University of Southern Denmark

MSc in Engineering - Electronics
Master Thesis

Development of a Pendulum Control System

Authors:
Mikkel Skaarup Jaedicke
Thomas Søndergaard Christensen

Supervisor:
Leon Bonde Larsen

Date: 28-12-2017

Preface

This report is written in partial fulfilment to the requirements of a master thesis of 40 ECTS points at the University of Southern Denmark. It is written as a part of the authors' education as electrical engineers at the Faculty of Engineering.
Leon Bonde Larsen has been the the supervisor throughout the project.

Acknowledgment

We would like to express our gratitude towards everyone that helped us throughout the duration of this thesis. A special word of thank goes to Jesper Nielsen for his assistance and feedback on developing our PCB and Jørgen Maagaard for his help and ideas in relation to development of the joint mechanics.
Thank you to Carsten Albertsen for educative discussions on electronics design, PCB layout and enjoyable lunch breaks.

Perhaps most importantly, a word of appreciation to our supervisor Leon Bonde Larsen for his invaluable assistance and patience.

Finally, we would like to acknowledge that this project would not have been possible without the endless support of our friends and families.

Kind regards

Mikkel Skaarup Jaedicke
Thomas Søndergaard Christensen

The report, source code, schematics, data, plotting scripts and simulations can be found at:

`github.com/mikkeljae/development_of_a_pendulum_control_system`

Abstract

Underactuated systems are a growing market in the industry, with segways and robotic grippers as good examples. Throughout this report a cart-mounted inverted double pendulum system is developed. The system is intended as a platform for experimentation with underactuated systems. The requirements related to functionality, design and safety of such a system are found. A pendulum assembly consisting of two aluminium joints capable of wirelessly transmitting their angular position is implemented. The electronics and PCB design for a controller board is made to house the MicroZed, several safety features and the ability to drive a motor. The pendulum system is assembled and software is written to verify the basic functionality of the system. Data extracted from the movement of single pendulum has the waveform of damped sinusoidal verifying correct sampling and transmission of the joint angle. Controlling of the joint angles and cart position was not done due to time constraints.

Contents

List of Figures

List of Tables

List of Listings

1 Introduction

This project is done in an effort to develop a double pendulum on a cart system for experimenting with control of unactuated systems. In the first part of the report he inverted pendulum will be analysed in an effort to determine the requirements of a system capable of controlling such a pendulum system. Electronics and mechanics will be developed to adhere to the found requirements and software will be written to serve as a base platform for the system. Finally the report will be concluded with a verification and evaluation of every requirement of the designed system.

1.1 System Overview

Since the reader is unlikely to have precognition of the report they are about to read, this section is written in an attempt to provide an overview of the terminology used going forward. Figure 1.1 is a depiction of the finished design. Every arrow points at a crucial part of the design which is referred to repeatedly throughout the report. When the part is referred to it will be done by that name to limit any potential confusion.
Before you, the reader, fully embark on this journey, a few notes on the writing of this report. All component names, pin names, variables and states are written using the `teletype font`. All components mentioned throughout the report will have the datasheet cited on the first mention of the component.

1.2 System Users

As will be elaborated on in future chapters, the system developed in this report is intended as an educational platform to be used in future projects. It was proposed by the supervisor of the project, Leon Bonde Larsen who also set some of the requirements. Throughout the report four types of actors are referred to:

- **Project Owner:** The project owner, in this case also the supervisor, has proposed a system and also set certain requirements in the development of the system. In addition the project owner has been responsible for accepting any purchases made for the project.
- **Authors:** Perhaps not surprisingly, this type refers to the authors of this report who, coincidentally, are also the original designers of the system. They are expected to have indepth knowledge of every aspect of the system.

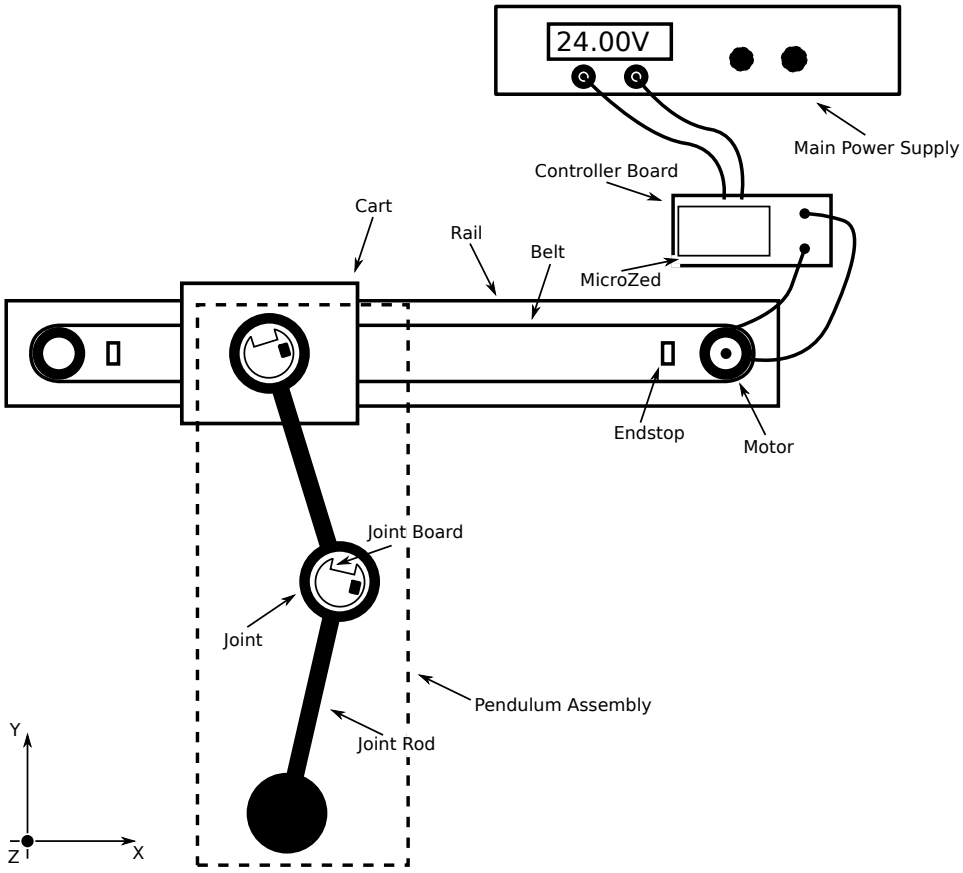


Figure 1.1: Overview of the completed system.

- **Developers:** A developer is a person who does work to improve or otherwise alter the functionality of the system. They are expected to have knowledge of the interfaces that exist, both in hardware and software. Some developers may wish to further improve the electronics or joint design, however, for the purposes of this report, a developer is expected to use the hardware as-is.

- **User:** The user is a person who applies control algorithms to the system in an effort to balance the system. They are expected to have knowledge of the software interfaces related to motor control, system calibration and joint control.

2 State of the Art

The system developed throughout this report is a proposal made by the project owner and developed by the authors. It is intended as a learning platform for students to experiment with and gain understanding of underactuated systems. An underactuated system is a system with more DoF (degrees of freedom) than actuators. Traditionally in control theory, an equal or greater number of actuators than DoF are used to cancel out the dynamics of a system. Underactuated systems is a vast field of research including examples in personal transportation such as the self-balancing scooter [40] or the Segway [39]. Both of these systems are essentially practical implementations of the inverted pendulum where an on-board control system will attempt to keep the device upright while maintaining a steady speed based on the tilt of the device. Also underactuated robotic grippers are seen in abundance in literature. [14] is an implementation of a three-phalanx finger gripper which utilises two underactuated fingers. When the fingers are brought together around an object, using the only actuated degree of freedom, the fingers wrap themselves around the object, regardless of size or shape. This means that a gripper can be far more versatile and much less dependent on the correct placement of the object. [27] implements a two-to-five finger gripper with belts on each finger to pull objects into the grasp of the gripper.

In control of underactuated systems it is required to exploit the natural dynamics of the system [47] in order to gain full control. Such systems are generally more agile and efficient than their fully actuated counter-parts.

A general example often used for teaching underactuated systems is the inverted pendulum. Numerous examples of such systems are described in literature, but three versions in particular are presented here to provide an overview:

- The pendubot [6] is a pendulum system consisting of one unactuated pendulum mounted on the end of an actuated pendulum.
- A rotary inverted pendulum [45] mounts an unactuated pendulum on a crossbar which is mounted horizontally on the axle of an upright DC-motor.
- The cart mounted inverted pendulum [49] actuates a cart along a rail in order to generate motion in the pendulum.

In common for all of the above systems is that the joint between the mounting platform and the pendulum assembly implements some form of encoder. This allows knowing the position of the pendulum and thereby controlling the system. The pendubot and the rotary inverted pendulum have the feature that they can control indefinetely without running out of space, as will eventually happen with the cart-on-a-rail system.

The system under development throughout this report is of the cart-on-a-rail type. Despite the apparent short-coming of limited control range, this version of the inverted pendulum system is quite popular in literature. [49] is an implementation of a single inverted pendulum while versions with series pendulums exist as well such as the double pendulum shown in [16] or the triple pendulum shown in [15].

Both [16] and [15] utilize encoders with a resolution of 8192 counts per revolution, or $\approx 0.044°$. [49] is implemented using just 5000 counts per revolution, or $\approx 0.072°$. In addition to pure positional control by encoders, some systems [7] also use current-mode control, CMC. CMC is a method whereby the voltage across a system can be controlled and, in the case of a motor, therefore also the velocity of the motor.

[16] and [15] both use a 1ms sampling time. The underlying system is different between the two with [15] being based on DSpace and [16] a Linux system with the real-time patch applied.

All of the literature mentioned above utilize the Euler-Lagrange equation of motion for developing the mathematical model required to control their respective pendulum systems. These equations assume point-masses and massless links between them. The authors have derived these equations for the double pendulum in appendix E. The original intent was to derive these equations for use in the control of the system. Since control was never done most of the value gained from this exercise was in gaining an understanding of the dynamics of the system to be developed.

3 System Analysis

Throughout this section an analysis of the system will be done and the overall design requirements of the system determined. In common for all of the cart mounted systems cited above is that they consist of a cart ca-

pable of moving along a rail with a pendulum mounted to it. This project will aim to create a double pendulum system. By actuating only the cart, the pendulums will not be directly actuated and the entire system therefore underactuated. The connection point between the cart and the first pendulum and between every consecutive pendulum should be a joint capable of rotating freely. Performing swing-up of the pendulum assembly requires knowledge of both the cart position on the rail and the position of every joint in the pendulum assembly.

The designed system should be a teaching platform first and foremost. This means that a number of users with varying levels of understanding of the underlying hardware are likely to handle the system throughout its lifetime. Therefore the system should be simple and safe to use. The exact definition of simple will clearly vary with different users but users are generally expected to have a working understanding of Linux. For this reason the project owner has requested that the main computation platform is based on the Xilinx Zynq-7 series architechture which provides the possibility of running Linux on the internal ARM cores while still maintaining the speed and flexibility of an FPGA. The authors have prior knowledge of the use and design of systems with the MicroZed development board and will therefore utilize this platform in the development of this project. The original intent was to develop and implement a kernel driver which would present the system to the user as a `tty` interface, however this quickly turned infeasible due to the time constraints of the project and will be left for future projects. Further design of the system should still maintain focus on the eventual goal of making the system accessible through Linux.

Safety around the system is important since the movement of a double pendulum is chaotic and therefore inherently unpredictable. A means of cutting power to the motor in case of emergency should be present. Cutting power to the motor should be possible both manually and automatically. It should happen independently of any programming so that faulty programming cannot interfere with the safety handling. End stops should be mounted on the rail in order to avoid damage to the system in case of a run-away cart. It is likely that the system should remain caged once finished so as to ensure that no person is in the path of the pendulum during run-time but the requirements to such a safety system are very much dependent on the system around which it should be developed and will be left as a future project to be handled before the pendulum system is given to a user. In addition to the above, the system should also be tolerable to some amount of misuse

such as blocking of the cart and similar unexpected situations.

Since the focus of the authors is electronics and embedded systems design, this will also be the main focus of this project. Therefore, rather than buying finished solutions for driving the motors, the design of electronics will be done by the authors. The electronics include a board for motor control and main computation as well as a board for interfacing the encoder in the joints, the specifics of which are discussed throughout the report.

Since creating the final version of the software is infeasible given the time frame, another solution is required to validate the functionality of the system. For this reason a real-time software system should be developed which correctly exercises every part of the developed pendulum system. Additionally the distribution of functionality between the FPGA and the ARM cores should be determined to optimally make use of their abilities. The overall software design should be determined prior to initiating programming. Programming of individual components such as drivers for sensory equipment should be done bottom-up and integrated into a larger context once confirmed functioning.

Early in the process the authors inherited the rail and cart system from a previous project. This includes a Maxon 148867 motor with an included `HEDS5540` encoder, both of which will be explored further in later sections. The rail system is used as-is since it is expected that any shortcomings of the rail system in relation to the double pendulum application will be revealed in using the finished system. An evaluation of the usefulness of the rail system should be done on the finished system.

3.1 Requirement Specification

The requirements specified below are tested and verified in section 8.

Functional:

1. System should consists of a double pendulum mounted on a moveable cart.
2. Cart should be actuated by the Maxon 148867.
3. The pendulum system should be controlled by a MicroZed.
4. Position of the cart and joint angles should be measured.

5. Any software developed for the pendulum system should be real-time.

6. System should be easily accessible to users.

Safety:

7. System should not break during operation by users.

8. System safety should not rely on the programming of the system.

Design:

9. Software design is done top-down, whereas subcomponents are written bottom-up.

4 Controller Electronics Development

Throughout this section the electronics concerning the controller is developed. This includes a thorough analysis of the required functionality which will yield a requirement specification. The remainder of the design is then done to comply with this specification. Finally the design will be tested to verify that it does comply with the given requirements.

4.1 Analysis

In order to develop a controller board capable of controlling a double pendulum, the required functionality of the board needs to be determined. This section will constitute an analysis of this functionality and of possible solutions to desired functionality.

4.1.1 Communication

The controller board should be able to communicate wirelessly with the joints as will be discussed in section 6.1.1. The wireless link should provide sufficient bandwidth that the angular position of the joint can be transmitted sufficiently frequently. From section 6.1.2 it can be seen that the encoder chosen for the joints allows 7200 counts per revolution. This can be represented using 13 bit. Adding a 3 bit identifier makes the payload of a message 16 bit. This is a rough estimate which will be used for the purposes of this analysis. A number of different protocols exist which could be used in this project. Three were under consideration:

- **TCP/IP over WiFi:** TCP/IP is the defacto standard for communication between computers, both wired and wirelessly. This, along with the very large throughput, makes it an enticing option. Considering the rather small message size this protocol does impose a significant amount of overhead. In addition to the communication overhead, TCP/IP also requires some form of driver, making it difficult to forego the use of Linux in the joints, an OS that increases the requirements to the processing unit in the joint considerably.

- **Bluetooth:** The Bluetooth protocol is developed specifically for low power, short range data transmission and is used extensively in various mobile applications [52]. As with the previous option, a considerable amount of overhead is required in setting up bluetooth as well as in the transmission of packets, making it less desirable.

- **Radio Frequency:** While not strictly a protocol, using an RF transceiver eliminates the additional overhead associated with the previously mentioned protocols. RF transceivers come with a number of different interfaces such as SPI or I2C and will simply transmit the data received through that interface. This makes it easy to communicate using this method through bare-metal code or even VHDL.

Considering the above it was chosen to use an RF transceiver to facilitate the communication between the joints and the controller board.

4.1.2 The MicroZed Development Platform

The MicroZed [5] development platform is a Zynq-7000 series based platform which the authors have worked extensively with in previous projects [25]. It was developed by Avnet specifically to be implemented in a product and as such there are few built-in features to the board, resulting in a reasonably small footprint and an efficient interface. Most of the design done by the authors in [25] can be reused with limited modifications.

One thing of note is that the Zynq-7000 series of chips requires a specific startup/shutdown sequency, detailed in [30], which ensures that signal integrity is maintained during startup and shutdown of the platform. This functionality should also be verified in this project.

4.1.3 Motor Driver

It is necessary to device some form of driving circuitry for the motor driving the belt. The motor should be able to move in both directions. An H-bridge allows this bidirectional movement. By properly switching the MOSFETs in the bridge the average voltage across the motor can be controlled and therefore also the speed and direction of the motor. As such, an H-bridge must be designed for the controller board. This will in turn require that an H-bridge driver circuit is designed.

4.1.4 Capacitor Bank

From simulation it was found that a capacitor bank across the supply near the H-bridge is required in order to protect the remainder of the PCB against ripple current and voltage spikes. See section 4.3.3 for an in depth discussion. The necessary amount of supply capacitance must be determined.

When introducing a large amount of capacitance in a circuit, a problem known as inrush current arises [26]. This happens when the circuit is powered on and the initial charging of the capacitors occur. Due to the amount of capacitance, the circuit will draw a very large current for a significant amount of time which can potentially destroy parts of the circuit. For this reason it is necessary to design circuitry which will temporarily limit the current through the circuit until the supply capacitors are charged. Such a circuit is often comprised of a relay in series with a resistor.

4.1.5 Power Dissipation and Heat

The MOSFETs driving the motor will have to carry and switch a sizeable current. Carrying the motor current will result in some amount of power dissipation in the MOSFETs as they have a non zero drain-source `on` resistance, $R_{ds(on)}$. Switching the motor current `on` and `off` also represents a source of power dissipation in the MOSFETs. The total power dissipation leads to an increase in the temperature of the MOSFETs. As the MOSFETs have a maximum operating temperature a heat sink should be connected to the MOSFETs in order to achieve a higher cooling effect.

4.1.6 Voltage Rails

The components chosen for the system all impose some requirement as to what voltage rails should be present on the controller board.

- **Maxon 148867 DC Motor:** This motor is rated at a nominal voltage of 24V. Therefore a 24V rail will be created to supply the motor. In addition to the motor, the 24V rail will also be used to generate any subsequent rails. The maximum current draw possible by the Maxon motor is its stall current. According to the datasheet [31] this is 80.2A. The authors wish for the system to be able to handle the stall current so the 24V rail should be capable of supplying at least that plus the current draw of any subsequent rails.

- **MicroZed:** This board is powered from a 5V rail and requires a 3.3V rail to power the I/O banks of the Zynq-7020 chip. The design of the power delivery for this board was done by the authors in the aforementioned project [25] and it was found that the MicroZed can draw a maximum of 1.8A from the 5V rail and up to 1A from the 3.3V rail. The 3.3V rail is generated from the 5V rail in that design. Total current draw from the 5V rail resulting from the MicroZed is therefore

no larger than 2.5A. An additional 2.5V reference is created in that design which is used only to facilitate the correct startup of the Zynq chip. The 5V rail was designed to be created from a 2S Li-Po battery at 8V and the maximum input voltage of the DC/DC converter is 18V. To reuse this design it is necessary to generate a rail lower than the 24V rail for the 5V rail to be generated from.

- **HIP4081:** This component is the motor driver used in the project. The choice of this component is discussed in more detail in section 4.3.1. A 12V rail will be used to power this component. This voltage is used by the `HIP4081` to generate the gate signals. The 12V rail will also be used to generate the 5V rail. This is within the specification for input voltage of the DC/DC converter of the 5V rail. Due to the choice of using the `HIP4081`, it is necessary to do bootstrapping (see section 4.3.2). The current drawn from the bootstrap circuit is decided in large part by the designer and as such the current requirement of the 12V rail is somewhat flexible. For this reason the current rating of the 12V rail should be as high as is practically and economically possible.

A number of additional components are likely to be required on the controller board which will all impose additional current draw on their respective rails. These are considered negligible when compared to the major consumers mentioned above.

An overview of the hierarchy of the different rails can be seen in figure 4.1. The required accuracy of each rail is determined by the most sensitive component on that rail and is presented in table 4.1. Since the 24V and 12V rails do not have any strict requirements imposed on them, a $\pm$1V requirement is set. The 2.5V reference is used only as a reference voltage and therefore does not have a formal current requirement.

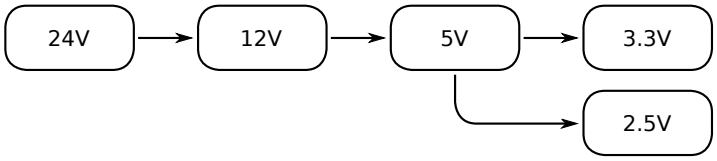


Figure 4.1: Overview of the voltage rail hierarchy in the controller board design. The 24V rail is external and is used to generate the 12V rail which generates the 5V rail which, finally, generates the 3.3V rail and 2.5V reference.

Rail [V]	24	12	5	3.3	2.5
Accuracy [V]	±1	±1	±0.5	±0.25	±0.03
Current [A]	85	-	2.5	1	-

Table 4.1: Voltage rails to be created on the controller board and their specification.

4.1.7 Motor Current Sensing

In the State of the Art analysis an example of an inverted pendulum system using current mode control was presented [7]. Allowing for this feature will further increase the usefulness of the platform for experimentation. In order to implement current mode control it is necessary to accurately measure the current being drawn by the motor at any given time. Current measurements are generally done by either hall effect based sensors or shunt resistors as described in [55]. Hall effect based sensors are usually better for high current applications where adding a resistor in series with the load is infeasible. Additionally, their interference with the circuit they are measuring is negligible. Hall effect sensors however, do come at a significant cost compared to the alternative described below. A shunt resistor used for current measurements is simply a resistor with a very low resistance, typically $< 100\text{m}\Omega$, and a high power rating ranging from a few watts into 10's of watts depending on the application. By adding such a resistor in series with the load, the voltage across the resistor can be measured and from that the current through the load determined. For this project a shunt resistor will be used since it is regarded as the simpler, cheaper option.

4.1.8 Safety

It is expected that, eventually, an error may occur that causes the cart to move uncontrollably. A safety system should be constructed in such a way that it will prevent the cart from crashing violently or the electronics to self-destruct. It should be independent from the remaining system and no programming should be involved in determining whether a safety condition is met.

A safety condition is defined as a situation that, when not met, indicates that a critical or dangerous situation has occured with the system. When a condition is met, that situation has not occured.

The safety system for this platform should be designed in such a way that,

when activated, it will cut power to the motor. In order to more easily identify the cause of the fault, the remaining electronics should remain operational to maintain the current program status. Clearly, the system should function only if all safety conditions are met. This will be ensured by creating an aggregation circuit which compares all of the safety conditions, creating one signal which will be capable of closing the main relay if necessary.

The system should be designed to have endstops. These are in place to quickly cut power to the system if the cart reaches the end of the rail. Creating these endstops can be done in different ways. Three potential implementations are presented here:

Mechanical Switch: The simplest form of switch is the mechanical switch. The switch should be mounted in such a way that the cart would move into the switch, therefore activating it. This approach is not without issues. Mounting of the switch should be done in such a way that it is not in the direct path of the cart. If the cart is traveling at full speed it is unlikely that the cart would stop before crashing into the switching mechanism. A switch mounted in this fashion would need to be rather robust. Some other mounting solutions could be thought of that do not suffer from this problem, especially if a flexible microswitch is used. These would require significant extra design to properly mount the switch. Mechanical switches can be fragile and may not be sufficiently durable, considering the usecase. All mechanical switches have one drawback in common: they are mechanical and will be subject to wear over time, eventually requiring maintainance or replacement.

Hall Effect Sensor: This type of sensor measures magnetic fields and produces a voltage proportional to the strength of the field. By placing a small neodymium magnet on the cart and mounting a hall effect sensor on the rail in such a way that they coincide would allow for detecting when the cart is above the sensor. If the sensor is combined with a schmitt-trigger circuit the output could be a binary result, either the endstop is reached or it is not. Using this method requires that a magnet is placed on the cart itself and that a sensor is mounted to the rail. It should be noted that accelerating rather strong magnetic fields back and forth on the platform may not be the least electrically noisy solution one could think of.

Infrared Transceiver: Infrared transceivers come in different configurations but generally the device is comprised of an LED and a photo transistor. The LED is optimised for emitting infrared light, or radiation (IR). In the configuration under consideration in this project the LED emits IR towards the photo transistor, opening it. When an object passes between the LED and the transistor the transistor is closed. This change can be detected using a simple schmitt-trigger circuit. With this sensor type it is important to realise that the LED will not be the only emitter of IR in the vicinity. Any type of lamp will emit IR, especially glowbulbs but also sunlight contain a significant amount of IR. When using this sensor the circuit should be designed in such a way that only the effect of the IR from the LED on the photo transistor will affect the circuit.

While all of the above have their drawbacks, it was decided to use the infrared transceiver. This sensor allows the greatest reliability while being reasonably simple to mount on the platform in that it requires no modification to the cart or rail. A simple 3D print can be done to mount the sensor.

In addition to the endstops, also an emergency button, preferably red, should be able to cut power to the motors.

4.1.9 Schematic Revision

From experience of the authors, a PCB layout is never error free in the first revision. This has been true even though a tremendous amount of work went into designing the PCB. Therefore a strategy was devised to increase the likelihood of producing an error free PCB. The strategy consist of five steps that should be done during the design of the board:

- Documentation
- General inspection
- Datasheet and report comparison
- Footprint inspection
- Peer-review

The content of each step will be elaborated here.

Documentation: Each circuit should be documented to ensure that the design is done correctly. Documenting a circuit requires studying the datasheet of the components and ensuring that each component meets the requirement. Furthermore it requires a thorough inspection of the expected functionality of the circuit. This step leads to correcting errors where a component is used incorrectly in respect to the desired functionality of the circuit.

General Inspection: General inspection of the schematic includes finding typographical errors on signals, components, resistor values, etc. Furthermore each signal should be inspected to ensure that it is connected correctly and it should also be decided if it should be connected to an external header for measurement purposes.

Datasheet and Report Comparison: Each circuit in the schematic should be compared to its datasheets recommendation and the documentation written in the report. This step essentially ensures that each circuit has been verified more than one time. One could think that this step is irrelevant, but the authors believe it is important due to the complexity of the schematic. After completion of this step, all circuits have been checked at least two times and the schematic should be correct.

Footprint Inspection: Provided that the schematic is correct, the only possibly source of errors is the component footprints. Inspecting the footprint consists of verifying the correct pin assignment and the correct land patterns. Both are done by inspecting the datasheet of each component. After completion of this step, both the schematic and footprints should be correct and the board is expected to be free from errors.

Peer-review: In this part of the process the board design, schematic and footprints should be correct and everything has been verified by the authors multiple times. It is generally a problem that when working with a design for a long period of time, some errors are very hard to find for the designers themselves. Therefore the design should be reviewed by a person that has not taken part of the design of the board. The review should be done by pointing out the main features of the board and discussing the found components, schematic and the board layout.

4.1.10 PCB Layout

Some considerations are important when designing high power or switching circuitry, both of which are categories which the controller board fit into. When part of the board is exposed to large currents, maintaining signal integrity on the remainder of the board becomes important. This is done in large part by controlling grounds in the PCB. An unbroken ground plane should be present in the design to ensure a low-impedance current path in the PCB [41]. In addition to unbroken ground planes, they should also be seperated based on their role in the circuit. Often two, three or even more ground planes exist in circuits. A common distribution is an analog ground, digital ground and power ground. Analog signals are often sensitive to noise and should be shielded from the switching noise of the digital circuitry. Similarly, high currents in a ground plane is likely to cause voltage drops across it. This may cause issues in the digital circuitry.
Having several ground planes can cause issues with currents circulating in them. In order to avoid this it is crucial that the ground planes are connected in one place only. This is known as a star ground connection [56].

Since at least part of the PCB is likely to carry significant currents it is necessary to appropriately size the width of traces and the thickness of the copper.

4.1.11 Component Selection and Placement

There are limits to the production capabilities of the equipment, both of the manufacturer producing the PCB and the equipment available to the authors at SDU. For this reason some restrictions are set on the casing of components and their placement on a PCB. Generally, throughhole components should be avoided where possible since these obstruct not only the layer they are placed on but also every other layer throughout the PCB and so surface mount technology (SMT) components are preferred. SMT components come in many sizes and some are simply too small to realistically route and solder with the available equipment. Generally, SOT-23, SOIC and 0805 packages and similar sizes are preferred where possible. With the equipment available at SDU only components on one side of the PCB can be placed and soldered with the pick-and-place machine and oven. Due to this, components should only be placed on the top layer when possible. Larger components, such as big capacitors, can easily be hand-soldered and can therefore be placed on the bottom side of the PCB.

A number of signals should be routed to headers to allow for easier debugging and monitoring of the circuits. For simplicity these headers should all be of the same type. The JST ZH-series connectors should be used as they are commonly used at SDU, they are SMT and sufficiently small.

4.1.12 Testing and Prototyping

Some of the developed circuits will consist of many components and complex ICs. With such complex circuits, errors in a schematics can easily appear. Therefore the functionality of the most important parts of the circuit should be tested on prototype PCBs before ordering an expensive PCB.

4.1.13 Board Layout and Considerations

A number of options are available when considering the general layout of the controller circuitry. Different circuits can be segregated into individual boards, simplifying the layout and production of each board but imposing requirements on board-to-board connections between different circuits. One possible configuration under consideration is the segregation of the digital circuit and the power circuits. This would allow a different number of layers and different copper thickness on the two PCBs. Extra layers are beneficial when routing the multitude of signals in a digital circuit but also add extra cost to the production of the circuit. Similarly, extra copper is likely to be necessary on the power circuit but also adds cost. Exploring various manufacturing options revealed that segregating the boards into one board with added copper and one board with added layers added significant cost to the order. For this reason it was decided to produce just a single PCB with extra copper and four layers, housing all of the electronics related to the controller.

4.2 Requirement Specification

The requirements specified below are tested and verified in section 4.6.

Functional:

10. Speed and direction of the motor should be adjustable through PWM from the MicroZed.
    - H-bridge of MOSFETs capable of switching motor current direction.
    - H-bridge driver translating logic signals to gate signals.
    - High-side drive capabilities.
11. Motor current should be measurable through the ADC input on the MicroZed.
    - Shunt resistor should be added to allow for measurement of the motor current.
12. Voltage rails for the various components in the circuit.
    - 2.5V ±0.03V.
    - 3.3V ±0.25V, 1A.
    - 5V ±0.5V, 2.5A.
    - 12V ±1V, The amperage of this rail should be as high as economically and practically viable.
    - 24V ±1V, 85A.
13. Voltage transients on voltage rails should be minimized.
    - Bulk capacitors on 24V rail to supply current to motor.
    - Properly sized input and output capacitance of DC/DC power supplies.
    - Appropriate copper volume for ground and power wires.
14. The MicroZed should be correctly interfaced.
    - All signals to the MicroZed should be routed on the PCB to appropriate I/O ports.
    - Start-up and shut-down power sequence and voltage levels should be correct.

15. A Heatsink should be connected to the MOSFETs to avoid overheating of the H-bridge.
16. Hardware for wireless communication using RF should be implemented.

Design:

17. Important parts of circuitry should be tested on prototype PCB.
18. The PCB layout should be reviewed according to the developed strategy.
    - Documentation.
    - General inspection.
    - Datasheet and report comparison.
    - Footprint inspection.
    - Peer-review.
19. The PCB layout should be designed with debugging in mind.
    - Test points should exist for all voltage rails and relevant signals.
    - All test points should be wired to JST ZH-series connectors.
20. The PCB should be designed to accomodate appropriate power levels.
    - Determine required trace width.
    - Ground planes should be segregated for power and digital circuitry.
    - Components of the motor driver should be sized to accomodate the stall current of the motor.
21. PCB soldering should be possible at SDUs facilities.
    - Component casings should be SOT-23, SOIC or 0805 where possible.
    - Avoid through-hole components.

Safety:

22. Any triggered emergency condition should halt the cart.

    - Emergency conditions should be implemented.
        - Emergency button for human interaction.
        - Endstops for detecting cart run-away.
        - Emergency signal from the MicroZed.
    - Relay should switch power to motor.
    - Relay should be active only when safety conditions are met.
    - Power to remaining electronics and MicroZed should not be affected by a safety condition.

4.3 Circuit Design

Multiple electronics components are required to realise the functionalities described in the requirement specification. This section explores the selection of those components and design of the corresponding circuits.

4.3.1 MOSFETs and H-Bridge Driver

The motor that needs to be driven is a Maxon 148867 Brushed DC Motor. This is a 150W motor with a nominal current of 6A and a stall current of 80A. Clearly, the system should under normal use not come anywhere close to stalling the motor but considering the use case, experimentation with control algorithms from users, it may be beneficial to design the circuitry such that it can withstand being stalled, for at least a short period of time. In order to reach this goal, it is necessary to size the components for at least some amount above 80A continuous. The `IRF100B202` MOSFET [44] is a 100V/97A MOSFET. It is made in the standard TO-220 housing. This housing allows for easy mounting of a heatsink and due to its wide application, there are many shapes and sizes to choose from. Table 4.2 holds a list of the relevant parameters.
It should be noted that even though V_{th} is maximum 4V, this is not enough to fully open the MOSFET. Rather, this voltage is where the MOSFET will start to conduct. Figure 4.2 reveals that the MOSFET is not fully conducting until $V_{\text{gs}} > 6\text{V}$.

Parameter	Min	Typ	Max
V_{th} [V]	2	-	4
$R_{\text{ds(on)}}$ [mΩ]	-	7.2	8.6
Q_{g} [nC]	-	77	116
I_{d} [A]	-	-	97
V_{ds} [V]	100	-	-

Table 4.2: Relevant parameters of the `IRF100B202` MOSFET [44] chosen for the full-bridge. Here, V_{th} is the gate threshold voltage, $R_{\text{ds(on)}}$ is the drain-source on resistance, Q_{g} is the total gate charge, I_{d} is the maximum continuous drain current and V_{ds} is the minimum guaranteed drain-source breakdown voltage.

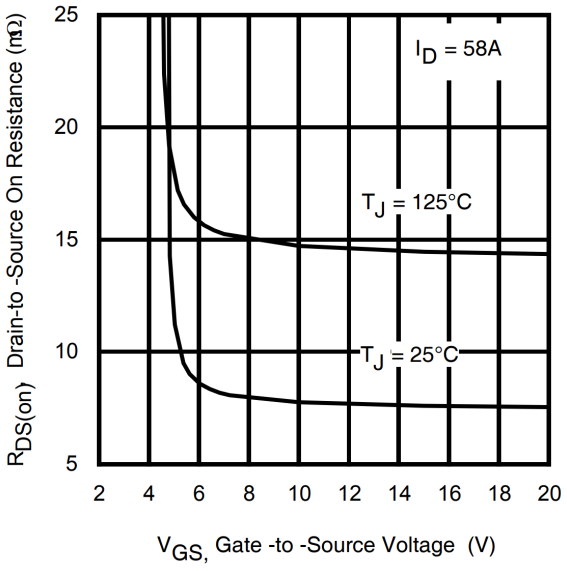


Figure 4.2: Transfer characteristic of the `IRF100B202` as per the datasheet. Inspecting the graph reveals that the maximum I_{d} of 97A is reached at $V_{\text{gs}} \approx 5.5 \rightarrow 6.5$V.

Having chosen a MOSFET for the full-bridge, some form of driver must be designed for the full-bridge. For this task, ready-made full-bridge drivers exist on the market that incorporate all of the electronics required to generate the driving signals, requiring only a PWM signal from the designer. In order to choose one, a few parameters must be fulfilled.

- **Output Current:** The amount of current necessary to properly drive the MOSFET. This value depends on the switching frequency of the application, the gate-to-source voltage (V_{gs}) and the total gate charge

of the MOSFET.

- **Driving Voltage:** In order to ensure that the MOSFET is turned on quickly and fully the driver should be able to supply a voltage well above the gate threshold of the MOSFET.
- **High-Side Drive Capabilities:** The high-side MOSFET of the full-bridge is referenced not to ground, but to a floating reference. This means that in order to switch on this MOSFET, V_{gate} needs to be boosted compared to the floating reference. This is known as bootstrapping and is explained in more detail in section 4.3.2.
- **Shoot-Through Protection and Deadtime:** Due to the structure of the full-bridge it is important that two MOSFETs on the same half-bridge are never `on` at the same time as this would result in a short from V_{cc} to ground. This is resolved with a combination of deadtime (a delay where no MOSFET is `on`) and a logic table that ensures that an illegal switch combination can never happen.

The `HIP4081AIBZ` [18] full-bridge driver meets all of the above requirements. It can supply up to 2.5A on the output pins with a voltage approximately 1V below V_{cc} , well above the required V_{th} . It also allows for an external bootstrap circuit, which makes it able to drive the high-side MOSFETs. Since the motor should, preferably, be driven outside the audible frequency range, it was chosen to set the switching frequency at 22kHz. This frequency was chosen over an even higher frequency because increasing the frequency also increases the switching losses. At this frequency there is $\frac{1}{22000} \approx 45\mu$S per cycle. Well within this time the gate should be fully opened. The switching time should be calculated to verify this. [51] describes switching characteristics of MOSFETs without taking parasitics into account. Approximating the turn-on time of the MOSFET is sufficient here as it just needs to be verified that the MOSFETs are fully open within a small fraction of the switching period. [51] approximates the turn-on time of a MOSFET in equation 4.1.

$$t_{switch} = R_G \cdot C_{iss} \cdot ln\left(\frac{1}{1 - \frac{V_{gp}}{V_{GS}}}\right) \tag{4.1}$$

Where t_{switch} is the switching time, R_G is the total gate resistance, C_{iss} is the effective input capacitance of the MOSFET, V_{gp} is the gate plateau voltage and V_{GS} is the voltage applied to the gate by the motor driver. Accounting for voltage drops in the bootstrap circuit discussed later, it is

assumed that V_{gs} is 8V. By inspection of figures in the datasheet, V_{gp} is estimated to 4.5V. R_G and C_{iss} are datasheet values.

$$t_{switch} = 9.4 \cdot 4.48 \cdot 10^{-9} \cdot ln\left(\frac{1}{1 - \frac{4.5}{8}}\right) = 34.8nS \tag{4.2}$$

This MOSFET switching time is orders of magnitude smaller than the PWM switching period.

The `HIP4081` also has floating drive circuitry, meaning that by referencing the high-side driver to source of the high-side MOSFET, the driver can be made to output a voltage higher than V_{cc} . This does require a few external components for bootstrapping. The choice of these components and a more in depth explanation of the procedure is given in section 4.3.2. Finally, the component has a user-programmable deadtime and shoot-through protection.

4.3.2 Bootstrap Circuit

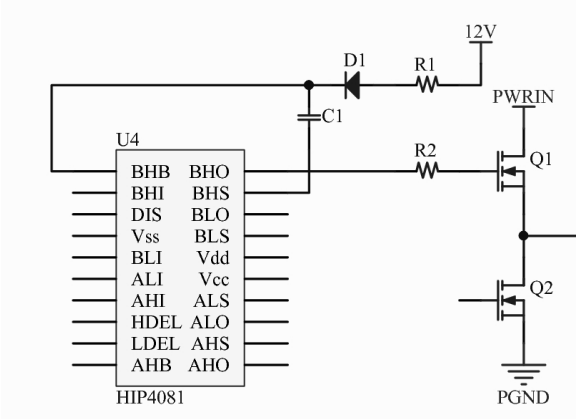


Figure 4.3: Bootstrap circuitry consisting of a diode, `D1`, a capacitor, `C1` and a resistor, `R1`.

The chosen gate driver has floating drive circuitry for the high-side MOSFETs but needs external bootstrap circuitry for providing the higher voltage. A standard bootstrap circuit consists of a diode, a capacitor and a resistor as shown in figure 4.3. The advantage of using a bootstrap circuit to supply a high voltage for the floating drive is that it is very simple. The disadvantage is that it sets a limit for the maximum and minimum duty cycle as the

bootstrap capacitor needs to be charged fully in each period. The bootstrap circuitry for this project will be designed to have a maximum duty cycle for the higher MOSFETs of 95%, thus meaning that the charging of the bootstrap capacitor should take no longer than 5% of a period.
The bootstrap diode needs to have a fast recovery time, preferably below 100 ns [29], to minimize the energy fed back to the supply from the bootstrap capacitor. `US1M-E3` [50] was chosen as it has a reverse recovery time of 75 ns and comes in SMD packaging. It allows for a maximum average forward current of 1 A and a maximum forward voltage drop of 1.7 V [50]. Determining the values for the bootstrap capacitor was done using the calculations shown in [10]. The maximum allowed voltage drop across the bootstrap capacitor during `on` time needs to be determined:

$$\Delta V_{boot} = V_{DD} - V_F - V_{GSMIN} \tag{4.3}$$

Where ΔV_{boot} is the voltage drop across the bootstrap capacitor during `on` time, V_{DD} is the supply voltage, V_F is the forward voltage drop across the diode and V_{GSMIN} is the minimum desired `gate-source` voltage. V_{GSMIN} was chosen to 8 V to to ensure a high enough gate voltage to turn on the MOSFET. The chosen MOSFETs have a maximum gate threshold voltage of 3.5 V. Inserting the values yields the maximum allowed value of ΔV_{boot}:

$$\Delta V_{boot} = 12 - 1.7 - 8 = 2.3V \tag{4.4}$$

The total charge that needs to be supplied from the bootstrap capacitor is the gate charge of the MOSFET along with the quiescent and leakage currents in the `on` time:

$$Q_{total} = Q_{gate} + Q_{ls} + (I_{lkgs} + I_{qbs} + I_{lk} + I_{lkd}) \cdot t_{on} \tag{4.5}$$

Where Q_{gate} is the maximum total gate charge, I_{lkgs} is the gate-source leakage current, I_{qbs} is the gate driver quiescent current, I_{lk} is the gate driver leakage current, I_{lkd} is the diode leakage current, t_{on} is the `on` time and Q_{ls} is the charge required by the internal level shifter in the gate driver. t_{on} can be calculated knowing that the worst case duty cycle for the high-side MOSFET is 95% and the switching frequency is 22 kHz. Inserting the values, shown in table 4.3, yields the minimum charge value of the capacitor:

$$Q_{total} = 118[nC] \tag{4.6}$$

The minimum value of the bootstrap capacitor can now be calculated:

$$C_{boot} = \frac{Q_{total}}{\Delta V_{boot}} = \frac{118}{2.3} = 51.2[nF] \tag{4.7}$$

Q_{gate}	Q_{ls}	I_{lkgs}	I_{qbs}	I_{lk}	$I_{lkdiode}$	t_{on}
116 [nC]	3 [nC]	100 [nA]	-30 [μA]	1 [μA]	1 [nA]	43 [μS]

Table 4.3: Parameter values used in equation 4.5. I_{qbs} and I_{lk} are found in the gate driver datasheet. Q_{ls} is set to 3nC for all high voltage gate drivers [10]. I_{lkgs} and Q_{gate} are MOSFET datasheet values and $I_{lkdiode}$ is a diode datasheet value. t_{on} is calculated using 95% duty cycle and 22kHz switching.

A ceramic capacitor will be used as it has a negligible leakage current. Looking for SMD ceramic capacitors it was found that in the range of the found minimum value is 56, 68, 82 and 100 nF. 56 nF is, in theory, a big enough capacitor for the circuit, but a bigger capacitor is an advantage as ΔV_{boot} will then decrease. When increasing the size of the capacitor it should be noted that the initial charge current will become larger. It was therefore chosen to use a 100 nF ceramic capacitor.
The initial charge current to the capacitor needs to be limited in order not to exceed the maximum ratings of the diode. Therefore a resistor is put in series with the diode. The resistor value should not be too big as the voltage drop across it is subtracted from the bootstrap capacitor voltage. A 4.7 Ω resistor was chosen. In this setting, the maximum current through this component is 2.6 A. This is clearly above the maximum average forward current of the diode at 1A. The diode can however, withstand an 8.3 ms single half sine-wave of maximum 30 A [50]. The capacitor voltage while charging in a RC circuit is described by equation 4.8, according to [38].

$$V_C = V_S(1 - e^{\frac{-t}{R \cdot C}}) \tag{4.8}$$

Where V_C is the capacitor voltage, V_S is the supply voltage, R is the resistance and C is the capacitance. The RC time constant is defined as $\tau = R \cdot C$. Equation 4.8 can be used to calculate the time it takes an RC circuit to charge to a specific level. After one time constant the capacitor is charged 63.2% and after five time constant the capacitor is approximately fully charged, 99.3%. This knowledge can be used to calculate the time it takes to initially charge the bootstrap capacitors as shown in equation 4.9.

$$T_{charge,init} = 5 \cdot \tau = 5 \cdot (R \cdot C) = 5 \cdot (4.7 \cdot 100 \cdot 10^{-9}) = 2.35 \mu S \tag{4.9}$$

Meaning that the diode can be exposed to a maximum of 2.6A for $2.35 \mu S$, which is well below the absolute maximum rating. It should also be noted that the bootstrap capacitor needs at least $2.35 \mu S$ for the initial charging.

Lastly it should be calculated what the maximum allowed duty cycle for the high-side MOSFETS are in this configuration, using the found components. The time it takes to charge the capacitor the energy that is used in each period is:

$$T_{charge,period} = 5 \cdot (R \cdot C_{boot}) = 5 \cdot (4.7 \cdot 51.6 \cdot 10^{-9}) = 1.21\mu S \tag{4.10}$$

The $1.21\mu S$ is equivalent to 2.7% duty cycle and the maximum theoretical duty cycle for the high-side MOSFETS is thus 97.3%.

4.3.3 Supply Capacitors

Driving an inductive load, such as a motor, requires capacitors placed close to the MOSFETS in order to minimize the voltage spikes created by the load current and parasitic inductance in the wires from the supply. This section will determine the size and requirements of the supply capacitors using calculations and simulation in PLECS.
First off, the ripple current across the motor needs to be determined as this is the current the supply capacitors need to supply. Equation 4.11 shows the general inductor equation and in 4.12 it is refactored to calculate the ripple current.

$$V(t) = L \cdot \frac{dI(t)}{dt} \tag{4.11}$$

$$\Delta I = \frac{\Delta V}{L} \Delta t \tag{4.12}$$

The highest ripple current is found when the duty cycle is 50% and the full voltage of 24V is applied as calculated in equation 4.13.

$$\Delta I = \frac{24}{82 \cdot 10^{-6}} \cdot 0.5 \cdot \frac{1}{22000} = 6.7[A] \tag{4.13}$$

The peak-to-peak ripple current of 6.7A is also found when simulating the system using PLECS as shown in figure 4.4. The load current experienced by the supply is also shown in figure 4.4, where abrupt changes in current are seen.
This is clearly problematic and even if the supply wires have just a few nH of parasitic inductance it would correspond to a huge voltage ripple which is unwanted. Therefore supply capacitors needs to be added to the circuit as close to the switching MOSFETs as possible. The minimum capacitance

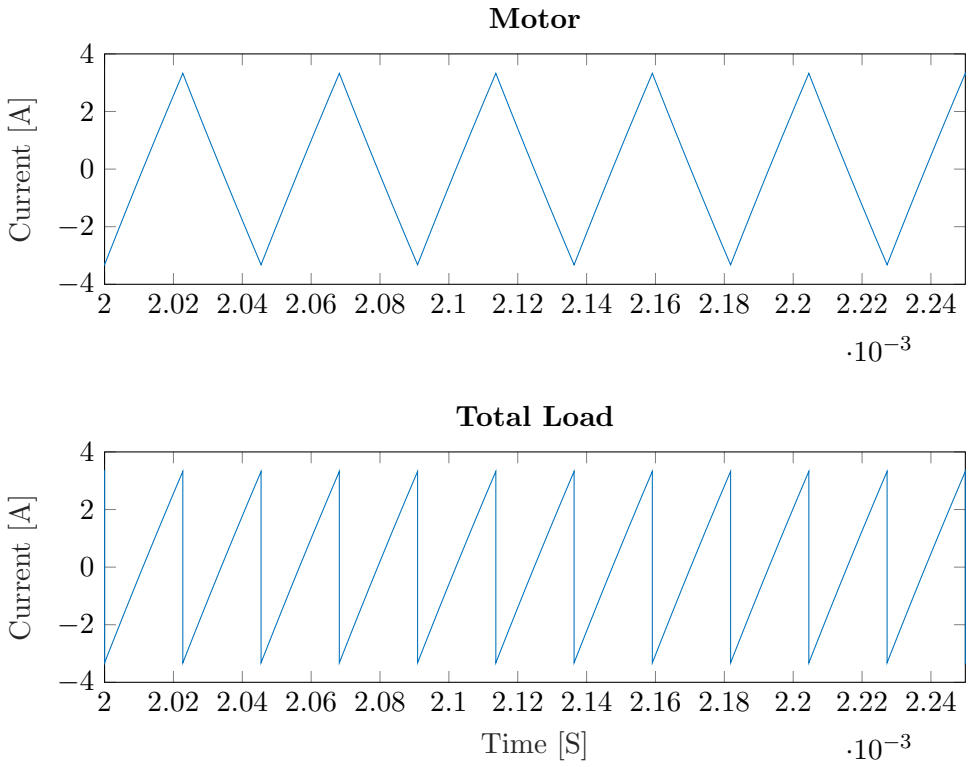


Figure 4.4: Current in the motor and the total load when when simulating a full bridge in PLECS.

can be calculated by refactoring the general capacitor equation shown in 4.14 into equation 4.15.

$$I(t) = C \cdot \frac{dV(t)}{dt} \tag{4.14}$$

$$C = \Delta I \cdot \frac{\Delta t}{\Delta V} \tag{4.15}$$

There is no specific downside to voltage ripple, short of the ripple being so great that the 12V rail is influenced by the ripple. Regardless the maximum allowable voltage ripple is set to 1V. Using the found ripple current and a maximum voltage ripple of 1 V, the minimum value can be found by equation 4.16.

$$C = 6.7 \cdot \frac{0.5 \cdot \frac{1}{22000}}{1} = 151[\mu F] \tag{4.16}$$

Parameter	Value
U_r	100 [V]
C_r	330 [μF]
I_r RMS @ 100[kHz]	1.4 [A]
I_r RMS @ 10[kHz]	1.3 [A]

Table 4.4: Relevant parameters of the `MAL215099911E3` capacitor. Here U_r is the rated voltage, C_r is the rated capacitance and I_r is the ripple current.

It can now be decided which capacitors should be used. It was chosen to use electrolytic capacitors as they are generally cheaper in this range of capacitance. Another reason to choose electrolytic capacitors is their equivalent series resistance, ESR, that will dampen the oscillations introduced by the LC circuit of the capacitors and the parasitic inductance of the wires. When inspecting capacitor datasheets the ripple current is given in RMS values. The RMS value of the load current is calculated to be 1.9A. It was chosen to use the `MAL215099911E3` [3] capacitors and connecting them in parallel. The relevant parameters of this capacitor are shown in table 4.4. Connecting two of these capacitors in parallel would yield a total allowed RMS ripple current of 2.6A, which would be sufficient for the 1.9A calculated RMS load current. To allow for some headroom it was chosen to use four, which yields a combined maximum RMS ripple current of 5.3A. Clearly the total capacitance of 1320μF is far above the minimum needed of 151μF, but this will only further decrease the voltage ripple.
The disadvantage of the electrolytic capacitors is that they are not usable for high frequency content. Therefore it was chosen to add ceramic capacitors as they perform much better at high frequencies. Calculating the needed size of these is very complicated as the high frequency content is determined by the switching speed of the MOSFETs, parasitics and the circuit board layout in general. Based on previous experience it was decided, to use four 1μF ceramic capacitors in the 0805 housing as they can easily be changed if it turns out that more capacitance is needed.
In order to be able to simulate the system, an estimate of the inductance of the supply wires is needed. According to [8] the inductance of a wire can be estimated using equations 4.17 to 4.19.

$$L_p = \frac{\mu_0}{2\pi} \cdot L \cdot \left(ln(\frac{2L}{r} - 1)\right) \tag{4.17}$$

$$L_i = \frac{\mu_0}{8\pi} \cdot L \tag{4.18}$$

$$L_w = L_p + L_i \tag{4.19}$$

Where μ_0 is permeability of free air, L is the length of the wire, r is the radius of the wire copper, L_p is the external self partial inductance, L_i is the internal partial inductance of the wire and L_w is the total self partial inductance. The wire length and radius is not known at this point, but it is assumed that it will be approximately 0.5m long and with a radius of 2mm. Inserting the appropriate values yields a wire inductance of 575nH.

The system was built in PLECS in order to simulate the currents and voltages and verify the designed system. The PLECS model is shown in figure 4.5 and can be found in the associated git repository.

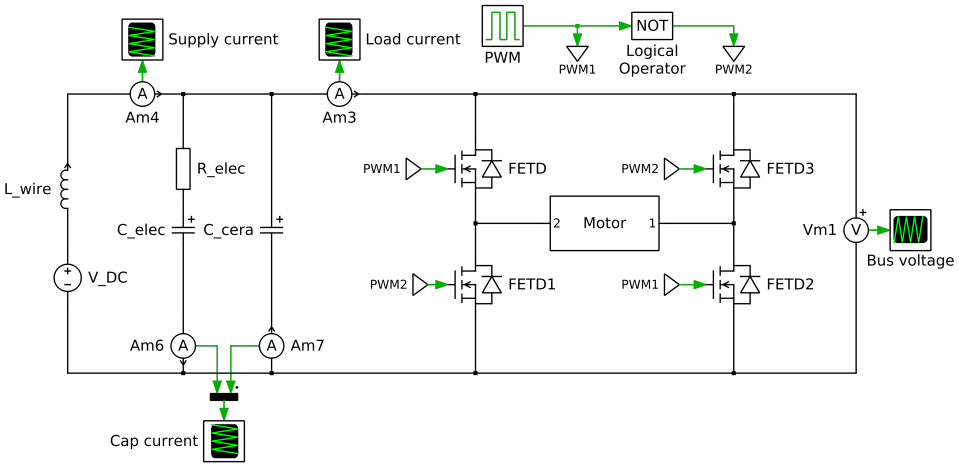


Figure 4.5: Full bridge circuit with supply capacitors and wire inductance modelled using PLECS. The Motor box consists of an inductor and a resistor in series. All components apart from the MOSFETS have the same values as the chosen components.

Simulating the system at 50% duty cycle yields a voltage ripple of approximately 24mV, which is definitely acceptable. The supply current and load current are shown in figure 4.6.
It can seen that an effect of the inductance in the supply wires and the supply capacitors is that the supply current is low pass filtered and is thereby protected against the ripple current and only needs to supply the DC component of the load current.
The current drawn from the capacitors is shown in figure 4.7. Here it can be seen that the electrolytic capacitors supply the majority of the current, while the ceramic capacitors supply a small spike with high frequency content. It should be noted that in the simulations there are no difference between the

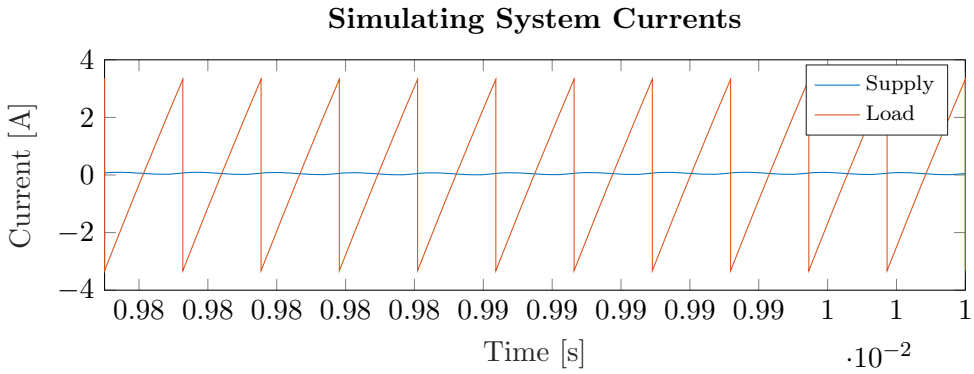


Figure 4.6: Current drawn from the power supply and the load current, when simulating in PLECS.

two types of capacitors other than the ESR put in series with the electrolytic capacitors.

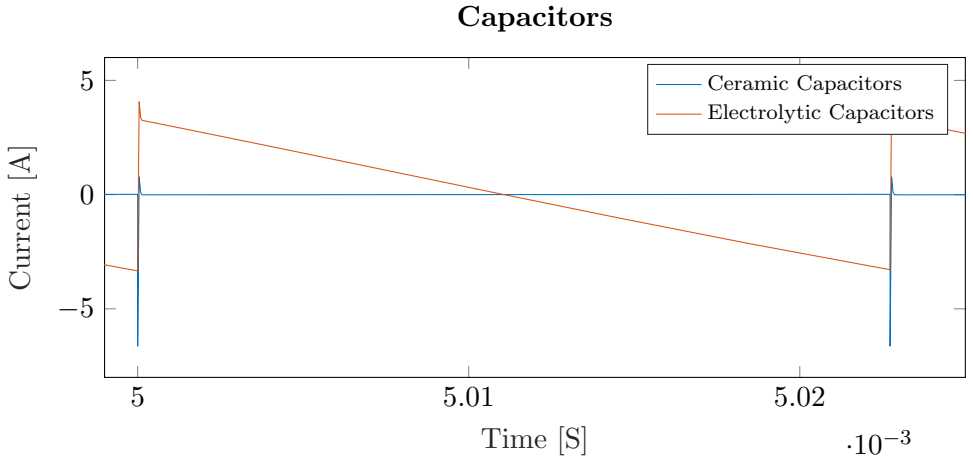


Figure 4.7: Current drawn from the electrolytic and ceramic capacitors, when simulating in PLECS.

4.3.4 Relay Circuitry

Two relays are necessary for the system, the main relay and an inrush relay. The main relay should be able to carry and break the total stall current, 80A. This requirement is met by the `LEV100A4ANG` [28]. It is capable of carrying up to 100A continuous current and can break significantly more. It has a

12V coil which requires ≈0.5A to hold in the closed state. As mentioned previously the inrush relay is necessary to facilitate the initial charging of the supply capacitors. The current to the capacitors is limited only by their ESR and a resistor should be added to limit this inrush current. Adding the resistor to the main current path would introduce considerable losses and limit the performance of the system. The inrush relay is implemented in parallel with the main relay to supply a temporary alternative current path in which the resistor is placed. The size of the resistor is somewhat irrelevant so long as it is large enough to protect the circuitry. Having a larger resistance will only increase the time it takes before the capacitors are sufficiently charged. For this reason a 180Ω 5W resistor is used. As shown by equation 4.9, the charging time can be found as

$$T_{charge} = 5 \cdot R_{inrush} \cdot C_{supply} = 5 \cdot 180 \cdot 1320 \cdot 10^{-6} = 1.188[\text{S}] \qquad (4.20)$$

Therefore, on bootup, the system will not be able to start running the motors until ≈1.2 S have passed. This should be taken into account in programming of the system. The relay chosen for this application is the `G6B-1114P` [13] with a 12V coil. Being a significantly smaller relay, carrying up to 5A, the `G6B-1114P` requires only 16.7mA to hold in the closed state.

Figure 4.8 shows the driving circuitry for the inrush relay. The circuitry is identical for the main relay and can be seen in the full schematic in appendix C. As can be seen, the relay can be switched `on` and `off` by a signal to the MOSFET Q18. This signal is `high` only when signals `INRUSH` and `EM_DIS` are `high`. `EM_DIS` is the signal from the safety circuitry and `INRUSH` allows the MicroZed to toggle the relay. Designing the circuit in this manner ensures that the motor will remain disengaged in case of an unmet safety condition. Finally, A flyback diode is placed in parallel with the inductor in the relay to create an alternative current path when the MOSFET is switched `off`.

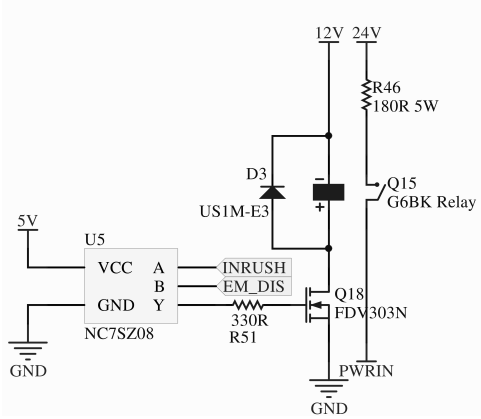


Figure 4.8: Circuitry related to the inrush relay.

4.3.5 Motor Current Sensing

Measuring current in an H-Bridge with a shunt resistor can be done in a number of ways. The shunt resistor can be placed in various positions in the H-bridge. The three main ones are high-side, in-line and low-side placement of the shunt resistor. [33] and [35] specifies the advantages and disadvantages of the three configurations.

High-side: Placing the shunt resistor in a high-side configuring gives the possibility to detect if the power supply is accidentally shorted anywhere in the H-bridge. The downside is a high common mode voltage.

In-line: This configuring allows for bi-directional current sensing and direct measurement of the motor current, but has large¡¡¡¡ voltage changes of the common mode voltage.

Low-side: This configuration has a low common mode voltage, the output voltage is referenced to ground and has a single-supply design. A downside is that the load is lifted slightly from ground.

It was decided to use the low-side configuration as it has low common mode voltage, it only needs a single sided supply and because lifting the load from ground has no effect in this application.

Current Measuring Circuit

The output of the current measuring circuit is an analog signal that is measured by the ADC of the Zynq chip. According to the Zynq ADC user guide [2], the measurable input range is from 0V to 1V, but the maximum allowed input voltage is 1.9V according to [53]. The absolute maximum current through the motor is the stall current which is 80A, but the motor will run well below this level in normal operation. The nominal current of the motor is 6A. Maximum current spikes during normal operation are not expected to exceed 30A. It was therefore decided that the circuit needs to be able to measure currents of at least up to 40A. In case of stall current through the H-bridge, the ADC input should not be harmed meaning that the input voltage should not exceed 1.9V.

Placing a shunt resistor in the low-side configuration raises the voltage potential of the low-side of the H-bridge. This may influence the gate signal required to open the MOSFETs of the H-bridge. In addition, the shunt resistor will, regardless of placement result in a powerloss proportional to its size. For these reasons minimizing the value of the shunt resistor is important. The smallest value available from the electronics supplier used at SDU is a $200\mu\Omega$ SMD shunt resistor. Such a small resistance yields a minimal voltage drop and thus keeps power consumption at a minimum.
The resulting voltage from the motor stalling, that is an 80A current flowing through the resistor would result in a 16mV voltage drop. Since the measurable input range of the ADC is 0-1V, this low voltage would yield very poor resolution in the area of actual interest, 0-40A. An amplifier circuit is necessary. Purpose built current-sense amplifiers exist which are used specifically for amplifying the voltage drop across a shunt resistor. One such amplifier is the `INA286AID` from Texas Instruments [19]. It has a gain of 100, lifting the voltage resulting from motor stall to 1.6V as seen from equations 4.21 and 4.22.

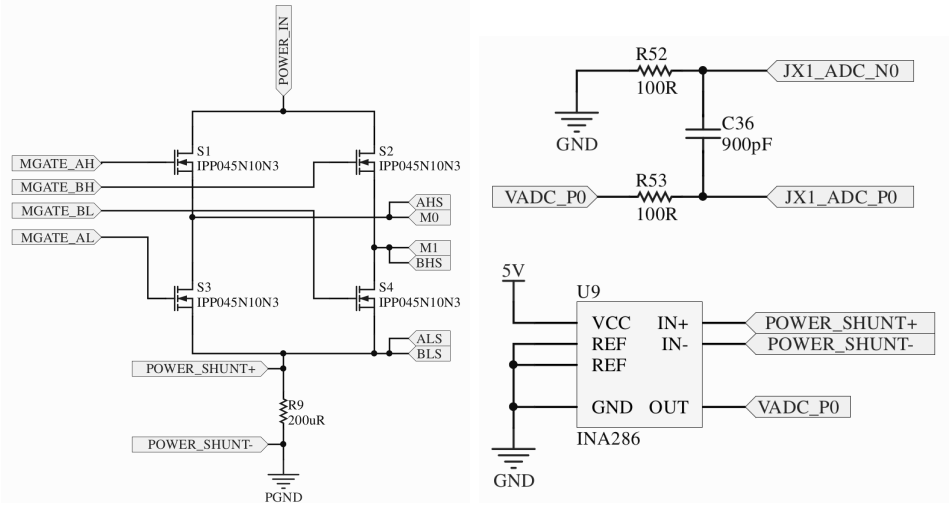
$$V_{max,input} = G \cdot V_{shunt} \tag{4.21}$$

$$V_{max,input} = G \cdot R_{shunt} \cdot I_{stall} = 100 \cdot 200 \cdot 10^{-6} \cdot 80 = 1.6V \tag{4.22}$$

This is below the maximum allowed input voltage of 1.9V, meaning that the input will not be harmed from stalling the motor. The measurable range using this IC is 0-50A as seen from 4.23.

$$I_{max,measurable} = \frac{V_{max,measurable}}{R_{shunt} \cdot G} = \frac{1}{100 \cdot 200 \cdot 10^{-6}} = 50A \tag{4.23}$$

It has a very low offset allowing it to be used in applications where the



(a) H-bridge with shunt resistor in low-side configuration.

(b) Schematic of the `INA286AID` as used in the project.

Figure 4.9: Schematics relating to the current measurement circuitry.

voltage drop across the shunt resistor is only a few mV.
Using this configuraion the resolution can be calculated as:

$$Res_{amp} = \frac{I_{max,measurable}}{Res_{ADC}} = \frac{50}{2^{12}} = 12.2mA \tag{4.24}$$

Where Res_{amp} is the resolution in amperes and Res_{ADC} is the resolution of the ADC on the Zynq.

This design meets the required design specifications and the schematics can be seen in figure 4.9. As can be seen, the `INA286` in this configuration requires only the voltage input, `POWER_SHUNT+/-` from the schematic on figure 4.9a, to generate a suitable output which is then fed directly to the ADC through the anti-aliasing filter shown on figure 4.9b.

4.3.6 Endstop Detection

As described in section 4.1.8 it was decided to use infrared transceivers to detect endstops. It was decided to use the `TCST2103` [48] as it has the required functionality. It contains an infrared emitter pointing at a photo transistor, that conducts current proportional to the infrared light received. In this project it will be used to detect if the infrared light is blocked by the

cart or not. The output of the TCST2103 is fed to a schmitt trigger circuit in order to be able to use the result as a boolean reference. The schmitt-trigger circuit is designed to have a high threshold of 3V and a low threshold of 1V. The circuit can be seen in appendix C.

The TCST2103 is mounted on the rail using a purpose-made 3D print. A rendering of the print can be seen in figure 4.10. The large hole in the bottom of the print is used as a cable lead-in for the rail. The cart will move parallel to the opening of the sensor. A pin is mounted underneath the cart which will block the IR transmitter and cause a safety condition.

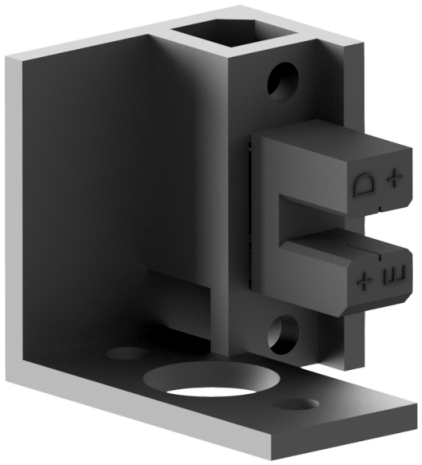


Figure 4.10: Fixture for mounting the TCST2103 to the rail.

4.3.7 Voltage Rails

In the design of the board it is necessary to determine which voltage rails are required for the system to function. The power delivery for the MicroZed was designed by the authors in an earlier project [25] and will be reused with minor changes. This power delivery system provides, amongst other voltages, the 3.3V rail, originally intended for powering the MicroZed IO banks only. In this system however, the RF tranceiver is also powered from this rail. As a result a review of the circuitry around the 3.3V rail is necessary to ensure that it can provide the required power. The MicroZed power delivery system, the encoders of the system, the endstops and part of the emergency circuitry are all driven from a 5V rail. The motor driver, the HIP4081 [18] and the relay driving circuitry is driven from a 12V rail.

Finally, the motor is driven from a 24V rail, which will also be the main supply for the system. This rail is not crucial to the design of the board as it is provided by an external, mains connected power supply and will not be explored further in this section.
The current requirement for each of these rails are discussed in the following paragraphs

3.3V: As mentioned, this rail is intended for powering the MicroZed IO banks. In the original design the `LMR10510XMF` DC/DC converter is used to provide the necessary power. This chip is capable of supplying up to 1A. Components exist in the same series which are pin compatible and are capable of supplying up to 2A. The RF Transceiver used in this project, however, requires only up to 15mA when receiving data and it is considered unnecessary to upgrade the current capacity of the rail.

5V: This rail supplies, mainly, the MicroZed. The maximum expected current draw from the MicroZed is 1.85A at 5V [5]. This includes the current drawn from the 3.3V rail. In addition, in this design, the 5V rail also powers the encoders, endstops and emergency circuitry. The `HEDS-5540` [17] encoder requires a maximum of 85mA. The endstops are realised using the `TCST2103` infrared sensor [48]. The majority of the current supplied to this sensor is used to power the infrared LED present in the component. This current is decided by the resistor put in series with the LED and is estimated to be around 50mA per LED for a total of 100mA. Another 10mA is added to that figure due to the collector current possible on the transistor side.
Finally the emergency circuitry along with some other digital electronics are powered from the 5V rail. As these are all digital IC's that operate on nothing but signals, their individual powers are negligible but a very conservative 25mA power budget is provided for all of the digital electronics on the 5V rail. This brings the total power budget for the 5V rail to ≈2.1A. See table 4.5 for a full overview.
In [25] the authors used a design in which the `PTH08080` is used to generate a 5V rail. Reusing this module will save on design time as well as the budget available to the project and as such is desirable. This module can supply 2.25A at 5V which is sufficient for this application and the design will be reused for this project.

12V: This rail powers the `HIP4081` motor driver, the relay coils, the bootstrap circuitry and the 5V DC/DC converter. The `PTH08080` DC/DC con-

Component	Current [mA]
MicroZed	1850
HEDS-5540	85
TCST2103	110
Digital	25
Total	2070

Table 4.5: Power budget for the 5V rail.

Component	Current [mA]
PTH08080	850
G6B	16.7
LEV100A4ANG	461
HIP4081	10
Total	1337.7

Table 4.6: Power budget for the 12V rail.

verter has a maximum input voltage of 18V and must be powered from the 12V rail rather than the 24V rail. The module can provide 2A at 5V and as such will require ≈0.85A from the 12V rail, assuming 100% efficiency.
There are two relays in the design, a smaller relay for controlling the inrush current and the larger main supply relay. Both of these require power to stay in the closed position. The first, the `G6B`, requires 16.7mA while the former, the `LEV100A4ANG`, requires 461mA.
The motor driver of the system, the `HIP4081` requires only 10mA. As mentioned, the bootstrap circuitry is also powered from the 12V rail. Rather than a steady supply, this circuitry requires a peak current for short periods while charging in between switching. For this reason the design choice here is to determine what is available and choose the component which yields the most headroom while still being economically feasible. Amongst all of the components the total power budget for the 12V rail is ≈1340mA. See table 4.6 for a full overview.
The `PTN78020` delivers 6A at 12V and is part of the same series as the `PTH08080` described earlier. This allows many of the same design procedures to be reused. In addition, the 6A current limit leaves sufficient room for the current spikes expected from the bootstrap circuitry.

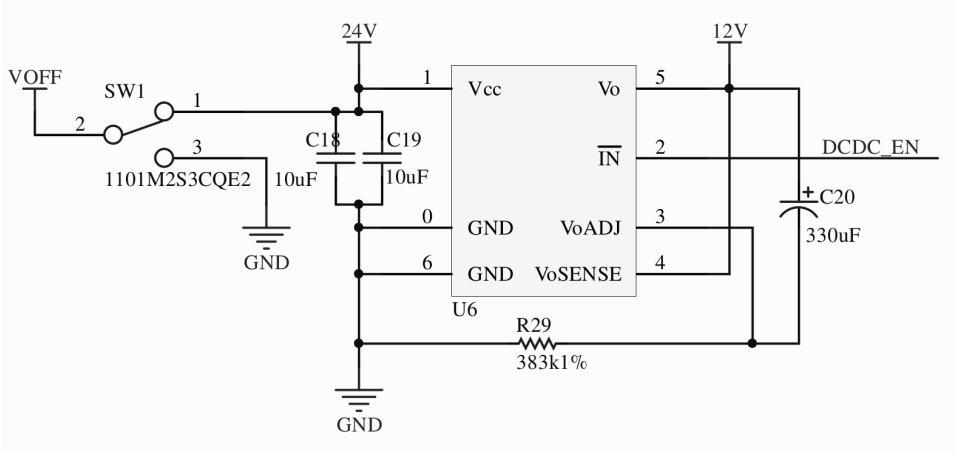


Figure 4.11: The PTN78020H DC/DC converter and the appertaining circuitry.

4.3.8 12V Rail Circuitry

The PTN78020H DC/DC converter is used to generate the 12V rail. The component requires input and output capacitors in order to function properly and a resistor is needed to set the desired output voltage. The circuit used is based on the typical application circuit shown in the datasheet [1] and is implemented as shown in figure 4.11. A resistor is required from VoADJ to ground to set the desired output voltage, in this case 12V. To generate 12V a 383 kΩ resistor is needed. VoSENSE should be connected close to the main load to allow for more precise regulation of the output. This feature is mainly useful when the load is placed far from the supply and can be left floating if the accuracy of the rail is not important. It was decided to simply short it to the 12V rail at the component since the load is placed relatively close to the supply. $\overline{\text{IN}}$ carries the inhibit signal. Pulling this low will disable the output of the component. This is done by toggling the switch SW1. This will open a MOSFET which in turn will pull the DCDC_EN signal low. The datasheet also specifies that a ceramic capacitance of at least 18.8 μF is needed at the input. It was chosen to use two 10 μF capacitors. In order to ensure stability the PTN78020H needs an output capacitance of 330 μF. The datasheet specifies that if the application has load transients additional capacitance should be added to improve the response of the regulator. While the load at the 12V rail is mostly steady, it does supply the bootstrap capacitors which create a slight ripple on the rail. It was therefore decided to use two 330 μF electrolytic capacitors in parallel. As suggested by the

datasheet an additional 20 μF of ceramic capacitance is added to the output to further improve the transient response.

4.3.9 5V, 3.3V and 2.5V Rail Circuitry

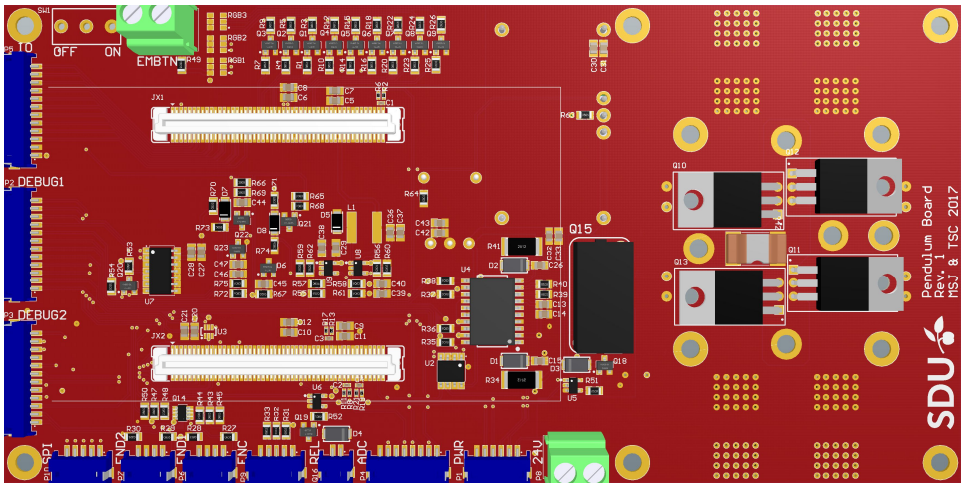
As mentioned previously the choice of these component and the design of the circuitry around them was done by the authors in a previous project [25] and the details of the design will be omitted for this discussion. The design can be seen in appendix C. The `PTH08080W` and `LMR10510XMF` are responsible for creating the 5V and 3.3V rails, respectively and the 2.5V reference is created by the `TL431AIDBZT`.

4.4 Board Layout

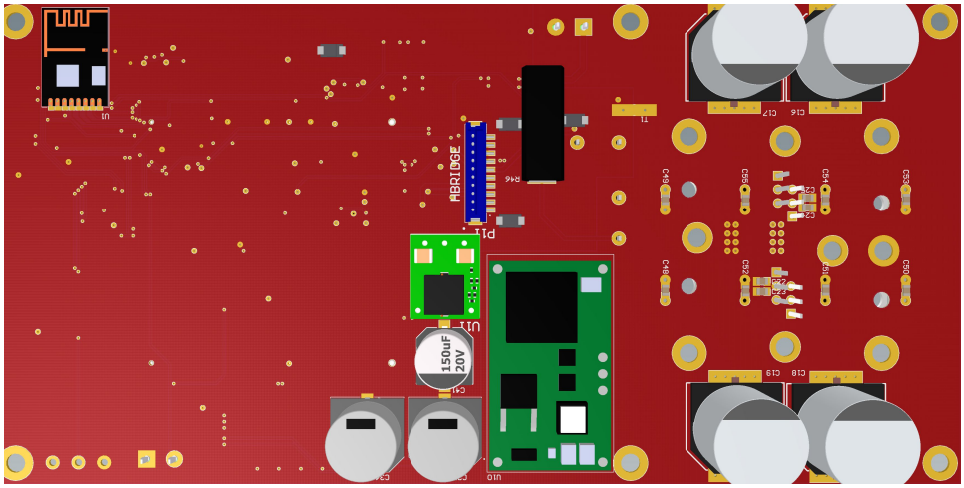
The layout of the board involved a number of considerations, especially concerning the layout of the power part of the circuit. These considerations involve the layer stack, the required copper thickness and various placements of components. This section will discuss in appropriate detail these considerations. Figure 4.12 is an overview of the final layout of board. Firstly, it is desirable to determine what requirements exist for the trace widths and thicknesses in the layout.

4.4.1 Copper Thickness

Part of the board is designed to house the H-Bridge that drives the motor. The H-bridge is designed to carry up to 80A, well above what can reasonably be carried at standard PCB copper thickness. This section seeks to estimate the required copper thickness to allow for the stall current. Allowing a current through a conductor will produce a temperature rise due to the power dissipated in the conductor. The current-carrying capacity (CCC) of a conductor is unlimited, so far as the current is not destructive due to the temperature rise permissable in the design. Essentially, the acceptable temperature rise is limited by the maximum operating temperature of the PCB material. This specification is known as the temperature index. The PCB material most commonly used is FR-4 which has a temperature index of 130°C [12]. At an ambient temperature of 20°C this would imply a permissable conductor temperature rise of 110°C . In all practical situations, this is unreasonably high. The material will discolor and degrade over time with the excessive heating and cooling of the board while also increasing the apparent ambient temperature of other components on the PCB, lowering



(a) Top side PCB view of controller board.



(b) Bottom side PCB view of controller board.

Figure 4.12: Top and bottom view of the controller board.

the lifespan of the circuit.

In actual designs, a much more conservative temperature rise of 10°C is often used, 15°C is acceptable and 20°C is considered high. This was discussed with Jesper Nielsen, a technician at SDU who has experience in designing high power PCBs and it was decided to accept 15°C increase. The actual temperature rise of a conductor is dependent on a great number of variables, among these are: conductor length, cross-sectional area, conductor material, PCB material, ambient temperature, convection, PCB layout (the amount of copper in the vicinity), etc.

Not all conductors carry the same current so different conductors have different thermal impacts on different locations on the PCB. Clearly, it is difficult to accurately calculate the required copper thickness, however an estimate can be made. In IPC-2221A [23] data is presented to show the temperature rise resulting from allowing a current through a trace on a PCB on different trace widths. This data was first published in [24], a progress report from the American National Bureau of Standards dating back to 1956. Much has happened in PCB manufacturing since then. The original data does not take into account the possibility of multi-layered boards or many of the newer board materials. In 2009 the new IPC-2152 [22] was published and reevaluates the topic of current-carrying capacity. This standard, however, is unavailable due to its high cost and the original charts are used instead. A number of PCB trace width calculators exist online which extrapolate from the original data to give estimates of the required copper thickness and width. One such calculator [34] was used to determine the required copper thickness and width required in this project.

There are economic constraints in place for this project which requires the authors to be somewhat conservative with respect to how exotic a solution will be considered. Most manufacturers support increasing the thickness of copper on a specific layer, allowing well above the 35μm standard copper thickness. The manufacturer used in this project was chosen due to a comparatively low price on 4-layer boards with 70μm copper on outer layers and 52.5μm on internal layers. On the power part of the board this results in a total of 122.5μm of copper for the 24V rail and power ground.

In order to simplify the design process, a distinction is made between three types of traces. This distinction is based on the expected amount of current carried by that trace.

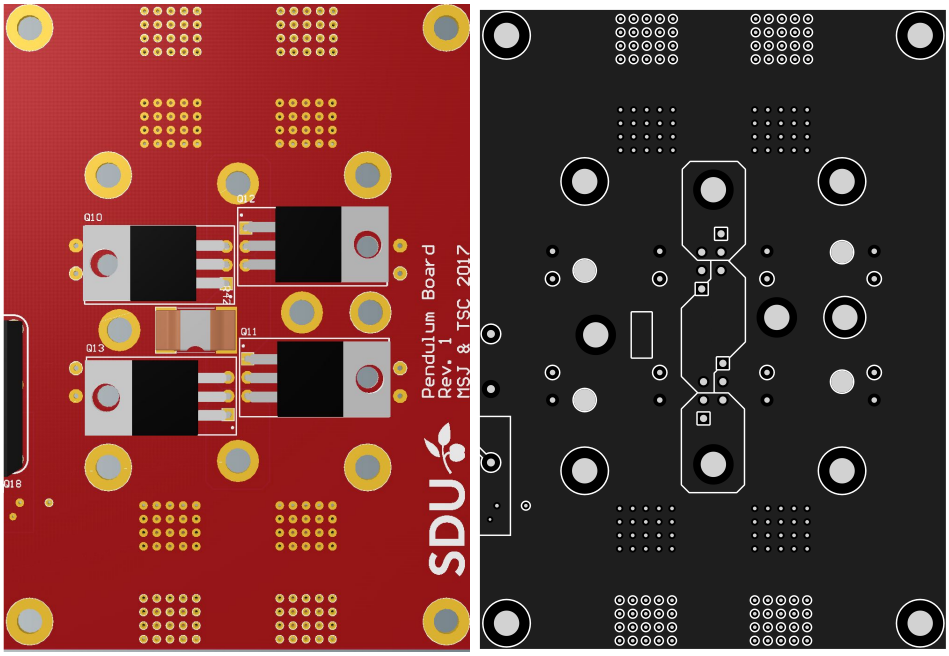
- **Signal:** This is the type with the lowest expected current and is used throughout digital circuitry and low power analog signals. Signal traces are by far the most abundant on the circuit and due to the very low currents the trace width should be as small as possible. The manufacturer is capable of creating traces at 8 mil or higher at the desired copper thickness and as such the trace width for this type is 8 mil. 1 mil is one thousandth of an inch and is approximately 0.0254mm.

- **Power:** This type is generally used for supply rails. The trace width for this type is 20mil. Accepting 15°C temperature increase will allow up to ≈1.5A of current. This is sufficient for supplying any of the IC's and other components connected to this trace type.

- **High Power:** This type is used exclusively for the 24V rail. The trace width for this type is 40mil. Accepting 15°C temperature increase will allow up to ≈2.4A of current. Since the inrush current is limited to ≈0.14A this leaves plenty of headroom for creating the 2.5V, 3.3V, 5V and 12V rails.

4.4.2 Copper Layout

The board can be split into two parts: a digital circuit and a power circuit. As seen on figure 4.12 the power circuitry is placed on the right third of the board while the digital circuitry occupies the remainder of the board. The two parts have seperate ground planes in order to protect the digital circuitry from the high and sudden currents generated by the motor. A ground plane, even though the resistance through it is low, will have a significant voltage drop across it if currents of sufficient magnitude are passed through it. For the same reason it is desirable to keep the current path as short as possible on the power side of the board. Figure 4.13 depicts a 3D view and 2D view of the power part of the board. As can be seen, the MOSFETs are bend such that a heatsink can be mounted on top. The upper pair is the left half-bridge and the lower pair is the right half-bridge. The outtake for the motor is positioned directly above and below the two half-bridges and a 3M bolt is mounted to allow for a robust connection. A large pad is made where the bolt is mounted as close as possible to the MOSFETs to allow for unrestricted flow of current. The pad between the motor outtakes is the connection from source of the low-side of the half-bridges to the shunt resistor used for current measurements. Since all current must pass through the shunt resistor, it is crucial that the flow of current is as unrestricted as possible.



(a) 3D layout of the power part on the top layer

(b) Copper layout of the power part on the top layer.

Figure 4.13: Power layout on the controller board.

Main power is input through M3 bolts mounted in the three holes between the half-bridges. Clearly, only two terminals are required but the three pads effectively divide the 24V rail into two parts, forcing any currents to flow around the pads. Instead it was decided to use two input terminals for the 24V rails that are mechanically fixed at the bolts. The ground plane is unrestricted and as such the current can flow freely and only one terminal is required. None of the planes on the power side of the PCB can be interrupted by traces, as this would severely hinder the flow of current in that plane. This means that it is necessary to add wires to carry the gate, return and shunt signals. These wires are soldered directly to the MOSFET pins and connected to the digital side through a header on the bottom side of the board.

As mentioned previously it was chosen to use a 4-layer board. This was done for two reasons: It gives more copper for the main power rail on the

power side of the board and it allows more space for routing signals on the digital part of the board. While the component density is not very high, the number of signals routed to headers does impose some difficulties routing on just two layers. On the digital side, a pour is made on each layer to fill the unused part of the layer with copper. Going from top to bottom the layers are as such: ground, ground, 5V, ground. The internal ground layer is uninterrupted, except for vias, to provide a solid ground layer. The 5V layer means that any components requiring 5V need only a via to access it, making routing simpler. This layer is also used for routing when routing through the top and bottom layers is infeasible.

4.5 Implementation

This section will go through the implementation of the various boards created in relation to the controller board. The prototyping boards and the experiences gained from them are presented. Hereafter the manufacturing process of the controller board is shown.

4.5.1 Prototyping

In order to test important parts of the controller board a home-made prototype PCB was made at the Universitys PCB lab.
Three main functionalities should be verified on the board:

- Schmitt-trigger circuitry and `AND` gate circuit.
- PWM logic level shifting.
- Motor driver circuit.

The remaining functionalities have either been tested previously or were too trivial to test.

Schmitt-trigger Circuitry and AND-Gate Circuit
The Schmitt-triggers were tested by varying the input voltage and verifying the switching threshold along with the the hysteresis levels. Testing the `AND`-gate was done by applying different logic levels and verifying the results. The complete circuit was found to work as intended and can be seen in appendix C, at component `U7` and `Q14`.

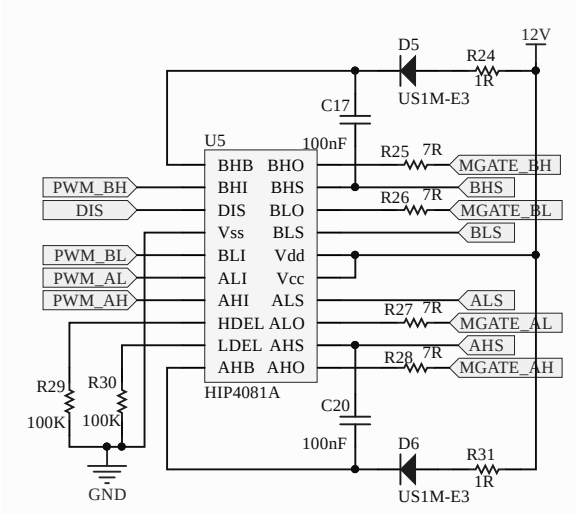


Figure 4.14: Schematics of the PCB developed to test the functionality of the motor drive circuitry.

PWM Logic-Level-Shifting

The functionality of the logic-level-shifter `74LVC2G17` was verified by measuring input and output on an oscilloscope.

Motor Driver Circuit

By measuring the outputs of the inverter `SN74LVC2G04` it was concluded that there was a design error in the way the component was used and the output signals were invalid. A thorough investigation of the schematic revealed that the component could be omitted without changing the functionality of the board. The motor driver circuit on the PCB could not be tested because of this design error. Therefore it was decided to develop another prototype PCB with the sole purpose of testing the motor driver circuit. The schematic of the circuit can be seen in 4.14 and a picture of the developed PCB in figure 4.15.

The boards gate signals were wired to the gates of four MOSFETs in a full-bridge configuration. A power resistor was used as a load and PWM signals were applied to the motor driver input pins. Measuring the gate signals showed strange behaviour and after a thorough inspection of the board and the schematic it was found that the bootstrap diodes were facing the wrong direction. The source of this was an error in the footprint of the component. After correcting the issue the driver and the bootstrap circuit was found to

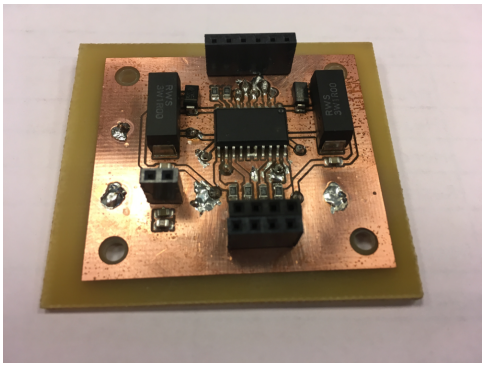


Figure 4.15: Picture of PCB developed to test the functionality of the motor drive circuitry.

function as intended. The measured signals are shown in figure 4.16.
It was verified that the low-side gate signal follows the waveform of low-side input signal, but with a logic level of ≈12V as expected. It was also verified that the high-side gate signals are 180° out of phase of the low-side signals and are correctly bootstrapped to ≈24V.

4.5.2 Controller Board

The controller board was designed by the authors but the PCB was manufactured using an out-of-house PCB fabricator. Adding components to the top of the board was done using the pick-and-place facilities available at SDU. A stencil had been ordered along with the design to allow for easy application of solder paste.

As mentioned previously, the bottom side of the board is populated mostly with larger components which are easily handsoldered. The extra copper on all four layers did, however, complicate the soldering process somewhat. Applying sufficient heat to the board becomes troublesome, especially with the lower-wattage soldering irons generally available to the authors. Eventually a heating plate was used to pre-heat the board, which made the process bearable.

A picture of the finished board can be seen in figure 4.17. It should be noted that while the authors took measures to avoid errors in the PCB, some amount of them found their way into the design regardless. Here is a list of the errors that have been found and the fix done to work around

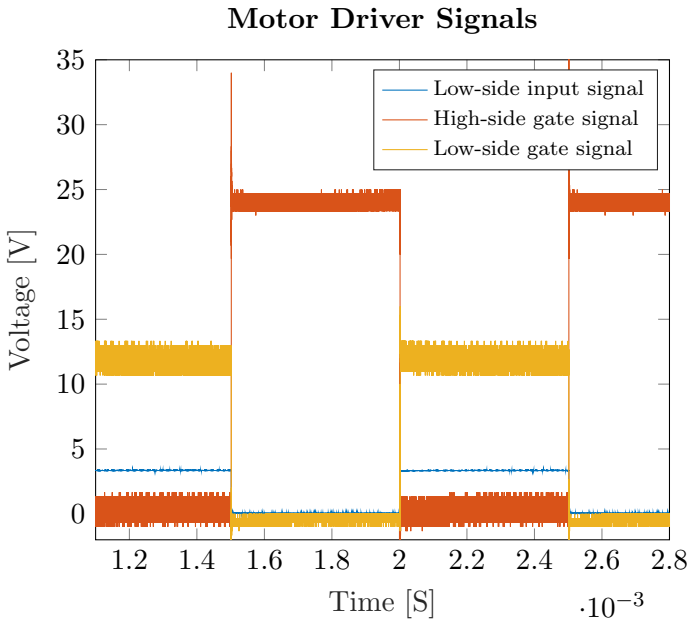


Figure 4.16: Measured low and high-side gate signals along with low-side motor driver input signals.

them. Component names are referenced to the controller board schematics found in appendix C.

- Missing ground on source of Q22: A wire is soldered from R69 (same net as source of Q22) to ground-side of R73.
- `EM_MCU` wired to wrong port: The original trace was cut and a wire is soldered from R54 to JX2 Pin14.
- Encoder signals not wired to header: Wires are soldered from `A` at R31, `B` at R32 and `Z` at R33 to JX2 Pin32, Pin36 and Pin38.
- `EM_END` signals not wired to header: Wire soldered from R44 to JX2 Pin30
- MOSFETs are reversed in footprint: MOSFETs are reversed in design to negate the reversed footprint. Standard TO220 heatsink used.
- Footprint of motor capacitors reversed: Capacitors mounted in reverse to negate the reversed footprint.

- Inhibit signals of `PTN78020` and `PTH08080` connected: Trace cut between the two so that `PTN78020` is still connected to the net.
- `nRF24L01` module, `nRFM` (see section 5.3.2) close to copper: `nRFM` mounting using wires to clear the copper of the board.

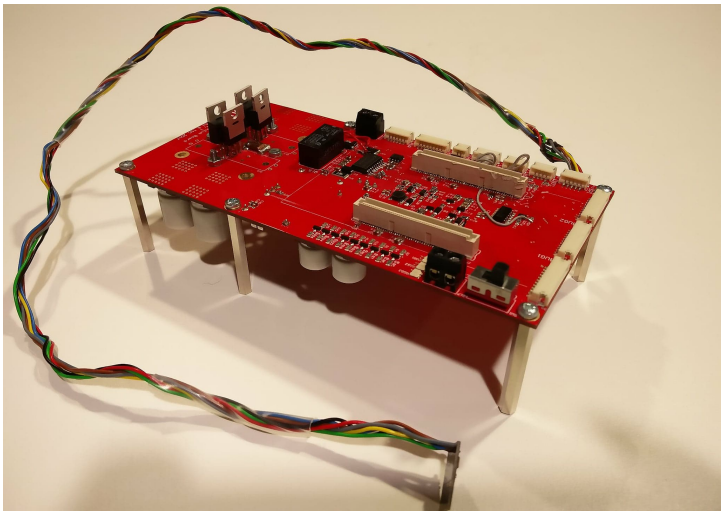


Figure 4.17: Picture of the implemented controller board.

4.6 Verification

This section will go through the verification of the manufactured controller board. Verification is split into two parts. The initial part verifies mainly that the board has been correctly manufactured, while the next part verifies the remaining design of the board.

4.6.1 General Verification Procedure

This procedure is created in an effort to allow for an easier debugging procedure of a newly produced controller board, should a developer require the production of a new board. A blank version appropriate for printing can be found in appendix B. It should be noted that the version seen below is not identical to the one found in the appendix. This is because some of the tests performed below are done to verify that the design is done correctly, rather than ensuring that production has been done correctly. ✓ is used to indicate the successful verification of a test while ✕ is used to indicate the

initial failure of a test. All pins, components and nets are typeset using the `teletype font` and are referenced to the schematics in appendix C.

Verify Voltage Rails

- Apply 24V to `24V`.
- Toggle power button to ON.
  - ✓ Verify that current draw is within reasonable limits (<150 mA).
    - The current draw was measured to be less than 100mA on average.
  - ✕ Verify 12V, 5V, 3.3V and 2.5V voltage rails.
    - None of the voltage rails were present. This lead to an investigation of `PTN78020`, which should produce the 12V rail. The `PTN78020`s inhibit pin is insufficiently floating. It has an internal pull-up resistor which should bring the pin voltage up to 1.5V when the signal is left floating externally, but the pin voltage was approximately 0V. Applying a 3V signal to the pin brings the 12V rail up, followed by the 5V and 3.3V rails. The issue was found to be that the inhibit pins of `PTN78020` and `PTH08080` are connected to the same net, `DCDC_EN`. The `PTN78020` is not able to correctly raise the voltage on its inhibit pin, because of `PTH08080`. Cutting the `DCDC_EN` track such that the two are effectively on separate nets fixed the issue. This results in not being able to disable the `PTH08080` manually, however this will happen in an acceptable manner by simply turning off the `PTN78020`.

Simulate No Emergency

- Apply 5V to `EM_END1` and `EM_END2`.
  - ✓ Verify that 0V is present on `EM_END1_OUT` and `EM_END2_OUT`.
- Apply 0V to `EM_END1` and `EM_END2`.
  - ✓ Verify that 5V is present on `EM_END1_OUT` and `EM_END2_OUT`.
- Set signals `EM_END1`, `EM_END2` and `EM_MCU` to 0V.
- Ensure `EM_BTN` is set to 5V.
  - ✓ Verify that 5V is present on `EM_DIS`

Simulate Emergency

- Set signals `EM_END1`, `EM_END2` and `EM_MCU` to 0V.
- Ensure `EM_BTN` is set to 5V.
- For each signal, toggle to either 5V or 0V.
  - ✓ Upon toggling one signal, verify that 0V is present on `EM_DIS`.

Verify Relay Circuitry Functionality

- Following the previous step, make `EM_DIS` high.
- Apply 0V to `INRUSH`.
  - ✓ Verify that 0V is present on the output of U5.
- Apply 5V to `INRUSH`.
  - ✓ Verify that 5V is present on the output of U5.
  - ✗ Verify that 24V is present on `POWER_IN`
    - Measuring the voltage on `POWER_IN` did not show the expected 24V, but approximately 6V. Investigation showed that the footprint of the four $330\mu F$ electrolytic capacitors was inversed. Desoldering the capacitors and soldering in the correct orientation yields the expected behaviour and the 24V can be measured at `POWER_IN`. The charging curve of the capacitors through the inrush current is shown in figure 4.18 and it can be seen that the initial charge of the capacitors takes approximately 1s. This is in accordance with the calculation done in section 4.3.4
- Apply 0V to `M_RELAY`.
  - ✓ Verify that 0V is present on the output of U6.
- Apply 5V to `M_RELAY`.
  - ✓ Verify that 5V is present on the output of U6.
- Apply 0V to `INRUSH`.
  - ✓ Ensure that 24V is still present on `POWER_IN`.

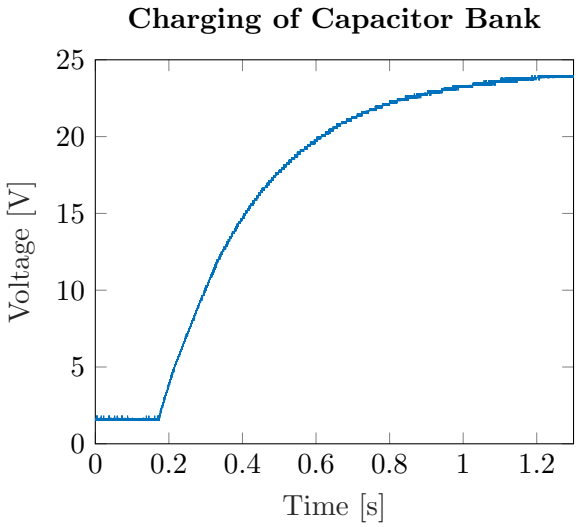


Figure 4.18: Voltage measured on `POWER_IN`, when the inrush relay is conducting. It can be seen that charging the capacitors takes approximately 1s.

Verify Motor Driver Functionality

- Apply 5V to `DIS`.
- Apply 3.3V PWM to signals `PWMA` and `PWMB`.
  - ✓ Verify corresponding 5V PWM on signals `PWM_AL` and `PWM_BL`.
  - ✓ Verify that 0V is present on signals `MGATE_BH`, `MGATE_BL`, `MGATE_AH` and `MGATE_AL`.
- Apply 0V to `DIS`.
- Ensure that 24V is still present on `POWER_IN`.
  - ✗ Verify that 12V PWM is present on signals `MGATE_AL` and `MGATE_BL`.
    - `MGATE_BL` works as intended. `MGATE_AL` follows the correct switching pattern but is at 4.8V low and 7V high.
  - ✗ Verify that 35V PWM is present on signals `MGATE_AH` and `MGATE_BH`.
    - `MGATE_BH` works as intended. `MGATE_AH` behaves similarly to `MGATE_AL`, indicating a short circuit between the two signals. Tracking down and removing the short circuit resulted in the expected behaviour on both channels. See figure 4.19 for a view of the signals on channel B.

✓ Verify that deadtime exists between the two channels.

- Deadtime exists on the gate signals as is shown in figure 4.20.

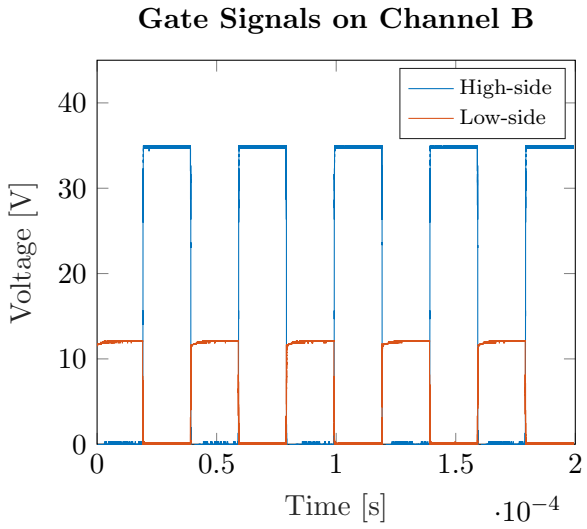


Figure 4.19: Gate signals as produced by the motor driver.

Verify nRF24L01 Module functionality

- Apply 3.3V to signals `RF_MISO`, `RF_MOSI`, `RF_SCK`, `RF_CSN` and `RF_CE` one at a time.

  ✓ Verify that 3.3V is present on the corresponding signals on the `nRFM` (component `U1`).

Verify Shunt Amplifier Functionality

- Apply 10mV to `POWER_SHUNT+`.

  ✕ Verify that 1V is present on `VADC_P0`.

  - When measuring `VADC_P0` only noise is present. The noise might be a product of the poor setup consisting of an extremely low voltage signal applied to a wire connected to the board through a header.

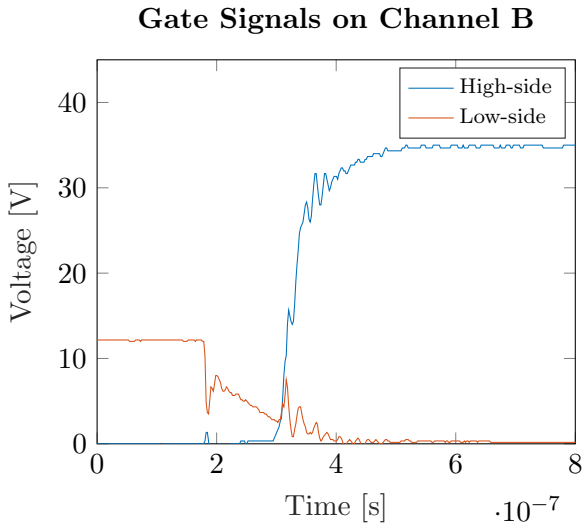


Figure 4.20: Gate signals as produced by the motor driver. It is verified that deadtime exists.

4.6.2 Verification of: Requirement 10

This requirement specifies that motor speed and direction should be adjustable through PWM output ports of the MicroZed.

Test
Verification of the requirement is done by applying a `low` level signal to `DIS` and `PWM_B` while applying a 3.3V PWM signal to the `PWM_A` pin. Gate signals should be measured and it should be verified that the motor is turning. Hereafter the dutycycle should be altered and the motor speed should change. Repeat for channel `B`.

Conclusion
The test was done on both channels and it was verified that direction and speed can be controlled through PWM signals on channel `A` and `B`. The gate signals for channel `B` are shown in figure 4.19, where it can be seen that the high-side gate signals are correctly bootstrapped.

Addendum
During further testing of the system the H-bridge suddenly gave up and released its smoke. This spurred an extensive quest to determine the root

cause of the issue. Eventually the data in figure 4.21 was found. This data suggests that a large current, > 20A is drawn when `DIS` is set to `low` and the output of the motor driver is enabled.

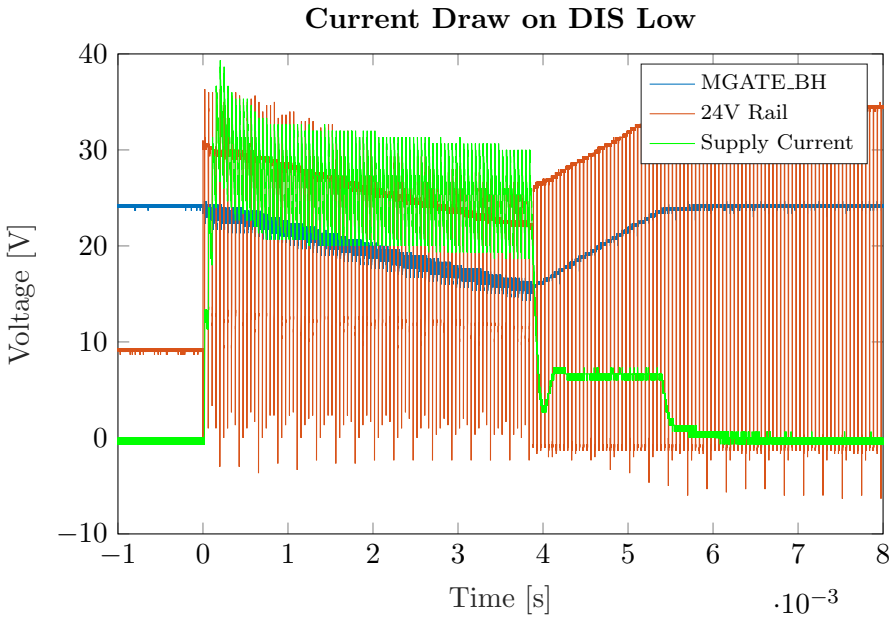


Figure 4.21: Current draw when setting `DIS` low and applying a 10% duty cycle PWM to the high-side MOSFET of channel B. Supply current is measured in amperes.

This current draw continues for approximately 4ms and then disappears. While the current draw is present the 24V rail falls which, eventually, will cause the 12V rail to fail, causing the 5V rail to fail and, finally, the MicroZed to power down. At low PWM duty cycle the 24V rail recovers before the failure happens.
By inspecting the data of the time immediately after `DIS` is set `low`, as seen in figure 4.22, it is apparent that the gate signal on the high-side MOSFET is never pulled below 12V and as a result the MOSFET is always conducting, creating shoot-through on the 24V rail when the low-side MOSFET is conducting. Due to this the current can be seen to step up at every pulse of PWM until at approximately 0.2ms where a steady state occurs. This steady state is maintained until approximately 4ms have passed where suddenly the gate signal does correctly go to 0V, meaning that the high-side MOSFET no longer conducts, allowing the 24V rail to recover.

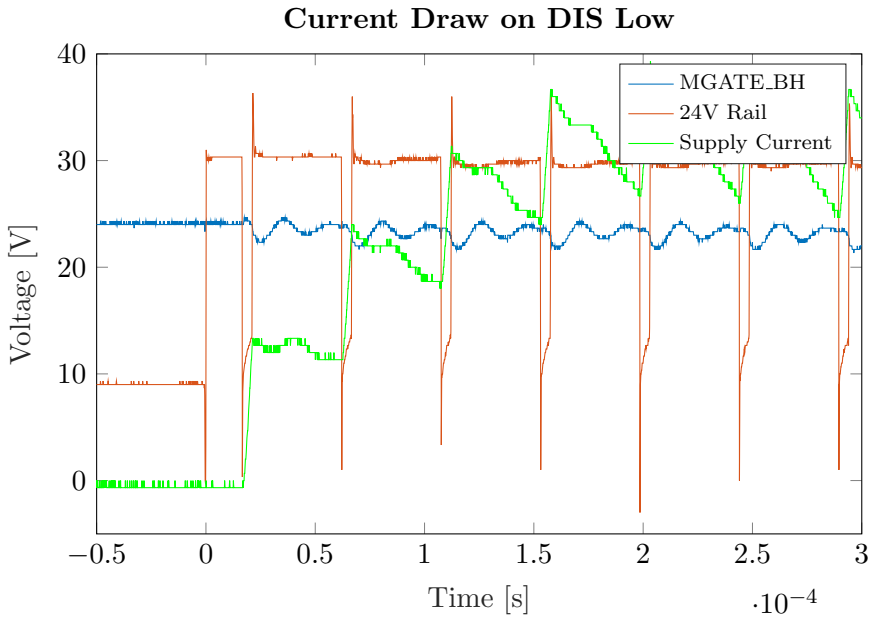


Figure 4.22: Current draw when setting `DIS` low and applying a 10% duty cycle PWM to the high-side MOSFET of channel B. Supply current is measured in amperes.

In an attempt to work around the issue the first 4ms after setting `DIS low`, were spent with 0% PWM, yielding the result seen in figure 4.23. This is the expected behaviour. PWM starts, slight current increase at the supply, steady 24V rail.

The issue was not detected during the initial testing because the digital circuitry and the H-bridge were powered using seperate power supplies. This was convenient at the time because it allowed testing parts of the circuit without risking shorting out the H-bridge or otherwise harming the circuit. Since the 12V and therefore the 5V rail were unaffected by the dipping 24V rail the problem did not cause the MicroZed to power down and the problem was therefore undetected.

4.6.3 Verification of: Requirement 11

This requirement specifies that it should be possible to measure the current through the motor using the ADCs on the MicroZed.

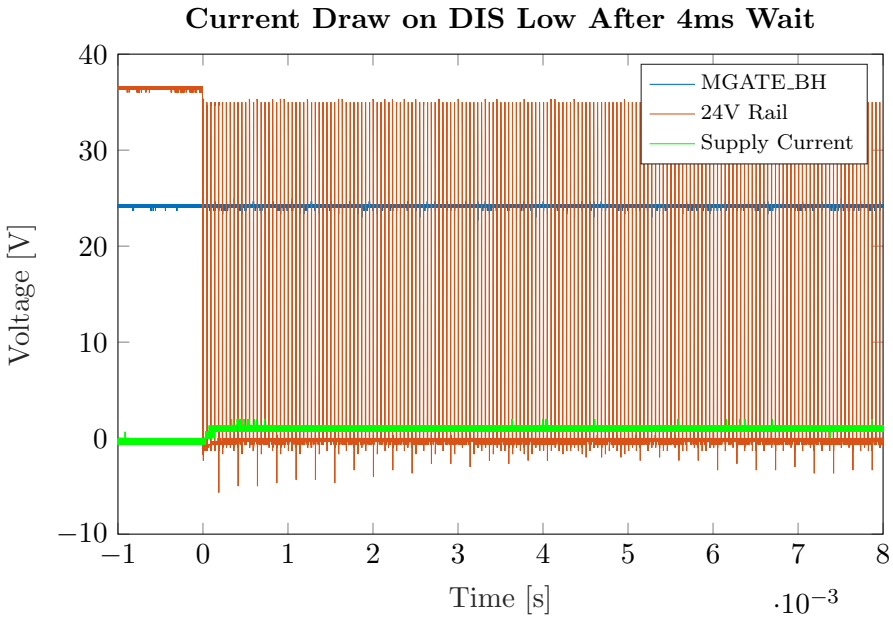


Figure 4.23: Current draw when setting `DIS` low and applying a 0% duty cycle PWM to the high-side MOSFET of channel B for 4ms before applying 10% PWM. Supply current is measured in amperes.

Test

Place a 3 Ω resistive load across the motor terminals and apply 24V to the board. Set `PWMA` to `LOW` and `PWMB` to `HIGH`. This will create a direct current path through the H-bridge and therefore through the 200$\mu\Omega$ shunt resistor. The resulting current will be 8A. This results in a voltage drop of 1.6μV which should result in a measurable 160mV on the output of the `INA286`

Conclusion

The output was measured and is shown in figure 4.24. As can be seen the output is approximately 300mV, twice as large as the expected output, 160mV. Furthermore the signal is very noisy with spikes as large as 200mV peak-to-peak.

Addendum

The data was gathered with NO switching of the MOSFETs. Further tests revealed that additional noise is introduced when the MOSFETs are actively switching. Spikes appeared in the data that do not in any way represent the

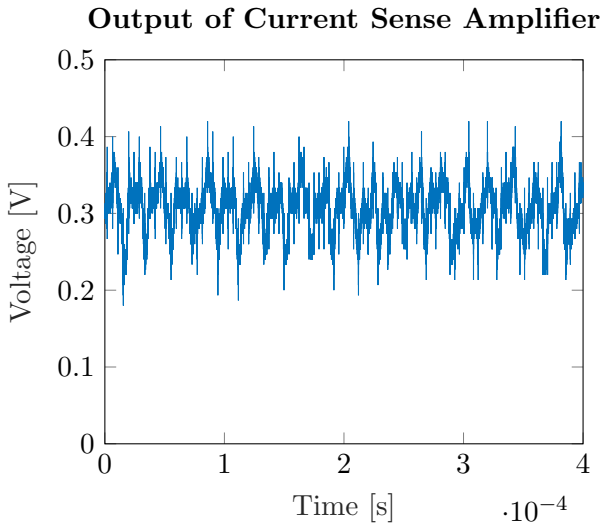


Figure 4.24: Measured output voltage of current sense amplifier.

current flowing through the load.

During design the necessity of an analog ground was overlooked, despite the obvious need for one when handling signals as delicate as are seen here. Unfortunately it is unlikely that this feature can be used without a redesign of the board. This redesign would include the addition of an analog ground and possibly an attempt at increasing the signal voltages. While this would cause more power loss due to the higher resistance to ground it would also make the signals more robust and therefore easier to measure correctly.

Lacking the current measuring feature is unfortunate but only excludes the possibility of current mode control.

4.6.4 Verification of: Requirement 12

This requirement states that five voltage rails should be present on the controller board.

Conclusion
In the verification procedure in section 6.6.1 it is verified that all rails are present. The voltages of the rails were measured for 20ms and the mean voltage of each rail is shown in table 4.7. All measured values are within

2.483V	3.305V	4.987V	12.00V

Table 4.7: Measured mean voltages on the 2.5V, 3.3V, 5V and 12V rails respectively.

the values specified in the requirement specification.

4.6.5 Verification of: Requirement 13

This requirement states that voltage transients on voltage rails should be minimized.

Test
The voltage across the H-bridge should be measured while driving the motor in order to detect voltage transients.

Conclusion
While conducting the test as described above it was not possible to detect any voltage transient due to the current draw by the motor. This is as expected, because of the capacitor bank. The motor current and the current draw from the main power supply was measured while driving the motor. The measured waveforms are shown in figure 4.25. Here it can be seen that even though the motor current is alternating the supply current is mostly DC.

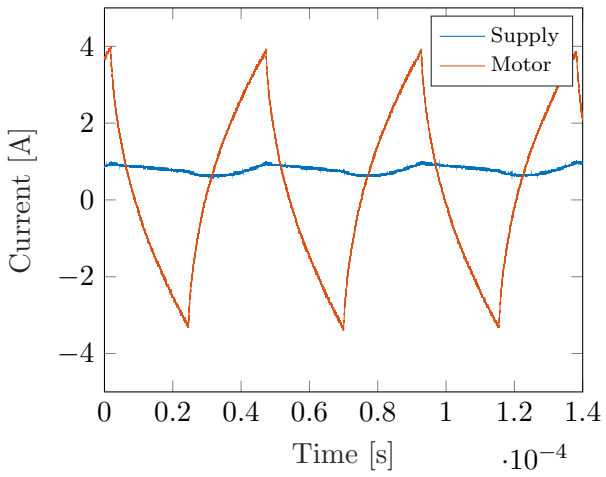


Figure 4.25: Measured supply and load currents while driving the motor using 50% duty cycle on one channel and 0% on the other.

4.6.6 Verification of: Requirement 14

According to requirement 14, the MicroZed should be interfaced correctly. This includes start-up and shut-down sequences.

Test

Testing if the board complies with the correct start-up and shut-down sequence should be done by measuring the appropriate signals when turning the board `on` and `off`.

Conclusion All appropriate signals during startup where measured and are plotted in figure 4.26.

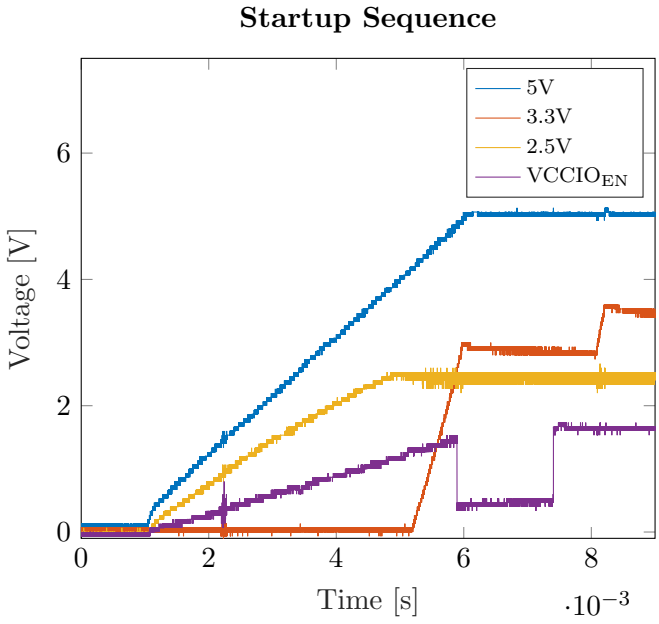


Figure 4.26: Signals measured doing startup of controller board with the MicroZed inserted. The 3.3V rail does only rise in voltage, when `VCCIO_EN` has a higher voltage level than 1.3V.

It should be noted that the `VCCIO_EN` signal is controlled by the MicroZed and not the developed controller board. It isverified that the voltage on the 3.3V rail does not rise when the voltage of `VCCIO_EN` is lower than 1.3V. This level is the high threshold of the comparator that enables the DCDC

converter that generates the 3.3V rail. Furthermore, using the work done in [25], it is verified that the measured signals correspond to the ones produced when powering `on` the MicroZed using Avnets own MicroZed carrier card.

The appropriate signals when shutting down the controller board and the MicroZed are measured and plotted in figure 4.27. It is verified that `VCCIO_EN`, `POWER_EN` and `DCDC_EN` are set to `GND` in the correct order.

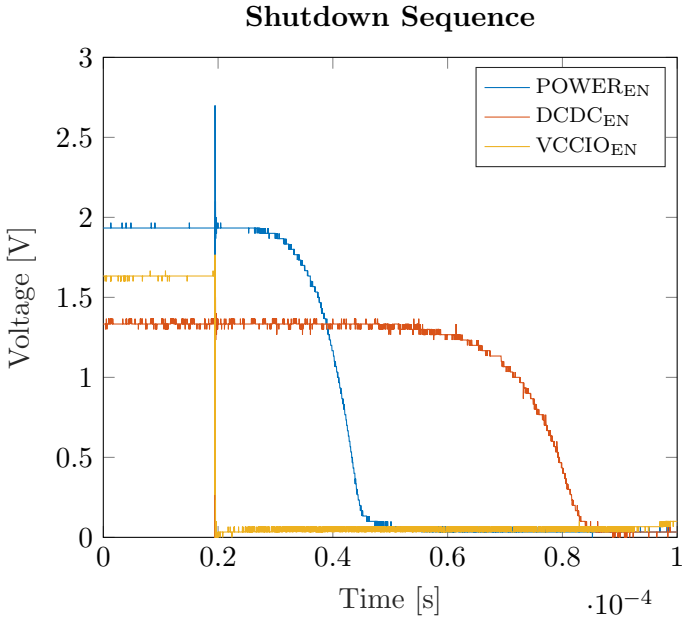


Figure 4.27: Signals measured doing shutdown of controller board with the MicroZed inserted. It can be verified that `VCCIO_EN`, `POWER_EN` and `DCDC_EN` are set to `GND` in that order, which is as inteded.

4.6.7 Verification of: Requirement 15

Requirement 15 specifies that MOSFETs should be cooled to avoid overheating.

Test
The MOSFETs should run the motor for several minutes at a high velocity. During the test, the MOSFETs should not get hot.

Conclusion
During production of the board an error in the footprint of the MOSFETs was found, causing the solution with the large common heatsink to be unachievable. Therefore individual standard TO-220 heatsinks were connected to each MOSFET. The motor was driven by the MOSFETs for several minutes with a duty cycle of 95%, while not getting warm. It should be noted that this test does not necessarily stress the motor sufficiently to fully conclude sufficient cooling. Once the system is finished a test should be designed which moves the cart back and forth repeatedly to have more current flow through the MOSFETs.

4.6.8 Verification of: Requirement 16

This requirement specifies that hardware for wireless communication using RF should be implemented on the controller board.

Conclusion
The `nRFM` was chosen and placed on the board with an SPI connection to the MicroZed.

Part of the reasoning behind choosing an RF module was the expectation that it would forego the need of an extensive protocol. The chosen module, the `nRF24L01`, does however, utilize a rather extensive protocol and transmission of the angular position is 75% overhead, 25% payload.

4.6.9 Verification of: Requirement 17

This requirement specifies that important parts of the circuitry should be tested on prototype PCB.

Conclusion
Two prototype PCBs were created which lead to correction of two errors and verification of the rest of the functionality. Time and money was saved this way, by correcting the errors before ordering the full board from a professional PCB vendor at a higher cost and a long delivery time.

A prototype PCB should have been done to verify the functionality of the shunt resistor and current measurement circuit. While the layout certainly influences the functionality of this circuitry, it is believed that a prototype

would have revealed the necessity of an analog ground plane. This was not done because the necessity of it was not apparent at the time.

4.6.10 Verification of: Requirement 18

According to this requirement the PCB layout should be reviewed according to the five steps of the developed strategy.

Conclusion
Using this method a PCB layout was produced with significantly less errors than the previous experience of the authors. The fact that the final PCB contained errors indicates that the steps were not followed thoroughly enough or that the method is flawed. Two of the errors originated in a difference between a components footprint and the symbol in the schematic. Therefore it should be added to the footprint inspection step that it should be checked for all components if their footprint is in accordance with the symbol in the schematic. The remainder of the errors are oversights from step four. This indicates that this step should be elaborated further so that these oversights are not possible. One example is the signals that are wired to a connector and the debug header, but not to the MicroZed. Additionally it would have been beneficial to further make use of the peer-review step.

4.6.11 Verification of: Requirement 19

This requirement specifies that the PCB layout should be designed with debugging in mind.

Conclusion
This was done by adding six debugging headers with a total of 55 signals to the PCB design. The debugging headers were used extensively in the debugging process.

4.6.12 Verification of: Requirement 20

This requirement specifies that the PCB should be designed to accomodate appropriate power levels.

Conclusion
Three trace widths are used, 8 mil for signals, 20 mil for power and 40 mil

for high power. MOSFETs, full-bridge driver and the accompanying components were sized to accomodate 80A, the stall current of the motor.

A power and a digital ground plane was created on the PCB of the controller board. It was found during debugging that an additional analog ground plane is required to properly utilize the current measurement capabilities designed into the PCB. The debugging leading to this conclusion can be seen in section B.

4.6.13 Verification of: Requirement 21

According to this requirement all PCB soldering should be possible at SDUs facilities.

Conclusion
The PCB soldering was successfully done using SDUs facilities.

4.6.14 Verification of: Requirement 22

This requirement states that any unmet safety conditions should halt the cart.

Conclusion
Signals from emergency button, endstops and emergency signal from MicroZed are wired to an `AND`-gate, resulting in one signal that determines if a safety condition is unmet: `EM_DIS`.
The power delivery to the H-bridge is delivered by the main relay. The `EM_DIS` signal is used to the control relay effectively cutting the power to the motor when a safety condition is unmet.

5 Controller Software Development

Throughout this section the software for the controller board is developed. An analysis is made to determine the required functionality of the software. An overview of the computing platform and the utilization of it in this project is also given. Software is implemented to test the hardware. Finally a verification is done of the controller board software requirements.

5.1 Analysis

This section will seek to analyse the requirements for the software that runs on the MicroZed mounted on the controller board. The software will need to serve as a base system, where control algorithms can be run for verification purposes. It is not intended as the final system as that system should be Linux based and imposes a completely different set of requirements on the system.

5.1.1 Cart Position

In order to control the position of the cart it is important to know its absolute position. The `HEDS5540` [17] encoder mounted on the motor has an `A`, `B` and `Z` incremental quadrature and can be used to measure the relative position of the cart. The absolute position of the cart should be referred to the rail and therefore the `Z` signal cannot be used to determine the absolute position. The endstops have fixed positions on the rail and the signals from them can therefore be used to infer the absolute position. Specifically, resetting the position when reaching endstop 1 should be done to calibrate the system. Hereafter the absolute position can be measured by reading signals `A` and `B`. Implementing this functionality on an FPGA using VHDL has the advantage that all signals can be read simultaneously. This means that a correct VHDL design will ensure that no edges on pins can go unnoticed. Furthermore it has the advantage that the processor does not need to do any computation and no jitter is introduced, since no interrupts are used. Therefore it was decided to implement the functionality of measuring absolute cart position in a VHDL module.

5.1.2 Controlling the Motor Driver

The motor driver has three inputs that need to be produced by the controller board software. More specifically, two PWM inputs and a disable pin that disables the outputs of the motor driver. The PWM signals will

be applied to the motor and therefore it was chosen to use a frequency of 22kHz in order to be outside of the audible range. It was decided to only apply a PWM signal to one channel in order to reduce the stress and current through the MOSFET and the motor. This means that in order to change direction of the motor, the PWM signal needs to be switched between the two PWM channels. The duty cycle of the PWM needs to be adjustable in order to control the speed of the motor.
As described earlier, the current through the motor needs to be sampled using an ADC. The current should be sampled at the center of a PWM period, as switching introduces noise that should not be measured. This is easily done by implementing center aligned PWM and sampling at the top signal.

It is beneficial to implement the PWM generation on the FPGA part of the MicroZed as it has no dedicated PWM generating hardware and implementing it in software would require computational time on the processor.

5.1.3 General Peripheral Hardware Control

As described in section 4.3.4, two relays should be controlled by the controller board software. An inrush relay that allows for safe charging of the capacitor bank and the main relay. Both are controlled by controlling the logic level of an output pin. Furthermore the software needs to control the three RGB LEDs on the controller board. In order to utilize the full functionality of the RGB LEDs each channel of the LED should be controlled by a PWM signal. Using this method an LED can be illuminated in all colours. The frequency of the PWM is not of great concern as long as it is faster than a human eye. A higher frequency means a higher loss in the MOSFETs and therefore a frequency of 1kHz was chosen, this is fast enough to avoid visible flickering.

5.1.4 Interboard Communication

The joint boards need to transmit joint angles to the controller board wirelessly using the `nRFM`. In section 7.3.2 it is determined that two bytes are needed to represent the joint angle data. The rate of transmission of joint angles should be as high as possible while still maintaining reliable transmission. Throughout this section this aspect is explored further.
With knowledge of the message size, it is possible to determine the theoretical maximum transmission rate. The `nRFM` is capable of transmitting up

to 2Mbps. It uses Enhanced Shockburst, a protocol developed by Nordic Semiconductor specifically for their RF modules. This protocol adds an overhead for each packet of 6 bytes, including a CRC check and an address, amongst a few other features. Each packet is therefore ≈ 8 bytes with only 25% of the packet being the intended payload. This effectively limits the data rate to 0.5Mbps. It is therefore possible to transmit a position at a rate of 31250Hz. Since the two joints are communicating with the same module on the controller board, this frequency is halved resulting in 15625Hz. Transmission at this frequency does entail a few problems, discussed in the following paragraphs.

nRF24L01 Mode Cycling

When transmitting at a rate of 15625Hz a message is to be sent every 64μs According to the datasheet of the `nRFM` it can be kept in transmit mode for no longer than 4ms, meaning that at the most 62 packets can be sent before cycling to standby-mode and back to TX-mode. Before entering TX-mode a 130μs settling period is inferred. This means that at least two packets are lost during this time. The problem can be overcome by lowering the transmission frequency such that the time between packets is above 130μs.

Air Collisions

There is a risk that two packets will collide if the joints transmit at the same time. Due to uneven drift between the two microcontrollers on the joints it is unlikely that collisions will occur indefinitely, however even a few lost packets in a row can cause severe problems in the control of the pendulum. The lowered transmission frequency alleviates this issue somewhat, but there is still a risk that multiple packets may be lost. Air collisions could be handled by using the acknowledge feature in the Enhanced Shockburst protocol, this enables the module to resend a message if an ACK message has not been received within 250μs. After 250μs however, the previous position is outdated and the new position should be sent rather than the lost position. Additionally, the ACK message adds 15 bytes to the transmission of each packet, greatly lowering the possible throughput. The `nRFM` is capable of operating on different frequency-bands. By adding a second receiver on the controller board the two joints can communicate on different frequencies, eliminating the risk of air collisions altogether. This realisation was made after the controller board was produced and so the additional `nRFM` has to be retrofitted on the board.

In order to simplify communication between the joint boards and the controller board communcation will be one way only. This allows the `nRFM`s on the controller board to stay in RX-mode, listening for new positions from the joints continuously. The joint boards will, upon powering up, immediately start transmitting the packet mentioned in figure 7.3.

5.1.5 Utilizing the Zynq Platform

The MicroZed has a Xilinx Zynq 7020 chip embedded on the board. This chip has both a Processing System, PS, and Programmable Logic, PL, which enables the use of both concurrent and sequential computing on the same board. The PS is a dual core ARM Cortex-A9 processsor and PL is an Artix-7 FPGA [54]. The two parts are connected through the AXI bus.

Concurrent Computing
Programming an FPGA will be done using VHDL, which is a description of hardware rather than a programming language. Everything in an FPGA is essentially electronics which makes it possible to do computation in true parallel. An FPGA is inherently real-time as it can be analysed exactly how many clock cycles propagation delay there is through a system.

Sequential Computing
Programming on a processing systems such as the ARM processor is sequential computing. Everything that should be calculated is done by the CPU, which means that code is run sequentially. True parallel computing cannot be done on such a chip, but using an OS and tasks it is possible to achieve an approximation to parallel computing. It is possible to make real-time software on a processing system if care is taken by the programmer to meet the required deadlines.

The AXI Bus and IP Cores
The AXI, Advance eXtensible Interface, standard is part of Arm AMBA, an industry standard for communication between components on SoCs. AXI4 has been co-developed with Xilinx in an effort to tailor it for use in FPGAs. Perhaps not surprisingly, Xilinx uses this protocol extensively in their IP, Intellectual Property. It increases portability by allowing a single interface between different components. The use of IP allows companies to share functionality without disclosing the underlying code, but also simplifies the connection of various components since every component can be verified independently of the others and can be easily connected using the Block

Design feature of Vivado. AXI is a memory-mapped interface, meaning that communication through the bus is essentially a controlled sharing of specific memory. Communication between PS and PL requires the use of the AXI bus.

There are three different implementations of the AXI bus:

- **AXI4-Full:** This implementation enables burst transmissions, allowing multiple data transmissions per interaction.
- **AXI4-Lite:** This implementation is lower bandwidth than the above implementation but is especially useful for setting control registers and transmitting low volumes of data.
- **AXI4-Stream:** This implementation foregoes the use of addresses and allows a continuous stream of data.

Since the data exchanged between PL and PS in this project is mostly the setting of control registers and reading cart positions at a steady rate the simpler AXI4-Lite implementation is going to be used for the components created in this project.

5.1.6 Real-Time Software

In general control systems are created as real-time systems. This ensures that any time-stepping done in the mathematics of the system is well-defined and that the control is therefore well-defined. Also the literature used as an inspiration for this project [7] [16] [15] describe real-time systems based on real-time Linux and DSpace. Therefore the software developed here, will also be real-time.

5.1.7 VHDL and Coding Practices

Some of the software components are written in VHDL. As the reader likely knows, programming in VHDL can be a complex venture. The Zynq platform adds another layer of complexity in requiring communication through the AXI Bus. In order to minimize the complexity of the VHDL programming task a number of practices will be employed by the authors:

- **Utilization of Test Benches:** VHDL was developed primarily as a means to simulate hardware. This ability means that any code can be verified in simulation before being synthesised. In simulation the value

of any register and signal can be monitored, greatly easing the task of debugging the system. Additionally, each iteration is significantly faster since it foregoes the need for the complete bitstream on every iteration.

- **Atomic Components:** In the experience of the authors, the complexity of writing VHDL rises rapidly with the number of features of a given component. Writing components in their atomic form, that is, writing components with as few features as is possible, makes it significantly easier to debug and finalise each component. More complex components can then be made up of several smaller, known-to-function components.

- **Utilization of IP Cores:** Vivado, the development environment for the Zynq platform, relies heavily on the concept of Intellectual Property Cores. VHDL components can be compiled into an IP core which can then be easily distributed amongst multiple users. They also provide a means to more easily build a system by connecting several IP cores of arbitrary complexity.

In addition to the points made above, it is the expectation that future developers will develop Linux drivers which provide an intuitive interface to the firmware developed in this project.

5.2 Requirement Specification

The requirements specified below are tested and verified in section 5.4.

Functional:

23. Correct cart position should be known at all times.
    - Encoder signals should be used to measure relative position.
    - Endstop signal should be used to calibrate and infer the absolute position.
24. General peripheral hardware should be controlled.
    - Main relay and inrush relay should be controlled through output pins.
    - 3 channel 1kHz PWM with adjustable duty cycle should control each RGB LED.

25. Software should produce correct signals to motor driver that allows for control of speed and direction of the motor.
    - 22kHz PWM signals on two channels.
    - Adjustable duty cycle.
    - Control `DIS` signal.
26. Receive joint angle data from the two joint boards using the `nRFM`.
    - MicroZed as SPI master.
    - Setup of `nRFM` settings.
    - Receive joint angle data form joint boards.
27. Real-time behaviour of software.

Design:

28. VHDL modules should be designed and implemented using the following practices.
    - Utilization of Test Benches.
    - Atomic Components.
    - Utilization of IP Cores.

Safety:

29. Software should handle emergency situations.
    - Should detect emergency condition.
    - Should be able to produce an emergency signal.

5.3 Design and Implementation

This section presents the design and implementation of the software written for the controller board. The process used for writing VHDL components in this project is presented and the associated C code shown. Finally the setup of wireless communication for the MicroZed is described.

5.3.1 Implementation of Cart Position Counter

A number of VHDL components were designed and implemented for use on the MicroZed. An indepth description of each is not necessary and the authors have decided to instead focus on the implementation of just the cart position counter as it provides a good description of the workflow used for each component.

The cart position counter is, as the name implies, responsible for maintaining knowledge of the current position of the cart. This is done by decoding the incremental quadrature of the motor and reading the state of the endstops. Absolute knowledge of the cart position can be maintained by initially moving the cart to one of the endstops, setting the position to zero and then keeping the cart between the two endstops. The decoding task is done by a seperate component which is then instantiated in the counting component.

Incremental Quadrature Decoding

The motor provides three signals for incremental quadrature, `A`, `B` and `Z`. Figure 5.1 is a depiction of this quadrature scheme. The two signals `A` and `B` are 90° out of phase and their frequency is determined by the angular velocity of the joint. As can be seen from the figure the signals create four unique stages, counting these allows knowledge of both the direction, position and, potentially, the velocity of the joint. The `Z` signal is a unique pulse which happens on every revolution of the motor.

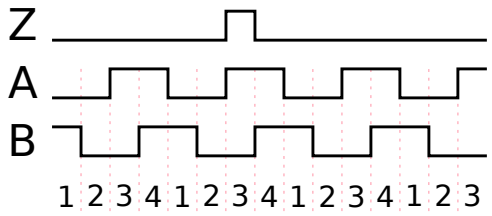


Figure 5.1: Incremental quadrature scheme as implemented on the `HEDS5540`.

State Machine Structure

Each state machine written for this project is done using a Moore SM in a three-part structure:

- **Sync Process:** This is the only process which is synchronized with the clock signal. It is used to ensure that state and signals are changed only on the rising edge of the clock.

- **Output Decode:** In each state the output signals are to be in a specific configuration. Given any state, this process ensures that the signals are in that configuration.
- **Next State Decode:** In this process the next state is determined based on the current state and the current inputs.

Using this structure ensures that each process has one responsibility and will generally result in cleaner, more simple code.

The component described in this section is comprised of two subcomponents. The first is a state machine responsible for reading the current direction of travel and generating a tick on every edge of signals `A` and `B`. A depiction of the SM responsible for decoding the incremental quadrature can be seen in figure 5.2. On this figure `A`, `B` and `Z` are the quadrature signals, `p` and `pl` are the `phase` and `last_phase` variables, `T_i`, `D_i` and `I_i` are the `tick`, `direction` and `index` variables. Note that VHDL notation has been omitted for clarity. As can be seen, the SM is comprised of six states, explained in more detail in the following paragraphs.

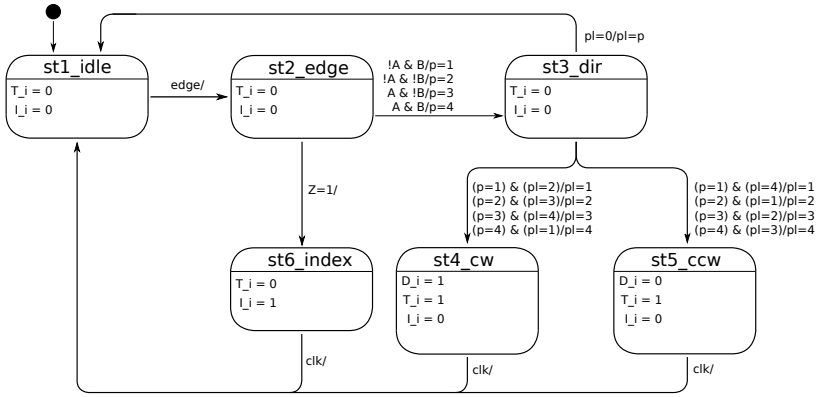


Figure 5.2: State machine of the incremental quadrature decoder created for the MicroZed.

st1_idle: Is the initial state and the state that is returned to between edges on the quadrature signal. Since the quadrature signal is not synchronous with the driving clock of the system the `rising_edge()` and `falling_edge()` functions cannot be used to determine edges. Listing 5.1 shows how an edge is detected on a signal. The current value of the signal is continuously compared to the last value, which is saved on every clock cycle. If the two values

are not equal an edge must have occured within the last clock cycle. In this component both rising and falling edges are used and so any difference between the two is registered. This code is repeated for all three input signals, with the exception that the `Z` channel sets `next_state` to `st6_index` instead.

```
case (state) is
    when st1_idle =>
        if A_i /= last_A_i then
            next_state <= st2_edge;
        end if;
        ⋮
```

Listing 5.1: VHDL edge detection of asynchronous signal.

st2_edge: Is responsible for determining which phase is currently active. The different phases can be seen in figure 5.1. The phase is found simply by comparing the signals as seen in listing 5.2. Regardless of the detected phase `next_state` is set to `st3_dir`.

```
when st2_edge =>
    if A = '1' then
        if B = '1' then
            phase <= p4;
        else
            phase <= p3;
        end if;
        next_state <= st3_dir;
        ⋮
```

Listing 5.2: VHDL for phase detection in incremental quadrature.

st3_dir: Determines the current direction of travel. This can be seen by comparing the current phase to the last phase. Listing 5.3 shows the structure of the code responsible for this step.

```
when st3_dir =>
    if last_phase /= p0 then
        if phase = p1 then
            if last_phase = p4 then
                next_state <= st5_ccw;
            end if;
            if last_phase = p2 then
                next_state <= st4_cw;
            end if;
            next_last_phase <= p1;
        end if;
        if phase = p2 then
        ⋮
    else
        next_last_phase <= phase;
        next_state <= st1_idle;
    end if;
```

Listing 5.3: VHDL for determining direction of movement in incremental quadrature.

As can be seen in the code there is a reference to phase 0. This is required since the initial phase is unknown. During the first run the direction-determination is skipped and `last_phase` is initialized to `phase`. On every subsequent run, depending on the direction, `next_state` is set to either `st4_cw` or `st5_ccw`.

st4_cw and st5_ccw: The SM is in these states for one clock cycle to allow for the outputs to be adjusted and a tick to be generated.

st6_index: The SM is in this state for one clock cycle when the `Z` input goes high and generates a signal to indicate that the index has been reached. While this feature is unused in the cart position counter as it currently exists, it was decided to include it to allow future developers to use it if the need arises.

Cart Position Counter

This component maintains a counter which is incremented or decremented at every tick generated by the incremental quadrature decoder depending on the state of the direction signal. If the endstop 1 signal is set high the counter is set to zero. Upon booting the system the cart should be moved to the rightmost end-stop in order to calibrate the system. Moving the cart

will now increment the position with leftward motion and decrement with rightward motion.

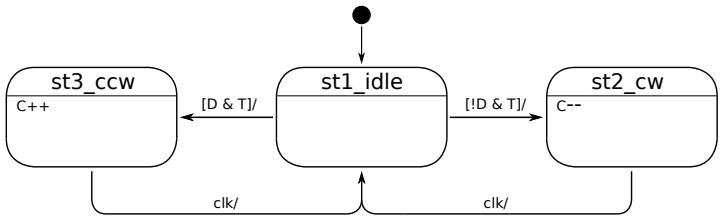


Figure 5.3: State machine of the cart position counter created for the MicroZed.

The SM of the system can be seen in figure 5.3. In this figure `C` is `counter_i`, `D` is `direction` and `T` is `tick`. It should be noted that the end signal is treated as a reset signal and as such is not part of the actual SM. Since it is essentially a reset signal handling of it is placed in the `sync process` to avoid an asynchronous reset of the values. The code in question can be seen in listing 5.4.

```
1  SYNC_PROC:  process(clk)
2      begin
3          if rising_edge(clk) then
4              if emend = '0' then
5                  counter_i <= (others => '0');
6                  state <= st1_idle;
7                  cal_i <= '1';
8              else
9                  state <= next_state;
10                 counter_i <= next_counter_i;
11                 last_T_i <= T_i;
12             end if;
13         end if;
14     end process;
```

Listing 5.4: Synchronous reset based on endstop signals.

The SM has only three simple states. `st1_idle` which is the state where the SM waits for a rising edge on the tick signal, `T`. If the direction signal, `D` is low when a tick arrives the SM proceeds to `st2_cw` if the opposite is true it proceeds to `st3_ccw`. In those states the position variable, `counter_i` is either decremented or incremented.

Test Benches

A part of developing these components has been to develop and use test

benches to verify code as it is being written. VHDL was originally thought as a language to describe and simulate hardware and logic and has later been used as the programming language used to create FPGA designs. All of the original features of the language are still present and as such it is possible to create a component which fully exercises whichever component is under development, the unit-under-test or UUT.

The test bench written for the cart position counter is comprised of three processes, the `clk_proc` which is responsible for generating the driving clock for the component, `quad_proce_ccw` or `quad_proce_cw` for simulating the quadrature signal in either the counter-clockwise or the clockwise direction and finally the `stim_proc` which is responsible for exercising any of the input signals. The `clk_proc` can be seen in listing 5.5. The remaining processes are similar in function but operate on different signals.

```
1  clk_proc : process
2      begin
3          clk_i <= '0';
4          wait for period/2;
5          clk_i <= '1';
6          wait for period/2;
7      end process;
```

Listing 5.5: `clk_proc` is the process responsible for generating the clock signal in the test bench.

The UUT is instantiated as a component in the test bench and Vivado provides tools for investigating the flow of execution. An excerpt illustrating the functionality can be seen in figure 5.4. As can be seen `B_i` is leading `A_i`, resulting in `D_i` going `high`. Since `emend_i` is `high` the component starts counting on every rising edge of `T_i`. When `emend_i` goes `low` the counter is reset and future ticks are ignored until `emend_i` returns to `high`.

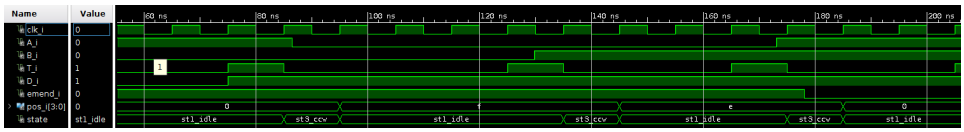


Figure 5.4: Excerpt of testbench simulation results.

Packaging an AXI-Enabled IP Core

Once the component is confirmed functioning as intended, the component

is packaged as an IP core capable of AXI communication. Vivado provides an IP packager which auto-generates the files required for instantiating the components for AXI-communication. The user can choose the number and width of the registers to be available in the instantiated AXI component. In the generated files the user is expected to add their own code and alter existing code such that it correctly implements the desired functionality.

In relation to the components created in this project, the `.vhd` code files are imported to the project and the component instantiated. Every user-defined port is brought out through the AXI component so that they can be constrained correctly in the `.xdc` constraint file. Since every component is made as an IP core the system can be created using a block design. For illustration purposes the block design of the full system has been included in figure 5.5. The rightmost blocks include three of the authors' designs and a Xilinx `axi_gpio` block. The GPIO block is used to signal an unmet safety condition from software, enabling the motor driver and similar tasks. As can be seen, a simple RGB LED controller has been written to handle the debug LEDs on the board. By creating the component to control just a single LED, that code can be instantiated as many times as is required by the application and in any application. In common for all of these blocks is that they require at least some of their input from the AXI bus.

Communication Through the AXI Bus

In this project the AXI bus is used to communicate between PS and PL. Doing this requires setup in both domains. In PL registers are generated which the user can input data to. An example of how this is set up can be seen in listing 5.6. Individual bits in a register can be set to different variables and the remaining bits left unchanged.

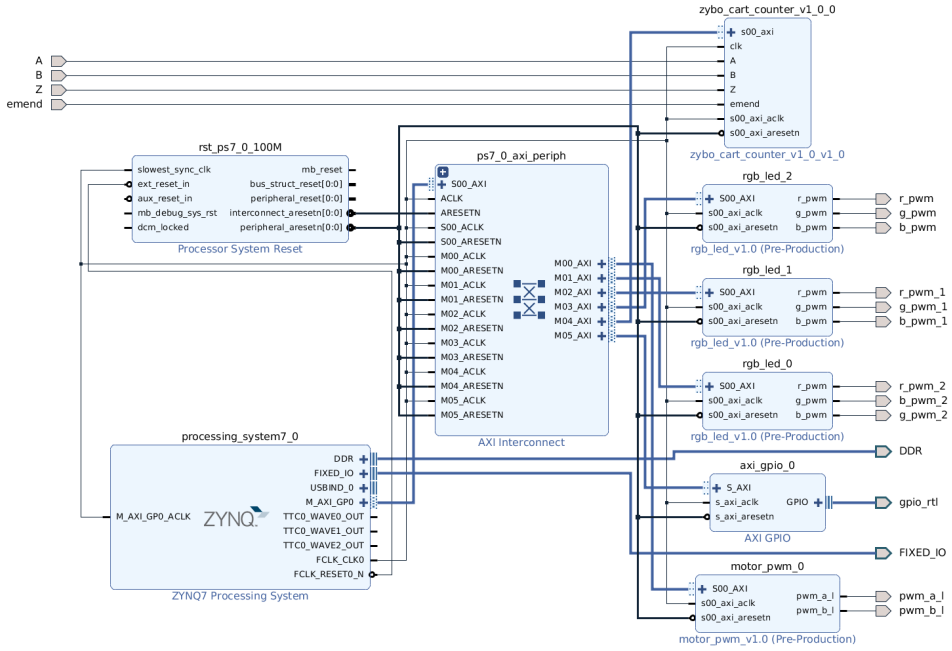


Figure 5.5: Vivado block design of the complete system.

```
cart_counter_inst : cart_counter
    port map(
        clk => clk,
        A => A,
        B => B,
        Z => Z,
        emend => emend,
        cal => slv_reg0(1),
        dir => slv_reg0(0),
        pos => slv_reg1
    );
```

Listing 5.6: Code showing the connection of AXI registers to the cart position counter signals.

In PS these registers can be accessed using the `Xil_In32()` and `Xil_Out32()`. These functions require an address and in the case of the output function, a payload to write to the register. Each device in the design is given a base address which can be found in the `xparameters.h` header file which is generated for each hardware design. Accessing a parameter is then done by adding the appropriate offset. The offset is determined by the width of the

registers. In this case registers are 32 bits wide and so every offset is `0x04` bytes. In code this is handled as seen in listing 5.7

```
#define CART_BASEADDR XPAR_ZYBO_CART_COUNTER_V1_0_0_BASEADDR

#define CAL_ADDR    0x00
#define DIR_ADDR    0x00
#define POS_ADDR    0x04

#define CALIBRATED  0x01
        ⋮

void get_position(uint32_t *position)
{
    if(is_calibrated() == CALIBRATED)
        *position = Xil_In32(CART_BASEADDR+POS_ADDR);
    else
        position = NULL;
}

        ⋮
void cart_task(void)
{
    uint32_t* pos = NULL;
    if(get_position(pos) != NULL)
    {
        //Do Something
    }
        ⋮
}
```

Listing 5.7: Excerpt from the code written to handle the communication with the cart position counter. `is_calibrated()` returns the status of calibration.

Here a wrapper function for `Xil_In32()` is shown. A number of these are written to simplify the code and reduce the risk of errors. In this case the position is only valid if the calibration bit is set `high`. When it is not, a null pointer is returned. Rather than checking both the calibration status and the position, both can be done using a single function call.

5.3.2 Interfacing the nRF24L01

The `nRFM` is interfaced through a standard SPI connection and a `CE` pin that is used to activate the chip. It also has an interrupt pin, but it was chosen

not to use it and instead poll the device. Configuration of the device is done through SPI.

Setup

The `nRFM` has multiple different modes of operations and care should be taken when setting it up. The flowchart of figure 5.6 gives an overview of what registers are set in order to gain the settings used in this project. A description of each step and register is given below.

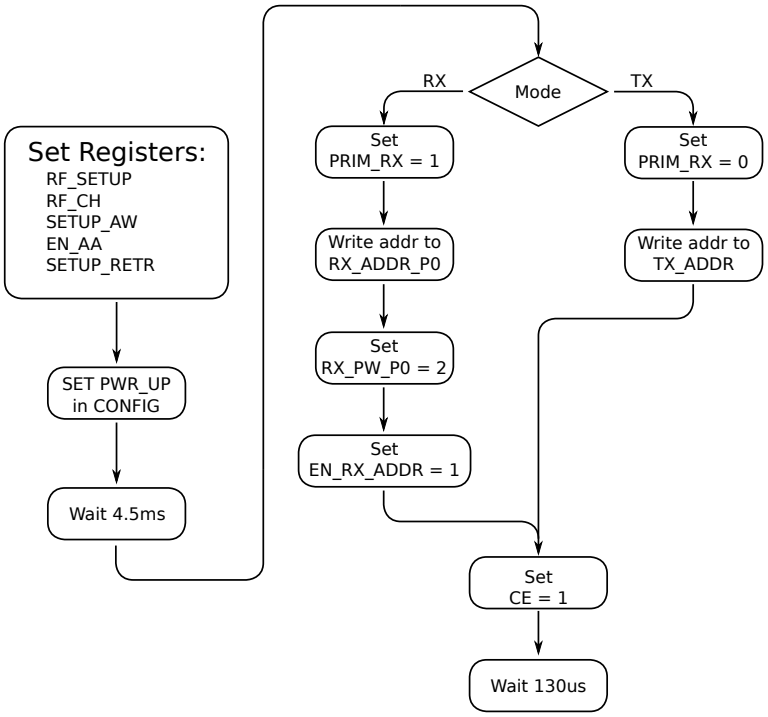


Figure 5.6: Flowchart showing the setup process of the `nRFM` module.

- `RF_SETUP` register is set in order to gain 1Mbps air transmission and maximum output power in TX mode.
- `RF_CH` register is used to set the channel used. It should be set differently for each `nRFM` pair.
- `SETUP_AW` register is set to use the shortest available address width of three bytes.

- `EN_AA` register is set to disable auto acknowledgement.
- `SETUP_RETR` register is set to disable automatic retransmission.

The `PWR_UP` bit is then set to `1` to power up the module. A wait of 4.5ms is then initiated to allow for proper start-up of the crystal.

If receive mode, `RX`, is needed the `PRIM_RX` bit should be set to `1` and the receive address written to `RX_ADDR_P0`. Furthermore `RX_PW_P0` is set to 2 in order to set the payload size to two bytes. Finally `EN_RX_ADDR` is set to enable data on data pipe 0 and `CE` is set `high` to initiate the transition to `RX` mode.

If transmit mode, `TX`, is needed `PRIM_RX` should be set to `0` and the transmit address should be written to `PRIM_RX`. Hereafter `CE` is set `high` to initiate the transition to `TX` mode.

Writing to a Register
Figure 5.7 illustrates the bits that are sent to the `nRFM` in order to write a value to a register. The three command bits specify if the action to be done is read or write. The five address bits specify address of the register that is to be written to. The next one to five bytes is the value that are to be written to the register.

Figure 5.7: Bits written to the `nRFM` in order to write a value to a register. `Command` specifies if it is a read or write action, `Address` specifies the register address and `Value` specifies the value written to or read from the register.

The C code in listing 5.8 is used to write a value to a register. Line 5 makes sure that a valid address and command is sent, regardless of the inputs. Line 6 collects all bits that needs to be sent in one variable. The 2 bytes are sent using SPI in line 7, where it can also be seen that the answer from the `nRFM` is ignored as the null pointer is given as the input buffer.
The corresponding SPI signals are measured and plotted in figure 5.8. It can be seen that the `MOSI` signal corresponds to `0b00100000, 0b00110101`, which means that the value `0b00110101` is written to the `RF_CONFIG` register which has address `0x00`. It can also be noted that data is available on the

```c
#define W_REGISTER      0b00100000
#define REGISTER_MASK   0b00011111

void RF_write_register(XSpiPs *SPI_inst, u8 reg, u8 value){
    u8 cmd_addr =  (W_REGISTER | ( REGISTER_MASK & reg));
    u8 output_buffer[] = {cmd_addr, value};
    XSpiPs_PolledTransfer(SPI_inst, output_buffer, NULL, 2);
}
```

Listing 5.8: Implementation of a C function that writes a register value to a specific register on the nRFM. Macros are shown for clarity.

MISO signal. The nRFM always responds with the value of the STATUS when detecting a falling edge on SS.

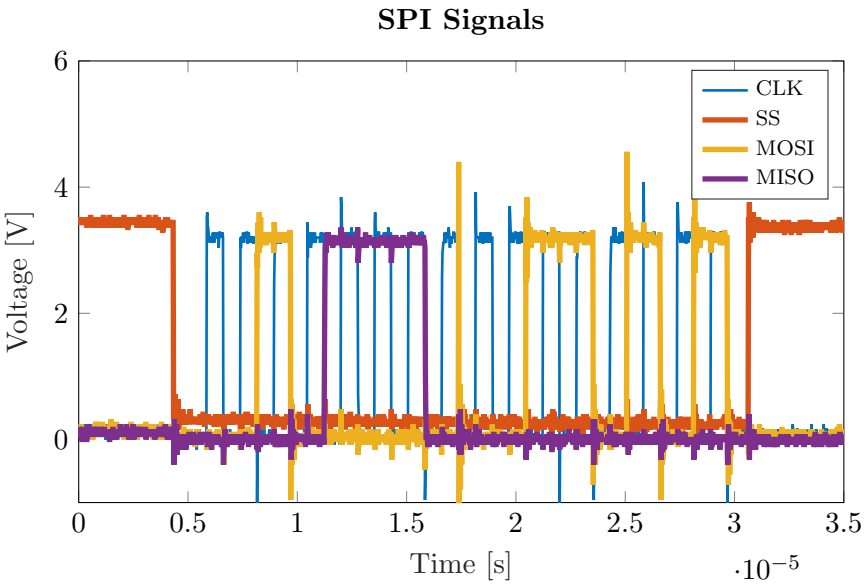


Figure 5.8: SPI communication between nRFM and the MicroZed. Shown is a write command of RF_CONFIG register, 0x00. The value written to the RF_CONFIG register is 0b00110101. 2 bytes are exchanged in total. The SPI transmission is setup and measured at 625kHz, for readability.

Reading a Payload

Reading a payload from the nRFM is done as shown in figure 5.9. In general the first byte corresponds to the command given. In this case it would be 0b01100001 in order to read a payload. The payload is the next one to 32

bytes sent by the nRFM.

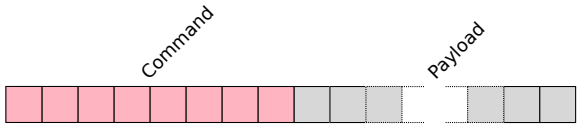


Figure 5.9: Bits written to and received from the nRFM when reading a payload. Command specifies the command given and Payload is the bits written from the nRFM.

Listing 5.9 shows how a payload of 32 bytes is read in C code. Line 5 defines the input and output buffers and line 6 assigns the read payload command to the first byte of the output buffer. 33 bytes of data is sent in line 7-8, this is needed in order to create the clock signal for the nRFM, when it sends the 32 bytes payload. The payload is also read and assigned to the input buffer in line 7-8. Line 9-12 extracts the 32 bytes payload and ignores the first byte sent from the nRFM, which is the value of the STATUS register.

```
1  #define R_RX_PAYLOAD       0b01100001
2  #define PAYLOAD_SIZE       32
3
4  void RF_read(XSpiPs *SPI_inst, u8 *buffer){
5      u8 input_buffer[PAYLOAD_SIZE+1], output_buffer[PAYLOAD_SIZE+1];
6      output_buffer[0] = R_RX_PAYLOAD;
7      XSpiPs_PolledTransfer(SPI_inst, output_buffer, input_buffer, ⇒
8          PAYLOAD_SIZE+1);
9      int i;
10     for(i = 1; i < PAYLOAD_SIZE+1; i++){
11         buffer[i-1] = input_buffer[i];
12     }
13 }
```

Listing 5.9: Implementation of a C function that reads 32 bytes payload from the nRFM. Macros are shown for clarity.

5.3.3 PWM Generation

In order to generate the two needed PWM signals from the MicroZed an IP core must be made in VHDL. As described the IP core needs to be able to switch the PWM signal between channel A and B and have a programmable duty cycle. It also needs to have a top signal that indicates the center of the PWM pulse. The designed PWM generator IP core has three inputs and three outputs as illustrated in figure 5.10. A counter is incremented to a high limit and decremented to zero using a state machine. The high limit

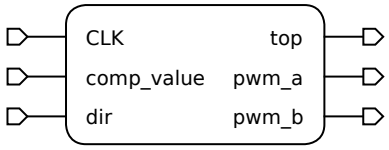


Figure 5.10: Input and output signals of the PWM generator component.

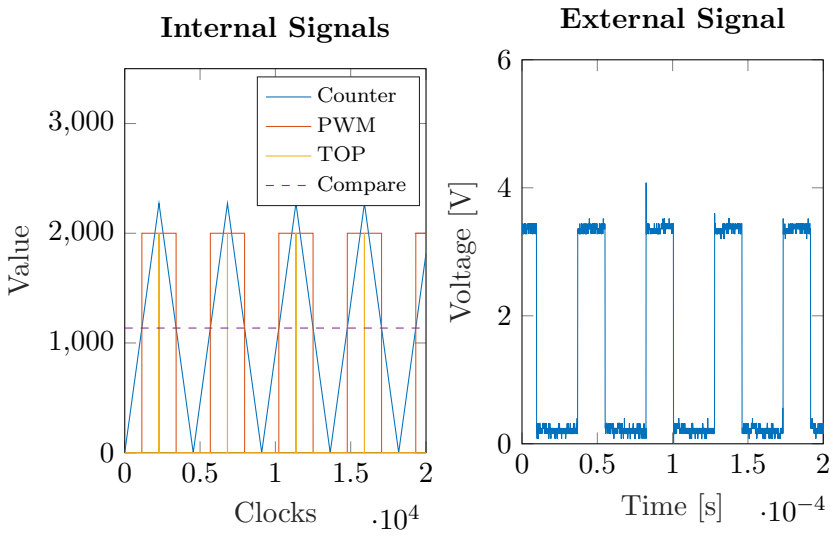
for the counter when using the MicroZed and a PWM frequency of 22KHz is calculated by:

$$H_{limit} = \frac{F_{Microzed}}{F_{PWM} \cdot 2} = \frac{100 \cdot 10^6}{22 \cdot 10^3 \cdot 2} = 2272.7 \simeq 2273 \tag{5.1}$$

Where H_{limit} is the high counting limit, $F_{MicroZed}$ and F_{PWM} are the frequencies of the MicroZed and the PWM, respectively. The PWM output will be set `high`, when the counter value is greater than a compare value. Using this mechanism the duty cycle of the PWM can be controlled through the compare value. The compare value for a 50% duty cycle should be calculated using:

$$Compare = (1 - D) \cdot H_{limit} = (1 - 0.5) \cdot 2273 = 1136 \tag{5.2}$$

Where $Compare$ is the compare value and D is the wanted duty cycle ratio. The described mechanism is illustrated in figure 5.11a, where the top signal is simply set `high`, when the counter reaches its high limit.
Generating the PWM signal from the counter and compare values is done in lines 4 to 10 of listing 5.10. Setting the PWM on to the correct channel is done in lines 11 to 15. As `pwm_a_l` and `pwm_b_l` are written to while in a process, they will only be updated with the last written value, at the end of the process.



(a) Internal VHDL signals illustrated by a Matlab plot.

(b) External PWM signal measured using an oscilloscope.

Figure 5.11: Internal and external signals associated with the PWM generator IP core. PWM and TOP are logic signals and 2000 represents their `high` value.

```
1  OUTPUT_DECODE: process (state,counter,dir)
2  begin
3      ⋮
4  if (counter > unsigned(comp_value)) then
5          pwm_a_l <= '1';
6          pwm_b_l <= '1';
7      else
8          pwm_a_l <= '0';
9          pwm_b_l <= '0';
10     end if;
11     if( dir = '1') then
12         pwm_b_l <= '0';
13     else
14         pwm_a_l <= '0';
15     end if;
16 end process;
```

Listing 5.10: VHDL code generating PWM and setting it to the correct channel.

5.3.4 KHAos and Real-Time Software

KHAos is a run-to-complete scheduler written by Karsten Holm Andersen, lecturer at SDU. It is written to provide a simple operating system, OS, for use on bare-metal systems. The OS includes a timer module and a CPU config module which are written specifically for the platform that the OS should be running on. The software also includes the scheduler itself, `schedule.c` and a pair of configuration files `rtcs.h` and `rtcscnf.h`.

It should be noted that while the authors did set up a functioning KHAos system it saw no actual use in the verification of features due to time constraints. The steps to properly configure KHAos are presented here to provide an overview of the process. Defining the task names is done in `rtcscnf.h`. An excerpt of the file can be seen in listing 5.11. As can be seen, every task created in the system is associated with an ID and is given an init function. The init function is used to set up hardware and other pre-requisites for the task and eventually to start the task.

```
1  #define T_TICK     1
2
3  #define LAST_TASK  1
4
5  #define TASK0      alive_task
6  #define TASK1      cart_task
7  #define TASK2      NULL
8  #define TASK3      NULL
9
10 #define INIT_TASK0 init_alive_task
11 #define INIT_TASK1 init_cart_task
12 #define INIT_TASK2 NULL
13 #define INIT_TASK3 NULL
14
15 #define ALIVE_TASK 0
16 #define CART_TASK  1
```

Listing 5.11: Excerpt of the configuration file of KHAos. Each task is given an ID for both the task itself and its init function. The number of tasks in the system is given in `LAST_TASK`.

The timer has been set up to provide a tick every 10μs. This means that the period of the scheduler can be set in multiples of 10μs by setting the `T_TICK` variable. In this project it was chosen to use the shortest possible frequency while setting up the system. Clearly, when more tasks are added to the system it is crucial to ensure that the deadline of each task is still upheld.

For the sake of verifying the system a simple `alive_task` is set up which toggles one of the RGB LEDs on the controller board. This task can be seen in listing 5.12. Every 0.5s this task flips the state of `RGB2` between the `off`-state and lighting the green LED. The functions `set_green()` and `set_off()` are wrapper functions which hide the calls to `Xil_Out32()` in an effort to simplify the code.

```
1  void alive_task(void)
2  {
3      if(led)
4          set_green(RGB2_BASEADDR);
5      else
6          set_off(RGB2_BASEADDR);
7
8      led = !led;
9
10     _wait(MILLI_SEC(500));
11 }
```

Listing 5.12: Alive task in KHAos. This task repeatedly switches one of the RGB LEDs on the controller board to green and `off`.

5.4 Verification

This section goes through the verification of the requirements set for the controller board software.

5.4.1 Verification of: Requirement 23

This requirement states that correct cart position should be known at all times.

Test
Cart should by moved back and forth and the cart position variable should increment and decrement accordingly. The position should be set to zero, when moving the cart to the position of endstop 1.
Drifting of the counter should be tested by placing the cart at a fixed position and noting the position variable. Then the cart should be moved back and forth for some time and then placed at the initial fixed position. The position variable should have the same value as when initially held at the fixed position.

Conclusion
The test procedure described above was followed and it was found that the cart position variable is correctly increased, decreased and it was set to zero when reaching endstop 1.
The drifting test was done by noting the counter position at the position of endstop 2. Hereafter the cart was moved back and fourth at different velocities for several minutes before returning it to the place of endstop 2. The position counter was the same, indicating that drifting is not present in the system.

The length of the rail from endstop 1 to endstop 2 was measured to be 72387 ticks.

5.4.2 Verification of: Requirement 24

This requirement states that general peripheral hardware should be controlled.

Conclusion
It was tested that the two relays could be turned `on` and `off` successfully from the software. Furthermore it was tested that the three RGB LEDs could be illuminated in all colors using the PWM generation.

5.4.3 Verification of: Requirement 25

This requirement specifies that software should produce correct signals to the motor driver that allows for control of speed and direction of the motor.

Test
PWM outputs and I/O pin `DIS` of the MicroZed should be applied to the motor through the controller board circuitry. Through programming it should be possible to set speed and direction of motor.

Conclusion
The MicroZed was programmed and inserted in the connector on the controller board. Setting different duty cycles yielded different velocities of the motor - as expected. By programming the `DIR` bit it was possible to change the motor direction. The motor was successfully stopped by setting the `DIS` variable `high`.

5.4.4 Verification of: Requirement 26

This requirement states that the software should receive joint angle data from the two joint boards using the `nRFM`.

Conclusion
Testing this requirement requires software to run on the joint board. The joint board has a similar requirement, which states that it should be able to transmit joint angle data to the controller board. This requirement is tested and concluding remarks are given in section 7.4.2.

5.4.5 Verification of: Requirement 27

This requirement states that all software should be real-time.

Conclusion
It was the intention to collect all of the written software in a real-time system based on KHAos. This goal was not reached due to time constraints of the project.

5.4.6 Verification of: Requirement 28

This requirement states that VHDL modules should be designed and implemented using the practices:

- Utilization of Test Benches.
- Atomic Components.
- Utilization of IP Cores.

Conclusion
This was successfully done as described in section 5.3.1.

5.4.7 Verification of: Requirement 29

This requirement specifies that software should be able to handle safety conditions.

Test
The `EM_DIS` signal should be driven high by moving the cart to one of the endstops. This indicates an unmet safety condition and the `EM_DIS` should be recorded using the MicroZed.
The `EM_MCU` signal should be driven `low` to ensure that this causes an unmet safety conditions.

Conclusion
It was found that the `EM_DIS` signal that indicates a safety condition was not wired to the MicroZed Bergstak connector. This should be done in order to record safety conditions.

No tests were performed on producing a safety condition because of time constraints.

6 Joint Development

In creating a pendulum the mounting point must be able to rotate freely about the mounting axis. To achieve this, a joint must be designed which allows this motion. There are two aspects to the development of the joint: a mechanical aspect and an electronic aspect. This section will cover the analysis, design, implementation and verification of both. The overview of the pendulum assembly is reproduced in figure 6.1 for convenience.

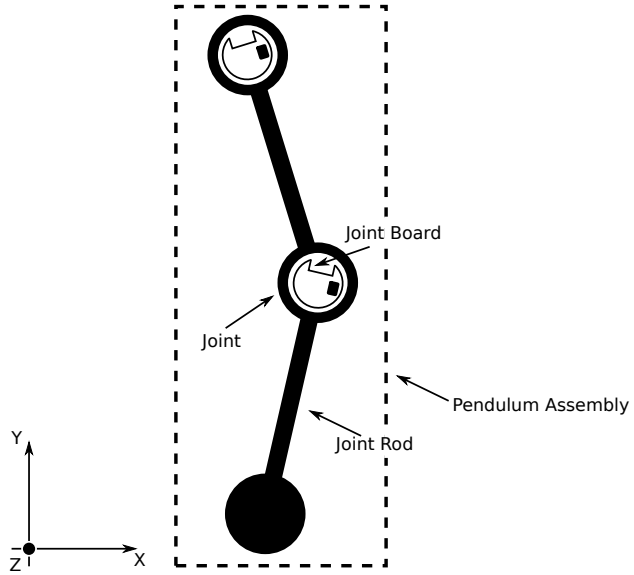


Figure 6.1: Overview of the pendulum assembly developed in this section.

6.1 Analysis

In the system analysis a number of requirements were found relating to the design of the joints. These requirements contain both mechanical and electrical aspects. The authors' main focus is on power and embedded design and as such the design of the mechanics of the system is limited in comparison to the electrical design. The mechanical design was developed with advice from mechanical engineer at SDU: Jørgen Maagaard.

6.1.1 Mechanics

The mechanical design consists of two parts, the joint and the rod connecting two joints. This project seeks to create a two-joint pendulum but the project

owner has set a few additional requirements to allow for more diverse control tasks in future projects:

- It should be possible to mount n joints in series on the pendulum assembly.
- The distance between the joints should be modular.

In addition to these two requirements:

- The pendulum assembly should be rigid and allow only movement in the X-Y plane so that it cannot collide with itself.

The first two requirements would benefit from a modular design, allowing the rod to be dismounted from the joint so that only the rod would require replacement, should the user wish to add joints or vary the distance between joints. This is somewhat in conflict with the last requirement, which would benefit from each joint-rod-joint assembly to be one solid piece of material. Since such an assembly would be rather expensive to manufacture and would require a new assembly for each desired configuration, a modular design is preferred.

The joint should be capable of unrestricted rotation. This rotation should be as close to frictionless as possible and so a low friction coupling should be used between the two parts of the joint. As will be discussed in section 6.1.2 some form of encoder is required in the joint. This encoder requires both power and signal wiring. Two solutions were considered to solve this problem

- **Slip Ring:** A slip ring allows wiring to pass through a freely rotating joint. They are available in versions of various sizes with up to 72 wires, allowing for several series joints before being limited by wire capacity. Clearly, more wires results in a longer slip ring, imposing additional requirements to the thickness of the joint to accomodate for the slip ring. The authors were in contact with Rotary Systems and Moog, both industry suppliers of slip rings and similar products. It was clear that the price of these components was well beyond what is reasonable for this project.
- **Wireless:** Several wireless solutions exist which could be employed in this situation. Using a wireless approach means that all computation and handling of signals must be done on the joint before transmitting

to the main controller. This means that a more complex PCB must be designed and a battery installed to power the PCB. While this solution does add electrical complexity it is practically free in comparison to the slip ring and allows for a much slimmer joint design. This is the solution chosen for the joint design.

As discussed in the State of the Art analysis the pendulum assembly is viewed as a collection of point-masses connected with massless rods. In order to more accurately uphold this assumption the material of the rods should be significantly lighter than the material of the joints while still maintaining sufficient rigidity. In general three materials are easily available to the authors for manufacturing:

- **Metal:** While this is a somewhat vague description that spans a vast number of materials, metals are generally significantly heavier than the following materials. Using metals requires manufacturing techniques and equipment which the authors do not have access to but the SDU Workshop can be hired to manufacture a design. The precision of metal-working is well into the sub-millimeter range.

- **3D Print:** This method of manufacturing is easily accessible and can be done for free (to the project). It also allows for creating shapes that are otherwise difficult to manufacture in metals. 3D printing at SDU is done in different types of plastics, neither of which is as stiff as would be desired for the pendulum assembly. Additionally, the encoder chosen in the following section sets strict requirements on the placement of the two parts of the encoder (This is discussed in more detail below) and the 3D printers at SDU are known to have problems accurately representing features in the sub-millimeter range.

- **Carbon:** This material is considered only for the rod as it was suggested as a light material with significant rigidity. Carbon tubes have been used previously at SDU Vikings, the SDU Formula Student racing team, as part of the suspension assembly.

Given the above materials the joint is to be made of metal while a carbon tube will be procured for the rod. The design should be made such that the SDU Workshop can manufacture the joints and any appertaining components.

6.1.2 Encoder

Each joint should implement an encoder to enable tracking the angular position of the joint. The resolution of each joint should be sufficient to allow for proper control. From the State of the Art analysis it was found that a resolution of 8192 ticks per revolution was succesfully used in two systems [16] [15]. In addition to resolution, ease of manufacture and design is also an important aspect. One encoder in particular was recommended to the authors; the Rolin Rotary Magnetic Encoders. This is a series of encoders which, as the name suggests, utilises magnetic fields in order to determine the angular position of the encoder. They consist of a magnetic ring with a number of poles and a PCB sensor. The magnetic ring is available in different sizes with different number of poles which, along with the type of PCB sensor decides the resolution of the encoder. The specific encoder configuration used in this project, the `RL2IC` [36], allows for 7200 CPR or one count per 0.05°. The encoder is incremental with an `A`, `B` and `Z` channel where `A` and `B` are 90° out of phase and `Z` is a unique reference mark. Using the reference mark it is possible to maintain absolute knowledge of the angular position of the joint after the initial encounter with the reference mark. This is only possible assuming that there is no drift or slippage in the system, a reasonable assumption considering the mechanical setup. Communication with the encoder is done using the RS422 standard. This standard describes a method of transmitting digital signals using differential signals and requires specialised hardware to translate between ordinary digital signals and RS422.

6.1.3 Electronics

As per the initial requirements listed in this section each joint should be able to track the position of the joint and communicate this information wirelessly to the main controller board. In order to accomplish this task some form of microcontroller is required along with the power delivery and the RS422 receiver mentioned previously. The microcontroller must fulfill the following requirements:

- Must have support for three interrupt pins, one for each channel of the encoder.
- Must have an SPI interface to maintain communication with the `nRFM`.

- Must have non-volatile memory to store the program through power cycling the board.

The `ATtiny84` [4] fulfills these requirements and provides sufficient I/O pins for the design. The design of the PCB and the layout in general should accomodate the joint design. By minimizing the height of the PCB design, the joint can be made thinner, reducing the stress on the pendulum assembly along the Z-axis. It should also be noted that any connectors should be placed with the expectation that the border of the PCB is not accessible during normal operation.

Since the PCB is powered from a battery the design should be optimized to be reasonably power efficient. As is explained in the next section, the design has a 3.3V rail and a 5V rail. Generally microcontrollers become more power efficient at lower voltages and as such the `ATtiny` will be driven from the 3.3V rail. The authors have immediate access to an AVR programmer which uses 5V signals for the programming. The design should be made to accept this voltage when programming. This can be done using level-shifting between the two voltage levels.

It was chosen to add an LED to the design for debugging purposes. Clearly, an LED does require some power and should not be used during normal operation.

Power Delivery
The required voltage rails on the joint board are dictated by the RF module and the encoder. The former requires 3.3V and the latter 5V. Since the joints are to be wireless it is necessary to power them from a battery. To avoid added complexity there will be no charging circuitry and charging the battery should be done externally to the joint. The battery of the joint should last for as long as is possible while still fitting within the enclosure. This requirement essentially narrows the battery choices to LiPo cells as these generally have the highest energy density of the commercially avaiable battery types. 1S LiPo cells are rated at 3.7V and so some form of convertion is required to reach both the 5V and 3.3V rails. In order to correctly dimension the converters it is necessary to determine the power draw from each rail:

- **3.3V:** This rail powers the `nRFM` which has a maximum supply current of 12.3mA [32] as well as the `ATTiny`. According to the datasheet of

the `ATtiny` it uses approximately 3.6mA at 3.3V when run at 8MHz. This yields a total of 15.9mA at 3.3V or 52.5mW. Additional digital circuitry placed on this rail is considered negligible in the power budget.

- **5V:** This rail powers the encoder with a maximum supply current of 30mA [36] as well as the RS422 receiver which has a supply current of 52mA [42]. This yields a total of 82mA at 5V or 410mW. Additional digital circuitry placed on this rail is considered negligible in the power budget.

For a battery to have sufficient capacity to power the joint for a full workday, estimated at 10 hours, it should have at least $\approx$ 4600mWh. This is infeasible in the required dimensions and a battery should be procured which can provide the most run time given the space available.

The voltage rails should be designed with sufficient overhead. Additionally, the accuracy of each rail is determined from the supply requirements set by the datasheet of the components used in the datasheet. The two rails should be specced as follows: One rail capable of delivering 200mA at 5V$\pm$0.2V. One rail capable of delivering 100mA at 3.3V$\pm$0.2V.

6.2 Requirement Specification

The requirements specified below are tested and verified in section 6.6.

Functional:

30. Pendulum assembly should allow movement only in the X-Y plane.
31. Voltage rails for the components in the circuit.
    - At least 100mA at 3.3V$\pm$0.2V.
    - At least 200mA at 5V$\pm$0.2V.
32. Joint should be able to rotate freely about the mounting axis.
33. The `nRFM` should be included for communication.
34. the RS422 interface of the `RL2IC` encoder should be correctly interfaced.
35. The design should allow programming of the `ATtiny` using 5V SPI.
36. An LED should be employed for debugging purposes.

Design:

37. Weight of rods should be minimized to uphold the point-mass assumption.
38. Joints should be designed in a modular fashion such that n joints can be mounted in series.
39. Distance between joints should be adjustable.
40. Friction in the joint should be minimized.
41. Design of the joint should employ mounting solutions for:
    - Joint board.
    - Encoder.
    - Battery.
42. The design of the joint should be done so that it can be manufactured by the SDU workshop.
43. The PCB should be designed with the enclosure in mind.
    - Height of components and general design should be minimized.
    - Placement of connectors should accomodate the enclosure.
    - Shape of the PCB should allow for easy mounting of the PCB and other components.
44. The PCB layout should be reviewed according to the developed strategy.
    - Documentation.
    - General inspection.
    - Datasheet and report comparison.
    - Footprint inspection.
    - Peer-review.
45. A Battery should be sourced which conforms with the following requirements:
    - Battery should provide power for the joint for as long as possible.
    - Battery should fit within the dimensions dictated by the enclosure.
46. Overall power consumption of the design should be minimized.

6.3 Mechanical Design

The design of the pendulum assembly is, as in the analysis, split into an electrical design and a mechanical design. This section will consist of a presentation of the designed mechanical components.

The mechanical design consists mainly of 3D modelling using Autodesk Inventor. The entire design is shown in figure 6.2 and is comprised of three parts:

Magnet Side: is shown in figure 6.2c. This part has a shallow groove milled for the magnetic disc of the encoder to fit into. The centre of the part is milled out to allow for two low-friction ball bearings to be press-fit into.

Reader Side: is shown in figures 6.2d and 6.2e. The visible sides of 6.2c and 6.2d are designed to mate. A cutout is made in 6.2d which is positioned such that it can be placed accurately above the magnetic disc. The placement of the reader head is done by pressing it against a feeler gauge blade positioned between it and the magnetic disc before securing it using two bolts. This ensures that the reader head is placed parallel with the disc and at the correct distance. The axle is milled as part of 6.2d. The manufactured version has a bore in the axle to accomodate a bolt which tightens the part to the bearings. On the other side of the part, 6.2e, most of the material has been bored away to allow for mounting the joint board as shown in the figure. A lip has been left on top of the part to allow for mounting a lid.

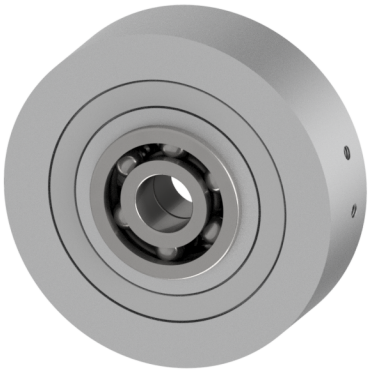
Joint Mount: is shown in 6.2b. This component was made to mate the joint with the joint rod. A hole pattern is added to this part as well as the other two which is used to mount 6.2b to 6.2d and 6.2c. On the top of the part a hole is made to accomodate the joint rod. This part is 3D printed since the shape is somewhat complex and would require access to a CNC machine to mill of metal. Additionally, making the part from metal would increase its weight significantly, potentially risking the point-mass assumption.

These three parts are connected into the pendulum assembly using carbon tubes as shown in 6.2a. Two parts are present on this design which were not described previously. The final disc, known as the dummy disc, is

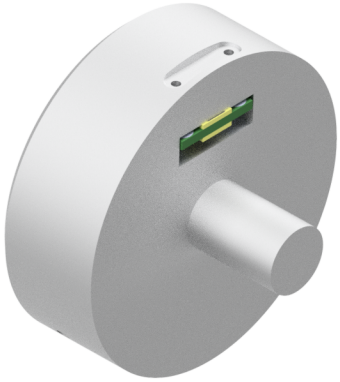
essentially just a solid piece of aluminium used to add some weight to the end of the pendulum assembly. The second part is a plate which the first joint is mounted on which is to be mounted on the cart.

(a) Pendulum assembly.

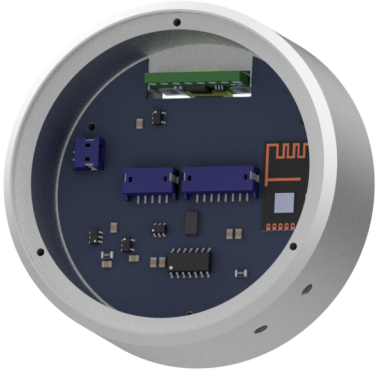
(b) Joint mount.



(c) Joint, magnet side.



(d) Joint, read side. Mating axle.



(e) Joint, read side. PCB cutout.

Figure 6.2: The various parts of the pendulum assembly.

6.4 Electrical Design

This section will give an in depth view of the design process of the electronics designed for the joint board.

6.4.1 RS422

The RL2IC uses the RS422 standard for its output. This means that each A, B or Z channel consists of a two wire differential signal. A line receiver is needed in order to translate the RS422 signals to logic inputs that the ATtiny can read. The DS26LS32CM is a quad differential line receiver that complies with the RS422 standard. When using RS422 there are different termination methods, with no termination being the simplest. The RS422 Standard Overview [37] describes that signal integrity is maintained in a setup of ≈30m wires carrying data at a signal rate of 200kbps without termination. The intended setup on the joint board requires wires no longer than 5cm. The maximum expected data rate can be calculated using the maximum expected angular velocity of the joint of 20Hz and the edges on one channel per revolution.

$$Datarate_{max} = 20 \cdot 3600 = 72kbps \tag{6.1}$$

This datarate is clearly below the 200kbps and it was therefore decided to use no line termination, which also has the advantage that a minimum amount of current is needed from the RL2IC.

6.4.2 Generating a 5V Rail

As described in the requirements specification, a 5V rail is needed on the joint board. Generating the 5V rail can be done by boosting the battery voltage up to 5V, as the battery voltage will always be lower than 5V. The boost regulator SP6641B [9] has an input range of 0.9V to 4.5V and a fixed output of 5V. The circuitry used in this project will be based on the reference design shown in the datasheet, with the one difference that the SHDN (shutdown) pin has been pulled permanently high. The circuit is shown on the full joint board schematics in appendix D.

6.4.3 Generating a 3.3V Rail

A 3.3V rail is needed according to the requirement specification. The task of generating the 3.3V rail is somewhat complicated since the battery voltage of a typical Li-Ion battery varies from 4.2V down to 3.0V. Three different solutions appeared after an initial analysis of the task:

- Buck converter
- Buck-Boost converter
- Linear regulator

The qualities and drawbacks of the three solutions are presented here.

Buck Converter
The voltage from the battery can be converted to 3.3V using a Buck converter. A Buck converter is relatively simple and circuitry can easily be designed for the task. It is only able to supply an output voltage that is lower than the input voltage. This means that when the battery voltage is lower than 3.3V the converter will stop producing a stable 3.3V output. Therefore a portion of the battery capacity cannot be used. The `TPS62220` is a Buck regulator and generally has good specifications for this task. With an input voltage of 3.7V and an output voltage of 3.3V it has an efficiency of approximately 95% with an output current in the range of 1mA to 20mA [20]. The minimum drop out voltage of the `TPS62220` is low, because it has a 100% duty cycle mode. In this mode, the drop out voltage is purely defined by the `ON` resistance of the internal switch, the DC resistance of the external inductor and the output current.
The drop out voltage can be calculated using equation 6.2, [20].

$$V_{drop} = I_O \cdot (R_{DS(on),max} + R_I) \tag{6.2}$$

$$V_{drop} = 0.02 \cdot (0.062 + 0.09) = 14[mV] \tag{6.3}$$

Where I_O is the output current, $R_{DS(on),max}$ is the `ON` resistance of the internal switch and R_I is the DC resistance of the external inductor. The inductor resistance used is the value of the `B82462G4` inductor [11] $R_I = 0.9\Omega$. This means that it can supply 3.3V output when the input voltage is 3.315V or greater

Buck-Boost Converter
The motivation for using a Buck-Boost converter is to allow for full utilization of the battery capacity. This requires that the converter can make a seamless transition between the stepping up and stepping down of the input voltage. `TPS6300` is a Buck-Boost regulator that has these features. The external circuitry is simple, but the drawback of this regulator and converter

is the efficiency of it. With an input voltage of 3.7V and an output voltage of 3.3V it has an efficiency of approximately 75% with an output current in the range of 1mA to 20mA [21].

Linear Regulator
A linear regulator is a component that can regulate the output voltage by varying an internal resistance. Linear regulators have a drop voltage that limit the output voltage. The `LD3985` has an ultra low drop voltage of 20mV at 50mA output current [43].
Therefore the efficiency is proportional to ratio between the output and input voltage and can be estimated using equation 6.4 [46].

$$\eta \simeq \frac{V_{out}}{V_{in}} \tag{6.4}$$

The voltage of a Li-Ion battery varies from 4.2 to 3.0V, but the nominal voltage is 3.7V and this will be used as an estimate of the mean voltage of the battery. Using this the mean efficiency can be estimated as shown in equation 6.5.

$$\eta \simeq \frac{3.3}{3.7} = 0.89 = 89\% \tag{6.5}$$

Comparison
The most important parameters of the three solutions discussed are shown in table 6.1.

	Buck	Buck-Boost	Linear
Efficiency	95%	75%	89%
Drop out [mV]	14	N/A	20

Table 6.1: Parameters of the three solutions using a Buck converter, Buck-Boost converter or a linear regulator. The efficiency shown is estimated with an input voltage of 3.7V, an output voltage of 3.3V and an output current in the range of 1mA to 20mA. The drop out voltage is estimated with an output current of 20mA (Buck) and 50mA (linear regulator).

The Buck converter solution has the highest efficiency and is therefore the natural choice. As already discussed the disadvantage of using a Buck converter is that the full battery capacity cannot be utilized. A discharge test should be conducted to determine the amount of capacity that cannot be utilized.

Battery Discharge

At the time of writing, the chosen battery has not yet been procured and the test will therefore be conducted on a similar battery instead. The battery under test is a 850 mAH Li-Ion battery with the dimensions 49x29x6mm. It should be noted that both the capacity and physical dimensions are similar to the chosen battery. The discharge curve of the two batteries will not be identical, but will be sufficiently close to to allow for deciding which solution should be used. The test was conducted by connecting the battery to an electrical load programmed to discharge with a constant power of 0.3 Watt, while measuring the battery voltage. The measured discharge curve is shown in figure 6.3.

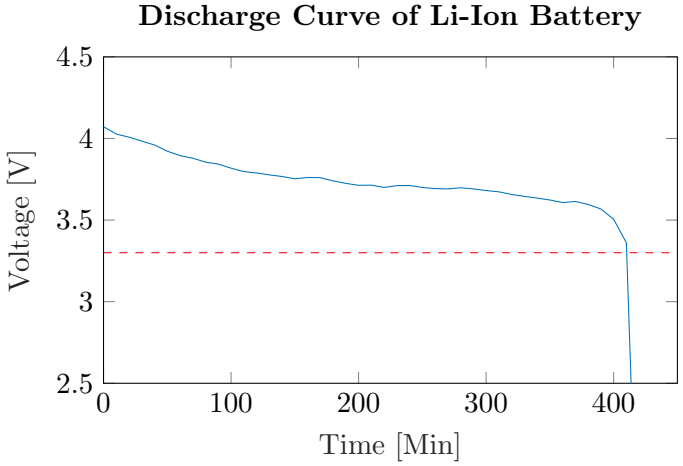


Figure 6.3: Voltages measured across a 850 mAH Li-Ion battery while discharging using an electrical load programmed to 0.3 Watt. Horizontal red line represents the 3.3V level.

It can be observed that the battery voltage only drops below 3.3V for a very short time before the battery is completely discharged. It is calculated that ≈99% of the battery capacity can be utilized.

Conclusion

The Buck converter solution has the highest efficiency and has a lower drop out voltage than the linear regulator solution. The disadvantage of using a Buck converter is that the full battery capacity cannot be utilized, but a test showed that ≈99% of the capacity can be used.
Therefore it was chosen to use a Buck converter with the `TPS62220` regulator. The Buck converter circuit used is based on the recommendations in the

datasheet of the TPS62220. The circuit is shown in figure 6.4.

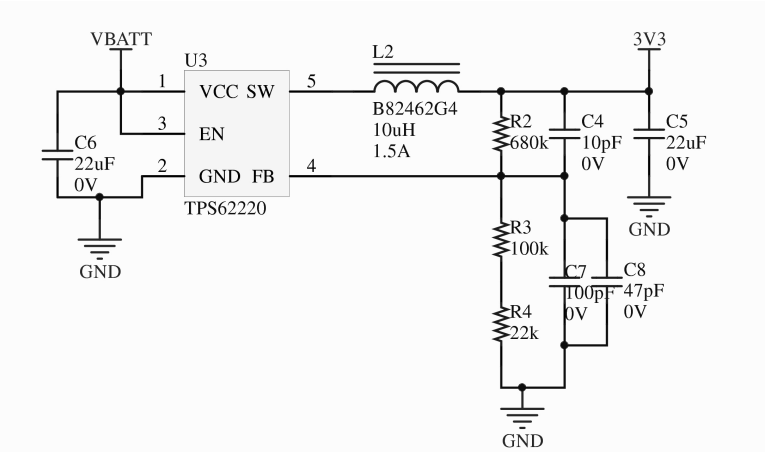


Figure 6.4: Circuitry of Buck converter with an output voltage of 3.3V. TPS62220 is used as the regulator.

6.4.4 Choosing a Battery

It turned out to be rather difficult to procure a LiPo battery. Apparently, they are in the habit of burning violently if damaged and for that reason, strict rules are in place concerning the transportation of them. It was necessary to find a supplier with an appropriate battery who would also be willing to send them. Eventually a danish distributor of Renata batteries was found. Renata provides the ICP582930PR-01 which has a capacity of 430mAH or 1600mWH. This battery is 32x29.5mm and therefore easily fits in the alotted space, a cylinder with a diameter of 57mm and a height of 9mm.

A lid was produced for the joints which included a mounting spot for both the battery as well as a small switch for turning off the joint when it is not in use. A rendering of this lid can be seen in figure 6.5.

6.5 Implementation

This section will elaborate on the implementation of the joint design. Specifically the methods and equipment used is discussed.

Figure 6.5: Lid produced for the joints. The protrution is used to mount the battery for the joint board.

6.5.1 Mechanics Implementation

The joint design created in the previous section was manufactured by the SDU Workshop. As the more experienced mechanical technicians at the workshop worked through the design they worked with the authors to improve on the design to make it practically possible to construct. The implemented system is shown in the picture of figure 6.6.

6.5.2 Electrical Implementation

The joint board was designed by the authors but the PCB was manufactured using an out-of-house PCB fabricator. A picture of the PCB is shown as delivered in figure 6.7a. Soldering the board was done by applying solder paste, adding the components using a pick-and-place machine and baking it in a reflow oven. The finished board is shown in figure 6.7b.

6.6 Verification

Throughout this section the verification of the requirements of the joint board is done. This includes the authors' verification procedure for a newly produced joint board.

6.6.1 Joint Board Verification Procedure

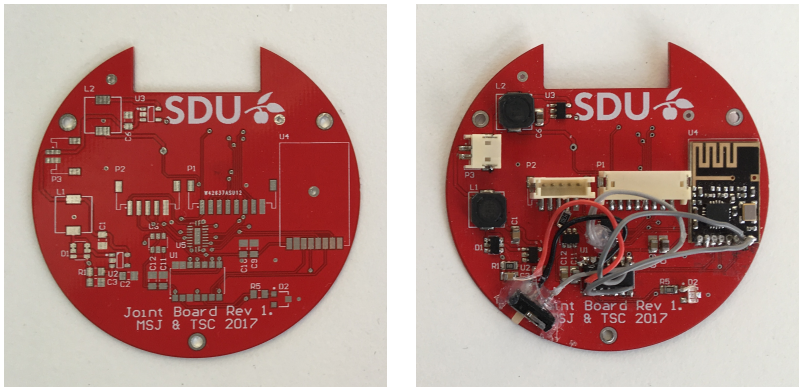
This procedure is created in an effort to allow for an easier debugging procedure of a newly produced controller board, should a developer require the production of a new board. A blank version appropriate for printing can be

Figure 6.6: Picture of implemented joint pendulum assembly.

found in appendix A. ✓ is used to indicate the successful verification of a test while ✕ is used to indicate the initial failure of a test. All pins, components and nets are typeset using the `teletype font` and are referenced to the schematics in appendix D.

Verify Voltage Rails:

- Apply 3.7V to `VBATT`.
  - ✕ Verify that current draw is within reasonable limits (<75 mA).
    - A short circuit seemed to exist on the 5V rail. See next paragraph.
  - ✕ Verify 5V and 3.3V voltage rails.



(a) Joint board PCB as delivered by the fabricator.

(b) Joint board PCB populated with components. Required fixes are applied.

Figure 6.7: Implemented joint board PCB.

- The 3.3V rail functions as intended but the 5V rail is missing. From inspection of the circuit it was discovered that the diode D1 has a fault in the footprint. The six-pin package can be flipped and the centre legs lifted to resolve the issue. During testing the lab supply used to power the board was limited to ≈250mA which, due to the required startup current, yielded what looked like a short circuit. Figure 6.8 shows the current spike and the resulting failure of the 5V rail to rise. In an attempt to find the offending component the current limit was raised to ≈1A. This allowed the 5V rail to properly boot, leading to the discovery of the required startup current. According to the datasheet of the `SP6641` [9] it requires a startup current of ≈0.65A. The resulting current draw and voltage can be seen in figure 6.9.

Verify Voltage Level Shifter Functionality

- Apply 5V to `SCLK_H`.

  ✓ Verify that 3.3V is present on `SCLK`.

- Repeat for signals `MOSI_H` and `RESET_H`.
- Apply 3.3V to `MISO`.

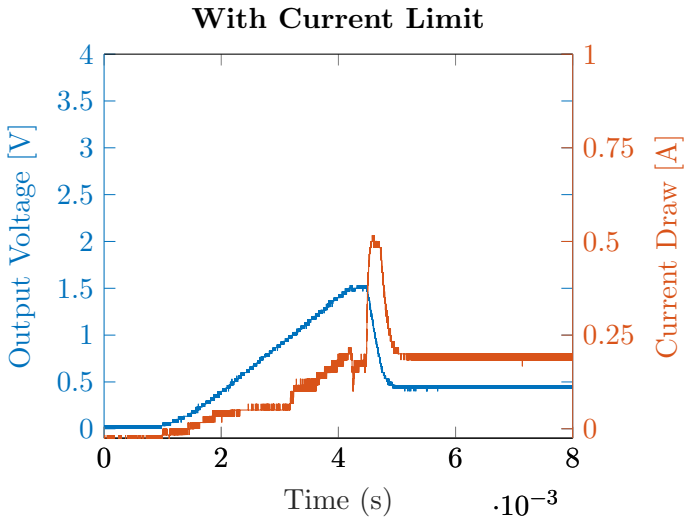


Figure 6.8: Voltage of the 5V rail and current draw of the joint board during startup with current limit of ≈ 250mA. Note that the left axis shows the output voltage while the right axis shows the current draw.

✓ Verify that 5V is present on `MISO_H`.

Verify RS422 Circuitry Functionality

- Apply 5V to `A_N`, 0V to `A_P`.
  - ✓ Verify that 0V is present on `A_H`.
  - ✓ Verify that 0V is present on `A`.
- Repeat for signal pairs `B_N`, `B_P` and `Z_N`, `Z_P`.
- Apply 0V to `A_N`, 5V to `A_P`.
  - ✓ Verify that 5V is present on `A_H`.
  - ✓ Verify that 3.3V is present on `A`.
- Repeat for signal pairs `B_N`, `B_P` and `Z_N`, `Z_P`.

Verify ATtiny84 Functionality

- Load the `ATtiny` with the blink software, available in the associated git repository.

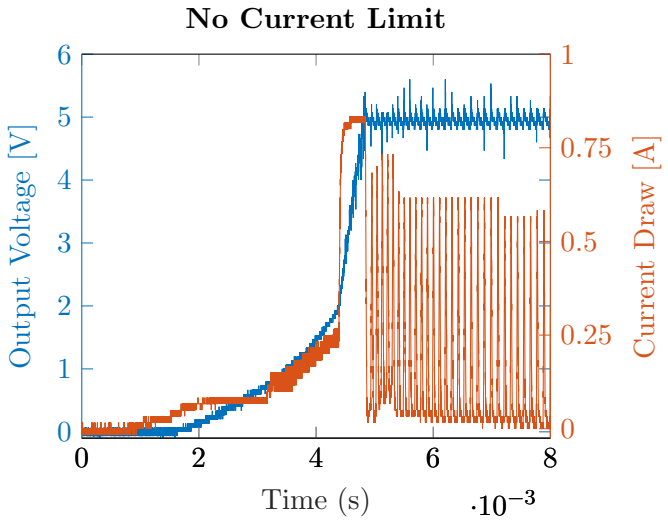


Figure 6.9: Voltage of the 5V rail and current draw of the joint board during startup with current limit of $\approx$ 1A. Note that the left axis shows the output voltage while the right axis shows the current draw.

✕ Verify that the on-board LED is blinking

- The software was loaded successfully occasionally but would often fail. This was found to be due to missing pull-up on the `RESET` pin. A pull-up resistor of 3.92kΩ can be placed on signals `RESET_H` and `5V`. The pull-up resistor cannot be placed on the `ATtiny` since the `74LCX244` and the `ATtiny` would otherwise both attempt to drive the `RESET` net.

6.6.2 Verification of: Requirement 30

This requirement states that the pendulum assembly should allow movement only in the X-Y plane.

Conclusion
When the pendulum assembly was mounted on the cart it was found that some movement in the Z plane was possible. By actuating the cart it was found that it is possible for the joints to collide, which is clearly problematic. As described the authors only have small knowledge in the world of mechanical engineering and it was decided to omit further work on the problem. Instead it is proposed that a future developer with more mechanical

knowledge should seek to find a solution to the problem.

6.6.3 Verification of: Requirement 31

This requirement states that 3.3V and 5V rails should be available for the components in the circuit.

Conclusion
The presence of the two voltage rails are verified in the procedure shown in section 6.6.1. Furthermore the mean value of the two rails were measured over a period of 25ms, yielding 3.27V and 5.07. This is within the required values.

6.6.4 Verification of: Requirement 32

This requirement specifies that joint should be able to rotate freely about the mounting axis.

Conclusion
This functionality was verified.

6.6.5 Verification of: Requirement 33

This requirement specifies that the `nRFM` should be included for communication.

Conclusion
The `nRFM` was included in the joint board PCB and wired to the `ATtiny`.

6.6.6 Verification of: Requirement 34

This requirement specifies that the `RL2IC` encoders RS422 interface should be correctly interfaced.

Conclusion
The RS422 receiver `DS26LS32CM` was placed on the joint board PCB to decode the signals from the `RL2IC` encoder. In section 6.6.1 the functionality of the `DS26LS32CM` is verified.

6.6.7 Verification of: Requirement 35

This requirement specifies that the design should allow programming of the ATtiny using 5V SPI.

Conclusion
Programming the ATtiny using 5V SPI was made possible using voltage level shifting. While developing software for the ATtiny it was revealed that the 74LCX244 and ATtiny both attempt to drive the SCLK and MOSI nets, resulting in the communication with the nRFM not functioning. This is a design error and should be fixed in later versions of the design. A temporary fix was done by soldering a wire to the enable pin of the 74LCX244 and connecting it to a switch connected to GND and 5V, effectively creating a programming mode and a running mode. In the programming mode the 74LCX244 is enabled and has control over the affected nets. In running mode the 74LCX244 is disabled and the ATtiny is capable of driving the affected nets.

6.6.8 Verification of: Requirement 36

This requirement states that an LED should be employed for debugging purposes.

Conclusion
An LED is present on the joint board PCB.

6.6.9 Verification of: Requirements 37, 38, 39, 40, 41, 42, 43, 44 and 45

All of the above requirements concern the design of the pendulum assembly and the joints.

Conclusion
They have all been followed throughout the design of the pendulum assembly.

6.6.10 Verification of: Requirement 46

This requirement states that overall power consumption of the design should be minimized.

Conclusion
In section 6.1.3 it was found that the circuit would require ≈462mW when transmitting. In testing it was found that the circuit draws 117mA at 3.7V or 432.9mW. Since the original numbers were calculated based on maximum values from the datasheets of the components used in the calculation it is reasonable to see a slightly lower number.

The battery chosen for this application has 1600mWH of capacity resulting in a theoretical run-time of ≈3:45 hours. This is clearly not as high as what was desired (10 hours). The main reason for this is the large power draw of the encoder and the accompanying RS422 decoder. The `DS26LS32CM` RS422 decoder draws 52mA at 5V. The Rolin Encoders are available in versions which output TTL signals meaning that the decoder could be removed. Unfortunately, the encoders with this output do not have a unique reference mark and as such it would not be possible to properly calibrate the joint.

7 Joint Board Software Development

Throughout this section the software for the joint board is developed. Initially an analysis is done to determine the required functionality of the software. The development of the wireless communication is designed and implemented and finally the implemented functionality is verified.

7.1 Analysis

The main responsibility of the joint software is to maintain knowledge of the angle of its joint and transmit that angle to the controller board. The following analysis will elaborate on the needed functionality and requirements for the software.

7.1.1 Joint Angle

The angle of the joint can be determined by analysing the signals from the `RL2IC` encoder mounted on the joint. This encoder implements incremental quadrature with three signals: `A`, `B` and `Z`. It is necessary to read all three signals using interrupts as missing a cycle in the counting procedure will cause the measured angle to drift. `Z` only goes `high` one time per revolution and can therefore be used to infer the absolute angle of the joint. Calibrating the joint angle is done by setting the measured angle to a fixed number, whenever `Z` goes `high`. Therefore data on angular position is not valid until the joint has been calibrated.

Since the angle is to be transmitted there is a risk that an interrupt may occur while this transmission takes place, potentially corrupting the data to be transmitted. This needs to be avoided and can be done by temporarily disabling interrupts and copying the needed data to other variables. The code that temporarily disables interrupts and copies the variable needs to be executed faster than two consecutive interrupts can occur, otherwise information on joint movement is lost. The time between two consecutive interrupts is only dependent on the angular velocity of the joint. It is estimated that the maximum angular velocity is $\approx$20Hz, which means that no interrupts should go unnoticed at this frequency. From visual inspection of graphs in [16] the maximum angular velocity of that system during swing-up is $\approx$2Hz. While the maximum velocity is influenced heavily by the setup in general, this is still well below the aforementioned 20Hz and this estimation is considered reasonable.

7.1.2 Wireless Transmission of Data

As mentioned in section 5.1.4, communication between the controller board and the joint boards is to happen wirelessly using the `nRFM` and a period between packets of more than 130μs. The software needs to communicate with the `nRFM` through an SPI connection with the `ATtiny` as the master. The `nRFM` has a number of options that needs to be set according to the desired functionality. As described in 5.1.4, it was found that the two joints need to send using two separate frequency bands to avoid air collisions. This needs to be set up individually in each `nRFM`. Latency should be minimized by reducing overhead in transmitted packages and sending the smallest possible payload that correctly conveys the joint data. The datasheet describes that the `nRFM` can operate at air rates of 250kbps, 1Mbps and 2Mbps. Based on the somewhat unstable transmission explained in section 7.3.4 it was chosen to use a transmission rate of 1Mbps, as it is more noise resistance than when using 2Mbps.

Transmission Rate
The datasheet of the `nRFM` module specifies that there is a settling time of 130μs when entering the transmit state. Several aspects should be considered in order to determine the required period time associated with transmission of one data packet. An overview is given in figure 7.1. Transmission of a two byte payload through SPI takes $\approx$ 36 μs as will be shown in 7.3.3. The `CE` pin should be `high` for at least 10μs in order for the transition to transmit mode is initiated. Hereafter a settling time of 130μs occurs. Then the `nRFM` will transmit the two byte payload in an eight byte data packet, which will take 68μs using an air transmission rate of 1Mbps. This totals 244μs, in which the computation time needed is not included. Therefore it was decided to use a period time of 333μs, which yields 84μs to computation and a sampling frequency of $\approx$3kHz. A sample latency of at least 249μs is incurred on the joint board software including air transfer before the data is available to the controller board nRFM.

7.1.3 Real-Time Software

In order to produce reliable data for the controller board, the joint angle should be sampled and transmitted at a constant rate. This requires the joint board software to be programmed as real-time software, where deadlines should be met.

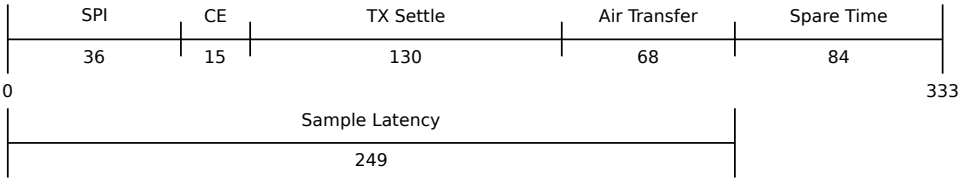


Figure 7.1: Period associated with transmission of one data packet. SPI refers to the SPI transmission, CE refers to the time the `CE` pin is set `high`, TX Settle is the TX settling time, Air Transfer is the time it takes for the air transfer of the packet and Spare Time is the leftover time.

7.2 Requirement Specification

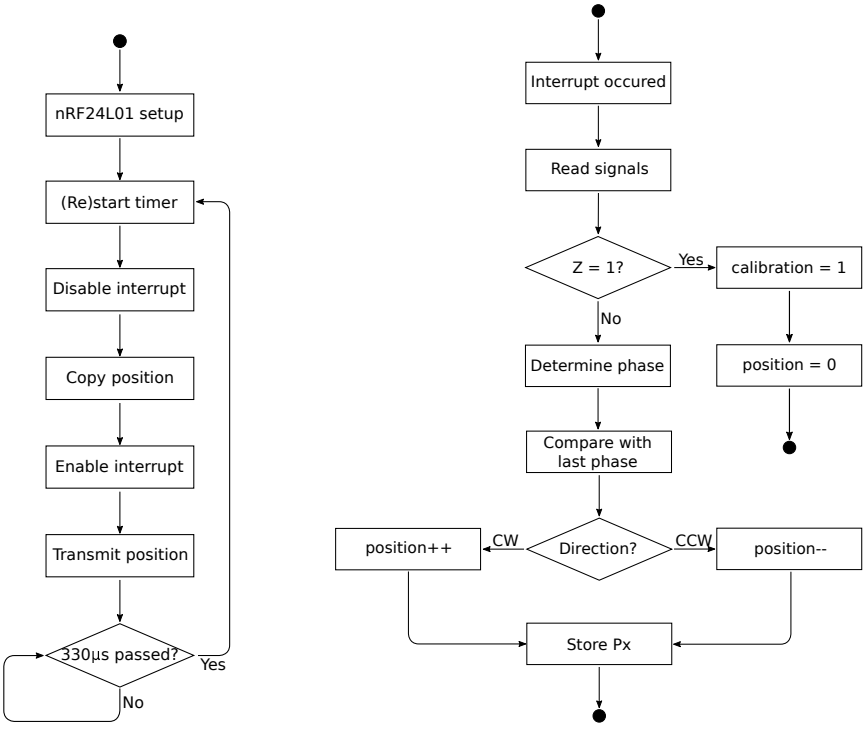
The requirements specified below are tested and verified in section 7.4.

Functional:

47. Correct joint angle should be known at all times.
    - Encoder signals `A`, `B` and `Z` should be interfaced using interrupts.
    - `A`, `B` should be used to measure the relative angle.
    - `Z` should be used to infer the absolute angle.
    - Determine movement and direction of movement based on encoder signals.
    - Ensure no data is corrupted by interrupts.
    - Ensure no interrupts lost at angular frequencies below 20Hz.
48. Transmit joint angle using the `nRFM`.
    - `ATtiny` should be setup as SPI master.
    - Transmit data every 333μs.
    - Minimize latency.
49. Software should be real-time.

7.3 Design and Implementation

A software design was made based on the requirement specification for the joint board software. The main functionality of the software is illustrated in the flowchart of figure 7.2a. Upon startup the `nRFM` is setup as explained in section 5.3.2. A timer is initiated to ensure that a payload, as shown in



(a) Main loop. **(b)** Encoder signal Interrupt Service Routine.

Figure 7.2: Flowchart of the software design for the joint board.

figure 7.3, is sent every 333μs. Now the copying procedure is started after which the packet is transmitted and the program is idling until 333μs has passed and the process starts over. An interrupt can happen at any given time and the ISR, will update the measured joint angle. The coming sections will elaborate on the different elements of the software.

7.3.1 Joint Angle Measurement

In figure 7.2b the interrupt routine is shown. This routine can be engaged at any given time and is engaged whenever a rising or falling edge appears on `A`, `B` or `Z`.
Initially the signals are read. If `Z` is `high` the index has been reached and the position counter is reset and the calibration bit is set. Hereafter the execution of the main loop is continued. If `Z` is `low`, `A` and `B` are inspected to determine which phase is active. Table 7.1 shows each phase and the combination of `A` and `B` in that phase. By comparing with the previous

	A	B
P1	0	1
P2	0	0
P3	1	0
P4	1	1

Table 7.1: Binary representation of the phases.

phase it is possible to determine the direction of movement and therefore whether the position counter should be incremented or decremented. Before returning execution to the main loop the current phase is stored.

7.3.2 Wireless Transmission of Data

The joint board should transmit the angular position of the joint and whether calibration has occured since startup. The `RL2IC` encoder used in the joint produces 7200 counts per revolution, requiring 13 bits to represent a full revolution of the joint. The calibration status is binary and represents one bit. 14 bit is required to transmit this information, but since the `nRFM` payload needs to be an integer number of bytes 16 bit are transmitted. See figure 7.3 for a visual representation of the message.

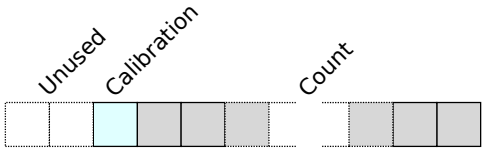


Figure 7.3: Sructure of a payload used on the RF transmission between the joint boards and the controller board. Bits 0-12 represent the current joint angle and bit 13 represents the state of calibration.

When information is to be transmitted the calibration and angle variables needs to be copied to other variables. The act of copying a variable takes several clock cycles and if an interrupt occurs while the variable is being transmitted the data can be corrupted. In order to avoid this problem it is necessary to temporarily disable interrupts while copying the original variable into a temporary variable. In effect the resulting code is as seen in listing 7.1. Disabling interrupts means that any incoming interrupts will not be processed until interrupts are reenabled, as described in the datasheet of the `ATtiny` [4]. Interrupts should therefore not be disabled for longer than the shortest expected time between two edges on any single signal.

The AVR-GCC compiler has the option to only compile the code, leaving a human readable assembly file. The assembly code corresponding to listing 7.1 can be seen in listing 7.2. Each of these instructions are described in the datasheet of the microcontroller where the number of cycles required to execute them is specified: rcall 3 cycles, _CLI 1 cycle, ldd 2 cycles, std 2 cycles, _SEI, 1 cycle. In total 16 clock cycles are spent executing the required commands. At 8MHz this is 2 μs. Since the RL2IC produces 7200 ticks per revolution only 3600 edges exist on a single signal per revolution. Using these numbers the theoretical maximum angular velocity possible while still maintaining more than 2 μs between each edge on a signal is ≈135Hz, which is clearly above the estimated maximum velocity of ≈20Hz.

```
1     _CLI();
2     cnt_temp=cnt;
3     _SEI();
```

Listing 7.1: Critical section for copying counter value. C version.

```
1     rcall _CLI
2     ldd r24,Y+1
3     ldd r25,Y+2
4     std Y+4,r25
5     std Y+3,r24
6     rcall _SEI
```

Listing 7.2: Critical section for copying counter value. Assembly version.

7.3.3 SPI

The ATtiny does not have an SPI controller, but it does have a Universal Serial Interface, USI, which is compatible with SPI, when it is used in its three wire mode. In this mode the interface has the pins, Data Out DO, Data In DI and a clock USCK. In order to comply with the SPI standard a Slave Select SS pin should be implemented using a general IO pin and custom written software. The nRFM also has a non SPI pin, CE that also needs to be controlled from the ATtiny. The full interface between the two is shown in figure 7.4.
The ATtiny is used as the SPI master and controls MOSI, SCLK, SS and CE on the nRFM.
With the two modules interfaced correctly, data can be transmitted between them. The function shown in listing 7.3 implements this functionality on

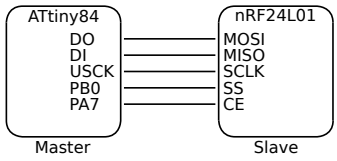


Figure 7.4: Interfacing the ATtinys USI and nRFMs SPI. PB0 and PA7 are general purpose IO pins.

the ATtiny. It takes pointers to data, input and the number of bytes that needs to be sent as inputs. Line 2 sets SS low to initiate an SPI transfer. The loop in lines 4 to 11 is run n times to send and receive n bytes. In line 5 the USI Data Register USIDR is loaded with the byte to be sent and in line 6 the counter overflow flag is cleared. Lines 7 to 9 writes 1 to USICR until the counter overflow flag USIOIF is raised. The USI clock USCK is toggled each time 1 is written to USITC. Data is shifted to DO by the USI by counting the clock edges generated. Data on DI is sampled at falling edges of the USCK and stored in USIDR. The counter overflow flag USIOIF will raise when 16 clock edges has been generated and eight bits of data has been sent and received. Line 10 copies the received data to the input pointer if there is one. When n bytes of data have been sent and received the SS pin will be set high.

```
1  void spi_transfer(uint8_t *data, uint8_t *input, uint8_t n){
2    PORTB &= ~_BV(SS);
3    int i;
4    for(i = 0; i<n; i++){
5      USIDR = data[i];
6      USISR = _BV(USIOIF);
7      while ( !(USISR & _BV(USIOIF)) ){
8          USICR |= _BV(USITC);
9      }
10     if(NULL |= input){input[i] = USIDR;}
11   }
12   PORTB |= _BV(SS);
13 }
```

Listing 7.3: Function that transmits n bytes of data between the ATtiny and the nRFM.

This functions enables setup of the nRFM, which is done with the same settings as explained in section 5.3.2. The frequency of the SPI clock is ≈675kHz and transmission of one byte takes 12μs.

7.3.4 Development of Wireless Transmission

During implementation of the software for the nRFM a lot of testing was done to verify that the developed code worked. Initially two nRFMs were wired up to two Arduinos running example code from the arduino community libraries. This was done to verify wiring and functionality of the nRFMs.

Afterwards an Arduino was programmed to receive data and the software was written to the ATtiny to transmit data using the nRFM. The functionality of the joint board software was verified by transmitting data from the ATtiny to an Arduino successfully. Afterwards an nRFM was wired to the MicroZed and the software was written.

The transmission of data between the ATtiny and the MicroZed was verified by transmitting data from the joint board nRFM to an nRFM in free air connected to the MicroZed. The transmission was successful.

The same software was used to test the nRFM wired on the controller board. Tests showed that no data was received in this setup. Thorough inspection of the schematic and PCB revealed no errors. Further testing with a nRFM in free air connected to the MicroZed showed that no packets were received when the controller board PCB was in near proximity of the nRFM. The problem is due to the large amount of copper in the controller board PCB. The error was fixed by desoldering the nRFM and wiring it with longer wires.

Further investigating the datasheet of the nRFM revealed that it is recommended to make a cutout in the ground plane for the PCB antenna. It is also possible to add an SMA connector instead and use an external antenna for added signal power. Both of these require design of a new version of the PCB.
The joint board did function but it is likely that signal integrity will be increased by also making a cutout on this board to accomodate the PCB antenna.

7.3.5 Software Timer

The timer shown in figure 7.2a, is implemented on the ATtiny using Counter 1, which is a 16 bit timer capable of generating software interrupts. Counter 1 is setup by setting a prescaler to the CPU clock and a compare value. Using a prescaler of 1, the compare value for setting up an interrupt each 333μs is

calculated as:

$$Comp = \frac{T}{T_{cpu}} = \frac{333 \cdot 10^{-6}}{1.25 \cdot 10^{-7}} = 2664 \tag{7.1}$$

Where $Comp$ is the compare value, T is the wanted period time and T_{cpu} is the CPU clock period. Setting up `Counter 1` with this compare value and the flag to clear the counter when reaching the compare value, results in interrupts on `TIM1_COMPA_vect` each 333μs.

```
1 volatile char timer;
2
3 ISR(TIM1_COMPA_vect)
4 {
5   timer = 1;
6 }
```

Listing 7.4: Counter `ISR` function and declaration of `timer`.

The ISR function associated with these interrupts is shown in listing 7.4 together with the declaration of `timer`. The variable is declared as `volatile` to tell the compiler that its value can be changed whenever during run-time. It is declared as a `char`, because this is the smallest datatype available on the `ATtiny` and it only takes up one byte. In the main loop of the software, the `timer` variable is used determine if the process of transmitting data should be initiated. This mechanism provides real time performance of the software when the code used to transmit data is executed before the next deadline, which is the next `counter 1` interrupt. This means that the transmitting code needs to be executed in less than 333μs to ensure real-time performance.

```
1 while(1){
2     if(timer==1){
3         timer = 0;
4             ⋮
5         // Transmit data
6             ⋮
7     }
8 }
```

Listing 7.5: Main loop of the software. The `timer` variable is used to transmit data at a fixed frequency.

In lines 2 and 3 of listing 7.5 the `timer` variable is accessed. Accessing a variable that is written to in a `ISR` can result in corrupted data, but in this

case it is not a problem because `timer` is a one byte variable. The two assembly instructions that will be used to access the value of `timer` is `ldi` and `cpi`, which are both instructions that will be executed in one clock cycle and can therefore not be interrupted.

7.4 Verification

Throughout this section the verification of the requirements of the joint board software is done.

7.4.1 Verification of: Requirement 47

This requirement specifies that the correct joint angle should be known at all times.

Test
This cannot be verified without utilizing the wireless transmission between the joint board and controller as the only other output the jointboard has is an LED. A test should be conducted where the joint angle information is sent to the controller board and printed to a serial console.

Conclusion
The test was done and by visual inspection the functionality was verified.

7.4.2 Requirement 48

This requirement specifies that the joint angle should be transmitted using the `nRFM`.

Test
The two `nRFM`s should be facing each other with a short distance between them. Data should be transmitted from one of them to the other and it should be counted how many packets are received. This test should be conducted with the pendulum being held still and being moved to determine if movement affects the transmission.

Conclusion
The test was conducted by letting the two `nRFM`s face each other with a distance of ≈10cm. A 32 byte payload was sent from the joint board every 1ms. The payload includes an ID, which is incremented for each packet.

The joint board was programmed to send 10000 packets before stopping transmition.
In the first series of tests the pendulum was held still. The number of received packets can be seen in the **S** row of table 7.2. The average number

#	1	2	3	4	5	6	7	8	9	10	Avg
S	7046	7245	7272	7241	7232	7212	7160	7217	7241	7221	7208
M	7148	7215	7208	7173	7221	7165	7138	7187	7274	7186	7191

Table 7.2: Number of received packets out of 10000 transmitted. The **S** row represents the received packets while the pendulum was held still and the **M** row represents the received packets while moving the pendulum.

of received packets is 7208 which corresponds to $\approx 72\%$ of the transmitted packages.

The same test was then conducted, but with the pendulum moving back and forth. The number of received packets can be seen in the **M** row of table 7.2. The average number of received packets is 7191 which corresponds to $\approx 72\%$ of the transmitted packages.
By looking at the two test results it can be determined that movement of the pendulum has no significant effect on the success rate of transmission.
In order to determine if there was a pattern in how the 28% of packages were lost the IDs of the received packages were plotted. The ID of a package corresponds to the number of milliseconds passed since program start. No pattern was successfully identified by visual inspecting. An excerpt of the data is shown in figure 7.5.

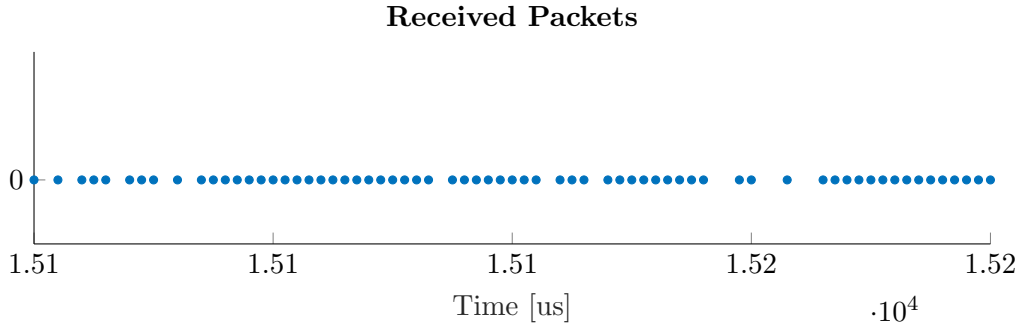


Figure 7.5: IDs of received packages plotted. ID corresponds to the number of milliseconds passed since program start. No pattern was identified.

7.4.3 Requirement 49

This requirement specifies that the software must exhibit real-time behaviour, which means that the software must meet its deadlines.

Test
It should be tested if the software meets its deadline, which is the next transmission of a packet. This means that it should be shown that the software executes everything related to sending a packet within the period of 333μs. This can be tested by setting an output pin `high` as the first part of the code responsible for the transmission and setting it `low` as the last line of code. It should be verified that the `on` time of this pin is less than 333μs.

Conclusion
The described test was performed with the output pin and SPI clock measured by an oscilloscope. The measured signals are shown in figure 7.6.

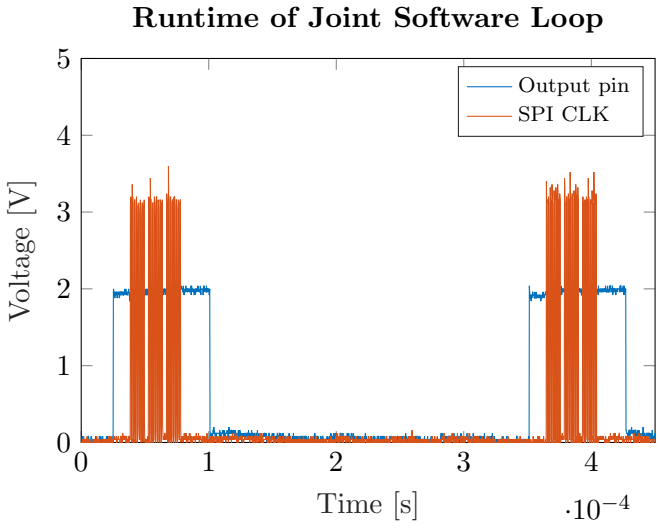


Figure 7.6: Measured signals associated with the runtime of the joint board software.

It can be seen that the `on` time of the output pin is ≈75μs, which leaves 258μs before the deadline. It should be noted that following the SPI transmission another 213μs are required for the `nRFM` to go to TX mode and transmit the data.

8 System Verification

8.1 Requirement Verification

This final verification evaluates upon the initial requirements of the system found in the system analysis.

8.1.1 Verification of: Requirements 1 and 2

These requirements specify that the system should consist of a double pendulum mounted on a moveable cart and that the cart should be actuated by a motor.

Conclusion
The implemented system consists of the designed and implemented pendulum assembly mounted on the rail system inherited from an earlier project. A picture of the full system setup is shown in figure 8.1 and a picture of the controller board interfaced with the full system is shown in figure 8.2. Specific requirements for designing and implementing the pendulum assembly are described and verified in section 6.
The cart can be actuated by the Maxon motor as they are connected by a belt. Even though the cart was never moved using the motor due to debugging troubles and time constraints, it was tested that the developed controller board is able to drive the motor. Since no actual control was done, it is still unknown whether the cart and rail system is sufficient for this application.

8.1.2 Verification of: Requirement 3

This requirement states that the pendulum system should be controlled by a MicroZed.

Conclusion
The controller board has been designed to fit the MicroZed and all relevant signals were wired to it as described in section 4.

In section 4.6.2 it was verified that the MicroZed can control motor speed and direction through implemented VHDL modules, the motor driver and the H-bridge on the controller board.

Figure 8.1: Picture of the full system.

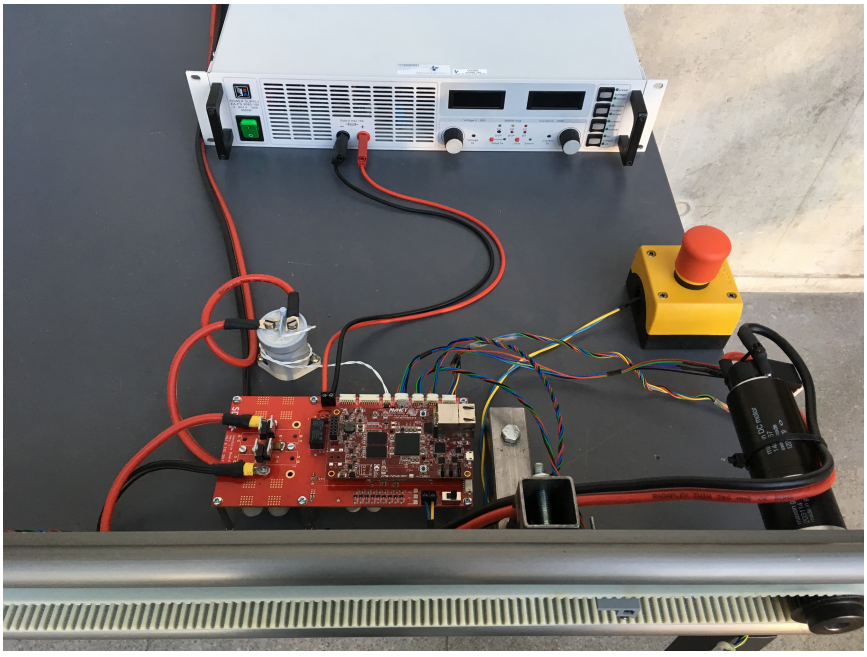


Figure 8.2: Picture of the controller board interfaced with the full system.

In section 4.6.3 it was found that it is not possible to determine the motor current using a shunt resistor and the `INA286`. Thus it is not possible to measure the motor current using the ADC on the MicroZed.

In section 4.6.6 it was verified that the MicroZed is interfaced correctly by measuring start-up and shut-down sequences.

8.1.3 Verification of: Requirement 4

According to this requirement the position of the cart and joint angles should be measured.

Test
In section 5.4.1 it is tested and verified that the position of the cart is measured correctly.

In section 7.4.1 it is verified by visual inspection that the joint angles are correctly measured on the joint board.
To further verify that the measurement is correct a test should be conducted where the joint angle data are stored and plotted. In a single pendulum setup, the joint should manually be moved $\frac{\pi}{2}$ radians from the equilibrium position and released, while collecting joint angle data.

Conclusion
The test described above was conducted while transmitting data at 1kHz, with the two `nRFM`s pointing at each other. An excerpt of the corresponding data is shown in figure 8.3 with the joint angle measured in ticks. A full rotation is equal to 7200 ticks. As expected the waveform is a damped sinusoidal. One period is shown in figure 8.4. It can be seen that some angles are missing, which is as expected based on the tests described in section 7.4.2, where it was concluded that 28% of data packets are lost. From figures 8.3 and 8.4 it can be verified that the joint angle are measured correctly as all data is as expected.

The full test was only realised on one of the joint boards, as only one board was finished due to time constraints.

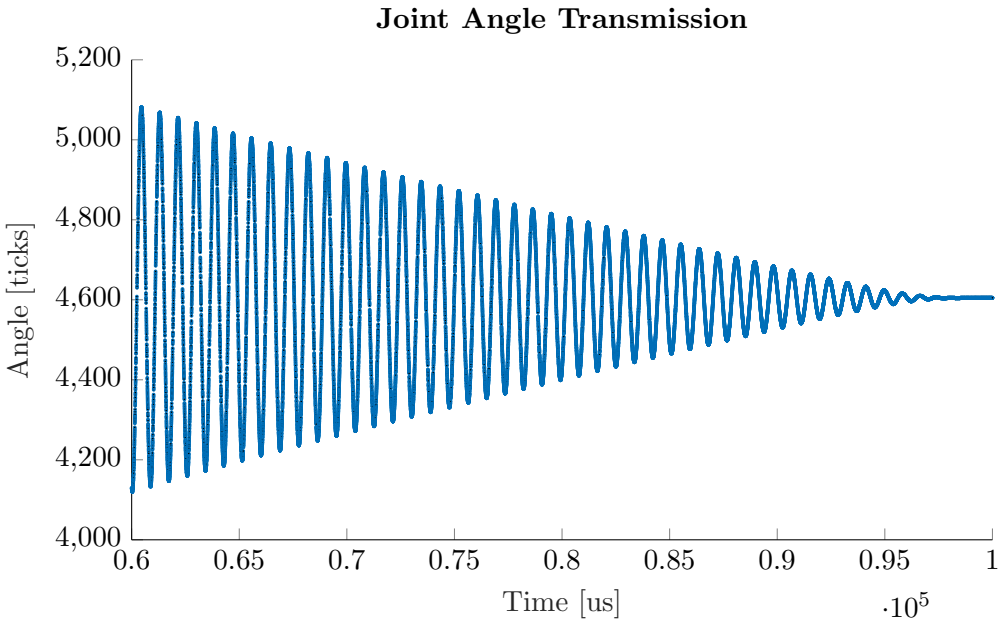


Figure 8.3: Excerpt of joint angle data. A full rotation is equal to 7200 ticks.

8.1.4 Verification of: Requirement 5

This requirement specifies that any software developed for the pendulum system should be real-time.

Conclusion
In section 5.4.5 it is described that due to time constraints the full real-time system is not realised. A number of smaller programs were written in order to verify the functionality of individual components.

In section 7.4.3 it is tested and verified that the joint board software is real-time.

8.1.5 Verification of: Requirement 6

This requirement states that the system should be easily accessible to users.

Conclusion
Throughout the design of the system, accessibility has been a focus. Section 4.6.11 describes how the PCB has been desgined with debugging in

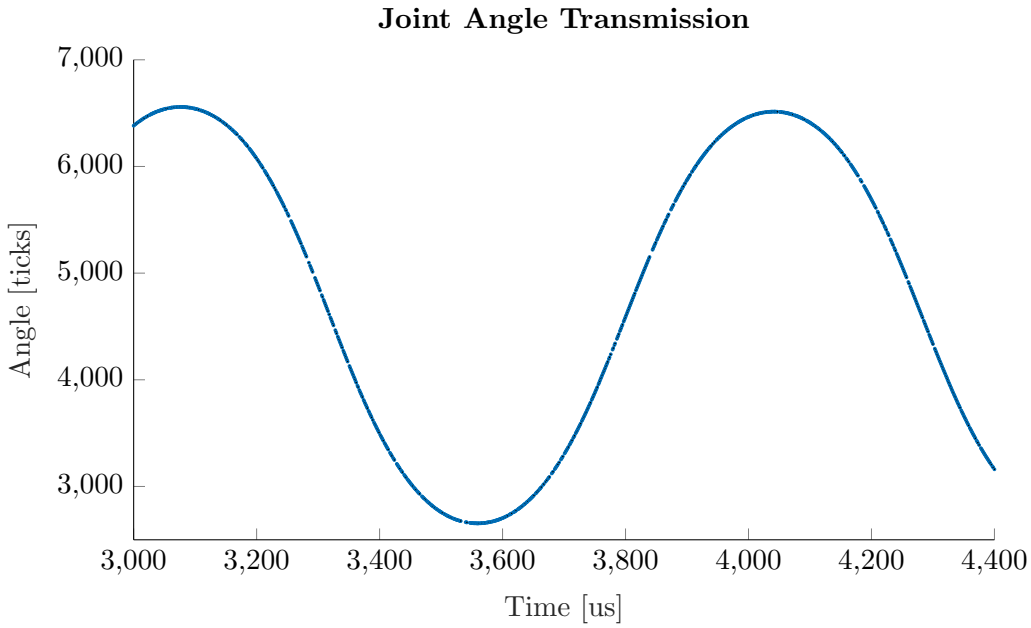


Figure 8.4: One period joint angle data. A full rotation is equal to 7200 ticks.

mind. Verification procedures have been developed for the joint and controller boards and is included in the report as appendix A and B.

The usability of the controller board should be verified by using students as test subjects once the system is finished.

8.1.6 Verification of: Requirement 7

This requirement specifies that the system should not break during operation by users.

Conclusion
The design of the controller board electronics is done with the requirement that the components should be able to handle the stall current of the motor. Section 4.6.14 verifies that the implemented endstops can cut off power to the H-bridge effectively stopping the motor.

8.1.7 Verification of: Requirement 8

According to this requirement the system safety should not rely on the programming of the system.

Conclusion
Signals from endstops and emergency button cuts off power to the H-bridge through digital electronics. The functionality is verified in section 4.6.14.

A test should be conducted where the cart is driven towards the endstop. It should be verified that the cart is stopped by the safety circuit before crashing into the motor. The test should be conducted again, but with a higher velocity of the cart. In this way the maximum allowed velocity can be determined or it can be determined that the endstops should be moved to another place on the rail.

8.1.8 Verification of: Requirement 9

This requirement states that software design should be done top-down, whereas subcomponents are written bottom-up.

Conclusion
This design procedure was followed while designing software for the joint and controller boards.

9 Conclusion

All of the requirements have been concluded upon throughout the report and will not be repeated in full for this discussion. Instead an overview of some of the accomplishments and shortcomings of the project will be presented. The goal of the project was to develop a cart-mounted inverted double pendulum for experimenting with underactuated systems. In order to do this an overview was gathered on the applications of underactuated systems, especially of similar pendulum systems.
The general design method used throughout the report is based on a workflow of analysis > design > implementation > verification. Adhering to this system not only provided the authors with a better design, but also with better documentation throughout the process.

Two PCBs were designed, the controller board and the joint board. For the former a motor driver was designed and the power delivery of the MicroZed, designed by the authors in a previous project, was added to enable the use of a Zynq-7 series SoC for the main computational platform. The joint board uses wireless communication in order to transmit the joint angles to the controller board.
These PCBs were designed following design steps determined by the authors. It is believed that following these steps greatly decreased the number of errors in the final design. They were, however, by no means error-free and would both benefit from a second iteration. The errors that are present on the boards can be either easily fixed or an acceptable work-around has been found which can be used until a second iteration of the PCBs is designed.
The `nRF24L01` module was used to do the wireless communication between the joint and the controller board. It was found that it had a far larger protocol than initially assumed but still had sufficient throughput to transmit the required positions.
The mounting of the module was found to not be in accordance with the recommendation of the datasheet. While this issue was worked around simply by moving the module away from the controller board, this is possibly the component most urgently in need of a redesign.

The intention of the authors was to design a real-time software system based on KHAos to show that the system is functioning as desired. As has been explained previously, this goal was unfortunately not met. Small pieces of software have been written to verify the functionality of individual components but a combined system was never written due to time constraints.

A number of features were implemented in VHDL to leverage the FPGA in the Zynq chip. The authors purposefully implemented every VHDL component with an accompanying testbench before packaging it as an IP core. This allowed for faster iterations when debugging VHDL code.

Data was gathered from the joints and is presented in the report. A number of difficulties arose with the debugging of the full-bridge, but the issue, or at least a symptom of it, was eventually tracked down and a workaround determined. The motor of the cart system has been driven by the designed motor driver.
Current measurement was not accomplished due to the noisy environment of the amplifier of the current signal.

Finally a safety system was implemented which is completely independent from any programming. This safety system is capable of cutting power to the motor using a relay whenever endstops or an emergency button are activated. In addition to these the software is capable of generating an emergency which will cut power to the motor. Power is maintained to the computational platform during an emergency so that the cause of the emergency can be recorded and investigated.

This concludes the work done in this project.

9.1 Future Work

As is customary with projects, this one did not fully accomplish what the authors originally intended. This section will go through some of the work required to correct mistakes done by the authors as well as present some of the features that may be of interest for future developers.

9.1.1 Corrections

A number of mistakes, mostly regarding the PCB design, were uncovered throughout the duration of the project. Both the controller board and the joint board, while functional, would benefit from a second iteration to correct these mistakes.

Perhaps most notably the 4ms error uncovered in section 4.6.2 requires a thorough investigation so that the root cause of the issue can be found and corrected.

9.1.2 Linux Implementation

The intent with the system is to make it controllable from Linux. This can be done using different methods. A kernel driver could be written such that a user can send commands to and receive data from the system simply by issuing read/write commands on a `tty` interface using userspace software.

Another approach involves setting up parallel processing on the two cores of the MicroZed. One core running Linux and another maintaining communication with the inverted pendulum system. Communication between the two can be done using direct memory access, DMA. Using this method the strictly real-time tasks can be done on a bare-metal system.

9.1.3 Determination of System Parameters

One interesting aspect of the pendulum system is model development. By experimentation the parameters of the system such as the friction and inertia of the joints can be determined. This would allow the creation of an accurate model of the developed system and open up for accurate simulations of the actual system.

9.1.4 Swing-up of the Pendulum

One obvious short-coming of the work done in this report is the lack of control of the pendulum. The system is believed to be sufficiently functional to do proper control of a single pendulum. In order to verify this assumption the software written in this project should be collected into a complete system and a small control loop should be written in order to do the actual swing-up.

9.1.5 Double Pendulum

The focus throughout this report has been to develop a double pendulum system, but due to different problems the finished system does need an amount of work before a double pendulum system on a cart can be controlled.
During analysis of the `nRFM` functionalities it was found that in order to create reliable air transmission, two pairs of `nRFM` are required. The second `nRFM` needs to retrofitted to the controller board before data can be received from two joints simultaneously.

Furthermore it was found that the rod and joint mount, made of carbon fiber and 3D printed plastic repectively, was not sufficiently stiff to avoid excessive movement along the Z-axis of the system. The authors invision an ISA (Individual Study Activity) for a mechanical engineering student who might be able to bring knowledge of material properties which the authors do not possess.

References

[2] *7 Series FPGAs and Zynq-7000 All Programmable SoC XADC Dual 12-Bit 1 MSPS Analog-to-Digital Converter User Guide*. Xilinx. Sept. 2016.

[6] Daniel Jerome Block. "Mechanical Design and Control of the Pendubot". In: (1996).

[7] Marvin Bugeja. "Non-Linear Swing-Up and Stabilizing Control of an Inverted Pendulum System". In: (2003).

[8] R. Paul Clayton. *The Concept of Partial Inductance*. Wiley, 2010.

[10] *Design and Application Guide of Bootstrap Circuit for High-Voltage Gate-Drive IC*. ON Semiconductor. Dec. 2014.

[14] Bin Gao et al. "Design and analysis of underactuated robotic gripper with adaptive fingers for objects grasping tasks". In: (Dec. 2016).

[15] T. Glück, A. Eder, and A. Kugi. "Swing-up control of a triple pendulum on a cart with experimental validation". In: *Automatica* 49.3 (2013), pp. 801–808.

[16] Knut Graichen, Michael Treuer, and Michael Zeitz. "Swing-up of the double pendulum on a cart by feedforward and feedback control with experimental validation". In: (July 2006).

[22] IPC – Association Connecting Electronics Industries. *Standard for Determining Current-Carrying Capacity In Printed Board Design*. IPC 2152. Norm. 2009.

[23] IPC – Association Connecting Electronics Industries. *Generic Standard on Printed Board Design*. IPC 2221. Norm. 2003.

[24] D. S. Hoynes. *CHARACTERIZATION OF METAL-INSULATOR LAMINATES*. IPC 2221. Norm. National Bureau of Standards. 2003.

[25] Mikkel S. Jaedicke and Thomas S. Christensen. *Individual Study Activity - Developing the Swarmbot*. Tech. rep. University of Southern Denmark, 2017.

[26] Alek Kaknevicius and Adam Hoover. *Managing Inrush Current*. Texas Instruments, May 2015.

[27] Atsushi Kakogawa, Hiroyuki Nishimura, and Shugen Ma. "Underactuated Modular Finger with Pull-in Mechanism for a Robotic Gripper". In: (Dec. 2016).

[29] A. Merello, A. Rugginenti, and M. Grasso. *Using Monolithic High Voltage Gate Drivers*. International Rectifier.

[30] *MicroZed Carrier Design Guide, Version 1.5*. Avnet. 2017.

[33] Arjun Prakash. *Current Sensing in an H-Bridge*. Texas Instruments. Dec. 2016.

[34] Qorvo. *Trace Width Calculator*. `http://www.qorvo.com/design-hub/design-tools/trace-width-calculator`. Oct. 2017.

[35] Tim Regan et al. *Current Sense Circuit Collection*. Linear Technology. Dec. 2005.

[37] *RS-422 and RS-485 Standards Overview and System Configurations*. Texas Instruments, May 2010.

[38] Paul Scherz. *Practical Electronics for Inventors*. 2000.

[39] *Segway-PT*. Wikipedia. Dec. 2017. URL: `https://en.wikipedia.org/wiki/Segway_PT`.

[40] *Self-Balancing Scooter*. Wikipedia. Dec. 2017. URL: `https://en.wikipedia.org/wiki/Self-balancing_scooter`.

[41] dialog Semiconductor. *Application note PCB Layout Guidelines*. Oct. 2015.

[45] Viroch Sukontanakarn and Manukid Parnichkun. "Real-Time Optimal Control for Rotary Inverted Pendulum". In: (2009).

[46] Linear Technology. *Basic Concepts of Linear Regulator and Switching Mode Power Supplies*. Oct. 2013.

[47] Russ Tedrake. *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832)*. Dec. 2017. URL: `http://underactuated.mit.edu/`.

[49] Meta Tum et al. "Swing-up Control of a Single Inverted Pendulum on a Cart With Input and Output Constraints". In: (Sept. 2014).

[51] Vishay. *Power MOSFET Basics: Understanding Gate Charge and Using it to Assess Switching Performance*. Feb. 16.

[52] Wikipedia. *Bluetooth*. Dec. 2017. URL: `https://en.wikipedia.org/wiki/Bluetooth`.

[53] Xilinx. *7 Series XADC / UltraScale SYSMON - Analog Input Range*. `www.xilinx.com/support/answers/44954.html`. Accessed: 12-09-2017.

[55] Bryan Yarborough. *Components and Methods for Current Measurement*. Aug. 2015.

[56] Hank Zumbahlen. *Staying Well Grounded*. Analog. June 2012.

Datasheets

[1] *6-A, Wide-Input Adjustable Switching Regulator, PTN78020H*. Texas Instruments. Aug. 2011.

[3] *Aluminum Electrolytic Capacitors SMD (Chip)*. Vishay. Dec. 2016.

[4] Atmel. *ATtiny84, 8-bit Microcontroller with 2/4/8K Bytes In-System Programmable Flash*. 2010.

[5] Avnet. *MicroZed Zynq Evaluation Kit and System on Module Hardware User Guide*. 2017.

[9] Sipex Corporation. *SP6641B, 500mA Alkaline DC/DC Boost Regulator in SOT-23*. May 2005.

[11] EPCOS. *SMT power inductors*. June 2012.

[12] *FR-4 Datasheet*.

[13] *G6B Power Relay - High Capacity and High Dielectric Strength Miniature Relay with Fully Sealed Construction*. Omron.

[17] *HEDM-55xx/560x & HEDS-55xx/56xx Quick Assembly Two and Three Channel Optical Encoders*. Avago Technologies. Nov. 2014.

[18] *HIP4081A Datasheet FN3659.8*. Intersil. Sept. 2015.

[19] *INA28x High-Accuracy, Wide Common-Mode Range, Bidirectional Current Shunt Monitors, Zero-Drift Series*. Texas Instruments. May 2015.

[20] Texas Instruments. *TPS6222, 400-mA, 1.25-MHz, HIGH-EFFICIENCY, STEP-DOWN CONVERTER IN THIN-SOT23*. Sept. 2003.

[21] Texas Instruments. *TPS6300x High-Efficient Single Inductor Buck-Boost Converter With 1.8-A Switches*. Oct. 2015.

[28] *KILOVAC LEV100 Series 900Vdc Contactor*. Tyco Electronics. 2008.

[31] maxon motor. *maxon DC Motor - RE 40*. May 2017.

[32] *nRF24L01 Single Chip 2.4GHz Transceiver Product Specification*. Nordic Semiconductor. July 2017.

[36] *RLC2IC miniature PCB level incremental magnetic encoder sensor system*. RLS. Sept. 2017.

[42] Fairchild Semiconductor. *74LCX244 - Low Voltage Buffer/Line Driver with 5V Tolerant Inputs and Outputs*. Dec. 2013.

[43] STMicroelectronics. *LD3985, Ultra low drop and low noise BiCMOS voltage regulators*. Jan. 2014.

[44] *Strong IRFET IRF100B202*. Infineon. Aug. 2014.

[48] *Transmissive Optical Sensor with Phototransistor Output*. Aug. 2009.

[50] *US1M, Surface Mount Ultrafast Rectifier*. Vishay. Apr. 2014.

[54] Xilinx. *Zynq-7000 All Programmable SoC Data Sheet: Overview*. June 2017.

A Joint Board Verification Procedure

This document holds the verification procedure which, when applied succesfully, ensures the correct functionality of the joint board. Each step is to be followed in the order they are shown to avoid previous faults to propegate throughout the process. A step consists of an action and the expected outcome of that action. Any text written with the `teletype font` refers to signals and pins on the PCB and can be seen on the schematic. The most recent schematic can be seen appendix D.

Verify Voltage Rails

- Apply 3.7V to `VBATT`.
  - Verify that current draw is within reasonable limits (<75 mA). Note that the PCB requires a short pulse of approximately 0.89A while booting.
  - Verify 5V and 3.3V voltage rails.

Verify Voltage Level Shifter Functionality

- Apply 5V to `SCLK_H`.
  - Verify that 3.3V is present on `SCLK`.
- Repeat for signals `MOSI_H` and `RESET_H`.
- Apply 3.3V to `MISO`.
  - Verify that 5V is present on `MISO_H`.

Verify RS-422 Circuitry Functionality

- Apply 5V to `A_N`, 0V to `A_P`.
  - Verify that 0V is present on `A_H`.
  - Verify that 0V is present on `A`.
- Repeat for signal pairs `B_N`, `B_P` and `Z_N`, `Z_P`.
- Apply 0V to `A_N`, 5V to `A_P`.
  - Verify that 5V is present on `A_H`.
  - Verify that 3.3V is present on `A`.
- Repeat for signal pairs `B_N`, `B_P` and `Z_N`, `Z_P`.

Verify ATTiny84 Functionality

- Load the ATTiny84 with the blink software.
  - Verify that the on-board LED is blinking

B Controller Board Verification Procedure

This document holds the verification procedure which, when applied succesfully, ensures the correct functionality of the controller board. Each step is to be followed in the order they are shown to avoid previous faults to propegate throughout the process. A step consists of an action and the expected outcome of that action. Any text written with the `teletype font` refers to signals and pins on the PCB and can be seen on the schematic. The most recent schematic can be seen on appendix C.

Verify Voltage Rails

- Apply 24V to `24V`.
- Toggle power button to ON.
  - Verify that current draw is within reasonable limits (<150 mA).
  - Verify 12V, 5V, 3.3V and 2.5V voltage rails.

Simulate No Emergency

- Apply 5V to `EM_END1` and `EM_END2`.
  - Verify that 0V is present on `EM_END1_OUT` and `EM_END2_OUT`.
- Apply 0V to `EM_END1` and `EM_END2`.
  - Verify that 5V is present on `EM_END1_OUT` and `EM_END2_OUT`.
- Set signals `EM_END1`, `EM_END2` and `EM_MCU` to 0V.
- Ensure `EM_BTN` is set to 5V.
  - Verify that 5V is present on `EM_DIS`

Simulate Emergency

- Set signals `EM_END1`, `EM_END2` and `EM_MCU` to 0V.
- Ensure `EM_BTN` is set to 5V.
- For each signal, toggle to either 5V or 0V.
  - Upon toggling one signal, verify that 0V is present on `EM_DIS`.

Verify Relay Circuitry Functionality

- Following the previous step, make EM_DIS high.
- Apply 0V to INRUSH.
  - Verify that 0V is present on the output of U5.
- Apply 5V to INRUSH.
  - Verify that 5V is present on the output of U5.
  - Verify that 24V is present on POWER_IN
- Apply 0V to M_RELAY.
  - Verify that 0V is present on the output of U6.
- Apply 5V to M_RELAY.
  - Verify that 5V is present on the output of U6.
- Apply 0V to INRUSH.
  - Ensure that 24V is still present on POWER_IN.

Verify Motor Driver Functionality

- Apply 5V to DIS.
- Apply 3.3V PWM to signals PWMA and PWMB.
  - Verify corresponding 5V PWM on signals PWM_AL and PWM_BL.
  - Verify that 0V is present on signals MGATE_BH, MGATE_BL, MGATE_AH and MGATE_AL
- Apply 0V to DIS.
- Ensure that 24V is still present on POWER_IN.
  - Verify that 12V PWM is present on signals MGATE_AL and MGATE_BL.
  - Verify that 35V PWM is present on signals MGATE_AH and MGATE_BH.
  - Verify that deadtime exists between the two channels.

Verify nRF24L01 Module functionality

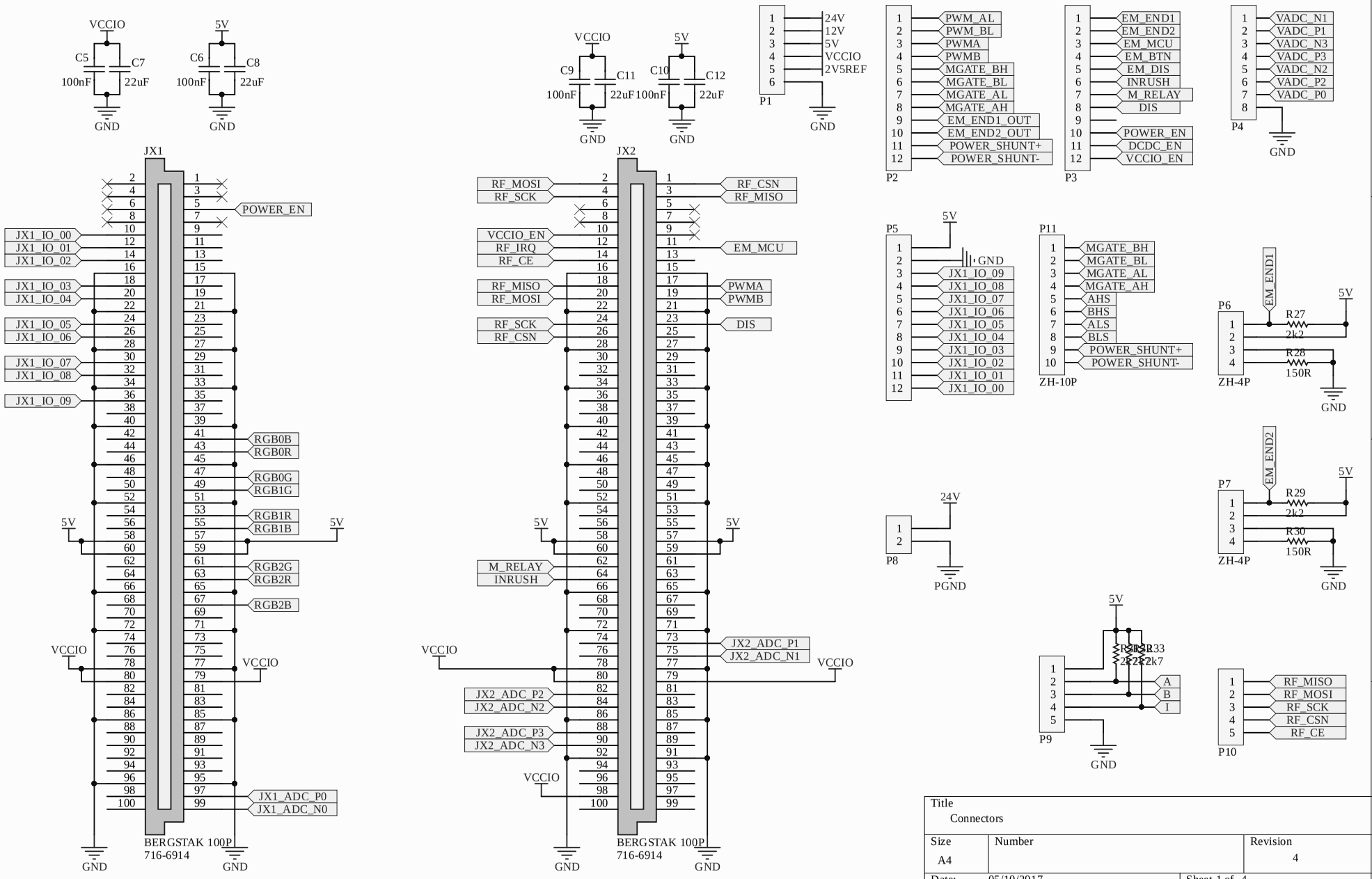
- Apply 3.3V to signals `RF_MISO`, `RF_MOSI`, `RF_SCK`, `RF_CSN` and `RF_CE`.
  - Verify that 3.3V is present on the corresponding signals on the `nRFM` module (component U1).

Verify Shunt Amplifier Functionality

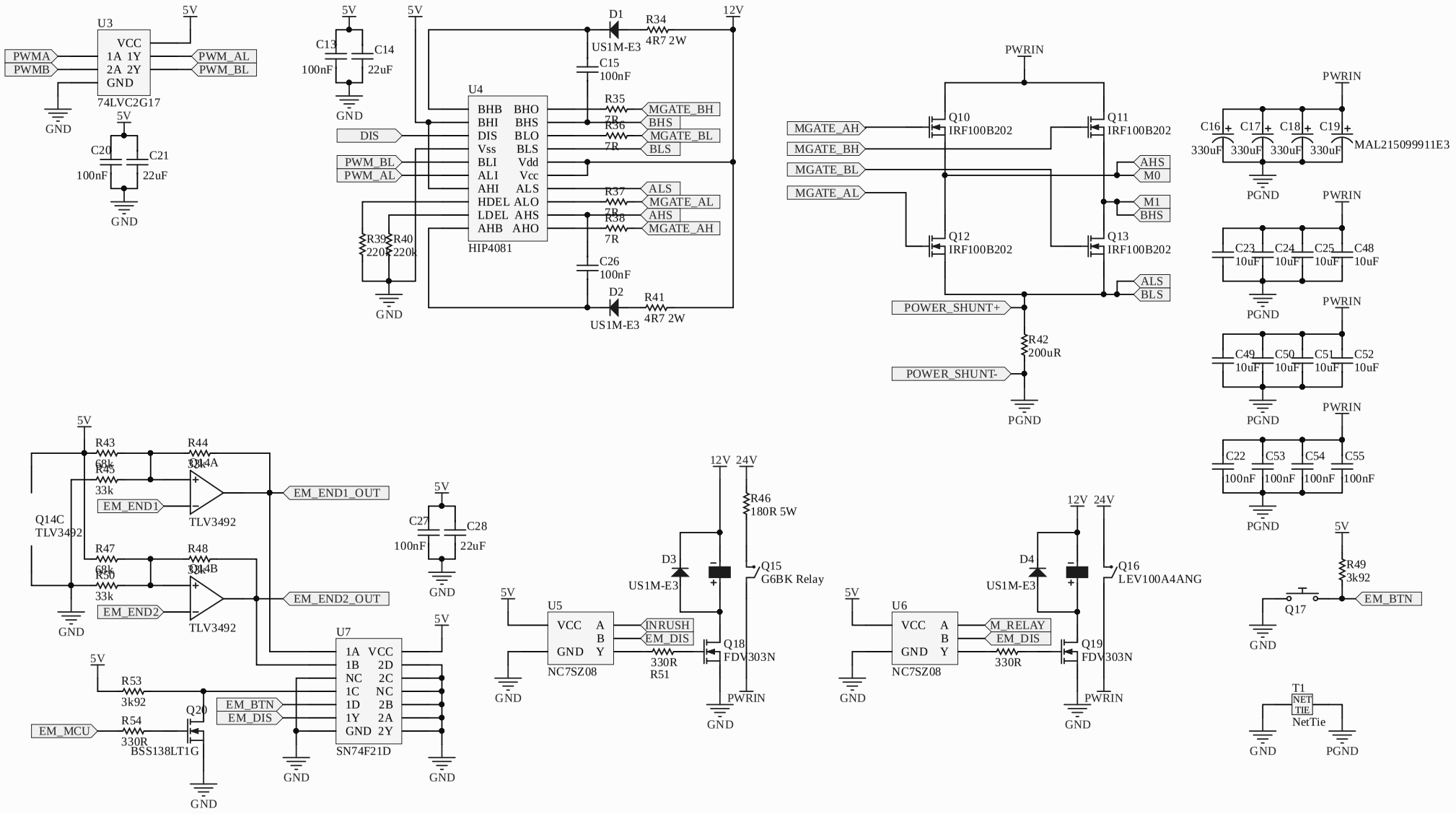
- Apply 10mV to `POWER_SHUNT+`.
  - Verify that 1V is present on `VADC_P0`.

C Controller Board Schematics

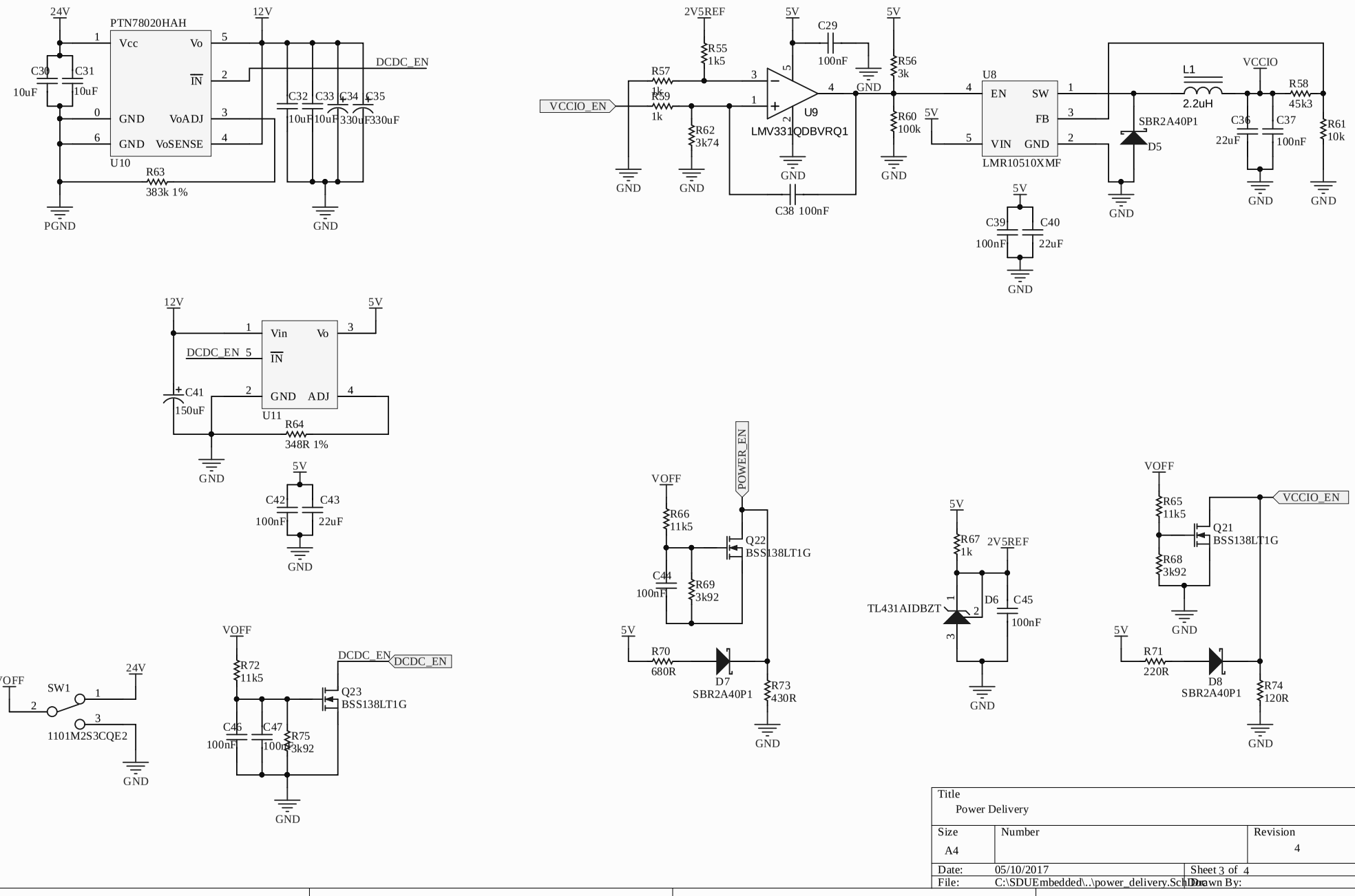
Controller board schematics can be found on the following pages.



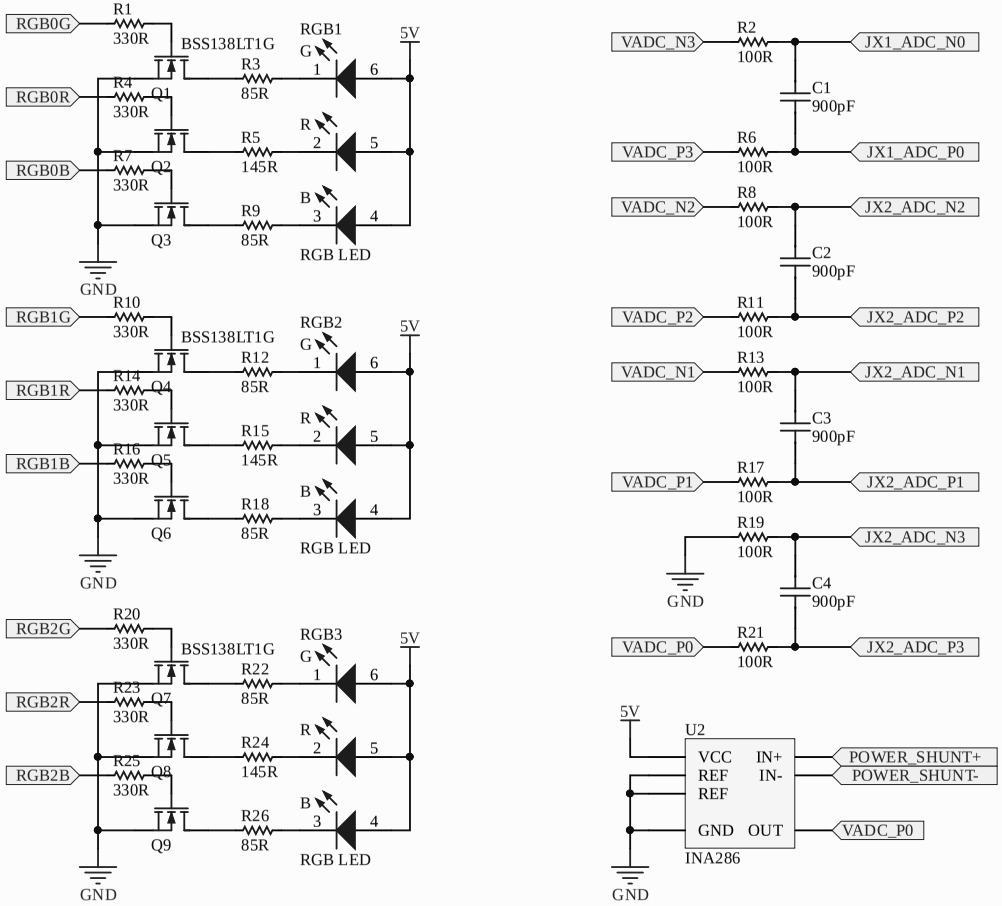
Title: Connectors		
Size: A4	Number:	Revision: 4
Date: 05/10/2017		Sheet 1 of 4
File: C:\SDUEmbedded\..\connectors.SchDoc		Drawn By:

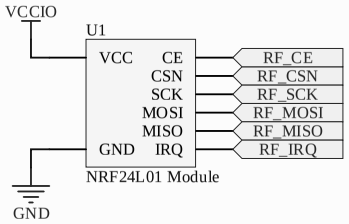


Title: Motor Driver		
Size: A4	Number:	Revision: 4
Date: 05/10/2017		Sheet 2 of 4
File: C:\SDUEmbedded\..\motor_driver.SchDoc		Drawn By:



Title: Power Delivery		
Size: A4	Number:	Revision: 4
Date: 05/10/2017		Sheet 3 of 4
File: C:\SDUEmbedded\..\power_delivery.SchDoc		Drawn By:

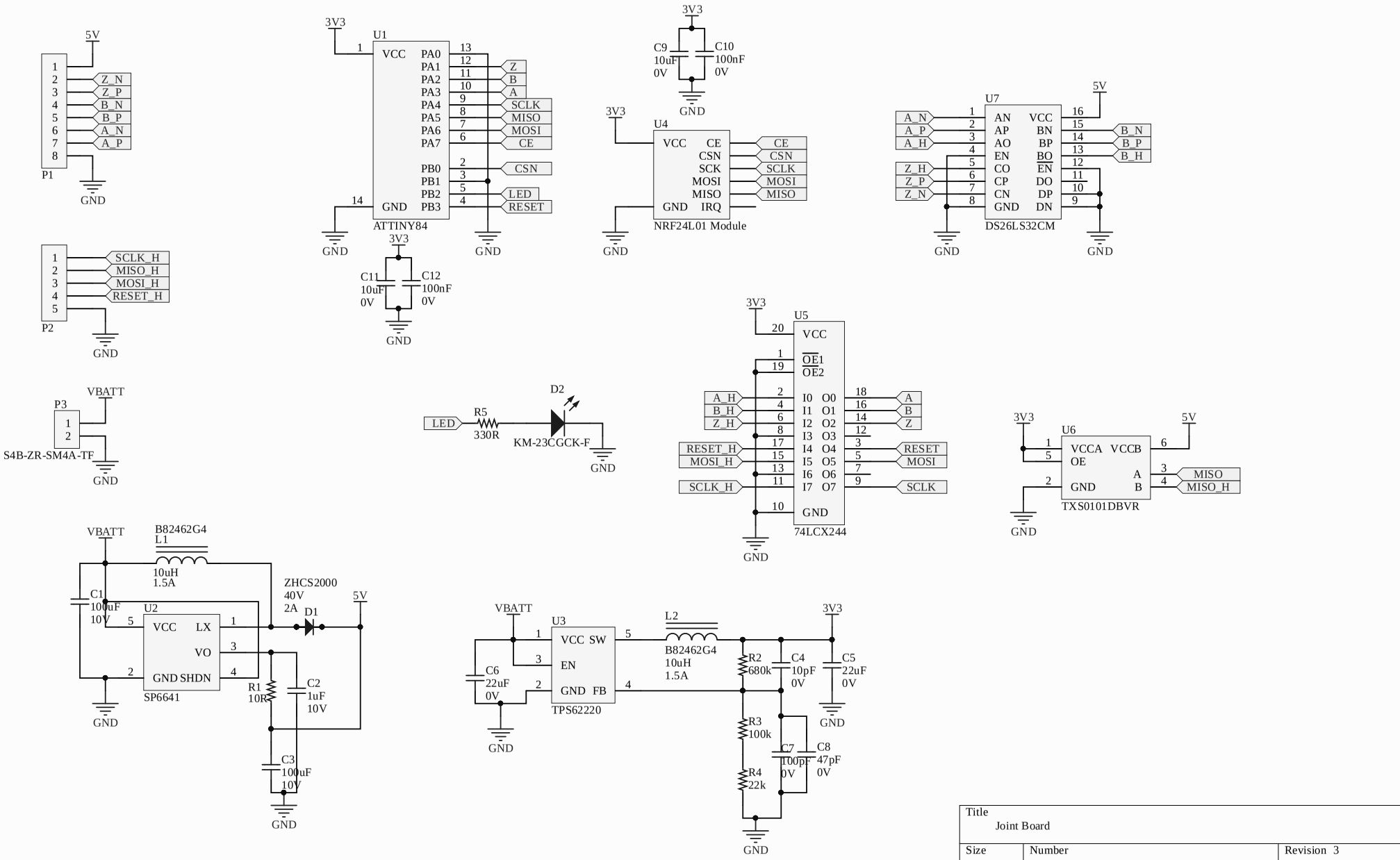




Title: Auxiliary		
Size: A4	Number:	Revision: 4
Date: 05/10/2017		Sheet 4 of 4
File: C:\SDUEmbedded\..\auxiliary.SchDoc		Drawn By:

D Joint Board Schematics

Joint board schematics can be found on the following page.



Title	Joint Board		
Size	Number		Revision 3
A4			
Date:	11/10/2017	Sheet 1 of 1	
File:	C:\SDUEmbedded\..\joint_board.SchDoc	Drawn By:	

E Model Development

In order to properly control the double pendulum it is necessary to develop a model of the system. Throughout this section the model will be developed iteratively. Initially including only the simple double pendulum, eventually including both friction and the cart.

E.1 Simple Double Pendulum

An illustration of the system can be seen in figure E.1. As can be seen, the pendulums have masses m_1, m_2, suspended using rods of lengths l_1, l_2. The rods are assumed to be massless. The angles of the pendulums θ_1, θ_2, are measured as the offset from the stable position. For convenience the two pendulums will be referred to as p_1 and p_2 where p_2 is suspended from p_1.

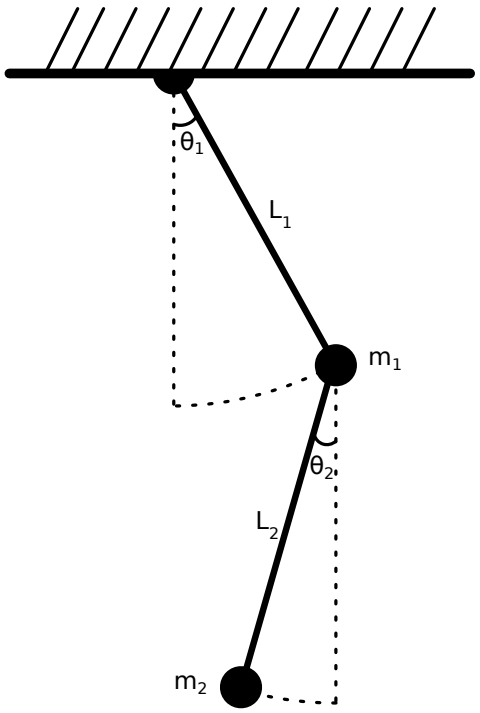


Figure E.1: Illustration of the simple double pendulum.

Equations should be developed such that, given an initial state, the position and subsequent movement can be derived from them. This can be achieved using the Euler-Lagrange differential equations. These will be found for this

system throughout the remainder of this section. Firstly, the position of the pendulums as a function of the angle must be determined. The pendulums are placed in the x-y plane with the origin placed at the point of suspension of p_1. When the pendulums are in the stable position, $\theta_1 = \theta_2 = 0$, they are aligned with the y-axis. The x-axis is positive to the right of the stable position. The coordinates of $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ are as follows:

$$x_1 = l_1 \sin\theta_1 \tag{E.1}$$
$$y_1 = -l_1 \cos\theta_1 \tag{E.2}$$
$$x_2 = l_1 \sin\theta_1 + l_2 \sin\theta_2 \tag{E.3}$$
$$y_2 = -l_1 \cos\theta_1 - l_2 \cos\theta_2 \tag{E.4}$$

While the derivatives are not used until later, they are presented here for clarity:

$$\dot{x}_1 = l_1 \cos\theta_1 \dot{\theta}_1 \tag{E.5}$$
$$\dot{y}_1 = l_1 \sin\theta_1 \dot{\theta}_1 \tag{E.6}$$
$$\dot{x}_2 = l_1 \cos\theta_1 \dot{\theta}_1 + l_2 \cos\theta_2 \dot{\theta}_2 \tag{E.7}$$
$$\dot{y}_2 = l_1 \sin\theta_1 \dot{\theta}_1 + l_2 \sin\theta_2 \dot{\theta}_2 \tag{E.8}$$

In order to calculate the Euler-Lagrange diff. eq. it is necessary to first determine the lagrangian, :

$$\mathscr{L} = E_k - E_p \tag{E.9}$$

Where E_k is the kinetic energy of the system and E_p, the potential energy. The derivation of E_p:

$$E_p = m_1 g y_1 + m_2 g y_2 \tag{E.10}$$
$$= -m_1 g l_1 \cos\theta_1 - m_2 g l_1 \cos\theta_1 - m_2 g l_2 \cos\theta_2 \tag{E.11}$$
$$= -(m_1 + m_2) l_1 g \cos\theta_1 - m_2 g l_2 \cos\theta_2 \tag{E.12}$$

The derivation of E_k:

$$E_k = \frac{1}{2} m_1 v_1^2 + \frac{1}{2} m_2 v_2^2 \tag{E.13}$$

As movement is present along both axes, the total velocity of either pendulum can be found as the derivative of the position and Pythagoras:

$$v_1^2 = \dot{x}_1^2 + \dot{y}_1^2 \tag{E.14}$$
$$= l_1^2 \dot{\theta}_1^2 \cos\theta_1^2 + l_1^2 \dot{\theta}_1^2 \sin{\theta_1}^2 \tag{E.15}$$
$$= (\cos{\theta_1}^2 + \sin{\theta_1}^2) l_1^2 \dot{\theta}_1 \tag{E.16}$$

Using the Pythagorean identity:

$$v_1^2 = l_1^2\dot{\theta}_1^2 \tag{E.17}$$

Similarly:

$$v_2^2 = \dot{x}_2^2 + \dot{y}_2^2 \tag{E.18}$$

$$= (l_1\cos\theta_1\dot{\theta}_1 + l_2\cos\theta_2\dot{\theta}_2)^2 + (l_1\sin\theta_1\dot{\theta}_1 + l_2\sin\theta_2\dot{\theta}_2)^2 \tag{E.19}$$

$$\begin{aligned} = {} & l_1^2\dot{\theta}_1^2\cos\theta_1{}^2 + l_2^2\dot{\theta}_2^2\cos\theta_2{}^2 + 2l_1l_2\dot{\theta}_1\dot{\theta}_2\cos\theta_1\cos\theta_2 \\ & + l_1^2\dot{\theta}_1^2\sin\theta_1{}^2 + l_2^2\dot{\theta}_2\sin\theta_2 + 2l_1l_2\dot{\theta}_1\dot{\theta}_2\sin\theta_1\sin\theta_2 \end{aligned} \tag{E.20}$$

Isolating cosine and sine from the terms pairwise, 1 and 4, 2 and 5, 3 and 6, yields:

$$\begin{aligned} v_2^2 = {} & (\cos\theta_1{}^2 + \sin\theta_1{}^2)l_1^2\dot{\theta}_1^2 + (\cos\theta_2{}^2 + \sin\theta_2{}^2)l_2^2\dot{\theta}_2^2 \\ & + (\cos\theta_1\cos\theta_2 + \sin\theta_1\sin\theta_2)2l_1l_2\dot{\theta}_1\dot{\theta}_2 \end{aligned} \tag{E.21}$$

By applying the trigonimetric identities:

$$1 = \sin\alpha^2 + \cos\beta^2 \tag{E.22}$$

$$\cos(\alpha \pm \beta) = \cos\alpha\cos\beta \mp \sin\alpha\sin\beta \tag{E.23}$$

The result is found:

$$v_2^2 = l_1^2\dot{\theta}_1^2 + l_2^2\dot{\theta}_2^2 + 2l_1l_2\dot{\theta}_1^2\dot{\theta}_2^2\cos(\theta_1 - \theta_2) \tag{E.24}$$

Using equations E.12, E.13, E.17 and E.24 the lagrangian can be constructed:

$$\begin{aligned} \mathscr{L} = {} & \frac{1}{2}m_1l_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2\left(l_1^2\dot{\theta}_1^2 + l_2^2\dot{\theta}_2^2 + 2l_1l_2\dot{\theta}_1^2\dot{\theta}_2^2\cos(\theta_1 - \theta_2)\right) \\ & + (m_1 + m_2)l_1g\cos\theta_1 + m_2l_2g\cos\theta_2 \end{aligned} \tag{E.25}$$

$$\begin{aligned} = {} & \frac{m_1l_1^2\dot{\theta}_1^2}{2} + \frac{m_2l_1^2\dot{\theta}_1^2}{2} + \frac{m_2l_2^2\dot{\theta}_2^2}{2} + l_1l_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_1 - \theta_2) \\ & + (m_1 + m_2)l_1g\cos\theta_1 + m_2l_2g\cos\theta_2 \end{aligned} \tag{E.26}$$

Which simplifies to:

$$\begin{aligned} \mathscr{L} = {} & \frac{(m_1 + m_2)l_1^2\dot{\theta}_1^2}{2} + \frac{m_2l_2^2\dot{\theta}_2}{2} + l_1l_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_1 - \theta_2) \\ & + (m_1 + m_2)l_1g\cos\theta_1 + m_2l_2g\cos\theta_2 \end{aligned} \tag{E.27}$$

The Euler-Lagrange diff. eq. are defined as:

$$\frac{d}{dt}\left(\frac{\partial \mathscr{L}}{\partial \dot{\theta}}\right) - \frac{\partial \mathscr{L}}{\partial \theta} = 0 \tag{E.28}$$

The three terms $\frac{\partial \mathscr{L}}{\partial \dot{\theta}}$,$\frac{d}{dt}\left(\frac{\partial \mathscr{L}}{\partial \dot{\theta}}\right)$ and $\frac{\partial l}{\partial \theta}$ are calculated next for each of the pendulums.
For p_1:

$$\frac{\partial \mathscr{L}}{\partial \dot{\theta}_1} = (m_1 + m_2) l_1^2 \dot{\theta}_1 + m_2 l_1 l_2 \dot{\theta}_2 \cos(\theta_1 - \theta_2) \tag{E.29}$$

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial \mathscr{L}}{\partial \dot{\theta}_1}\right) = {} & (m_1 + m_2) l_1^2 \ddot{\theta}_1 + m_2 l_1 l_2 \ddot{\theta}_2 \cos(\theta_1 - \theta_2) \\ & - m_2 l_1 l_2 \dot{\theta}_2 \sin(\theta_1 - \theta_2)(\dot{\theta}_1 - \dot{\theta}_2)\end{aligned} \tag{E.30}$$

$$\frac{\partial \mathscr{L}}{\partial \theta_1} = -m_2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \sin(\theta_1 - \theta_2) - (m_1 + m_2) l_1 g \sin\theta_1 \tag{E.31}$$

$$\tag{E.32}$$

$$\begin{aligned}0 = {} & (m_1 + m_2) l_1 \ddot{\theta}_1 + m_2 l_2 \ddot{\theta}_2 \cos(\theta_1 - \theta_2) \\ & + m_2 l_2 \dot{\theta}_2^2 \sin(\theta_1 - \theta_2) + (m_1 + m_2) g \sin\theta_1\end{aligned} \tag{E.33}$$

And p_2:

$$\frac{\partial \mathscr{L}}{\partial \dot{\theta}_2} = m_2 l_2^2 \dot{\theta}_2 + m_2 l_1 l_2 \dot{\theta}_1 \cos(\theta_1 - \theta_2) \tag{E.34}$$

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial \mathscr{L}}{\partial \dot{\theta}_2}\right) = {} & m_2 l_2^2 \ddot{\theta}_2 + m_2 l_1 l_2 \ddot{\theta}_1 \cos(\theta_1 - \theta_2) \\ & - m_2 l_1 l_2 \dot{\theta}_1 \sin(\theta_1 - \theta_2)(\dot{\theta}_1 - \dot{\theta}_2)\end{aligned} \tag{E.35}$$

$$\frac{\partial \mathscr{L}}{\partial \theta_2} = m_2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \sin(\theta_1 - \theta_2) - m_2 l_2 g \sin\theta_2 \tag{E.36}$$

$$\tag{E.37}$$

$$\begin{aligned}0 = {} & m_2 l_2 \ddot{\theta}_2 + m_2 l_1 \ddot{\theta}_1 \cos(\theta_1 - \theta_2) \\ & - m_2 l_1 \dot{\theta}_1^2 \sin(\theta_1 - \theta_2) + m_2 g \sin\theta_2\end{aligned} \tag{E.38}$$