Glasgow Caledonian University
Electrical and Electronic Engineering
Mechatronics

Michael Jeans
S2027770

eCargo Trailer Project
Final Report

Version 1.3

Date: 23 April 2022.

Abstract

This report details the design of an innovative cargo trailer, which is attached to an electric bike to increase the capacity for last mile delivery purposes. Unlike conventional single axle trailers, which are difficult to load when not attached to a pilot vehicle, this eCargo Trailer is a self-supporting trike, allowing for a more convenient loading. The load area exceeds the dimensions of common pallet sizes, giving the trailer a similar load capacity as a small light goods vehicle.

While a conventional trailer depends on a powered vehicle for propulsion, the eCargo Trailer is independently driven by a front hub brushless motor controlling its own speed and braking using strain sensing technology. The eCargo Trailer only requires a connecting arm from the pilot bicycle to steer. Importantly, regenerative braking technology used in electric vehicles such as cars and vans to extend range and to reduce brake pad wear is incorporated in the design of this novel type of trailer.

In addition, motor monitoring software strengthens the trailer's design by using sensor technology to capture speed, torque, current and temperature data, which allows for motor efficiency to be calculated and recorded along with other important data through the I^2C protocol.

The Arduino Uno controls the motor via software written in 'C' language. Pulse width modulation (PWM) techniques were used and evaluated along with interrupt timings, duty cycle and commutation steps for both regeneration and drive modes. Proteus was used for the circuit design and to design a printed circuit board (PCB) for the motor control circuit.

.

Acknowledgements

Thanks to Mario and Gabi.

1. Contents

2. Table of figures

3. List of tables

4. List of acronyms

Brushless direct current	BLDC
Amp hours	Ah
Analogue to digital converter	ADC
Battery management system	BMS
Bill of materials	BOM
Electromagnetic force	EMF
Electrically erasable programmable read-only memory	EEPROM
Finite element analysis	FEA
Input and output	I/O
Insulated-gate bipolar transistor	IGBT
Light emitting diode	LED
Light goods vehicles	LGV
Low emission zones	LEZ
Metal-oxide-semiconductor field-effect transistors	MOSFET
Pedal assist electric bike	Pedelec
Personal protective equipment	PPE
Printed circuit board	PCB
Proof of concept	POC
Pulse width modulation	PWM

5. Introduction

This project presents a proof of concept for electronic control of an innovative electric trailer called the eCargo Trailer, seen in Figure 1. The eCargo Trailer is a self-supporting trike, enabling more convenient loading, in contrast with conventional single axle trailers, which are difficult to load when not attached to a pilot vehicle. This type of small electric vehicle fills a gap in the commercial goods supply chain market by increasing the volume of last mile deliveries made by bikes [1]. Although the eCargo Trailer might not be able to completely replace deliveries made by light goods vehicles, it holds exciting potential to expand the volume capacity of current cargo bikes to include the dimensions of the commonly used euro pallet [2]. When the trailer is attached to a cargo bike, the average loading capacity, both in volume and weight, equals that of a typical light goods vehicle (LGV) [3]. EU-wide regulations stipulate a maximal speed of 25 kph for pedal-assisted electric bikes [4], which has been implemented in the design of the eCargo Trailer to allow it to travel on both roads and bike infrastructure, leading to superior movement efficiency around city environments.

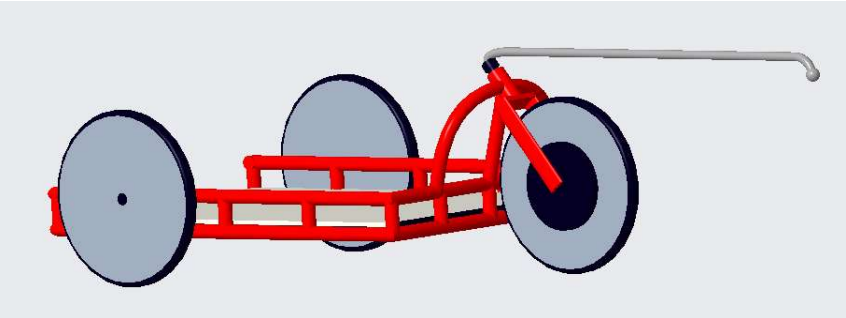


FIGURE 1 – ECARGO TRAILER

Figure 2 shows the basic structure of the circuits needed for this control.

To instruct the motor when to power and when to brake, a load cell is placed on the connecting arm from the eCargo Trailer to the bike. When the rider starts to pedal, propelling the pilot bike, the arm stretches under tensile loading conditions, and the load cell detects the change in force, causing the eCargo Trailer's motor to accelerate. When the pilot bike slows down, the arm compresses under the compressive force, and the eCargo Trailer's motor decelerates using regenerative braking to recharge the battery. This is realised by the acceleration and deceleration of a Brushless Direct Current (BLDC) hub motor, controlled by an Arduino Uno, which supplies a pulse width modulation (PWM) signal via a motor control circuit. Sensor data is collected to monitor motor efficiency and is saved to memory through the I^2C protocol.

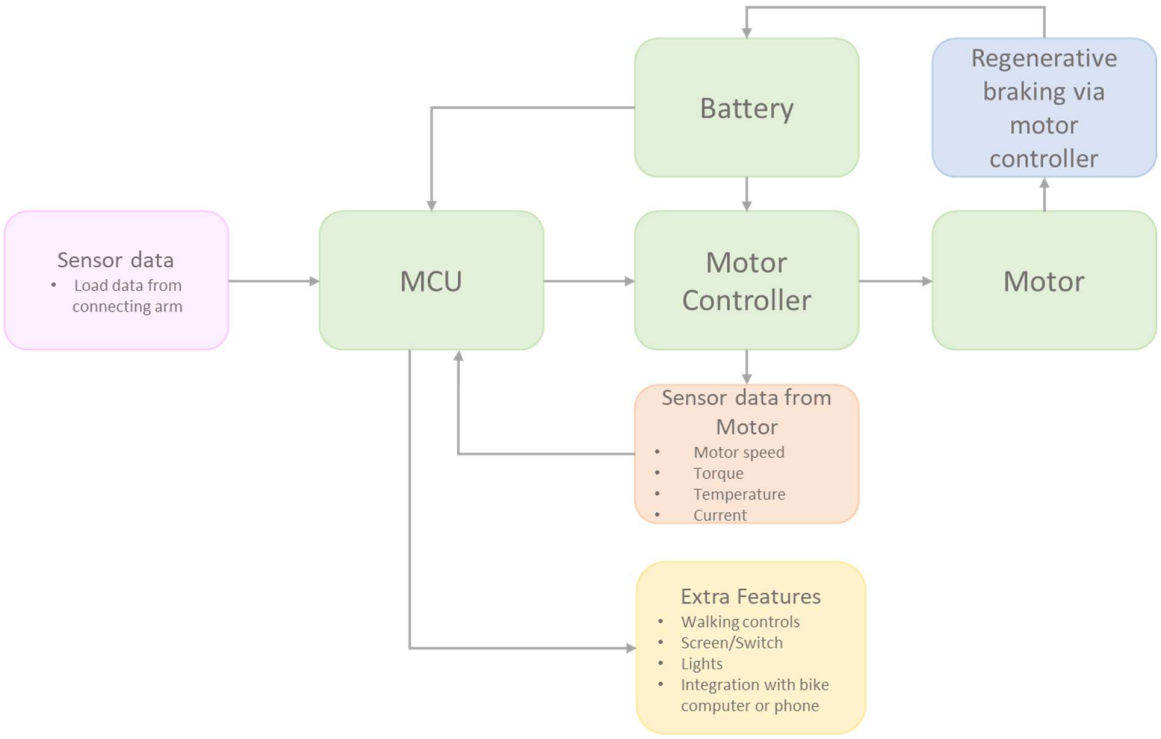


FIGURE 2 - BLOCK DIAGRAM OF PRODUCT

Acceleration uses a set of high side and low side transistors to direct current along the correct phases to the stator coils, forcing the magnets on the rotor to rotate. Halls sensors detect the position of the rotor in relation to the stator, making sure the correct phases and coils are energised. Regeneration uses the halls sensors to detect the rotor position and uses the same set of six transistors to form a buck boost circuit, by charging up the inductive coils in the motor and discharging them back into the battery. These methods have been widely applied in other electric vehicles and have only recently received attention from electric cargo bike manufacturers. Given the rising demand for last minute deliveries and the lack of environmentally considerate options, the eCargo Trailer presents an innovative solution to a bottleneck of supply chain.The aims and objectives of the project are detailed below:

Project aim:

Proof of concept (POC) of a smart trailer used in conjunction with an electric bike for last-mile delivery in a city environment.

Project objectives:

This project includes all aspects of the POC process, including:

- Management of the POC process
- Demonstrate need including a literature review
- Ideate a solution
- Create a prototype
- Test prototype
- Gather and document feedback
- Present POC for approval

The project adhered to the product schedule and met the deadlines.

The project aimed to include the following technical objectives:

- A motorised trailer that is controlled using signals from a pilot bike.
- Regenerative braking to assist with slowing the trailer while in normal use.
- A maintenance management system that oversees trailer performance and create flags when performance is inhibited.
- A strain gauge is used as a safety mechanism for slowing the trailer.

The following report details important aspects of the proof of concept. It justifies the need for the product and demonstrates that the design is feasible, through market research and a discussion of technical methods that have been researched from peer reviewed literature. Implementation of the researched technical methods are detailed in the circuit and printed circuit board (PCB) design, as well as through software, which is analysed and tested. Conclusions follow on whether the POC was successful, and suggestions on what needs be considered next to move the eCargo Trailer towards engineering validation tests and closer to production.

6. Demonstrate Need

The rise of e-commerce over the last decade has caused an increase in last-mile deliveries [5]. The majority of these deliveries are made by diesel-fuelled light goods vehicles (LGV), which are heavy polluters and governments across the globe are moving quickly to impose transport restrictions to discourage their use in big cities. [6] [7]. Low emission zones (LEZ) were first introduced in the 1990s and have since decreased the use of larger diesel vehicles in inner cities across several large cities [8] [9]. There are plans to introduce and extend LEZ in cities all across the UK [10]. Recently Glasgow hosted the United Nations summit for climate change COP26, where cleaner and more efficient vehicles were a major point of discussion for the new deals reached [11]. However, the move to electric vehicles is slow, and although sales are increasing, they are partly limited by infrastructure. This includes low levels of vehicle production due to electronic chip shortages caused by supply chain issues and which market leader Bosch announced is not fit for service [12].

The UK government lists "e-cargo bikes and similar" as a solution to this problem in the city environment [1] and there are now dedicated funds for businesses in the UK to purchase e-cargo delivery bikes [13]. The UK government has estimated that around 33% of the last mile deliveries could be made by bike or E-bike [14]. EU cities have been trialling schemes that increase green last-mile deliveries for up to 50-mile deliveries in a bid to meet environmental targets [15]. E-cargo bikes and trailers are a major part of the success of these schemes, as they get closer to matching the load an LGV can carry [3]. To avoid costly carbon offsetting rules large international companies such as Amazon and UPS are developing and using their own cargo bikes to keep volume capacity high [16].

6.1 Governmental rules/laws

Electric bikes are under the pedal assist (pedelec) category of EVs, which stipulates a maximum speed limit of 25kph, although there are varying power limit differences between cargo and normal e-bikes [4]. Recently the EU has decided not to overregulate this category in a bid to encourage the use of this form of transport [16]. Electric bikes designed to these regulations require no licence, no insurance, there are no age limits, and they do not even require personal protective equipment (PPE) such as helmets. Other countries within the EU are pushing to extend these regulations, for example, the Netherlands have created new laws to allow the use of faster e-bikes on specially designed E-bike highways [17].

6.2 Electric bike and trailer usage

Often the weather, particularly in the UK, is used as a reason for people to use a car instead of a bike. However, studies have been carried out in countries with similar climates that show that people with an electric bike are more likely to use it as a form of transport [18]. For those who cannot afford the upfront cost or do not want to store or deal with the maintenance of an e-bike, there are local hire schemes, which is making e-bike use available for everyone [19].

In addition, E-cargo bike-share schemes are now available, and case studies have been carried out in Basel, Switzerland [20] [21]. This study shows that cargo bike use is higher in areas that have limited car parking spaces and that young people are more likely to use the service [20]. Another case study mentions that other road users are more likely to avoid a bike towing a trailer, allowing more space during overtaking manoeuvres [21].

6.3 Market research

The last-mile delivery market leaders for volume and weight are LGV. However, studies show that this is an inefficient use of time and costs are high. LGVs are parked at the kerb for longer than they are moving, with delivery drivers spending time walking and sorting between deliveries, as the loading area inside these vehicles is often too big for the cargo [22].

FIGURE 3 - "WAREHOUSE ON WHEELS"

LGV manufacturer Ford has recently proposed that a "warehouse on wheels" would be better placed to conduct deliveries, working alongside couriers on bikes to create an efficient delivery system. With the LGV only being used for "high load, less congested environments" [23].

6.4 Bike trailer specific market research

Non-powered bike trailers are the biggest category of bike trailers and are generally designed for use with a non-powered bike. Due to this they are lightweight and used to carry low volume lightweight objects [24]. Most have an axle with two wheels at around the midpoint, although there are many single wheel options. None of these types of trailers are advised for transporting heavy or high-volume cargo.

The second category of trailer is powered and generally follow the same design as the non-powered trailer in focusing on being lightweight. Trailers for the everyday transport of children are rarely powered but may be retrofit, outside of usage guidelines. There are a few outliers, which serve the more industrial cargo market. The CarlaCargo have a trike design with a powered front wheel, which can stand by itself for easy loading and unloading and. It has a load capacity of 200kg [25]. The Cargo Eurobox is again a two-wheel design with an 80kg max load limit [26].

6.5 Distinctive idea

There are currently no powered trailers for use within the market that take advantage of regenerative braking. This would decrease the percentage of time the trailer would need to return for charging or battery changes and increase the efficiency of last-mile deliveries.

7. Methodology and Project planning

7.1 Methodology of data collection

The project was broken down into sections with the aid of a block diagram, Figure 2which shows the general overview of the project, with each block being further broken down into possibilities through the ideation process Figure 2. This allowed for a research objectives list for each block to be created, see example Research block in Figure 4, which was used as keywords to search for peer-reviewed papers using Google scholar. Only English language papers were selected. Abstracts were analysed for each paper and important papers were marked using Mendeley reference manager and analysed further. For market research, information was acquired directly from manufacturers.

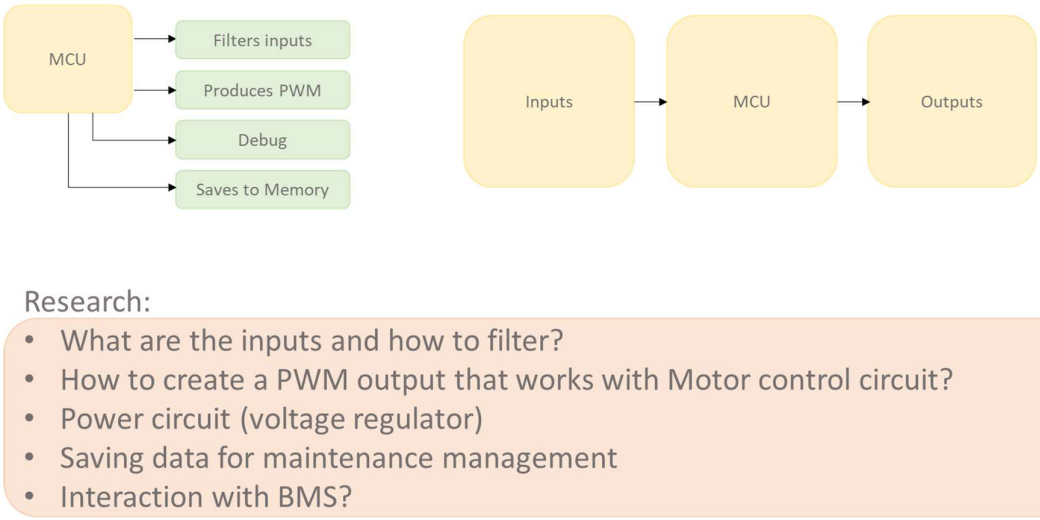


FIGURE 4 - RESEARCH BLOCK EXAMPLE

7.2 Project planning

This project was a POC but an appreciation of the full development process for bringing a product to market is presented in Figure 5. Once the POC stage is completed it is reviewed and a decision is made to repeat the process or move onto the next stage.

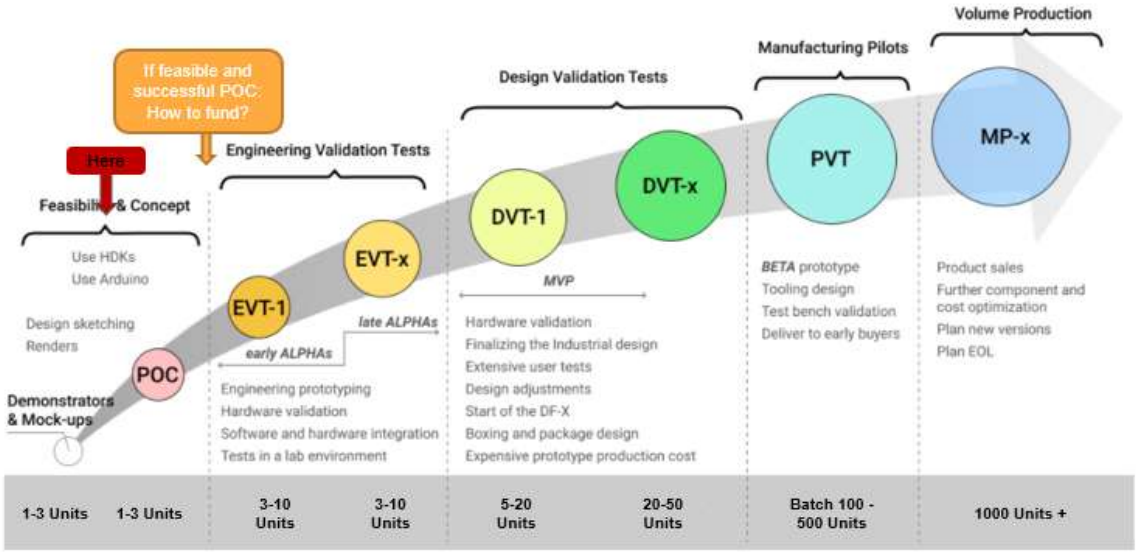


FIGURE 5 - DEVELOPMENT PROCESS [27]

FIGURE 6 - PROJECT SCHEDULE

This project was designed in sections and a schedule was created, Figure 6. This schedule was adhered to from the start of the project and was reported on in the interim report. The project timelines have changed, as parts of the project were investigated in more detail and other sections of the project were relegated to next steps. The aims and objectives of the project were updated as the project progressed. Challenges were forecasted to provide an opportunity to be resolved, and unforeseen challenges were added to the list (see Table 1).

Number	Challenge
1	Component expense – Mechanical, batteries, motor etc.
2	Mechanical expertise for eCargo Trailer chassis design
3	Component sourcing issues
4	Design stage longer than expected

TABLE 1 - CHALLENGES

Given that expected challenges such as Number 1 and 2 in Table 1 were encountered, it was decided that the scope of the POC would be to focus on the electronic control of the motor, rather than providing a comprehensive prototype. Component sourcing issues were also expected, with the global shortage of electronic chips. Changing components involves changes to the PCB design and changes to software, which ultimately required longer time for design solutions to be developed.

The schedule shows an overview of the timeline, to keep track of the day-to-day tasks weekly Google slides was used, Figure 7.

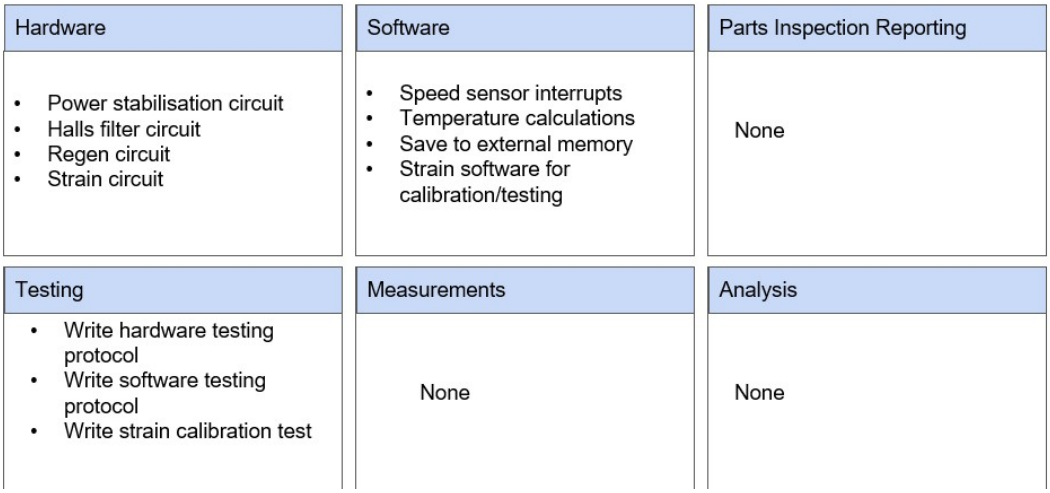


FIGURE 7 - PROJECT UPDATE SLIDE

All the project management slides were kept together in the same presentation and were used to update progress to the project supervisor during meetings.

8. Design methods discussion

This section details the research and design methods for each block in Figure 2. The theory behind the methods is discussed before implementation is explained and tested in the following sections.

8.1 Motor Control

A motor control unit for Brushless Direct Current (BLDC) consists of 6 metal-oxide-semiconductor field-effect transistors (MOSFET) in the configuration seen in Figure 9. These MOSFETs are used to control the direction of current flow through the coils to pull and push magnets in the rotor and rotate the motor. There are 6 of these phases for moving the motor clockwise and using the low side MOSFETs in a different commutation allow for regenerative braking.

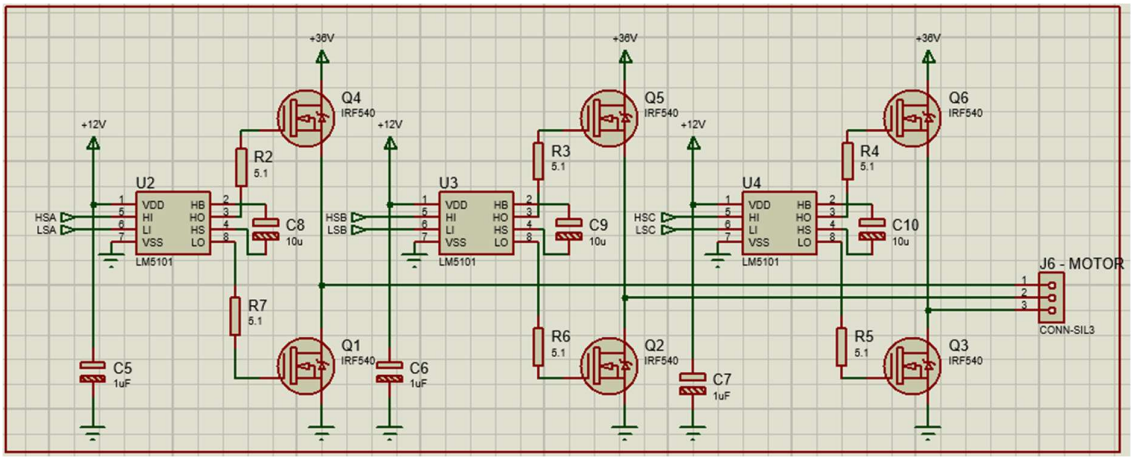


FIGURE 8 - MOTOR CONTROL CIRCUIT

Figure 9 shows one of the phases, where Q4 and Q3 are switched on by the Arduino, allowing current to flow from the supply to ground, in this case via coils A and C. The rotor rotates

around the stator, so the magnets line up with the opposing pole of the magnetised coils. To move the rotor again, the next phase needs to be activated by the processor.

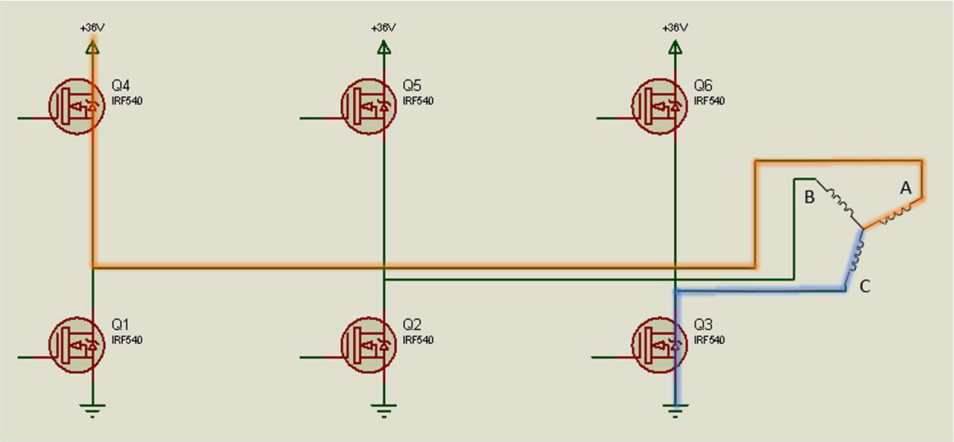


FIGURE 9 – PHASE 1

The next phase can then be magnetised Figure 10, the rotor magnet pole that was being pulled towards coil A is now getting pushed away, and another pole of the rotor is being pulled round to coil B, rotating into the next position. The rotor position can be detected by halls sensors, which detect the position and pole of the magnet as it moves past the sensor. A sensorless approach can also be taken by detecting the back electromagnetic force (EMF) at the centre point of the coils. This leads to the trapezoidal phase graph, where the phase gets switched at the zero points.

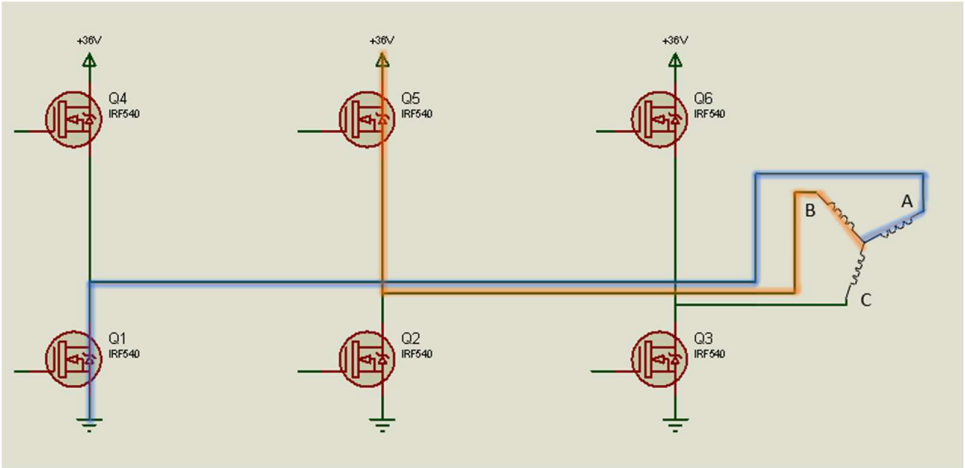


FIGURE 10 - PHASE 2

The torque and speed of the rotor can be controlled by using PWM on either the high side or low side MOSFETs. A higher percentage of duty cycle moves the rotor towards the magnetised coils quicker than a low duty cycle. Commutation graphs show the order at which the coils need to be magnetised in relation to the halls sensor outputs. The graph below shows the commutation steps for a halls sensor spacing of 60 degrees, Figure 11.

The high current that runs through the MOSFETs can be harmful to the processor. Motor drivers need to be placed in between the processor supplying the PWM signal and the MOSFET. The motor driver and MOSFETs need to switch at a very high rate, upwards of thousands of switches per second. The rate of the switching can roughly be estimated when looking at the motor datasheet and calculating the inductance and resistance of the coils and allows the time constant to be calculated. The period of the PWM frequency needs to considerably higher than the time constant to reduce torque ripple. The voltage drops across the inductance rather than the resistor. This keeps the current flow smooth, lowering the peak current and reduces losses across the circuitry [28].

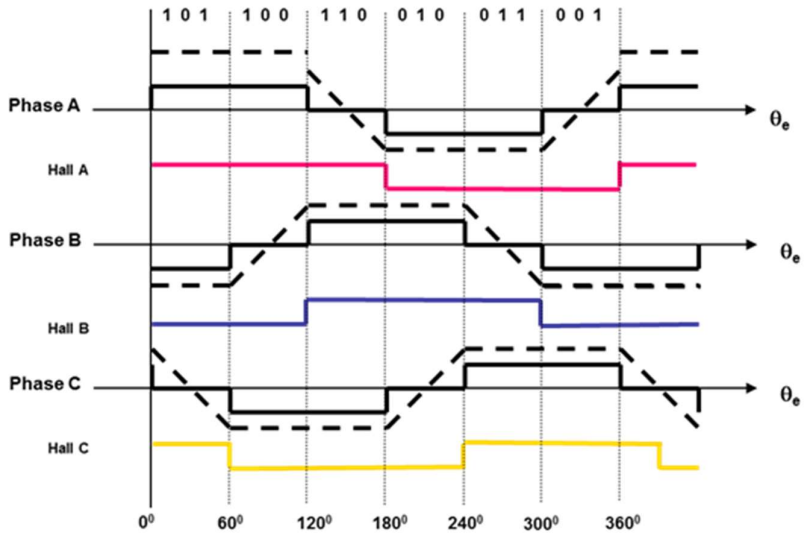


FIGURE 11 - COMMUTATION STEPS [28]

The magnetising of the coils by PWM and reading of the halls sensors is controlled by the processor.

Regeneration commutation does not follow the steps in Figure 11. Instead, the MOSFETs switch in a different order, to take advantage of the inverse torque by converting the kinetic energy into electrical energy [28].

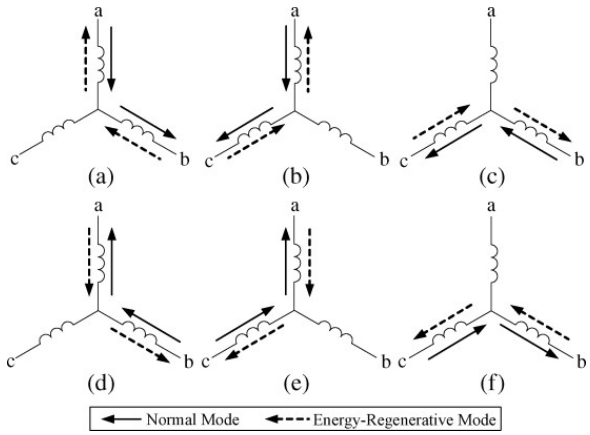


FIGURE 12 - REGENERATION COMMUTATION [29]

The coil sets that are energised in forward motion are again utilized in braking and regeneration. For the boost convertor method regeneration method only the low side MOSFETs are used, Figure 12Figure 12 - Regeneration commutation [29]. Again, the halls sensors are used to detect the rotor position, so that the correct set of MOSFETs are switched on. The PWM is used in this mode, with a higher percentage of duty cycle generating more braking.

8.2 Regenerative braking

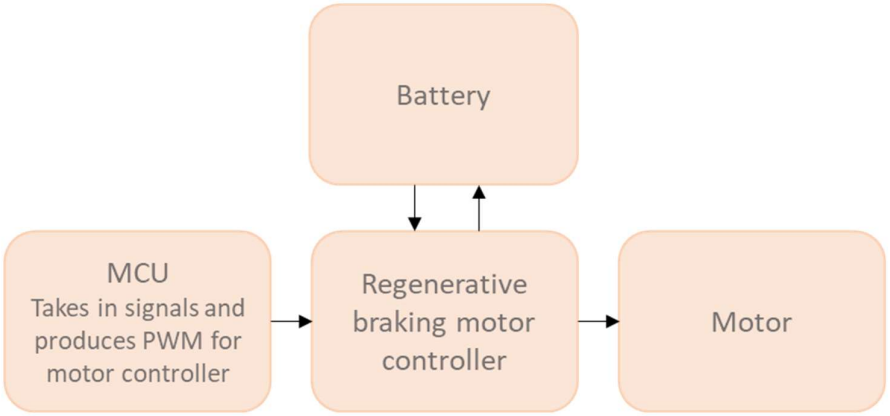


FIGURE 13 - REGENERATION BLOCK

Regenerative braking can increase the range and is a good way of reducing costs, instead of adding a bigger battery to gain a similar range. It also reduces the need for mechanical braking, particularly at higher rotational wheel speeds which can cause excessive brake pad wear. Regenerative braking methods can generally be split into 2 categories. The first category uses

the existing motor driver components, and is generally the method used when size, weight and cost are factors, The second category needs additional hardware, which adds complexity, cost and weight, either by using a supercapacitor or by adding converters [30]. Each of these methods has differing levels of efficiency of energy recovery.

By utilising a DC-to-DC buck or buck-boost converters the energy can be stored even when the generated voltage is less than the battery voltage [31][32]. This method requires extra circuitry and adds weight. The supercapacitor method involves adding an electric double layer capacitor to the circuitry. The use of supercapacitors in regenerative braking is extremely efficient, however, this is with a high braking rate. These systems are normally used in electric vehicles due to the increase in size and weight [33]. They are also expensive, so are not seen regularly in low-cost electric vehicles [34]. For lightweight, low-cost electric vehicles using existing circuitry is the better method [35]. This can work by utilising the motor circuit as a buck converter via the diodes in the insulated-gate bipolar transistors (IGBT) and the charging and discharging of the coils in the motor [36]

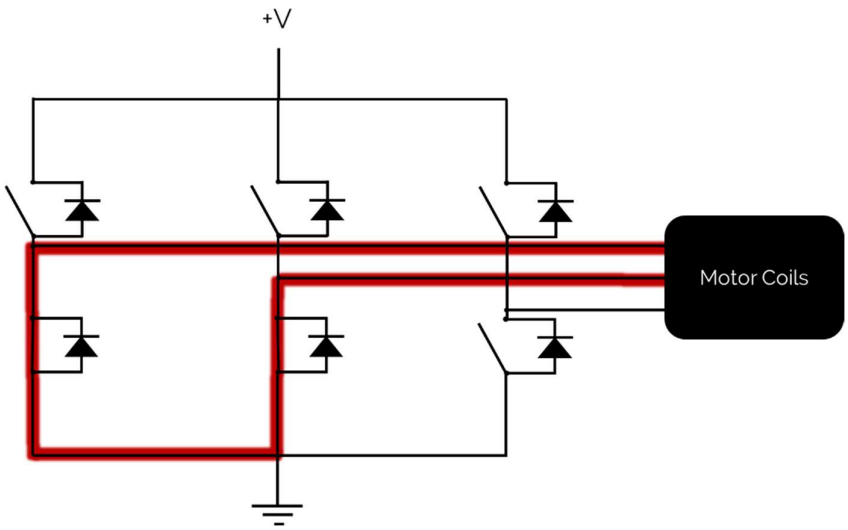


FIGURE 14 - INDUCTOR CHARGING

If the lowside MOSFET switches are switched on, a current is created in the inductive coils in the motor, Figure 14. This current is allowed to build up before the PWM duty cycle switches the MOSFETs off. The inductive coils now discharge and current now flows back through the diodes into the battery, Figure 15.

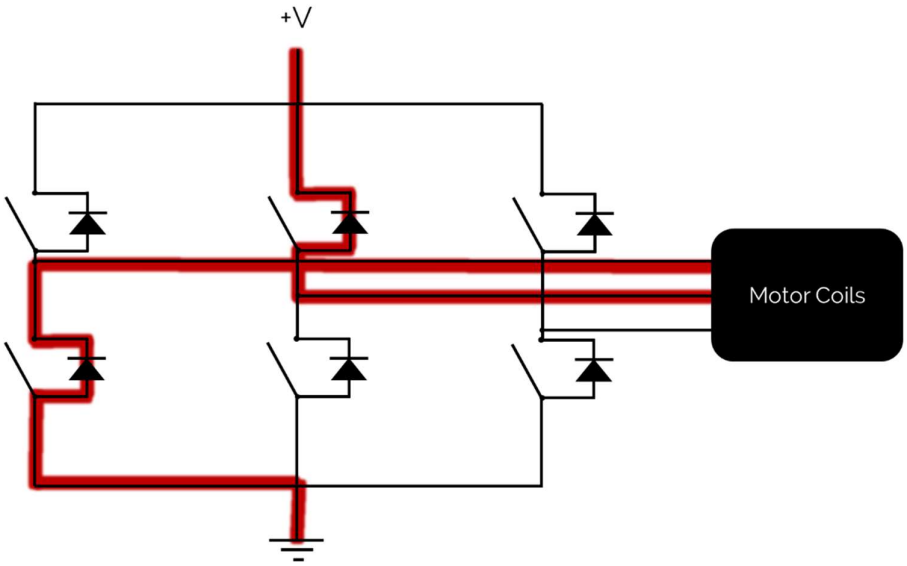


FIGURE 15 - INDUCTOR DISCHARGING

A longer duty cycle gives the coils more time to charge, which increases the time that the eCargo Trailer is braking. A 100% duty cycle means the eCargo Trailer is braking at full capacity which is 100% of the time. However, at a duty cycle of 100% there is no energy getting back to the battery. Maximum regeneration occurs for the period of discharge time that the back EMF is above the battery voltage, which even for the boosted back EMF is only or a short time after discharging begins.

Some of the limitations of regenerative braking include poor braking performance at low motor speed, as the back EMF created is low; the use of a direct-drive motor. which makes the eCargo Trailer harder to move when not powered, and the tendency of overcharged batteries to switch to dynamic braking rather than energy recovery.

The eCargo Trailer is intermediately positioned between an electric bike and an electric vehicle, with bike speeds but with increased weight. Calculating recovery is complex and can vary massively depending on the environment-specific usage. For example, in urban locations, with traffic and junctions that require heavy braking and accelerating, the recovery potential is higher than in rural locations, and it is the highest for hilly terrain. When fully loaded the eCargo Trailer can create a lot of energy while moving downhill, allowing for massive regeneration potential. A heavy trailer creates more force under braking, which is translated into higher energy recovery potential [37].

While travelling downhill the eCargo Trailer quickly gains the maximum speed of 25kph, and the energy created from holding the eCargo Trailer back from accelerating beyond this speed can directly be transferred back to the battery. In the US market the maximum speed is higher at 20mph [38], this means the rotor magnets are passing the coils faster creating a larger back EMF, which allows further recovery potential [37]. Electric bikes have been range tested [39], using various methods of regeneration, adding up to 17% to the overall range. Values in this range have also been seen in real world testing [40].

8.3 Motor

One of the major requirements for regenerative braking is the motor gear setup. The motor needs to be able to use momentum when braking to act as an electric generator, pushing the magnetised rotor past the coils. The back electromotive force generated from this motion requires a direct drive motor and can be used to re-charge the battery via various methods which are discussed in detail below,

FIGURE 16 - HUB MOTOR (GRIN TECHNOLOGIES) [41]

Direct drive hub motors are commonly used in e-bike applications. Hub motors have the advantage of versatility, being easily retrofitted to almost any bike. They are also low cost compared to the more complex mid-drive motors that drive the more expensive e-bike options. The coils in these motors are generally set up in a star rather than delta. Hub motors

alter the handling on electric bikes, by moving the centre of gravity away from the centre point of the bike.

The momentum generated while braking forces vehicles into forward lean, moving the weight and force through the front wheel [37]. This phenomenon dictates the positioning of the motor at the front of the vehicle. Whilst placing the hub motor in the front wheel of an e-bike makes the steering harder, it has little impact on the user experience of the eCargo Trailer, whilst proving better efficiency for regenerative braking, justifying the use of forward motor positioning in the design of the eCargo Trailer chassis.

On a conventional setup the magnets sit inside of the stator windings, while hub motors are commonly designed with an external rotor, Figure 17. The larger external rotor typically means higher inertia, which overcomes torque ripple and increases the air gap radius, which adds more torque. Magnetic flux can be further increased by using the extra space on the rotor to add more pole pairs. The increase in inertia increases the mechanical time constant, the time to reach 63.2% of the motors finishing speed, which means the motor takes longer to accelerate than conventional systems. Hub motors are particularly useful where the weight of the motor is not important, allowing more magnets to be placed on the rotor. Larger motors, with larger diameters can be selected to allow more space for coils and pole pairs, increasing the torque of the motor. High torque is vital to move the eCargo Trailer at lower speeds.

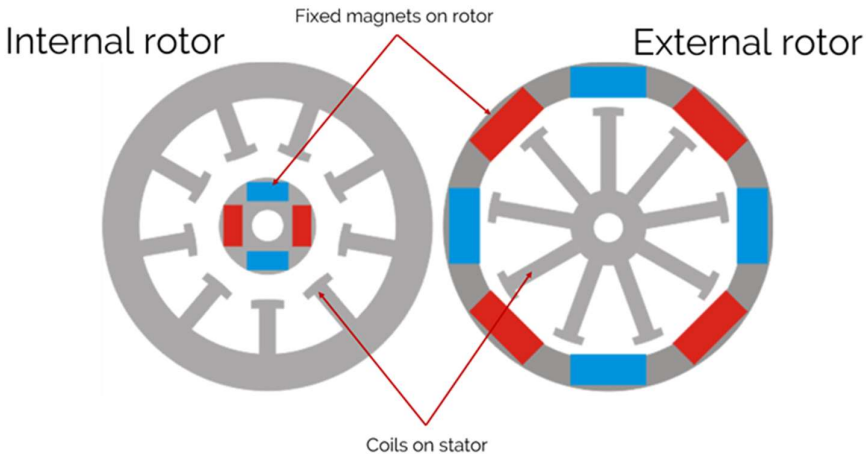


FIGURE 17 - CONVENTIONAL VS EXTERNAL ROTOR [42]

As the motor is large and slow, compared to other applications of BLDC motors, such as drones which can spin at thousands of revolutions per minute [43], it needs many magnetic pole pairs and corresponding motor windings to create enough torque to move the motor. The

axle of the hub motor can contain a torque sensor, which allows the efficiency of the motor to be calculated.

8.4 Battery

Many battery options could be chosen for this project, including both designed in house or by purchase.

Battery design is influenced by factors such as size, capacity, portability, especially to reduce the need for dedicated charging stations that limit larger electric vehicles [44]. Purchased batteries are advantageously certified for safe transport and environmental checks such as water ingress and dust protection, but cost more [4]. The other option is designing the eCargo Trailer so that it can take advantage of battery share opportunities, such as German company Greenpack [45]. This comes with some limitations as battery sizes and connections as well as Battery Management System (BMS) may vary but could be added as aftermarket options.

Battery management systems control the input and outputs from the battery. Batteries can be run too flat, so when the battery is running low on charge, the BMS can switch the power off, allowing the battery to protect itself. The BMS also stops the battery from becoming overcharged, both when charging from the mains or from regenerative braking.

As for chemistry, lithium-ion batteries are currently the market leader for e-bike applications due to their low weight [46]. Commonly used battery voltages are usually 36V or 43.2V and can provide continuous current of 20 Amps. This gives an estimated continuous power of around 720 to 864 Watts. Battery capacity is often measured in Amp hours (Ah) and for normal electric bikes is between 10 and 25 Ah [47] and is limited by available space and weight, which are both less important on the eCargo Trailer. Unfortunately, the capacity cannot directly be converted to a maximum distance of travel per battery discharge. This is affected by outside influences such as wind, terrain, weight, urban or countryside environment and temperatures.

8.5 Microcontroller Unit

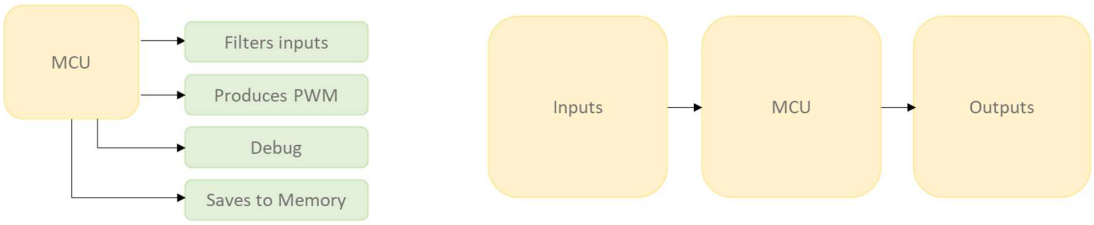


FIGURE 18 – MICROCONTROLLER UNIT (MCU)

The microcontroller unit (MCU) uses PWM to control the motor speed and for the regeneration, by quickly switching the MOSFETs on and off in order via their drivers. The MCU filters inputs from inside the motor and external sensors on the wheels, calculates the efficiency of the motor and saves the data to memory, which makes it a vital focus point for this POC. Arduino Uno devices are often used for testing during the POC process [48]. They can supply multiple PWM signals needed for driving the motor. They can take in analogue as well as digital signals and have the required processing power to analyse the data gained from sensors and save that information to external memory [49]. Importantly, they can be easily connected to and programmed repeatably to add improved firmware and for debugging. These devices have room for expansion to add extra features, such as the electric brake, however for cost effectiveness they would ideally be replaced with an alternative processor and circuitry at the production-intent stage.

8.6 Sensors

Various sensors that are needed for the operation of the eCargo Trailer. An overview of the types of sensors necessary for the project is provided below.

8.6.1 Strain Sensor

The load cell sensor is placed on the arm connecting the eCargo Trailer to the bike. It senses the force of the eCargo Trailer being pulled by the bike, instructing the controller to power the motor and it senses the force of the eCargo Trailer pushing the bike when decelerating, allowing for regenerative braking to occur. A common method to calculate the force is by using a strain gauge. A strain gauge is a relatively simple design consisting of thin wires contained inside foil. The foil is glued on to the surface of the material to be tested. If the material is stretched, in this case the eCargo Trailer being pulled by the bike, the wires stretch at the same rate. This generates a change in resistance, which can be converted to a force

reading. Conversely, if the material is compressed, the wires become shorter, providing a new force value.

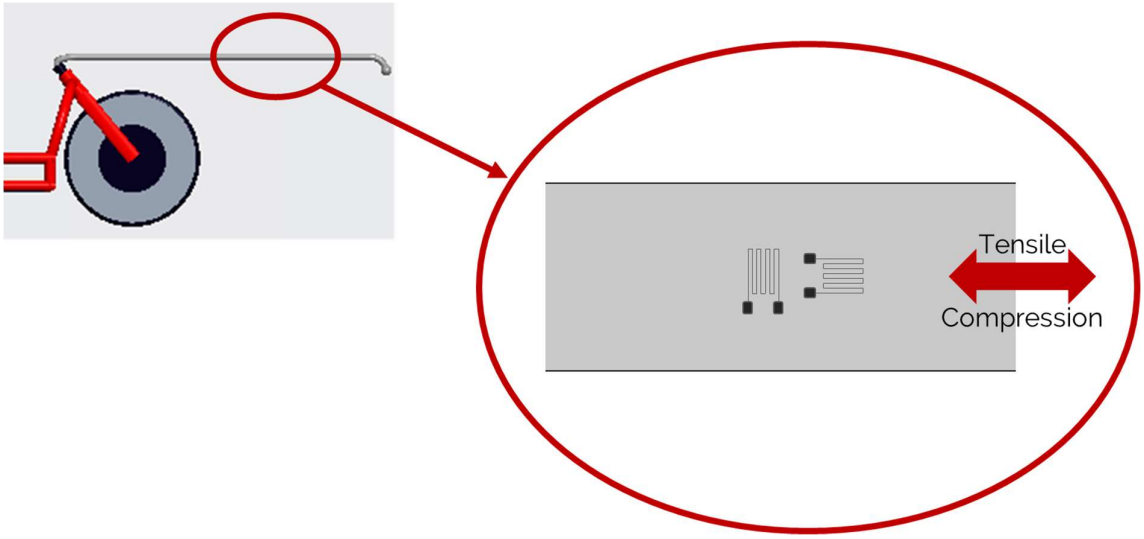


FIGURE 19 - COLUMN LOAD CELL

To detect axial forces, tensile or compression, a column load cell is set up on the connecting arm, Figure 19. A half bridge torque sensor is required on both sides of the connecting arm, as demonstrated in the cross-section view, Figure 20, This formation allows the forces that are not parallel to the axis to be nullified, by comparing the resistances on each half bridge on either side of the connecting bar. The half bridge setup, by having two gauges perpendicular to each other, allows for thermal affects to also be nullified.

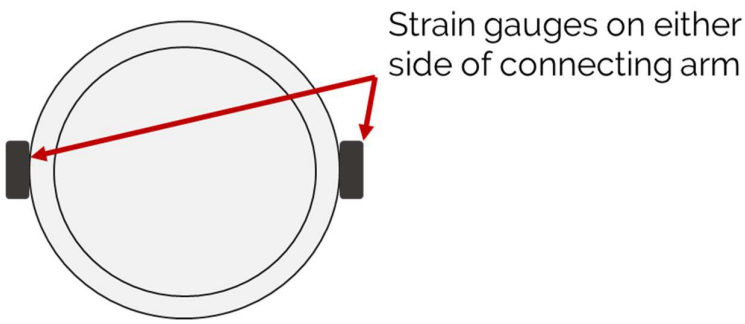


FIGURE 20 - LOAD CELL CROSS SECTION

For the POC a single half bridge strain gauge setup was used for testing, Figure 21 - Half bridge. When axial loading is present along the connecting arm, only one of the gauges is loaded, changing the resistance and voltage at TP1. The outputs TP1 and 2.5V are connected to an

operational amplifier to give a useable signal for the processor. This is explained further in chapter 9.1.1 Circuit Design.

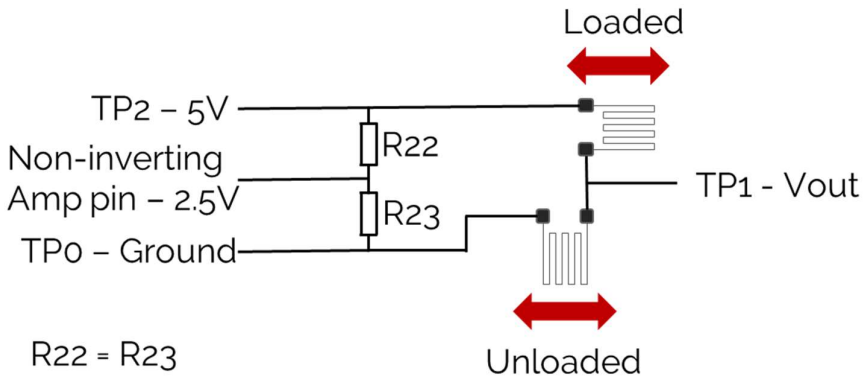


FIGURE 21 - HALF BRIDGE

When the eCargo Trailer is cornering, the strain gauge may also be getting stretched or compressed and generating too much power from the motor at this point may be dangerous and cause a crash. For a production model having locations of strain gauges on either side the arm is vital to predict how the force is being applied.

Other methods of corner detection, acceleration and/or deceleration should be employed alongside the strain sensors. An accelerometer on the eCargo Trailer would give extra insight to direction and when added to the algorithm allows for smoother and safer application of power. Gradient, both laterally and longitudinally, can also be calculated using the accelerometer, which can also be used to optimise the algorithm.

An Omega K-Series strain gauge with a nominal resistance of 1000 Ohm was selected, as this could be tested on an offcut of mild steel and sets the values of R22 and R23 [50]. The choice of material for the connecting arm is beyond the scope of this project but influences the signal due to the strength and elasticity of the material when forces are applied.

8.6.2 Temp Sensor

Temperature provides a great insight to power expended in the motor, battery, and electronics; therefore, temperature sensors are vital for the operational use of the eCargo Trailer. A sensor, which can reach upwards of 150degC would be required and would be ideally positioned on the motor windings inside the stator, as the windings have a large current draw and lose a lot of energy to heat. Another vital location to place a temperature sensor is

close to the battery. The current draw can run close to 20Amps for long periods of time, pulling a lot of power from the battery, and protections should be put in place.

A high temperature motor threshold alert added in the software would save the components from reaching their maximum allowed temperatures saving them from damage and lengthen the life of the product. This feature is also useful to system performance analysis. If the temperatures are consistently high, the system could flag this to the user via a warning LED, allowing the system to be serviced before catastrophic damage to components.

One option for a temperature sensor is the Maxim Integrated DS18B20+ which comes with waterproof connector, has a range between -55˚C to +125˚C, and works through a OneWire communications protocol.

FIGURE 22 - DS18B20+ [51]

Another option is the Texas Instruments LM35 is an inexpensive temperature sensor commonly used in Arduino projects and has a range of -55 to 150 °C.

FIGURE 23 - TI LM35 [52]

The LM35 can be connected directly into the analogue pin of the Arduino, the other pins go to 5V and ground. This allows for an analogue to digital conversion before adjusting the limits of the device to get an accurate reading.

8.6.3 Halls Sensors

Halls sensors generate location data of the rotor inside the stator. Three halls sensors are placed around the stator at either 60deg or 120deg apart, which send almost identical signals except for the third halls sensors' signal is inverted. They are placed close to stator windings so need to be able to handle high temperatures. The halls sensors come already installed in the motor and for most electric bike systems 60deg spacing is used to make cable routing easier.

8.6.4 Speed sensor

The wheel speed can be calculated using the motor speed, as the motor is direct drive. Two methods were selected that both use changes in halls sensor location to determine speed.

The first method uses a logic circuit to flag an interrupt pin every time there is a change in halls output. The number of clock ticks that have passed can be used to calculate the speed.

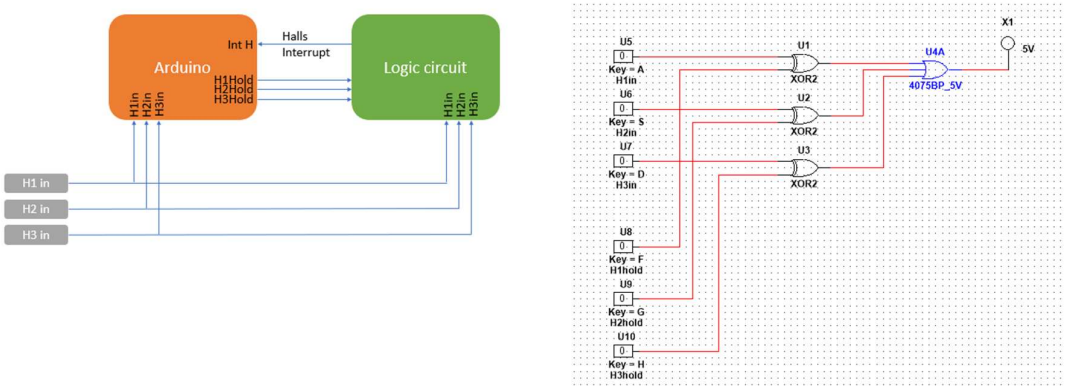


FIGURE 24 - LOGIC CIRCUIT SOLUTION

The old halls values are held on the pin outs, 'Hxhold', and compared against the new hall values, 'Hnin'. When there is a change the intH pin goes 'HIGH' and flags an interrupt. This has the added benefit of only updating the halls value when they change rather than checking using an interrupt or constantly checking in the main loop.

The other method uses only software and consists of using the timer interrupt and calculating the number of steps that have happened since the last interrupt. This is fully described in the software section, 9.4.2.

8.7 Recorded data and maintenance management

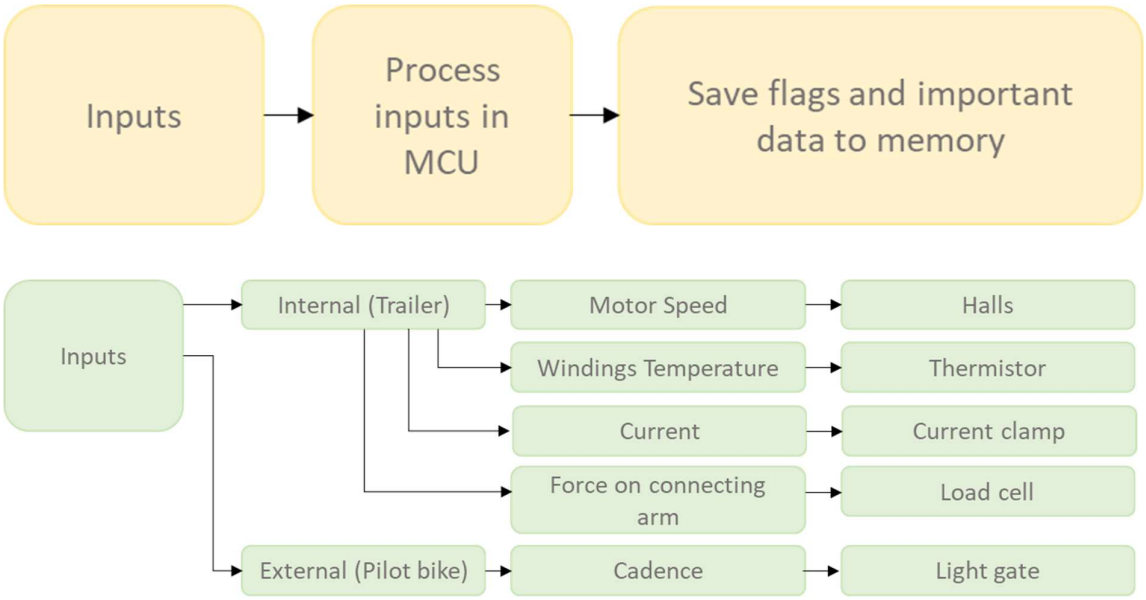


FIGURE 25 - RECORDING DATA FLOWCHART

A lot of data needs to be gathered to get an accurate representation of motor efficiency and performance. Torque output is gained from using a strain gauge on the motor axle. Current draw is calculated, using a current sense circuit at the battery outputs, the motor and wheel speed, are easily estimated based on the wheel circumference and rate of rotation which is acquired from the hall sensor data. And the case and winding temperatures are calculated from thermistor data. The gradient the eCargo Trailer is travelling on can be calculated using an accelerometer but other environmental information is more difficult to ascertain with wind resistance and rolling resistance being more difficult to calculate

A good method of rating performance is by comparing motor efficiency and winding temperature [53][54]. Using motor mapping data from testing allows for the comparison of estimated temperature versus known temperature in the motor windings. The data can be compared with the mapped data to estimate performance, with higher-than-expected temperatures signifying a potential issue. These calculations can be carried out by the Arduino in real-time and saved to electrically erasable programmable read-only memory (EEPROM) memory [55]. I²C protocol has a maximum clock speed of 400kHz in fast mode, while the

processor clock speed is much faster, close to 16Mhz. Therefore, it is more efficient for data to be calculated by the processor before being distilled into as few bytes as possible and then be transferred to the memory IC.

If the torque output from the motor is estimated or known, then the losses can be calculated after the current at the input is detected, Table 2 - Efficiency calculations. A current sensing IC can be used across the battery terminals to gain voltage and current data.

TABLE 2 - EFFICIENCY CALCULATIONS

$Power\ in = Power\ out + Power\ losses$	$\tau = \text{Torque (Nm)}$
$Pin = Pout + Plosses$	$\omega = \text{Angular velocity } \left(\frac{\text{Radians}}{\text{second}}\right)$
$V\ .I = \tau\ .\omega + Plosses$	$\text{V} = \text{Input voltage (V)}$
$Efficiency = \frac{Pout}{Pin} = \frac{\tau\ .\omega}{\text{V.I}}$	$\text{I} = \text{Input current (Amps)}$

Current sensing is needed to gain overall efficiency and is sometimes included in the BMS data. However, if it is not available then there are a few locations where current sensing can take place that give an insight into what is happening in the control circuitry.

Current sensing can be carried out using a series current sense amplifiers, such as the AD8219, [56], which are designed specifically for motor control current sensing. An overall efficiency can be calculated by taking the current draw at the battery terminal and includes the processor and all other components in the controller. It can also be placed on each phase wire going to the motor, which gives insight into potential losses in the motor.

8.8 eCargo Trailer chassis

Some initial design specifications that need to be met influence the design of the chassis.

- Self-support itself and cargo
- Typical footprint of cargo sizes can be loaded
    - 1 euro pallet (1200x800mm)

    - 4 euro boxes (600x400mm)
- Easily manoeuvrable
- Low centre of gravity for stability and loading
- Simple design for easily assembly and easily maintained by local bike shop
- Battery can be quickly changed

These specifications led to a trike type trailer, Figure 1with several options on how to drive the motor of the eCargo Trailer, Figure 26. A motor connected to the rear axle driving both rear wheels is too complex for a bike shop to maintain and it raises the height of the load bay. In comparison, hub motor options can be easily maintained. As described in the motor selection section 8.3, a large hub motor should be able to power the eCargo Trailer sufficiently.

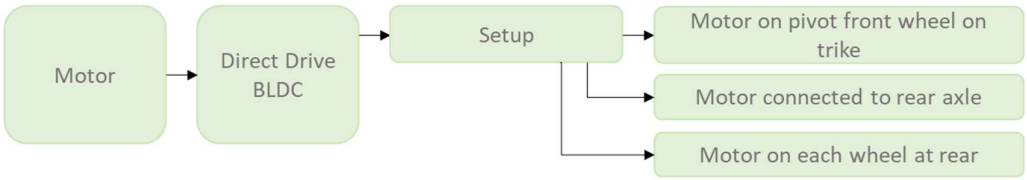


FIGURE 26 - TRAILER DESIGN

For the first stage of the mechanical optimisation Creo Parametric was used to sketch a design in CAD, Figure 27. Finite Element Analysis (FEA) uses a CAD model to optimise design to a range of loads the eCargo Trailer encounters. Currently 40 mm tube diameters have been used in the design, however the thickness and diameter of these tubes may need to be enlarged or reduced depending on the FEA. By running through several iterations of the FEA process an optimised design can be completed, reducing time and cost formultiple iterations of real life testing.

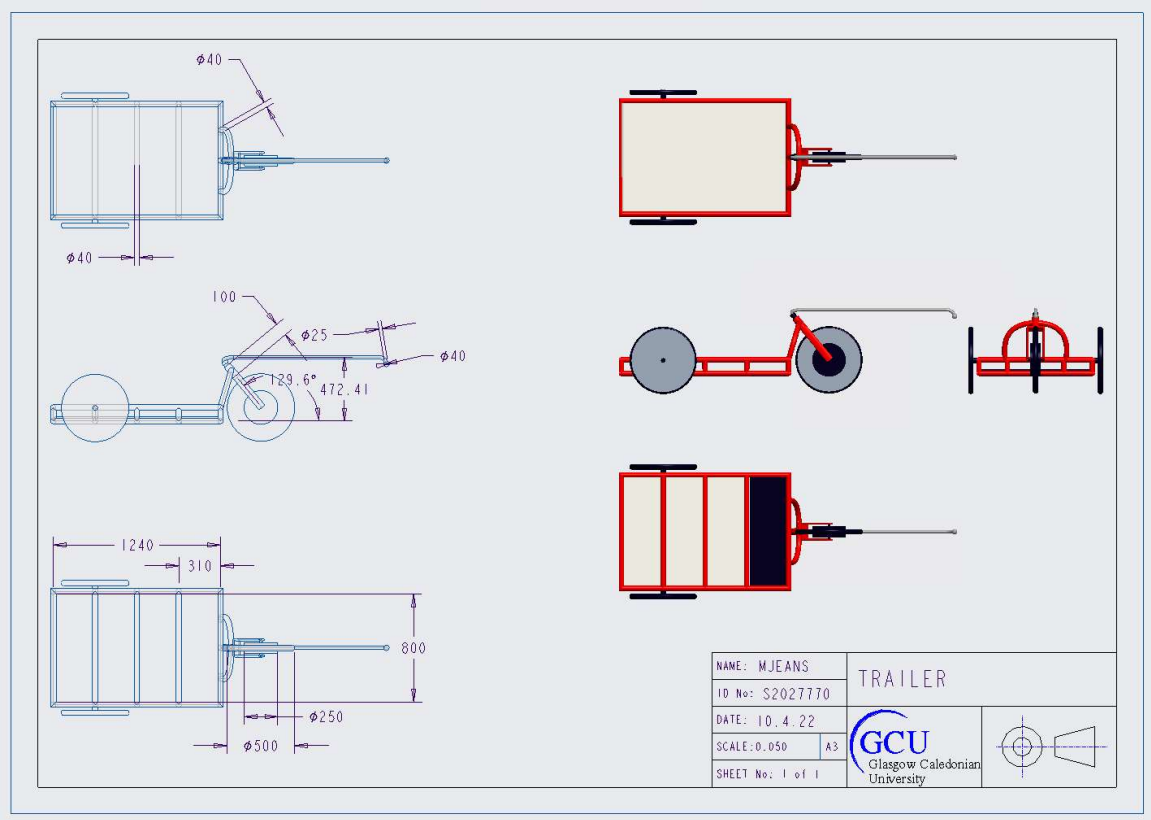


FIGURE 27 - 2D DRAWING

The eCargo Trailer is relatively short as the fork, which holds the wheel and allows it to rotate to turn, is attached to the main frame at a relaxed angle compared to a bicycle. This may need further optimisation, as the angle of the fork may affect the steering capabilities of the eCargo Trailer

Large cavities, 800 x 270 x 40 mm, are strategically located underneath the chassis between the supporting tubes. The cavity closest to the front wheel would be optimal for a waterproofed compartment that can contain the electronics and battery.

Reflective material is placed on the side and rear of the eCargo Trailer, along with lights at the rear, to allow the eCargo Trailer to conform to safety regulations [4]. Additional stringent external testing regulations need to be considered before the eCargo Trailer can legally be used on public roads.They are enforced by the regulatory body and need to be completed before the product is fully certified and are described below.

8.9 Regulations and safety features

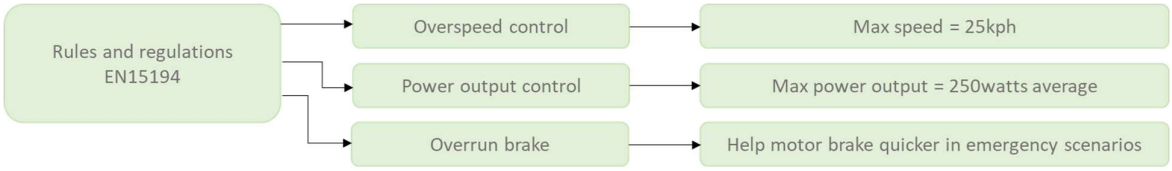


FIGURE 28 - SAFETY MECHANISMS

8.9.1 Regulations

The regulations in place in the UK and Europe are covered in "Cycles - Electrically power-assisted cycles - EPAC Bicycles" EN15194:2017 [4], This document features structural strength rules and electromagnetic testing guidelines and details the requirement for operational lights and a mandatory brake for emergency scenarios. These policies would have to be closely adhered to during the testing of the production-intent model. Of major importance to this stage of product design is a stipulated maximum speed limit of 25 kph while under power, which necessitates the use of a speed sensor. The regulations also require a mandatory brake that can bring the vehicle to a stop for emergency scenarios.

8.9.2 Bike to eCargo Trailer interface

As previously mentioned, regenerative braking only reduces speed effectively at higher motor speeds, as the EMF created is not great enough at lower speeds. Braking at lower speeds, when parking or emergency braking needs to be designed into the product and be run through rigorous testing processes. Two systems could be employed to slow and stop the eCargo Trailer.

Overrun brakes are a mechanical method of slowing the trailer when the speed is faster than the bike speed. The force created by this speed differential pulls a lever that actuates the brake. To adhere to the regulations, this type of brake would have to be implemented in the design.

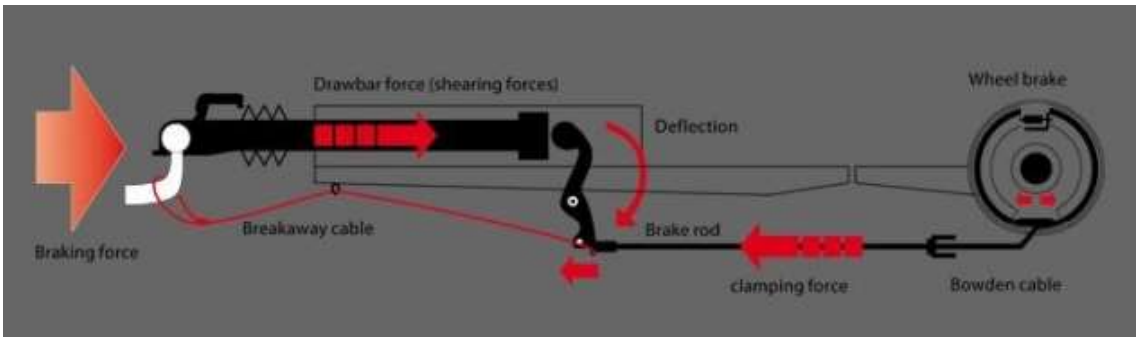


FIGURE 29 - OVERRUN BRAKE [57]

Electric brake controller systems are also used in industry to slow heavy trailers [58][59]. This type of system can be used to slow the eCargo Trailer efficiently by using a variety of techniques, including the sensors that are already included in the design discussion. These systems are programmable and can be designed to improve safety by using techniques including anti-lock braking.

9. Design Implementation

This section details the circuit and PCB design, along with the software implementation of the discussed methods. The solutions for circuit hardware are considered and discussed and the software is discussed in detail.

9.1 Circuit and PCB Design

There is an economical benefit to purchasing a complete motor control unit for a standalone project. However, the production stage would benefit from the design of the PCB at the POC stage. There are many motor controllers that either have extra functionality or not enough to cope with this project. Hence, during the POC phase an Arduino Uno, commonly used for the design phase, is used to control the drivers. For the end design a more efficiently chosen processing unit should be used to reduce cost, and potentially also space on the PCB. The current schematic shows a motor controller unit with an Arduino Uno and extra circuitry to be able to run the functionality needed for the POC, Figure 30. Proteus was used for both the circuit design and PCB design.

9.1.1 Circuit Design

The circuit design was facilitated by downloading the component database and adding more unique components, such as the Arduino to the library. The schematic is arranged into sections, Processor, Connectors, Motor control, Power, Memory, and Sensor inputs Figure 30.

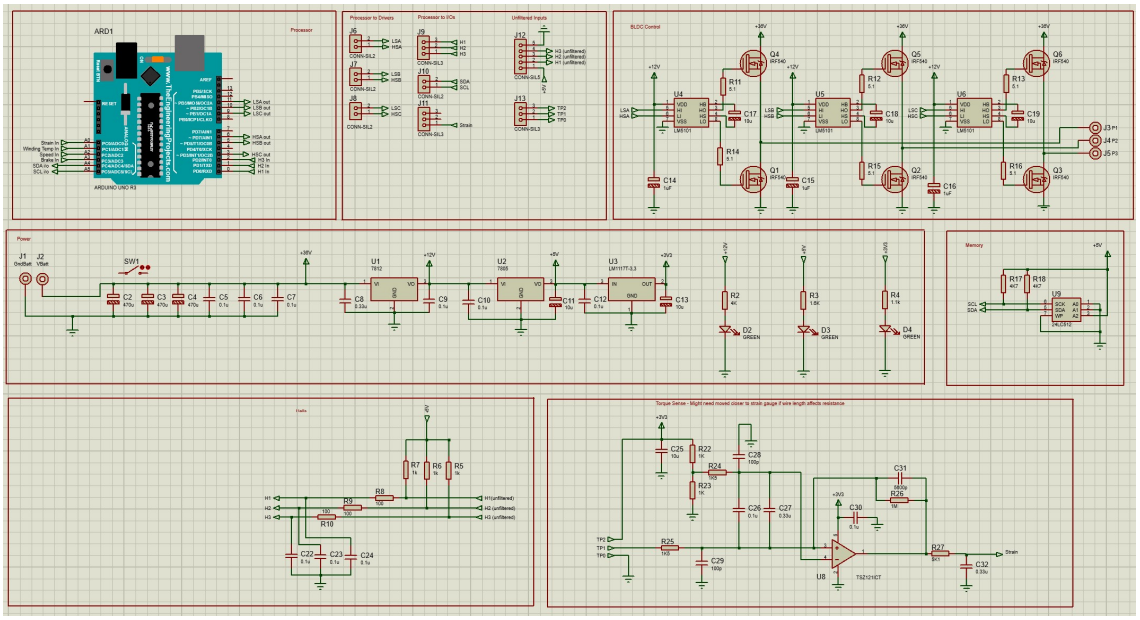


FIGURE 30 - FULL SCHEMATIC

The ICs need to be protected from the high voltage supply used to drive the motor. Voltage regulators were used to reduce the voltage from the battery to 12v, 5v and 3.3v, respectively.

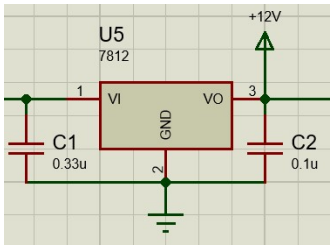


FIGURE 31 - 12V – MOTOR DRIVER POWER SUPPLY

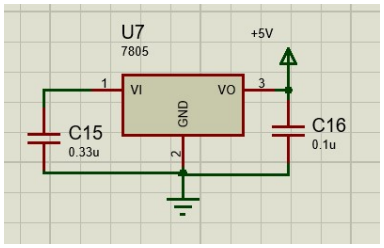


FIGURE 32 - 5V – ARDUINO POWER SUPPLY/HALLS SENSORS

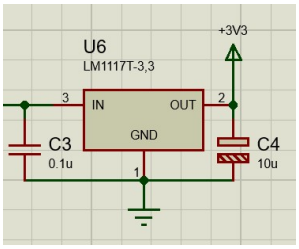


FIGURE 33 - 3V3 – EXTERNAL SENSORS

The expected current draw from the components powered by each of the regulators was calculated from the data sheets. The operating current from the motor drivers is 1.5mA, [60] and the maximum current draw allowed from the L7812 is 6mA [61]. The 5V supply and 3V3 supply are well within their limits, as they are currently powering low current draw ICs. For a production ready PCB, the smaller voltage regulators need to be recalculated, as extra sensors such as an accelerometer and gyro sensor need to be added and the 12V supply is used to power powerful LEDs, so that the eCargo Trailer can be seen at night. The regulator datasheets, [61][60] were used to set up each device. All the regulators have internal thermal shutdown, and all could be connected to a heatsink to further transfer away heat.

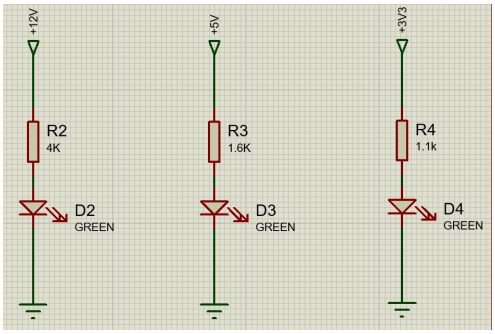


FIGURE 34 - POWER INDICATOR LEDS

LEDs are connected to each power rail to allow for quick checks to see that the board has power. For simplicity these are through-hole components but could be replaced with smaller surface-mount LEDs for a production-ready board.

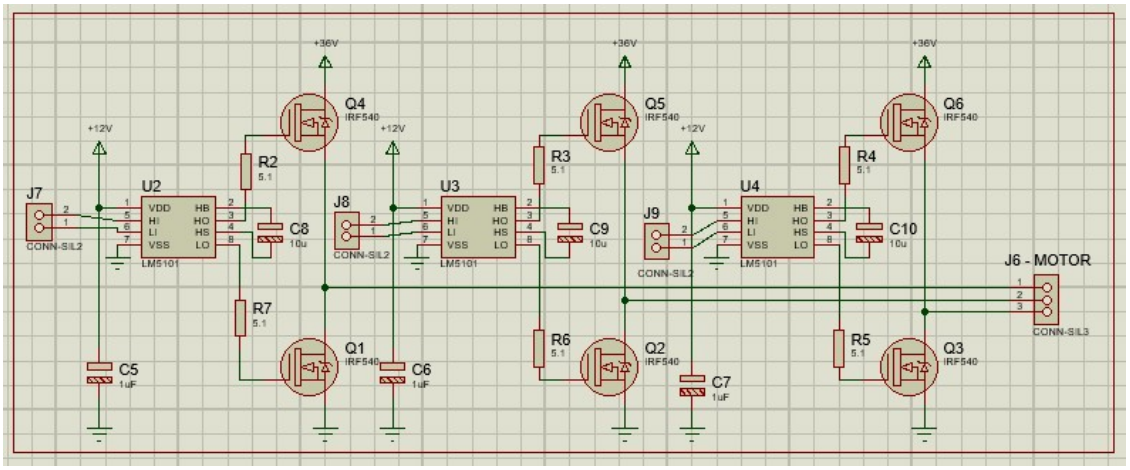


FIGURE 35 - MOTOR CONTROL INCLUDING DRIVERS

The motor control circuit is set up using LM5101 drivers, named as such because of the 5.1 Ohm resistors that connect them to the MOSFETs [60]. The LM5101 reaches well beyond the switching frequency of the 32kHz PWM signal. It can cope with supply voltages more than the 36V or 43.2V supplies normally used by the motor. These drivers have been selected in the preliminary design, as they were immediately available and inexpensive. A smarter driver, such as the DRV8803, would also supply overcurrent protection which is vital to increasing the lifetime of motor components, [62].

The Vishay IRF540 power MOSFETs have a high operating temperature limit and can be attached to a heat sink [63]. The continuous drain current limit is above the max operating current of the motor. The pulsed current maximum current value can reach 110A, which the motor is unlikely to request. The connectors for each phase leaving the board to get to the motor are not to be used during development and instead wires are soldered directly via through holes.

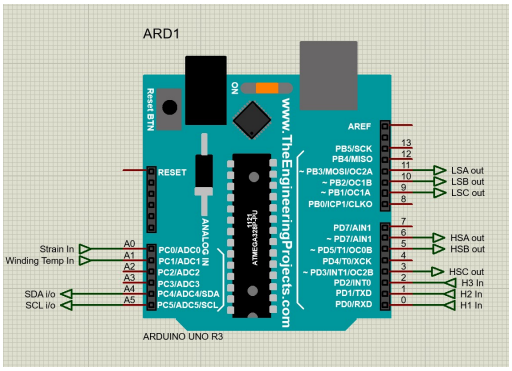


Figure 36 - Arduino with inputs and outputs

The Arduino Uno has been selected as it can easily be used to test the software without needing a PCB. The clock speed of 16Mhz is used to produce the required PWM signal from 3 pins and has still has a timer available to set up an interrupt. It has all the required pin inputs and outputs (I/Os); three halls inputs, three PWM outputs, three digital outputs, SDA and SCL I/Os, and analogue inputs for the strain and temperature sensors. And can add more sensors if needed.

Inputs 9/10/11 produce PWM signals utilised by the LM5101 drivers. A0 and A1 take in the analogue signal created by the temperature and strain gauge and run them through an ADC.

while A4/5 are set up as digital I/Os for saving data to memory. Pins 0 to 6 are used is digital I/Os.

The SCL and SDA pins have been given 4k7 Ohm pull up resistors to a 5V supply as detailed in the datasheet [64].

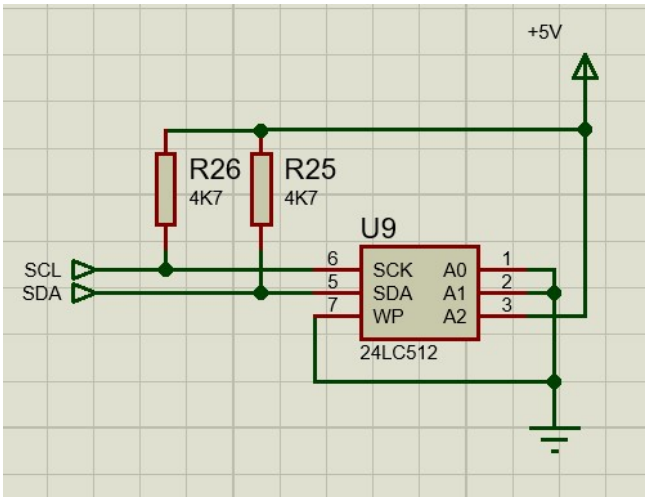


FIGURE 37 - I2C EEPROM

The strain gauge input needs complementary circuitry before the signal is received by the ADC pin of the Arduino Uno. For the POC process only the half bridge on one side of the connecting arm is to be used. The TSZ121 is very low power consuming less than 40uA at 5V, [63] and is set up as an inverting amplifier, using TP2 and supply and ground as a reference. The resistors R22 and R23 complete the half bridge set up and are used as a voltage divider splitting the input voltage and setting the reference voltage of the non-inverting input. R22 and R23 are 1kOhm the same resistance as each strain gauge on the film. R17 and R18 are set at 1.5kOhm to limit the current into the device to under 10mA. The values of R26 and R24 allow for the gain to be worked out and the output can be calculated using Equation 1 - Inverting amplifier.

EQUATION 1 – INVERTING AMPLIFIER EQUATIONS

$$Vout = Vref - \left(\frac{R26}{r25}\right)(Vin - Vref)$$

$$Vref = 2.5V$$

Vout is the voltage that gets passed into the ADC on pin A0. Vref is the resultant voltage after the voltage divider. Vin is the input voltage from the remaining pin on the strain gauge which is TP1.

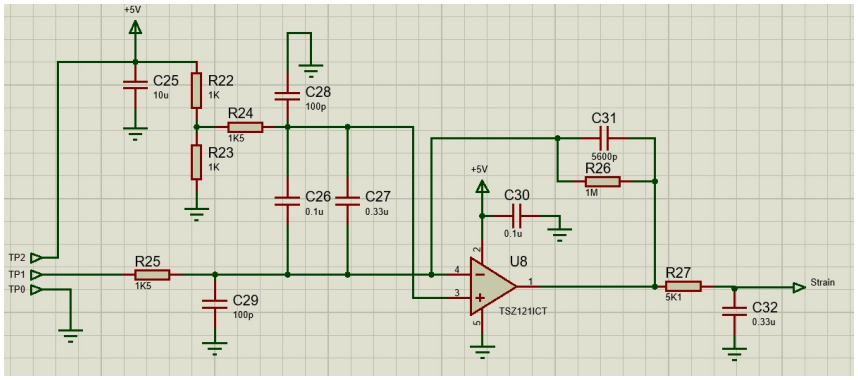


FIGURE 38 – ELECTRIC BRAKE - STRAIN GAUGE AMPLIFICATION

The halls inputs are given pull up resistors and pass through a low pass filter, Figure 39. The low pass cut off frequency is set at ~16kHz, Equation 2.

EQUATION 2 - CUT-OFF FREQUENCY

$$f_{cutoff} = \frac{1}{2\pi RC} = \frac{1}{2\pi(100)(0.1u)} = 15915\ Hz$$

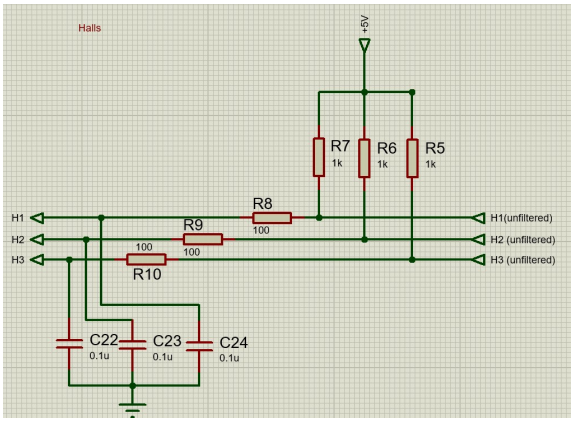


FIGURE 39 - HALLS PULL UP RESISTORS

9.1.2 PCB Design

After completing the circuit design, it was exported to Proteus PCB Layout Builder. The component library was updated with all the footprint and component dimensions. The board

size was selected at 110mm x 90mm, with both top and bottom copper layers. Multiple layer boards were more expensive; however, they were needed to reduce the size of the board and allow all the connections to be made without adding external wiring.

The traces were made as wide as possible for the high current carrying connections. These connections can be improved further on both the top and bottom layer by adding solder on top of the traces. The larger traces were present at the battery connection into the board and went to and from the power MOSFETs.

The traces were made without acute angles of less than 90degrees to improve the quality of the traces that can be degraded during the manufacturing process. The traces were kept as far as possible from each other so signal lines are not affected by current carrying power lines. Proteus allows a limit to be set and a warning to appear if they are too close. Signal traces were kept away from high current carrying traces and. connectors were kept to the outside of the board, particularly the power and phase through-holes. These were designed to be easily accessed for soldering, as they and require a lot of heat due to the thicker gauge of wire.

Connections, larger components, and all the through-hole components in this design were placed on the top layer of the board. On the bottom layer only small surface mount components were placed and through-hole component pins could be cut. Surface mount components are more robust than the through-hole and their height from the board is minimal. This side of the board was mounted on plastic stands into a case.

The version number was added to the corner of the board, so different versions are easily identified.

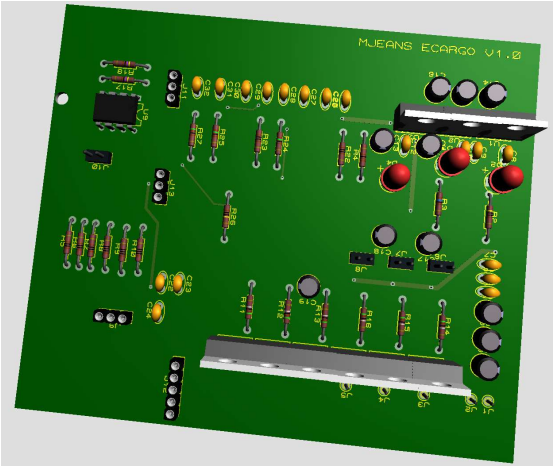


FIGURE 40 - TOP LAYER

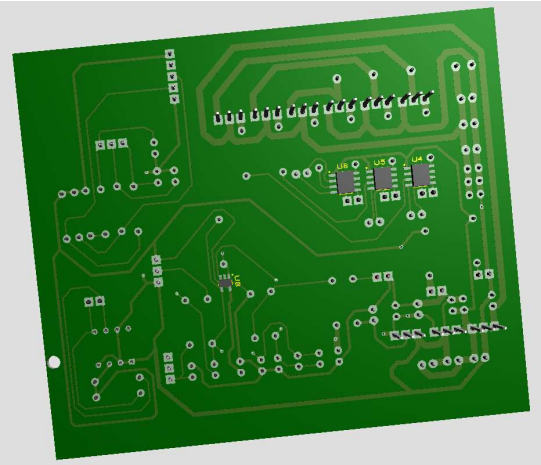


FIGURE 41 - BOTTOM LAYER

The sizes of through-holes and vias were marked on the CAD package exported from Proteus, Figure 40, Figure 41. Gerber files were created, as these are commonly requested by PCB manufacturers, Figure 42 - Gerber file. A bill of materials (BOM) was exported to Excel, Table 3 - Bill of materials (BOM)Table 3.

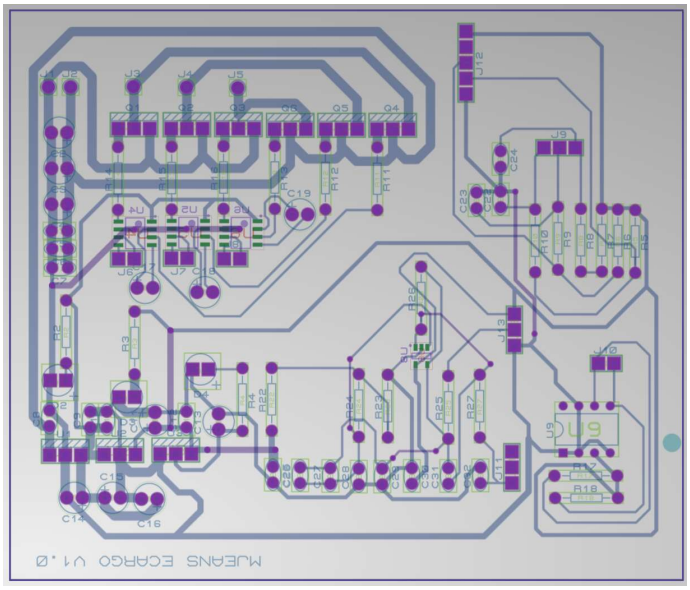


FIGURE 42 - GERBER FILE

9.2 Bill of materials (BOM)

The BOM was created to keep all component data together. Sourcing information for components exported from Proteus was collated, which allows the total cost of components to be calculated. Although this is less important during prototyping, it becomes vital leading up to production. Multiple sources and minimum order quantities allows production quantities to be efficiently calculated.

TABLE 3 - BILL OF MATERIALS (BOM)

Quantity	Description	Value	Tolerance	Package	Manuf.	Manuf. Part No	Supplier 1 Order Code (RS)	Supplier 2 Order Code (Farnell)	Supplier 3 Order Code (Mouser)	Cost per unit			MOQ		
1	Arduino Uno R3	n/a	n/a	n/a	Microchip	PIC18F45KK22-I/P	703-7740	2E+06	C18F45K	2.24	£2.09	£2.210	1	1	1
3	LM5101 Half bridge Mosfet Driver	n/a	n/a	Soic-8	Texas Instruments	LM5101AMX/NOPB	551-2638	3E+06	5101AMX	£2.56	2.44	£2.040	1	1	1
6	100v/30Amp Power Mosfet	100V	n/a	3pin through hole	Infineon	IRF540NPBF	688-6872	9E+06	-IRF540I	£1.200	0.854	£1.280	2 (come in 2 pack for 2.40)	1	1
1	3.3V Voltage Regulator	3.3v	n/a	3pin through hole	Texas Instruments	LM1117T-3.3/NOPB	535-9256	3E+06	1117T-3.3	£1.19	£1.210	1.23	5	1	1
1	5v Voltage Regulator	5v	n/a	3pin through hole	Texas Instruments	7805									
1	12v voltage regulator	12v	n/a	3pin through hole	Texas Instruments	LM1085IT-12/NOPB	533-9381	3E+06	1085IT-3	£2.010	£1.890	£1.520	1	1	1
1	Op Amp	1.8v to 5v5	n/a	SC70-5	STMicro	TSZ121ICT	etail/STMicroelectronics/TSZ121ICT?qs=1grwF8CI%252BwkIOkWsAgL7SQ%3D								
1	512kb memory	512	n/a	Soic-8	Texas Instruments	24lc512									
3	Capacitor	0.33u	10%	Through hole	AVX	SR305C334KARTR	699-5165	1E+06	5C334KA	£0.51	£1.24	£0.76	5	1	1
12	Capacitor	0.1u	10%	Though hole	Kemet	C320C104K5R5TA	538-1310	1E+06	C320C104	£0.200	£0.490	£0.170	5	1	1
6	Capacitor	10u	10%	Though hole	Vishay	MAL203038109E3	108-5462	222-030-3	SEPC27	£0.450	£1.060	£0.550	5	1	1
3	Capacitor	1uF	10%	Though hole	Vishay	MAL203038108E3	08-5434	1E+06	2030381	£0.618	£0.951	£0.900	5	1	1
3	Capacitor	470u	10%	Though hole											
2	Capacitor	100p	10%	Though hole	Kemet	C320C104K5R5TA	538-1310	1E+06	C320C104	£0.200	£0.490	£0.170	5	1	1
1	Capacitor	5600p	10%	Though hole											
7	Resistor	1k	±1%	Though hole	Vishay	CCF551K00FKE36	336-1788	2E+06	F551K00	£0.013	£0.136	0.143	5000	1	1
6	Resistor	5.1	±1%	Though hole	TE Connectivity	LR1F5K1	148-253	2E+06	9-LR1F5	£0.026	£0.039	£0.087	10	10	1
1	Resistor	10k	±1%	Though hole	Vishay	MBA02040C1002F	66-7659	3E+06	063JD10	£0.037	£0.118	£0.087	1000	1	1
2	Resistor	15	±1%	Though hole	Vishay	PR01000101509JA	169-5422	2E+06	01000101	£0.013	£0.147	£0.111	5000	1	1
3	Resistor	100	±1%	Though hole	Vishay	MRS25000C1008F	169-5135	9E+06	25000C	£0.019	£0.086	£0.220	5000	10	1
4	Resistor	4k7	±1%	Though hole	Vishay	MRS25000C4701F	169-5287	9E+06	25000C	£0.018	£0.086	£0.220	5000	10	1
4	Sil socket - double	2	n/a	Though hole	Preci-DIP	801-87-002-10-001	702-2741	3E+06	18700210	£0.16	£0.14	£0.17	5	100	1
3	Sil socket - 4input	4	n/a	Though hole	Preci-DIP	831-87-004-10-001	102-0604	3E+06	18700410	£0.35	£0.29	£0.37	10	5	1
1	Sil socket - 6pin	6	n/a	Though hole	Preci-DIP	831-87-006-10-001	702-0626	622806-	18700610	£0.40	£0.15	£0.48	5	6000	1
3	LED	n/a	n/a	Though hole	Cree	C503D-WAN-CCbE	ot in stoc	3E+06	DWANC	n/a	0.22	0.19	n/a	5	1
3	LED	n/a	n/a	Though hole	Cree	C503D-WAN-CCbE	ot in stoc	3E+06	DWANC	n/a	0.22	0.19	n/a	5	1

9.3 Software description and setup

This section includes a general overview of the software flow, as well as an in-depth description of the solutions and setup. Arduino IDE was used to write and upload code to the Arduino Uno.

9.3.1 Software Flow Diagram

The three Arduino Uno timers were utilised in this project. Timers 1 and 2 set the PWM, 32kHz, for the output pins that go to motor drivers. Timer 0 sets the interval time to 1mS for the interrupt. This interrupt uses counters to trigger further functions at 100mS and 1S, Figure 43.

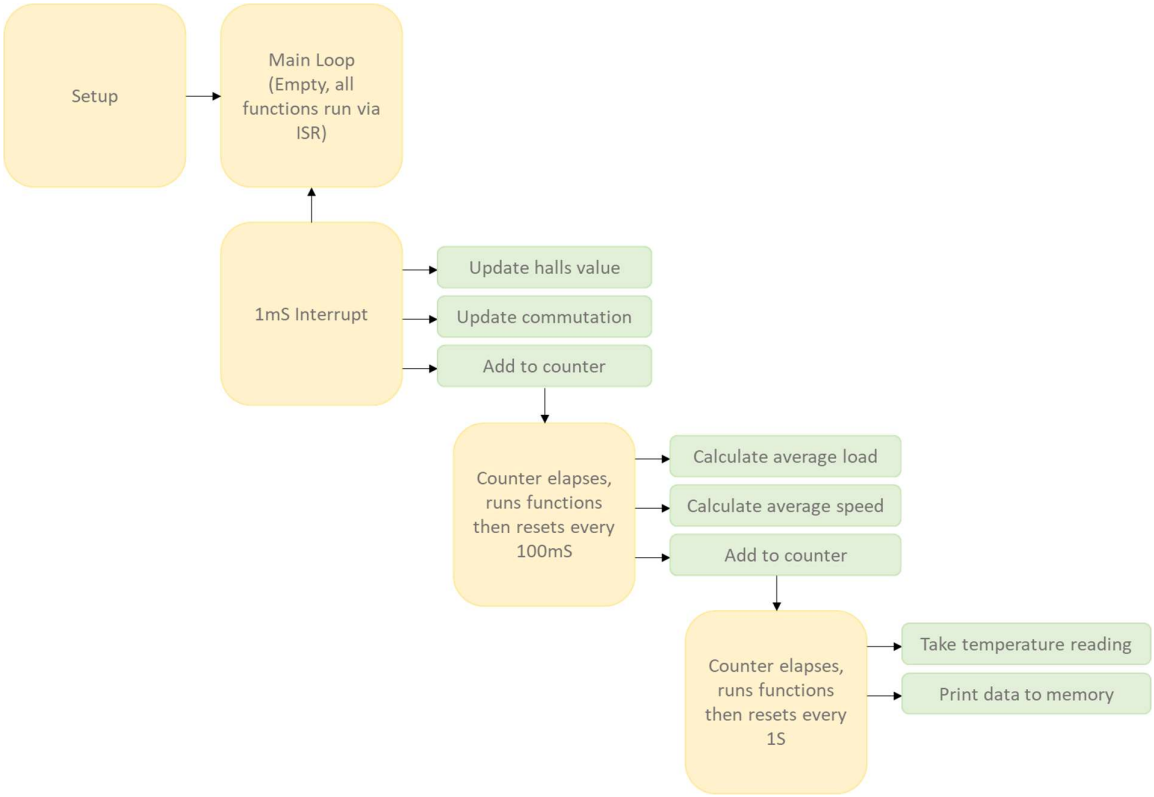


FIGURE 43 - SOFTWARE FLOWCHART

9.3.2 Interrupt

The following code, Figure 44, was used to set up the interrupt in the setup. The timer compare enable TIMSK0 needs to be set last, just before all interrupts are enabled, as when enabled it starts counting immediately.

```
cli();//stop interrupts
  //set timer0 interrupt at 1kHz
  TCCR0A = 0;// set entire TCCR0A register to 0
  TCCR0B = 0;// same for TCCR0B
  TCNT0  = 0;//initialize counter value to 0
  // set compare match register for 1khz increments
  OCR0A = 249;// = (16*10^6) / (1000*64) - 1 (must be <256)
  // turn on CTC mode
  TCCR0A |= (1 << WGM01);
  // Set CS01 and CS00 bits for 64 prescaler
  TCCR0B |= (1 << CS01) | (1 << CS00);
  // enable timer compare interrupt
  TIMSK0 |= (1 << OCIE0A);
sei();//enable interrupts
```

FIGURE 44 - INTERRUPT SETUP

To set the Timer0 interrupt to 1mS, instead of counting to the maximum count of 256, before overflowing the OCR0A is set to 249, giving the desired period as seen in Equation 3. As this value is less than 256, the current prescaler value of 64 works. The minus one must be used as the compare match register is zero indexed.

$$OCR0A = \frac{Clock\ Frequency}{Desired\ Frequency\ x\ Prescaler} - 1$$

EQUATION 3 - OCR0A

When the timer overflows at 249 the interrupt service routine is called, and the count begins again.

The Atmega328P datasheet [65] shows, Figure 45, that WGM01 must be selected to set up the interrupt to CTC mode, CS01 and CS00 are set to the correct prescaler, Figure 46 .

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, phase correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, phase correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

FIGURE 45 - CTC MODE [65]

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(no prescaling)
0	1	0	$clk_{I/O}$/8 (from prescaler)
0	1	1	$clk_{I/O}$/64 (from prescaler)
1	0	0	$clk_{I/O}$/256 (from prescaler)
1	0	1	$clk_{I/O}$/1024 (from prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

FIGURE 46 - PRESCALE SETUP [65]

The interrupt service routine contains the full program, the main loop is empty. Every time the interrupt overflows the 'halls' and 'commutation' functions are called, Figure 47. These functions calculate the rotor position and which commutation step that is needed. It is important that these are updated are a high frequency as the motor might stutter or vibrate if the commutation step doesn't match with the actual rotor position.

```
ISR(TIMER0_COMPA_vect){//timer0 interrupt 1kHz
  //PORTB ^= B00100000;// toggles bit which affects pin13     //debug
halls();
commutation();
  i++;
  if(i==100){  //100ms
  //digitalWrite(13, HIGH);  //debug
  digitalWrite(LED_BUILTIN, HIGH);  //debug
    checkspeedload();   //check speed and load
    i=0;        //reset counter
  m++;
  if(m==10){ //1000ms
  //digitalWrite(13, LOW); //debug
  digitalWrite(LED_BUILTIN, LOW);  //debug
    temp();   //check temp
  //  memory(); //print to memory
    m=0;      //reset counter
    } //close1000ms
  }//close100ms
}//close isr
```

FIGURE 47 - INTERRUPT CODE

Every time the interrupt overflows a counter 'i' is updated. When this reaches one hundred, which is 100mS, it overflows, and the speed and load values are updated. Another counter 'm' is updated when 'I' overflows and the counter 'i' is reset to zero.

The new counter 'm' counts to ten, which takes one second, 10 times 100mS. When this counter overflows, a temperature reading is taken using the 'temp' function. The 'memory' function is then called, which saves useful data to an EEPROM integrated chip before the counter is reset.

9.3.3 Pulse width modulation setup

Although there is not a formula for determining the frequency of PWM, an estimation can be made depending on the frequency of electrical rotations and the resistance and inductance of coils in the motor [66]. The L/R time constant needs to be much longer than the PWM frequency. If the frequency is too low, it may cause voltage ripple. A frequency of ~32kHz has been selected, and if voltage ripple is present during testing, a higher frequency can be selected.

All lowside output pins need to be capable of PWM. In the Arduino Uno this means that the remaining two timers need to be utilised. Timer 1 gives a PWM output at pins 9 and 10. Timer 2 gives a PWM output at pins 3 and 11.

Using the Arduino function "analogueWrite()" the duty cycle can be set between 0 and 255, where 0 is 0% duty and 255 is 100% duty. Setting the analogueWrite value is completed in the 'commutation' function, which is detailed later. The period (T) of the PWM signal is set by calculating the frequency using the TCCRxB register, Figure 48 - TCCR1B register. The clock select bits (0 to 2) divisor is set to '001', Figure 49 - CS bits table.

Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

FIGURE 48 - TCCR1B REGISTER [65]

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/1 (no prescaling)
0	1	0	$clk_{I/O}$/8 (from prescaler)
0	1	1	$clk_{I/O}$/64 (from prescaler)
1	0	0	$clk_{I/O}$/256 (from prescaler)
1	0	1	$clk_{I/O}$/1024 (from prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

FIGURE 49 - CS BITS TABLE [65]

```
// This sets PWM for pins 9 and 10 to 31372.55
Hz //CS divisor =1
TCCR1B = TCCR1B & B11111000 | B00000001;

// This sets PWM for pins 3 and 11 to 31372.55
Hz //CS divisor =1
TCCR2B = TCCR2B & B11111000 | B00000001;
```

FIGURE 50 - PWM FREQUENCY

The TCCRxB register can be set using the bitmath in Figure 50.

9.4 Software functions

The code can quickly become difficult to read if all the functionality is in the ISR. Functions are instead called by the ISR when they are to be run. Six functions are called from the ISR:

- commutation
- halls
- checkspeedload
- loadcalc
- temp
- memory

9.4.1 'commutation'

The 'commutation' function uses the rules in chapter 8.1 to switch between the commutation steps based on the hall sensor signal inputs. Before it can do this, the function first checks the speed; if it is above 25kph the regenerative braking is applied. The braking is applied for one millisecond before the speed is checked again, if the speed continues to be above 25kph then consistent braking is applied. During consistent braking downhill there is an added benefit that the force of the bike and rider trying to accelerate away from the braking eCargo Trailer is now applied through the arm and added to the weight to the eCargo Trailer and regeneration potential is at its maximum.

When travelling below 25kph, the function checks the value of the throttle level. The throttle level tells us two things: if the eCargo Trailer should be braking or driving, and how hard the brake or drive should be applied.

The switch case checks the 'HallVal' and sends a signal to the corresponding motor drivers, a PWM signal to a low side driver and an 'ON' signal to a high side driver in drive mode or a PWM signal to multiple low side drivers in braking mode.

```
if (throttle > 127){
     switch (HallVal)
      {
       case 3:
         //Highside
         PORTD  &= B10010111;
         PORTD  |= B01000000;
         //Lowside
         analogWrite(9,MotorSpeedLevel);
         analogWrite(10,0);
         analogWrite(11,0);
         break;
```

FIGURE 51 - COMMUTATION SWITCH CASE

The highside driver is set to on using bitmath. It is especially important not to edit the whole register as the lowest 3 bits of portD contain the halls input data. The 'MotorSpeedLevel' variable sets the PWM duty cycle, where zero is 0% and 255 is 100% duty. The full function can be viewed in the appendices.

9.4.2 'halls'

The main task for the 'halls' function is to read the location of the rotor via the halls sensors, Figure 52. These are added together, with a multiplier, to gain a value that is used to change the commutation step. The output of the NewHallVal is between 1 and 6. This value is not in the correct order of commutation, and instead moves from 3,1,5,4,6,2 before repeating. When reversing the values are reversed and go in the opposite direction.

```
• Hall1 = digitalRead(0);  // read input value from Hall 1
•   Hall2  = digitalRead(1);  // read input value from Hall 2
•   Hall3  = digitalRead(2);  // read input value from Hall 3
•   NewHallVal = (Hall1) + (2*Hall2) + (4*Hall3); //Computes the binary value of t
  he 3 Hall sensors
```

FIGURE 52 - HALLS VALUE

The secondary purpose of halls function is to calculate the speed, which equals distance divided by time. As the motor is fixed gear, every electrical commutation step directly affects the rotation of the mechanical wheel and has a known ratio once the number of pole pairs on

the rotor is calculated. The distance the wheel travels per rotation and the circumference of the wheel are known, so is the time taken between readings, every 1 mS.

A pointer is used to point at the current commutation step and to calculate the number of steps made since the last reading, Figure 53.

```
switch (NewHallVal){
      case 3:
        neworderpointer = 1;
      break;
      case 1:
        neworderpointer = 2;
      break;
      case 5:
        neworderpointer = 3;
      break;
      case 4:
        neworderpointer = 4;
      break;
      case 6:
        neworderpointer = 5;
      break;
      case 2:
        neworderpointer = 6;
      break;
    }//switch
```

FIGURE 53 - NEWORDERPOINTER

The steps can now be calculated by taking away the old pointer value from the new pointer value, Figure 54. The code needs also counts when the pointer rolls over from 6 to 1. This is done by checking the values against each other to decide on which one is higher. The eCargo Trailer is designed to have a max speed of 25 kph, which means it is unable to move round

one or more electrical cycles every 1 mS at this speed. It is also important to note that the POC only has a powered forward mode but could get pushed back manually.

```
if(orderpointer <= neworderpointer){
    steps = neworderpointer - orderpointer;
  }
  else{
    steps = (6-orderpointer) + neworderpointer;
```

FIGURE 54 - STEPS CALCULATION

The speed can then be calculated, Equation 4, with the resultant value being placed in an array. The array is used for averaging later. The speed calculation is below; however, it does depend on the number of pole pairs and coils in the motor. This data is not available on the datasheet for the motor and needs to be calculated either by opening the motor and counting, or by rotating the wheel back a full revolution while counting the pulses read by the halls sensor.

$$Speed\ (kph) = \frac{\frac{Tyre\ circumference}{6\ x\ Number\ of\ pole\ pairs}\ x\ number\ of\ steps}{1mS\ in\ hours}$$

EQUATION 4 - SPEED

Number of pole pairs	Speed Calculation
2	Speed (kph) = 480 x Number of steps
3	Speed (kph) = 320 x Number of steps
4	Speed (kph) = 240 x Number of steps
5	Speed (kph) = 192 x Number of steps

TABLE 4 – SPEED CALCULATION

This is a problem when calculating speed, as one step per millisecond on a 5-pole pair rotor is 192 kph, which necessitates the speed values to be entered into an array.

9.4.3 'checkspeedload'

Every hundred milliseconds, the 'checkspeedload' function continues the speed calculation. It sums the speed array and divides the sum by 100 to give an average of the speed. This means the minimum speed detectable is 1.92kph, which can be further reduced by pole pairs.

```
void checkspeedload(){    //average speed/load for check EVERY 100ms
loadcalc();
  for(n = 0; n < 100; n++){
  ecargospeed = 0;
  ecargospeed += speed[n];
    }
  n=0;
  ecargospeed = ecargospeed/100;
}//checkspeedload
```

FIGURE 55 - CHECKSPEEDLOAD

Even with a 2-pole pair, the maximum allowed speed for the eCargo Trailer speed of 25 kph can be detected.

9.4.4 'loadcalc'

'checkspeedload' calls the 'loadcalc' function to calculate the load through the connecting arm, Figure 56. This uses a similar averaging filter, but this time averages ten readings over one second.

```
void loadcalc(){                  //include array for averaging
//Calculate load
  load[p] = analogRead(A0) - loadinit;
  p++;
  if (p > 9){
    p=0;
  }
  for(q = 0; q < 10; q++){
  aveload = 0;
  aveload += load[q];
  aveload = aveload/10;
  }
```

FIGURE 56 – LOADCALC PART1

To get a load value, the load signal, which has already passed through the motor circuitry explained earlier, is taken at an analogue pin of the Arduino Uno which then goes through an analogue to digital converter (ADC) to give a value between 0 and 1023. This value is compared to the initial load reading when the eCargo Trailer is not connected to a bike, to give a zero

calibration. Now negative load numbers now mean the connecting bar is in compression and positive values mean the bar is getting stretched.

The 'aveload' is used to calculate whether the eCargo Trailer is driving or braking by creating a 'throttle' value using the map function. The bottom half of the 'throttle', 0 to 127, is used for braking and the top half, 128 to 255, is used for driving. The map function is used again on both halves of the 'throttle' value to gain a 'MotorSpeedLevel' and a 'BrakeSpeedLevel' which are values between 0 and 255 and are used to set the duty cycle in the 'commutation' function.

9.4.5 'temp'

The 'temp' function gets called every second and reads a temperature off the ADC to a value between 0 and 1023. The 'map' function on the Arduino allows an integer conversion, in this case we can put the limits of the temperature probe, Figure 57,

```
void temp(){
//Caculate winding temperature
  windingtemp = analogRead(A1);
  windingtemp = map(windingtemp, 0, 1023, -
40, 180);
}//temp
```

FIGURE 57 – TEMP

9.4.6 'memory'

The 'memory' function saves data to EEPROM every second through the I²C protocol. A library, wire.h, has been used for this. Firstly, the data is arranged to reduce time taken for a data transfer, Figure 58. The data gets transferred in bytes.

The speed cannot go above 25kph, so uses at maximum the first 5 bits. The load is edited into eight load levels using the map function and is shifted to bits 6, 7 and 8 and merged with the speed value.

```
void memory(){        //print every second
//Loadlevel
loadlevel = map(throttle, 0, 255, 0, 7);
loadlevel = loadlevel << 5;
//Create loadlevelspeed byte hi largest bits are load level, 5 lowest are speed
loadlevelspeed = 0x0000 || ecargospeed;
loadlevelspeed = loadlevelspeed || loadlevel;

//print speed/temp/load
Wire.beginTransmission(EEPROM_I2C_ADDRESS); // transmit to device
  Wire.write(windingtemp);   // sends 1 byte
  Wire.write(loadlevelspeed);   // sends 1 byte containing load and speed
  Wire.endTransmission();    // stop transmitting
}//memory
```

FIGURE 58 - MEMORY

The transmission is opened to the defined address of the memory and the windingtemp byte and the combined speed and load byte are sent before ending the transmission. This data is not used again by the eCargo Trailer and can be requested later using another program.

9.5 Implementation Conclusion

Various steps of the design have been looked at in this report. Although not all the stages were required in aims and objectives, they influenced the design decisions that have been made. They give a better overview for readers of this report and are important for funding applications and deciding what the next steps are in the project.

The stages of the design that are marked out in the aims and objectives move on to the testing stage.

10. Testing

This section details the test protocols for both hardware and software.

Equipment for software tests:

- Laptop
- Arduino Uno
- USB Power cable
- Oscilloscope (Teledyne T3DSO1104)
- Probes
- Connecting cables
- Updated code (LEDs and output pins setup can be seen in the appendices. Code for each test is included below.)

The test equipment was set up as in Figure 59 and Figure 65

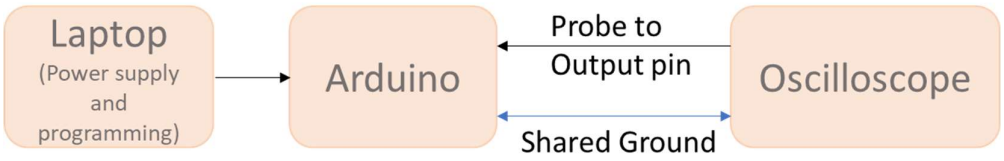


FIGURE 59 - TEST SETUP

10.1 Testing timings

There are several different timings that need tested to allow the full operation of the software:

1. Interrupt triggering at a frequency of 1kHz
2. Counter elapsing at 100mS
3. Counter elapsing at 1S

Pin13 is used to test the Arduino output of the 1kHz timings. The built in LED and pin13 is used to test the output of the 100mS and 1S timings together.

10.1.1 1kHz Test Protocol – Test 1

- Upload the code to the Arduino Uno This toggles pin13 output at 1kHz, Figure 60.

```
• ISR(TIMER0_COMPA_vect){//timer0 interrupt 1kHz
•   PORTB ^= B00100000;// toggles bit which affects
  pin13//debug
```

FIGURE 60 - 1KHZ TEST CODE

- Check the LED status to confirm upload
- Place probe on pin 11 (Use shared ground)
- Find signal on oscilloscope, adjust scale, and take measurement

10.1.2 1S and 100mS Test Protocol – Test 2

- Upload the code to the Arduino Uno. This blinks the LED and pin13 by setting it HIGH at each 100mS count overflow and setting it LOW at the one second count overflow, Figure 61.

```
ISR(TIMER0_COMPA_vect){//timer0 interrupt 1kHz
  //PORTB ^= B00100000;// toggles bit which affects pin13     //debug
halls();
commutation();
  i++;
  if(i==100){  //100ms
  //digitalWrite(13, HIGH);  //debug
  digitalWrite(LED_BUILTIN, HIGH);  //debug
    checkspeedload();   //check speed and load
    i=0;        //reset counter
  m++;
  if(m==10){ //1000ms
  //digitalWrite(13, LOW); //debug
  digitalWrite(LED_BUILTIN, LOW);  //debug
    temp();   //check temp
  //  memory(); //print to memory
    m=0;      //reset counter
    } //close1000ms
  }//close100ms
}//close isr
```

FIGURE 61 - 1MS AND 1S TEST CODE

- Check the LED status to confirm upload
- Place probe on pin 13 (Use shared ground)
- Find signal on oscilloscope, adjust scale, and take measurement

10.2 Pulse width modulation – Test 3

PWM to the low side drivers can be changed by setting either the 'MotorSpeedLevel" or "BrakeSpeedLevel" depending whether the system is on brake or drive mode which is set by the throttle variable. The motor also needs to be travelling below 25kph. The variables are set in the commutation function Figure 62, so that only the MotorSpeedLevel needs to be adjusted to gain different duty cycles.

- Upload the code to the Arduino Uno, Figure 62.

```
//Inside commutation function
  ecargospeed = 1;
  throttle = 200;
  BrakeSpeedLevel = 200;
  MotorSpeedLevel = 127;

//Inside halls function
HallVal = 6; //Debug and for testing
```

FIGURE 62 - SET VARIABLES

- Check the LED status to confirm upload
- Place probe on pin 11 (Use shared ground)
- Find signal on oscilloscope, adjust scale, and take measurements at 0%. 25%, 50%. 75% and 100% duty cycle.

10.3 Commutation steps – Test 4

To check that the PWM signal is being output at the correct pins the PWM duty is held at 50% and each commutation step is tested by changing code. Again, the oscilloscope is used with probes on each of the pins that go to the low side motor drivers.

- Upload the code to the Arduino Uno, 'HallVal' is changed from 1 to 6.
- Check the LED status to confirm upload
- Place probe on pin 11 (Use shared ground)
- Find signal on oscilloscope, adjust scale, and take measurements at each 'HallVal'.

10.4 Hardware testing

Limited hardware testing has taken place. The following testing protocols and ideas have been written and implemented.

10.4.1 Strain gauge test - Test 5

Load testing of the strain gauge can be tested using the following test rig, Figure 63.

The strain gauge is applied to the steel test tube and the connecters are attached to the PCB

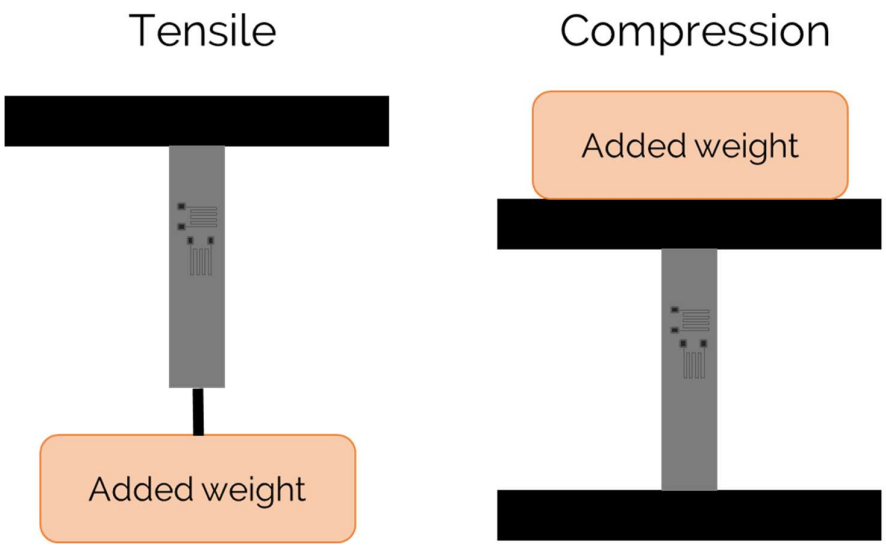


FIGURE 63 - LOAD TEST RIG

The initial resistance needs to be first tested without added weight to zero offset the sensor. This is done in software during the setup process. Weight is then added to increase the tensile

force and is then repeated in the compressive direction. The results can be taken by monitoring the 'load' value in debug mode on the Arduino IDE.

10.4.2 Initial board checks, grounds/power/data

Power and signal connections on the PCB need to be tested before the board is powered up. Damage can be caused the components not designed to withstand high voltage levels. Each pad needs tested to see that it has the correct connectivity. This is a basic test that involves a visual inspection and a continuity test completed with a multimeter. The board dimensions should be checked including measurement of the traces to make sure they are sized correctly, particularly for the high current carrying rails.

10.4.3 LEDs

The board can now be powered up after the initial checks. LEDs have been placed on the PCB on each of the power rails, this gives an easy and quick indication to see that everything is powered up correctly. The onboard LED on the Arduino retains the code to go HIGH for 900mS and LOW for 100mS, this shows that the code has been correctly moved across from laptop to the Arduino Uno.

10.4.4 Next stages of testing – Motor test rig

The motor test rig is used to test the efficiency of the motor.

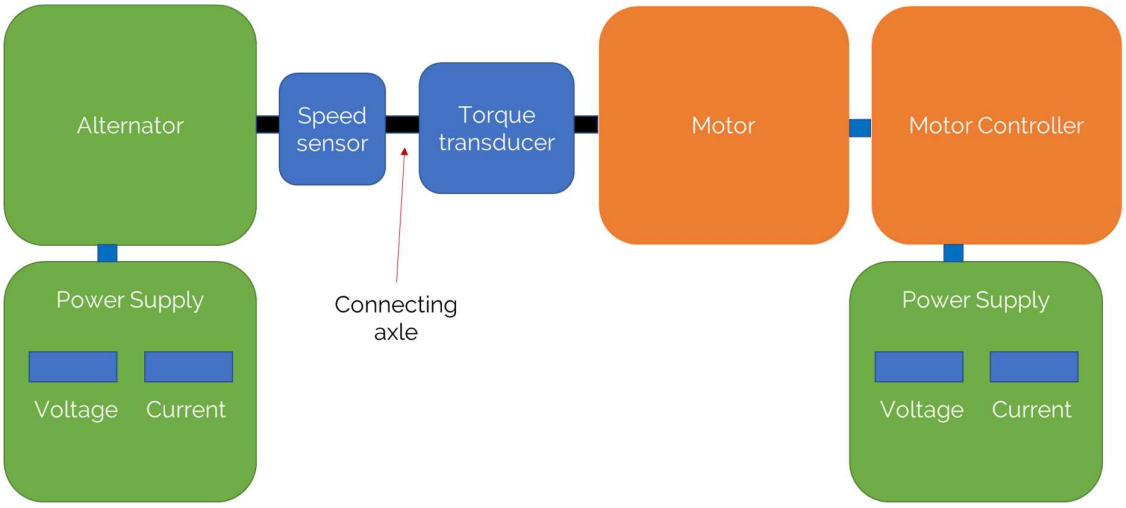


FIGURE 64 - MOTOR TEST RIG

Using the calculation discussed in 8.7 - Recorded data and maintenance management chapter, an efficiency profile of the motor can be assessed.

When the coils of the alternator are energised this adds load to the motor. If the voltage at the alternator coils is increased the motor must work harder to overcome the magnetic force. The alternator has its own inefficiencies, so the output power is calculated by taking the torque and rotational velocity of the axle between the motor and the alternator.

The input power can be calculated by setting the voltage on the motor's power supply to 36V or 43.2V and recording the current draw from the system. An edited version of the software would need uploaded to Arduino to set the motor to fully on, 'MotorSpeedLevel = 255;'. Temperature profiles and noise levels of the motor could also be recorded.

The voltage of the alternator should be increased until the maximum current is reached. The motor speed decreases as more power is required. This gives a full profile of the motor's efficiency through the normal running speeds of the motor.

11.Results

The test equipment was set up as prescribed, Figure 65, for Test 1, Test 2 and Test 3.

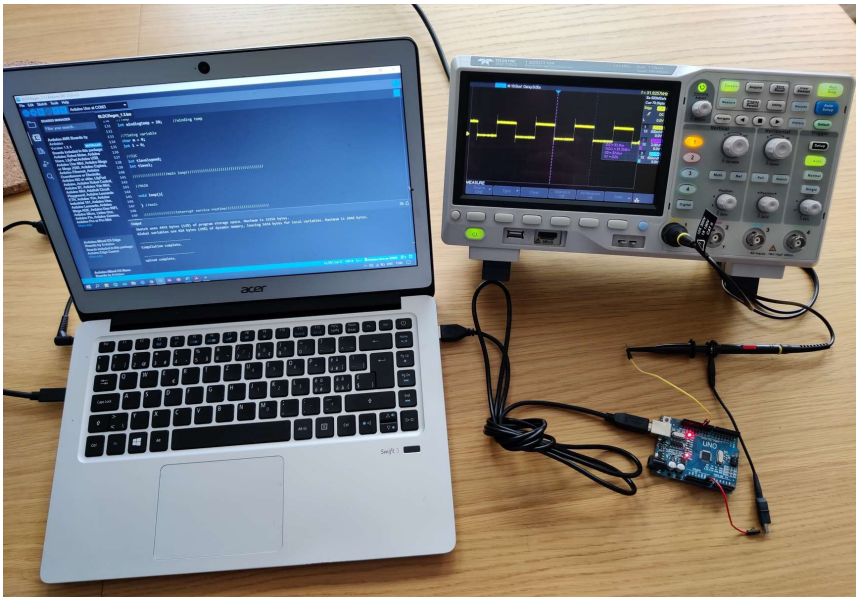


FIGURE 65 - SOFTWARE TEST SETUP

11.1 Test 1 results - 1 kHz interrupt

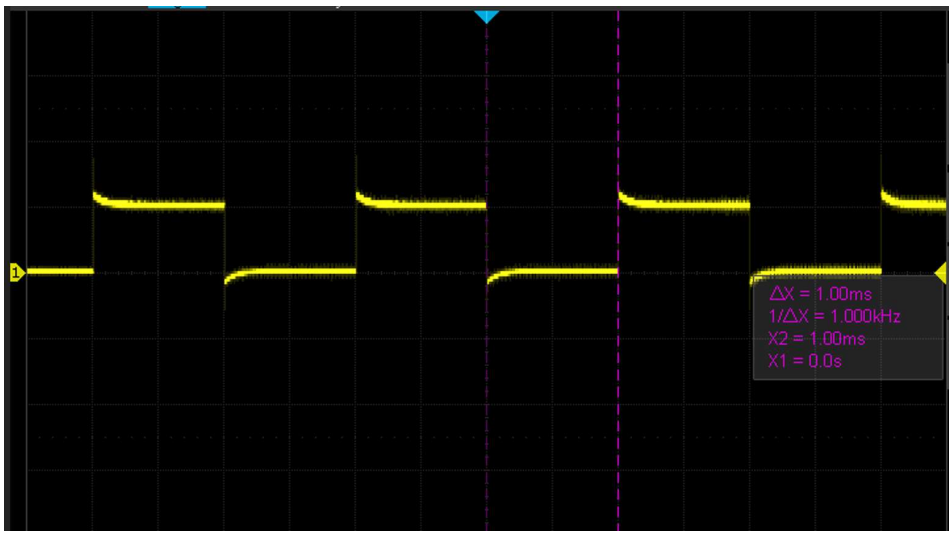


FIGURE 66 - TEST 1 RESULTS

Test 1 protocol was followed and as can be seen in Figure 66 the results are as expected. Pin 13 output toggles between high and low every 1mS. This allows testing to continue.

11.2 Test 2 results – Counter timings

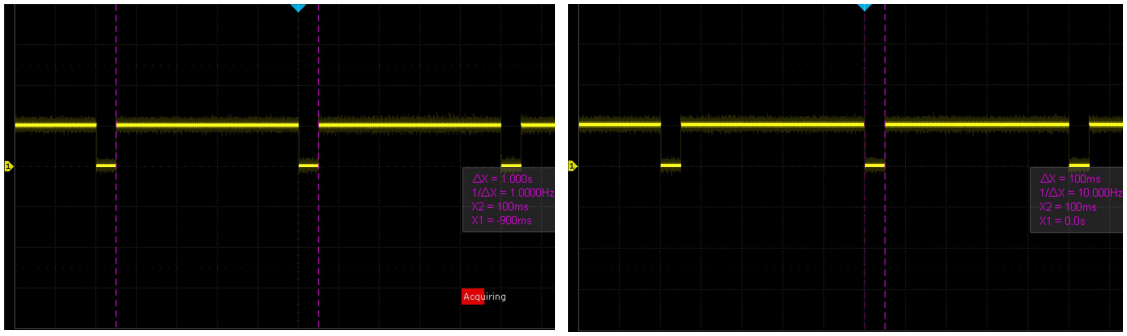


FIGURE 67 - TESTING COUNTERS

Test 2 protocol was followed and as can be seen in Figure 67, the results are as expected. The LED started flashing and the Pin 13 output matches this output. The timings can be seen to be one second and 100 milliseconds.

11.3 Test 3 results – Pulse width modulation

The 'MotorSpeedLevel' value was changed in the code between 0 to 255 to change the duty cycle. Results for timings for both the period and duty cycle were taken.

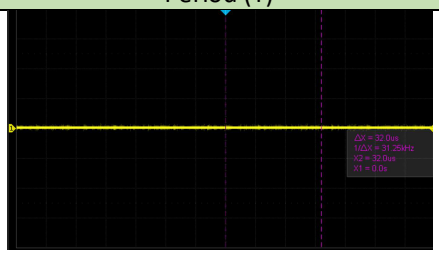
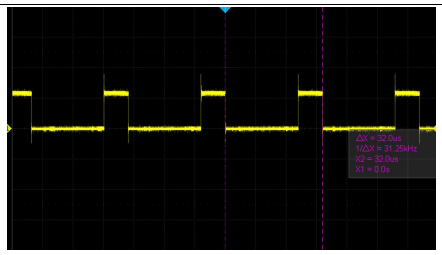
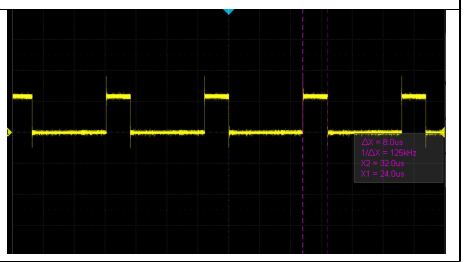
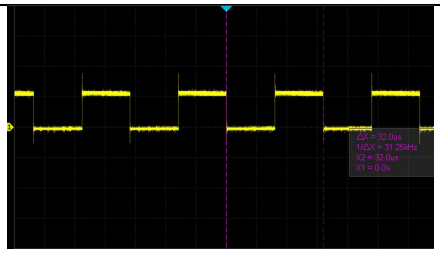
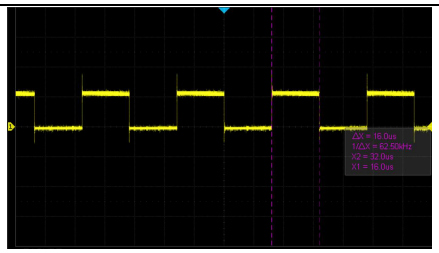
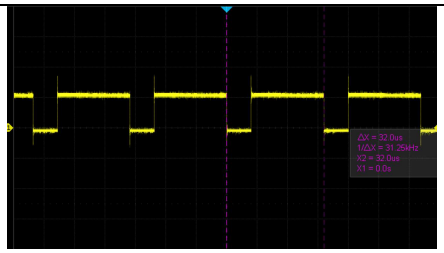
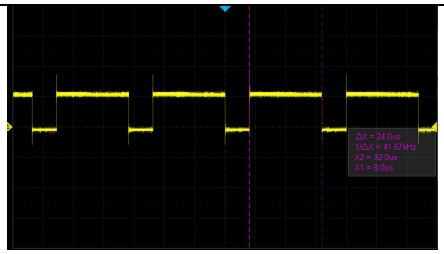
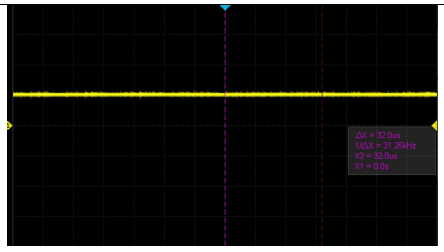
Duty cycle	Period (T)	Duty (T_{on})
0%		
25%		
50%		
75%		
100%		

TABLE 5 - DUTY CYCLE

The period of the duty cycle is 32uS which means the frequency is 31.25kHz. This differs slightly from the defined frequency of 31372.55Hz. However, this frequency is not getting used for timings or calculations, only the PWM signal. This is close enough to the prescribed frequency to complete its task.

What is more important is the duty cycle. The percentage of time the signal is HIGH in the period. As can be seen from Table 5 - Duty cycle, the setup described in 9.3.3 - Pulse width modulation setup is working well.

11.4 Test 4 results – Commutation

The test equipment was set up with 3 probes as seen in Figure 68.

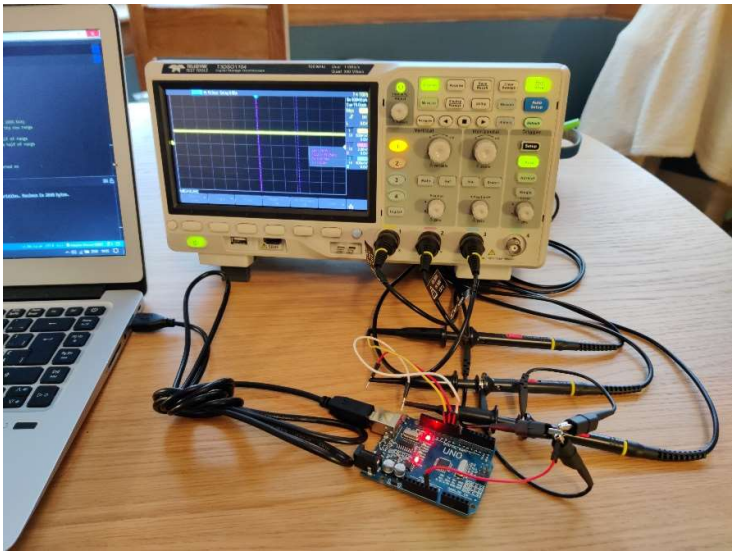


FIGURE 68 - TEST SETUP

First the probes were connected to the low side pin outputs The results can be seen in Table 6 - Lowside steps.

Pin 9 = Blue trace
Pin 10 = Pink trace
Pin 11 = Yellow trace

The high side pins were then tested. The results can be seen in Table 7 - Highside StepsTable 6 - Lowside steps

Pin 3 = Blue trace
Pin 5 = Pink trace
Pin 6 = Yellow trace

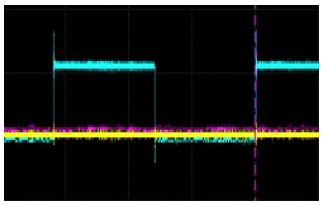
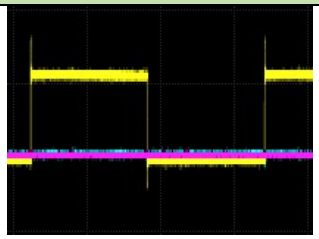
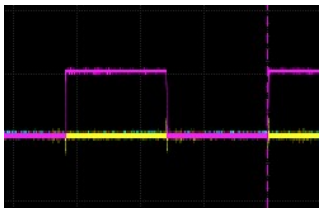
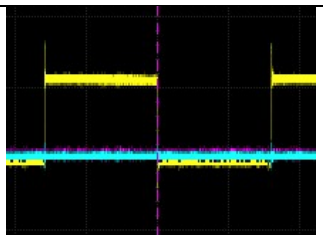
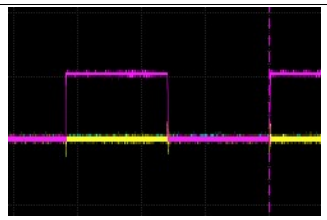
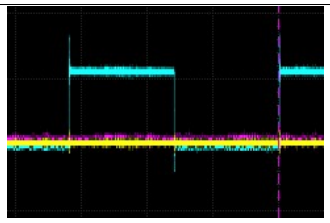
Step	Result	Step	Result
3		4	
1		6	
5		2	

TABLE 6 - LOWSIDE STEPS

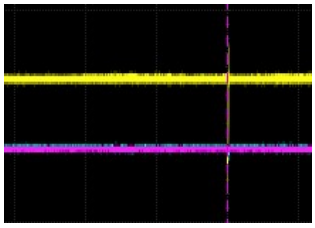
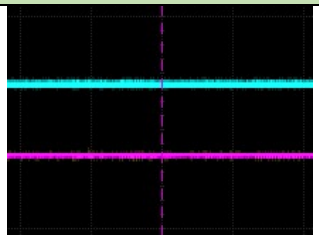
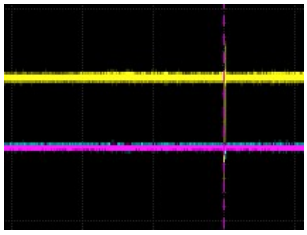
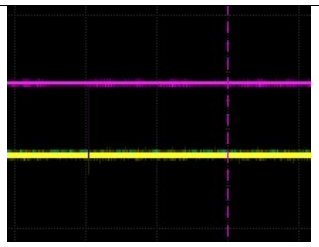
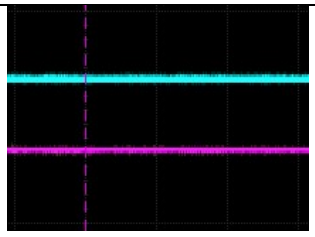
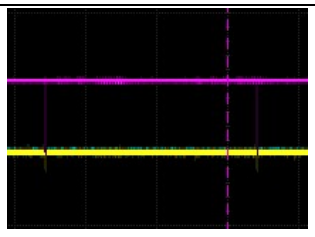
Step	Result	Step	Result
3		4	
1		6	
5		2	

TABLE 7 - HIGHSIDE STEPS

These results match up with the steps on the software and from the theory and concludes the software testing until more components arrive.

11.5 Test 5 results – Strain gauge

The strain gauge tests were not completed during this project as the PCB was not received in time. The attachment of the strain gauge to the steel tube was still tested, Figure 69.

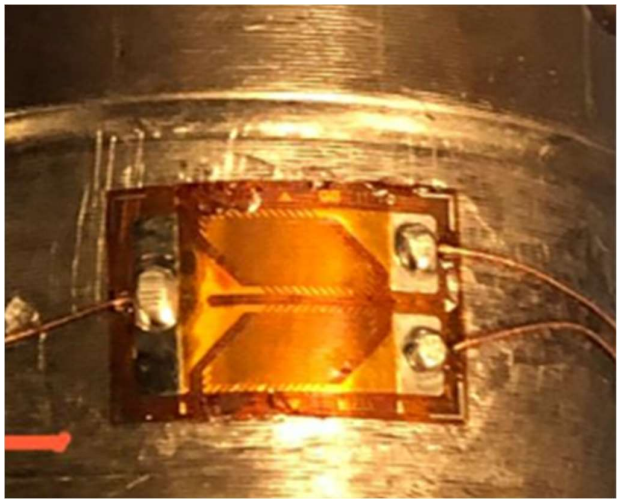


FIGURE 69 - GLUED STRAIN GAUGE

Problems from the gauge application process arose during this test.

The strain gauge that arrived had each grid at 45 degrees to the film. This made it difficult to align the gauge axially along the tube. These types of gauges are commonly used to gain torque values along the axle. They are designed this way as it is easier to glue the film around the axle, rather than across. With this diameter of steel tubing, it was impossible to glue the film at the correct angle without having air gaps between the film and the steel surface. The air gaps would cause inconsistencies in the wires stretching under load, given inconsistent load values. A new half bridge strain gauge will be used with film that has gauges running at zero and 90 degrees.

12. Conclusions and Further Work

This work presented the first iteration of the POC for an electric cargo trailer called of the eCargo Trailer. The hardware design was thoroughly discussed and software tests on the major aspects of the design were demonstrated. Although delays in receiving PCB components have hindered the progress of the design stage, the work presented here provides the blueprint for the design of this product.

The process of designing an electronic module for a motorised trailer with regenerative braking has been documented and testing has proven that a strain gauge can be used successfully to control forward motion and braking.

This POC could be advanced further with the aim of creating a working prototype of the eCargo Trailer. This would involve a more costly investment into components, such as batteries, a motor, and a chassis, along with testing equipment. For the working prototype other sensors, such as an accelerometer and gyro meter should be added to the design. The software would require testing on operational hardware, by setting up the motor test rig and following the protocol in 10.4.4. Optimisation of all the structural components of the trailer using FEA is warranted for the chassis design, so that a full-size working prototype can be realised.

13. References

[1] Department for Transport, "Position statement on last mile logistics," 2020, Accessed: Nov. 28, 2021. [Online]. Available: https://www.gov.uk/government/publications/review-of-last-mile-logistics-2019/position-statement-on-last-mile-logistics.

[2] "Cargo space: The overview – Cargobike Magazine." https://cargobikemag.com/cargo-space-the-overview/ (accessed Apr. 23, 2022).

[3] R. Nocerino, A. Colorni, F. Lia, and A. Luè, "E-bikes and E-scooters for Smart Logistics: Environmental and Economic Sustainability in Pro-E-bike Italian Pilots," *Transp. Res. Procedia*, vol. 14, pp. 2362–2371, Jan. 2016, doi: 10.1016/J.TRPRO.2016.05.267.

[4] I. Standard, "Cycles - Electrically power assisted cycles - EPAC Bicycles," *BS EN 151942017*, 2017.

[5] L. Ranieri, S. Digiesi, B. Silvestri, and M. Roccotelli, "A review of last mile logistics innovations in an externalities cost reduction vision," *Sustain.*, vol. 10, no. 3, p. 782, Mar. 2018, doi: 10.3390/su10030782.

[6] "Glasgow's Low Emission Zone (LEZ) - Glasgow City Council." https://www.glasgow.gov.uk/LEZ (accessed Nov. 28, 2021).

[7] J. Allen *et al.*, "Understanding the impact of e-commerce on last-mile light goods vehicle activity in urban areas: The case of London," *Transp. Res. Part D Transp. Environ.*, vol. 61, pp. 325–338, Jun. 2018, doi: 10.1016/j.trd.2017.07.020.

[8] A. Broaddus, M. Browne, and J. Allen, "Sustainable freight impacts of the London congestion charge and low emissions zones," *Transp. Res. Rec.*, vol. 2478, pp. 1–11, Jan. 2015, doi: 10.3141/2478-01.

[9] L. Thoma and J. Gruber, "Drivers and barriers for the adoption of cargo cycles: An exploratory factor analysis," in *Transportation Research Procedia*, Jan. 2020, vol. 46, pp. 197–203, doi: 10.1016/j.trpro.2020.03.181.

[10] R. B. Ellison, S. P. Greaves, and D. A. Hensher, "Five years of London's low emission zone: Effects on vehicle fleet composition and air quality," *Transp. Res. Part D Transp. Environ.*, vol. 23, pp. 25–33, Aug. 2013, doi: 10.1016/J.TRD.2013.03.010.

[11] "COP26 declaration on accelerating the transition to 100% zero emission cars and vans - GOV.UK." https://www.gov.uk/government/publications/cop26-declaration-zero-emission-cars-and-vans/cop26-declaration-on-accelerating-the-transition-to-100-zero-emission-cars-and-vans (accessed Nov. 28, 2021).

[12] S. Shead, "Chip shortage: UK car production falls to lowest level since 1956." https://www.cnbc.com/2021/08/26/uk-car-production-falls-to-lowest-level-since-1956-amid-chip-shortage.html (accessed Nov. 28, 2021).

[13] "Funding boost for green last mile delivery bikes - GOV.UK." https://www.gov.uk/government/news/funding-boost-for-green-last-mile-delivery-bikes (accessed Nov. 28, 2021).

[14] energysavingtrustorguk, "Energy Saving Trust Electrifying last mile deliveries: A guide for businesses A guide for businesses 2 Energy Saving Trust Electrifying last mile deliveries: A guide for businesses," 2020, Accessed: Nov. 28, 2021. [Online]. Available: http://one.cyclelogistics.eu/docs/111/CycleLogistics_.

[15] M. Faccio and M. Gamberi, "New City Logistics Paradigm: From the 'Last Mile' to the 'Last 50 Miles' Sustainable Distribution," *Sustain. 2015, Vol. 7, Pages 14873-14894*, vol. 7, no. 11, pp. 14873–14894, Nov. 2015, doi: 10.3390/SU71114873.

[16] "UPS tests electric cargo bike in Switzerland." https://supplychaindigital.com/supply-chain-2/ups-tests-electric-cargo-bike-switzerland (accessed Nov. 28, 2021).

[17] "Netherlands: New Rules Pending for E-Bikers | Library of Congress." https://www.loc.gov/item/global-legal-monitor/2016-04-12/netherlands-new-rules-pending-for-e-bikers/ (accessed Nov. 28, 2021).

[18] A. B. Jahre *et al.*, "Public employees in South-Western Norway using an e-bike or a regular bike for commuting – A cross-sectional comparison on sociodemographic factors, commuting frequency and commuting distance," *Prev. Med. Reports*, vol. 14, p. 100881, Jun. 2019, doi: 10.1016/J.PMEDR.2019.100881.

[19] T. Bieliński and A. Ważna, "Electric Scooter Sharing and Bike Sharing User Behaviour and Characteristics," *Sustain. 2020, Vol. 12, Page 9640*, vol. 12, no. 22, p. 9640, Nov. 2020, doi: 10.3390/SU12229640.

[20] A. K. Hess and I. Schubert, "Functional perceptions, barriers, and demographics concerning e-cargo bike sharing in Switzerland," *Transp. Res. Part D Transp. Environ.*, vol. 71, pp. 153–

168, Jun. 2019, doi: 10.1016/J.TRD.2018.12.013.

[21] T. Ampe *et al.*, "The impact of a child bike seat and trailer on the objective overtaking behaviour of motorized vehicles passing cyclists," *Transp. Res. Part F Traffic Psychol. Behav.*, vol. 75, pp. 55–65, Nov. 2020, doi: 10.1016/J.TRF.2020.09.014.

[22] "Impacts of last-mile parcel delivery on transport and the environment – Future Towns Innovation Hub." https://futuretowns.soton.ac.uk/2020/06/30/impacts-of-last-mile-parcel-delivery-on-transport-and-the-environment/ (accessed Nov. 28, 2021).

[23] "Ford hybrid vans for smart, sustainable, last mile deliveries | Ford UK." https://www.ford.co.uk/experience-ford/ford-blog/new-take-on-getting-parcels-to-your-door (accessed Nov. 28, 2021).

[24] "The 18 Best Bike Trailers for Bicycle Touring - CyclingAbout." https://www.cyclingabout.com/best-bike-trailers-bicycle-touring/ (accessed Dec. 14, 2021).

[25] "Take on your City with CARLA CARGO!" https://www.carlacargo.de/ (accessed Dec. 14, 2021).

[26] "10 Electric Bike Motorized Trailers for an Extra Boost [VIDEOS] | Electric Bike Report | Electric Bike, Ebikes, Electric Bicycles, E Bike, Reviews." https://electricbikereport.com/electric-cargo-trailers-guide-video/#more-19428 (accessed Dec. 14, 2021).

[27] Encanta, "Overview of the hardware product development stages: POC – EVT – DVT – PVT explained." https://www.encata.net/blog/overview-of-the-hardware-product-development-stages-explained-poc-evt-dvt-pvt (accessed Apr. 26, 2022).

[28] A. Joseph Godfrey and V. Sankaranarayanan, "A new electric braking system with energy regeneration for a BLDC motor driven electric vehicle," *Eng. Sci. Technol. an Int. J.*, vol. 21, no. 4, pp. 704–713, Aug. 2018, doi: 10.1016/J.JESTCH.2018.05.003.

[29] M. J. Yang, H. L. Jhou, B. Y. Ma, and K. Ka. Shyu, "A cost-effective method of electric brake with energy regeneration for electric vehicles," *IEEE Trans. Ind. Electron.*, vol. 56, no. 6, pp. 2203–2212, 2009, doi: 10.1109/TIE.2009.2015356.

[30] M. K. Yoong *et al.*, "Studies of regenerative braking in electric vehicle," *IEEE Conf. Sustain. Util. Dev. Eng. Technol. 2010, STUDENT 2010 - Conf. Bookl.*, pp. 40–45, 2010, doi:

10.1109/STUDENT.2010.5686984.

[31] S. Mehta and S. Hemamalini, "A Dual Control Regenerative Braking Strategy for Two-Wheeler Application," in *Energy Procedia*, Jun. 2017, vol. 117, pp. 299–305, doi: 10.1016/j.egypro.2017.05.135.

[32] F. Caricchi, F. Crescimbini, F. G. Capponi, and L. Solero, "Study of bi-directional buck-boost converter topologies for application in electrical vehicle motor drives," *Conf. Proc. - IEEE Appl. Power Electron. Conf. Expo. - APEC*, vol. 1, pp. 287–293, 1998, doi: 10.1109/APEC.1998.647705.

[33] J. Partridge and D. I. Abouelamaimen, "The Role of Supercapacitors in Regenerative Braking Systems," *Energies 2019, Vol. 12, Page 2683*, vol. 12, no. 14, p. 2683, Jul. 2019, doi: 10.3390/EN12142683.

[34] "XV3560-2R7407-R | Eaton Bussmann Series 400F Supercapacitor -5 → +10% Tolerance, Supercap XV 2.7V dc, Through Hole | RS Components." https://uk.rs-online.com/web/p/supercapacitors/1699488/?cm_mmc=UK-PLA-DS3A-_-google-_-CSS_UK_EN_Passive_Components_Whoop-_-Supercapacitors_Whoop-_-1699488&matchtype=&pla-301057795080&gclid=Cj0KCQiAzMGNBhCyARIsANpUkzN17NDGZ619Fxh_vVl6uKW_MMsp-6Xp5tHUi6gIRg (accessed Dec. 08, 2021).

[35] M. Bahrami, H. Mokhtari, and A. Dindar, "Energy regeneration technique for electric vehicles driven by a brushless DC motor," *IET Power Electron.*, vol. 12, no. 13, pp. 3397–3402, Nov. 2019, doi: 10.1049/iet-pel.2019.0024.

[36] A. Adib and R. Dhaouadi, "Performance Analysis of Regenerative Braking in Permanent Magnet Synchronous Motor Drives," doi: 10.25046/aj030156.

[37] Y. Gao, L. Chen, and M. Ehsani, "Investigation of the effectiveness of regenerative braking for EV and HEV," 1999, doi: 10.4271/1999-01-2910.

[38] Consumer Product Safety Commission, "Consumer Product Safety Act (CPSA)," *Regul. Laws Stand.*, pp. 7–25, 1972, Accessed: Apr. 24, 2022. [Online]. Available: https://www.cpsc.gov/Regulations-Laws--Standards/Statutes/Summary-List/Consumer-Product-Safet-Act.

[39] B. Long, S. T. Lim, J. H. Ryu, and K. T. Chong, "Energy-regenerative braking control of electric vehicles using three-phase brushless direct-current motors," *Energies*, vol. 7, no. 1,

pp. 99–114, 2014, doi: 10.3390/EN7010099.

[40] "Analysis of regen on an ebike - Endless Sphere." https://endless-sphere.com/forums/viewtopic.php?f=2&t=7891 (accessed Apr. 23, 2022).

[41] Grin, "FH212_Std." https://ebikes.ca/shop/electric-bicycle-parts/motors/fh212-std.html (accessed Nov. 28, 2021).

[42] "Brushless Motor | Rozum Robotics." https://rozum.com/brushless-motors/ (accessed Apr. 23, 2022).

[43] dronenodes(organization), "Drone Motor Fundamentals – How Brushless Motor Works," *DN*, 2015. https://dronenodes.com/drone-motors-brushless-guide/ (accessed Apr. 25, 2022).

[44] "The fastest way to get more people to buy electric vehicles: Build more charging stations - Vox." https://www.vox.com/22463219/electric-vehicles-charging-station-infrastructure (accessed Nov. 28, 2021).

[45] "Greenpack – the smart Battery System – Standard Wechsel Akkus fuer mobile Anwendungen." https://www.greenpack.de/en/ (accessed Nov. 28, 2021).

[46] J. P. Aditya and M. Ferdowsi, "Comparison of NiMH and Li-ion batteries in automotive applications," 2008, doi: 10.1109/VPPC.2008.4677500.

[47] Bosch, "The eBike battery: long range, low weight, easy to charge - Bosch eBike Systems," 2020. https://www.bosch-ebike.com/en/products/batteries (accessed Apr. 19, 2022).

[48] A. Mohammad, M. A. Abedin, and M. Z. R. Khan, "Microcontroller based control system for electric vehicle," *2016 5th Int. Conf. Informatics, Electron. Vision, ICIEV 2016*, pp. 693–696, Nov. 2016, doi: 10.1109/ICIEV.2016.7760090.

[49] D. Chowdhury, S. Paul, R. Ghosh, A. Kundu, and T. Ghosh, "A Propose System of an Efficient Power Regeneration Technique in Automobile Using Arduino UNO," in *Communications in Computer and Information Science*, Jul. 2019, vol. 1030, pp. 361–377, doi: 10.1007/978-981-13-8578-0_29.

[50] "Omega Karma Strain Gages." https://br.omega.com/omegaFiles/pressure/pdf/SGK_2-ELEMENT90.pdf (accessed Apr. 21, 2022).

[51] "DS18B20 Programmable Resolution 1-Wire Digital Thermometer | Maxim Integrated." https://www.maximintegrated.com/en/products/sensors/DS18B20.html (accessed Apr. 21, 2022).

[52] "LM35 LM35 Precision Centigrade Temperature Sensors," 1999, Accessed: Apr. 21, 2022. [Online]. Available: www.ti.com.

[53] T. Ishikawa, T. Tsuji, S. Hashimoto, and N. Kurita, "A simple equivalent circuit for efficiency calculation of brushless DC motors," in *2013 International Conference on Electrical Machines and Systems, ICEMS 2013*, 2013, pp. 1133–1138, doi: 10.1109/icems.2013.6754394.

[54] E. Peponakis *et al.*, "A Simple Low Cost Setup for Thrust and Energy Efficiency Calculation for Small Brushless DC Motors EriGrid: Multi-Island project View project PV-ESTIA-Enhancing storage integration in buildings with Photovoltaics View project A Simple Low Cost Setup for T," 2016. Accessed: Dec. 12, 2021. [Online]. Available: https://www.researchgate.net/publication/301591985.

[55] S. G. Hemeda, A. M. Aboukarima, and M. Minyawi, "DEVELOPING A LOGGING UNIT FOR MEASURING AND RECORDING POWER DATA USING ARDUINO BOARD," *Misr J. Ag. Eng*, vol. 34, no. 2, pp. 2053–2072, 2017.

[56] A. Devices, "GENERAL DESCRIPTION," Accessed: Apr. 19, 2022. [Online]. Available: www.analog.com.

[57] "Understanding over-run brake systems | Caravan Chronicles." https://caravanchronicles.com/guides/understanding-over-run-brake-systems/ (accessed Dec. 12, 2021).

[58] "10 Best Trailer Brake Controllers In 2021." https://mechanicbase.com/reviews/best-trailer-brake-controller/ (accessed Dec. 12, 2021).

[59] M.-G. Unguritu and T.-C. Nichițelea, "Mechanical Systems from Automotive Industry: Modelling and Simulation of Trailer Systems," *Ann. Univ. CRAIOVA Ser. Autom. Comput. Electron. Mechatronics*, vol. 14, no. 41, 2017.

[60] TI, "LM5100, LM5101 LM5100 /LM5101 High Voltage High Side and Low Side Gate Driver Check for Samples: LM5100, LM5101 1FEATURES PACKAGE 2• Drives Both a High Side and Low Side N-• SOIC-8 Channel MOSFET • WSON-10 (4 mm x 4 mm) • Independent High and Low Driver L," 2004, Accessed: Apr. 18, 2022. [Online]. Available: www.ti.com.

[61] ST Microtechnologies, "L7800 Series Datasheet - Positive Voltage Regulators Datasheet," no. February. pp. 1–29, 2003.

[62] "DRV8803 datasheet | TI.com." https://www.ti.com/document-viewer/DRV8803/datasheet (accessed Apr. 18, 2022).

[63] "Vishay Siliconix TO-220AB ORDERING INFORMATION ABSOLUTE MAXIMUM RATINGS (T C = 25 °C, unless otherwise noted)." Accessed: Apr. 18, 2022. [Online]. Available: www.vishay.com/doc?91000.

[64] Microchip, "24LC512 Datasheet." http://ww1.microchip.com/downloads/en/devicedoc/21754m.pdf (accessed Apr. 25, 2022).

[65] AVR Studio®, "ATmega328P 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash DATASHEET," *ATmega328P [DATASHEET]*. pp. 6–288, 2016, Accessed: Apr. 16, 2022. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf.

[66] C. L. Huang, G. R. Chen, S. C. Yang, and Y. L. Hsu, "Comparison of High Speed Permanent Magnet Machine Sensorless Drive using Trapezoidal BLDC and Sinusoidal FOC under Insufficient PWM Frequency," in *2019 IEEE Energy Conversion Congress and Exposition, ECCE 2019*, Sep. 2019, pp. 321–325, doi: 10.1109/ECCE.2019.8912495.

14.Appendix

14.1 Full Code

```
//comms to memory
#include <Wire.h>
#define EEPROM_I2C_ADDRESS 0x50 //External memory chip address

/*
*ECARGO CONTROLLER
*MICHAEL JEANS
*BLDCREGEN_1.3
*HALLS VERSION
*PWM CONTROLLED SPEED/BRAKING
*DEVELOPMENT STAGE
*/

//////////////////SETUP////////////////////////////

void setup() {

////////////PIN DATA////////////////////////

//debug LED
pinMode(13, OUTPUT); //LED //change to pin13 for actual board
pinMode(LED_BUILTIN, OUTPUT);

// Halls input
  pinMode(0,INPUT);    // Hall 1
  pinMode(1,INPUT);    // Hall 2
  pinMode(2,INPUT);    // Hall 3

// Outputs for the Motor Drivers
  pinMode(3,OUTPUT);   // IN 1
  pinMode(5,OUTPUT);   // IN 2
  pinMode(6,OUTPUT);   // IN 3
  pinMode(9,OUTPUT);   // EN 1
  pinMode(10,OUTPUT);  // EN 2
  pinMode(11,OUTPUT);  // EN 3

// Sensor Inputs
  pinMode(A0, INPUT);   //Strain gauge
  pinMode(A1, INPUT);   //Winding Temp
  pinMode(A2, INPUT);   //Bike Speed

//Comms
  pinMode(SDA, OUTPUT); //SDA
  pinMode(SCL, OUTPUT); //SCL
```

```
  Wire.begin();         //initialize I2C busses
  Serial.begin(9600);   //To write to serial monitor for debug

//////////////////PWM AND INTERRUPT SETUP////////////////////////

//////////////////1ms timer zero interrupt////////////////////////

cli();//stop interrupts
  //set timer0 interrupt at 1kHz
  TCCR0A = 0;// set entire TCCR0A register to 0
  TCCR0B = 0;// same for TCCR0B
  TCNT0  = 0;//initialize counter value to 0
  // set compare match register for 1khz increments
  OCR0A = 249;// = (16*10^6) / (1000*64) - 1 (must be <256)
  // turn on CTC mode
  TCCR0A |= (1 << WGM01);
  // Set CS01 and CS00 bits for 64 prescaler
  TCCR0B |= (1 << CS01) | (1 << CS00);
  // enable timer compare interrupt
  TIMSK0 |= (1 << OCIE0A);
sei();//enable interrupts

// This sets PWM for pins 9 and 10 to 31372.55 Hz //CS divisor =1
TCCR1B = TCCR1B & B11111000 | B00000001;

// This sets PWM for pins 3 and 11 to 31372.55 Hz //CS divisor =1
TCCR2B = TCCR2B & B11111000 | B00000001;
} //closesetup

//////////////////Declarations//////////////////////

//////////////////////Functions//////////////////////
void halls();
void checkspeedload();
void memory();
void temp();
void loadcalc();
void commutation();

//////////////////////Interrupt//////////////////////

ISR(TIMER0_COMPA_vect);

/////////////Variables ////////////////////

//Hall sensors (3,2,1) + Hall value for easy calc
int Hall1;
int Hall2;
```

```
int Hall3;
int HallVal = 1;
int NewHallVal;

//speed levels for drive and brake modes
int MotorSpeedLevel = 0; //speed level of the motor
int BrakeSpeedLevel = 0; //braking level
int throttle = 0; //mapped throttle difference to set brake and motor speed.

//Load
int loadinit = analogRead(A0); //What is the load at rest/unloaded? Make this
the centre point
int load[10];
int p = 0;
int q = 0;
int aveload = 0;

//Bike Speed
int ecargospeed = 0; //real ecargo speed in rpm
//float wheelsize = 0.0016; //circumference in km for 20inch wheel plus hefty
tires
char steps= 0 ;
char neworderpointer = 0;
char orderpointer = 0;
int speed[100]; //speed averaging filter
int n = 0; //speed array pointer

//Temp
int windingtemp;     //winding temp

//Timing variable
char m = 0;
int i = 0;

//I2C
int loadlevelspeed;
int loadlevel;

//////////////////main loop/////////////////////////////////////

//MAIN

void loop(){

} //main

////////////////interrupt service routine//////////////////////
```

```
ISR(TIMER0_COMPA_vect){//timer0 interrupt 1kHz
  //PORTB ^= B00100000;// toggles bit which affects pin13     //debug
halls();
commutation();
  i++;
  if(i==100){  //100ms
  //digitalWrite(13, HIGH);  //debug
  digitalWrite(LED_BUILTIN, HIGH);  //debug
    checkspeedload();   //check speed and load
    i=0;        //reset counter
  m++;
  if(m==10){ //1000ms
  //digitalWrite(13, LOW); //debug
  digitalWrite(LED_BUILTIN, LOW);  //debug
    temp();   //check temp
  //  memory(); //print to memory
    m=0;      //reset counter
    } //close1000ms
  }//close100ms
}//close isr

//////////////////FUNCTIONS////////////////////////////////////

void halls(){     //check every 1ms
  Hall1 = digitalRead(0);  // read input value from Hall 1
  Hall2  = digitalRead(1);  // read input value from Hall 2
  Hall3  = digitalRead(2);  // read input value from Hall 3
  NewHallVal = (Hall1) + (2*Hall2) + (4*Hall3); //Computes the binary value of
 the 3 Hall sensors

    switch (NewHallVal){
      case 3:
        neworderpointer = 1;
      break;
      case 1:
        neworderpointer = 2;
      break;
      case 5:
        neworderpointer = 3;
      break;
      case 4:
        neworderpointer = 4;
      break;
      case 6:
        neworderpointer = 5;
      break;
      case 2:
        neworderpointer = 6;
```

```
      break;
    }//switch

  //calculate no of steps  //order = [3,1,5,4,6,2];
  if(orderpointer <= neworderpointer){
    steps = neworderpointer - orderpointer;
  }
  else{
    steps = (6-orderpointer) + neworderpointer;
  }
  orderpointer = neworderpointer;

  speed[n] = steps*320; //3 pole
  n++;
  if (n > 99){
    n=0;
  }
  HallVal = NewHallVal;
  HallVal = 2; //Debug and for testing
}//halls

void checkspeedload(){    //average speed/load for check EVERY 100ms
loadcalc();
  for(n = 0; n < 100; n++){
  ecargospeed = 0;
  ecargospeed += speed[n];
    }
  n=0;
  ecargospeed = ecargospeed/100;
}//checkspeedload

void memory(){        //print every second
//Loadlevel
loadlevel = map(throttle, 0, 255, 0, 7);
loadlevel = loadlevel << 5;
//Create loadlevelspeed byte hi largest bits are load level, 5 lowest are spee
d
loadlevelspeed = 0x0000 || ecargospeed;
loadlevelspeed = loadlevelspeed || loadlevel;

//print speed/temp/load
Wire.beginTransmission(EEPROM_I2C_ADDRESS); // transmit to device
  Wire.write(windingtemp);   // sends 1 byte
  Wire.write(loadlevelspeed);   // sends 1 byte containing load and speed
  Wire.endTransmission();    // stop transmitting
}//memory

void temp(){
```

```
//Caculate winding temperature
  windingtemp = analogRead(A1);
  windingtemp = map(windingtemp, 0, 1023, -40, 180);
}//temp

void loadcalc(){                  //include array for averaging
//Calculate load
  load[p] = analogRead(A0) - loadinit;
  p++;
  if (p > 9){
    p=0;
  }
  for(q = 0; q < 10; q++){
  aveload = 0;
  aveload += load[q];
  aveload = aveload/10;
  }

//map(value, fromLow, fromHigh, toLow, toHigh)
//Changing fromHigh and fromLow values to changes the max/min load value which
 is at 100% Duty
throttle = map(aveload, -
400, 400, 0, 255); //exceeding the init range will max out the new range
//Motor settings for braking vs driving
MotorSpeedLevel = map(throttle, 127, 255, 0, 255); //motoring is mapped to the
 top half of range
BrakeSpeedLevel = map(throttle, 0, 127, 255, 0);    // regenerative braking on
 bottom half of range
}//end loadcalc

// Commutation for Motoring
/*
PORTD contains the outputs for the HIGHside pins
AnalogWrite command is used to set the LOWside pins
(0 = OFF, 255 = ON or motorspeedlevel value that is controlled by the load sen
sor)
*/
void commutation() {

  //debug values to set PWM to drive @200, below max speed.
  ecargospeed = 1;
  throttle = 200;
  BrakeSpeedLevel = 200;
  MotorSpeedLevel = 127;
  //'HallVal' IS ALSO SET FOR DEBUG IN THE 'halls' FUNCTION

  if(ecargospeed>25){  //this 'if' may need optimised so downhill speed is hel
d at 25kph smoothly
```

```
  throttle = 0;
  BrakeSpeedLevel = 255;
  }
if (throttle > 127){
    switch (HallVal)
     {
      case 3:
        //Highside
        PORTD  &= B10010111;
        PORTD  |= B01000000;
        //Lowside
        analogWrite(9,MotorSpeedLevel);
        analogWrite(10,0);
        analogWrite(11,0);
        break;
      case 1:
        //Highside
        PORTD  &= B10010111;
        PORTD  |= B01000000;
        //Lowside
        analogWrite(9,0);
        analogWrite(10,MotorSpeedLevel);
        analogWrite(11,0);
        break;
      case 5:
        //Highside
        PORTD  &= B10010111;
        PORTD  |= B00001000;
        //Lowside
        analogWrite(9,0);
        analogWrite(10,MotorSpeedLevel);
        analogWrite(11,0);
        break;
      case 4:
        //Highside
        PORTD  &= B10010111;
        PORTD  |= B00001000;
        //Lowside
        analogWrite(9,0);
        analogWrite(10,0);
        analogWrite(11,MotorSpeedLevel);
        break;
      case 6:
        //Highside
        PORTD  &= B10010111;
        PORTD  |= B00100000;
        //Lowside
        analogWrite(9,0);
```

```
          analogWrite(10,0);
          analogWrite(11,MotorSpeedLevel);
          break;
        case 2:
          //Highside
          PORTD  &= B10010111;
          PORTD  |= B00100000;
          //Lowside
          analogWrite(9,MotorSpeedLevel);
          analogWrite(10,0);
          analogWrite(11,0);
          break;
       }
     }

   // Commutation for Regenerative Braking
   /*
   HIGHside transistors are always set to OFF during regen braking.
   AnalogWrite command is used to set the LOWside pins
   (0 = OFF, 255 = ON or brakespeedvalue value that is controlled by the load
sensor)
   */

*/
   else{
          //SET HIGHSIDE TO ZERO
            PORTD  &= B10010111;  //keep halls values etc.

          switch (HallVal)
         {
          case 3:
            analogWrite(9,BrakeSpeedLevel);
            analogWrite(10,0);
            analogWrite(11,BrakeSpeedLevel);
            break;
          case 1:
            analogWrite(9,0);
            analogWrite(10,BrakeSpeedLevel);
            analogWrite(11,BrakeSpeedLevel);
            break;
          case 5:
            analogWrite(9,BrakeSpeedLevel);
            analogWrite(10,BrakeSpeedLevel);
            analogWrite(11,0);
            break;
          case 4:
            analogWrite(9,BrakeSpeedLevel);
            analogWrite(10,0);
```

```
          analogWrite(11,BrakeSpeedLevel);
          break;
        case 6:
          analogWrite(9,0);
          analogWrite(10,BrakeSpeedLevel);
          analogWrite(11,BrakeSpeedLevel);
        break;
       case 2:
          analogWrite(9,BrakeSpeedLevel);
          analogWrite(10,BrakeSpeedLevel);
          analogWrite(11,0);
          break;
      }
  }
}
```