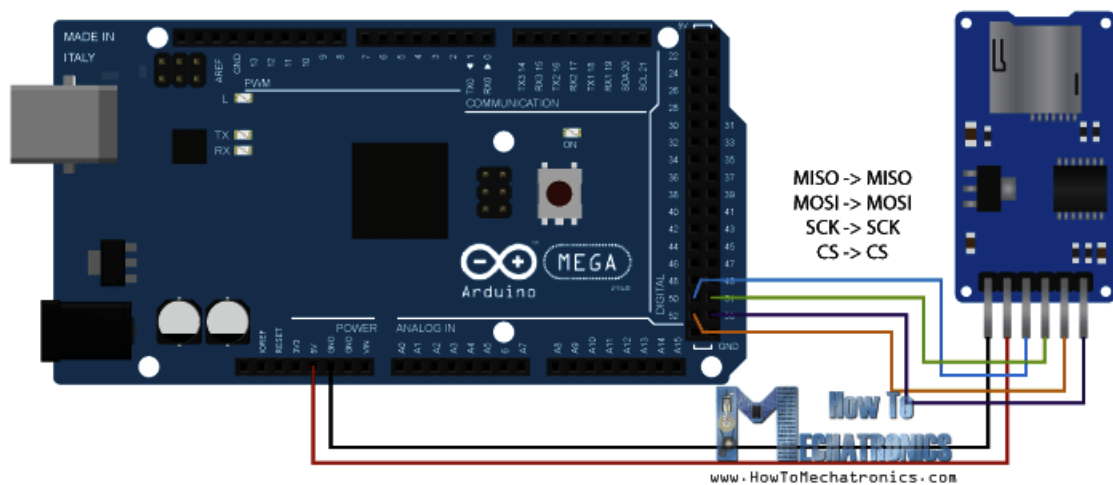


FIGURE 32: SD CARD PIN CONNECTIONS FOR ARDUINO MEGA, (HOW TO MECHATRONICS 2020)

## 4.3.2. Gathering Data

### 4.3.2.1. GNSS

The main loop was relatively simple and based off Mikal Hart's TinyGPS++ Library. This library works with NMEA sentences which were able to be consistently updated every 500ms. The code first checked that serial communication was available and then checked if GPS was being sent through the software serial connection. If no characters were sent within five seconds the program would stop and display "No GPS detected: Check wiring" on the Serial Monitor which assisted in troubleshooting.

If everything was working as intended the file on the SD Card would open, a time value would be written to it using the millis() function followed by the Latitude and Longitude

in degrees to a precision (note: not accuracy) of 8 decimal places (cm). If the device was unable to locate enough satellites the latitude and longitude values would be recorded as “INVALID”.

#### *4.3.2.2. Accelerometer and Gyroscope*

To ensure accelerations were recorded with a high enough sampling rate a function called `AccelGyro()` was added to all conditions within the main loop. This meant that no matter what state the sensor was in acceleration data would be captured. The `AccelGyro()` function called on `getmotion6(ax, ...)` from the `MPU6050.h` library to read the six different acceleration and gyro values, then printed those values to both the file in `.csv` format and the Serial port for troubleshooting.

Comparing time values using the `millis()` function showed that the time taken to read these six values and write them to file and Serial was approx. 4-5ms. The average loop time increased to 6-7ms having to then and close the file on each loop so that the data would write however this still gave a sampling rate of 150Hz. This did not meet the requirements of the testing procedure detailed in Section 5.2 however as these tests aimed at developing data gathering techniques as opposed to gathering acceleration data for safety or compliance this was not deemed to be an issue.

#### *4.3.2.3. Hall effect Sensor*

This part of the code effectively counted a single rotation of one of the vehicle’s road wheels when the magnet glued to the wheel hub passed by the hall effect sensor. The code read the sensor as a digital input and then reversed its state so that it made logical sense, i.e. when flagged the reading was normally 0 which was counterintuitive.

To ensure that only one increment was applied on each rotation the code was structured using a variable called “`sensorState`” that saved that status of the hall effect sensor, i.e. high or low and another variable called “`countBlock`” which was set true or false. Together these variables were tested in a nested ‘if’ statement within another ‘if’ statement to create a scenario where the counter would be blocked from incrementing until the hall effect sensor had registered low. This is described in pseudocode below:

```
if (sensorState = high)
  if (countBlock = false)
    increment counter
    countBlock = true
else
  countBlock = false
```

### 4.3.3. Digital filtering

#### *4.4.3.1. Pre filtering and anti-aliasing*

Unlike some of the more sophisticated accelerometer equipment used by VRTP the software setup for this project did not employ any pre-filtering or anti-aliasing code. Consideration was given to taking the maximum directional values of the accelerations between wheel counts but at this stage in the research this technique was not employed. NMEA sentences were automatically error corrected from the signal end, the only filtering conducted was to set the receiver signal rate and turn off any unwanted messages.

#### *4.4.3.2. Post filtering*

Post filtering was limited to taking the averages of multiple acceleration readings from individual wheel counts. The NMEA sentences gathered from the GPS module were left unchanged and input straight into Google Earth.

# 5. Experimental Design

## 5.1. Collection of Data

After designing a test model which can capture the data required, it must be set up in accordance with ASTM F2137 – 16 to ensure repeatability, reliability and utility of the test results. This standard covers the testing and set up of transducers in general and will apply to all sensors used in the experiments for this project. It also specifies setup requirements and performance specifications for accelerometer measurement which has been developed into a procedure template and is detailed in Section 5.2. (ASTM International 2016).

## 5.2. Testing Procedure

### 1.1 Hardware and Software Setup

#### 1.1.1 G force Input Range

- Set accelerometer to full scale resolution min. +/-10g (SARC-EN 13814)

#### 1.1.2 Calibration Configuration

Filtering - SARC-EN 13814 requires a low pass filter which adheres to CFC20 for each channel

- $F_L = 0\text{Hz}$
- $F_H = /> 20\text{Hz}$
- $F_N = /> 33.4\text{Hz}$
- $F_S = /> 240\text{Hz}$
- Set Low Pass Filter Cut-off to above 33.4Hz and do not use High Pass Filter

#### 1.1.2 Calibration Configuration

- Offset/drift is to be calculated using the field test calibration and marginalized in the program
- Units are G's (acceleration)

#### 1.1.3 Sample Rate


- Set to highest frequency reasonably possible to prevent aliasing, minimum 240Hz


## **1.2 Ride Setup**

- 1.2.1** Reference calibration must be documented annually on test equipment, maximum error 1.5% of data channel full scale (10g full scale – 0.15g maximum error)
- 1.2.2** Field calibration test must be undertaken before any data is recorded. A 2g “roll-over” test is sufficient to test all 3 axis channels which is achieved by aligning each axis with gravity in both the positive and negative direction
- 1.2.3** Zero bias must be accounted for in each data channel (x,y,z) by compensating in the Software set-up program
- 1.2.4** Ride must be operated for a full 3 cycles BEFORE data collection commences to ensure normal operating conditions are represented
- 1.2.5** Ride cycle times (as recorded on Operator’s Control Panel) shall have a variability of <5% and must be included in the test report. If time is not recorded on control panel, 1.2.4 shall be sufficient to begin testing
- 1.2.6** Accelerometer is to be secured to test dummy and then harnessed securely to the amusement ride/device
- 1.2.7** All test documentation must be completed, and multiple test runs are to be recorded to ensure test data acquired is accurate and repeatable


## 5.3. Inspection Template

TABLE 6: INSPECTION TEMPLATE FOR RECORDING ACCELERATION DATA

USQ RESEARCH PROJECT				
Student: Zachary Montgomery	Supervisor: John Billingsley	Date of testing:	Page 1 of 3	
OPERATIONS		SIGN OFF (Initial each task)		COMMENTS
TASK	DESCRIPTION			
	<b>Test Documentation</b> a) Ride/device name b) Serial number c) Location d) Test date and time e) Ambient Temperature f) Humidity g) Wind Conditions h) Accelerometer model and serial no. i) Brake settings on ride (if applicable)	a)	b)	
		c)	d)	
		e)	f)	
		g)	h)	
		i)		
<b>Hardware and Software Set up</b>				
10	<b>Confirm hardware and software complies with ASTM F2137 – 16.</b>  <i>SARC-EN 13814 requires CFC20 which is achieved by:</i>  Set resolution +/-10g Set LPF Cutoff to >33.4Hz Leave HPF Disabled Set Sample rate to >240Hz			

USQ RESEARCH PROJECT				
Student: Zachary Montgomery	Supervisor: John Billingsley	Date of testing:	Page 2 of 3	
20	<p><b>Ensure that accelerometer has been calibrated and drift corrected through software.</b></p> <p><i>Reference calibration must be documented annually on test equipment, max error 1.5% of data channel full scale (10g full scale – 0.15g error max). Do not use equipment if calibration is not in date.</i></p> <p><b>Conduct a 2g “roll-over” test on each individual axis to ensure each axis is functioning correctly.</b></p> <p><i>Must be undertaken before any data is recorded! Line up x-axis so that gravity acts upon it. In the software program plotter, the chosen axis should read 1g. Roll the node over 180° and plotter should now read -1g. Repeat for y- and z-axes.</i></p> <p>(a) x-axis calibrated (b) y-axis calibrated (c) z-axis calibrated</p> <p>If significant error occurs (&gt;0.15g) then device shall not be used and must be recalibrated.</p>	<p><i><b>Error reported:</b></i></p> <p>(a) x-axis calibrated _____g</p> <p>(b) y-axis calibrated _____g</p> <p>(c) z-axis calibrated _____g</p>		
<b>Ride set up</b>				
30	<p><b>a) Secure test dummy to ride ensuring that inadvertent release is not possible.</b></p> <p><b>b) Record equipment location on train, this is to be kept consistent with previous recordings.</b></p>			



USQ RESEARCH PROJECT				
Student: Zachary Montgomery	Supervisor: John Billingsley	Date of testing:	Page 3 of 3	
40	Mount main unit housing the accelerometer device inside the test dummy. Ensure the x-axis is facing forward in the direction of travel. Mounting should be accurate to within 5 degrees of the Patron Coordinate System (Figure 4).			
50	Mount sensor on axle/wheel hub and ensure it is counting.			
60	Ride shall be operated for a full 3 cycles BEFORE data collection commences  <i>If possible, check that ride cycle times (as recorded on Operator's Control Panel) have a variability of &lt;5%</i>			
70	Ensure all sensors are active and recording, i.e. microcontroller is switched ON.  <i>Multiple runs will be required to establish an accurate GPS map</i>			
80	Results to be post filtered and analysed paying particular attention to:  i) Overview (x,y,z) ii) Points of interest (high g zones) iii) High resolution at points of interest iv) Comparative analysis of previous data to detect trends/possible changes.			

## 6. Experimental Programme

### 6.1. Programme Overview

Testing was conducted over a two-week period and focused on gathering acceleration data and GPS data separately. The measurement sensors were placed inside a test dummy which was restrained within the first train carriage for consistency. The same vehicle was used throughout and similar environmental conditions were seen across all nights of testing. The program developed over each evening with the learnings from the previous night improving the quality of testing.

### 6.2. Test 1 – 24/9/2020

Much preparation had gone into this initial test which was focused on performing a large volume of test runs gathering data for GPS only. VRTP maintenance staff assisted with packaging and shock proofing the electronics and then modifying the test dummy to allow the test box to be placed inside. The u-blox NEO M6 GPS module was being used for this test however it was unable to record data. Later it was found that the antenna's connecting wire had been damaged and was subsequently substituted with the NEO M8. The M8 is a technically superior device and has its antenna integrated onto its board.

### 6.3. Test 2 – 25/9/2020

This test ran much smoother with the test dummy ready to go and the NEO M8 GPS Module receiving signals from multiple satellites. The code used the TinyGPS++ library to read and parse NMEA messages from the u-blox sensor to the SD Card, it logged the data at 1Hz and included GPS time, latitude, longitude and altitude.

It was difficult to get a signal within the station, most likely because it was under cover which interfered with the satellite signals. This was overcome by parking the train outside of the station and using a Satellite Tracking program from TinyGPS++ to monitor the satellites. The module picked up a strong signal rather quickly and testing was commenced soon after.

Successive runs were completed with no pause or reset in the station to separate them, the intent being to minimize the chance of losing the signal as the train was parked in the station. This proved to be a mistake as it was quite difficult to separate the runs when

analysing the data. The results yielded little correlation with the coaster's track as shown below in Figure 33, but all plot points were well within the ride's envelope which was encouraging.



*FIGURE 33: GPS PLOT TAKEN DURING TEST 2*

#### 6.4. Test 3 – 4/10/2020

While further work was being conducted to increase the accuracy of the GPS data, Test 3 was aimed at gathering video data of the ride and designing a suitable mount for the hall effect sensor and considering how to fix magnets to the wheel hub. As shown in Figure 34, the clearance between the road wheel hubs and the wheel carrier was minimal which meant that the magnets could not protrude over the side of the hub. Fortunately, the hubs have a shelled section illustrated in Figure 35 and the magnets were able to be filed down and glued to sit within this.



*FIGURE 34: TIGHT CLEARANCE BETWEEN ROAD WHEEL AND WHEEL CARRIER*



*FIGURE 35: MAGNETS TO SIT WITHIN WHEEL HUB RECESS*

## 6.5. Test 4 – 6/10/2020

This test was focused on gathering acceleration data at specific wheel counts. The code was written as a set of four subsequent if statements and while it worked it to a point it was obvious after reviewing the results when the train arrived back in the station that wheel counts were being missed. This was most likely due to the program structure cycling through the 'if' statements at a rate slower than the window of the magnets were able to flag.

On the night, the wheel hub started with four equally spaced magnets glued to it before they were removed so that only one remained. This did help with increasing the wheel count but not to the expected level, raising them from ~60 counts to 120 counts per cycle. Time had not been recorded to compare the data sets which was an oversight that was corrected before the next test.

## 6.6. Test 5 – 7/10/2020

This test again focused on gathering acceleration data at specific wheel counts. The code was significantly altered to narrow the window of opportunity for a missed wheel count and time values were captured on each file line using the millis() function. The results were immediate with the wheel count jumping from 120 counts the previous night to ~360 counts quite consistently for each run. The time data also proved invaluable in being able to cross check cycle times and understand the frequency of click counts and thus acceleration/gyro sampling rates.

## 6.7. Test 6 – 8/10/2020

Having successfully gathered acceleration/gyro vs wheel count (and time) data, the objective of this test was to record GPS data to build a 3D map of the track. Much work had gone into improving the precision and accuracy of the data by changes to the data gathering program and the setup of the NEO M8 GPS module.

U-blox NEO GPS Modules are supported with u-center, a Windows-based application that can manipulate the settings of individuals, monitor both NMEA packets and proprietary UBX-Binary packets and display/record signal information through an interactive GUI as illustrated in Figure 36.

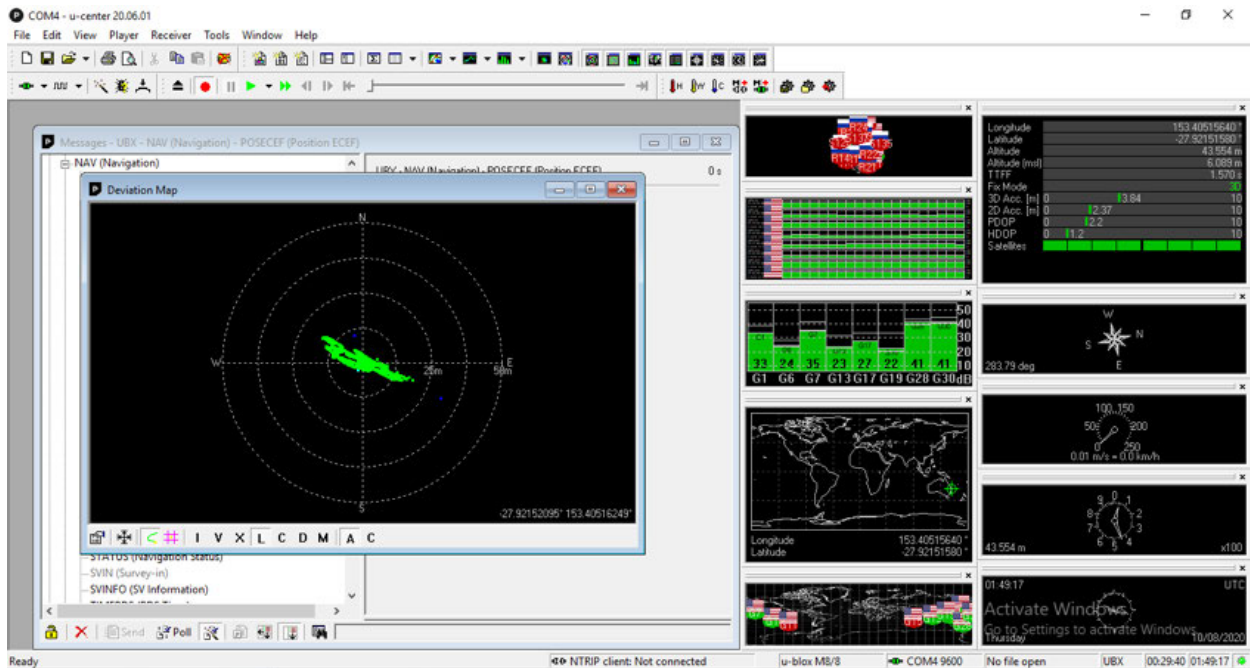
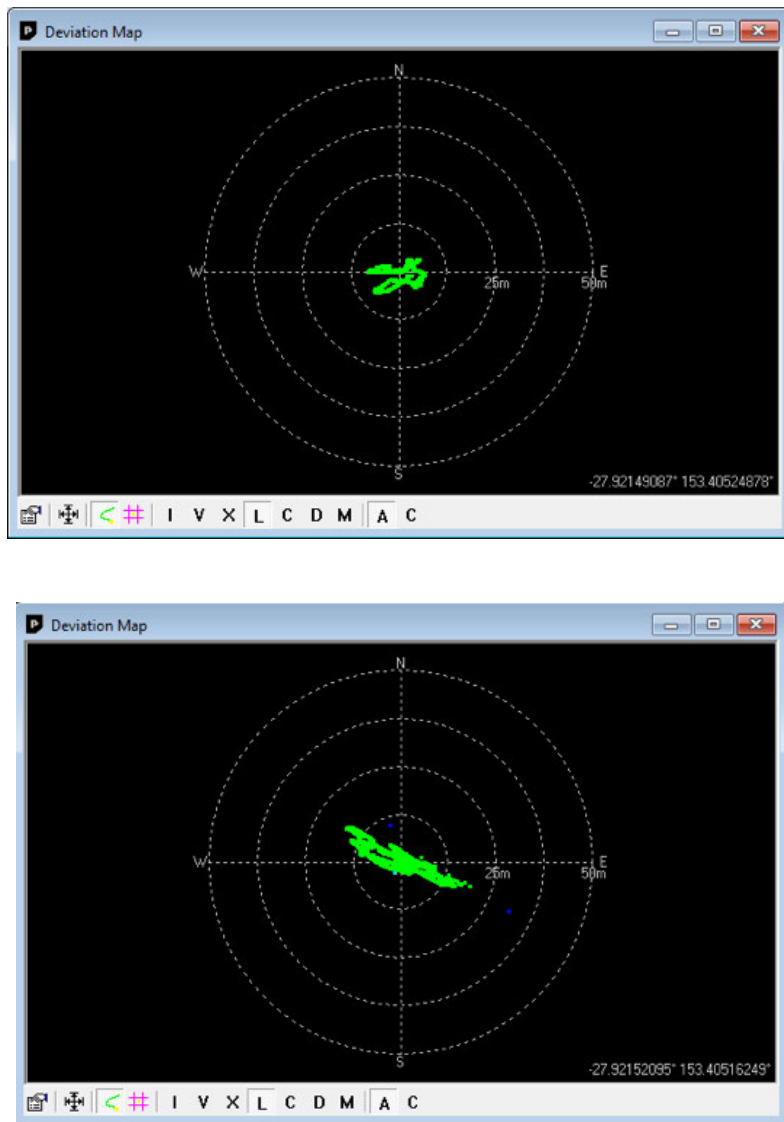


FIGURE 36: UBLOX U-CENTER GNSS GRAPHICAL INTERFACE

By connecting the NEO M8 directly into a laptop via USB, u-center was able to communicate with the device. This in itself became a testing playground with the intention to compare accuracy, precision and sampling rate differences between NMEA and UBX-Binary protocols.

The most important of these would be precision and both NMEA and UBX were capable of providing latitude and longitude to 8 decimal places – equivalent to cm level precision. This is not to be confused with accuracy which would become the determining factor. As highlighted in Figure 37 the UBX protocol was prone to drift, likely because the smaller binary packet messages do not contain any error correcting information unlike the standard NMEA messages. (NEO M8 2020).

The downside to this is that UBX messages were configurable to a stable sampling rate of 4Hz whereas the NMEA messages were restricted to 2Hz after limiting incoming messages. Default settings of the device were to receive all NMEA messages and no UBX messages at a rate of 1Hz only. The decision was made that accuracy of data for this particular application was critical and so the NEO M8 was configured to read only the \$GPGGA NMEA message at a rate of 2Hz.



*FIGURE 37: GPS MESSAGE ACCURACY – NMEA (TOP) VS UBX (BOTTOM)*

Having set up the u-blox for NMEA, the code used to gather data was based on the TinyGPS++ library along with the millis() function to record time intervals. The code was modified to record cm level precision and for this test only focused on latitude and longitude as the accuracy of altitude was deemed to be too unreliable.

This was the longest test and having captured over 50 ride cycles only four of these could be considered full captures. The testing rig experienced dropouts after 30-60 seconds since dispatch where data recording would stop. Ride cycle time is approx. 1min 30secs and upon re-entry to the station all connections were checked and the GPS Module was confirmed to be receiving over Arduino's Serial Monitor. It is likely that the high g-forces on the ride were disrupting the connection between the Arduino and the SD Card. This was disappointing as the intent was to capture multiple runs to be able to filter and average them to better match the track.

# 7. Results and Analysis

## 7.1. 3-D Acceleration versus Time

Data from testing was compared with those from historical records to evaluate the authenticity of the tests. It is noted that the sampling rate of the recorded data was insufficient to be properly filtered and is unlikely to be truly representative. Even so, the fore-aft (x-axis) shows some level of correlation with the exception of the sharp spike at about 34 seconds into the ride cycle which is caused by the braking zone.

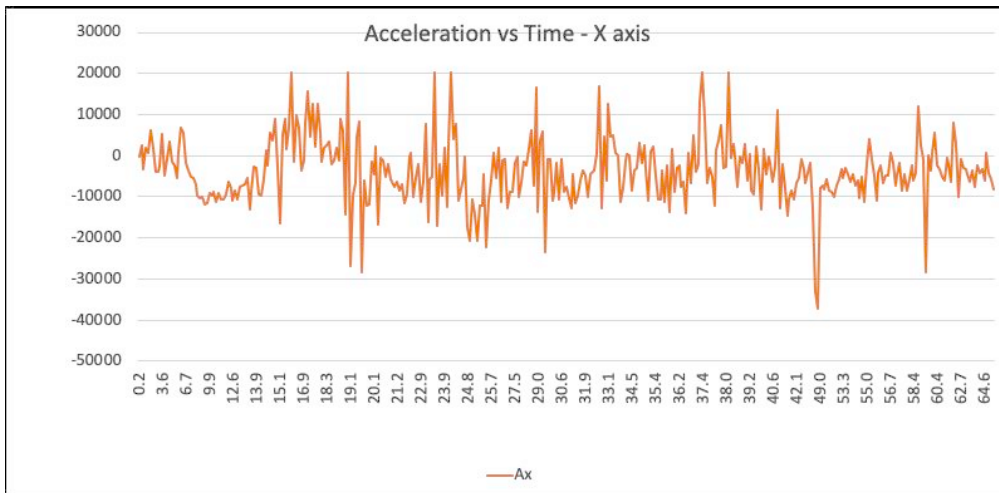


FIGURE 38: PROJECT ACCELERATION DATA X-AXIS 7/10/2020

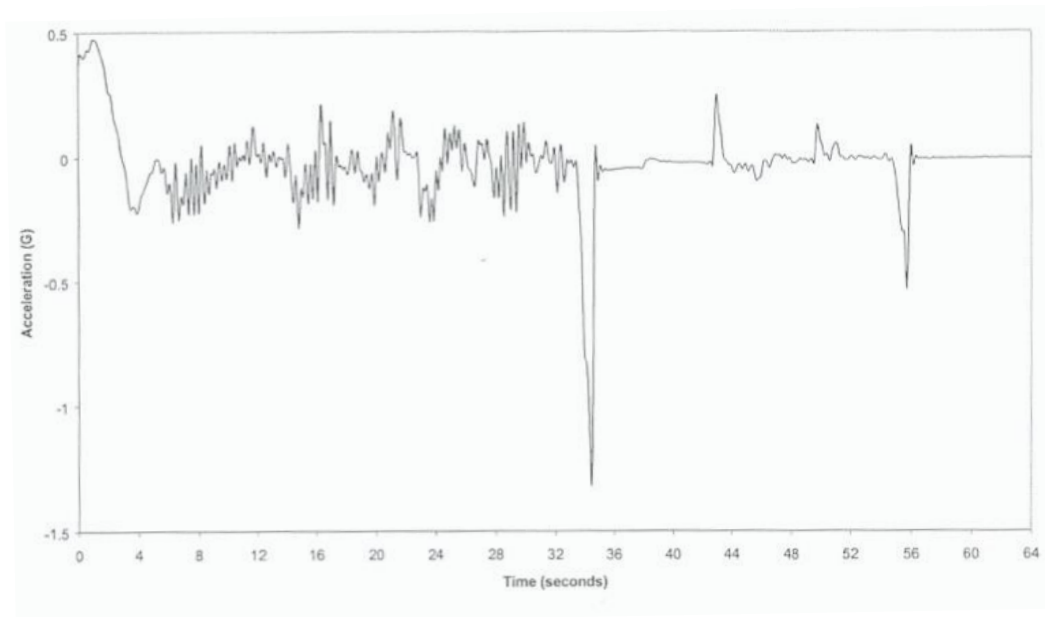


FIGURE 39: HISTORICAL ACCELERATION DATA X-AXIS



Comparisons between the vertical z-axis data of the project's tests and a historical report yielded a poor correlation and what can only be described as an unrepresentative result. The test results were spiking heavily and seemed to clip in the positive, downwards direction which was most likely due to excessive vibration creating noise.

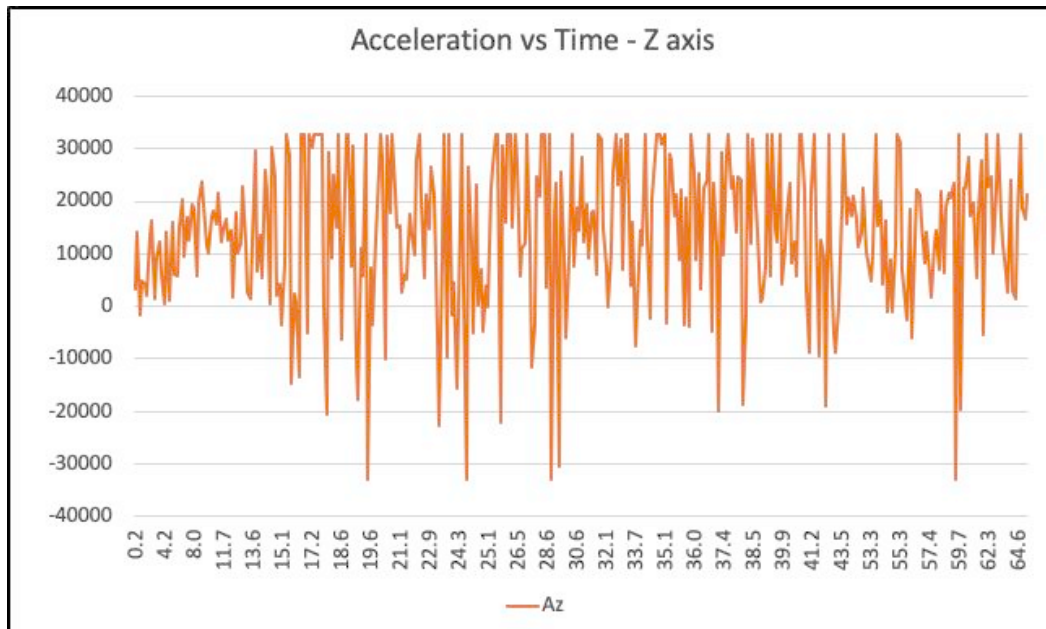


FIGURE 40: PROJECT ACCELERATION DATA Z-AXIS 7/10/2020

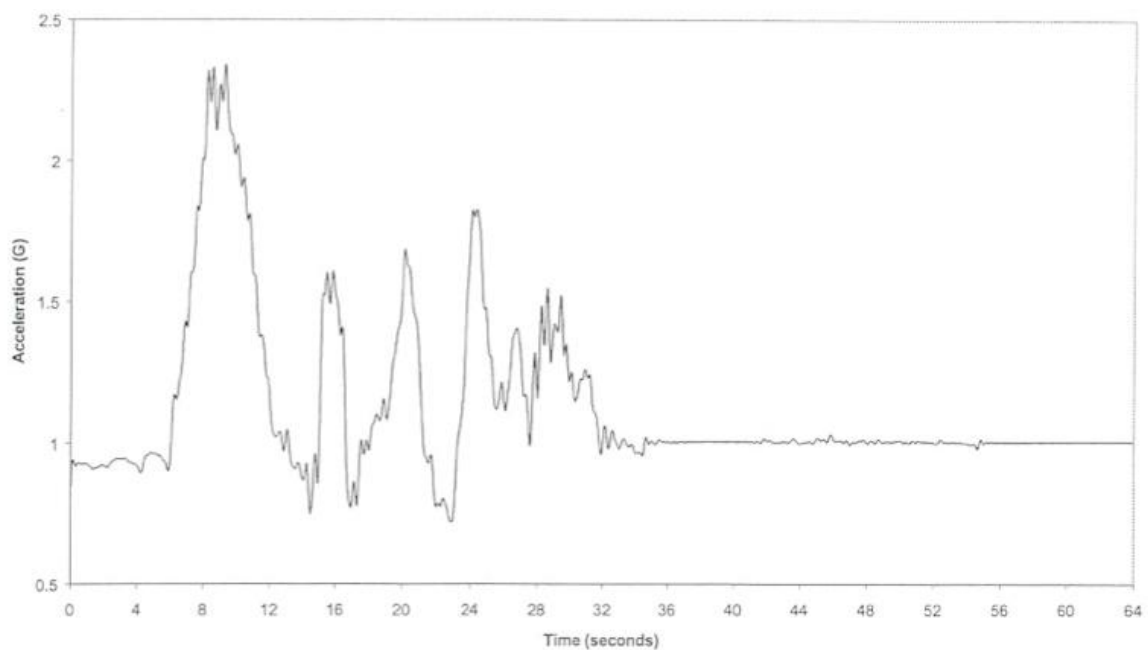
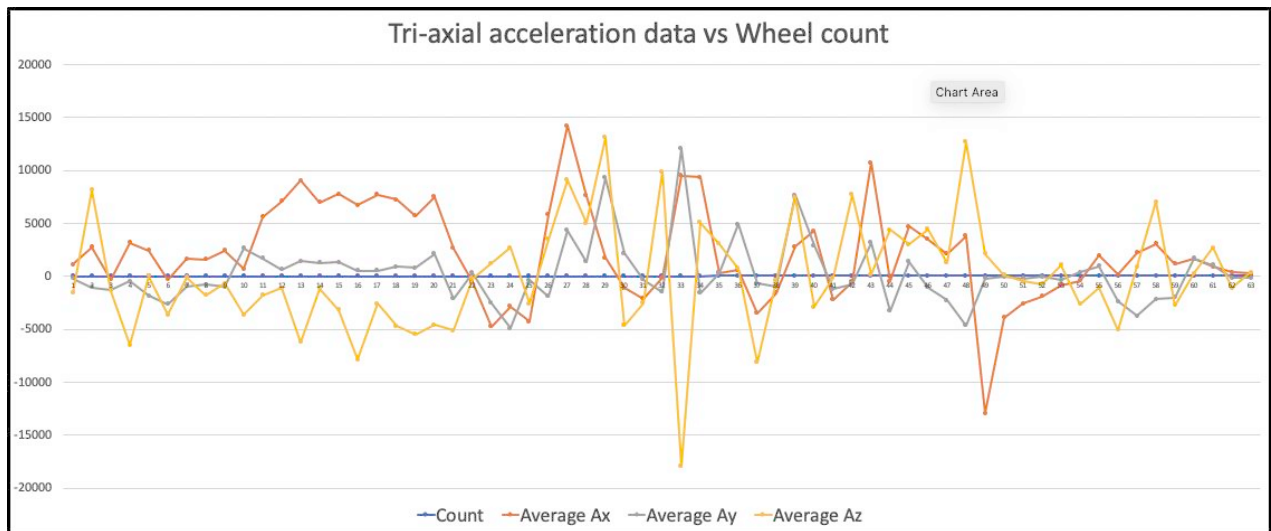


FIGURE 41: HISTORICAL ACCELERATION DATA Z-AXIS

## 7.2. 3-D Acceleration versus Distance

The first test runs, a sample of which is shown below in Figure 42 were not representative. The wheel hub was embedded with four magnets with the intent of increasing precision but after watching the train on track the hall effect sensor appeared to be high. It was hypothesized that the wheel was spinning too quickly and the sensor was unable to register that the magnetic field had gone low. As such, three magnets were removed so that only one was being read per revolution.



*FIGURE 42: ACCELERATION VS WHEEL COUNT USING 4 MAGNETS RECORDED 6/10/2020*

This produced an immediate result with wheel counts doubling to ~120, while this was an improvement it was still not sufficient as the predicted wheel count for one ride cycle was estimated at over 700 counts. During test runs it was clear to see the hall effect sensor pulsating and so the problem was most likely in the code.

After disseminating the data and averaging the recorded values for each wheel click it can be seen from Figure 43 that there is some resemblance to the baseline historical data. All three axes are clearly distinguishable and operating in a range that was to be expected. The lateral forces (grey) are consistent and relatively low. The fore-aft forces (orange) produce the characteristic negative gradient at the 35-count mark – seen at the start of the historical graph and also show the high braking peak point at the 97-count mark. The down forces (yellow) seem to wander about the 10000 – 15000 range as would be expected from the effects of gravity as the train navigates its way around the undulating track.

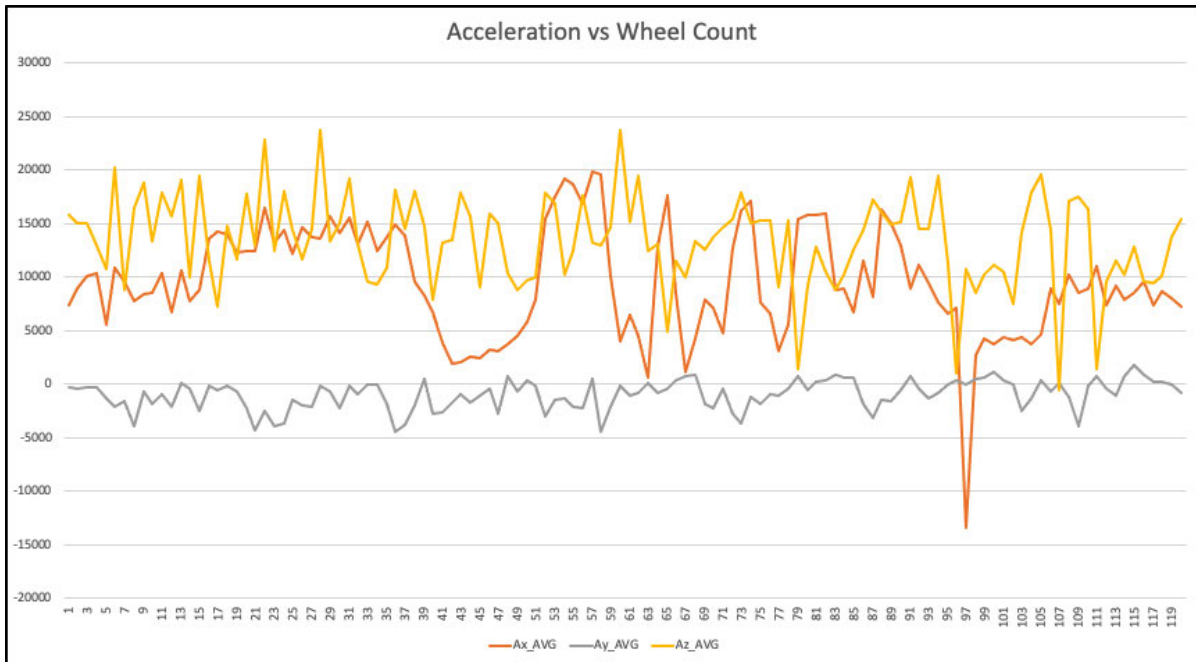


FIGURE 43: ACCELERATION VS WHEEL COUNT USING 1 MAGNET RECORDED 6/10/2020

In an attempt to produce a data set with a higher click count the program code was rewritten under the guidance of Prof. John Billingsley. The results showed an immediate increase in recorded wheel counts jumping to a consistent ~370 counts per cycle. The data appears much noisier though in its unfiltered form and there was unexpected clipping on the z-axis. This was most likely due to vibration from the equipment working its way loose in the test box. Even with this increased wheel count the number achieved was barely half of what was expected.

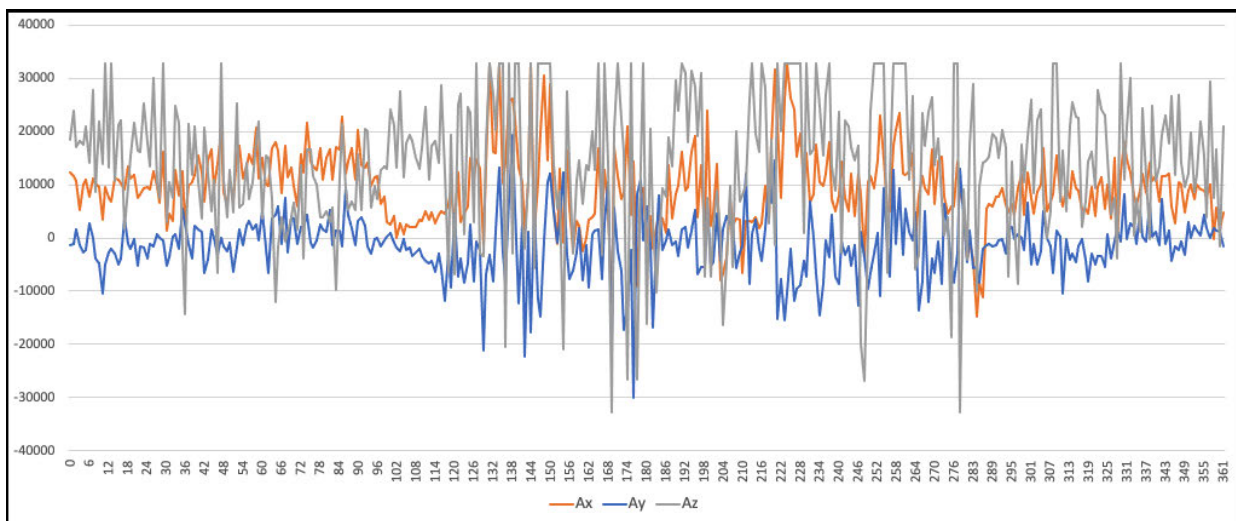
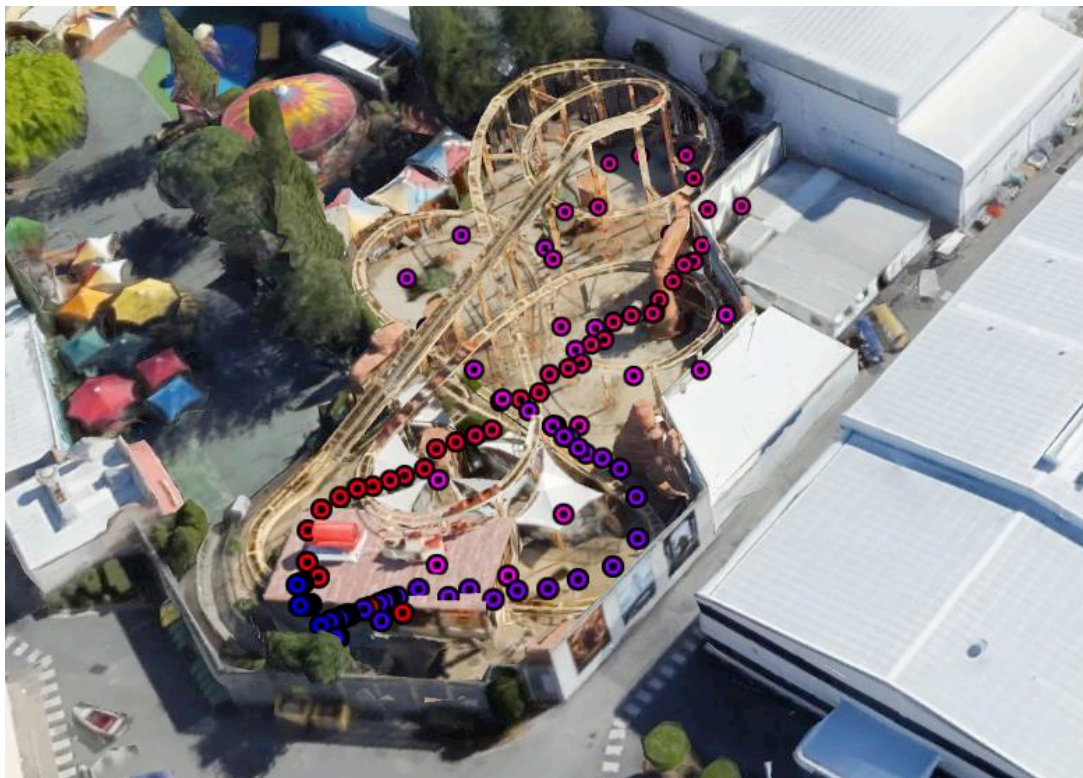


FIGURE 44: ACCELERATION VS WHEEL COUNT USING NEW CODE 7/10/2020

### 7.3. Spatial Position Map versus Distance

The first steps were taken to producing a spatial position map as displayed in Figure 45. The GPS plot used only latitude and longitude which limits its accuracy, this was decided as the performance of the GPS module during pretesting using u-center was showing significant altitude drift. Encouragingly this limitation can be inferred as the plot appears to follow the track almost like a shadow. The plot has been colored red (start) to blue (finish) in the direction of travel to assist in making sense of the dots.



*FIGURE 45: GPS TRACE LAT & LON ONLY 8/10/2020*

The signal points are bunched on the lift, brake and station sections and spread out over the high speed, radial parts of the track. This was a result of using NMEA sentences for better accuracy which resulted in a lower signal rate of 2Hz. The data seems accurate enough to attempt modelling a track in software however the distance between plot points will make it more difficult to interpolate the rounded sections of track. A hand drawn interpolation is illustrated in Figure 46 which accentuates the need for a faster sampling rate or an amalgamation of several cycles.



*FIGURE 46: GPS TRACE WITH HAND DRAWN CONNECTIONS 8/10/2020*

## 8. Conclusions

The testing was conducted with inexpensive equipment that had limited capabilities, nevertheless the results were encouraging. For acceleration data, higher sampling rates which can be filtered and better vibration dampening of the test box are both required to filter noise and produce more representative results. This was shown when taking the averages of multiple readings, the data began to reflect what had been historically recorded.

Effective counting of wheel revolution appears to be the best way forward to tying the map and the forces together. There were big improvements seen in this area both from the written code and the physicality of the sensor. Hall effect sensors most likely do not have a large enough window to see the magnet pass by which may have caused missed counts or errors. There was evidence to suggest this in the raw data first with variations in both the readings per count and then time intervals between counts.

The approximate track length is 335m and with each road wheel measuring 140-145mm the wheel counts which meant that total count would be over 700. After careful consideration the assumption that the road wheels maintain full roll on the track may have been incorrect. Whilst improvements can be made to avoid missing counts such as using optical sensors and painting the wheel into quadrants, there is also the possibility that the road wheel is not in contact with the track especially when the train approaches a negative g-acceleration. If this is the case then an alternative solution such as a separate spring loaded wheel attached to the train which rolls along the track could be investigated.

Finally, the GPS 'map' has shown promise. Configuring the module produced much better results than what the default settings originally gave. Investment in a superior model that offers a faster sampling rate and better accuracy could now be justified.

## 9. Ideas for Potential Future Works

The work here could be continued best by first concentrating on the design weaknesses unveiled in the testing – GPS mapping and wheel count capture. The most important of which is establishing the wheel count so that both the accelerometer and GPS can be measured at precise counts.

Equipment for industry acceptable acceleration testing is already available and has been used successfully outside the scope of this project. The techniques and knowledge gained from those tests will assist in future testing once wheel counts are able to be repeatedly recorded. Work in this area could involve painting the road wheel hubs with reflective and black paint to be distinguished by an optical sensor.

As part of ongoing Technical Assessments of all amusement devices within the parks, 3D rendered models of the track structures have been created. Upon successfully establishing wheel counts the data could be applied straight to these models bypassing the need for GPS mapping altogether.

Another technology that could be used to supplement or bypass the GPS map would be correlating acceleration data with video footage taken of the ride. This gives the added benefit of being able to ‘see’ the forces as they are recorded on the ride. Or rather than changing technologies, an investment in a higher end GPS module could provide the accurate map of the ride and would be an interesting project to set up and test.

Work could also be undertaken to use the gyroscopic data from the accelerometer to model and visualize the orientation of the train as it moves around the track, this would also provide more accurate readings of how the forces are interacting with the structural members.

# References

*About Mu* 2019, online article, CodeWithMu, viewed 27 October 2019, <<https://codewith.mu/en/about>>.

*Alcatel Pixi 4 (3.5) – Specifications* 2019, online article, Device Specifications, viewed 27 October 2019, <<https://www.devicespecifications.com/en/model/049e393b>>.

Amy, M-W 2016, *Dreamworld told to fix problems after check. (News)*, newspaper article, 26 November 2016, Fairfax Media Publications Pty Limited, viewed 26 October 2019, <<http://web.a.ebscohost.com.ezproxy.usq.edu.au/ehost/detail/detail?vid=2&sid=d5655440-4e0e-478e-9b3e-e5e5422fa6fa%40sessionmgr4008&bdata=JnNpdGU9ZWwhvc3QtbGl2ZQ%3d%3d#AN=DOC6SHY0EELN9I1Z2WZDA8&db=n5h>>.

*Arduino 1.8.12*, online article, Arduino, viewed 30 May 2020, <<https://www.arduino.cc/en/Main/Software>>.

*Arduino Mega 2560 Datasheet* Year Unknown, datasheet, RobotShop, viewed 27 October 2019, <<https://www.robotshop.com/media/files/pdf/arduinomega2560datasheet.pdf>>.

*Arduino SD Card and Logging to Excel Tutorial*, online article, How To Mechatronics, viewed 9 October 2020, <<https://howtomechatronics.com/tutorials/arduino/arduino-sd-card-data-logging-excel-tutorial/>>.

*Arduino Uno* Year Unknown, datasheet, Farnell, viewed 27 October 2019, <<https://www.farnell.com/datasheets/1682209.pdf>>.

Australian Standards 2009, *Amusement rides and devices – Design and construction*, AS 3533.1 2009, Standards Australia Ltd, GPO Box 476, Sydney, NSW 2001.

ASTM International 2016, *Standard Practice for Measuring the Dynamic Characteristics of Amusement Rides and Devices*, ASTM F2137 – 16, ASTM International, 100 Barr Harbor Drive, PO Box C700, West Conshohocken, PA 19428-2959, United States.

Benedetti, U, Dermanis, A & Crespi, M 2017, 'On the feasibility to integrate low-cost MEMS accelerometers and GNSS receivers', *Advances in Space Research*, vol. 59, no. 11, pp. 2764-2778.

Billingsley, J 2006, *Essentials of Mechatronics*, 1<sup>st</sup> edition, John Wiley & Sons, Hoboken NJ.



*Bluetooth* 2019, Britannica Academic, online encyclopedia article, viewed 27 October 2019, <<https://academic-eb-com.ezproxy.usq.edu.au/levels/collegiate/article/Bluetooth/471609>>.

*Building a GPS System* Year Unknown, online article, Sparkfun, viewed 26 July 2020, <<https://www.sparkfun.com/gps#overview>>.

Corke, P 2017, *Robotics, Vision and Control*, 2<sup>nd</sup> edition, Springer International Publishing, Switzerland.

*Coriolis Effect* Year Unknown, online encyclopedic article, National Geographic, viewed 26 July 2020, <<https://www.nationalgeographic.org/encyclopedia/coriolis-effect>>.

Eriksson, U & Pendrill, A-M 2019, 'Up and down, light and heavy, fast and slow—but where?', *Physics Education*, vol. 54, no. 2, p. 025017.

Fernandez, A & Dang, D 2013, 'Chapter 8 - Analog: The Infinite Shades of Gray', in A Fernandez & D Dang (eds), *Getting Started with the MSP430 Launchpad*, Newnes, Oxford, pp. 81-125.

*GPS Trilateration*, 2019, digital image from online article, GISGeography, viewed 27 July 2020, <<https://gisgeography.com/gps-accuracy-hdop-pdop-gdop-multipath>>.

*GPS Basics* 2012, online article, Sparkfun, viewed 26 July 2020, <<https://learn.sparkfun.com/tutorials/gps-basics? ga=2.121257355.2097698259.1595723196-457674538.1595723196>>.

Grace, G (Minister for Education and Minister for Industrial Relations) 2019, *Tougher regulations for Queensland amusement rides*, media statement, 4 March 2019, Department of Industrial Relations, Queensland, viewed 26 October 2019, <<http://statements.qld.gov.au/Statement/2019/3/4/tougher-regulations-for-queensland-amusement-rides>>.

Groves, PD 2013, *Principles of GNSS, inertial, and multisensor integrated navigation systems*, Second edition. , Artech House, Boston.

*Interface ublox NEO-6M GPS Module with Arduino*, online article, Last Minute Engineers, viewed 17 August 2020, <<https://lastminuteengineers.com/neo6m-gps-arduino-tutorial/>>.

Kai-Wei, C, Cheng-An, L & Thanh-Trung, D 2014, 'The Performance Analysis of the Tactical Inertial Navigator Aided by Non-GPS Derived References', *Remote Sensing*, vol. 6, no. 12, pp. 12511-26.

LM317 2002, Datasheet, ON Semiconductor Components Industries, Colorado 80217 USA, viewed 11 May 2019, < <http://onsemi.com>>

Meet Android Studio 2019, online article, Android Studio, viewed 27 October 2019, <<https://developer.android.com/studio/intro>>.

Micro:bit: Hardware Description 2019, v1.5, online article, Micro:bit Educational Foundation, viewed 27 October 2019, <<https://tech.microbit.org/hardware/>>.

MPU-6050 2019, 3 Axis Gyroscope and Accelerometer Sensor Module for Arduino Projects, online article, Aus Electronics Direct, viewed 27 Oct 2019, <[https://www.auselectronicdirect.com.au/mpu-6050-3-axis-gyroscope-and-accelerometer-sensor?gclid=EAIAIqobChMIrvmPwMC75QIViZOPCh3b6wMFEAQYASABEgJhw\\_D\\_BwE](https://www.auselectronicdirect.com.au/mpu-6050-3-axis-gyroscope-and-accelerometer-sensor?gclid=EAIAIqobChMIrvmPwMC75QIViZOPCh3b6wMFEAQYASABEgJhw_D_BwE)>.

NEO-6 2019, u-blox6 GPS Modules, GPS.G6-HW-09005-E, Data Sheet, u-blox AG Zuercherstrasse 68 CH-8800 Thalwil Switzerland, viewed 27 October 2019, <[https://www.jaycar.com.au/medias/sys\\_master/images/9292330336286/XC3712-dataSheetMain.pdf](https://www.jaycar.com.au/medias/sys_master/images/9292330336286/XC3712-dataSheetMain.pdf)>.

NEO-M8 2020, u-blox M8 concurrent GNSS modules, Data Sheet, u-blox AG Zuercherstrasse 68 CH-8800 Thalwil Switzerland, viewed 30 May 2020, <[https://www.u-blox.com/sites/default/files/NEO-M8-FW3\\_DataSheet\\_%28UBX-15031086%29.pdf](https://www.u-blox.com/sites/default/files/NEO-M8-FW3_DataSheet_%28UBX-15031086%29.pdf)>.

Oshana, R (ed.) 2012, 'DSP in Embedded Systems: A Roadmap', *DSP for Embedded and Real-Time Systems*, Newnes, Oxford, pp. xxiii-xxxiv.

Rashid, RA & Yusoff, R 2006, 'Bluetooth Performance Analysis in Personal Area Network (PAN)', *2006 International RF and Microwave Conference*, pp. 393-7.

Read and Parse NMEA Data Directly From GPS Receiver, online article, MathWorks, viewed 17 August 2020, < <https://au.mathworks.com/help/fusion/examples/read-and-parse-nmea-data-directly-from-gps-receiver.html>>.

Rotary Encoder 2020, online catalogue, Jaycar Electronics, viewed 31 May 2020, < <https://www.jaycar.com.au/rotary-encoder-with-pushbutton/p/SR1230>>.

Stone, L 2019, *Queensland amusement parks ready for stricter safety regulations*, online article, 1 May 2019, Brisbane Times, viewed 26 October 2019, <<https://www.brisbanetimes.com.au/national/queensland/queensland-amusement-parks-ready-for-stricter-safety-regulations-20190501-p51izr.html>>.

*What is Arduino?* 2019, online article, Arduino, viewed 27 October 2019,  
<<https://www.arduino.cc/en/Guide/Introduction>>.

Wi-Fi 2019, *Britannica Academic*, online encyclopedia article, viewed 27 October 2019,  
<<https://academic-eb-com.ezproxy.usq.edu.au/levels/collegiate/article/Wi-Fi/443952>>.

*Wi-Fi Wireless Shield for Arduino Projects* 2019, online article, Aus Electronics Direct,  
viewed 27 October 2019, <[https://www.auselectronicdirect.com.au/wi-fi-wireless-shield-for-arduino-projects?gclid=EAIaIQobChMI\\_5mO55i75QIVICQrCh0poggcEAQYAiABEgIEtPD\\_BwE](https://www.auselectronicdirect.com.au/wi-fi-wireless-shield-for-arduino-projects?gclid=EAIaIQobChMI_5mO55i75QIVICQrCh0poggcEAQYAiABEgIEtPD_BwE)>.

*Wireless Bluetooth Module for Arduino Projects* 2019, online article, Aus Electronics Direct,  
viewed 27 Oct 2019, < [https://www.auselectronicdirect.com.au/wireless-bluetooth-4.0-module-for-arduino-projects?gclid=EAIaIQobChMIkoCv2My75QIV2RwrCh33XQaGEAQYASABEgLy5vD\\_BwE](https://www.auselectronicdirect.com.au/wireless-bluetooth-4.0-module-for-arduino-projects?gclid=EAIaIQobChMIkoCv2My75QIV2RwrCh33XQaGEAQYASABEgLy5vD_BwE)>.

*XC-4434 Hall Effect Sensor Module* Year Unknown, Duinotech, Datasheet, China, viewed 31  
May 2020,  
<[https://www.jaycar.com.au/medias/sys\\_master/images/images/9403718828062/XC4434-dataSheetMain.pdf](https://www.jaycar.com.au/medias/sys_master/images/images/9403718828062/XC4434-dataSheetMain.pdf)>.

## Appendix A – Project Specification

## ENG4111/4112 Research Project

### Project Specification

For: Zachary Hugh Montgomery

Title: Mapping rollercoaster forces using acceleration and GNSS data

Major: Mechatronic Engineering

Supervisors: John Billingsley

Enrolment: ENG4111 – EXT S1, 2020    ENG4112 – EXT S2, 2020

Project Aims:

1. Accurately match tri-axial acceleration data to displacement locations on a rollercoaster track.
2. Develop an autonomous data gathering device to mount on a rollercoaster which relays captured acceleration data to a receiver each time the vehicle returns to station (completes a lap/cycle).

**Programme:            Version 1, 9<sup>th</sup> March 2020**

1. Research current methods of capturing data for both displacement and acceleration used in industry, possibly found in the autonomous vehicle industry
2. Research different methods of capturing data focusing on specific types, i.e. GPS, accelerometer, encoder, other sensor/mechanism
3. Research software to correlate the data and present it in a usable way (may need to learn new coding language)
4. Research ways of processing raw data so that it better reflects the actual forces experienced on the ride
5. Design a test model which can capture the data required and set up experiments record real data on a rollercoaster track
6. Process and analyse the raw data and compare results with track models and existing acceleration reports

*If time and resources permit:*

7. Design a wireless, transmittable data recorder
8. Investigate feasibility of powering device off 24V station bus bar
9. Fabricate and assemble device
10. Test on rollercoaster

## Appendix B – Risk Assessment

## RISK ASSESSMENT FORM

### RISK MATRIX

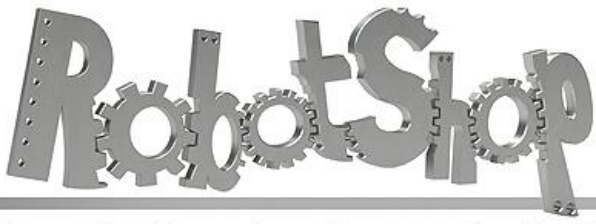
#### Consequence of Impact

DETERMINE CONSEQUENCE AND PROBABILITY	1. Insignificant	2. Minor	3. Moderate	4. Major	5. Catastrophic
<b>KEY</b>	No injury or injury does not require treatment	First aid treatment by Nurse or First Aider	Medical treatment by Nurse or Medical Professional. (MTI) (1-5 days off work)	Long term severe injury (more than one week off work)	Fatality or severe irreversible disability
E-Extreme					
H-High					
M-Medium					
L- Low					
<b>Likelihood</b>					
Almost Certain - Could expect event in most circumstances	11	7	4	2	1
Likely – could occur multiple times per year	16	12	8	5	3
Possible – could occur once per year	20	17	13	9	6
Unlikely - could occur once per 1 – 5 years	23	21	18	14	10
Rare - Could occur once greater than 5 years	25	24	22	19	15

Hazard/Impact Identification	Risk/Significance	Practicable		Risk Control /Actions			Risk/Significance
Describe in detail the task, location & type of hazard	Select the most appropriate consequence/severity and then assess likelihood to determine risk.	Is it practicable to remove or mitigate the hazard or risk?		Determine the short- and long-term measures, recommended dates for completion. Refer to Hierarchy of Control for order of implementing control measures. Include reasoning for the control measures chosen.  * Control Status – NS = Not Started IP = In Progress C = Complete			Select the most appropriate consequence/severity and then assess likelihood to determine risk.
Description	Unmitigated Risk Score	Yes	No - why not?	Controls	Control Status *	Date completed	Controlled Risk Score
Electrical shock when building and testing electronic equipment	20	Yes		Power source limited to 5V - both laptop and battery. Sound understanding of electrical principles and good practice used throughout. Good quality, well tested, extra low voltage components used.	C	30/3/20	25
Burn from hot soldering iron during building of electronic equipment	12	Yes		Good quality soldering station purchased with temperature control and ventilated stand for hot iron to rest on. Protective glasses worn during work.	C	30/3/20	24
Inadvertent vehicle movement leading to crushing or striking	9	Yes		Lock Out Tag Out (Isolate) the amusement device before commencing any work. Always have a competent operator on the panel during testing and have clear communication to ensure no train movement until all personnel are clear.	C	Existing control - VRTP procedure	19
Equipment falls out during testing causing a striking hazard to people in the ride envelope	13	Yes		Follow testing template and procedure to ensure test dummy and equipment is secure in vehicle before launch to reduce the likelihood of parts falling.	C	30/3/20	22
Data acquired is not accurate or representative	20	Yes		Follow testing template and procedure. Calibrate hardware before field test and take as many samples as time allows.	C	30/3/20	25
Cannot collect data due to unscheduled maintenance	16	Yes		Testing window long enough to accommodate. Multiple amusement device options available in case of unavailability.	C	30/3/20	25
Travelling on the ride with the equipment to gain a subjective comparison	9	Yes		Only ride with trained and competent operator at the control panel. Ensure safely secured in restraint before dispatch.	C	Existing control - VRTP procedure	19

## Appendix C – Datasheets





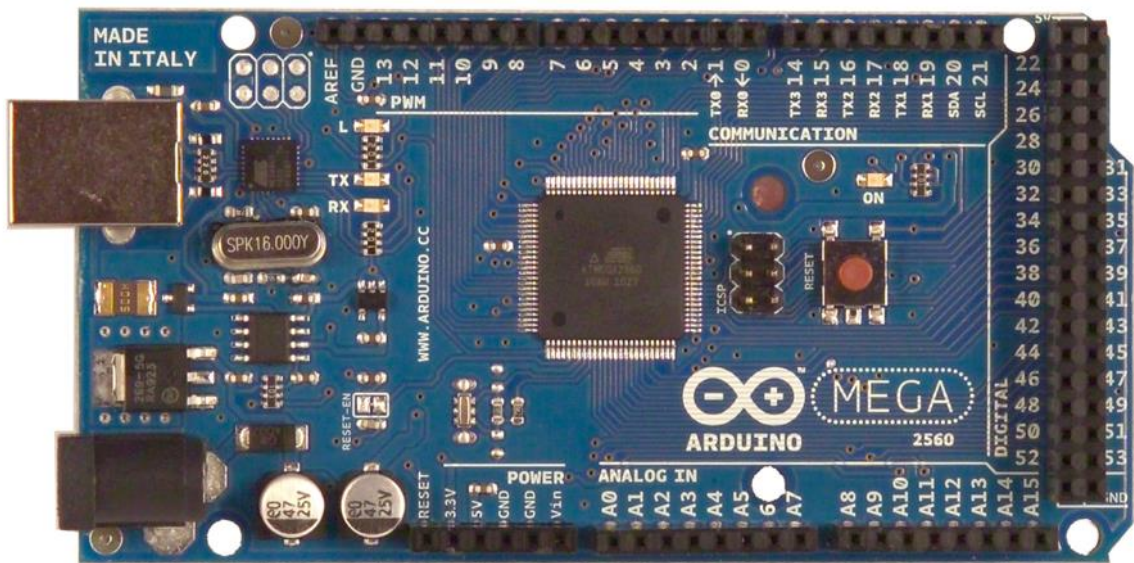
[www.robotshop.com](http://www.robotshop.com)

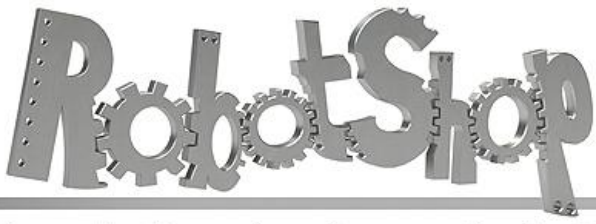


La robotique à votre service! - Robotics at your service!



## Arduino Mega 2560 Datasheet

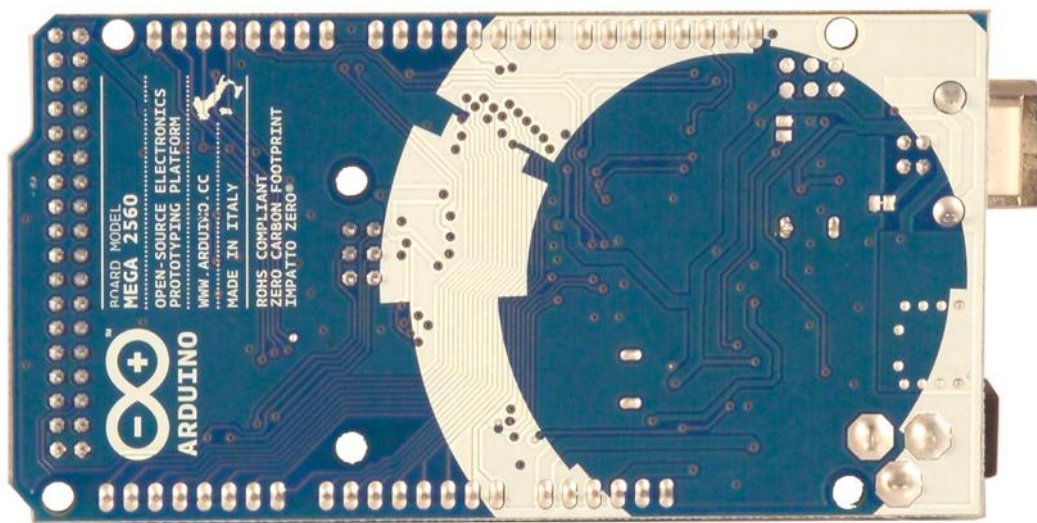




[www.robotshop.com](http://www.robotshop.com)



La robotique à votre service! - Robotics at your service!

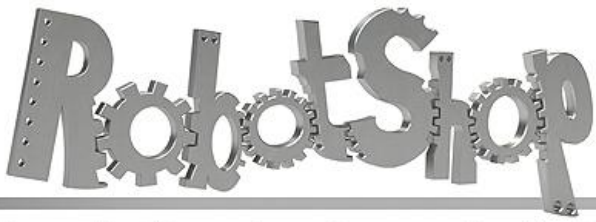


## Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 ([datasheet](#)). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC- to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

## Schematic & Reference Design

EAGLE files: [arduino-mega2560-reference-design.zip](#)



[www.robotshop.com](http://www.robotshop.com)



**La robotique à votre service! - Robotics at your service!**

Schematic: [arduino-mega2560-schematic.pdf](#)

## Summary

Microcontroller  
Operating Voltage  
Input Voltage (recommended) Input Voltage (limits)  
Digital I/O Pins  
Analog Input Pins  
DC Current per I/O Pin  
DC Current for 3.3V Pin  
Flash Memory  
SRAM  
EEPROM  
Clock Speed

## Power

ATmega2560 5V  
7-12V  
6-20V

54 (of which 14 provide PWM output) 16  
40 mA  
50 mA

256 KB of which 8 KB used by bootloader 8 KB  
4 KB  
16 MHz

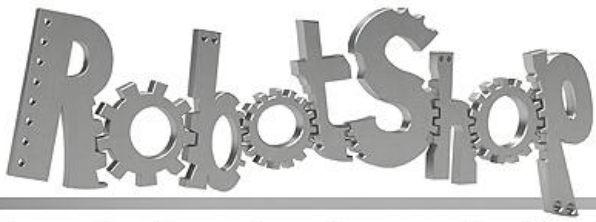
---

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.



www.robotshop.com



La robotique à votre service! - Robotics at your service!

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins. **Memory**

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the [EEPROM library](#)).

## Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using [pinMode\(\)](#), [digitalWrite\(\)](#), and [digitalRead\(\)](#) functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the [attachInterrupt\(\)](#) function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the [analogWrite\(\)](#) function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI

communication using the [SPI library](#). The SPI pins are also broken out on the ICSP

header, which is physically compatible with the Uno, Duemilanove and Diecimila.

- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH



value, the LED is on, when the pin is LOW, it's off.

- **I2C: 20 (SDA) and 21 (SCL).** Support I2C (TWI) communication using the [Wire](#)

[library](#) (documentation on the Wiring website). Note that these pins are not in the same location as the I2C pins on the Duemilanove or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and `analogReference()` function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with [analogReference\(\)](#).
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset

button to shields which block the one on the board.

## Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A [SoftwareSerial library](#) allows for serial communication on any of the Mega2560's digital pins.

The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a [Wire library](#) to simplify use of the I2C bus; see the [documentation on the Wiring website](#) for details. For SPI communication, use the [SPI library](#).

# Programming

The Arduino Mega can be programmed with the Arduino software ([download](#)). For details, see the [reference](#) and [tutorials](#).

The ATmega2560 on the Arduino Mega comes preburned with a [bootloader](#) that allows you to upload new code to it without the use of an external hardware programmer. It



communicates using the original STK500 protocol ([reference](#), [C header files](#)).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see [these instructions](#) for details.

## Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code),

it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega2560 contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN".

You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see [this forum thread](#) for details.

## USB Overcurrent Protection

The Arduino Mega2560 has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

## Physical Characteristics and Shield Compatibility



The maximum length and width of the Mega2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega2560 and Duemilanove / Diecimila. *Please note that I2C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).*



# NEO-6 series

## Versatile u-blox 6 GPS modules

### Highlights

- UART, USB, DDC (I<sup>2</sup>C compliant) and SPI interfaces
- Available in Crystal and TCXO versions
- Onboard RTC crystal for faster warm and hot starts
- 1.8 V and 3.0 V variants



NEO-6:  
12.2 x 16.0 x 2.4 mm

### Features

- u-blox 6 position engine:
  - Navigate down to -162 dBm and -148 dBm coldstart
  - Faster acquisition with AssistNow Autonomous
  - Configurable power management
  - Hybrid GPS/SBAS engine (WAAS, EGNOS, MSAS)
  - Anti-jamming technology
- Simple integration with u-blox wireless modules
- A-GPS: AssistNow Online and AssistNow Offline services, OMA SUPL compliant
- Backward compatible (hardware and firmware); easy migration from NEO-5 family or NEO-4S
- LCC package for reliable and cost effective manufacturing
- Compatible with u-blox GPS Solution for Android
- Based on GPS chips qualified according to AEC-Q100
- Manufactured in ISO/TS 16949 certified production sites
- Qualified according to ISO 16750

### Product description

The NEO-6 module series brings the high performance of the u-blox 6 position engine to the miniature NEO form factor. u-blox 6 has been designed with low power consumption and low costs in mind. Intelligent power management is a breakthrough for low-power applications. These receivers combine a high level of integration capability with flexible connectivity options in a small package. This makes them perfectly suited for mass-market end products with strict size and cost requirements. The DDC interface provides connectivity and enables synergies with u-blox LEON and LISA wireless modules.

All NEO-6 modules are manufactured in ISO/TS 16949 certified sites. Each module is tested and inspected during production. The modules are qualified according to ISO 16750 - Environmental conditions and electrical testing for electrical and electronic equipment for road vehicles.

### Product selector

Model	Type	Supply	Interfaces	Features
	Standalone GPS Standalone GLONASS Timing & Raw Data Dead Reckoning	1.75 V – 2.0 V 2.7 V – 3.6 V	UART USB SPI DDC (I <sup>2</sup> C compliant)	Programmable (Flash) FW update Oscillator RTC crystal Antenna supply and supervisor Configuration pins Timepulse External interrupt / Wakeup
NEO-6G	•	•	• • • •	T • ○ 3 1 •
NEO-6Q	•	•	• • • •	T • ○ 3 1 •
NEO-6M	•	•	• • • •	C • ○ 3 1 •

○ = requires external components and integration on application processor

C = Crystal / T = TCXO

## Receiver performance data

Receiver type	50-channel u-blox 6 engine GPS L1 C/A code SBAS: WAAS, EGNOS, MSAS	
Navigation update rate	up to 5 Hz	
Accuracy <sup>1</sup>	Position	2.5 m CEP
	SBAS	2.0 m CEP
Acquisition <sup>1</sup>	NEO-6G/Q	NEO-6M
	Cold starts:	26 s    27 s
	Aided starts <sup>2</sup> :	1 s    < 3 s
	Hot starts:	1 s    1 s
Sensitivity <sup>3</sup>	NEO-6G/Q	NEO-6M
	Tracking:	-162 dBm    -161 dBm
	Cold starts:	-148 dBm    -147 dBm
	Hot starts:	-157 dBm    -156 dBm

<sup>1</sup> All SV @ -130 dBm

<sup>2</sup> Dependent on aiding data connection speed and latency

<sup>3</sup> Demonstrated with a good active antenna

## Electrical data

Power supply	2.7 V – 3.6 V (NEO-6Q/6M) 1.75 V – 2.0V (NEO-6G)
Power consumption	111 mW @ 3.0V (continuous) 33 mW @ 3.0V Power Save Mode (1 Hz) 68 mW @ 1.8V (continuous) 22 mW @ 1.8V Power Save Mode (1 Hz)
Backup power	1.4 V – 3.6V, 22 µA
Supported antennas	Active and passive

## Interfaces

Serial interfaces	1 UART 1 USB V2.0 full speed 12 Mbit/s 1 DDC (I <sup>2</sup> C compliant) 1 SPI
Digital I/O	Configurable timepulse 1 EXTINT input for Wakeup
Serial and I/O	Voltages    2.7 – 3.6 V (NEO-6Q/6M) 1.75 – 2.0 V (NEO-6G)
Timepulse	Configurable    0.25 Hz to 1 kHz
Protocols	NMEA, UBX binary, RTCM

### Legal Notice

u-blox reserves all rights to this document and the information contained herein. Products, names, logos and designs described herein may in whole or in part be subject to intellectual property rights. Reproduction, use, modification or disclosure to third parties of this document or any part thereof without the express permission of u-blox is strictly prohibited.

The information contained herein is provided "as is". No warranty of any kind, either express or implied, is made in relation to the accuracy, reliability, fitness for a particular purpose or content of this document. This document may be revised by u-blox at any time. For most recent documents, please visit [www.u-blox.com](http://www.u-blox.com).

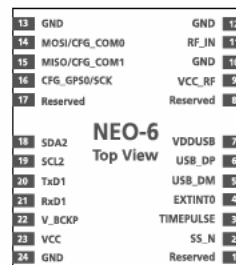
Copyright © 2011, u-blox AG

Specification applies to FW 7

## Package

24 pin LCC (Leadless Chip Carrier): 12.2 x 16.0 x 2.4 mm, 1.6 g

Pinout



## Environmental data, quality & reliability

Operating temp.    -40° C to 85° C

Storage temp.    -40° C to 85° C

RoHS compliant (lead-free)

Qualification according to ISO 16750

Manufactured in ISO/TS 16949 certified production sites

## Support products

u-blox 6 Evaluation Kits:

Easy-to-use kits to get familiar with u-blox 6 positioning technology, evaluate functionality, and visualize GPS performance.

EVK-6H:    u-blox 6 Evaluation Kit with TCXO, suitable for NEO-6G, NEO-6Q

EVK-6P:    u-blox 6 Evaluation Kit with crystal, suitable for NEO-6M

## Ordering information

NEO-6G-0	u-blox 6 GPS Module, 1.8V, TCXO, 12x16mm, 250 pcs/reel
NEO-6M-0	u-blox 6 GPS Module, 12x16mm, 250 pcs/reel
NEO-6Q-0	u-blox 6 GPS Module, TCXO, 12x16mm, 250 pcs/reel

Available as samples and tape on reel (250 pieces)

## Contact us

HQ Switzerland  
+41 44 722 7444  
[info@u-blox.com](mailto:info@u-blox.com)

China  
+86 10 68 133 545  
[info\\_cn@u-blox.com](mailto:info_cn@u-blox.com)

EMEA  
+41 44 722 7444  
[info@u-blox.com](mailto:info@u-blox.com)

Japan  
+81 3 5775 3850  
[info\\_jp@u-blox.com](mailto:info_jp@u-blox.com)

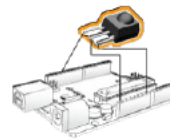
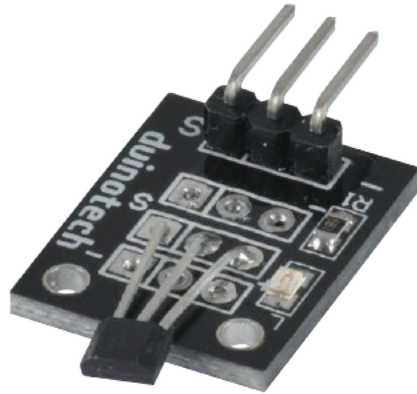
Americas  
+1 703 483 3180  
[info\\_us@u-blox.com](mailto:info_us@u-blox.com)

Korea  
+82 2 542 0861  
[info\\_kr@u-blox.com](mailto:info_kr@u-blox.com)

APAC – Singapore  
+65 6734 3811  
[info\\_ap@u-blox.com](mailto:info_ap@u-blox.com)

Taiwan  
+886 2 2657 1090  
[info\\_tw@u-blox.com](mailto:info_tw@u-blox.com)

## Hall Effect Sensor Module



■ Type: Module  
 Application: Add On Module  
 Magnetic Field Sensor

Dimensions: 20(L) x 15(W) x 3(H)mm

Specifications	
	Hall Effect Sensor Module
Activation threshold	30 Gauss
Deactivation threshold	10 Gauss
Operating Voltage	4.5VDC - 48VDC
Output Type	Schmitt Trigger(Digital) Active Low
Dimensions	20(L) x 15(W) x 3(H)

Specifications		
Module	Duinotech	Function
-	GND	Ground
Middle	5V	Power
S	D13	Output Signal

### Hall Effect Sensor Module Overview:

Sense magnetic presence, rotating wheels and magnets, door sensors and any other magnets near this device.

What is included: 1 x Hall Effect Sensor Module

Essential Accessories: Magnets (LM1622)  
Jumper Leads (WC6028)

Optional Accessories:

### Hall Effect Sensor Module Sample Projects:



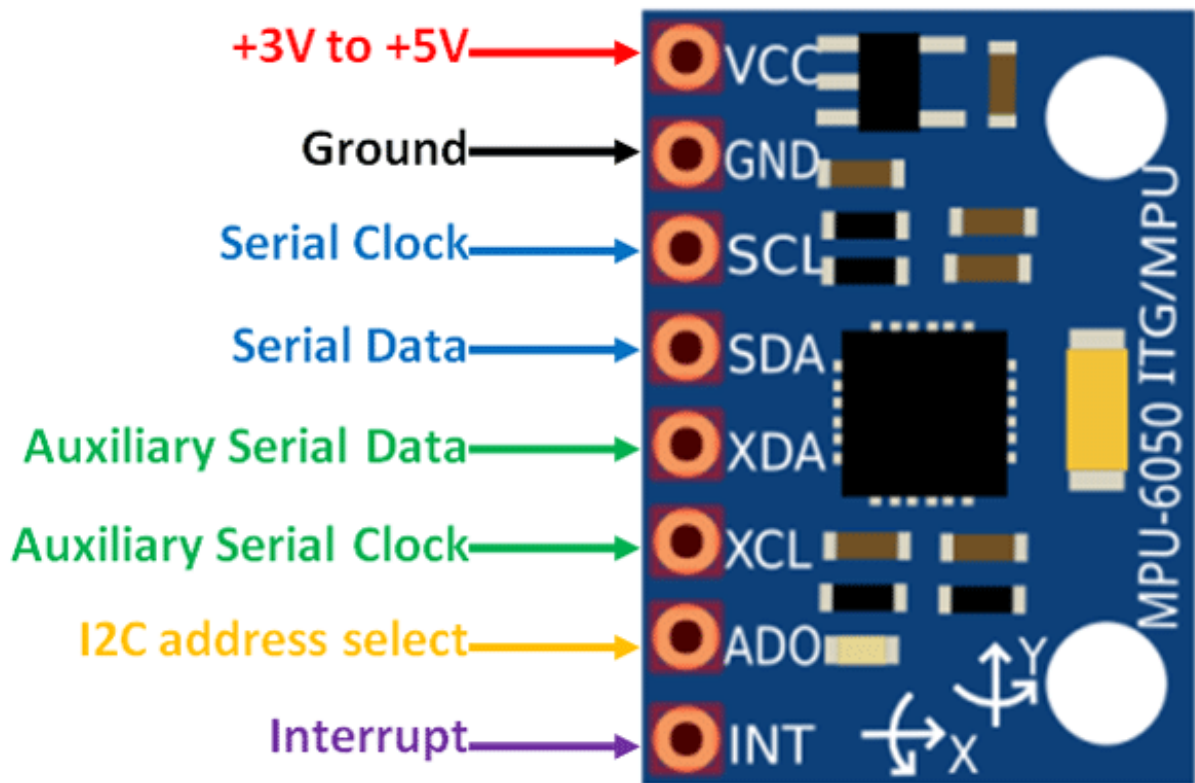
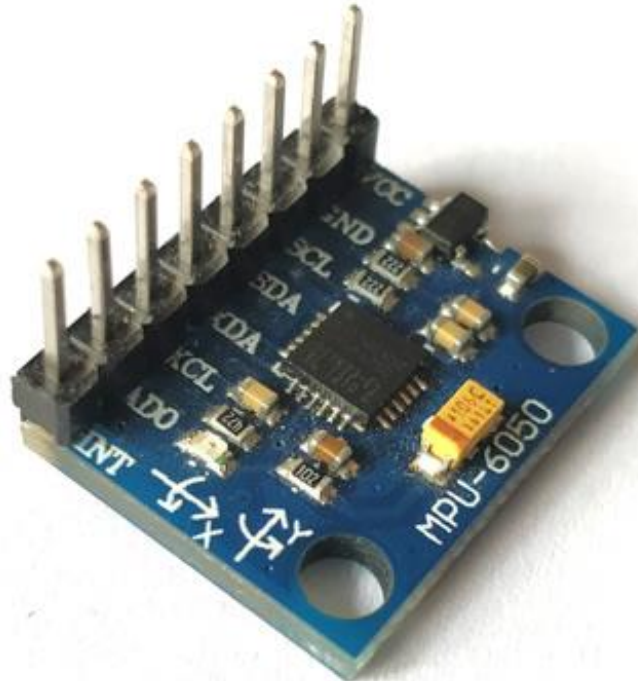
### Did you know:

There are also hall effects sensors like the Jaycar (ZD1902) which ran output as an analog signal proportional to the magnetic field strength



Distributed by  
 TechBrands by Electus Distribution Pty. Ltd.  
 Ph 1300 738 555  
[www.techbrands.com](http://www.techbrands.com)  
 Made in China

# MPU6050 - Accelerometer and Gyroscope Module



## MPU6050 Module

### MPU6050 Pinout

[Click the image to enlarge it]

## MPU6050 Pin Configuration

Pin Number	Pin Name	Description
1	Vcc	Provides power for the module, can be +3V to +5V. Typically +5V is used
2	Ground	Connected to Ground of system
3	Serial Clock (SCL)	Used for providing clock pulse for I2C Communication
4	Serial Data (SDA)	Used for transferring Data through I2C communication
5	Auxiliary Serial Data (XDA)	Can be used to interface other I2C modules with MPU6050. It is optional
6	Auxiliary Serial Clock (XCL)	Can be used to interface other I2C modules with MPU6050. It is optional
7	AD0	If more than one MPU6050 is used a single MCU, then this pin can be used to vary the address
8	Interrupt (INT)	Interrupt pin to indicate that data is available for MCU to read.

## MPU6050 Features

- MEMS 3-axis accelerometer and 3-axis gyroscope values combined
- Power Supply: 3-5V
- Communication : I2C protocol
- Built-in 16-bit ADC provides high accuracy
- Built-in DMP provides high computational power
- Can be used to interface with other IIC devices like magnetometer
- Configurable IIC Address
- In-built Temperature sensor

## Appendix D – Program Code

Program: ZMGPSTime

```
#include <SoftwareSerial.h>
```

```
#include <SD.h>
```

```
#include <TinyGPS++.h>
```

```
/*
```

This program has been created by Zachary Montgomery to record both GPS and time data on a SD card. SD Card source code by Dejan Nedelkovski and Tom Igoe, TinyGPS++ source code by Mikal Hart,

```
*/
```

```
static const int RXPin = 10, TXPin = 11;
```

```
static const uint32_t GPSBaud = 9600;
```

```
const int chipSelect = 53; // Arduino Mega specific
```

```
String headers = "Time,Latitude,Longitude";
```

```
String dataFile = "ZMdata"; // used as file name, need to follow 8.3 naming convention
```

```
unsigned long Time;
```

```
SoftwareSerial ss(RXPin, TXPin);
```

```
TinyGPSPlus gps;
```

```
File logfile;
```

```
void setup() {
```

```
  Serial.begin(115200);
```

```
  ss.begin(GPSBaud);
```

```
  Serial.println("Testing device connections...");
```

```
  Serial.println(SD.begin(chipSelect) ? "SD card is ready to use." : "Card failed, or not present");
```

```
  bool file = false; // tracks when found a usable filename
```

```

int count = 0; // used for naming the file
while (!file) {
  if (SD.exists(dataFile)) {
    count++; // up the count for naming the file
    dataFile = "ZMdata" + (String)count; // make a new name
    dataFile = dataFile + String(".csv");
  }
  else file = true; // once newly named file created, exit the loop
}
File logfile = SD.open((dataFile), FILE_WRITE); // open file to write labels to CSV
Serial.println(SD.exists(dataFile) ? "File exists." : "File doesn't exist.");
Serial.print("Filename: "); Serial.println(dataFile);
delay(1000);
if (logfile) {
  logfile.println(headers);
  Serial.println(headers);
  logfile.close();
}
else {
  Serial.println("Logfile not found!");
  while (1) {
    ; // if logfile not found, do nothing...
  }
}
}

```



```

void loop() {
  while (ss.available() > 0)
    if (gps.encode(ss.read())) {
      logfile = SD.open((dataFile), FILE_WRITE); // open file to write labels to CSV
      PosTime();
      GPS();
      logfile.close();
      //delay(1); // delay in between reads for stability
    }
  if (millis() > 5000 && gps.charsProcessed() < 10) {
    Serial.println(F("No GPS detected: check wiring."));
    while(true);
  }
}

```

```

void PosTime() {
  // add time value to file / print to Serial:
  Time = millis();
  logfile.print(Time); logfile.print(",");
  Serial.print(Time); Serial.print(",");
}

```

```

void GPS() {
  // if available add GPS data to file / print to Serial:
  if (gps.location.isValid()) {
    logfile.print(gps.location.lat(), 8); logfile.print(",");
    logfile.print(gps.location.lng(), 8); logfile.println("");
  }
}

```

```
Serial.print(gps.location.lat(), 8); Serial.print(",");  
Serial.print(gps.location.lng(), 8); Serial.println("");  
}  
else {  
  logfile.print("INVALID"); logfile.println("");  
  Serial.print("INVALID"); Serial.println("");  
}  
}
```

Program: ZMAccelCountv3

```
#include <SoftwareSerial.h>
```

```
#include <SD.h>
```

```
#include <SPI.h>
```

```
#include "I2Cdev.h"
```

```
#include "MPU6050.h"
```

```
#include "Wire.h"
```

```
/*
```

This program has been created by Zachary Montgomery to record both wheel count and accelerometer data on a SD card. SD Card source code by Dejan Nedelkovski and Tom Igoe, MPU6050 source code by Jeff Rowberg, Main Loop pseudocode by Prof. John Billingsley.

```
*/
```

```
static const int RXPin = 10, TXPin = 11;
```

```
const int chipSelect = 53; // Arduino Mega specific
```

```
String headers = "Time,Count,Ax,Ay,Az,Gx,Gy,Gz";
```

```
String dataFile = "ZMdata"; // used as file name, need to follow 8.3 naming convention
```

```
unsigned long Time;
```

```
const int hallEffect = 12;
```

```
const int hallPower = 13;
```

```
const int accelGnd = 2;
```

```
bool countBlock;
```

```
int count, sensorState;
```

```
int16_t ax, ay, az;
```

```
int16_t gx, gy, gz;
```

```
SoftwareSerial ss(RXPin, TXPin);
```

```
File logfile;
```

```
MPU6050 accelgyro;
```

```
#define OUTPUT_READABLE_ACCELGYRO
```

```
#define LED_PIN 13
```

```
bool blinkState = false;
```

```
void setup() {
```

```
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE // join I2C bus
```

```
  Wire.begin();
```

```
#elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
```

```
  Fastwire::setup(400, true);
```

```
#endif
```

```
  Serial.begin(115200);
```

```
  pinMode(hallEffect, INPUT);
```

```
  pinMode(hallPower, OUTPUT);
```

```
  digitalWrite(hallPower, HIGH);
```

```
  pinMode(accelGnd, OUTPUT);
```

```
  digitalWrite(accelGnd, LOW);
```

```
  pinMode(LED_PIN, OUTPUT);
```

```
  count = 0;
```

```
  countBlock = false;
```

```
  Serial.println("Initializing I2C devices...");
```

```
  accelgyro.initialize();
```

```
  Serial.println("Testing device connections...");
```

```
  Serial.println(accelgyro.testConnection() ? "MPU6050 connection successful" :  
"MPU6050 connection failed");
```

```
Serial.println(SD.begin(chipSelect) ? "SD card is ready to use." : "Card failed, or not present");
```

```
bool file = false; // tracks when found a usable filename
```

```
int count = 0; // used for naming the file
```

```
while (!file) {
```

```
    if (SD.exists(dataFile)) {
```

```
        count++; // up the count for naming the file
```

```
        dataFile = "ZMdata" + (String)count; // make a new name
```

```
        dataFile = dataFile + String(".csv");
```

```
    }
```

```
    else file = true; // once newly named file created, exit the loop
```

```
}
```

```
File logfile = SD.open((dataFile), FILE_WRITE); // open file to write labels to CSV
```

```
Serial.println(SD.exists(dataFile) ? "File exists." : "File doesn't exist.");
```

```
Serial.print("Filename: "); Serial.println(dataFile);
```

```
delay(1000);
```

```
if (logfile) {
```

```
    logfile.println(headers);
```

```
    Serial.println(headers);
```

```
    logfile.close();
```

```
}
```

```
else {
```

```
    Serial.println("Logfile not found!");
```

```
    while (1) {
```

```
        ; // if logfile not found, do nothing...
```

```
    }
```

```
}
```

```
}
```

```
void loop() {
```

```
  logfile = SD.open((dataFile), FILE_WRITE); // open file to write labels to CSV
```

```
  sensorState = digitalRead(hallEffect);
```

```
  sensorState = !sensorState;
```

```
  if (sensorState == true) {
```

```
    if (countBlock == false) {
```

```
      PosTime();
```

```
      SensorCount();
```

```
      AccelGyro();
```

```
      count++;
```

```
      countBlock = true;
```

```
      logfile.println("");
```

```
      Serial.println("");
```

```
    }
```

```
  else {
```

```
    PosTime();
```

```
    SensorCount();
```

```
    AccelGyro();
```

```
    logfile.println("");
```

```
    Serial.println("");
```

```
  }
```

```
}
```

```
else {
```

```
  PosTime();
```

```
  SensorCount();
```

```
    AccelGyro();  
    countBlock = false;  
    logfile.println("");  
    Serial.println("");  
}  
logfile.close();  
//delay(1); // delay in between reads for stability  
}
```

```
void PosTime() {  
    // add time value to file / print to Serial:  
    Time = millis();  
    logfile.print(Time); logfile.print(",");  
    Serial.print(Time); Serial.print(",");  
}
```

```
void SensorCount() {  
    // add wheel count data to file / print to Serial:  
    logfile.print(count); logfile.print(",");  
    Serial.print(count); Serial.print(",");  
}
```

```
void AccelGyro() {  
    // add accelerometer data to file / print to Serial:  
    accelgyro.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);  
    logfile.print(ax); logfile.print(",");  
}
```

```
logfile.print(ay); logfile.print(",");  
logfile.print(az); logfile.print(",");  
logfile.print(gx); logfile.print(",");  
logfile.print(gy); logfile.print(",");  
logfile.print(gz);  
Serial.print(ax); Serial.print(",");  
Serial.print(ay); Serial.print(",");  
Serial.print(az); Serial.print(",");  
Serial.print(gx); Serial.print(",");  
Serial.print(gy); Serial.print(",");  
Serial.print(gz);  
}
```