THE UNIVERSITY OF
NEW SOUTH WALES

SCHOOL OF ELECTRICAL ENGINEERING
AND TELECOMMUNICATIONS

Steering-Integrated Driver Controls for Sunswift IV

by

Andrew Wrigley

A thesis submitted as a requirement for the degree
Bachelor of Engineering (Electrical Engineering)

Submitted: October 20, 2009 Student ID: z3099727

Supervisor: Gernot Heiser Topic ID: GH87

Abstract

Sunswift IV is the UNSW Solar Racing Team's entrant in the 2009 World Solar Challenge, a 3000km solar car race that runs biennially from Darwin to Adelaide. Recent changes to the international technical regulations for solar racing were required to slow the cars below the road speed limit and improve safety. One new regulation is a requirement for a steering wheel (as opposed to other methods such as push-pull steering previously employed several teams, including the UNSW team). This thesis integrates the driver controls and display into a steering wheel, consolidating all driver interfaces within the car in a similar way to Formula 1 racing cars. In addition, the greater visibility and area for controls allows for more functionality to be built into the steering wheel, including data logging, automated speed control, and a graphical display. This report details the research and development undertaken to build a system with such functionality. It also outlines the development of a first revision steering wheel that performs only essential driver controls and display functions.

Acknowledgements

A number of people have contributed to this project, and I would like to acknowledge their help and support here.

My supervisor Gernot Heiser, for supporting me with my topic, providing the resources to make the project possible, and making sure I kept working hard.

David Snowdon and Etienne Le Sueur, for being on-hand at NICTA to discuss design ideas and help out when needed.

Rudino Salleh, for all his help in getting access to a PCB milling machine (generally at late notice), and for lending various pieces of equipment when required.

Yael Augarten, for proof reading documents (many, many times), coffee breaks, generally keeping me sane, and for making sure I didn't have to eat McDonald's food.

Simon Li, for proof reading documents, being a consistent source of crazy but generally useful ideas, and for trying (but usually failing) to keep me fit throughout the year.

Konrad Zurcher, for giving me a crash course in mechanics and helping with the design of the mechanical steering wheel.

Members of competing solar car teams who shared information on strategy and quick release hubs.

Members of the UNSW Solar Racing Team for their help and support, in particular - Claudia Williams, John Durrant, Glen Summers and Helga Prochaska for laying up the carbon fibre plates used for the steering wheel; Irving Tjiptowarsono for helping with debugging ideas; Carson Au and Teng Zhang for soldering up the brake sensor board; Luke Bycroft and Campbell McLaren, for always answering my "where can I find an M4 bolt?" type questions at the workshop; and finally Jono Pye for not complaining too much every time he had to pull out the credit card and purchase something for this project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Brain Sport

In 1982 Hans Tholstrup and Larry Perkins drove a home built solar powered car named the Quiet Achiever across Australia, from Perth to Sydney. The 4000 km journey took 20 days to complete, with the car maintaining an average speed of just 23 km/h. In January 2007, the *University of New South Wales* (UNSW) *Solar Racing Team* (SRT) [59] completed the same journey with their solar powered car Sunswift III in just 5.5 days, averaging approximately 70 km/h. This threefold increase in average speed demonstrates clearly the significant development of the technology behind solar powered cars over recent decades.

Following his pioneering journey Hans Tholstrup created the *World Solar Challenge* (WSC), a solar car race designed to promote alternative transport technologies. The WSC is now a biennial event running from Darwin to Adelaide, where up to 50 solar car teams from over 12 countries compete to finish the 3000 km drive in the shortest time possible.

The average speed of the leading solar cars has steadily increased to an impressive 102.75 km/h achieved by the Nuon Solar Team in the 2005 WSC [8]. This led race organisers to make significant changes to the technical regulations [9] of the event in order to slow the cars down below the road speed limit, improve safety and to try and bring solar powered cars one step closer to a conventional road car. Changes included a 25% reduction in allowed solar cell area and the mandatory use of a steering wheel for control of the vehicle. The work in this thesis addresses the design and construction of this steering wheel.

Following the safety of the car's design, the primary concern of each team is the efficiency of the car. To be competitive in an event such as the WSC, each component of a solar car must be as efficient as possible. A modern day solar car has high aerodynamic efficiency, low weight, high efficiency photovoltaic cells and a high efficiency electric motor. A competitive team must also employ race strategy by efficiently managing the energy available. In the 2007 WSC, after 3000 km and 5 days of driving, the UNSW SRT missed out on 8th place by a mere 3 minutes, demonstrating how competitive the race is, and the importance of an efficient race strategy. Traditionally, all strategic analysis is performed in a support car following the solar car. The system developed in this thesis will allow the strategy calculations to be performed on-board the solar car.

1.2 Vehicle description

The steering wheel in this thesis is being designed for Sunswift IV, the solar car being developed by the UNSW Solar Racing Team. The team is currently in the process of designing and manufacturing components for Sunswift IV, which will meet the latest World Solar Challenge regulations. This description provides an overview of the electrical system that will be in Sunswift IV. Some of the electronics in the new vehicle will be based on the electrical system in its predecessor, Sunswift III, but much of it will be redesigned from the ground up.

Sunswift IV will have two separate data networks, each referred to as a *controller area network* (CAN) [24]. Each CAN consists of a series of nodes on a bus. For the purpose of this report, the two networks will be referred to as telemetryCAN and motorCAN. The steering wheel will be required to communicate on both of these networks.

TelemetryCAN is a custom built network which was designed in 2002 by a member of the UNSW Solar Racing Team [55]. The vast majority of nodes in Sunswift IV communicate over this network. Each node performs a specific function and sends regular messages over the network. Nodes in the car can also be configured to listen to another node for commands. All of the data on the telemetryCAN is relayed to a support car for analysis via a wireless connection. Strategy software is used to calculate an optimal cruising speed which is then relayed to the solar car driver via CB radio.

The protocol which runs motorCAN was developed by Tritium [36], a Queensland based power electronics company who produce the Wavesculptor motor controller used in Sunswift IV. The only nodes currently on the motorCAN are the motor controller itself, the driver controls and a node called the sculptorbridge. The sculptorbridge provides a uni-directional link between the two networks, relaying all data on the motorCAN to the telemetryCAN.

Finally, the existing driver display consists of a 40x2 *liquid crystal display* (LCD) connected to the driver display node via a ribbon cable. The node listens for messages from other nodes on the network and displays the relevant data.

1.3 Problems with existing setup

The existing setup, described in the previous section, has a significant drawback: all data logging and strategy calculations to optimise speed are performed in the support car travelling behind the solar car. Should the wireless connection between the solar car and its support car fail, data will be lost and the solar car will have no means to calculate its most efficient speed. As previously stressed, performing strategy calculations such as optimal speed is crucial in remaining competitive in the WSC.

A second drawback of the existing system is with the driver display. The driver's cockpit is very limited in terms of space and on a small number of occasions the ribbon cable has been accidentally pulled out of place by the driver. This results in loss of data to the LCD, and potentially having to stop the solar car to rectify the problem wasting essential race time.

1.4 Proposed solution

The introduction of the mandatory use of a steering wheel provides an opportunity to integrate the driver controls, driver display, sculptorbridge and data analysis and storage into one system and overcome all of the problems detailed in the previous section.

It is clear that the implementation of on-board data analysis and storage in the solar vehicle has many advantages. For example, if a certain level of data analysis were to be performed on-board the vehicle then a cruising speed could be calculated without the use of the support car. This could then form part of a real time control loop, which would not be possible if the data had to be sent via the wireless link to the support car. The use of a real time control loop would allow much higher frequency changes in the set speed, enabling the system to respond quickly to changes in conditions, thereby increasing the overall energy efficiency of the solar car.

For example, the UNSW SRT has extensive data on the terrain between Darwin and Adelaide thanks to a comprehensive survey carried out by the School of Geomatic Engineering at UNSW [71]. This means that every gradient is known in advance, and an adaptive cruise control system could be implemented to allow continual adjustment of the cruising speed. This, and other potential strategies that could be implemented are discussed in Chapter 2.

By integrating these systems into the steering wheel, rather than elsewhere in the car, other advantages become apparent. The system would require less cabling and would be compact and modular, allowing a somewhat generic, transferable system that could be used in other solar powered and electric vehicles. This would hopefully allow solar cars to become a little more like conventional road cars, in the sense that they wouldn't need to always travel with a support car.

1.5 Objectives

The aim of this thesis is to integrate several electronic systems into the steering wheel, namely the driver controls, driver display, sculptorbridge and data analysis and storage. The result will be a sophisticated driver interface similar to that seen in Formula 1 racing cars, but unique in the solar racing car field. The system has the following key requirements. It must:

- Provide a user interface to allow the driver to control all aspects of driving the car. This includes primary functions such as throttle control and also ancillary functions such as turning indicators.
- Display real time data and graphs of key variables such as array power, motor power and battery state of charge.
- Provide a bidirectional communication link between the telemetryCAN and motorCAN.
- Be able to perform data analysis and storage.
- Allow a real time speed control loop (adaptive cruise control) to be implemented.
- Accept a rear vision video camera signal and display the video in real time on the screen.

This thesis involves the design, manufacture, testing and implementation of an electronic system that meets all of these requirements. The end result should provide increased functionality, a more reliable system and a better driving experience. The author is also conducting the mechanical design and construction of the physical steering wheel, but this report will deal primarily with the electronic systems.

1.6 Outline

The general background of solar racing cars and the World Solar Challenge has been introduced. Chapter 2 contains a more detailed summary of vehicle embedded systems and the theory behind efficient energy management, which will be crucial for the successful implementation of this thesis. Chapter 3 details the design specifications, along with a detailed comparison of the various design options available. Chapter 4 details the hardware development achieved. Results are discussed in Chapter 5. Finally, Chapter 6 provides an outline of future work and conclusions.

Chapter 2

Literature review and background theory

This chapter examines the use of embedded systems in road vehicles, Formula 1 racing and solar car racing, and how the proposed adaptive cruise control system will have unique features not seen in any of these fields. This chapter also provides a review of the theory behind solar car strategy, focusing on particular strategies that would be difficult to implement with the existing setup, but will be possible with the system proposed in this thesis.

2.1 Vehicle embedded systems

2.1.1 Road vehicles

The number of embedded computer-based functions in road vehicles has increased drastically over the last 10 years [40]. A range of electronic functions such as navigation, engine management and active safety systems are commonplace in a modern vehicle. Many luxury vehicles contain an adaptive cruise control system that automatically reduces the speed of the vehicle if it senses a slower moving vehicle or other obstacle ahead [6, 15]. However, there is evidence [21] to suggest that this type of technology has not been well received in the USA, and as such its use is not prevalent. On the other hand, a paper by M. Parent [45] suggests that there are several projects based in Europe that are working on autonomous transport and are making significant progress. One of the problems is that standards and certification processes do not yet exist for autonomous transport. Solar car racing does not have this problem, making it ideal for developing such systems. In addition, the systems seen in road vehicles are generally for the purpose of driver convenience and accident avoidance, unlike the system proposed in this thesis, which is concerning energy efficiency.

2.1.2 Formula 1 racing

Complex electronics have been integrated into the steering wheels of Formula 1 cars since the mid 1990's [60]. The needs of a solar car, detailed below, are similar to that of a Formula 1 car, so it is likely that similar solutions will be appropriate. Research has been conducted into the steering wheel interface and electronic systems in a typical Formula 1 car, however Formula 1 racing is extremely competitive and so unfortunately there is not a large amount of detailed information publicly available. A general report by J. Waldo on embedded computing and Formula 1 [70] details how many of the racing cars contain an *electronic control unit* (ECU). The ECU has a real time operating system and is responsible for controlling many systems within the vehicle as well as all data logging. However, the ECU does not autonomously control the speed of the vehicle because Article 5.5 of the *Federation Internationale de L'Automobile* (FIA) regulations [19] strictly prohibit any form of cruise control.

As detailed in a paper by L. Cocco et al [10], when discussing data acquisition and analysis on Formula 1 cars, all electronic systems must be capable of operating reliably in a high temperature and high vibrations environment. Solar racing cars experience similar environmental conditions and also require large amounts of data to be analysed to run efficiently. Whilst Formula 1 racing is at an elite level not yet reached by solar car racing, it provides a good standard for which to aim.

2.1.3 Solar car racing

The Tritium Wavesculptor motor controller that will be used in Sunswift IV has an inbuilt cruise control system that attempts to keep the solar car at a relatively constant speed. This particular motor controller is popular in the solar racing car field and so many teams have this basic cruise control system in their vehicle. Like Formula 1, solar racing can be quite competitive and so detailed information on electronic systems is often, but not always, kept secret. However, from the information that the author was able to obtain, it appears that very few other solar car teams have implemented an adaptive cruise control system. The Nuon team [44] calculate the optimal speed for their car Nuna 4 in a support car in a similar way to the UNSW team. However, Nuna 4 has a cruise control system that receives its target speed through a wireless connection from the support car travelling behind the solar car [50]. From anecdotal evidence the Umicore Solar Team [11] have a similar setup, but no specific details are available.

The problem with the Nuna 4 system is that wireless connections between the solar car and support car are not always reliable, which introduces the possibility of data loss and potential failure of the cruise control system. In addition, a real time system would be difficult to implement as the the round trip from the solar car to the support car and back will not necessarily have consistent timing. The advantage of the system proposed in this thesis is that the support car is not needed for strategy calculations to be made, which eliminates these two problems. By moving data analysis traditionally performed in the support car to the solar car itself, the solar car is able to manage itself much like a conventional road car.

2.2 Efficient energy management

The importance of strategy in solar car racing has been introduced in Section 1. The relevant theory and possible strategies will now be examined. An accurate model of the power consumption of a solar car is a vital starting point in forming an optimal strategy. The model, at a minimum, is used to estimate the amount of power a solar car requires at any given speed using ideal conditions. A more complex model will take into account other variations such as cross-winds and terrain gradients. The most commonly used model to estimate the amount of power required to drive a solar car is given in Equation 2.1. This model is used by the UNSW SRT and is based on The Speed of Light by David Roche et al [52] with an additional variable, the second coefficient of rolling resistance, as used by Paul Vincent [16].

$$P = \frac{1}{\eta}\left[mgv\left(\sin\left(G\right) + C_{rr1}\cos\left(G\right)\right) + NC_{rr2}v + \frac{1}{2}C_d A\rho\left(v - v_w\right)^2\right] \quad (2.1)$$

where:

m = mass (kg)

g = gravity (ms^{-2})

v = velocity (ms^{-1})

G = gradient (rads)

C_{rr1} = First Coefficient of Rolling Resistance (no units)

C_{rr2} = Second Coefficient of Rolling Resistance ($kg\ s^{-1}$)

N = number of wheels

C_dA = drag area (m^2)

P = power used (W)

η = drive train efficiency (no units)

ρ = density of air (kg/m^3)

v_w = wind velocity parallel to road (ms^{-1})

As detailed by Peter Pudney [51], there are different strategic approaches a solar racing team can employ during a competitive race such as the World Solar Challenge. Pudney proposes that on a level road with a battery with constant energy efficiency the optimal strategy contains four driving modes: maximum power, lower speed holding, upper speed holding and maximum regenerative braking. The two holding speeds are generally no further than 10 km/h apart and the lower speed is held when solar power is low and the upper speed is held when solar power is high. This can be relatively easily implemented by the solar car driver and a basic cruise control system.

2.2.1 Climbing hills

When driving a course with steep gradients, the optimal strategy is to switch to maximum power mode before an incline so that the speed increases before the incline and is allowed to decrease whilst climbing and then return to the cruising speed once on level road. By using this strategy the hill can be overcome relatively quickly and with the smallest amount of energy. This technique is illustrated in Figure 2.1. A similar approach can be taken with steep declines, except that regenerative braking should be engaged before the decline, and the car allowed to increase in speed during the decline. An example optimal speed curve over a hill is shown in Figure 2.2.

This strategy is not as easy to implement as a simple holding speed strategy, as it requires the terrain ahead to be known and the driver must change the speed

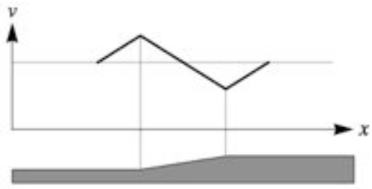

Figure 2.1: Power phase of a solar car for a steep incline. Car velocity versus distance. Road gradient shown in grey. *(Image courtesy Pudney [51].)*

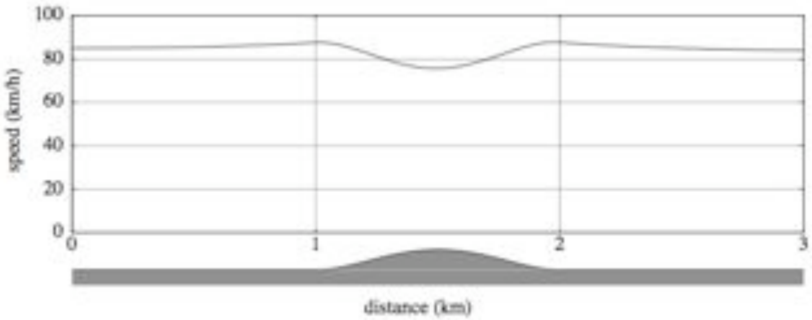

Figure 2.2: Optimal speed of a solar car versus distance over a hill. Road gradient shown in grey. *(Image courtesy Pudney [51].)*

appropriately. The adaptive cruise control system proposed in this thesis would be able to automatically increase the speed of the solar car on approach to a hill without any driver input. This speed increase would occur at precisely the right time and rate for the specific incline.

The strategy of increasing speed before inclines has an added benefit apart from the optimum efficiency already detailed. Many of the competitive teams use in-wheel electric motors developed by the *Commonwealth Scientific and Industrial Research Organisation* (CSIRO) [17]. The stator is encapsulated in a resin that is rated at approximately 100^oC and if this is exceeded, irreparable damage may occur. Compared with a level road, climbing gradients requires significantly more current through the stator, resulting in greater heat generation. If maximum power (and therefore maximum motor current) is used prior to the incline but not on the incline then the total heat generation will be reduced and the stator temperature is less likely to reach dangerous levels.

2.2.2 Sun chasing

Sun chasing is a term used by Pudney to a describe an optimal strategy during periods of patchy clouds, where the resultant solar irradiation varies in space. If the solar car travels faster whilst under a cloud and slower whilst in bright sunlight then the amount of energy collected is increased compared with a constant speed strategy. The additional energy obtained is more than the extra energy needed to vary the speed, so an increase in the overall average speed is possible. An example of optimal speed for a given solar power is shown in Figure 2.3.

This strategy is also difficult to implement manually, as the driver must try and keep track of the solar irradiation and respond accordingly. An adaptive cruise control system could automatically vary the solar car speed based on the power generated by the solar array, much more effectively than manual control.

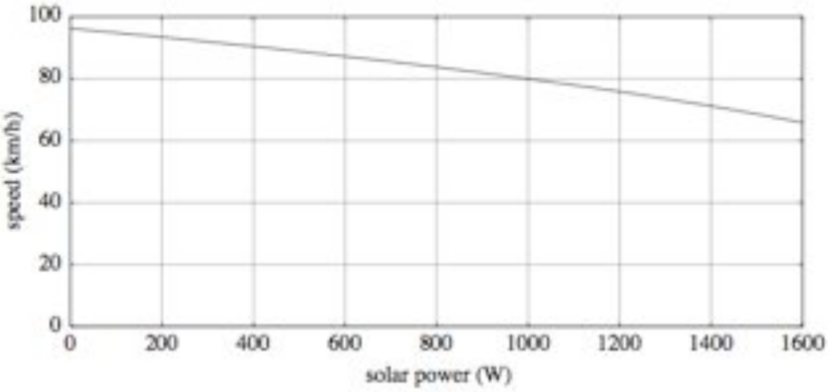

Figure 2.3: Optimal speed of a solar car for the available solar power. *(Image courtesy Pudney [51].)*

2.2.3 Long term strategy

Competitors in the WSC start the race with a full battery pack, and from then on are only allowed to charge the battery from the solar array. If the optimal speed is calculated correctly, the solar car will reach the finish line with a completely flat battery pack and in the shortest time possible. With course profile data, weather prediction data and an accurate solar car model it is possible to simulate the battery *state of charge* (SOC) over the 5 days of the WSC in advance. During the race, one must then try and follow this battery SOC curve as closely as possible. If conditions (such as the weather) change, then the course must be simulated again and a new battery curve produced.

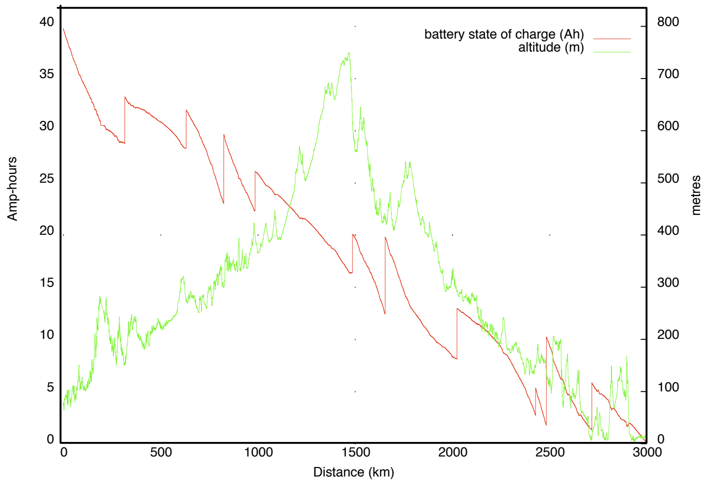

Figure 2.4: Example battery state of charge curve (red) and course altitude profile (green) along WSC course. *(Image courtesy David Snowdon.)*

The adaptive cruise control system proposed will be able to accept an already simulated battery SOC curve and adjust the speed as necessary to ensure the curve is followed. Ideally, the system would then also be able to accept updated files from the support car or make its own changes to the curve. This is required because conditions such as weather may change throughout the day. An example of a simulated battery SOC curve for Sunswift IV during the WSC is shown in Figure 2.4. Note that the sudden increases in SOC are during control stops where the solar car is stationary but the battery is still charging from the solar array. Also note that the SOC is exactly zero at the end of the 3000 km race.

Chapter 3

Design Specifications and Requirements

This chapter discusses the technical considerations necessary to achieve the requirements detailed in Section 1.5.

3.1 Embedded system requirements

The vast majority of the nodes currently in Sunswift III use the Texas Instruments [31] MSP430 microcontroller. To achieve the functionality required for the integrated driver controls, a more capable microprocessor is required. There are a large number of different embedded system platforms that could be used. To decide on the most appropriate platform the following requirements were placed on the system:

1. Operating System - The hardware should be capable of running a lightweight Linux operating system.

2. Data storage - The system must have the capacity to store up to 250 MB of telemetry data per 24 hours of use.

3. Real-time - Access to a real time clock is required for accurate control loop functions.

4. Power - The complete system must use as little power as possible and a maximum of 3 Watts.

5. Video/Graphics - The system must be able to perform analogue to digital conversion of the rear-view camera video signal and to handle graphical display functions.

6. Processing capability - The processor must be able to handle all calculations required for the control loops and for strategy.

7. Peripherals support - Peripherals are required to allow programming, input buttons, interfacing with the telemetryCAN and motorCAN and common communication buses.

8. Monetary cost - The system should be developed using a minimal amount of money to ensure the UNSW SRT budget is met.

Additionally, the system would ideally have on-board wireless communication hardware. This is not a hard requirement, but it is preferable. The steering wheel must be removable and so a wireless connection would allow the steering wheel to continue to log data even when disconnected from the car during a pit-stop.

3.2 Development platform selection

Initially, the plan was to design the embedded system from the ground up. However, many of the available microprocessors are contained in a *plastic ball grid array* (PBGA) package which necessitates the use of a 6-layered *printed circuit board* (PCB) due to the close proximity of the electrical contacts. Whilst the design of such a PCB would be possible, it would likely take a considerable amount of time to design and debug, which may prevent some of the more advanced features proposed for this thesis from being implemented in time. Additionally, a 6-layered PCB is considerably more expensive to manufacture than a conventional 2-layered PCB. For these reasons, an alternative solution was devised. There are many embedded system development boards available which are designed to provide a relatively fast way of building embedded system software without needing to be concerned with the associated hardware. One of these such boards will be used to develop the required software for the integrated steering wheel. If time allows, then specific hardware can be designed to replace the development board. A selection of available options are compared in the following sections.

3.2.1 gumstix Overo Earth

The Overo Earth computer module is made by gumstix [27] and contains the Texas Instruments OMAP3503 [31] applications processor with an ARM Cortex-A8 core [35].

Advantages include:

- Compact size, so could be used in the car as is.
- Additional features can be easily added through gumstix expansion boards.
- Contains a camera input module.
- Relatively inexpensive.

Disadvantages include:

- Not open source so schematics are not available.
- No on-board wireless capability.

3.2.2 gumstix Overo Fire

The Overo Fire is similar to the Earth, but contains several additional features. It contains the Texas Instruments OMAP3530 applications processor with an ARM Cortex-A8 core and has the same physical dimensions as the Overo Earth. It can also be easily expanded through other gumstix boards and contains a camera input module.

Advantages include:

- Contains on-board wireless communication via 802.11(g) and Bluetooth.
- Contains a *digital signal processor* (DSP) and a *graphics processing unit* (GPU).

Disadvantages include:

- Higher cost than the Overo Earth.

3.2.3 phyCORE ARM11

The phyCORE ARM11/i.MX31 is a development board made by Phytec America [2] and contains the Freescale Semiconductor i.MX31 [26] applications processor with an ARM11 core. A rapid development kit version of this board is available immediately for use at NICTA. The rapid development kit contains the central development board, along with an expansion board with a large number of peripherals.

Advantages include:

- Contains a large amount of peripheral support on board.
- Hardware schematics are available.
- Contains a CAN controller on board.
- Has a camera input module to handle video signals.
- Is immediately available.

Disadvantages include:

- Relatively large physical size.
- The most expensive out of the development boards investigated.

3.2.4 Beagle Board

The Beagle Board [1] is an open source design PCB containing the Texas Instruments OMAP3530 series processor with the ARM Cortex-A8 core, as in the gumstix Overo Fire board.

Advantages include:

- Two research students at NICTA are using the board, which would allow potential collaboration in case of technical problems.
- Contains a DSP and GPU.
- Completely open source, so hardware schematics are available.
- Relatively inexpensive.

Disadvantages include:

- Does not have on-board support for ADC or 1-wire interfaces.
- Cannot connect an analogue video camera to the board as it has no video A/D conversion capabilities.

3.2.5 Conclusion

Table 3.1 provides a comparison between the different development boards. The phyCORE board was eliminated as it is at least double the cost of the other boards examined. Whilst one is already available for use, a spare would most likely have to be purchased if it is to be used in the solar car. Comparing the two gumstix Overo boards, the Fire version has the advantage of a digital signal processor. A DSP is capable of performing large number of mathematical operations quickly, so much of the planned data analysis can be performed on the DSP instead of the main processor. The Fire version also has a graphics processing unit which, in a similar fashion to the DSP, will take much of the processing required to display real time graphs away from the main processor. The Overo Earth lacks a DSP and GPU and so was eliminated.

The Overo Fire and the Beagle Board have the same processor, DSP and GPU. The Beagle Board does not have any on-board wireless communication capabilities, where as the Overo Fire supports both 802.11(g) and Bluetooth. The Overo Fire is also considerably smaller than the Beagle Board. The primary advantages of the Beagle Board are its affordability and the fact that two research students are working with the board, allowing potential collaboration.

The final decision was made in favour of the gumstix Overo Fire. This development board meets all of the requirements set out in Section 3.1 and whilst not the cheapest board, it offers the best value for money. The design of the gumstix board is also more suitable than the Beagle Board for connecting directly to another PCB. Whilst not having hardware schematics is a disadvantage, there is a well developed online community mailing list [56] which may be useful if other technical information is required.

3.3 Adaptive cruise control

The idea of an adaptive cruise control system was introduced in Section 2.1. How this can be practically implemented will now be examined. To be able to continually adjust the speed of the car based on gradients and solar irradiation, an accurate course profile containing detailed information on the terrain is required. In addition, the following sensors must be installed in the solar car:

- *Global Positioning System* (GPS) receiver to determine exactly where the solar car is along the course.
- Tilt sensor to allow compensation for course profile or GPS errors.
- Solar array power measurement to determine solar irradiation.
- Accurate speed sensor.
- Current integrator to accurately determine battery state of charge.

At the time of writing, the current integrator node is under development, and all of the other sensors exist on Sunswift III and will also be used in Sunswift IV. The car speed is obtained from both the Wavesculptor motor controller and the GPS receiver. The high level system setup is shown in Figure 3.1. A proposed cascade control loop is shown in Figure 3.2. The desired battery state of charge is maintained by the outer control loop whilst the inner control loop regulates the speed of the car required to achieve the set battery SOC.

It is also important that an accurate model of the solar car exists so that the vehicle dynamics are known. If the system is to increase the speed of the car before an incline it must, for example, know the maximum acceleration rate of the solar car. This can be estimated based on the design of the car, but would need to be calibrated based on real data during on-road testing and incorporated into the proposed control loop shown.

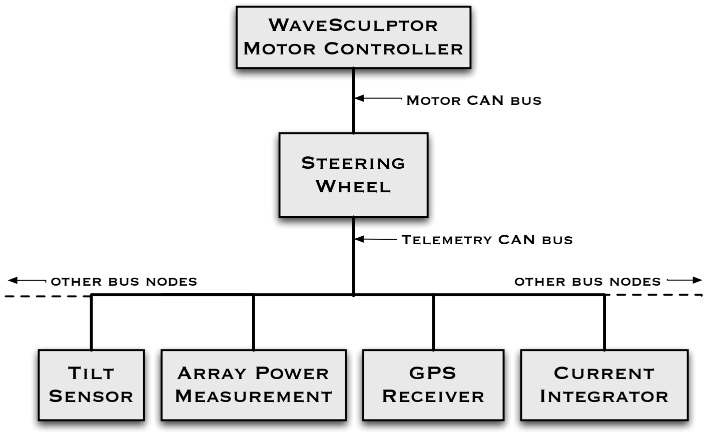

Figure 3.1: Proposed high level system of relevant nodes for an adaptive cruise control system in Sunswift IV.

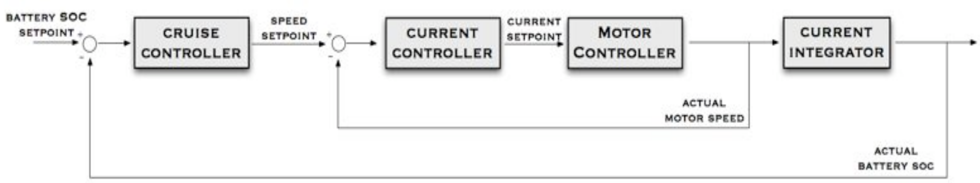

Figure 3.2: Proposed control loop structure for an adaptive cruise control system in Sunswift IV.

	Overo Earth	*Overo Fire*	*phyCORE*	*Beagle Board*
Applications Processor	OMAP 3503 with ARM Cortex-A8	OMAP 3530 with ARM Cortex-A8	i.MX31 with ARM1136JF-S	OMAP 3530 with ARM Cortex-A8
Clock Speed	600 MHz	600 MHz	532 MHz	600 MHz
DDR RAM	256 MB	256 MB	128 MB	256 MB
NAND flash	256 MB	256 MB	64 MB	256 MB
Digital Signal Processor	N/A	430 MHz TMS320C64x	N/A	430 MHz TMS320C64x
Graphics Processing Unit	N/A	110 MHz POWERVR SGX	Yes	110 MHz POWERVR SGX
On-board memory slot	microSD	microSD	SD, CF	SD
Real time clock	TPS65950	TPS65950	Yes	TWL4030
Supported Interfaces	USB 2.0, SPI, I2C, PWM, ADC, UART, 1-wire, Camera-in	USB 2.0, SPI, I2C, PWM, ADC, UART, 1-wire, Camera-in	USB 2.0, SPI, I2C, UART, 1-wire	USB 2.0, I2C, SPI, UART
On-board wireless	N/A	802.11(g) and Bluetooth	N/A	N/A
Dimensions (mm)	17 x 58 x 4.2	17 x 58 x 4.2	58 x 84	76.2 x 76.2
Cost (USD)	$149	$219	$439	$149
Power: Processors (max)	686mA @ 1.20V	1137mA @ 1.20V	470mA @ 1.60V	1137mA @ 1.20V
Power: Core (max)	328mA @ 1.15V	433mA @ 1.20V	270mA @ 1.65V	433mA @ 1.20V

Table 3.1: Comparison of selected embedded system development boards.

Chapter 4

Hardware Development

This chapter details the design procedures and component choices undertaken to achieve the requirements for this project detailed in Section 1.5. Section 4.1 discusses the development of an interface for the gumstix Overo board and Section 4.2 details how communication with the vehicle networks is achieved. The remaining sections discuss the development of a user interface, rear vision system, power supply and other required components.

All schematic design and *printed circuit board* (PCB) component and track layout was done using Altium Designer Winter 09 [34]. The vast majority of component footprints were manually created by the author to ensure suitability for hand soldering. All component layout and track routing was also done manually.

As discussed previously, a basic version of the driver controls PCB (version 1) was designed by the author just prior to the commencement of this thesis. This was to allow experimentation of the user interface and to ensure the Sunswift team had at least a basic, functional driver control system. Much of the development of the version 1 driver controls was undertaken during the thesis period and is discussed further in Chapter 5.

4.1 gumstix Overo Interface

The Overo board contains the processor and other critical devices, but must be interfaced with an expansion board which provides power and routes out the pins to additional hardware and peripherals. The gumstix Overo board mates with the expansion board via two 70 pin AVX-5602 connectors [33]. This allows for communication between the Overo motherboard and the expansion board and also provides power to the power management *integrated circuit* (IC) on the Overo. Every gumstix Overo development board has the same pin configuration for these connectors, so the PCB designed for this thesis is compatible with all Overo series boards. All I/O pins on the Overo have a logic high of 1.8V, however most other ICs used in this design are only compatible with 3.3V logic high. Therefore voltage shifting transceivers were required for most of the pins used on the Overo. The specific transceivers used will be discussed in the relevant sections.

In order to interface with the Overo during the development process, a console connection was included via one of the *universal asynchronous receiver/transmitter* (UART) ports on the Overo. The FTDI FT232RL [22] USB to serial converter IC was added to allow standard communication via USB. In order to allow a direct connection to a standard USB port on a personal computer, a mini-USB connector was used. The console connection allows *command line interface* (CLI) to be used on a personal computer, and is used to connect to the Overo, login in as a user, configure the system, install software and so on.

In addition to the console connection, a USB *on-the-go* (OTG) connector was added. USB OTG is a supplement to the standard USB 2.0 specification[14]. Unlike standard USB, it allows a USB device (in this case, the Overo) to perform both master and slave roles, depending on whether the ID pin on the USB cable is either grounded or floating. The pin state is determined by the orientation of the connecting cable used. In host/master mode, standard peripherals such as a keyboard and mouse can be connected, allowing the user to interact with the Overo as they would with a personal computer. The Overo could also be used in slave mode, where for example it could be connected to a computer as a mass storage device, allowing easy file transfer. As with the console connection, the OTG connection will most likely be used only during development period and for maintenance.

4.2 Communication with Motor Controller and Telemetry bus

As stated in Section 1.5, the system must be capable of communicating on both the Sunswift telemetryCAN bus and the Tritium motorCAN bus. This is required so that the system can send and receive data from other nodes in the car and for communication with the motor controller. Several different topologies were investigated and their relative merits are discussed here.

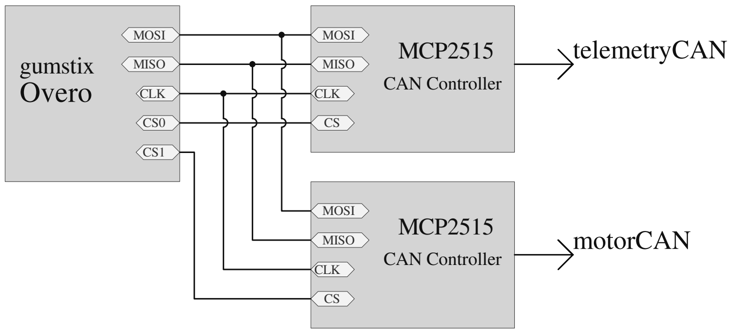

Figure 4.1: Overo communication topology option 1.

4.2.1 Overo

The OMAP processor used on the gumstix Overo does not have in-built CAN support, and so is not capable of communicating with telemetryCAN and motorCAN without additional hardware. One solution is to use two MCP2515 CAN controllers [39] to communicate with both CAN buses, as in Figure 4.1. The MCP2515 would communicate with the Overo via SPI. This solution is potentially troublesome for two reasons. First, despite the OMAP processor having several SPI ports, only one is mapped out by the Overo board. This would mean the two CAN controllers would have to be connected on the one SPI bus and the chip select line would be used to share time between them. The MCP2515 receive buffers are relatively small, so the chances of missing CAN messages is quite high. A possible solution to this problem is discussed in Section 4.2.2.

The second problem with this solution is that all driver control operations will be performed on the Overo. This means that the driver controls are reliant on the Linux operating system, and should the system hang or crash the car would be undrivable. With considerable testing and debugging, the chances of an operating system crash could be reduced, but certainly not eliminated. In the short development time available, this would be very difficult to achieve. Therefore, this option will not be used.

4.2.2 MSP430 MCU

The second option, shown in Figure 4.2, would be to use an MSP430 microcontroller [30], as well as two MCP2515 CAN controllers, which would solve the operating system reliability problem. All critical driver control functions

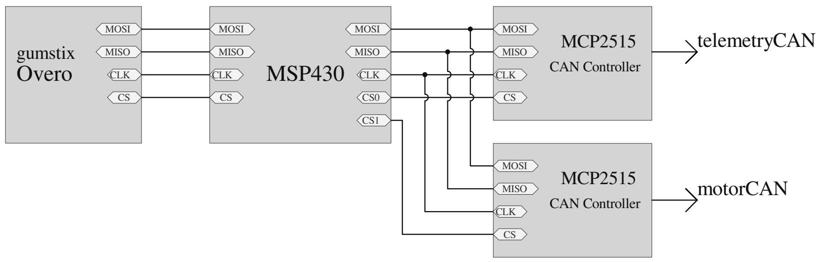

Figure 4.2: Overo communication topology option 2.

would be performed on the microcontroller, and the Overo would handle the information display, the control loop and the strategy software. It would also allow all the software that has been developed by the author for the version 1 driver controls, discussed in Chapter 5, to be directly transferred over with minimal modification. In addition, the vast majority of Sunswift nodes use the MSP430 and MCP2515, so schematics are readily available.

At the time of design, none of the MSP430 family microcontrollers had more than two communication ports. This presented a problem, since two ports are required for communication with the CAN controllers, and a third for communication with the Overo. Again, this problem could potentially be resolved by putting both CAN controllers on the one SPI bus and using the chip select line. However, this solution is unfavourable for the same reasons outlined in the previous section. A possible solution is a technique known as bit banging, which involves handling SPI through software instead of dedicated hardware, allowing an SPI device to be connected to any GPIO pins on the MSP430. However, this technique would consume more processing power than a hardware solution and is generally considered a 'hack'. Therefore this solution is not ideal and will not be implemented. Note that at the time of writing, Texas Instruments has just released a new microcontroller, the MSP430F5438, which has four communication ports.

4.2.3 STM32 MCU

The ST Microelectronics STM32 is a relatively new ARM-based microcontroller which has two in-built CAN controllers. The STM32, as used in Figure 4.3, eliminates the need for the external MCP2515 CAN controllers, and allows the

STM32 to communicate with the Overo via a dedicated SPI bus, resolving the major problems raised by the previous two options.

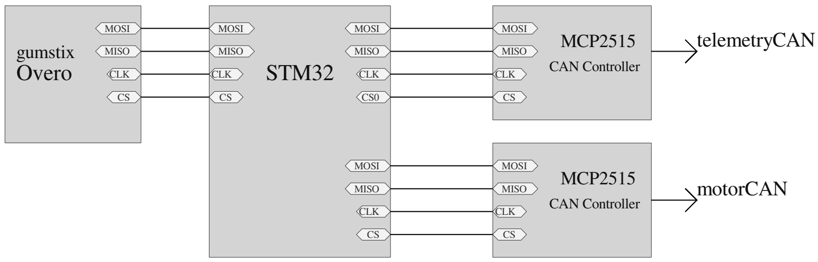

Figure 4.3: Overo communication topology option 3.

However, this option would require extra software development work, as all of the existing software would need to be ported over to ARM. This has the advantage that should it prove successful, it would provide the Sunswift team with an alternative to the significantly more expensive MSP430. Another benefit is that the STM32 supports programming via CAN, which would be a useful feature to have on other Sunswift boards, as it would allow programming of the CAN nodes already installed in the car. The current setup requires a node be removed from the car for it to be programmed. For these reasons, the STM32F107 MCU [57] was chosen to perform all driver control functionality and communication on both CAN buses. It has 256 kB of Flash memory, 64 kB of SRAM and an ARM Cortex M3 core with a CPU frequency of 72 MHz.

Support for three different programming interfaces was included in the schematic design for the STM32. The STM32 programming manual [58] was used as a guide to determine the required hardware for each of the programming interfaces. The bootloader, stored in the internal system memory, is used to download the program to the internal Flash memory through one of three communication interfaces. Two micro-switches were added to set the BOOT pins on the MCU either high or low, to allow selection of either 'programming' or 'normal' boot modes.

To program through the *universal synchronous-asynchronous receiver/transmitter* (USART) port, an RS-232 transceiver chip and 4-pin header were added. The intersil ICL3232 [32] transceiver provides voltage level shifting between the 5V logic used by a standard computer serial port and the 3.3V logic of the microcontroller. This method of programming is the least convenient due to the decline in the availability of serial ports on laptop computers, but was included as it is likely to be the simplest to setup.

To program via USB, a mini-AB USB-OTG connector was added, with the communication lines directly connected to the MCU. No transceiver is required as the USB-OTG pins on the MCU are 5V tolerant. NXP PESD5V0X [43] *electro-static discharge* (ESD) protection didoes were added to the signal lines as a precaution to protect the MCU from voltage spikes.

Finally, to communicate via CAN the only hardware required was two 3.3V CAN transceivers for communication on each bus. The Texas Instruments SN65HVD232 [62] was chosen as it is used on all other Sunswift boards. No other additional hardware is required specifically for programming via CAN.

4.3 User Interface

The user interface consists of buttons and switches to allow the driver to control all aspects of driving the solar car, and a screen for displaying useful data to the driver.

The Samsung LTE430WQ-F0C *[53] thin film transistor* (TFT) *liquid crystal display* (LCD) was chosen for the driver display. This is a 4.3 inch widescreen display with 480x272 resolution. Gumstix produce an expansion board for the Overo development boards, the Palo43 [28], that uses this LCD and the schematics [29] for which are available under a Creative Commons Licence [12]. This was the primary motivation for the choice of LCD, as it allowed a reasonably quick development time and the LCD was known to work well with the Overo. Unfortunately, the schematics are only available in Eagle [23] format, a program with which the author had little experience. Therefore some time had to be spent learning the program so that the schematics could be properly interpreted and transferred into Altium Designer.

The LCD uses a standard 24 bit RGB interface, with 3.3V logic high. The LCD has a *flexible printed circuit* (FPC) that connects to the PCB through a 45 pin Hirose FH12-45 connector [25]. All signal lines between the LCD and Overo are passed through Texas Instruments SN74AVCB16425 bus transceivers [67]. These transceivers were chosen as they are also used on the Palo43 expansion board. Again, selecting components already used on the Palo43 reduced the time taken to design the associated schematics. This proved the most challenging bus to route, due to tight space constraints and the seemingly random pin order of the LCD lines on the Overo.

The chosen LCD has a backlight unit consisting of 10 white LEDs. These must be driven by a boost regulator. The Micrel MIC2297 [38] 40V boost regulator white LED driver was selected, again because it is used on the Palo43. The MIC2297 has a *pulse width modulation* (PWM) input from the Overo, to allow control of the brightness of the backlight. The enable pin has been connected to a GPIO on the Overo to reduce power consumption by ensuring the backlight is only on when needed. The MIC2297 is only available in a *micro lead frame* (MLF) chip package, which has proven very difficult to solder. This is discussed further in the Chapter 5.

The LCD also has a touch screen panel. There are no specific plans to use the touch screen functionality whilst driving the car, but hardware support has been added in case it is needed in the future. A touch screen controller is required to resolve the co-ordinates of the pressed area of the screen and communicate this information to the Overo. The Texas Instruments ADS7846N [65] 4-wire touch screen controller was selected, again because this chip is used on the Palo43 expansion board.

The user controls designed were very similar to that used in the version 1 PCB mentioned previously. A number of pushbuttons are connected to the STM32 MCU to activate functions such as left and right indicators, hazards, horn, ignition, reverse, and push to talk for the *citizen's band* (CB) radio. The controls were designed to be as intuitive as possible to ensure ease of operation. To control acceleration and regenerative braking two Penny & Giles JC120 joysticks [48] were used. These are variable resistors, mounted as 'paddles' on the rear of the steering wheel. They are connected to the ADC port on the STM32, and are operated by pulling the paddles towards the driver. The right hand paddle directly controls the amount of current supplied to the wheel motor, and therefore the torque of the motor, to between 0% and 100% of the configured maximum. This is analogous to a conventional road vehicle, where the accelerator pedal controls the *revolutions per minute* (rpm), and therefore torque, of the engine. The left hand paddle operates in a similar way, but controls the amount of regenerative braking current.

A 2-way hat switch was added to adjust the maximum bus current setpoint to between 0% and 100% of the configured motor controller maximum. This allows the driver to reduce the maximum current used by the motor if, for example, it is starting to overheat or the battery pack has a low state of charge. A 2-way plus centre push hat switch and a speed-hold pushbutton were added for cruise control operation. This also operates in a similar way to a conventional road vehicle. The driver accelerates to the desired speed using the right paddle, then engages cruise control by pressing the speed-hold button. The set speed can be fine-tuned by moving the hat switch up or down. The centre push on the hat switch de-activates cruise control and saves the speed to resume later.

Finally, a number of *light emmitting diodes* (LEDs) were added to indicate various conditions to the driver, such as 12V and 3.3V power active, MCU running, indicators operational and error conditions.

4.4 Rear Vision

A rear vision video camera signal must be displayed to the solar car driver at all times, as stated in the WSC regulations. The Overo board can accept a 12 bit digital video signal on the BT.601/BT.656 standard [54], via a separate 27 pin connector. Two video camera options were investigated. Firstly, a *Complementary metal–oxide–semiconductor* (CMOS) image sensor could be used, either on a custom designed PCB or as an 'off-the-shelf' unit. This could then be directly connected to the Overo. The second option is to use an analogue camera and connect it through a video decoding IC to the Overo. The analogue camera option was chosen, as this would only require two cables (signal and ground) to be physically connected to the steering wheel, as opposed to at least 10 if the digital sensor was used. Due to the difficulties involved in routing cables to the steering wheel, they should be minimised wherever possible.

The video decoder is required to decode a standard *phase alternating line* (PAL) or *national television system committee* (NTSC) encoded video signal from the analogue camera and provide a digital signal of a maximum of 12 bits to the Overo. A fellow student at NICTA recommended the Texas Instruments TVP5150A video decoder [64], as it had been used successfully in a project in the past. This IC uses 3.3V logic high levels, so the use of a transceiver is required for communication with the Overo. Other video decoder IC's available in 1.8V logic high were investigated to avoid the use of the transceiver, but none were found that met the requirements. Therefore the TVP5150A was selected. A SN74AVCB164245 bus transceiver was used for the 8 bit video data and associated signal lines such as clock and enable. To allow the Overo to control reading and writing operations on the decoder IC an *inter-integrated circuit* (I2C) bus is used. I2C is a bi-directional bus, and so a bi-directional bus transceiver was required. The Texas Instruments PCA9306 [69] was selected for this purpose.

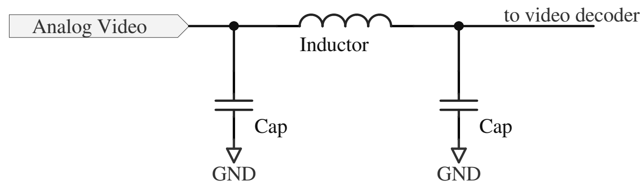

Figure 4.4: Video signal anti-aliasing filter.

4.4.1 Anti-aliasing filter

The video chip has a sampling rate of 27MHz for PAL and NTSC video signals. An anti-aliasing filter was designed for the video input to reduce the effect of any noise outside the normal video bandwidth. The filter was designed to have a 3dB cut-off frequency of 10MHz, as recommended by the manufacturer's application guide [61]. A Cauer topology was used, and an excerpt from the designed schematic is shown in Figure 4.4.

4.5 Power

The voltages required on the board are 12V, 3.3V and 1.8V. The board is supplied with 12V via the CAN bus, and 3.3V and 1.8V must be generated on board. Switching converters are generally significantly more efficient than linear regulators, but require several external components and a longer design time in order to calculate the component values. The majority of components on the board use 3.3V and so a switching converter was used to convert 12V to 3.3V. However, only a small number of voltage transceivers use 1.8V and so for simplicity and space reasons a linear regulator was used to convert 3.3V to 1.8V.

The switching converter chosen was the Texas Instruments TPS62111. The component values and tolerances for the required external components were calculated and are worked through here. For an input voltage of 12V the TPS62111 has a maximum output current of 1.5A. The total current draw of all components on the board is estimated to be a maximum of 450mA. The output control loop contains an inductor and capacitor, the values and ratings of which are critical to stable operation of the switching converter. A lower inductance value gives a smaller voltage overshoot during load transients, but has an increased output current ripple. The datasheet [63] recommends an inductance value of

$6.8\mu H$. The inductor must be current rated to at least the maximum output current plus the inductor ripple. The equation to calculate the maximum ripple ($\triangle I_L$) is provided in the datasheet as:

$$\begin{aligned} \triangle I_L &= V_{out} \times \frac{1 - \frac{V_{out}}{V_{in}}}{L \times f} \\ \triangle I_L &= 3.3 \times \frac{1 - \frac{3.3}{12}}{6.8\mu \times 1000} \end{aligned}$$

Using an inductor of $6.8\mu H$ and assuming a switching frequency of 1000 kHz, the maximum current ripple will be 350mA. The total current through the inductor is given by:

$$\begin{aligned} I_{Lmax} &= I_{out} + \frac{\triangle I_L}{2} \\ &= 450mA + \frac{350mA}{2} \end{aligned}$$

Giving a maximum current of 625mA. So, the inductor was chosen to have a maximum rating of 650mA.

4.5.1 Backup Battery Supply

A backup battery power supply was required for several reasons. The Overo board is running an operating system which takes approximately 30 seconds to boot up. If the steering wheel loses bus power, such as when the steering wheel is removed from the car during a driver change, the driver will have to wait for the operating system to reboot before commencing driving. Secondly, the steering wheel is being used for all data storage, and so having a battery powered supply would allow telemetry data collection via a wireless link.

World Solar Challenge regulations [9] specify a total permitted battery weight in the solar car. The vast majority of the battery weight is taken up in the main traction battery. At the time of design, the total amount of 'spare' battery weight was unknown. Therefore a reasonably standard battery cell size was chosen so that the capacity, and therefore weight, could be changed as required. Several different rechargeable battery chemistries and sizes were investigated. *Nickel metal hydride* (NiMH) and *lithium ion* (LiON) batteries are both readily available in a variety of case sizes. However, LiON was selected for its superior energy density and higher cell voltage. This allows a single cell to be used, instead of multiple cells in series, reducing charge complexities due to battery mismatch.

A charging system was required for the LiON battery cell. This ensures the cell is never over charged, and allows the level of charging current to be regulated to a desired level. This is critical as overcharging a LiON battery can result in fire or explosion. There are a large number of charge management ICs available and many were investigated. However, only a small handful met the requirements of 12V input and supporting constant current and constant voltage charging. Of these, the Texas Instruments BQ24012 [68] was chosen as it has an extensive datasheet and was the most reasonably priced.

As stated, the exact battery capacity was unknown at time of design, but is estimated to be approximately 500 mAh. Under maximum load and assuming a maximum depth of discharge of 80%, the discharge time is calculated as follows:

$$\begin{aligned} T_{discharge} &= \frac{capacity \times \%DOD}{I_{load}} \\ &= \frac{500mAh \times 80\%}{450mA} \\ T_{discharge} &= 53\, minutes \end{aligned}$$

The discharge period calculated will be used as an approximate guide only, as resistive losses and other battery cell characteristics have not been taken into consideration. It is expected that the steering wheel will be operating on battery power at any one time for a maximum of 5 to 10 minutes, so the calculated discharge time is more than sufficient.

The BQ24012 has a configurable charge current limit. The charge current is set to a relatively low value in order to reduce unnecessary loading of the system bus. A standard Sunswift node has a power supply with a current limit of 50 mA, so this was chosen as an ideal current limit for the battery charger. The maximum charge current will naturally govern the minimum charge time, which can be calculated as follows:

$$\begin{aligned} T_{charge} &= \frac{capacity \times \%DOD}{I_{charge}} \\ &= \frac{500mAh \times 80\%}{50\, mA} \\ T_{charge} &= 8\, hours \end{aligned}$$

World Solar Challenge regulations [7] specify the length of each race day to be 9 hours. This means that if the battery cell is completely flat at the start of the day, in ideal conditions it will be fully charged by the end of the race day. The current limit is set with a current sense resistor, calculated using equations provided in the BQ24012 datasheet:

$$\begin{aligned} R_{sense} &= \frac{K_{set} \times V_{set}}{I_{out}} \\ R_{sense} &= \frac{430 \times 2.5}{0.05} \\ R_{sense} &= 21.5\, k\Omega \end{aligned}$$

Using a preferred value of 22kΩ for R_{sense} gives a maximum current of 48.9mA and a minimum charge time of 8 hours and 10 minutes. If a battery cell of significantly different capacity is used the sense resistor can easily be changed to allow a lower or higher charging current.

A 5V transient suppressor diode and fuse were used with the battery as in Figure 4.5. This protects the rest of the circuit from the battery in two ways. First, if the battery is inserted with the incorrect polarity the diode will be forward biased and allow a large current to pass through the fuse, causing it to blow. Second, if an incorrect battery with too high voltage is inserted, the transient suppressor diode will breakdown and conduct, again causing the fuse to blow. The enable line of the BQ24012 is setup so charging commences automatically when 12V is present, but can be disabled through a GPIO on the Overo.

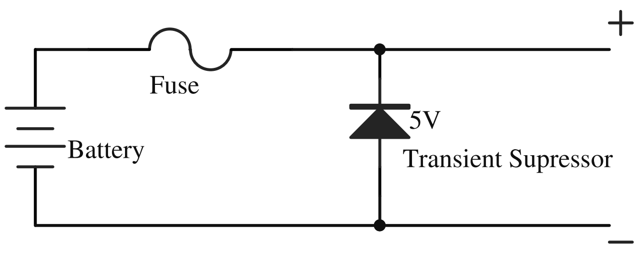

Figure 4.5: Battery protection circuit.

A linear regulator was required to regulate the battery voltage, between 2.7V and 4.2V, to the required board voltage of 3.3V. A linear regulator was chosen over a potentially more efficient switching converter to reduce complexity and footprint space, as during normal vehicle operation the regulator will only be enabled for relatively short periods of time. The regulator chosen was the Texas Instruments REG113-33 [66], as it has a reasonably high efficiency and is designed specifically for use with LiON batteries.

The power circuitry was designed to ensure automatic switching between 12V bus power and battery power when bus power is lost. It can also be switched manually via GPIOs on the Overo. Finally, a switch was added to the PCB to completely shutdown all power circuitry, so the steering wheel can be switched off at the end of the race day.

4.6 Miscellaneous

4.6.1 Accelerometer

An accelerometer was added to the design to determine when the steering wheel is being turned and by how much. This data will then be used to tilt the rear vision video signal in order to ensure the rear vision is always the right way up for the driver. The accelerometer chosen was the Analog Devices ADXL335 3-axis, +/-3g [3], because of its low cost and its compatibility with 1.8V supply, which allows direct connection to the gumstix Overo. The ADXL335 has an analog output for each axis, which are connected through a filter to the ADC pins on the gumstix Overo. The filter is required to remove unwanted high frequency accelerations from normal vehicle vibrations. The outputs of the ADXL335 are passed through 32 $k\Omega$ resistors, and so capacitors were added to form a simple RC filter circuit. The required capacitance is calculated by:

$$f_{-3dB} = \frac{1}{2\pi RC} \tag{4.1}$$

Therefore, for a 3dB cut-off frequency of 10Hz, 0.47μF capacitors are required.

4.6.2 External Connectors

The steering wheel must be able to be quickly removed to allow the driver to exit the solar car. Initially a commercial quick release steering boss with an 8 pin internal electrical connector was to be used. However, due to the design of the steering column it is not possible to route cables inside the column. Therefore a more basic quick release hub, the 3400-3/4ALU [47] (which has no electrical connector), was sourced from Pegasus Auto Racing. The cables will instead be connected via a 8 pin Weidmuller PCB connector, the pin-out of which is shown in Table 4.1. The pin-out was limited to 8 to ensure compatibility with an electrical quick release should it be used in the future with a different steering column design.

Pin	Description
1	GND
2	camera signal
3	motorCAN High
4	motorCAN Low
5	telemetryCAN Low
6	telemetryCAN High
7	GND
8	12V

Table 4.1: Main connector pin-out.

Many of the pins on the Overo were not used. 12 of the unused pins are routed to a header to allow the connection of an expansion board, should it be needed in the future. The pin-out of the header is provided in Table 4.2. Note that this header is routed directly to the Overo without buffers, so 1.8V logic or voltage transceivers must be used on any expansion board designed.

Pin	Description
1	GPIO_163
2	GPIO_146
3	GPIO_151_RXD1
4	GPIO_147
5	GPIO_148_TXD1
6	GPIO_186
7	ADCIN2
8	GPIO_184
9	ADCIN6
10	GPIO0_WKUP
11	ADCIN7
12	GPIO10

Table 4.2: Expansion header pin-out.

A 7 pin connector, shown in Table 4.3, is used to connect each of the JC120 joystick paddles to the PCB. Note that only pins 4,5 & 6 are used in this design.

Pin	Description
1	switch common (NC)
2	switch, lever forward (NC)
3	switch, lever backward (NC)
4	GND
5	output voltage signal
6	+3.3V
7	centre tap (NC)

Table 4.3: Joystick paddle connector pin-out.

4.6.3 Remote Driver Controls

Wireless driver controls may be useful to eliminate cables to the physical steering wheel and allow quicker removal. Support for this feature has been added for possible future use, but was not intended to be fully implemented in this thesis. To allow for this feature, a header has been placed on the PCB for a Sparkfun breakout board containing the Nordic Semiconductor nRF24L01+ transceiver [42]. It is a low power, short range transceiver popular in small battery operated wireless devices and communicates via SPI with the STM32 MCU. If wireless driver controls were to be fully implemented, a battery powered PCB with buttons and an MCU could be designed to sit inside the steering wheel and communicate with the PCB designed in this thesis. The MCU would be active whenever a button is pressed, and stay in sleep mode all other times.

4.7 Brake Sensor

The motor controller must know when the driver is applying the mechanical brake to ensure that the motor does not continue to drive forwards. Traditionally this has been implemented with a microswitch on the brake pedal, with a ground and signal cable connecting back to the driver controls. When the switch is closed, the software will cut power to the motor during forward motoring but not during regenerative braking. To enable cabling to be routed away from the driver's compartment and for more intelligent brake sensing techniques, part of the work in this thesis was the design of a brake sensor board that connects to motorCAN, instead of directly to the driver controls.

Rather than a simple on or off switch, the board was designed to measure the level of braking being applied, so that the regenerative braking power could be scaled to match. Two different ways of measuring the level of braking applied were investigated - sensing how much the brake pedal has moved (pedal travel) or sensing how much force the driver is applying. Measuring brake pedal travel was deemed unsuitable, as this can change drastically with varying conditions such as brake line pressure and the level of brake pad wear. Conversely the amount of force applied by the driver should provide a good indication of the desired braking effort.

A piezoresistive force sensor was selected. The resistance of the sensor varies with the amount of force applied. The sensor used is the Tekscan FlexiForce A201 from Sparkfun electronics [20]. Using a force sensor allows regenerative braking to be applied prior to full mechanical braking, by detecting the initial force applied by the driver. This is advantageous, as mechanical braking is inefficient because the energy used to slow down the car cannot be recovered as it is with regenerative braking. The sensor also allows the amount of regenerative braking to be varied in relation to the amount of force exerted by the driver, which is not possible in the existing system.

The board was designed with an MSP430 microcontroller and an MCP2515 CAN transceiver - a standard setup for most UNSW SRT boards. Schematics and PCB design are provided in Appendix B. It contains an LTC1474 switching converter to supply 3.3V to the board from the 12V on the CAN bus. It also contains connectors for several inputs - the force sensor, a microswitch as a backup for the force sensor and an encoder for measuring the amount of pedal travel. The pedal travel measurement was included to allow analysis by the mechanical team. A simple diode clamp circuit was used for all external inputs, to protect the microcontroller from over-voltage spikes. The circuit prevents the input voltage from going above 3.3V or below 0V. The PCB component and track layout was done using Altium Designer Winter 09.

4.8 Summary

The design choices and calculations relevant to the hardware design of the version 2 steering wheel have been presented. The schematics and PCB layout for all hardware design discussed in this chapter are contained in Appendices A & B. The quantity and complexity of the hardware development was beyond that estimated by the author, and took significantly longer than anticipated to complete.

Chapter 5

Results

This chapter details the results of all work undertaken as part of this thesis. The first section discusses general preparatory work. Sections 5.2 and 5.3 detail the results specific to the version 1 and version 2 steering wheel designs. Section 5.4 addresses the system requirements, and in the final section the mechanical manufacture of the steering wheel is briefly discussed.

5.1 General preparatory work

Significant time was spent conducting background research into the areas of embedded systems and solar car strategy. Time was also spent researching various products available to meet the design requirements detailed in Section 3.1.

The program MATLAB [4] was used to perform some simple power usage modelling based on the predicted parameters of Sunswift IV. MATLAB was also used to improve the author's understanding of the power equation discussed in Section 2.2, particularly the behaviour and the significance of specific parameters.

A test system was set up in the hardware lab at NICTA, (Figure 5.1). This consists of a laboratory power supply, Tritium Wavesculptor motor controller, Lillington DC Brushless Permanent Magnet T-Flux motor and a laptop computer for monitoring the system. This system has been used to develop and test the steering-integrated driver controls.

Finally, a program which is currently being developed by a member of the UNSW SRT has been installed on the author's laptop computer to allow enhanced monitoring, logging and graphing of telemetry data. This was used to

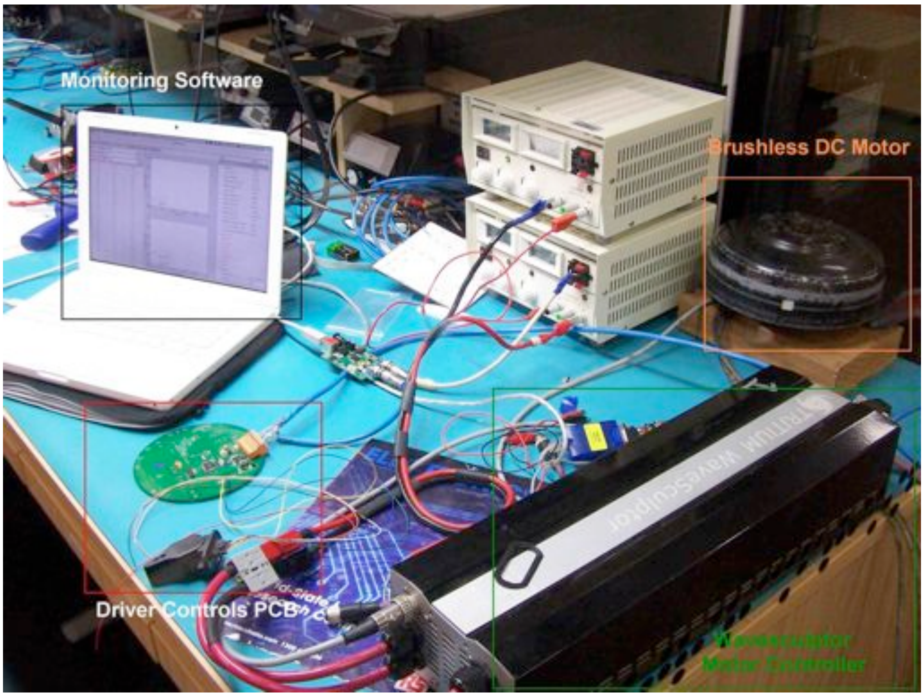

Figure 5.1: The testing system in the NICTA hardware lab used to develop the version 1 driver controls.

monitor the laboratory setup. Initially problems were encountered with some libraries required by the software, and time was spent liaising with the program developer, who eventually resolved the issue.

5.1.1 Development board setup

The Palo43 expansion board was purchased prior to completion of the hardware design for the version 2 steering wheel, to allow experimentation with the gumstix Overo motherboard. The Overo was connected to the Palo43 as shown in Figure 5.2. After establishing a console connection, the operating system was booted, and various systems were tested. The wireless module was enabled and was successfully connected to the NICTA wireless system. However, trouble was experienced with sending data via Bluetooth. Some time was spent trying to debug the problem, but no solution was found. Eventually gumstix were contacted, and they advised a recent batch of Overo boards have faulty Bluetooth modules. The Overo will be returned to gumstix later this year for repair under warranty.

The Samsung LCD was installed, as shown in Figure 5.3, and the touch screen was tested and found to work well. The Overo is shipped with a working Linux

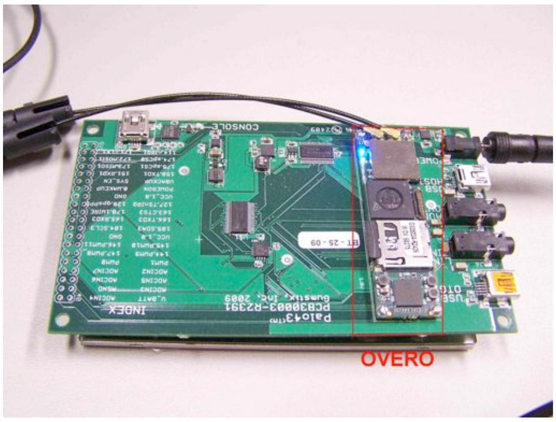

Figure 5.2: Overo motherboard sitting on Palo43 expansion board.

Figure 5.3: Samsung LCD displaying Overo boot screen.

operating system on the on board Flash memory. It was decided to leave this working system intact and do all development on bootable microSD cards. A microSD card was purchased and was appropriately partitioned and formatted. The required boot file and Linux kernel were copied to the card. The Overo successfully booted from the microSD card.

In order to cross-compile software to be run on the Overo, and configure the kernel as required, a build environment was setup on the author's computer. Gumstix uses the OpenEmbedded [49] software framework to build Linux distributions for the Overo. OpenEmbedded uses information in BitBake [5] 'recipes' to track dependencies, cross-compile packages and create binary packages. It also creates complete images of boot loaders, kernel and root file system. The build environment was setup, and a test root file system image and kernel image were created, using an Overo BitBake recipe. This initial build took approximately 24 hours to complete. As described previously, the Bluetooth module was faulty. So, in order to speed up boot time and reduce the image file size, Bluetooth was disabled and a second build was completed, in significantly less time than the initial build.

5.2 Version 1 results

Prior to the commencement of this thesis, a version of the driver controls was designed by the author using the MSP430 microcontroller. The schematic design and printed circuit board layout was done using Altium Designer 2004 and Summer08 [34]. The purpose of this version was to provide the UNSW SRT with a set of driver controls suitable for testing purposes while a more sophisticated version was being designed. It has also provided a useful way to test various user interface controls and will be used during the 2009 race. Over the course of the year this design has been realised and, in spite of some major delays, several key functions have been implemented. A photograph of the PCB is provided in Figure 5.4.

5.2.1 PCB population

Significant delays were encountered due to a manufacturing fault in the PCB (a hole was not properly plated), which caused an essential crystal oscillator to be open circuited. The MSP430 microcontroller communicates with the Microchip MCP2515 CAN controller [39] over an SPI bus. The crystal oscillator is required for this communication bus to operate correctly. The bus is used to send and receive data on the telemetry network, and is the only way to print debugging messages from the microcontroller. Since the bus was not functioning, the program running on the MSP430 could only be traced by flashing LEDs at particular points in the code. After a significant amount of testing with an oscilloscope and debugging code, the manufacturing fault was eventually found. Once repairs were made, effective communication over the SPI bus was achieved and development proceeded more rapidly. Despite the delay, significant progress was still possible and by the end of the first semester, the PCB performed the following functions:

- Receiving all of the data on the motorCAN bus and re-sending it over the telemetryCAN bus, as per the requirement in Section 1.5.
- Sending drive commands to the motor controller.

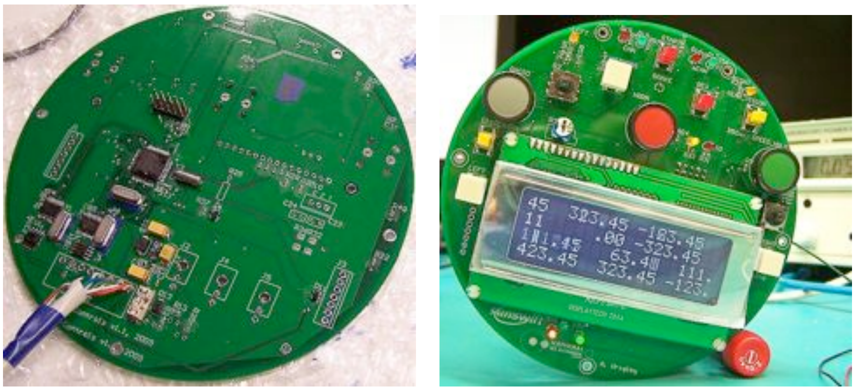

Figure 5.4: The MSP430 based, version 1 driver controls PCB developed throughout the thesis.

- Sampling the joystick throttle position and sending a corresponding motor current request to the motor controller.

The throttle was found to be too sensitive at the lower (starting) end. Scaling of the motor current was implemented so the throttle would be less responsive during initial acceleration.

5.2.2 Software development

During the second semester, further development was undertaken on the version 1 steering wheel. This was primarily software development, but also significant time was spent building the mechanical enclosure, as described in Section 5.6.

Software was written and tested to implement cruise control capability. The driver is able to accelerate to the desired speed and then hold a cruising speed. Minimum speed checks were added to ensure that a saved cruise control speed could not be accidentally engaged whilst the car is stationary. As an additional safety measure, a kill-switch function was implemented in case of erroneous behaviour. When the driver holds the kill switch a zero current and zero velocity command is sent to the motor controller. The software was written so that the regenerative braking paddle takes priority over the accelerator paddle. This is to ensure that if, in a panicked situation, the driver pulls both paddles braking will still occur.

The majority of the other buttons are self explanatory, and all had software filtering written to eliminate errors from switch bounce. The indicators, radio, horn and rear vision buttons send out channel messages on the telemetry network. Various nodes are configured to listen to these messages. For example, the switch-card node that turns the indicators on and off is configured to listen to the indicator channel messages.

The reverse button, as the name suggests, allows the car to travel in reverse. During bench testing it was found that the motor controller will obey any command to go into reverse, regardless of the speed of the motor. For example, if the motor is travelling forwards, full regenerative braking will be engaged until the speed reaches zero, then the motor will travel in reverse. This is undesirable for safety reasons, so additional checks were programmed for reverse mode. First, the reverse button is ignored if the car is travelling greater than 10 km/h. Second, the button must be held for 3 seconds before engaging, to ensure the driver does not accidentally engage reverse. Finally, the maximum reversing speed is set to 30 km/h.

The start button turns the motor controller on or off by sending a channel message to the DC/DC converter node to provide power to the motor controller. The driver control system then waits for a return message from the DC/DC converter before sending any drive commands to the motor controller. If the start button is pressed again, power is disconnected from the motor controller. Like the reverse button, the start button must be held for 3 seconds to ensure the driver does not accidentally turn the motor on or off. Note that if there is an error with the driver controls and no commands are sent to the motor controller for 250 ms, the watchdog timer in the motor controller will automatically shut it down. This is an important safety feature, as otherwise the motor controller could become 'stuck' executing the last command received from the driver controls prior to failure.

The version 1 steering wheel contains a 20x4 cell character LCD for displaying information to the driver. The MCU is programmed so the display is broken up into 12 equal segments, each displaying different data from the solar car. Four of the segments are hard-coded to display information from the driver controls, such as set speed and current speed. The remaining eight segments are set up as in-channels, and so can easily be configured at any time. These in-channels can be set to listen to various nodes in the car, for example, the negsum node, which sends out current and voltage data for the solar array, battery and motor. The working LCD can be seen in Figure 5.11. If data from a configured in-channel is not received after approximately 3 seconds, a '!' character is displayed to indicate the data is old.

5.2.3 Use in Sunswift IV

All of the functions discussed in this section were tested and debugged on the bench. Once the the solar car was constructed, the version 1 steering wheel was installed in the solar car and tested. At the time of writing, the Sunswift team and solar car are on their way to the World Solar Challenge. Prior to departure, the car was tested three times and the steering wheel was found to work well. On the first testing run, the team reported cruise control was not working. This was found to be because of a last minute programming change by another team member, which resulted in the velocity data not being acquired from the motor controller. Once this was resolved, cruise control was tested and worked without issue.

Finally, a complete spare set of driver controls were assembled in case it was required on the race. The manufacture of the spare steering wheel was considerably quicker than the first, as most of the software development was already complete. However, one significant problem was encountered with the MSP430 MCU. The MCU initially programmed without error, but within a day it could no longer be programmed. After spending considerable time working through all other options, the MSP430 was eventually replaced, which solved the problem. It is believed the original MSP430 was either faulty or suffered ESD damage. The soldering of the spare PCB, mechanical manufacture and debugging took approximately two weeks of the author's time.

5.3 Version 2 results

5.3.1 PCB manufacture

Following completion of the board design, the version 2 PCB was sent to PCB-Cart [46] for manufacture. A PCB panel file was created by the author with three copies of the steering wheel version 2 board, and other PCB designs from another student at NICTA. The delivery of the PCBs was unexpectedly delayed by approximately two weeks, for two reasons. Firstly, time was spent waiting for the other PCB design to be complete, (and time was also required to integrate the two designs into the one PCB file), and secondly, the PCB manufacturer required clarification on several different aspects of the board design. These were resolved through back and forth emails, but delayed the manufacturing process.

When the blank PCBs (shown in Figures 5.5 & 5.6) arrived, the impedance of the critical tracks was tested with a digital multimeter. The manufacturer does

Figure 5.5: Photograph of top side of unpopulated version 2 PCB.

perform quality control testing, but given the problems experienced with the first PCB, it was deemed prudent to check prior to commencement of soldering. Unfortunately, due to the design taking significantly longer than anticipated and the manufacturing delays, very little time remained to populate the PCB and commence testing.

5.3.2 PCB population

Several components selected in the design are only available in *quad flat no-lead* (QFN) or *micro-lead frame* (MLF) packages. In these packages, the leads do not extend out from the package sides as in conventional packages. The majority (and sometimes all) of the electrical contact between pads is underneath the chip, making hand soldering extremely difficult, if not impossible. Another student at NICTA had previously attempted to solder such packages using solder paste with a stencil. However, this proved unsuccessful. One of the difficulties was in ensuring the solder paste did not bridge across pads and cause shorting. Additionally, solder paste needs to be heated to a minimum of $220^o\ C$, which places significant stress on the PCB, and resulted in delamination of the fibreglass and copper. For these reasons, I decided to experiment with a slightly different method that did not require the use of solder paste or a stencil. This method, which uses standard solder and a reflow machine, proved successful and is detailed in Appendix D.

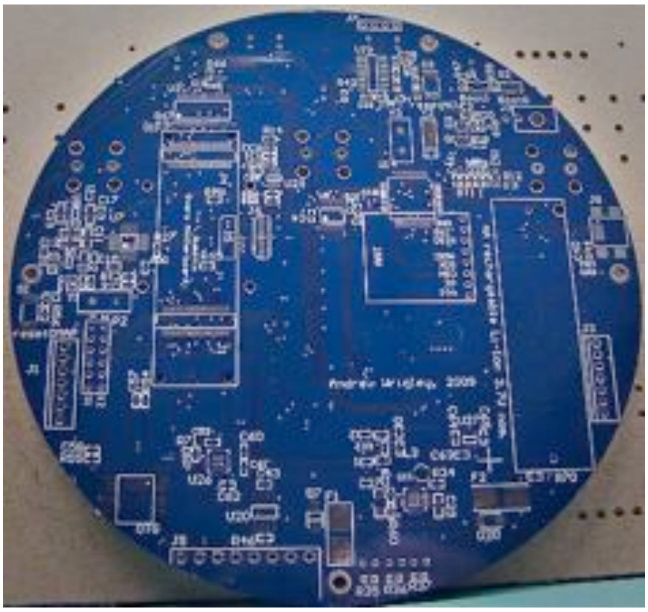

Figure 5.6: Photograph of bottom side of unpopulated version 2 PCB.

5.3.3 Design errors

During the initial population of the PCB several minor design errors were discovered. They are briefly discussed here to assist anyone wishing to design a revision. First, the MIC5247 1.8V regulator enable pin is active high, but was mistakenly designed for active low. Therefore the pin was connected to ground instead of 3.3V. A modification was made to the PCB to correct this fault, and the 1.8V regulator works without issue. This should be corrected in any future design revision.

Second, the TPS62111 3.3V regulator enable line is pulled to 3.3V, not 12V. This was a last minute change in the design to allow the regulator to be turned off via a GPIO on the Overo. However, since the regulator is responsible for generating 3.3V, no 3.3V power is present to enable the IC, resulting in a catch 22 situation. If the battery cell is connected this is not an issue, as 3.3V power will be generated from the REG113-33 IC. However this is not ideal, as the board should be capable of functioning without the battery cell. A simple solution to this problem is to pull the enable pin to 12V, and use a *metal-oxide-semiconductor field-effect transistor* (MOSFET) to pull it to ground via a 3.3V GPIO, as shown in Figure 5.7. This way, the TPS62111 will be enabled by default when 12V power is present, but still can be turned off by the Overo if desired.

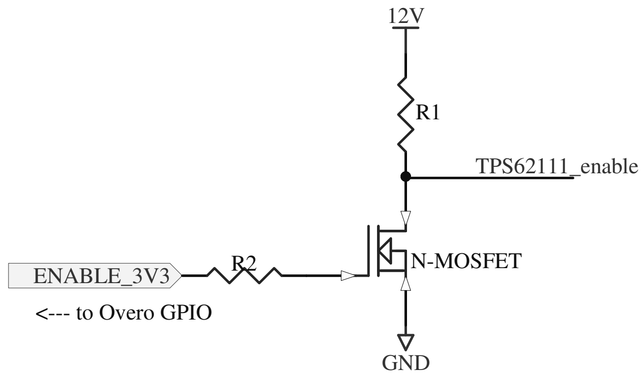

Figure 5.7: Solution to design error in 3.3V regulator enable pin.

Third, the MIC2297 LED driver was placed on the top layer of the PCB, whereas the other no-lead packages were placed on the bottom layer. This meant that the MIC2297 could not be soldered using the reflow solder method described in Appendix D, but had to instead be soldered by hand. This proved very difficult to achieve. The tolerance between the pads underneath the component and the ground pad on the PCB is approximately 0.15 mm. If the component is not placed in exactly the correct position, the pads are likely to short to ground. Several different soldering techniques were attempted in order to overcome this problem. The shorting was eliminated by putting only a small amount of solder on the ground pad to elevate it and thus increase the clearance.

However, this solution created a new problem. When tested, the MIC2297 worked briefly, but then immediately attempted to draw a large current. The chip was removed and tested, and was found to have an internal short between 3.3V and ground. A replacement chip was soldered on, but the same problem occurred. It is believed the problem is with the ground pad on the MIC2297 being poorly connected to the ground pad on the PCB. Using a soldering iron, it is virtually impossible to get the required heat into the ground pad to liquefy the solder and achieve a high quality connection, due to the large ground plane on the PCB dissipating the heat. The purpose of the ground pad is for thermal relief of the IC. Without this thermal relief the IC rapidly heats up and suffers complete failure.

If a design revision is performed, the MIC2297 should ideally be replaced with an LED driver with a leaded package. However, this could be difficult as many ICs that are commonly used in portable devices are now only available in a no-lead package due to space constraints. Alternatively, the MIC2297 could be moved to the same layer as the other no-lead packages and the size of the ground pad on the PCB reduced slightly. This should allow soldering using the reflow solder method described in Appendix D.

The issues encountered with the MIC2297 significantly delayed progress in populating the PCB. The chip is a critical component for the LCD, and when it fails the 3.3V line is shorted to ground, as described previously. Therefore it was decided that no further components should be soldered to the board until the problem is fully resolved. The author is actively working on a suitable solution, but is awaiting the delivery of additional MIC2297 ICs.

5.4 Fulfilment of Requirements

In Section 1.5, the key requirements for the system were outlined. The system developed fulfils those requirements in the following ways:

- *Provide a user interface to allow the driver to control all aspects of driving the car.* This has been implemented and tested successfully in the solar car with the version 1 steering wheel. Drivers of the solar car have reported the user interface was intuitive and they were able to learn the controls quickly. The same controls were designed for the version 2 steering wheel.
- *Display real time data and graphs of key variables such as array power, motor power and battery state of charge.* Whilst the software has not yet been written to achieve this objective, the high resolution LCD and powerful gumstix Overo motherboard provide the required hardware to ensure this will be possible.
- *Provide a bidirectional communication link between the telemetryCAN and motorCAN.* This has been implemented and tested successfully with the version 1 steering wheel. Some issues have been experienced with CAN messages from the motorCAN bus being missed. It is believed this is due to the MSP430 not being able to service interrupts from both networks fast enough before the input buffers on the CAN controllers overflow. The system designed in the version 2 steering wheel should resolve this issue. Firstly, because the CAN controllers are part of the MCU so latency should be reduced, and secondly, the STM32 operates at significantly higher processing speeds than the MSP430.
- *Be able to perform data analysis and storage.* The gumstix Overo will provide more than sufficient processing power to run software written by

the strategy team. A microSD card, capable of storing telemetry data, has been tested with the Overo and found to work well.

- *Accept a rear vision video camera signal and display the video in real time on the screen.* The hardware for this objective has been designed into the version 2 steering wheel, and is detailed in Section 4.4. A separate rear vision screen is installed in the car for use with the version 1 steering wheel.

- *Allow a real time speed control loop (adaptive cruise control) to be implemented.* Whilst no specific software has been written, the Overo will be capable of performing any required calculations for the control loop. In addition, a control loop structure has been proposed. The author spent some time trying to further develop the control system, but little progress was made. The aim was to derive a transfer function to represent the solar car, and then design a controller that would ensure stable operation and good output response characteristics. Deriving the solar car transfer function was attempted using time-domain techniques, which are superior to frequency-domain techniques as they can be applied to non-linear, time-variant systems. A standard text, Nise [41], was consulted, but it unfortunately considers only linear, time-invariant systems, even for time-domain modelling. The key parameters of the solar car are generally non-linear and time-variant. For example, the motor controller and motor efficiency change drastically with speed, load and temperature. The development of a complete, well understood, and functional control system for the solar car appears worthy of an undergraduate thesis topic in itself.

5.5 Brake Sensor

The brake sensor board, detailed in Section 4.7, was sent for manufacture with a panel of other Sunswift PCBs to BEC Manufacturing [37]. The fabricated PCB was soldered by other members of the electrical team and some simple code was written to handle a microswitch input. The force sensor was not purchased due to financial and time constraints, so this has not been tested. However, the microswitch input works well and has been tested successfully in the solar car. The steering wheel has been configured to listen to messages from the brake sensor node, and a zero current and zero velocity command is sent to the motor controller if the switch is activated. The switch-card node that controls the rear brake lights is also configured to listen to the brake sensor node.

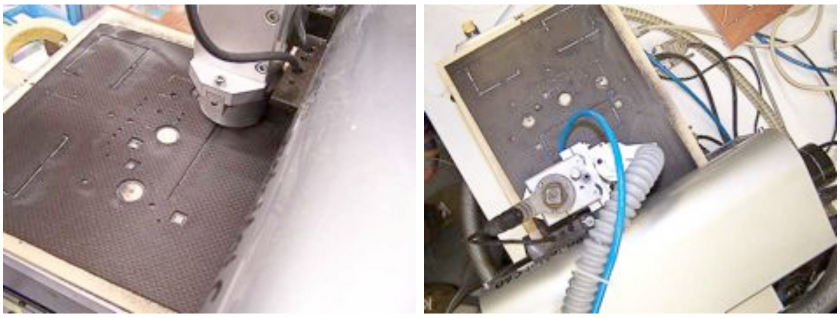

Figure 5.8: Photograph of carbon fibre plate being milled with a PCB milling machine.

5.6 Mechanical enclosure

In addition to the electronics, this project also required the design and manufacture of the mechanical enclosure for the steering wheel. This will be only briefly discussed, as the electronics are the primary focus of this document.

The enclosure is made from two carbon fibre composite plates, with the PCB sandwiched in the middle. The carbon fibre plate was manufactured at Boeing Hawker deHavilland [13] by members of the Sunswift composites team. The backing plate then had to be cut out to allow mounting of the quick release steering hub and access to the external connectors. The front cover plate had to be cut out to fit all the buttons and the LCD display. To ensure accuracy this was done using a 3-axis PCB milling machine. A mechanical design was drafted in Altium Designer and exported as a routing path file for the milling machine. The design can be seen in Appendix C.

It took approximately one day to setup the machine and mill the first steering wheel. The spare steering wheel was able to be milled in approximately half a day, as the setup time was greatly reduced. The milling of the plate can be seen in Figure 5.8. Complete assembly of the steering wheel was quite time consuming. The milling machine is unable to cut an exact 90 degree corner, so the holes for some of the buttons had to be filed until square. Whenever any filing or cutting needed to be done all of the electronics had to be removed, as carbon dust is conductive and therefore needs to be kept away from the PCB. Once the filing was complete, the plate had to be thoroughly cleaned of all carbon dust, then the electronics could be re-mounted. When the filing or cutting was not enough, then the process had to be repeated. The clearance between the PCB and the carbon plate is quite small, and so time was also spent ensuring no conductive components contacted the carbon. This clearance can be seen in Figure 5.10.

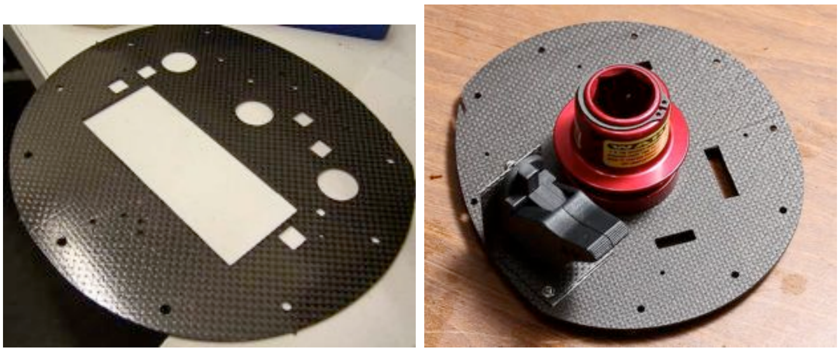

Figure 5.9: Photograph of completed steering wheel front cover (left image) and back cover (right image). The quick release and accelerator pedal are mounted on the back cover.

Figure 5.10: Photograph showing tight clearance between PCB and carbon plate. (*Photo courtesy Cameron.G.Cooke.*)

Figure 5.11: Photograph of steering cover mounted onto electronics. (*Photo courtesy Cameron.G.Cooke.*)

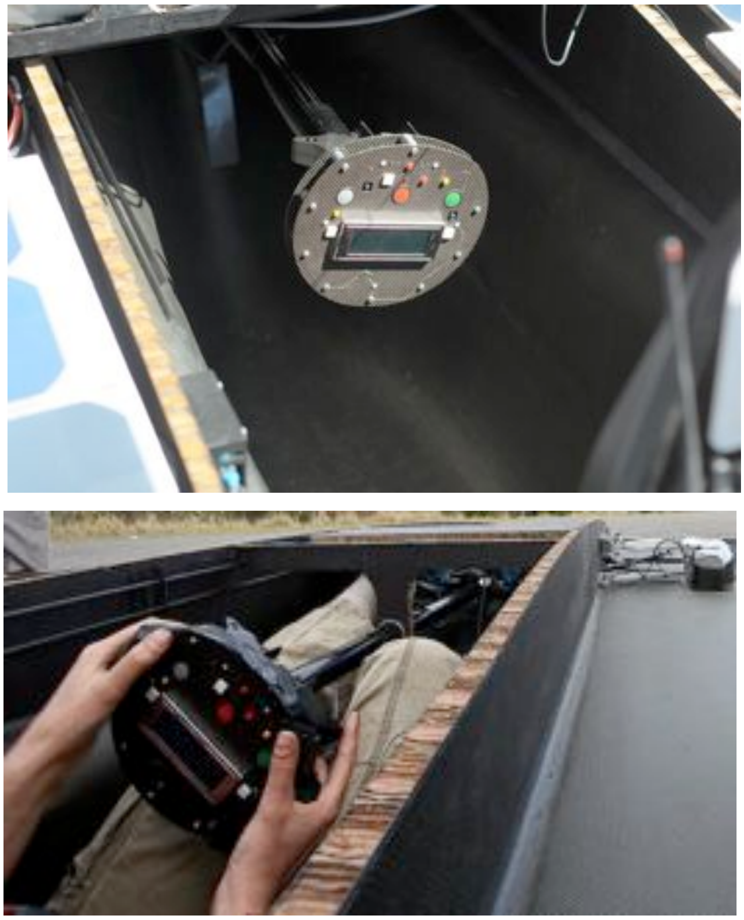

Figure 5.12: Photograph of completed version 1 steering wheel mounted in Sunswift IV. (*Top photo courtesy Matt Cumming* [18], *bottom photo courtesy Cameron. G. Cooke.*)

Chapter 6

Future Work and Conclusions

The aim of this thesis was to design an electronic system that forms part of the steering wheel of the UNSW Solar Racing Team's newest vehicle, Sunswift IV. The system was required to control the motor and operate ancillary vehicle functions, communicate on two different CAN buses, and allow the implementation of a graphical display, automated speed control and data logging.

The requirement to provide a user interface to allow the driver to control all aspects of the car has been met. The requirement to provide a bidirectional link between the telemetryCAN and motorCAN has also been met. These requirements have been fully implemented in the working steering-integrated driver control system that has been developed and in use in the Sunswift IV solar car. This system will be used during the 2009 World Solar Challenge. Although it was not specified in the initial thesis aims, a spare of these driver controls has also been completely assembled, tested and is ready for use in the solar car, if required.

The design of the version 2 system has been completed, which means that all other requirements can now be satisfied, specifically - real time graphing of data, a real time speed control loop, data storage and a fully integrated rear vision system. However, population of the board was not complete, for reasons discussed in Chapter 5, and so further development, discussed in Section 6.1, is required before the system can be made fully operational, thoroughly tested, and subsequently used on a race.

6.1 Further development

The hardware development for the version 2 integrated steering wheel has been completed. However, only minimal software development has been achieved, and so this would likely be the focus of any future work. The version 2 PCB population should be completed, after which time the software development can be tested on-board. The software for the ARM microcontroller and the gumstix Overo can be developed in parallel, by two different team members if desired, as the Palo43 expansion board allows testing of the Overo without the steering wheel PCB.

If any hardware revision is performed in the future, the design errors detailed in Section 5.3.3 should be corrected. These errors should be relatively trivial to rectify, and possible solutions have been provided in Section 5.3.3. Once the system is in a 'working state', a thorough testing program should follow to ensure the system is both reliable and safe before installation in the solar car.

The initial work on the adaptive cruise control system has been presented. However, there is a significant amount of further work and refinement that could be performed to develop and implement this control system. Such development should be completed incrementally, with thorough bench testing and in-car testing at each stage. This will ensure the system can be made reliable and effective.

6.2 Use for external parties

It is hoped that the work achieved in this thesis will be useful to other solar car teams who require a similar system. The Tritium Wavesculptor, the motor controller the system in this thesis is designed to interface with, is by far the most common among teams competing in the World Solar Challenge. Therefore all the hardware and software related to driving the motor should be directly transferable. Any team wishing to use this system may need to modify the hardware and software relating to telemetryCAN, in order to interface with their own telemetry system.

With the increasing use of electric motors to drive cars, particularly in hybrid vehicles, the system developed in this thesis could have widespread applications. Electric cars have a limited energy budget, based on the size of their battery pack, and so the efficient management of that energy will be critical. Unfortunately, we cannot trust the driver to appropriately manage this, so electronic systems will need to be in place to ensure the vehicle is driven in the most efficient manner possible. The system developed in this thesis will allow energy management strategies to easily be implemented.

6.3 Final comments

Through continual development of the system designed in this thesis, the UNSW Solar Racing Team could gain a significant competitive advantage over other solar car teams. However, it is the belief of the author that all research and development in the solar car racing field should be open for use by all teams. This will ensure designs are continually developed and improved upon, mistakes are not repeated, and solar car racing might reach its true goal of making sustainable transport a reality.

Bibliography

[1] Beagle Board homepage. http://beagleboard.org, 2009.

[2] PHYTEC America. phyCORE i.MX31 Development Kit. http://www.phytec.com/products/rdk/ARM-XScale/phyCORE-ARM11-iMX31.html, 2009.

[3] Analog Devices Inc, http://focus.ti.com/lit/ds/symlink/reg113-25.pdf. *ADXL335 datasheet*, 2009.

[4] MathWorks Australia. MATLAB. http://www.mathworks.com.au/products/matlab/, 2009.

[5] BitBake. BitBake homepage. http://developer.berlios.de/projects/bitbake/, 2009.

[6] BMW. Active Cruise Control. http://www.bmw.com/com/en/insights/technology/technology_guide/articles/active_cruise_control.html, 2009.

[7] South Australian Motor Sport Board. World Solar Challenge Race Regulations. http://www.globalgreenchallenge.com.au/assets/pdfs/wsc/2009.general.regulations.pdf, 2009.

[8] South Australian Motor Sport Board. World Solar Challenge Results History. http://www.globalgreenchallenge.com.au/wsc-evolution, 2009.

[9] South Australian Motor Sport Board. World Solar Challenge Technical Regulations. http://www.globalgreenchallenge.com.au/assets/pdfs/wsc/2009.technical.regulations.pdf, 2009.

[10] L. Cocco and P.Daponte. Metrology and Formula One Car. *IEEE International Instrumentation and Measurement*, pages 755–760, 2008.

[11] Group T International University College. Umicore Solar Team. http://www.solarteam.be/en, 2009.

[12] Creative Commons. Attribution-Share Alike 2.5 Generic. http://creativecommons.org/licenses/by-sa/2.5/, 2009.

[13] The Boeing Company. Boeing Australia homepage. http://www.boeing.com.au, 2009.

[14] LSI Corporation, Hewlett-Packard Company, Intel Corporation, Microsoft Corporation, NEC Corporation, and ST-NXP Wireless Company. On-The-Go and Embedded Host Supplement to the USB Revision 2.0 Specification. http://www.usb.org/developers/onthego/USB_OTG_and_EH_2-0.pdf, 2009.

[15] Toyota Motor Corporation. Radar Cruise Control. http://www.toyota.co.jp/en/safety_presen/tech/38.html, 2009.

[16] Paul Vincent Craven. Solar Car Power Management. Master's thesis, University of Missouri-Rolla, http://www.cravenfamily.com/thesis/table_of_contents.html, 1996.

[17] CSIRO. In-wheel motor technical documents. http://www.csiro.au/resources/pf11g.html, 2009.

[18] Matt Cumming. Matt Cumming blog page. http://downsouthsurfing.posterous.com/introducing-ivy, 2009.

[19] Federation Internationale de l'Automobile. FIA Formula One World Championship Technical Regulations. http://www.fia.com/en-GB/sport/regulations/Pages/FIAFormulaOneWorldChampionship.aspx, 2009.

[20] Sparkfun Electronics. Sparkfun Electronics homepage. http://www.sparkfun.com, 2009.

[21] B. Fleming. Endangered Technologies. *IEEE Vehicular Technology Magazine*, 2(3):58–60, 2007.

[22] FTDI Chip, http://www.ftdichip.com/Documents/DataSheets/DS_FT232R_V204.pdf. *FT232RL datasheet*, 2009.

[23] CadSoft Computer GmbH. CadSoft homepage. http://www.cadsoftusa.com/index.htm, 2009.

[24] Robert Bosch GmbH. CAN specification 2.0B. http://www.semiconductors.bosch.de/pdf/can2spec.pdf, 2009.

[25] Hirose, http://portal.fciconnect.com/res/en/pdffiles/doc_search/62684.pdf. *FH12-45 datasheet*, 2009.

[26] Freescale Semiconductor Inc. Freescale homepage. `http://www.freescale.com`, 2009.

[27] Gumstix Inc. gumstix computer-on-module. `http://www.gumstix.com/store/catalog/motherboards.php`, 2009.

[28] Gumstix Inc. Palo43 Expansion Board. `http://www.gumstix.com/store/catalog/product_info.php?products_id=229`, 2009.

[29] Gumstix Inc. Palo43 expansion board schematics. `http://pubs.gumstix.com/boards/PALO/PALO43/`, 2009.

[30] Texas Instruments Inc. MSP430 MCU homepage. `http://www.msp430.com`, 2009.

[31] Texas Instruments. Texas Instruments homepage. `http://www.ti.com/`, 2009.

[32] Intersil Americas Inc., `http://www.intersil.com/data/fn/fn4805.pdf`. *ICL3232 datasheet*, 2006.

[33] Kyocera Elco, `http://www.kyocera-elco.com/prdct/pdf/5602.pdf`. *AVX-5602 datasheet*, 2009.

[34] Altium Limited. Altium Designer. `http://www.altium.com/products/altium-designer/`, 2009.

[35] ARM Ltd. ARM documentation. `http://infocenter.arm.com/help/index.jsp`, 2009.

[36] Tritium Pty. Ltd. Tritium Wavesculptor Motor Controller. `http://www.tritium.com.au/products/TRI50/index.html`, 2009.

[37] B.E.C. Manufacturing. B.E.C. Manufacturing homepage. `http://www.becman.com.au/`, 2009.

[38] Micrel Inc., `http://www.micrel.com/_PDF/mic2297.pdf`. *MIC2297 datasheet*, 2008.

[39] Microchip Technology Inc, `http://ww1.microchip.com/downloads/en/DeviceDoc/21801d.pdf`. *MCP2515 datasheet*, 2005.

[40] Nicholas Navet. *Automotive Embedded Systems Handbook*. Taylor & Francis Ltd, 2008. ISBN 084938026X.

[41] Norman S Nise. *Control Systems Engineering*. John Wiley & Sons, Inc., 2004. ISBN 0-471-44577-0.

[42] Nordic Semiconductor, http://www.nordicsemi.no/files/Prod_brief_RFSilicon_nRF24L01.pdf. *nRF24L01 datasheet*, 2009.

[43] NXP Semiconductors, http://www.nxp.com/acrobat_download/expired_datasheets/PESD5V0X1BA_PESD5V0X1BL_1.pdf. *PESD5V0X datasheet*, 2008.

[44] Delft University of Technology. Nuon Solar Team homepage. http://www.nuonsolarteam.nl/, 2009.

[45] M. Parent. Advanced Urban Transport: Automation Is on the Way. *IEEE Intelligent Systems*, 22(2):9–11, 2007.

[46] PCBCART. PCBCART homepage. http://www.pcbcart.com, 2009.

[47] Pegasus Auto Racing Supplies, http://www.pegasusautoracing.com/productselection.asp?Product=3400. *Hex Type Quick-Release Steering Wheel Hub*, 2009.

[48] Penny and Giles. JC120 joystick datasheet. http://www.pennyandgiles.com/docGallery/97.PDF, 2009.

[49] OpenEmbedded Project. OpenEmbedded homepage. http://www.openembedded.org, 2009.

[50] IOP Publishing. Dutch solar car speeds ahead. http://environmentalresearchweb.org/cws/article/futures/32031, 2007.

[51] Peter Pudney. *Optimal energy management for solar-powered cars*. PhD thesis, University of South Australia, http://scg.levels.unisa.edu.au/Pudney/thesis, August 2000.

[52] D.M. Roche, A.E.T. Schinckel, J.W.V. Storey, C.P. Humphris, and M.R. Guelden. *Speed of Light*. Photovoltaics Special Research Centre, UNSW, 1997. ISBN 0-7334-1527-X.

[53] Samsung Electronics, http://pubs.gumstix.com/documents/Displays/LTE430WQ-F0C_PI_Rev001_GENERAL.PDF. *LTE430WQ-F0C datasheet*, 2006.

[54] Radio Communications Sector. International Telecommunication Union. http://www.itu.int/ITU-R, 2009.

[55] David Snowdon. Hardware and Software Infrastructure for the Optimisation of Sunswift II. http://snowdon.id.au/thesis/index.html, November 2002.

[56] SourceForge. gumstix community mailing list. https://lists.sourceforge.net/lists/listinfo/gumstix-users, 2009.

[57] STMicroelectronics, http://www.st.com/stonline/products/literature/ds/15274.pdf. *STM32F105xx and STM32F107xx datasheet*, 2009.

[58] STMicroelectronics, http://www.st.com/stonline/products/literature/an/14156.pdf. *STM32F105xx and STM32F107xx system memory boot mode - Application note AN2662*, 2009.

[59] UNSW Solar Racing Team. UNSW Solar Racing Team homepage. http://www.sunswift.com, 2009.

[60] F1 Technical. Steering Wheel. http://www.f1technical.net/articles/30, 2009.

[61] Texas Instruments, http://www.ti.com/litv/pdf/sleu044. *TVP5150A Quick Start Guide*, 2003.

[62] Texas Instruments, http://www.ti.com/lit/gpn/sn65hvd232. *SN65HVD232 datasheet*, 2009.

[63] Texas Instruments, http://www.ti.com/lit/gpn/tps62111. *TPS62111 datasheet*, 2009.

[64] Texas Instruments Inc., http://focus.ti.com/lit/ds/symlink/tvp5150a.pdf. *TVP5150A datasheet*, 2003.

[65] Texas Instruments Inc., http://focus.ti.com/lit/ds/symlink/ads7846.pdf. *ADS7846 datasheet*, 2005.

[66] Texas Instruments Inc., http://focus.ti.com/lit/ds/symlink/reg113-25.pdf. *REG113 datasheet*, 2005.

[67] Texas Instruments Inc., http://focus.ti.com/lit/ds/symlink/sn74avcb164245.pdf. *SN74AVCB datasheet*, 2005.

[68] Texas Instruments Inc., http://focus.ti.com/lit/ds/symlink/bq24014.pdf. *BQ24012 datasheet*, 2008.

[69] Texas Instruments Inc., http://focus.ti.com/lit/ds/symlink/pca9306.pdf. *PCA9306 datasheet*, 2008.

[70] J. Waldo. Embedded Computing and Formula One Racing. *IEEE Pervasive Computing*, 4(3):18–21, 2005.

[71] K. Wong, C. Rizos, and UNSW School of Geomatic Engineering. Surveying the Stuart Highway with GPS for the 1996 World Solar Challenge. *The Australian Surveyor*, 41(3):194–203, 1996.

Appendix A

Steering Wheel Schematics and PCB Artwork

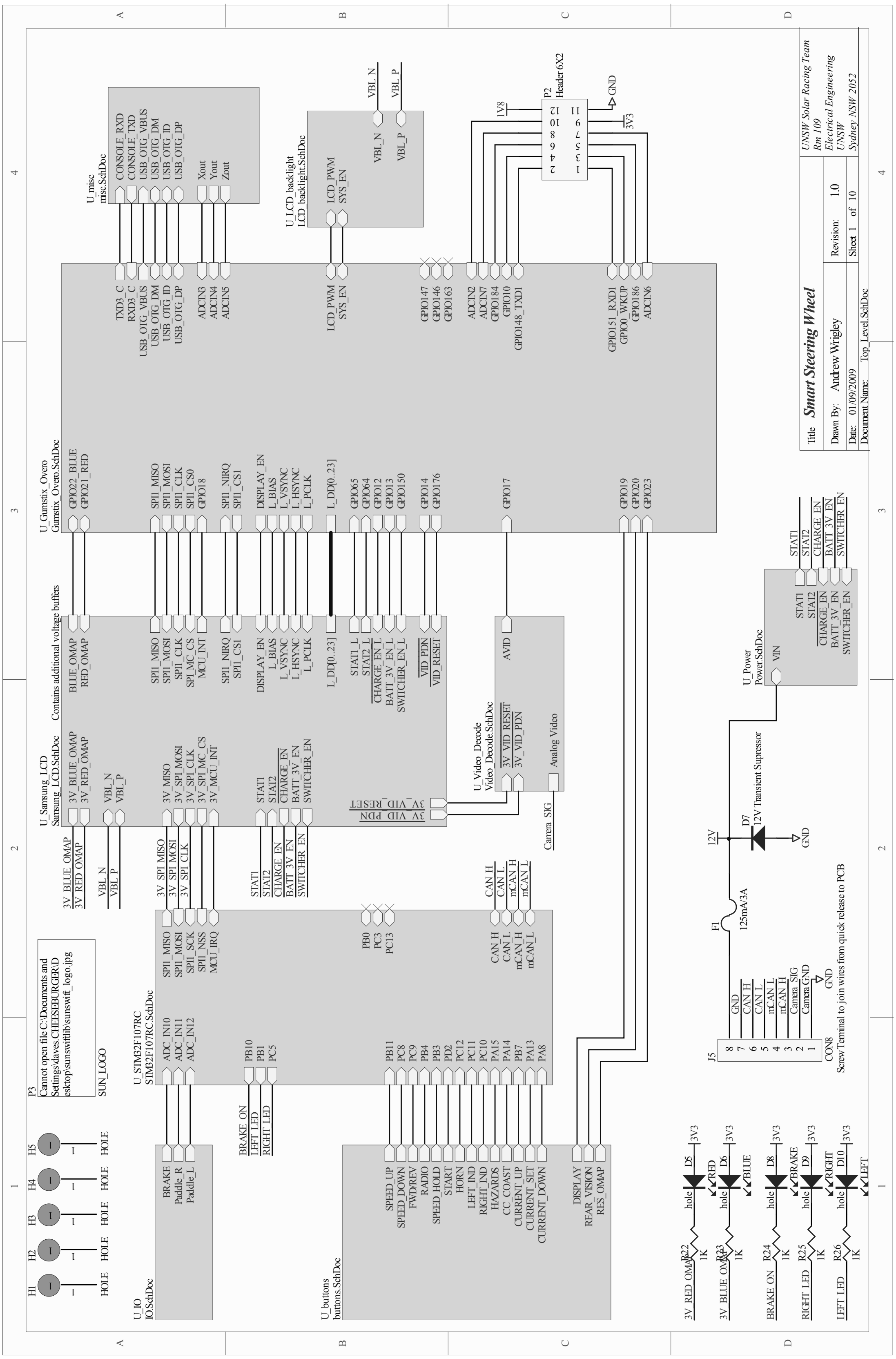

Figure A.1: Steering Wheel schematic: Top Level.

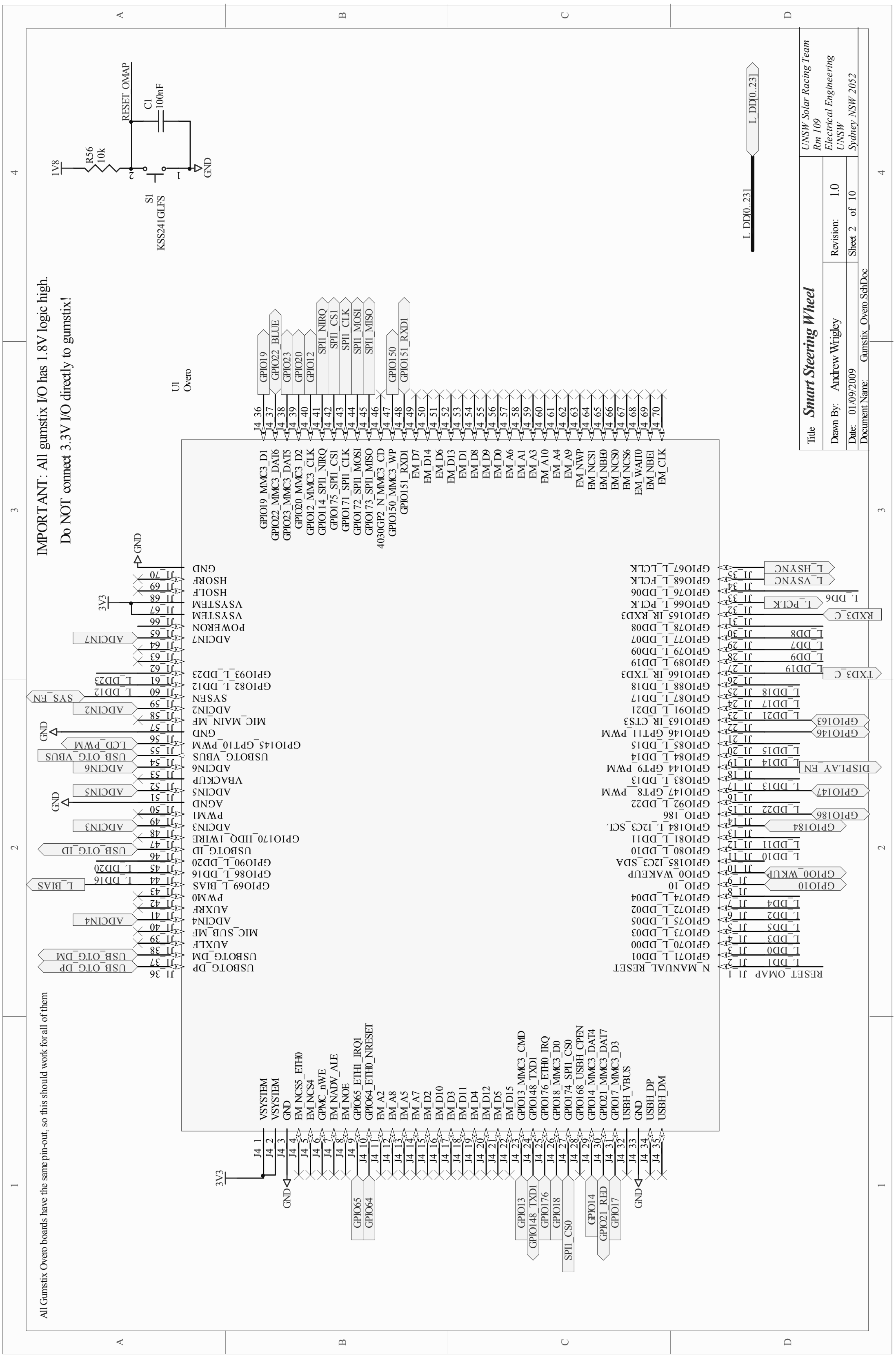

Figure A.2: Steering Wheel schematic: Gumstix Overo.

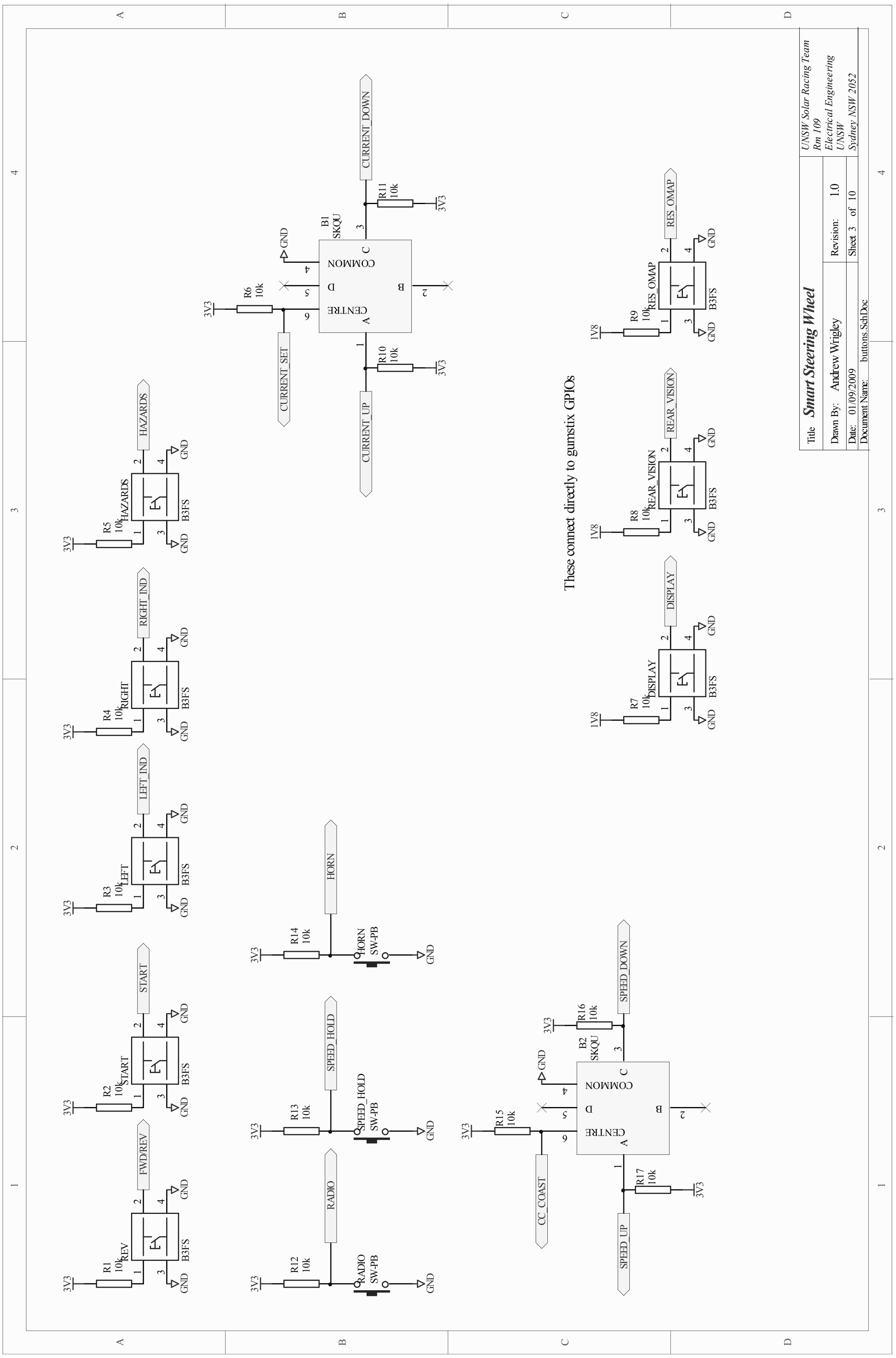

Figure A.3: Steering Wheel schematic: Input buttons.

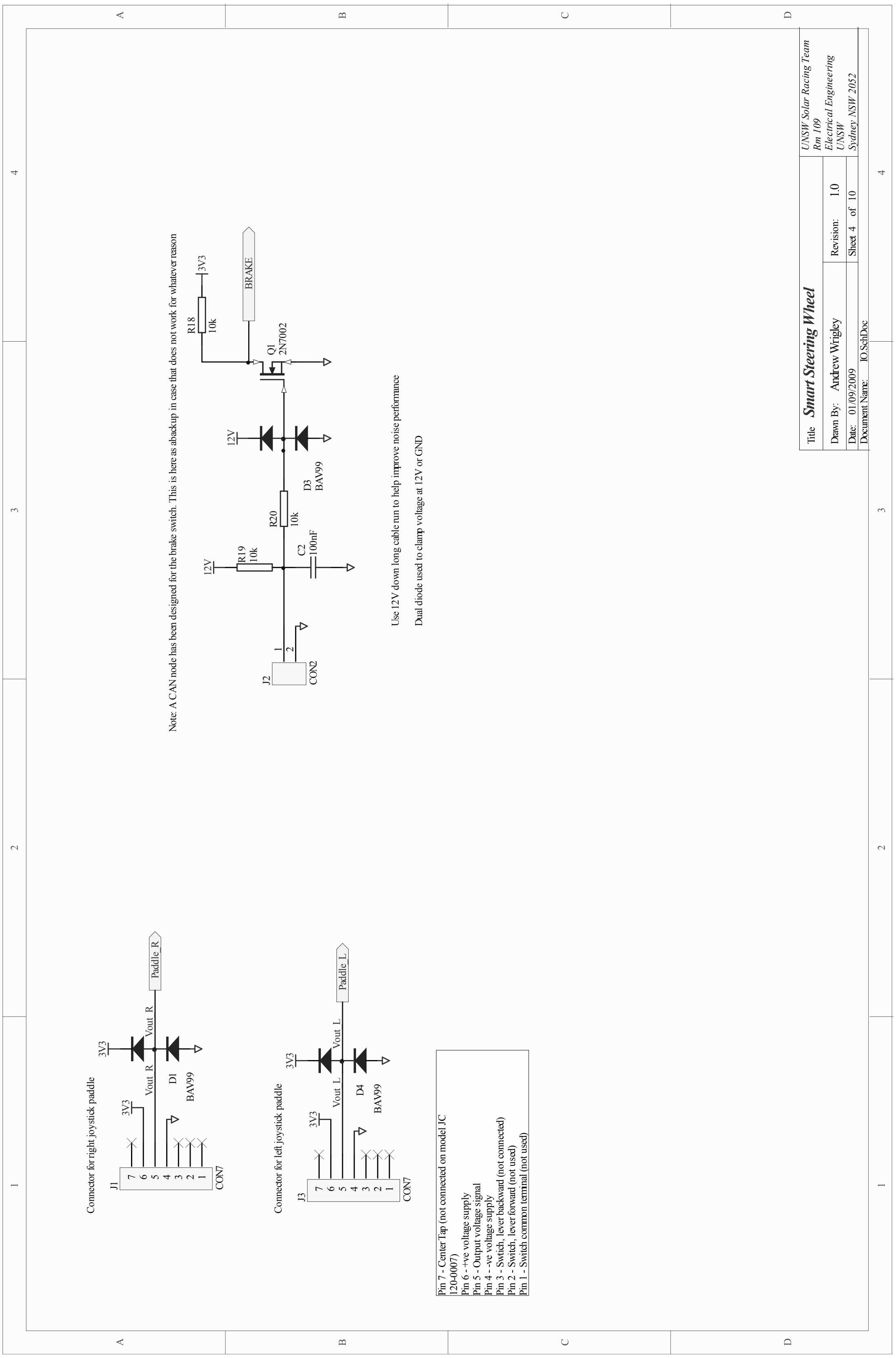

Figure A.4: Steering Wheel schematic: External I/O.

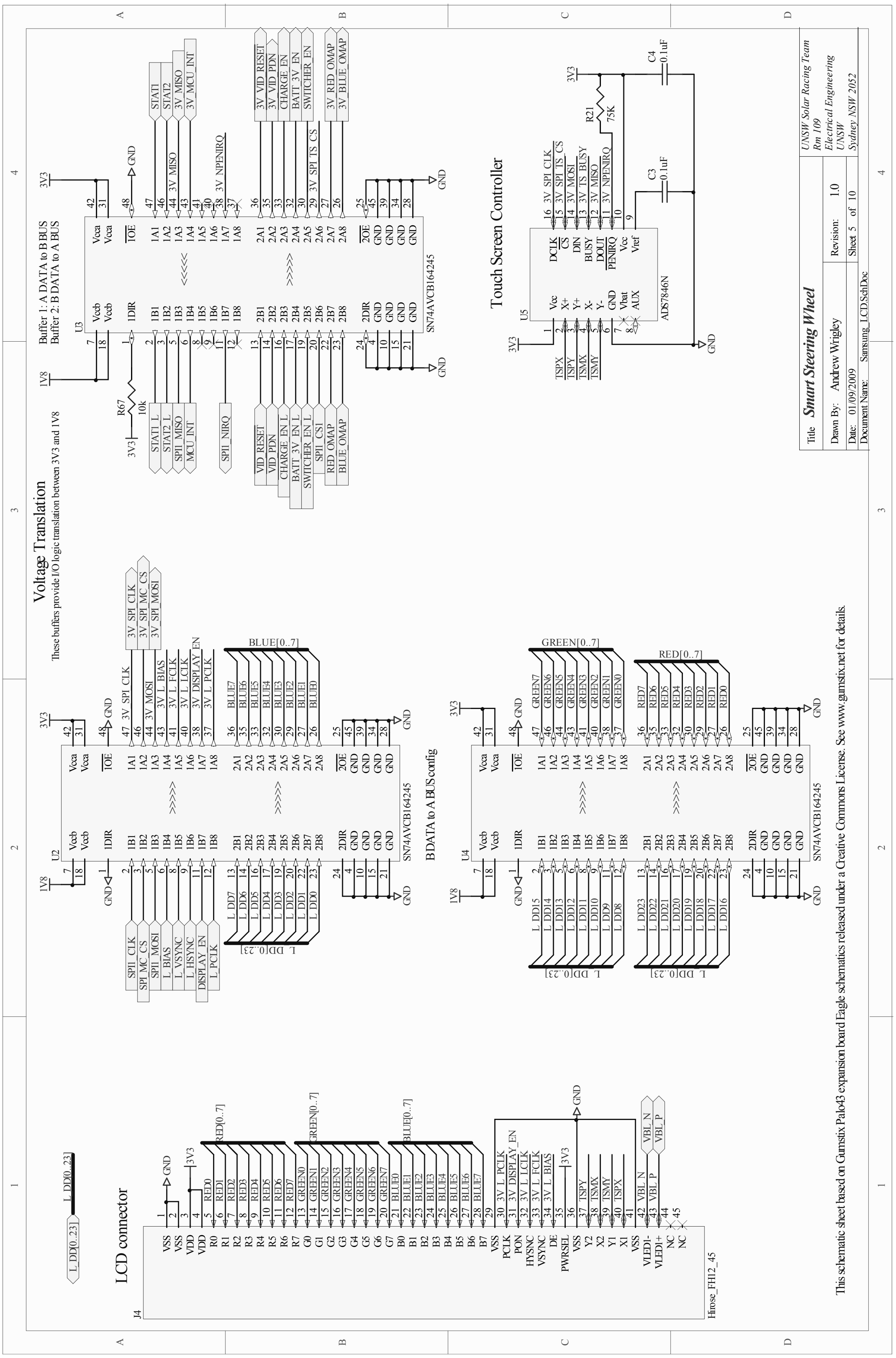

Figure A.5: Steering Wheel schematic: Samsung LCD.

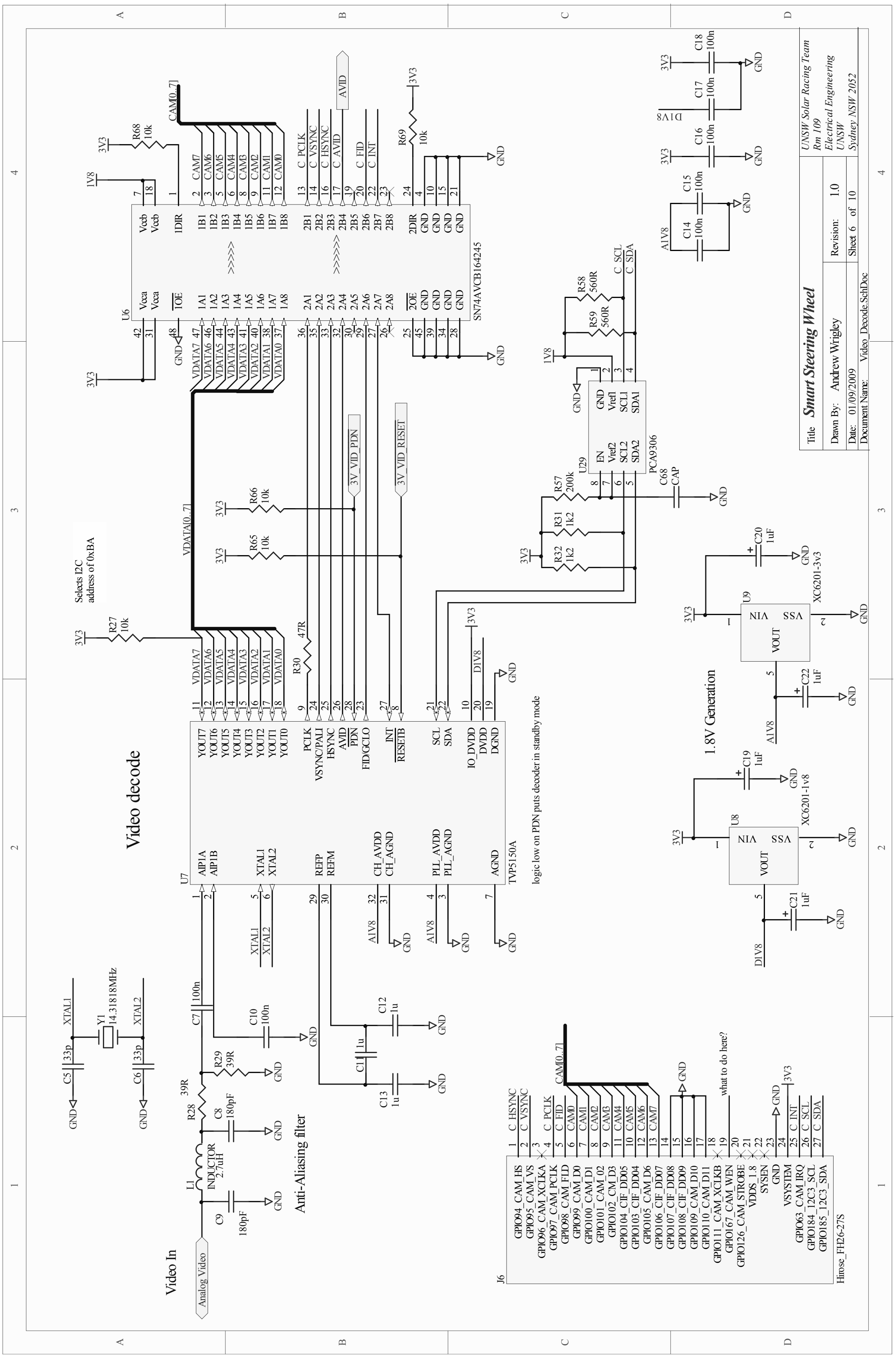

Figure A.6: Steering Wheel schematic: Video decoding.

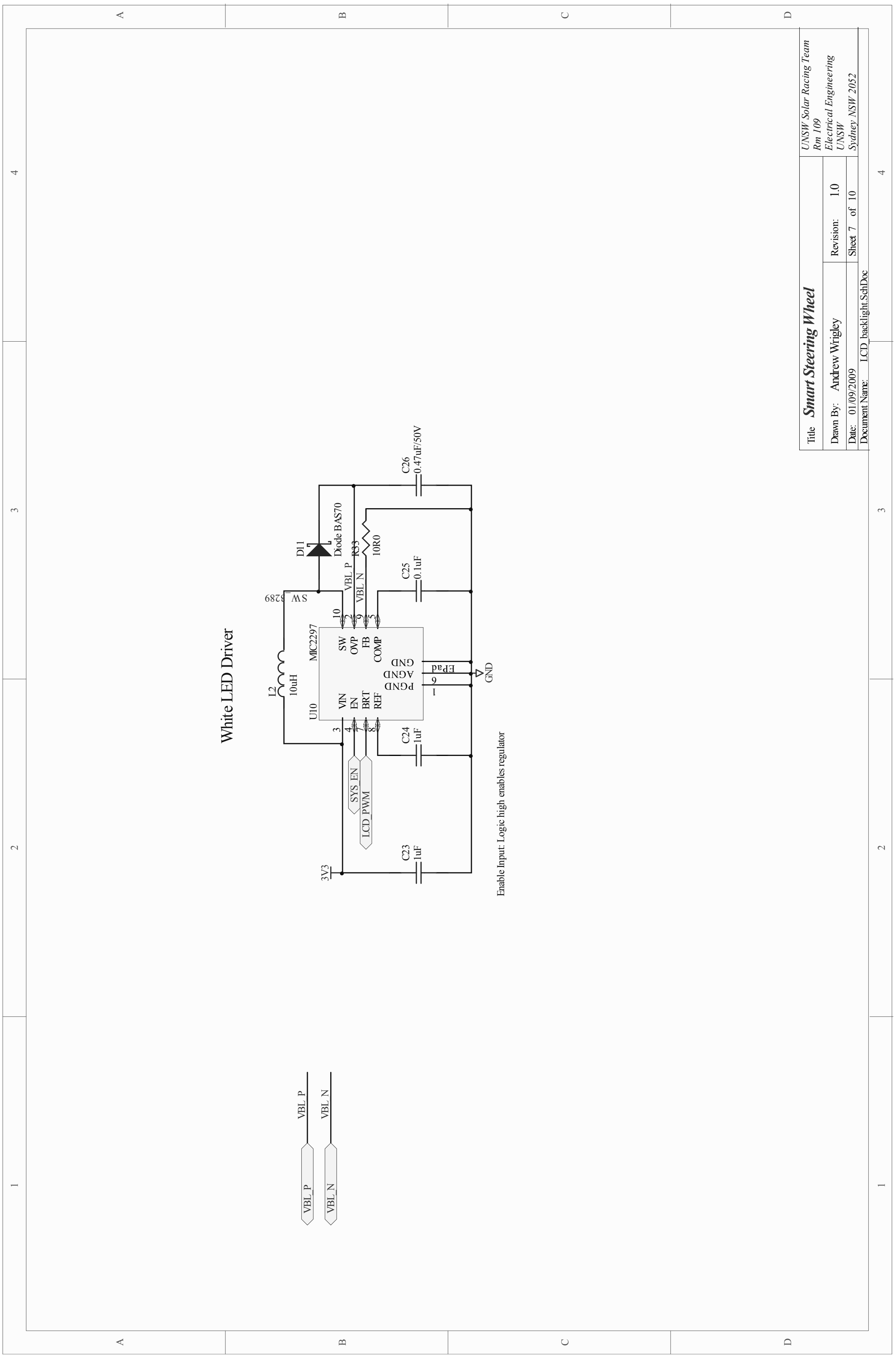

Figure A.7: Steering Wheel schematic: LCD backlight driver.

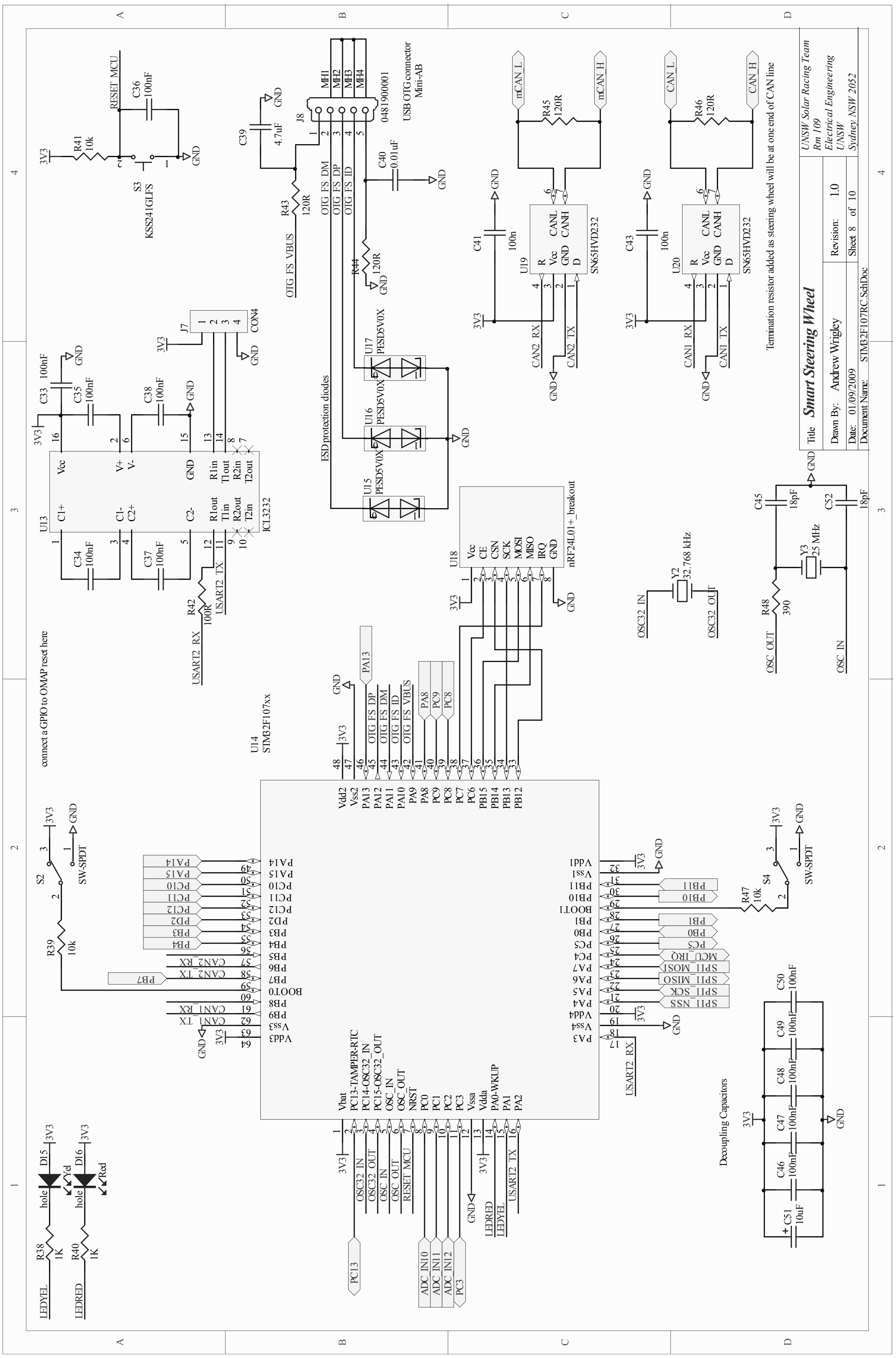

Figure A.8: Steering Wheel schematic: STM32F107 MCU.

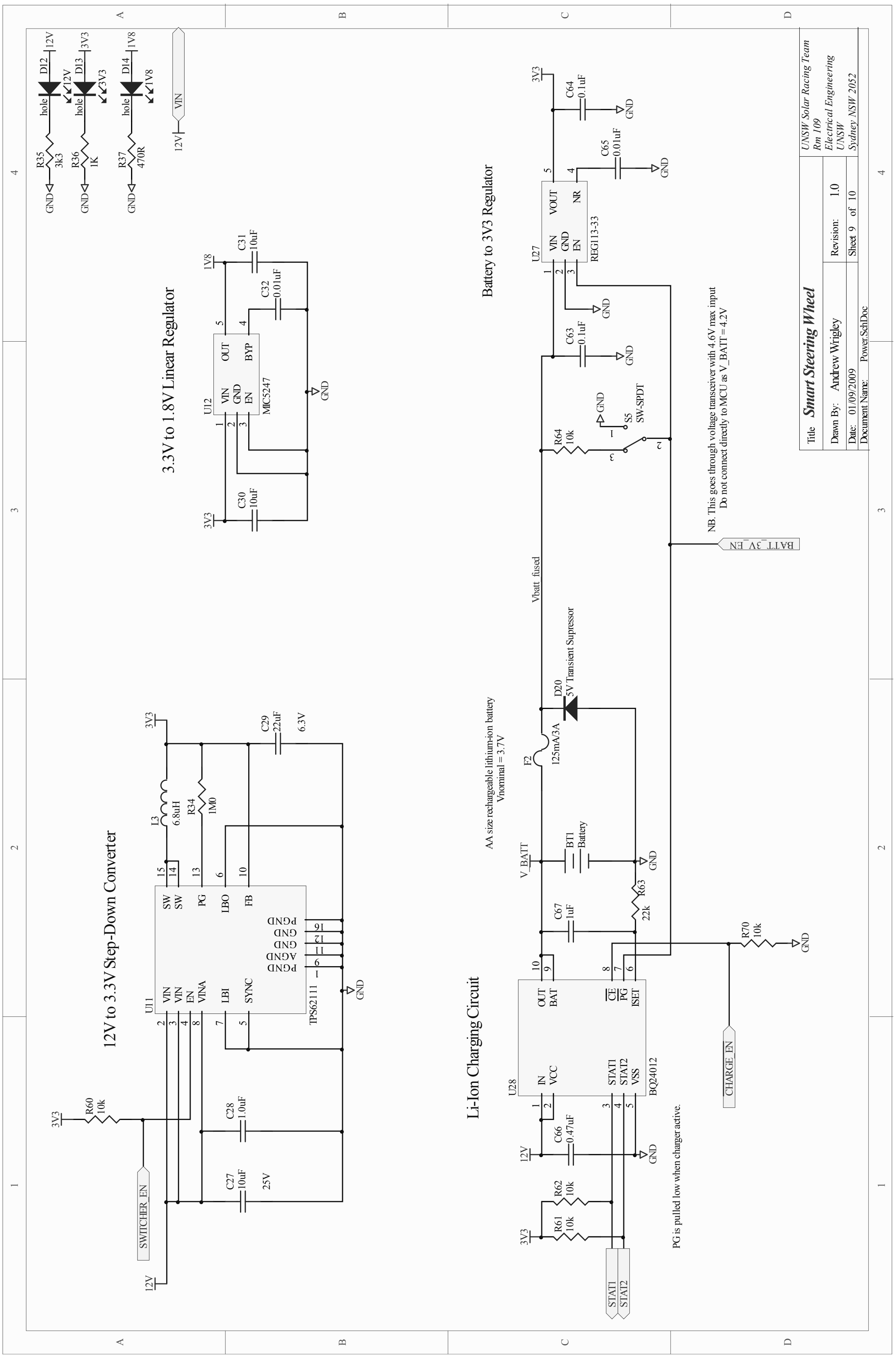

Figure A.9: Steering Wheel schematic: Power supply.

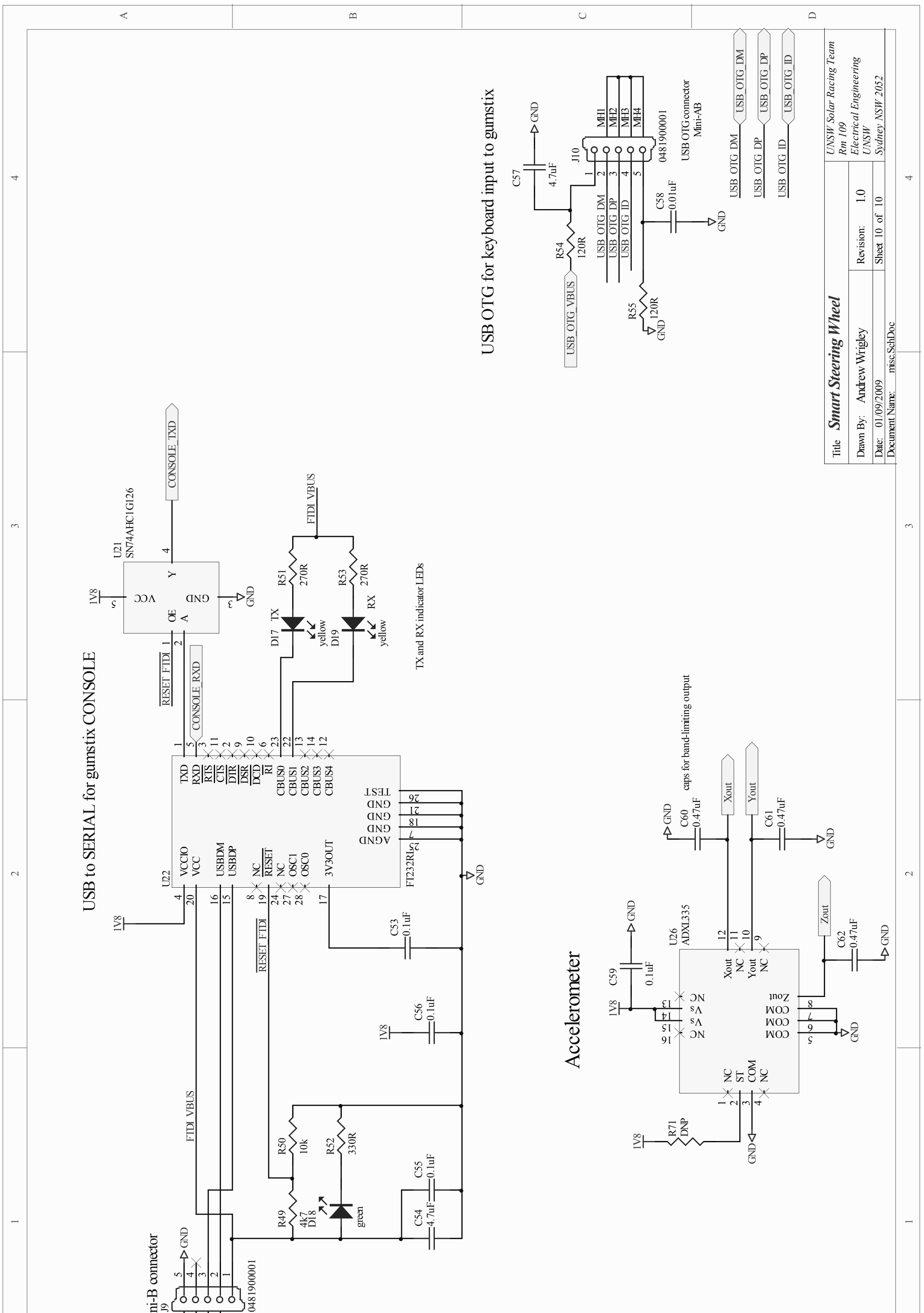

Figure A.10: Steering Wheel schematic: Miscellaneous devices.

Figure A.11: Steering Wheel PCB: Top layer with silkscreen.

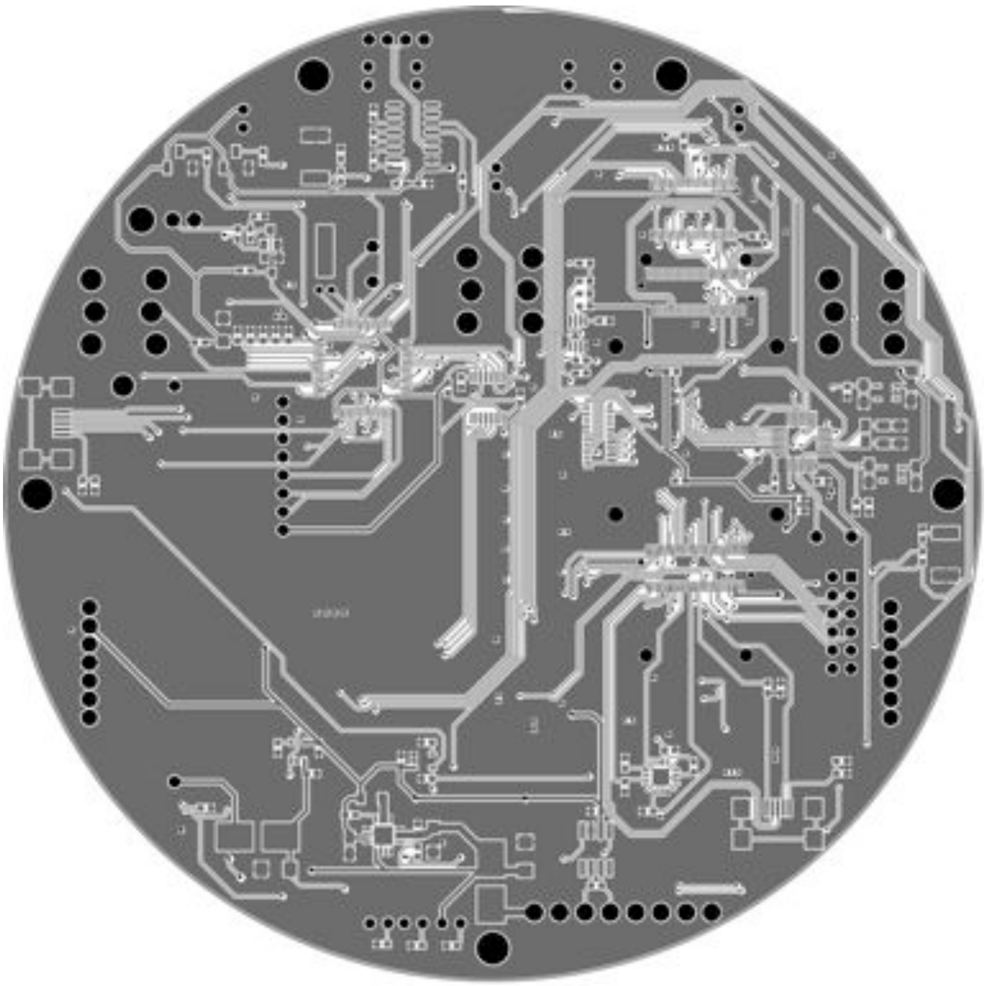

Figure A.12: Steering Wheel PCB: Bottom layer with silkscreen.

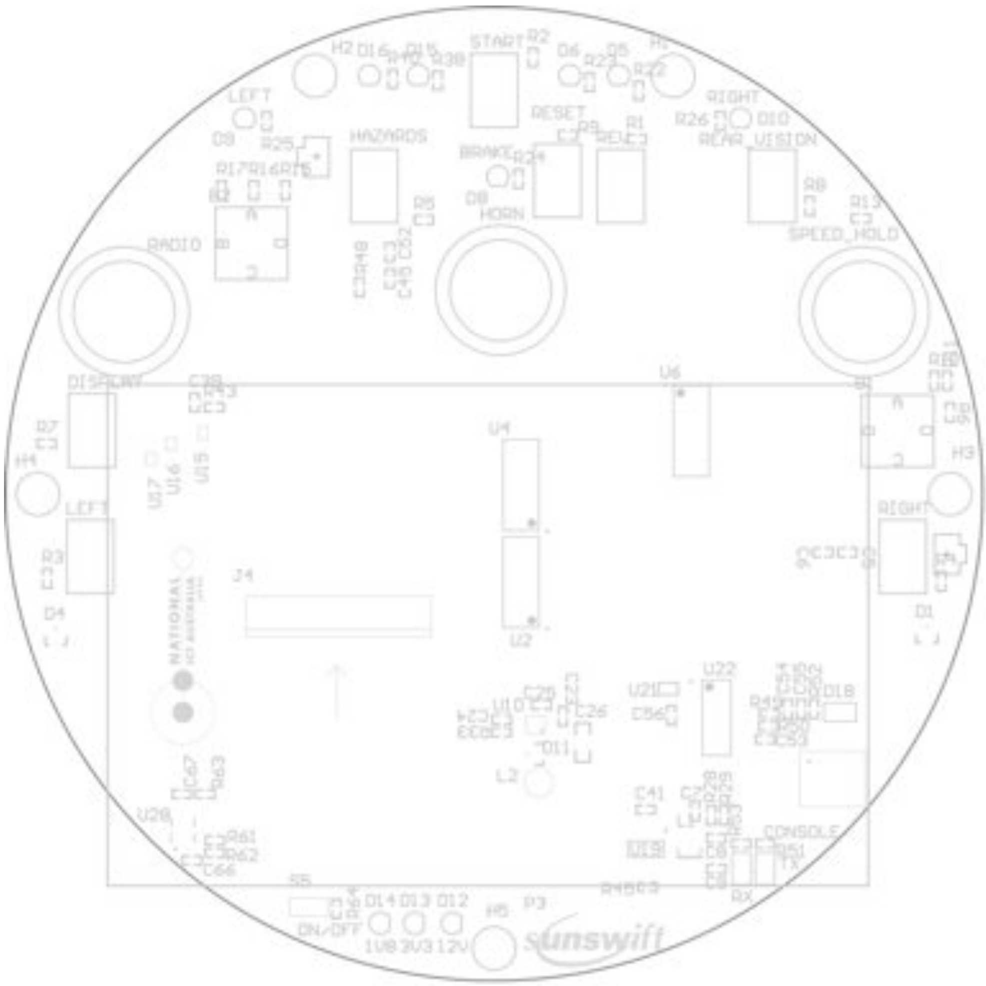

Figure A.13: Steering Wheel PCB: Top layer component layout.

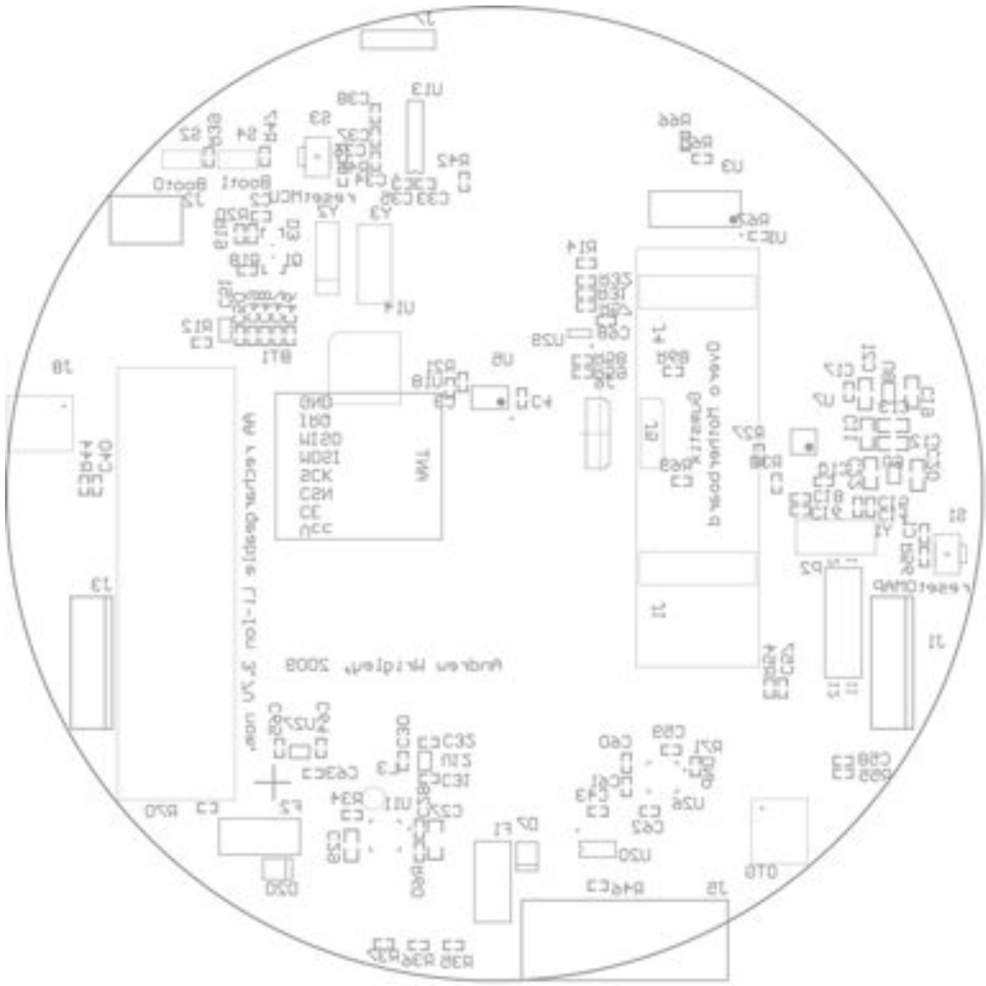

Figure A.14: Steering Wheel PCB bottom layer component layout.

Appendix B

Brake Sensor Schematics and PCB Artwork

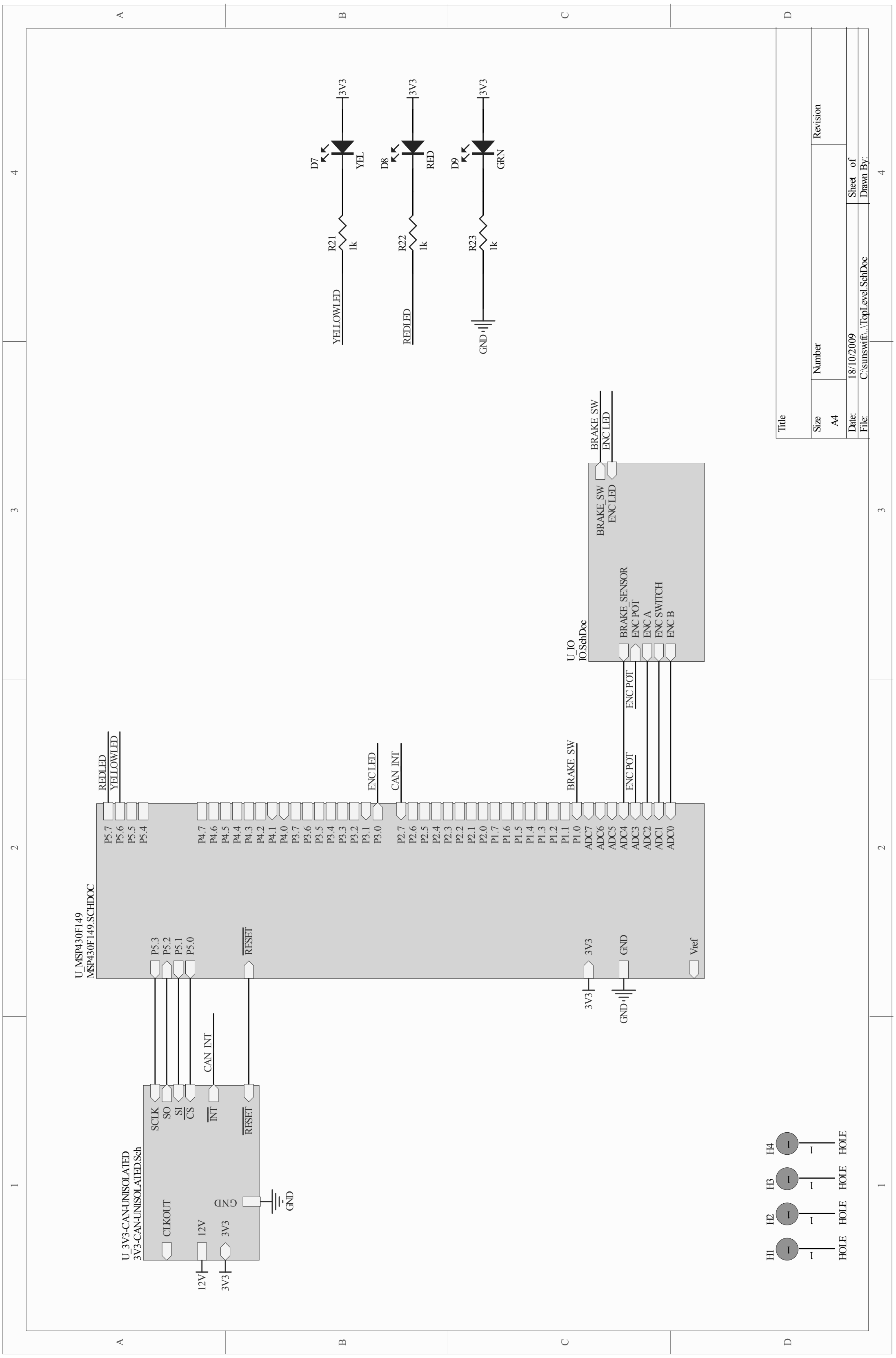

Figure B.1: Brake Sensor schematic: Top Level.

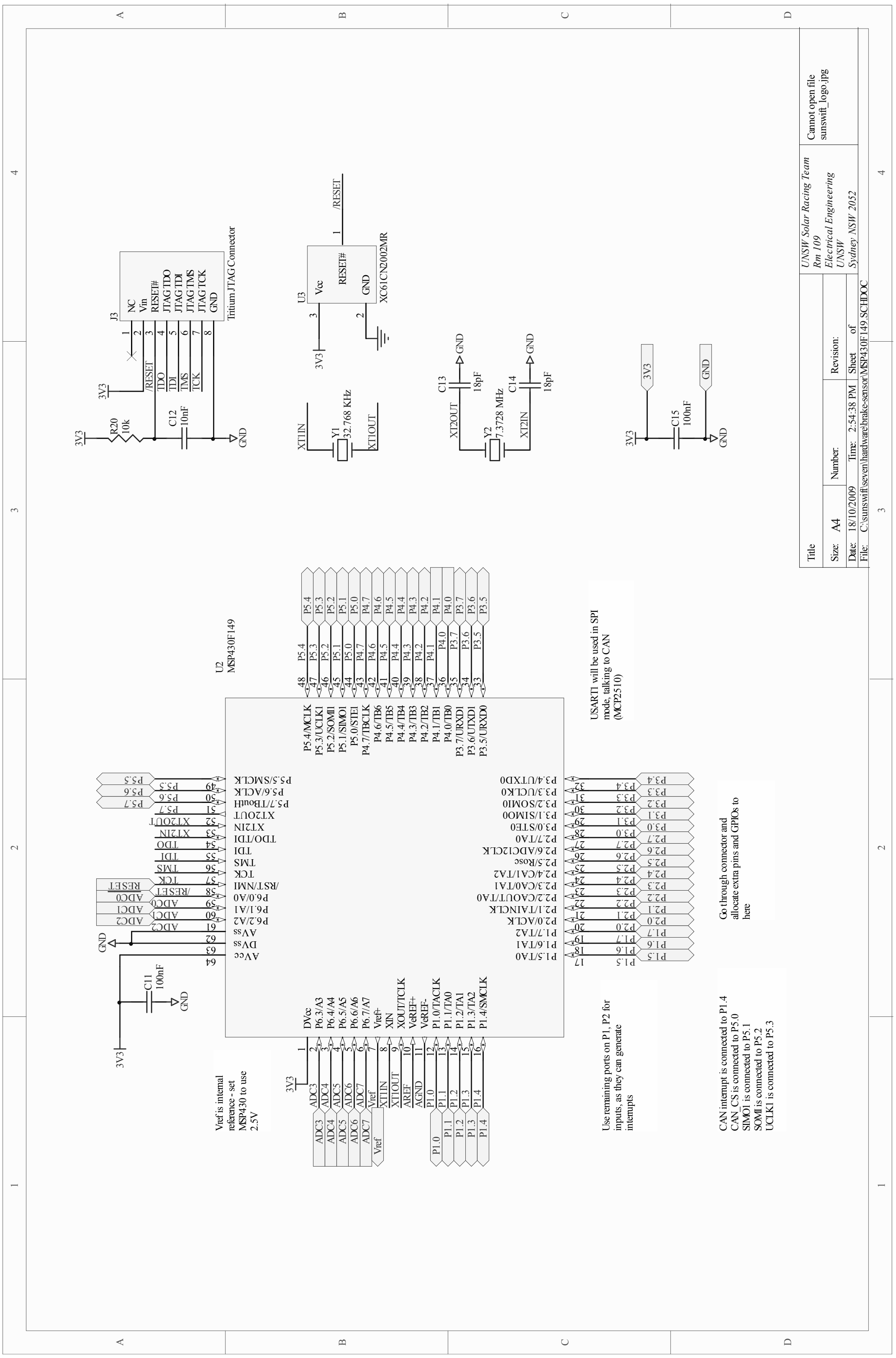

Figure B.2: Brake Sensor schematic: MSP430 MCU.

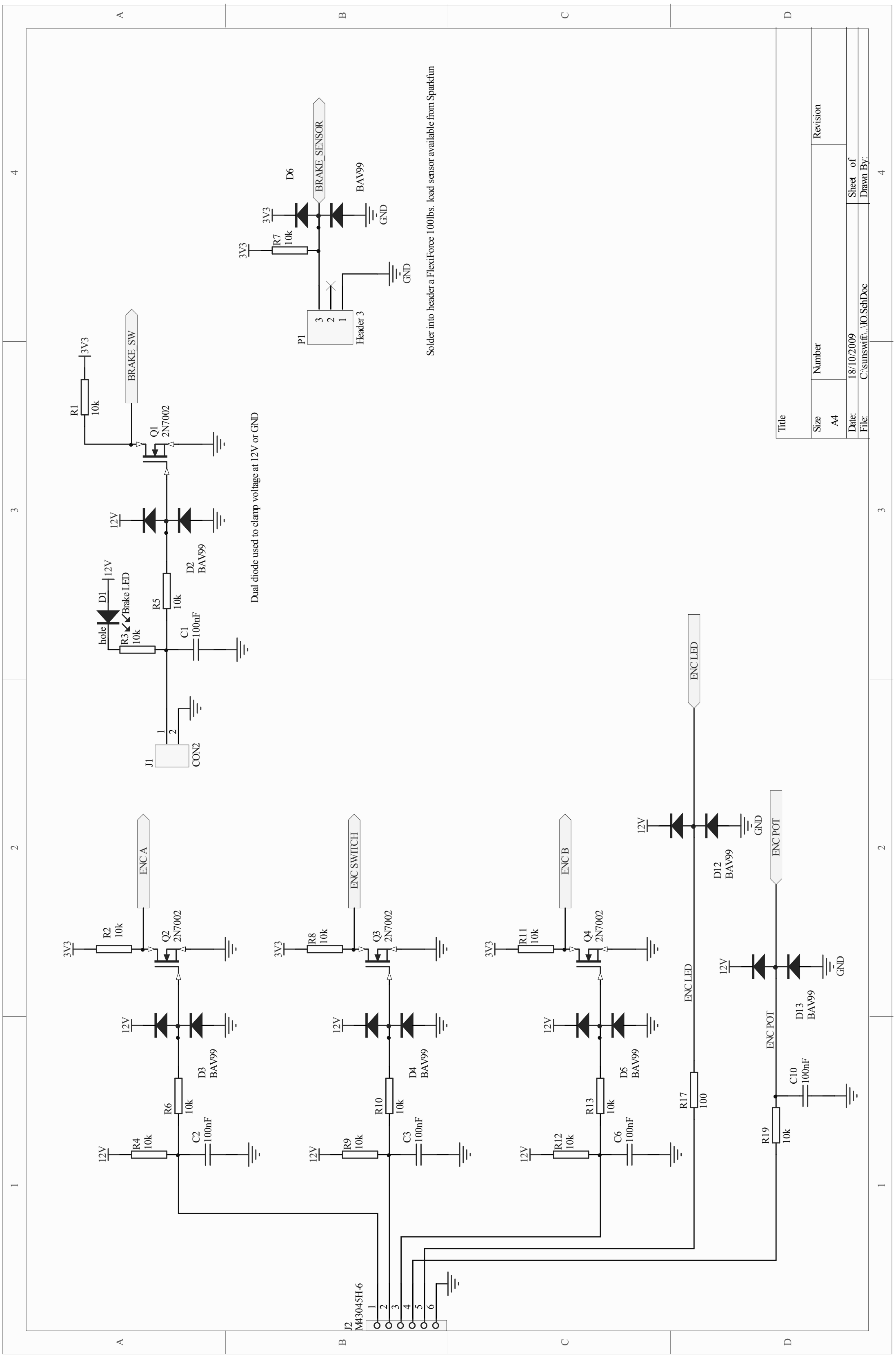

Figure B.3: Brake Sensor schematic: External I/O.

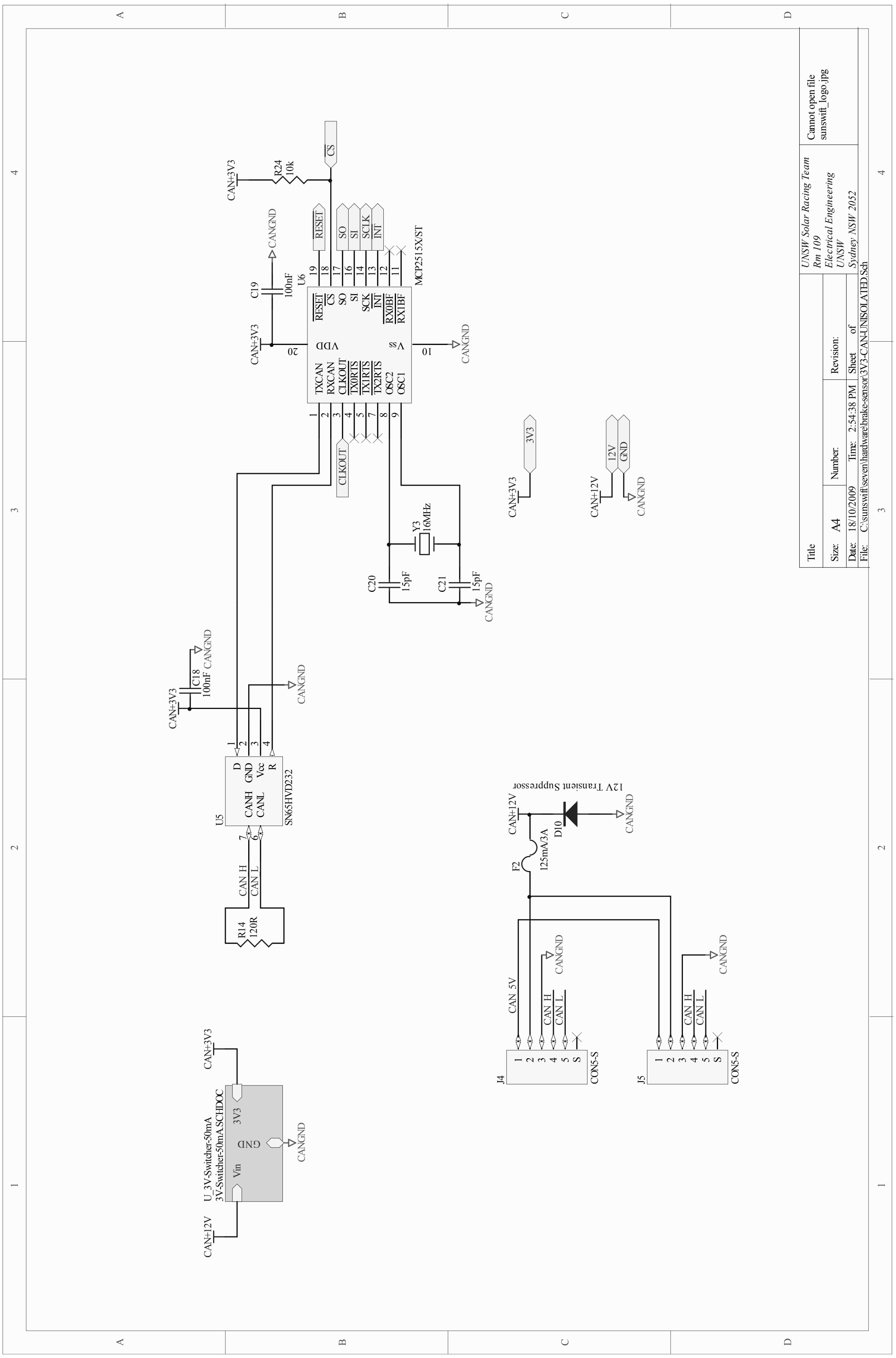

Figure B.4: Brake Sensor schematic: CAN.

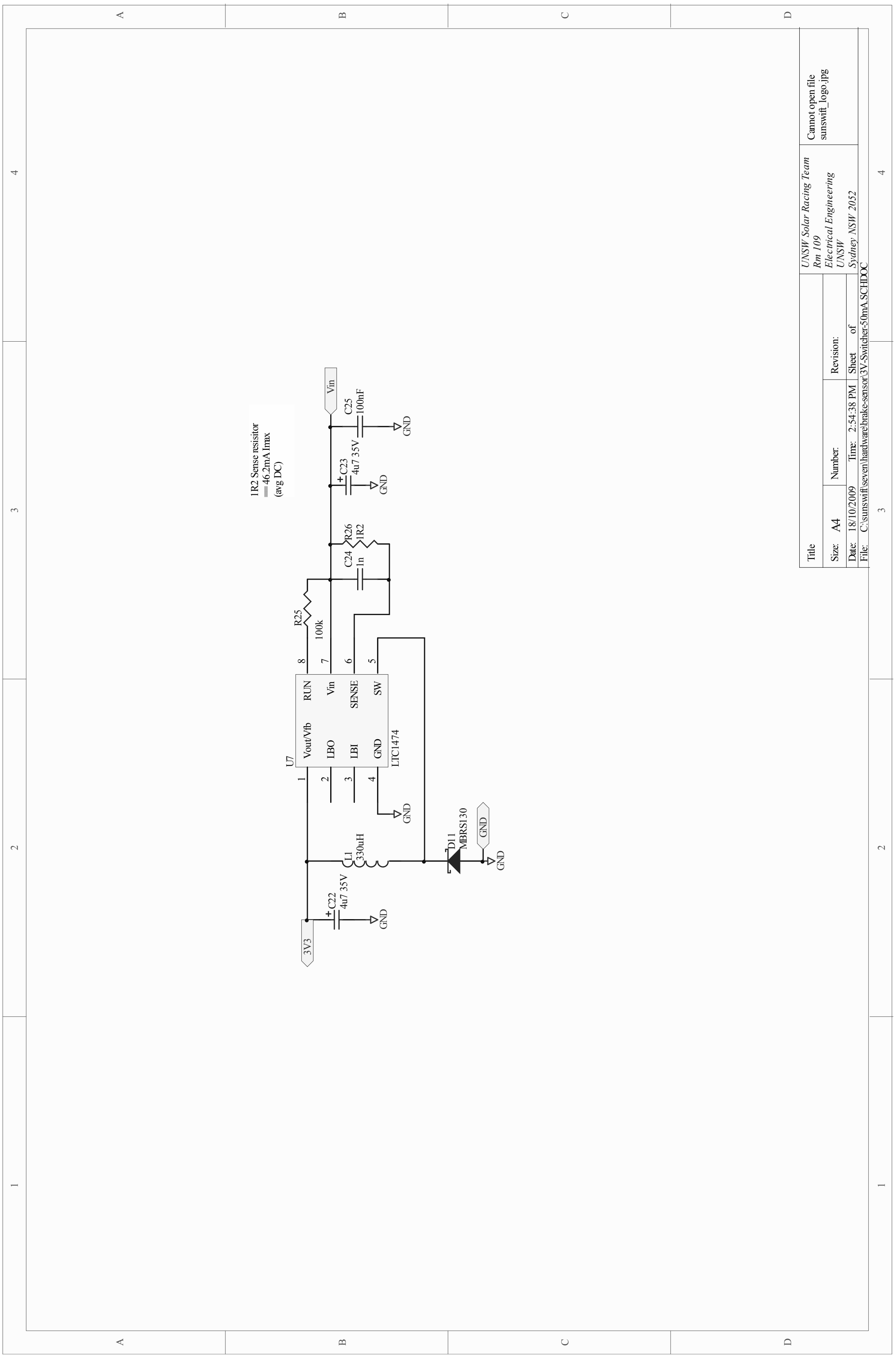

Figure B.5: Brake Sensor schematic: Power supply.

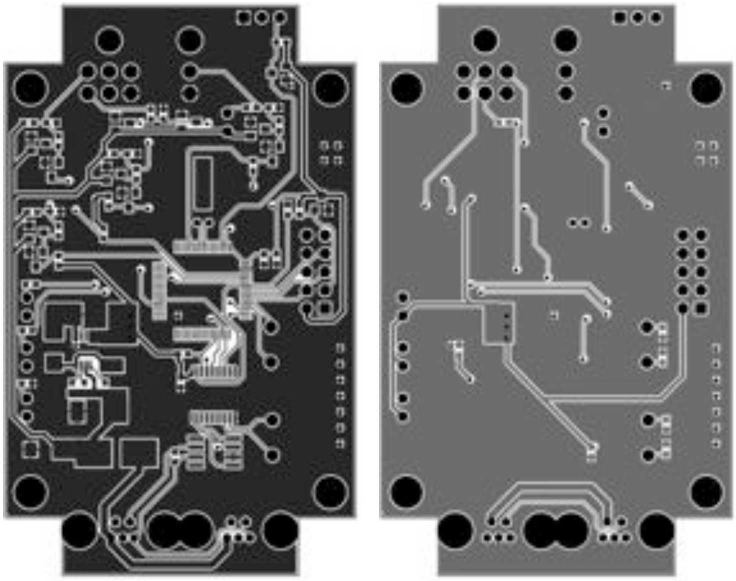

Figure B.6: Brake Sensor PCB: Top and bottom layer silkscreen.

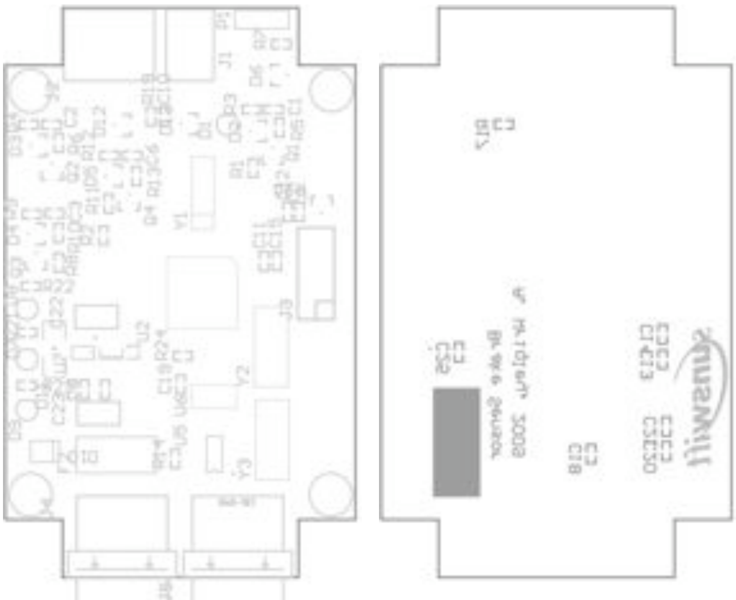

Figure B.7: Brake Sensor PCB: Top and bottom layer component placement.

Appendix C

Mechanical manufacturing drawing

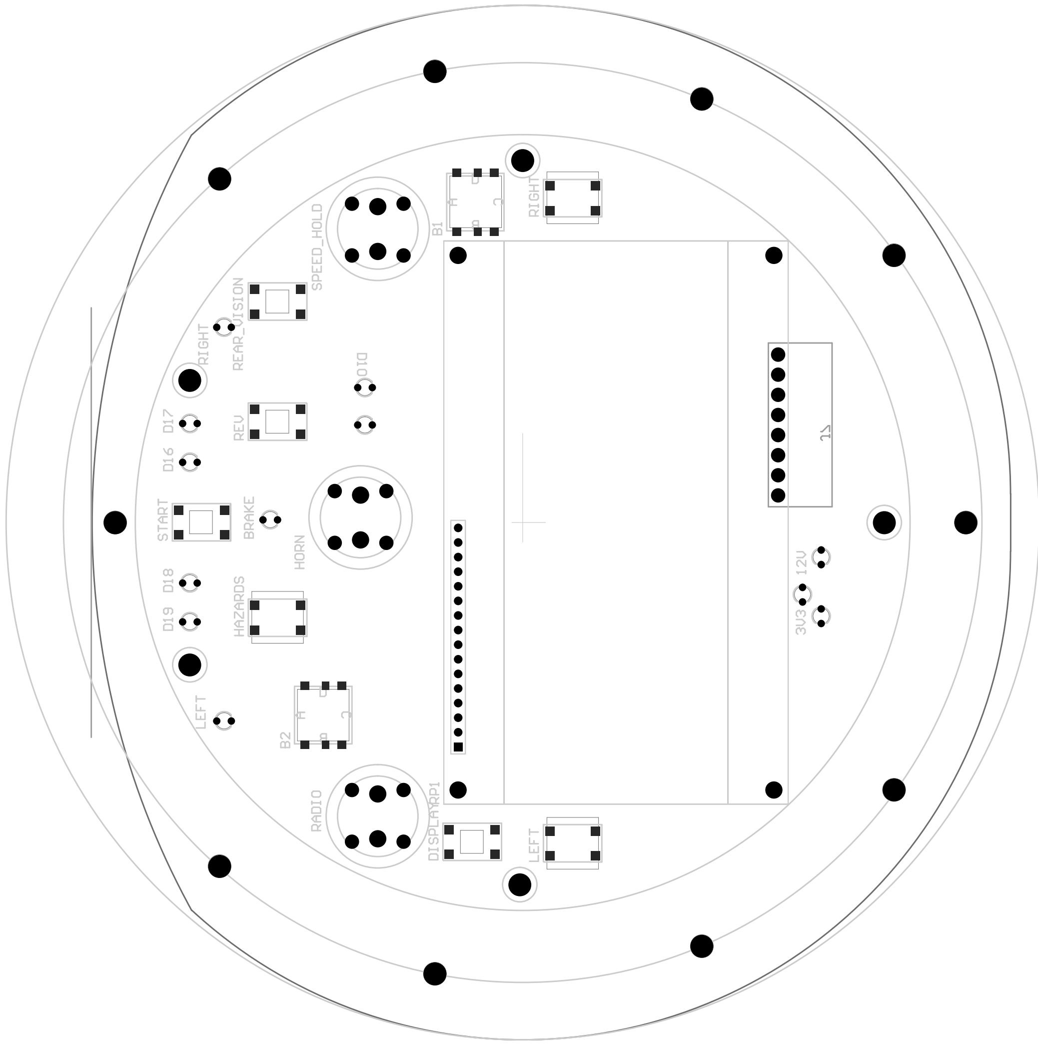

Figure C.1: Steering Wheel front cover manufacturing drawing.

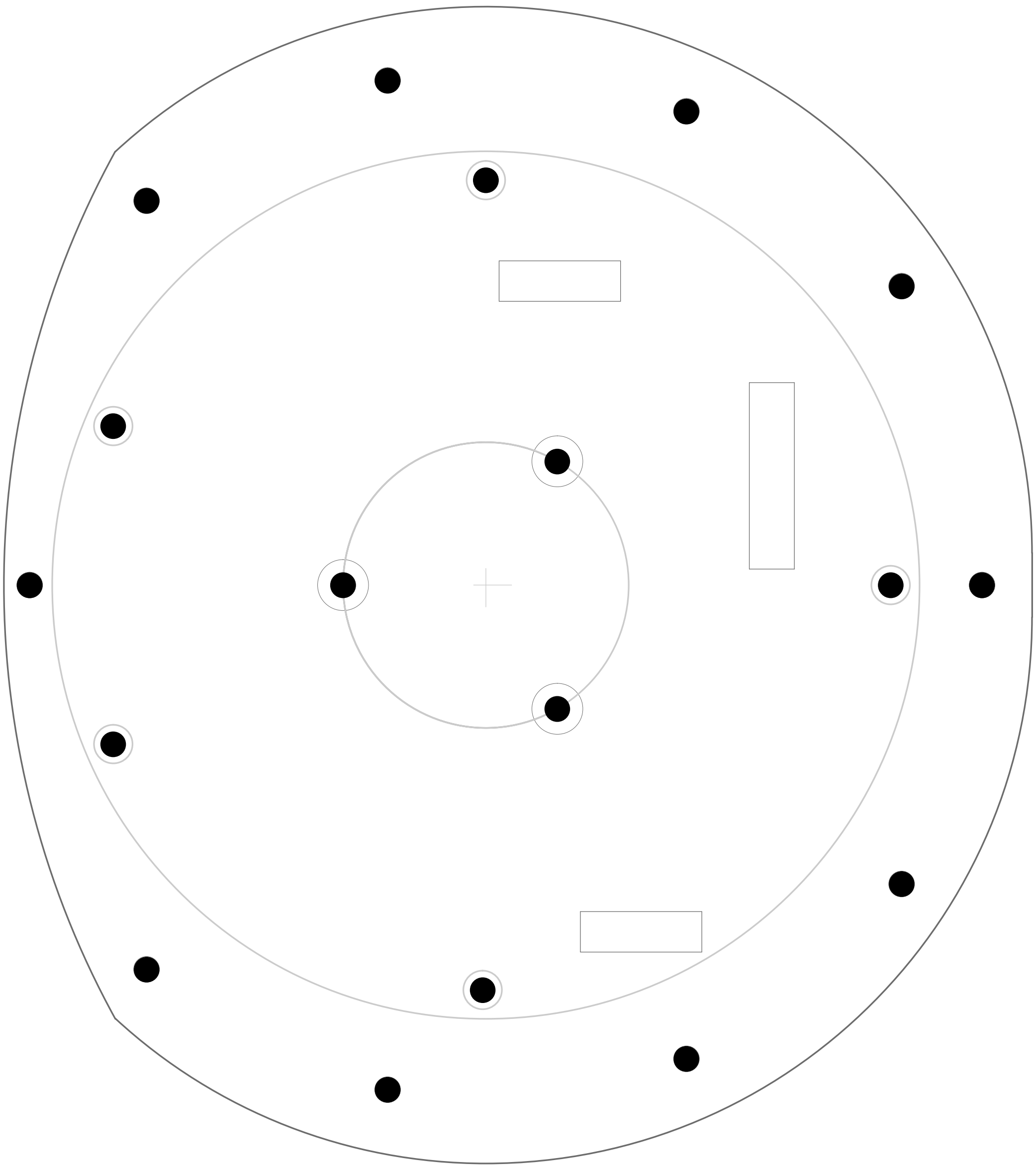

Figure C.2: Steering Wheel back cover manufacturing drawing.

Appendix D

Soldering method for no-lead packages

This method was developed and experimented with by the author as an alternative to using solder paste for no-lead package ICs. Solder paste requires a stencil, either for the whole PCB or for individual components, that ensures the solder paste is only on the PCB pads. The stencil is placed over the PCB and the solder paste is wiped over the top, then the stencil is removed. The problem with this method is that if the stencil is accidentally moved, solder paste becomes smeared across the PCB. If this occurs, the PCB must be cleaned and the process attempted again. If the solder paste is not cleaned it will cause shorting between pads when the PCB is heated on the reflow machine. Another problem with this method is that it requires the PCB be heated to at least $220^o C$, which can stress the PCB if sustained for too long or repeated too many times. The method detailed in this appendix avoids these two problems and was successfully used to solder two QFN package ICs.

D.1 PCB preparation

Ensure the PCB is clean and free of any old flux. Ideally, the reflow soldering described here should be completed early in the PCB population process. This is because it involves heating the entire board, and components already soldered on the underside of the board may fall off when the solder liquefies. The relevant pads on the PCB need to be lightly tinned with solder. Note that care must be taken to ensure only a very small amount of solder is present, and is equal between all the pads. It is particularly important that the ground pad does not

Figure D.1: Photograph of tinned MLF package IC.

have too much solder, otherwise it will raise the height of the chip too far away from the other pads. Remember there are no leads on these packages, so there is no room for error! Once the tinning is complete, a very small amount of fresh flux may be applied to the pads.

D.2 IC preparation

The preparation of the IC is similar to the PCB. This process will be made significantly easier if a stereo microscope or magnifying glass is available to inspect the pads. Observe good ESD protection practices, and remove the IC from its package and hold upside down with tweezers. Apply a very small amount of flux, and then tin the pads very lightly with solder. Again, it is important to have only a small amount of solder. Inspect under a microscope to ensure every pad is tinned and has the correct amount of solder. See Figure D.1 for a photograph of an appropriately tinned IC.

D.3 Reflow setup

The next step is to carefully place the IC in the required position on the PCB. Care should be taken to ensure the placement is as accurate as possible - again, a microscope will help. Note that if the IC moves slightly, the surface tension of the solder will re-align the IC to the correct position. Following this the PCB should be carefully placed on the reflow machine, as in Figure D.2. It is best to start with the machine at a low temperature, to ensure the PCB can be heated up as slowly as possible.

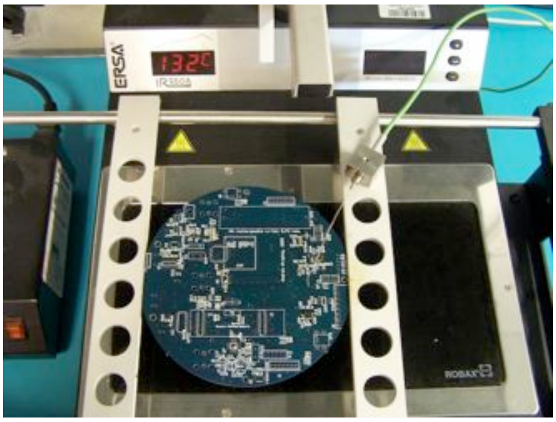

Figure D.2: Photograph of version 2 PCB on reflow machine.

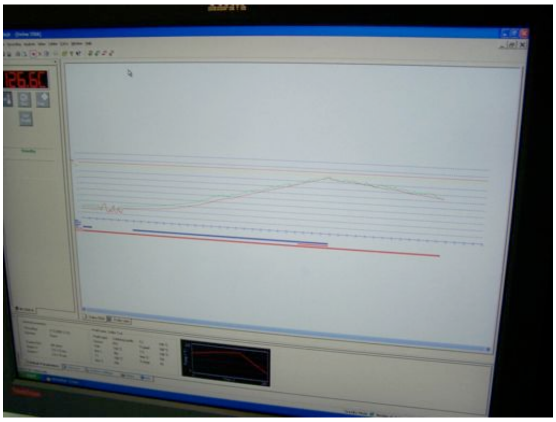

Figure D.3: Photograph of example heating profile for reflow machine.

The reflow machine, if possible, should be configured to slowly heat up to approximately 185^oC. The solder should liquefy at exactly 180^oC, but it is prudent to heat the PCB to just above this in case of temperature measurement calibration error. The PCB is heated slowly to ensure an even heat distribution, to minimise the delamination of the PCB. The heating profile used for the version 2 PCB in this thesis is shown in Figure D.3. Using this profile, the PCB takes approximately 10 minutes to reach temperature, and approximately 20 minutes to cool down again. It can be difficult to visually confirm the solder has melted under the IC, so it is useful to place a small amount of solder nearby on the PCB to verify the correct temperature has been reached.

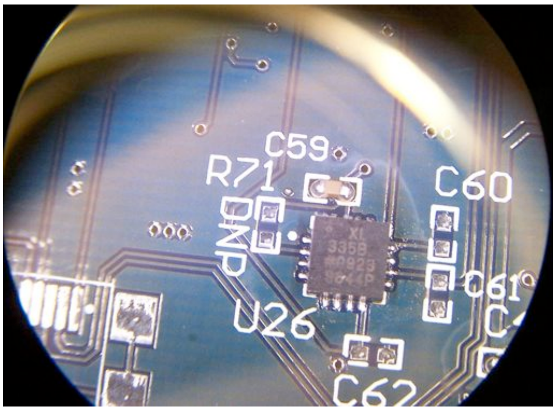

Figure D.4: Photograph of QFN package IC successfully soldered to PCB, viewed under a stereo microscope.

D.4 Inspection and testing

Once the PCB has been allowed to cool sufficiently, carefully remove it from the reflow machine and inspect under a microscope. Check, where possible, that all pads are connected with good quality joints. Figure D.4 shows a QFN package IC that has been successfully soldered onto the PCB using the method described in this appendix. Use a multimeter to test the resistance of each pad to ground and Vcc, as well as to every other pad on the IC. If this is satisfactory, the PCB can now be tested using a current limiting power supply. If any pads are found to be poorly connected, first try repairing them with a soldering iron. If this is not successful the process on the reflow machine may need to be repeated.