



Degree Project in Technology

Second cycle, 30 credits

# **PCB design and performance evaluation of miniaturized electronics**

A case study for the SOMIRO project

**ALBERT JANSSON**

# **PCB design and performance evaluation of miniaturized electronics**

**A case study for the SOMIRO project**

ALBERT JANSSON

Degree Programme in Electrical Engineering

Date: December 11, 2022

Supervisors: Matthias Becker, Gustaf Mårtensson

Examiner: Carl-Mikael Zetterling

School of Electrical Engineering and Computer Science

Host company: Mycronic AB

Swedish title: Konstruktion och utvärdering av miniaturiserad elektronik

Swedish subtitle: En fallstudie för SOMIRO-projektet



## **Abstract**

Electronics miniaturization is an ever-important subject in the industry of consumer electronics, where smaller, lighter and more powerful electronics is expected. This thesis investigates the miniaturization challenge in the EU-funded project SOMIRO, that aims to construct an energy autonomous swimming millirobot for remote sensing in agriculture. The current prototype Generation 1 (G1) prototype design is used as a base and a smaller version with additional features is constructed to evaluate possible performance differences. The Printed Circuit Board (PCB) that is produced is of a folding flex-rigid construction that sandwiches several layers of components to fit all components required. The performance of the new Generation 2 (G2) prototype is very similar to the existing G1 prototype in all electrical performance tests with the notable exception being the current draw for actuation of the swimming platform. The G2 prototype consumes significantly less current in this case, which is beneficial for the limited energy availability the millirobot will be operating in. There is still room for improving the PCB design with additional advanced PCB manufacturing techniques. Some of the external parts for the final version of the millirobot still needs to be finalized, for which this PCB may need additional changes, but this is not part of this thesis.

## **Keywords**

Electronics design, Flex-rigid, Energy autonomous, Swimming millirobot



## Sammanfattning

Miniatyrisering av elektronik är ett ständigt aktuellt problem i industrin för konsumentelektronik, där mindre, lättare och mer kraftfulla produkter förväntas. Detta mastersarbete undersöker miniatyriseringsutmaningen i EU-projektet SOMIRO som ska utveckla en energiautonom simmande millirobot för distribuerad mätning inom vattenbaserade jordbruk. Den nuvarande prototypen, Generation 1 (G1), lägger grunden till detta arbete som producerar en mindre version som dessutom innehåller fler funktioner. Den nya prototypen, Generation 2 (G2), utvärderas och jämförs med G1-versionen för att se om det är någon skillnad i elektrisk prestanda. Kretskortet som konstrueras är hopvikbart för att få plats med alla komponenter. Det nya kortet presterar mycket likt G1-versionen, förutom i testet för drivningen av aktuatorplattformen, där det nya kortet drog mindre ström. Det är en fördel då en mycket begränsad mängd energi kommer finnas tillgänglig i de tänkta miljöerna för milliroboten. Det finns fortfarande förbättringsmöjligheter då ytterligare avancerade konstruktionstekniker kan användas i design och tillverkning av kretskortet för att minska storleken ytterligare. Vissa förändringar kan också krävas för att kretskortet ska kunna monteras ihop med de externa delarna som ingår i den kompletta milliroboten, vilket dock inte är del av detta arbete.

## Nyckelord

Elektronikkonstruktion, Flex-rigid, Energiautonom, Simmande millirobot



## Acknowledgments

I want to thank Mycronic for providing the facilities and tools used in designing and assembling the prototypes for this thesis.

I want to thank EPFL for providing the prototype design and IMDEA for providing their experimental code and communication system design, as well as all other participants in the SOMIRO project that have helped me throughout the design process.

I want to thank my supervisor at Mycronic, Gustaf Mårtensson for helping me with any administrative issues regarding the project, Fredrik Lindén and Joacim Lundberg for the electronics design review and discussions, as well as the Jet Technology group for the great support during the project.

I also want to thank my academic supervisor Matthias Becker and my examiner Carl-Mikael Zetterling at KTH for their feedback throughout the project.

The SOMIRO project is funded by the European Union and as this thesis is part of the SOMIRO project, the following text should be included:

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101016411*

Stockholm, December 2022  
Albert Jansson



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Background . . . . .	1
1.2	Research Question . . . . .	2
1.3	Goals . . . . .	2
1.4	Delimitations . . . . .	2
1.5	Structure of the thesis . . . . .	3
<b>2</b>	<b>Background</b>	<b>5</b>
2.1	Common PCB types in industry . . . . .	5
2.1.1	FR4 PCBs . . . . .	5
2.1.2	Flexible PCBs . . . . .	6
2.1.3	Flex-rigid PCBs . . . . .	7
2.1.4	Ceramic substrate PCBs . . . . .	7
2.2	System integration levels . . . . .	7
2.2.1	SMD . . . . .	8
2.2.2	COB . . . . .	8
2.2.3	Hybrid . . . . .	8
2.2.4	ASIC . . . . .	8
2.3	The SOMIRO soft millirobot . . . . .	9
2.3.1	Background . . . . .	9
2.3.2	SOMIRO Generation 1 . . . . .	9
2.3.3	SOMIRO Generation 2 . . . . .	10
2.4	Related work . . . . .	11
2.4.1	RoACH . . . . .	11
2.4.2	mROBerTO . . . . .	11
2.4.3	SINTEC . . . . .	11
2.5	Summary . . . . .	12
<b>3</b>	<b>PCB Design</b>	<b>13</b>
3.1	Schematics . . . . .	14
3.1.1	High voltage actuation . . . . .	14
3.1.2	Power management . . . . .	15
3.1.3	Sensing . . . . .	15

3.1.4	Computation . . . . .	15
3.1.5	Communication . . . . .	16
3.2	Layout . . . . .	17
3.3	Assembly . . . . .	18
3.4	System documentation . . . . .	20
<b>4</b>	<b>Experimental Evaluation</b>	<b>21</b>
4.1	Power consumption measurements . . . . .	21
4.1.1	Idle power consumption . . . . .	22
4.1.2	Active power consumption . . . . .	22
4.1.3	Sensing power consumption . . . . .	22
4.1.4	Actuating power consumption . . . . .	22
4.1.5	Communication power consumption . . . . .	23
4.1.6	Summary of results . . . . .	23
4.2	Efficiency measurements . . . . .	24
4.2.1	Charging efficiency . . . . .	24
4.2.2	Discharging efficiency . . . . .	24
4.3	Physical parameters . . . . .	26
4.4	Reliability and validity assessment . . . . .	26
<b>5</b>	<b>Discussion</b>	<b>27</b>
5.1	Comments on results . . . . .	27
5.2	System design . . . . .	28
5.3	Component selection . . . . .	29
5.4	PCB layout considerations . . . . .	29
5.5	Software considerations . . . . .	30
<b>6</b>	<b>Conclusions and Future work</b>	<b>33</b>
6.1	Conclusions . . . . .	33
6.2	Limitations . . . . .	34
6.3	Future work . . . . .	34
	<b>References</b>	<b>35</b>
	<b>Appendix A Schematics</b>	<b>39</b>
	<b>Appendix B Layout</b>	<b>47</b>
	<b>Appendix C Software</b>	<b>52</b>
C.1	Main file . . . . .	52
C.2	Tasks . . . . .	57
C.3	G1 header file . . . . .	65
C.4	G2 header file . . . . .	68

# List of Figures

2.1	Multi-layer FR4 PCBs. . . . .	6
2.2	Flexible PCB used to connect a few sensors in a modern smartphone. . . . .	6
2.3	A block diagram of the complete millirobot. . . . .	10
3.1	Top side of the G2 prototype PCB. . . . .	18
3.2	Bottom side of the G2 prototype PCB. . . . .	19
3.3	The G2 prototype PCB folded into the intended shape, a USB connector for reference. . . . .	19
3.4	Top side of the G1 prototype PCB. . . . .	20
3.5	Bottom side of the G1 prototype PCB. . . . .	20
4.1	Charging efficiency for a set of representative storage element voltages and solar input voltages. . . . .	25
4.2	Discharge efficiency for a representative set of storage element voltages and load resistances. . . . .	25



# List of Tables

3.1	Performance characteristics of down-link front-end solutions .	17
4.1	Power measurement results . . . . .	24



## List of acronyms and abbreviations

ADC	Analog to Digital Converter
ASIC	Application Specific Integrated Circuit
BGA	Ball Grid Array
BOM	Bill of Materials
CAD	Computer Aided Design
CMOS	Complementary Metal Oxide Semiconductor
COB	Chip on Board
DAC	Digital to Analog Converter
EPFL	Ecole Polytechnique Federale de Lausanne
FRAM	Ferroelectric Random-Access Memory
G1	Generation 1
G2	Generation 2
GPIO	General-Purpose Input/Output
HDI	High-Density Interconnection
HTCC	High Temperature Co-fired Ceramic
IC	Integrated Circuit
IMDEA	Fundacion IMDEA Networks
JKU	Johannes Kepler Universität Linz
LDO	Low Dropout Regulator
LTCC	Low Temperature Co-fired Ceramic
MC	Mycronic AB
MOSFET	Metal Oxide Field Effect Transistor
MPG	Max-Planck-Gesellschaft zur Förderung der Wissenschaften e.V.
MPP	Maximum Power Point
PCB	Printed Circuit Board
PWM	Pulse Width Modulation

RAM	Random-Access Memory
RP	Battioli Paola Società Agricola S.S.
SAR	Successive Approximation Register
SMD	Surface Mount Device
SMT	Surface Mount Technology
SNR	Signal-to-Noise Ratio
TC	The Circle Società Agricola a Responsabilità Limitata
TPM1	Timer/PWM Module 1
UU	Uppsala Universitet
VLC	Visible Light Communication
VLLS	Very Low Leakage Stop
VLPR	Very Low Power Run
VLPW	Very Low Power Wait
WH	Warrant Hub S.p.A.

# Chapter 1

## Introduction

This thesis investigates the challenges of electronics miniaturization, specifically focusing on Printed Circuit Board (PCB) layout, using only commercially available components.

This chapter describes the background, research question and the goals set out for this thesis, together with some delimitations and a description of the overall structure of the report.

### 1.1 Background

Electronic systems have followed a trend of size reduction, density increase and higher performance levels for several decades. Higher integration levels in custom-made Integrated Circuits (ICs) are more and more common, allowing more functionality and higher performance systems to fit smaller and smaller footprints.

With this, power consumption and heat generation must be considered in order to produce an electronic system that still is reliable and fit for purpose in battery operated scenarios that are becoming ever more popular.

Reducing the footprint of an electronic system is not always a straightforward process and many trade-offs must be considered. This thesis aims to reduce the size of an electronic system without producing any custom ICs. This may or may not be viable, depending on the electronic system at hand and the sizing requirements.

This approach is applied to the SOMIRO project, which aims to construct a swimming millirobot for remote sensing applications, utilizing energy harvesting of ambient light.

## 1.2 Research Question

This thesis should reduce the size of the SOMIRO swimming millirobot, integrate a complete communications system and sensing system in addition to the existing systems present in the current prototype. The miniaturized electronics should consume the same amount or less energy when compared to the existing prototype. With this, the following research question should be answered:

*Is it possible to reduce the footprint of the SOMIRO swimming millirobot to 100 mm<sup>2</sup>, the volume to 1 cm<sup>3</sup> and the weight to 1 g while consuming the same amount of energy and integrating additional features?*

## 1.3 Goals

The goal of this project is to reduce the size of the SOMIRO swimming millirobot electronics while including all features intended for the final version of the millirobot. This is divided into the following sub-goals:

- Produce a PCB design that incorporates all functionality intended for the final version of the millirobot.
- Manufacture and assemble the PCB.
- Verify functionality and performance of the assembled PCB

The deliverables from the thesis project are the following:

- An updated set of schematic documents that include all functionality of the millirobot.
- Layout of the miniaturized PCB containing all components specified in the schematics.
- Documentation of the design decisions made for the PCB.
- Test results and performance metrics for the main functionality of the assembled PCB.

## 1.4 Delimitations

This thesis project focuses on the electronics hardware of the SOMIRO millirobot and hence the software aspects or any other parts of the millirobot are not covered. The exception is software used for testing the hardware functionality.

Since the hardware of SOMIRO G1 is available as a reference, no major circuit design work should be part of this thesis project.

The mechanical assembly of the millirobot is also not part of this thesis project, although the constraints of packaging the electronics must be considered.

This project may not produce a design that meets all requirements for the final product and hence it should not be expected that this thesis project produces a design that will be directly implemented in the final version of the SOMIRO swimming millirobot. The design produced should help the SOMIRO team in the design of the final product by demonstrating techniques and methods for reducing the size of the electronics package.

## 1.5 Structure of the thesis

The thesis is organized as follows:

- Chapter 2 presents some relevant background on common PCB types used in industry and common integration levels for electronic components. Then, the background on the SOMIRO project is presented, together with some related work.
- Chapter 3 describes the circuit design, components used and the PCB layout.
- Chapter 4 presents the tests performed on the PCB produced and compares the performance of this to the existing prototype PCB.
- Chapter 5 contains the discussion, where the design and other aspects of the project are lifted.
- Chapter 6 contains the conclusions based on the results of the evaluation and the discussion.



# Chapter 2

## Background

This chapter describes some of the available options for miniaturized electronics design, manufacturing and materials commonly used. The SOMIRO project is presented as well as some related work.

### 2.1 Common PCB types in industry

There are several commercially available PCB types that can be used in electronics manufacturing. The most relevant types are presented in this section.

#### 2.1.1 FR4 PCBs

The most common type of PCB is the FR4 PCB that consists of epoxy resin and glass fiber. Several layers of the base material can be laminated with copper layers in between to construct complex boards with a large number of interconnects. [1] describes this and some variations of this type of PCB. Because of its ubiquity, the cost is relatively low and the material properties are adequate for most applications. Figure 2.1 shows some example PCBs of this type.

The main problem with this type of PCB is the fact that the material is rigid throughout the board. This makes it more difficult to increase the volumetric density of a system without using multiple boards and some way of interconnecting these boards. In space constrained systems, there may not be enough room for the connectors required to join the different boards.

This type of PCB is however well suited for relatively low-cost High-Density Interconnection (HDI) structures, such as microvias and buried vias as described in [2]. The rigid nature allows for these structures to remain reliable in a large range of operating conditions.

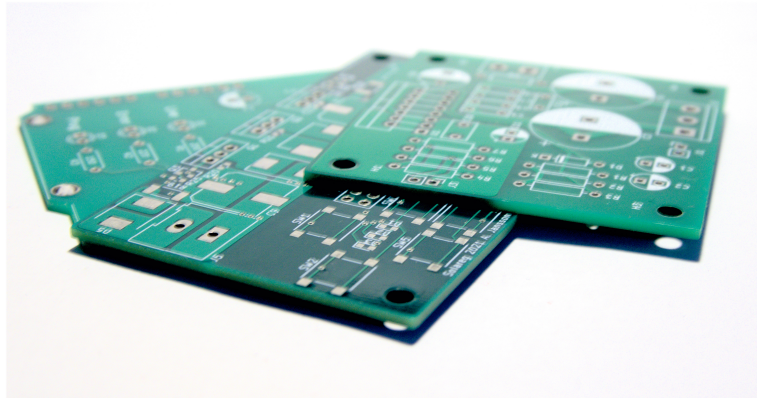


Figure 2.1: Multi-layer FR4 PCBs.



Figure 2.2: Flexible PCB used to connect a few sensors in a modern smartphone.

### 2.1.2 Flexible PCBs

An alternative to FR4 are the flexible types of substrates commonly used in places where space is limited, such as hand-held consumer devices. Figure 2.2 shows examples of this type of flexible PCB. [1] also describes this type of PCB briefly. PCBs using flexible substrates are more limited in terms of the number of conductive layers, but in return they can be significantly thinner than FR4 PCBs.

A folded construction could be realized using a flexible PCB, where parts of the board are folded over each other to reduce the footprint of the system. This makes it possible to include a larger number of components at the expense of some vertical height while not using right-angle or stacked boards for extra component area.

An important consideration is the allowable bending radius of the flexible

laminate. A more complex flexible PCB is thicker with more conductive layers and requires a larger bending radius than thinner variants with fewer layers. The trade-off between layer count and final installed geometry must be carefully considered.

### 2.1.3 Flex-rigid PCBs

In some cases, both the properties of rigid FR4 and the flexible board types are combined in a single piece. This gives the ability of high layer count rigid sections and minimal space interconnects between rigid sections on the flexible sections of the board.

This type of board can be used for complex and compact designs that require more than one or two layers for signal routing between high density ICs and still require the interconnect capabilities of the flexible substrate. A rigid section could have several additional layers that are not present on the flexible sections, where the thickness of the flexible section makes it possible to bend in a smaller radius due to the low layer count.

[3] and [4] describe the flex-rigid technology and its applications.

### 2.1.4 Ceramic substrate PCBs

Higher integration levels can be achieved by using a ceramic substrate PCB. Commonly, High Temperature Co-fired Ceramic (HTCC) or Low Temperature Co-fired Ceramic (LTCC) substrates are used where high frequency loss needs to be minimized [5]. The layer count of ceramic boards can be significantly higher than for FR4 boards. This makes ceramic boards suited for high density systems with large numbers of interconnects.

Another advantage is the ability to integrate passive components in the substrate to a larger degree than other materials, saving footprint area for the design. Higher grade conductors are generally used, meaning thinner traces and the possibility of further increasing the interconnect density.

The previously mentioned HDI structures are also available, making this type of PCB the obvious choice for certain applications.

## 2.2 System integration levels

There are a few integration levels commonly used in industry for constructing parts or components in electronic systems. This section presents the basis for the most relevant techniques that are considered in this thesis project.

All these techniques are described in [1].

### **2.2.1 SMD**

The most common integration level for electronic systems is Surface Mount Device (SMD), also known as Surface Mount Technology (SMT). All integrated circuits and passive components are packaged for soldering to a PCB that provides mechanical and electrical connections. Components can be mounted to both sides of the PCB and tracks can be routed on one of several layers depending on the PCB stack-up.

This is the cheapest option, given that most components are available as SMD components and can be used with standard processes such as pick-and-place, reflow etc.

### **2.2.2 COB**

A variation of SMD technology is Chip on Board (COB) where integrated circuits are mounted directly to the PCB as a bare silicon die. This eliminates one intermediary packaging step that takes up additional board space, but the silicon die must be electrically connected using bond wires if it is not a Ball Grid Array (BGA) style package at this level. Then, the mechanical strength lost by eliminating the intermediary package may need to be substituted by some other encapsulant, for example epoxy.

For large volume production, this can be cheaper than using SMD ICs, since one packaging step is replaced with a more cost-effective solution.

### **2.2.3 Hybrid**

Increasing the integration level, hybrid PCB designs can incorporate some of the components into the PCB substrate and to a larger degree use bare silicon dies without additional packaging. This saves additional space and the hybrid module encapsulates all the integrated circuits as a new custom package.

This requires additional design effort, but can be beneficial in cases where a module of higher complexity can be integrated in an otherwise lower complexity system, reducing the overall cost of the complete system.

Both FR4 and ceramic substrates may be used for hybrid modules, depending on the requirements for the application.

### **2.2.4 ASIC**

The highest level of integration is achieved in an Application Specific Integrated Circuit (ASIC), where the features of several discrete integrated circuit packages are integrated on the same piece of silicon. This gives the highest level of interconnect density and performance as well as reducing the space taken up by packaging.

This also results in the highest investments in design and manufacturing, but the return may be of critical importance in certain applications.

## 2.3 The SOMIRO soft millirobot

A short background on the SOMIRO project is presented in Section 2.3.1. Sections 2.3.2 and 2.3.3 describe the existing prototype and the new prototype that this thesis focuses on respectively.

### 2.3.1 Background

The SOMIRO project is meant to develop, build and demonstrate an energy-autonomous swimming millirobot for distributed remote sensing applications. These millirobots are a more flexible and dynamic monitoring system that can more easily cover a larger area compared to existing stationary systems. The aim is to reduce the environmental impact of agriculture by employing these millirobots as a more accurate monitoring system of chemical compounds in the water of rice farms, aquaponics and hydroponics [6]. Powering the robot is a major challenge, since both the storage and uptake of energy is limited. Months long, untethered, autonomous operation in greenhouse environments is the ambition for the SOMIRO project.

Several universities, institutions and companies in Europe are involved in the SOMIRO project, namely: Uppsala Universitet (UU), Ecole Polytechnique Federale de Lausanne (EPFL), Max-Planck-Gesellschaft zur Förderung der Wissenschaften e.V. (MPG), Johannes Kepler Universität Linz (JKU), Fundacion IMDEA Networks (IMDEA), Mycronic AB (MC), Battioli Paola Società Agricola S.S. (RP), The Circle Società Agricola a Responsabilità Limitata (TC) and Warrant Hub S.p.A. (WH).

RP and TC are customers of the final product, WH handles dissemination and communication during the project and UU, EPFL, MPG, JKU, IMDEA and MC are closely involved in the technical aspects of the project.

Of these project participants, a few have the responsibility for certain subsystems or parts of the complete millirobot. UU is responsible for the sensing subsystem, EPFL is responsible for the locomotion platform and JKU is responsible for the solar panel and energy storage. UU also leads the work of system architecture and integration together with MC, where this thesis is performed.

### 2.3.2 SOMIRO Generation 1

G1 of the SOMIRO millirobot incorporates the main features of computation, locomotion and power system, but leaves out the sensing and communication subsystems originally intended due to the details of these subsystems not being

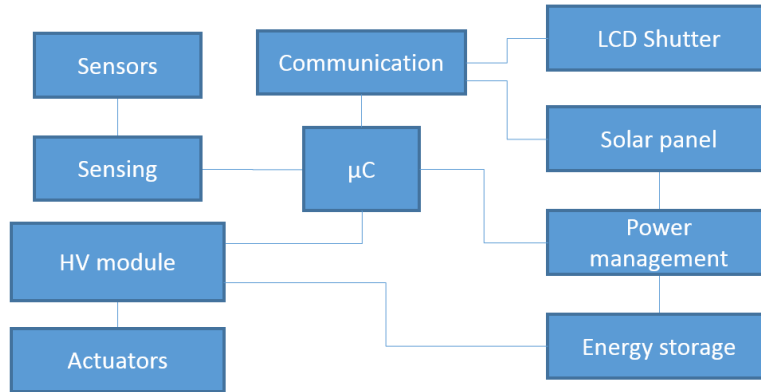


Figure 2.3: A block diagram of the complete millirobot.

finalized. This version of the hardware is meant to provide a basis for the continued work in the SOMIRO project. Alternate components are selected due to the availability problems at the time of writing.

Two designs of the G1 electronics have been produced; the first design has a size of 30 mm by 45 mm resulting in an area of 1350 mm<sup>2</sup> and the smaller version of the same design is 20 mm by 35 mm resulting in an area of 700 mm<sup>2</sup>.

These PCBs integrate an NXP KL02 ARM Cortex microcontroller [7], an e-peas AEM10941 energy harvester [8], a high-voltage flyback converter based on the Linear Technology LT3484 photo-flash charger IC [9], high-voltage switching Metal Oxide Field Effect Transistors (MOSFETs) [10] and an infrared receiver [11] in place of the Visible Light Communication (VLC) down-link.

### 2.3.3 SOMIRO Generation 2

The overall size and area of the robot needs to be reduced while retaining an enhanced feature set. The sensor and communication systems need to be added while the existing solar energy harvesting, energy storage and low power, high voltage actuation systems are retained.

A block diagram of the millirobot can be seen in Figure 2.3.

The complete G2 millirobot should have an area of 100 mm<sup>2</sup> or less, a volume of 1 cm<sup>3</sup> or less and a weight of 1 g or less.

Given these target specifications, it is apparent that several methods of reducing the overall size of the electronics must be utilized. The PCB may need to be of a folding construction or in some other way sandwich several component layers to meet the footprint requirement. Some components may have to be substituted for smaller equivalent parts or the circuit may need to be redesigned.

This thesis investigates the challenges regarding integrating all functionality meant for the final product into the above-mentioned constraints and

produce a PCB design that is tested to evaluate the electrical performance of the millirobot.

The locomotion, solar panel, energy storage and sensor systems are not at a stage where they can be integrated into the G2 version of the millirobot at the time of writing. Therefore, this thesis cannot take these parts into account when designing the footprints for connecting these parts to the PCB. However, the footprints should contain the correct signal connections for these parts.

## 2.4 Related work

Some related work is presented in this section that has served as inspiration or is related to this thesis.

### 2.4.1 RoACH

A project performed at the University of California, has produced a small autonomous robot called RoACH [12]. The electronics package has a main PCB measuring 24 mm by 14 mm and an additional power conversion PCB that measures 18 mm by 14 mm. This includes a PIC microcontroller, power regulation circuitry including a DC to DC boost converter, infrared communication and actuator control circuitry.

This project is relevant to SOMIRO since it integrates a few similar features into a similar size and weight, from which some inspiration may be gathered for the PCB design.

### 2.4.2 mROBerTO

Another related project is mROBerTO 2.0 [13], which is an autonomous millirobot that employs an interesting module-based approach for its construction, where several boards are stacked on top of each other. This uses the footprint area of the millirobot efficiently to include a large amount of electronics.

The construction technique of mROBerTO serves as inspiration for this thesis, being able to reduce the footprint of the electronics while keeping the amount and size of the parts the same as before.

### 2.4.3 SINTEC

The SINTEC project works towards producing flexible and stretchable electronics that use liquid metal interconnects to replace parts of existing flexible or rigid electronics substrates. The flexible substrates can be worn like band-aids, while providing connectivity to sensors and data logging devices [14].

This type of stretchable PCB could possibly be used in the SOMIRO project as a replacement for the previously presented PCB types, but this application has not yet been demonstrated.

## 2.5 Summary

This chapter has presented background information for different types of PCBs and their properties, as well as component packaging levels used in industry. Further, the SOMIRO project organization and existing work has been presented together with a more detailed description of the work that should be performed in this thesis.

Lastly, some related work has been presented that has served as inspiration or is related to this thesis.

With this information, it is apparent that a flexible PCB or a flex-rigid PCB is required to effectively use the area and volume available for the electronics of the SOMIRO millirobot. The interconnect density and stability of the rigid PCB combined with the small bending radius of a one or two-layer flexible interconnect between rigid sections makes a flex-rigid board the most suitable option for this project. Additionally, the components that are used need to be standard, "off the shelf" components due to the limited time available. Therefore, only standard SMD components are used.

# Chapter 3

## PCB Design

Reducing the size of the millirobot is done through the following main steps:

- Produce a set of schematics containing all components intended for the final product, based on the existing prototype schematics.
- Produce a PCB layout containing all components specified in the schematics, conforming to the sizing specifications and PCB design rules.
- Pass design review of team members and manufacturer agents.
- Order and assemble the PCB.
- Perform electrical measurements on the assembled PCB and the existing prototype.

A simple theoretical method of determining if the given circuit will fit in a given footprint is to analyze the area taken up by only the components without considering any specific layout on a PCB.

The flaw with this method is that unless the density, i.e. what percentage of the PCB is covered with components, is estimated accurately, it potentially gives a poor indication of the required PCB area. The interconnects and minimum feature sizes on the PCB can contribute significantly to the final area required.

Therefore, it may be hard to draw any conclusions as to the viability of reducing the size of a given circuit without attempting to do so using a suitable Computer Aided Design (CAD) tool with a set of design rules that are obtained from a prominent PCB manufacturer in industry.

A initial estimate using the component area was done and suggests that all components should fit in less than  $200 \text{ mm}^2$  assuming 50 % additional area for placement and routing. With this, the design process is started.

## 3.1 Schematics

The schematics for the complete millirobot are assembled from several sources. The main source being the existing G1 design produced by EPFL, that includes the main features of computation, energy harvesting and high voltage actuation.

In addition to this, the communication and sensing subsystems are added, based on the testing done by IMDEA and UU respectively.

Some changes are made to the schematics, such as selecting alternate components with similar performance that are more suited to the decreased size of the G2 version.

### 3.1.1 High voltage actuation

The high voltage generation circuit is based on a photo-flash charger IC, LT3484 [9]. This IC is paired with a miniature transformer to form a flyback converter. With a transformer winding ratio of 1:10.2, this circuit should produce approximately 300 V to power the actuation platform, given a nominal input voltage of 3.6 V.

The input decoupling capacitor for this circuit was a 4.7  $\mu\text{F}$  ceramic capacitor. This is replaced by a 10  $\mu\text{F}$  capacitor to provide additional stability and to consolidate the Bill of Materials (BOM).

The BAS521LP [15] diodes selected by EPFL were not available and CMOD2004 [16] diodes are used instead. They have a lower reverse voltage rating, but since two of them are connected in series, this should not be a problem.

There are discharge resistors across the output of the high voltage generator and each of the actuator outputs in order to dissipate the high voltage when the circuit is switched off and not in use. There is a trade-off to be made between parasitic power draw with the high voltage generator enabled and the discharge time of the actuators. The tests done by EPFL suggests that 20 M $\Omega$  resistors should be used to achieve a good balance between power draw and actuator performance. These resistors need to withstand the voltage applied across them, and therefore specialty high voltage SMD resistors were chosen that have a working voltage of up to 400 V.

To control each actuator, a low-side MOSFET switch is used. The suggested DMN30H4D0LFDE [10] MOSFETs were not available and TN2130 [17] MOSFETs were used instead. This is a reasonable compromise due to the low average power dissipation in these devices and the smaller size is beneficial in this case.

### 3.1.2 Power management

The power management circuit is based on the AEM10941 IC [8]. This IC incorporates a DC to DC boost regulator for charging a storage element, selectable Maximum Power Point (MPP) voltage and Low Dropout Regulators (LDOs) for regulated outputs. Both under-voltage and over-voltage limits are configurable for different storage elements.

Of the two regulated outputs available on the IC, the higher-voltage variant is used to provide a 3.3 V supply for the microcontroller and all subsystems on the millirobot.

A switched, unregulated status signal is available from the IC, which can be used to measure the storage element voltage using a voltage divider. This feature makes it possible to estimate the available energy in the storage element and decide what software tasks to run at which intervals, given that some tasks are more energy intense than others. At the same time, the voltage divider always consumes some power, but is in this case switched off when the storage element voltage reaches the lower threshold set by the storage element configuration, eliminating additional parasitic draw. High value resistors are used to minimize the static power draw, but any leakage on the microcontroller input or the PCB can change the measured voltage.

The solar panel developed by JKU is connected to the solar input of the harvester after having passed through the low-pass filter described in 3.1.5.

### 3.1.3 Sensing

The sensing subsystem consists of three sensing elements, measuring two chemical concentrations and temperature. All three sensing elements are resistive and are therefore paired with reference resistors to form voltage dividers as described in [18]. The resulting voltages can be measured using the Analog to Digital Converter (ADC) in the microcontroller.

To save energy, the sensors are only active when performing measurements through supplying them with an output from the microcontroller that is switched off when not actively performing measurements.

The sensing elements are developed by UU and prototypes are available, but for simplicity, a set of representative fixed resistors are used instead of the sensing elements in this thesis. The reference resistors are 100 k $\Omega$  and the substitute resistors are 1 k $\Omega$ , 10 k $\Omega$  and 1 M $\Omega$  to give a range of values for testing.

### 3.1.4 Computation

An NXP Kinetis KL02 ARM Cortex-M0+ microcontroller [7] is used because of its versatility, small size and low power consumption. 4 kB of Random-

Access Memory (RAM) and 32 kB of flash storage is available on the MKL02Z32CAF4R model that has been selected for the millirobot at this time.

The microcontroller has a 12-bit Successive Approximation Register (SAR) ADC, an analog comparator with an internal reference Digital to Analog Converter (DAC), two two-channel timers with Pulse Width Modulation (PWM) capability and a low-power timer. These features are present on a subset of the 18 General-Purpose Input/Output (GPIO) pins available on the microcontroller and this requires some attention to the pin assignment.

By default, most internal peripherals are disabled in order to reduce the power consumption. This is important to keep in mind when writing the software for the microcontroller, as the software and strategies for performing tasks has a large impact on overall power consumption for the millirobot.

### 3.1.5 Communication

The communications channel should be implemented using VLC technology. Since a solar panel is already used for energy harvesting, this can also be used for the communication down-link to the millirobot. A filter network separates the down-link signal from the energy harvesting system.

Since the energy harvesting system should primarily handle DC current, a low-pass filter is used to both isolate the communication down-link from the harvester, avoiding signal strength reduction and to reduce the switching noise from the harvester that risks severely degrading the Signal-to-Noise Ratio (SNR) of the communication down-link. Likewise, a high-pass filter is used to separate out the communication down-link before the detection circuitry.

The filter frequencies should be as low as possible to not attenuate the communication down-link, but the size of the filter components should also fit in the limited space available. Currently, the values suggested by IMDEA are  $100\ \Omega$  and  $100\ \mu\text{F}$  for the low-pass filter together with  $5.1\ \text{k}\Omega$  and  $1\ \mu\text{F}$  for the high-pass filter. This gives filter frequencies of around 16 Hz and 31 Hz respectively. The suggested down-link frequency range is 100 Hz to 1 kHz.

A  $100\ \mu\text{F}$  capacitor is physically large if factors such as DC leakage should be taken into consideration. A tantalum capacitor would be small enough but has significant leakage current compared to a ceramic capacitor. Ceramic capacitors have negligible leakage, but instead have a voltage dependent capacitance. The trade-off made is to use four  $22\ \mu\text{F}$  ceramic capacitors that are as small as possible and more easily conform to the space available, at the expense of some additional area on the PCB. The capacitance will be less than the specified  $88\ \mu\text{F}$  when the capacitors are charged to a nominal DC level, increasing the filter frequency.

After the communication down-link has passed through the filter network, there are several ways of detecting the low-level signal that is received from the

solar panel. The suggested solution is to use a comparator with a 0 V reference, giving a logic level signal that indicates whether the analog signal is above 0 V. With the high-pass filter in place, the signal received will be centered around 0 V and hence it is possible to recreate a signal that modulates the light that shines on the solar panel from an external transmitter unit. The limitations being the comparator offset voltage, hysteresis and noise levels from various sources.

An alternative to the comparator is the ADC in the microcontroller that, given its 12-bit resolution, should have good performance at low voltage levels. While the microcontroller also has a comparator built in, the performance specifications may not suffice when compared to the suggested external comparator, as can be seen in Table 3.1.

Table 3.1: Performance characteristics of down-link front-end solutions

Circuit	Supply current	Offset voltage (max.)
MCU comparator	20 $\mu$ A	20 mV
MCU ADC	215 $\mu$ A	6 mV
TS881	1 $\mu$ A	10 mV
TLV3691	150 nA	15 mV

The external comparator suggested is a TS881 [19] but this is not available at the time of writing and a similarly performing alternative, the TLV3691 [20], was chosen instead.

## 3.2 Layout

The PCB layout attempts to arrange all components in a way that simplifies the interconnects between subsystems on the PCB itself and to the components that attach to the PCB, such as actuator platform, sensors, solar cell and energy storage.

The PCB is split into several sections that house a subset of the complete circuit. Each section is a rigid FR4 board and these are interconnected using flexible sections. This forms a Flex-Rigid PCB that can be folded to make a multi-layer sandwich, giving the board area required for all components, while conforming to the footprint area requirement.

Special care has to be taken when placing the high voltage connections and related features on the PCB since the physical distances required by the IPC-2221B [21] standard are significant when the overall dimensions of the PCB are of the same order of magnitude.

The minimum feature sizes on the PCB have to be considered to allow for manufacturing within a reasonable timeframe. The component sizes chosen

should also allow for manual assembly of the PCB while at the same time not occupying large amounts of the limited space available.

The design rules for the PCB play a significant role in the design process as they dictate many parameters, such as trace width and spacing, minimum pad diameters, hole to edge distances, copper to edge and inner copper to flex transition, among others. [22] and [23] are used to provide the design rule parameters for the PCB.

### 3.3 Assembly

The PCB produced is manually assembled using a combination of hot air reflow and hand soldering with a soldering iron. The BGA pads for the microcontroller are lightly tinned and solder flux is applied before it is placed on the board using tweezers. Then, hot air is used to flow the existing solder on the microcontroller onto the pads on the PCB. For all other components, the pads are tinned and components are flowed into place using a soldering iron and hot air where beneficial.

Figures 3.1 and 3.2 show the assembled PCB. The high voltage connections near other low voltage connections are covered with insulating tape to minimize the risk of unintentional charge transfer. A pinout modification can be seen on the bottom side. This is explained in Section 5.3. The connections to the external parts emulating the actuators, the storage element and the solar panel are left out on these images for clarity.

Figure 3.3 shows the PCB in the folded state, held together with hot melt glue. The alignment of the rigid sections is not perfect since no assembly jig was used.

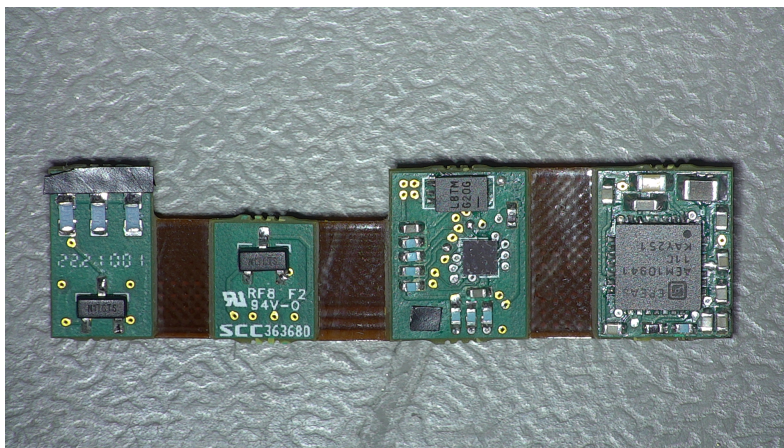


Figure 3.1: Top side of the G2 prototype PCB.

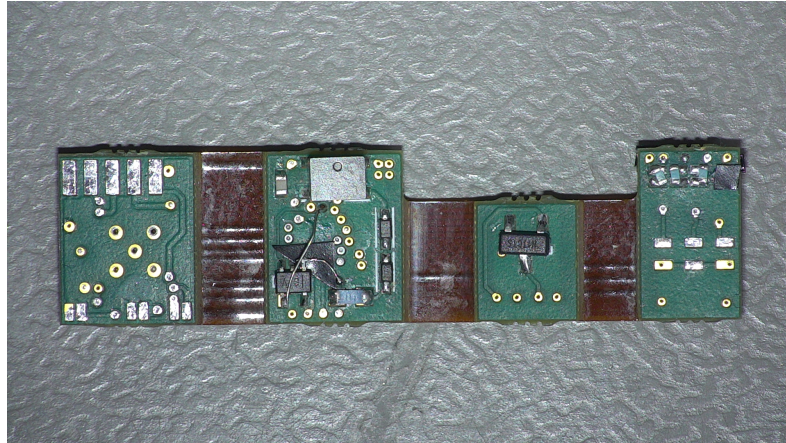


Figure 3.2: Bottom side of the G2 prototype PCB.



Figure 3.3: The G2 prototype PCB folded into the intended shape, a USB connector for reference.

The G1 PCB can be seen in Figure 3.4 and 3.5, where the same components are used. There is one additional  $1\ \mu\text{F}$  filter capacitor by the comparator on the bottom side and an LED in place of the IR receiver to read out status messages. The filter capacitor for the low-pass filter on the solar input is not mounted, otherwise both prototype boards contain the same set of components.

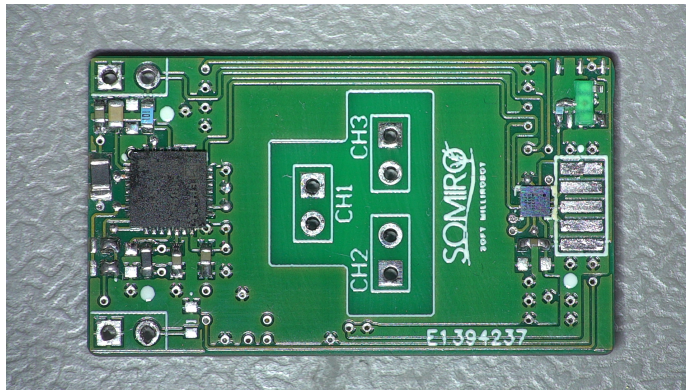


Figure 3.4: Top side of the G1 prototype PCB.

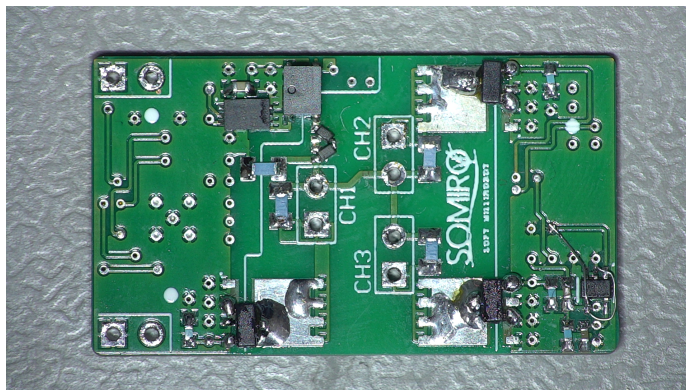


Figure 3.5: Bottom side of the G1 prototype PCB.

### 3.4 System documentation

The schematics can be found in Appendix A and include all subsystems of the millirobot. The layout of the PCB can be found in Appendix B where the individual layers of the board and 3D views are presented.

In addition to these documents, there are design files that are shared with the SOMIRO team to allow for further development of the schematics and layout as the G2 version of the millirobot is finalized.

The software used for testing in Chapter 4 is available in Appendix C.

## Chapter 4

# Experimental Evaluation

In this chapter, the prototype produced is evaluated and the measurement results are presented.

The measurements to be performed on the existing prototype and the newly designed PCB are several power consumption measurements under various conditions, such as idle, sensing and communicating, as well as efficiency measurements for charging and discharging the battery via the power management system. The physical parameters of the PCB are also presented.

### 4.1 Power consumption measurements

The power consumption measurements of the complete millirobot are done by supplying a regulated voltage at the battery connections through a  $10\ \Omega$  current sense resistor. The voltage across the resistor is measured using a precision voltmeter, a Keithley 2182A, set to the highest averaging time in order to minimize the effects of measurement noise and to get a stable readout.

The fully assembled PCB is connected to circuit equivalents of the external components, such as the sensors and actuators. A set of  $1\ \text{nF}$  capacitors were used instead of the actuator platform, one per actuator connection. This value has been used in previous tests by EPFL and is a good representation of the actuator. The sensing elements are substituted by a set of resistors,  $1\ \text{k}\Omega$ ,  $10\ \text{k}\Omega$  and  $1\ \text{M}\Omega$ , which are paired with  $100\ \text{k}\Omega$  reference resistors. This gives a range of values for testing the ADC inputs.

These equivalents are chosen in order to give a consistent test setup for both the G1 prototype and the G2 prototype, simplifying the comparison between the prototype versions.

For all power consumption tests, Timer/PWM Module 1 (TPM1) is used as a timekeeping device that periodically sends an interrupt, possibly waking the microcontroller from Very Low Power Wait (VLPW), to increment a counter variable and continue execution of any tasks that are due to run. Software

timekeeping produces the control waveform for the high-voltage generator circuit when it is active. All outputs are used as digital outputs and all inputs are sampled by the internal ADC, except for the communication down-link, which is passed through the external comparator, resulting in a digital signal.

#### **4.1.1 Idle power consumption**

By disabling all tasks and keeping TPM1 running, the duty cycle of active versus wait mode is as low as possible. All external parts such as sensors, actuators and communication are inactive.

With the microcontroller in VLPW mode, where operation can be resumed by timer interrupts, the power consumption is reduced significantly compared to Very Low Power Run (VLPR) without switching off the 3.3 V output from the power management circuit.

#### **4.1.2 Active power consumption**

With the microcontroller in VLPR mode, no-op instructions are executed in a loop to generate a consistent power draw. Here, no other internal parts of the microcontroller and no external parts, such as sensors, actuators or communication, are active. The VLPW mode is disabled to keep the microcontroller in VLPR mode.

#### **4.1.3 Sensing power consumption**

When the sensing components are active, the ADC in the microcontroller is also active in order to read out the resulting voltage levels from the respective sensing elements. These tests implement periodic sensing, where both the sensors and ADC are switched off in-between measurement cycles.

However, this test is set up to run the task continuously without entering VLPW mode. This gives the maximum power consumption value, which can be used in conjunction with a duty cycle value to estimate the average power consumption for a certain task schedule.

#### **4.1.4 Actuating power consumption**

Controlling the actuators is done though periodically stepping through a state-machine toggling the pins controlling the high-voltage generator and the actuators.

The high-voltage generator is only active for a short period of time after it has been enabled, since it is not loaded with a large capacitor. Therefore, it is sent a pulse each time the actuator control signals are changed. This

charges the active actuator or actuators and the discharge resistances across the actuators discharges them.

Here, the actuation task is also configured to run continuously with minimal wait time between cycles to generate a maximum power consumption value.

#### **4.1.5 Communication power consumption**

Receiving communication on the VLC downlink entails counting the number of edges in the received signal during a given time period to estimate the frequency of the received signal. The frequency received represents a symbol that can trigger a certain action or be part of a binary data stream for instance. The amount of time spent in VLPR is increased due to the interrupts that are triggered for each edge in the received signal.

Transmitting using the VLC uplink enables the timer used to supply a bipolar waveform to the VLC shutter, or alternatively a software equivalent can be used. The two waveforms sent to the VLC shutter should either in phase or out of phase, resulting in an inactive or active shutter, representing binary high and low values respectively.

This test is performed with the VLPW mode active between runs of the tasks. Higher power consumption is expected due to the additional interrupts when receiving a signal on the downlink. This test does not use a timer to produce the bipolar waveform due to the resource conflict of using the same timer for sending and receiving tasks present in the code. This is solved by using the low power timer for one of these tasks, but this was not implemented in this project.

A down-link frequency of 800 Hz is applied to the solar panel connections from a low impedance source without DC bias to simulate the conditions that can be expected when receiving communication.

#### **4.1.6 Summary of results**

From the tests described, the results in Table 4.1 are obtained. Both the G1 and G2 prototypes are running the same software except for the header files that specify which pins on the physical microcontroller are used for what function.

The last row of Table 4.1 contains power consumption values where the microcontroller is in Very Low Leakage Stop (VLLS) mode. This mode is the lowest power consumption mode available on the microcontroller and is used to evaluate the charging and discharging efficiencies in Section 4.2.

Table 4.1: Power measurement results

Scenario	G1 prototype	G2 prototype
Idle	0.686 mW	0.690 mW
Active	1.112 mW	1.119 mW
Sensing	N/A	5.66 mW
Communicating	N/A	2.5 mW
Actuating	27 mW	19 mW
VLLS	9.75 $\mu$ W	8.97 $\mu$ W

## 4.2 Efficiency measurements

The charging tests are done in a similar way of using current sense resistors, in this case on both the battery and solar connections. 100  $\Omega$  resistors were used to increase the voltages as they are read out using an oscilloscope, a Rigol DS1054Z, instead of a voltmeter. This is done in order to obtain a visual indication of when the current draw has stabilized and to increase the number of samples used in the averaging calculations. It is also possible to take a snapshot of all relevant values at the same point in time, potentially yielding more accurate results as the measured voltages do not have time to change between multiple voltage measurements for the same scenario.

### 4.2.1 Charging efficiency

The charging efficiency of the power management circuit is evaluated by disabling all outputs from the microcontroller and entering VLLS mode to reduce the power consumed by the microcontroller and other components. A range of input voltages is applied to the solar input, which results in a range of input currents due to the MPP mode of the harvester. This is done for a few representative storage element voltages.

Figure 4.1 shows the conversion efficiency graphs for this test.

### 4.2.2 Discharging efficiency

The discharging efficiency of the power management circuit is likewise evaluated by disabling all outputs from the microcontroller and entering VLLS mode. A fixed load resistance is applied across the 3.3 V output of the harvester and the storage element voltage is varied.

The discharging efficiency graphs can be seen in Figure 4.2.

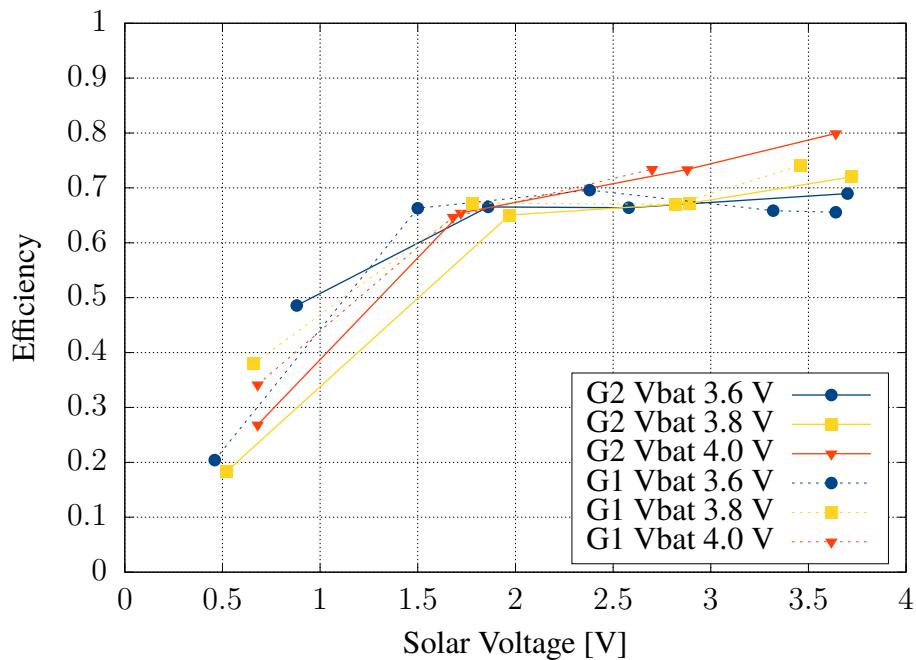


Figure 4.1: Charging efficiency for a set of representative storage element voltages and solar input voltages.

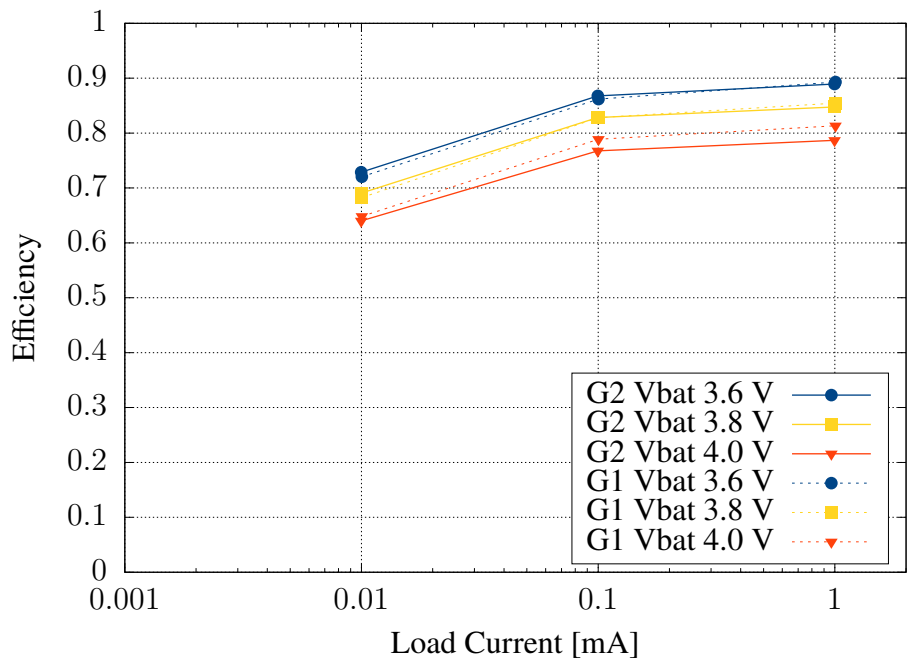


Figure 4.2: Discharge efficiency for a representative set of storage element voltages and load resistances.

### 4.3 Physical parameters

The PCB without components, as extracted from the delivery panel, weighs 580 mg. This is more than the 296 mg estimate for pure FR4 without any copper.

After all components are mounted, the weight is 950 mg. This gives a combined component and solder weight of 370 mg. The estimated component weight is 258 mg.

When the assembled board is folded, the dimensions are measured to 8 mm by 9.5 mm by 10 mm. These dimensions are close to the 8.1 mm by 10 mm by 10 mm estimate based on the CAD model.

### 4.4 Reliability and validity assessment

The equipment used is deemed accurate enough for the measurements performed. The current sense resistors used are also selected to produce suitable voltages for the measurement equipment used. The resistors are measured to be well within their 1 % tolerance and are therefore assumed to be their ideal value when calculations are performed.

The value of the resistors is assumed to be constant throughout the tests performed, as the power dissipated in the resistors is minimal. The measurement equipment is also assumed to not drift in value as the tests are performed and the accuracy is deemed sufficient for the precision of data presented in the results chapter.

# Chapter 5

## Discussion

This chapter contains the discussion and comments on the results presented, the design and choices made for this thesis.

### 5.1 Comments on results

The two PCBs compared are similarly populated, in order to make the comparison more between the PCBs and the circuit differences of adding the sensing and communication. It should be noted though, that the G1 PCB does also have a comparator and the provisions for both high-pass and low-pass filters for the solar input connections. The low-pass filter does not have the filter capacitor installed due to assumptions on physical size for the footprint.

The G2 PCB is fully populated according to the schematics in Appendix A, with the only exception being the sensing elements that are replaced with fixed resistors according to Section 3.1.3.

The most prominent difference is the actuating power consumption, where the G2 PCB consumes significantly less power. The voltage on the battery connections, after the current sense resistor, was observed to be more stable and contain less noise on the G2 PCB compared to the G1 PCB. The G1 PCB even shut down due to low voltage after a few seconds of actuation, even though the average voltage was above the 3.6 V minimum with a good margin. This can most likely be attributed to the placement and routing of connections to the decoupling capacitor that should act as an energy reserve for the high voltage generator. The placement of this on the G2 PCB attempts to provide a minimal loop area and minimal inductance between the transformer, capacitor and switching IC. This seems to have made a significant difference. Note that the performance on the high voltage side has not been studied closely for either PCB, however the performance should be similar in both cases.

The other results presented in 4.1 are quite similar, as expected by the nearly identical circuits. The main addition that should increase the power consumption is the voltage divider for measuring the storage element voltage.

With the 20 M $\Omega$  load added, an additional 195 nA or 0.76  $\mu$ W can be expected for the G2 PCB. Table 4.1 shows a larger difference than this for most scenarios, except for the VLLS mode where the microcontroller is in its lowest power mode. Measurement noise is a likely contributor.

## 5.2 System design

The configuration of the microcontroller can have a significant impact on the power consumption, as noted when the boards were tested during the assembly process where some boards had significantly higher power consumption than others while running the exact same software. The cause was narrowed down to the floating pins on the microcontroller that, when configured as inputs as they are by default, can result in higher current draw when charged to an arbitrary voltage. Either disabling the clock signals to the GPIO peripherals in the microcontroller or discharging all floating pins resulted in a reduced current as the amount of unintentional switching in the input logic was reduced.

The power consumption for communication without the external comparator was not tested but could be interesting to explore. Using the internal ADC may work as well or better than the external comparator and is more flexible in the ability to dynamically select a threshold voltage. The additional power consumption should not be a problem, since a strong light is used when communicating with the millirobot. The additional light will likely generate enough power to offset the ADC power consumption. In addition, the parasitic draw of the external comparator will be eliminated.

A solution for the parasitic draw of the comparator is to power it from a GPIO output on the microcontroller. The current pin assignments do not leave any outputs unused for this purpose, but the two outputs used for enabling the sensors could be combined into a single output and the now free output could be used to power the external comparator when needed. The power saved is not insignificant when looking at Tables 3.1 and 4.1, but it is also not a major part of the total power consumed when the microcontroller is in VLPW or VLPR modes.

The energy harvesting circuit assumes that a solar panel is directly connected to the input without passing through a filter as is present here. This can skew the MPP operation of the harvester, since there is a large amount of capacitance at the input of the harvester. The voltage across these capacitors will slowly change depending on the power drawn by the harvester. A typical behavior that has been observed is that the voltage will drop as the harvester is drawing a nominal amount of power, then the MPP evaluation takes place, the harvester reduces the power draw and the voltage rises while under the lighter load and the cycle repeats. The average voltage across the harvester

input and across the solar panel may differ from the ideal values that would extract the maximum power from the solar panel. A possible mitigation is to place a switch in series with the large capacitance, effectively disconnecting it during the MPP evaluation. There is a convenient output on the harvester that is active as it is performing the MPP evaluation and this could be used to control the switch. Because of the series resistance due to the filter, a different MPP level may also be needed to achieve a more optimal voltage across the solar panel.

### 5.3 Component selection

Since several components previously used by, or recommended by, the SOMIRO team were unavailable during this thesis project, some alternative components had to be selected. Most notably, the comparator for the communication downlink and the MOSFET switches were replaced by similarly performing variants. The G2 prototype uses a SOT-23 footprint for the MOSFETs, for which there are a few alternatives available. The original MOSFETs recommended uses a DFN6 package and matching footprint, which is a bit more compact than the SOT-23 variant. The comparator footprint and pinout were left as-is, since there only is a minor pinout difference between the original TS881 and the TLV3691 used instead. Two pins need to be swapped in order to use the alternate comparator, which was deemed acceptable due to the additional time required to change the PCB layout at the stage this decision was made.

Alternate microcontrollers are being considered at the time of writing, such as the MSP430 series, with Ferroelectric Random-Access Memory (FRAM) instead of flash-based non-volatile memory to allow for retaining the complete memory contents of program execution during a loss of power situation. The alternative is saving the execution status in flash memory when a low energy situation is detected. The main concern being the significantly fewer specified write cycles of flash memory compared to FRAM. The design effort and software complexities in both cases must be considered when selecting the microcontroller for the final version of the millirobot.

### 5.4 PCB layout considerations

The PCB design was done with relative simplicity in mind when it comes to features such as microvias, blind or buried vias and over-plated vias. More of these features can be included in the PCB, but the manufacturing time and cost would likely be increased, which was not desirable for this thesis. The density of the PCB could be increased and clearance to the high voltage connections could be achieved with less area taken up for the same purpose by utilizing

blind and buried vias, together with an increased number of vias placed in solder pads.

In-pad vias should be filled and over-plated with copper to provide an even surface that the automated soldering processes can be used on with consistent results. For the PCB produced, the few vias placed in pads are deemed to be acceptable, since the prototype boards for this project are hand soldered. When the board for the final version of the millirobot is designed, this should be taken into consideration to allow for fully automated assembly processes.

The flex-rigid construction of the board brings some additional considerations for the design process with several additional design rules. Limitations include placement of through-holes close to the flex-rigid transition, minimum length of the flexible regions, minimum copper to edge distance for inner layers next to the rigid-flex transition among others.

In this project, several of these additional design rules had an impact on the design, necessitating reordering of tracks and vias on several sections. The creepage and clearance distances for the high voltage connections also required track and via placement to be carefully considered, should the IPC requirements be met.

The flexible material of the flex-rigid board has a maximum copper coverage percentage to allow for the material to dry properly in manufacturing. This means that solid copper planes cannot be used on the layers that are laminated to the flexible material. A solution to this problem is to use a hatched fill pattern that does not exceed the maximum coverage allowable for the material. This allows for similar performance to a solid plane but is compatible with the flexible substrate.

A feature of the PCB layout software that was not used is the pin-swapping functionality, that allows for simplified routing in cases where several pins of a component can provide the same functionality. A relevant example is the microcontroller, where several pins are interchangeable due to the same features being available on these pins and the software can be configured according to the connections on the schematic. It is however important to verify that the functionality required is available on the pins in question, since the pin mux in the microcontroller does not provide unlimited flexibility. The pin-swapping feature was not used since the layout was possible to produce with the pinout selected, stack-up chosen and design rules for the same stack-up, but it could be a useful tool in further revisions of the PCB provided it is set up correctly.

## 5.5 Software considerations

The software that runs on the millirobot has a significant impact on the power consumption. The usage of low power modes is required in order

to achieve energy autonomy. Especially the various stop-modes that were not investigated in this thesis can be useful in reducing the idle power consumption. The software scheduling of tasks regarding periodicity must be carefully considered for the final millirobot. Charge state indication from the storage element should be used to decide when to run certain tasks.

Any peripherals in the microcontroller should only be enabled when running tasks that require them, in order to decrease the power consumed. The relevant peripherals are the ADC, timers and all external components such as sensors and actuators.

Additionally, GPIO pins that are floating should be disabled or set to a defined voltage through setting the pin to an output. When this is not done, the charge present on the pin can cause higher power consumption through random switching of the logic in the input circuit, due to the high impedance nature of the Complementary Metal Oxide Semiconductor (CMOS) input.

The software in Appendix C is not intended to be used directly in the final version of the millirobot, but some methods used may be important to consider. Whenever possible, do not use busy-wait, do implement a dynamic task schedule based on external factors such as storage element charge level, charge rate, time since last communication message successfully received, etc. The overall duty cycle of the millirobot can be exceedingly low if the tasks and schedule are done properly.



## Chapter 6

# Conclusions and Future work

This chapter contains the conclusions, limitations and future work.

### 6.1 Conclusions

This thesis has investigated some of the challenges related to electronics miniaturization, specifically regarding PCB design.

The design rules for commercially available PCB manufacturing services allow for quite small electronic systems using industry standard materials and standard components.

The PCB design produced meets the main goals set out for this thesis of incorporating all features intended for the SOMIRO millirobot. The design does not leave large amounts of weight or height to the external parts that are to be attached to the PCB, but as these parts are not yet finalized, they cannot be included in the overall assessment of sizing and weight for the complete millirobot. It is apparent that the energy storage element must be exceedingly thin and light in order to fit within the 50 mg and 2 mm tall envelope that is left together with the solar panel and actuator platform. It may not be impossible to construct a supercapacitor with sufficient performance that fits within these bounds, but it may be very difficult.

There is still room for improvement of the PCB using additional HDI structures and possibly customized stackups to potentially reduce the volume of the electronics even further, leaving a bit more room for the external parts.

With this, the research question can be answered: *Yes, it is possible to reduce the footprint of the SOMIRO swimming millirobot to 100 mm<sup>2</sup>, the volume to 1 cm<sup>3</sup> and the weight to 1 g while consuming a similar amount of energy and integrating additional features.*

The goals of producing a PCB design integrating all functionality intended for the final millirobot, manufacturing, assembling and verifying performance of said PCB, have been met. Additionally, all deliverables are included in this

report and the design files produced have been made available to the SOMIRO team.

## 6.2 Limitations

The PCB does not utilize HDI structures such as blind or buried vias which could help in reducing the PCB area further. This is mainly due to time limitations and concerns regarding design rules for these extra features, should they be possible to manufacture with the stackup used.

The test software written does not implement all functionality that should be included in the final version of the millirobot. Not all peripherals available in the microcontroller were used to their full potential and therefore, the power consumption results may not fully represent the power draw that can be expected for a more complete software solution. The results presented may be optimistic in some scenarios and pessimistic in others. The aim was to get as close as possible to a set of realistic power consumption values with the limited time and effort that could be put towards the software.

## 6.3 Future work

The design produced in this project should provide a good starting point for the final design of SOMIRO. Producing the final design together with the SOMIRO team is the next major step when all the external components, such as solar panel, energy storage, sensors and VLC shutter are finalized.

The sizing constraints for this project were applied to the electronics package, i.e. the PCB but the external components may indicate that a redesign of the PCB is required. Either because of sizing, if the given limits are to be respected, or because of the footprints interfacing with the external components. The footprints used are only a connection surface for test leads and should be customized for the external components they interface with, once the design of these components is finished.

A more advanced structure utilizing blind and buried vias between inner layers should allow for an even more compact PCB, where the outer layers are primarily used to mount components, not route any signals. This technique is commonly found on more advanced smartphone PCBs, where the component density is exceedingly high in some cases.

# References

- [1] R. R. Tummala, *Fundamentals of Microsystems Packaging*, S. Chapman, Ed. McGraw-Hill, 2000. [Pages 5, 6, and 7.]
- [2] Würth Elektronik. HDI Design Guide. Accessed 2022-03-04. [Online]. Available: [https://www.we-online.com/web/en/index.php/show/media/04\\_leiterplatte/2011\\_2/relaunch/produkte\\_5/microvia\\_hdi/180924\\_W E\\_CBT\\_DesignGuide\\_HDI-12\\_EN\\_screen.pdf](https://www.we-online.com/web/en/index.php/show/media/04_leiterplatte/2011_2/relaunch/produkte_5/microvia_hdi/180924_W E_CBT_DesignGuide_HDI-12_EN_screen.pdf) [Page 5.]
- [3] Würth Elektronik. Flex-rigid circuit boards: Standard for the third dimension. Accessed 2022-06-08. [Online]. Available: [https://www.we-online.com/web/en/leiterplatten/produkte\\_/3d\\_starr\\_flexible\\_leiterplatten/Einleitung\\_3D\\_starrflex\\_leiterplatten\\_pcb.php](https://www.we-online.com/web/en/leiterplatten/produkte_/3d_starr_flexible_leiterplatten/Einleitung_3D_starrflex_leiterplatten_pcb.php) [Page 7.]
- [4] PCBWay. What is a Rigid-Flex PCBs. Accessed 2022-06-08. [Online]. Available: [https://www.pcbway.com/pcb\\_prototype/What\\_is\\_a\\_Rigid\\_Flex\\_PCBs.html](https://www.pcbway.com/pcb_prototype/What_is_a_Rigid_Flex_PCBs.html) [Page 7.]
- [5] iPCB Circuits Limited. Low Temperature Co-fired Ceramic PCB(LTCC PCB). Accessed 2022-02-08. [Online]. Available: <https://www.ipcb.com/spcb/461.html> [Page 7.]
- [6] Soft Milli-robots. Accessed 2022-02-08. [Online]. Available: <https://cordis.europa.eu/project/id/101016411> [Page 9.]
- [7] NXP Semiconductors. Kinetis KL02 32 KB Flash 48 MHz Cortex-M0+ Based Microcontroller. Accessed 2022-07-15. [Online]. Available: <https://www.nxp.com/docs/en/data-sheet/KL02P20M48SF0.pdf> [Pages 10 and 15.]
- [8] E-Peas. Highly efficient, regulated dual-output, ambient energy manager for up to 7-cell solar panels with optional primary battery. Accessed 2022-09-11. [Online]. Available: <https://e-peas.com/wp-content/uploads/2022/09/e-peas-AEM10941-datasheet-solar-energy-harvesting.pdf> [Pages 10 and 15.]
- [9] Linear Technology. LT3484-0/LT3484-1/LT3484-2 Photoflash Capacitor Chargers. Accessed 2022-06-08. [Online]. Available:

- <https://www.analog.com/media/en/technical-documentation/data-sheets/3484012f.pdf> [Pages 10 and 14.]
- [10] Diodes Incorporated. DMN30H4D0LFDE N-Channel Enhancement Mode Mosfet. Accessed 2022-07-15. [Online]. Available: <https://www.diodes.com/assets/Datasheets/DMN30H4D0LFDE.pdf> [Pages 10 and 14.]
- [11] Vishay Semiconductors. IR Receiver Modules for Remote Control Systems. Accessed 2022-07-15. [Online]. Available: <https://www.vishay.com/docs/82599/tsop372.pdf> [Page 10.]
- [12] A. M. Hoover, E. Steltz, and R. S. Fearing, “Roach: An autonomous 2.4g crawling hexapod robot,” in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008. doi: 10.1109/IROS.2008.4651149 pp. 26–33. [Page 11.]
- [13] K. Eshaghi, Y. Li, Z. Kashino, G. Nejat, and B. Benhabib, “mROBerTO 2.0 – an autonomous millirobot with enhanced locomotion for swarm robotics,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 962–969, 2020. doi: 10.1109/LRA.2020.2966411 [Page 11.]
- [14] Soft intelligence epidermal communication platform. Accessed 2022-02-08. [Online]. Available: <https://cordis.europa.eu/project/id/824984/reporting> [Page 11.]
- [15] Diodes Incorporated. BAS521LP High Voltage Switching Diode. Accessed 2022-07-15. [Online]. Available: [https://www.diodes.com/assets/Datasheets/products\\_inactive\\_data/BAS521LP.pdf](https://www.diodes.com/assets/Datasheets/products_inactive_data/BAS521LP.pdf) [Page 14.]
- [16] Central Semiconductor Corp. CMOD2004 Surface Mount High Voltage Silicon Switching Diode. Accessed 2022-07-15. [Online]. Available: <https://my.centralsemi.com/datasheets/CMOD2004.PDF> [Page 14.]
- [17] Microchip Technology Inc. TN2130 N-Channel Enhancement-Mode Vertical DMOS FET. Accessed 2022-07-15. [Online]. Available: <http://www1.microchip.com/downloads/en/DeviceDoc/TN2130-N-Channel-Enhancement-Mode-Vertical-DMOS-FET-Data-Sheet-20005944B.pdf> [Page 14.]
- [18] N. X. Thai, N. Van Duy, C. M. Hung, H. Nguyen, T. M. Hung, N. Van Hieu, and N. D. Hoa, “Realization of a portable h2s sensing instrument based on sno2 nanowires,” *Journal of Science: Advanced Materials and Devices*, vol. 5, no. 1, pp. 40–47, 2020. doi: <https://doi.org/10.1016/j.jsamd.2020.01.003>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2468217920300034> [Page 15.]

- [19] STMicroelectronics. TS881 Rail-to-rail 0.9 V nanopower comparator. Accessed 2022-07-15. [Online]. Available: <https://www.st.com/resource/en/datasheet/ts881.pdf> [Page 17.]
- [20] Texas Instruments Incorporated. TLV3691 0.9-V to 6.5-V, Nanopower Comparator. Accessed 2022-07-15. [Online]. Available: <https://www.ti.com/lit/ds/symlink/tlv3691.pdf> [Page 17.]
- [21] IPC-2221B, *Generic Standard on Printed Board Design*. IPC International Inc., 2012. [Page 17.]
- [22] Würth Elektronik. Basic Design Guide. Accessed 2022-03-04. [Online]. Available: [https://www.we-online.com/web/en/index.php/show/media/04\\_leiterplatte/2011\\_2/relaunch/produkte\\_5/012012\\_Basic\\_Design\\_Guide.pdf](https://www.we-online.com/web/en/index.php/show/media/04_leiterplatte/2011_2/relaunch/produkte_5/012012_Basic_Design_Guide.pdf) [Page 18.]
- [23] Würth Elektronik. Design rules Flex-rigid xRi - 2F - xRi. Accessed 2022-03-24. [Online]. Available: [https://www.we-online.com/web/en/index.php/show/media/04\\_leiterplatte/2012\\_2/3d\\_2/design\\_rules\\_neu/\\_CBT\\_Check\\_PM\\_02\\_en.pdf](https://www.we-online.com/web/en/index.php/show/media/04_leiterplatte/2012_2/3d_2/design_rules_neu/_CBT_Check_PM_02_en.pdf) [Page 18.]



# **Appendix A**

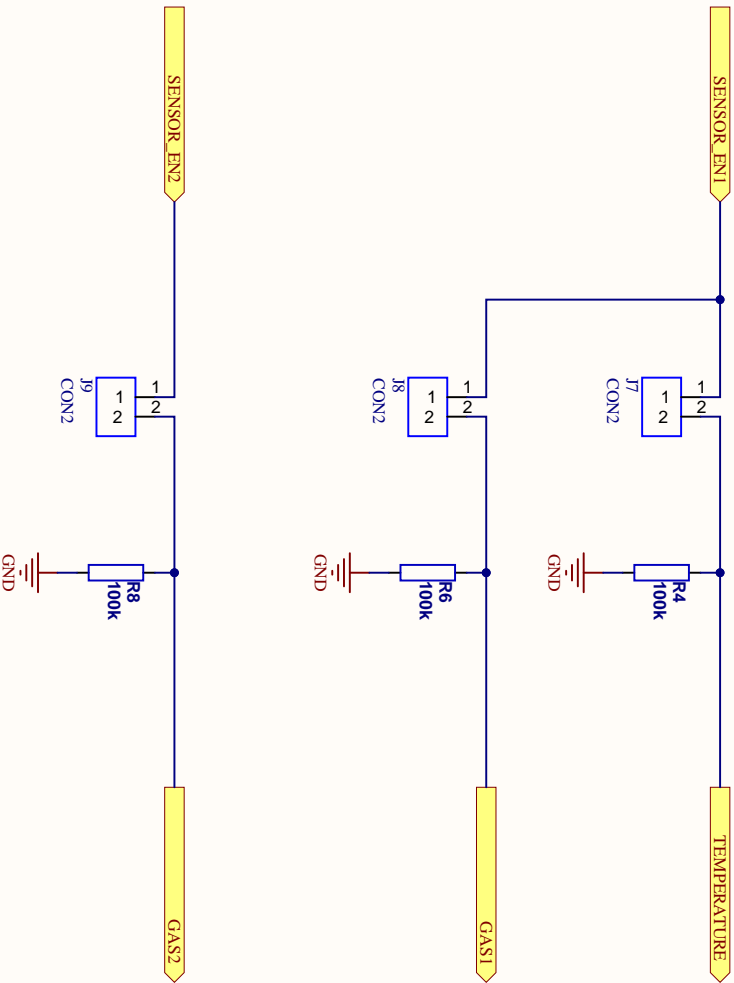
## **Schematics**

This appendix contains the schematic diagrams for the electronics produced in this thesis.









Title: sensing\_SchDoc

Project: SOMIRO\_G2\_prototype.RjPcb

Time: 14:11:06

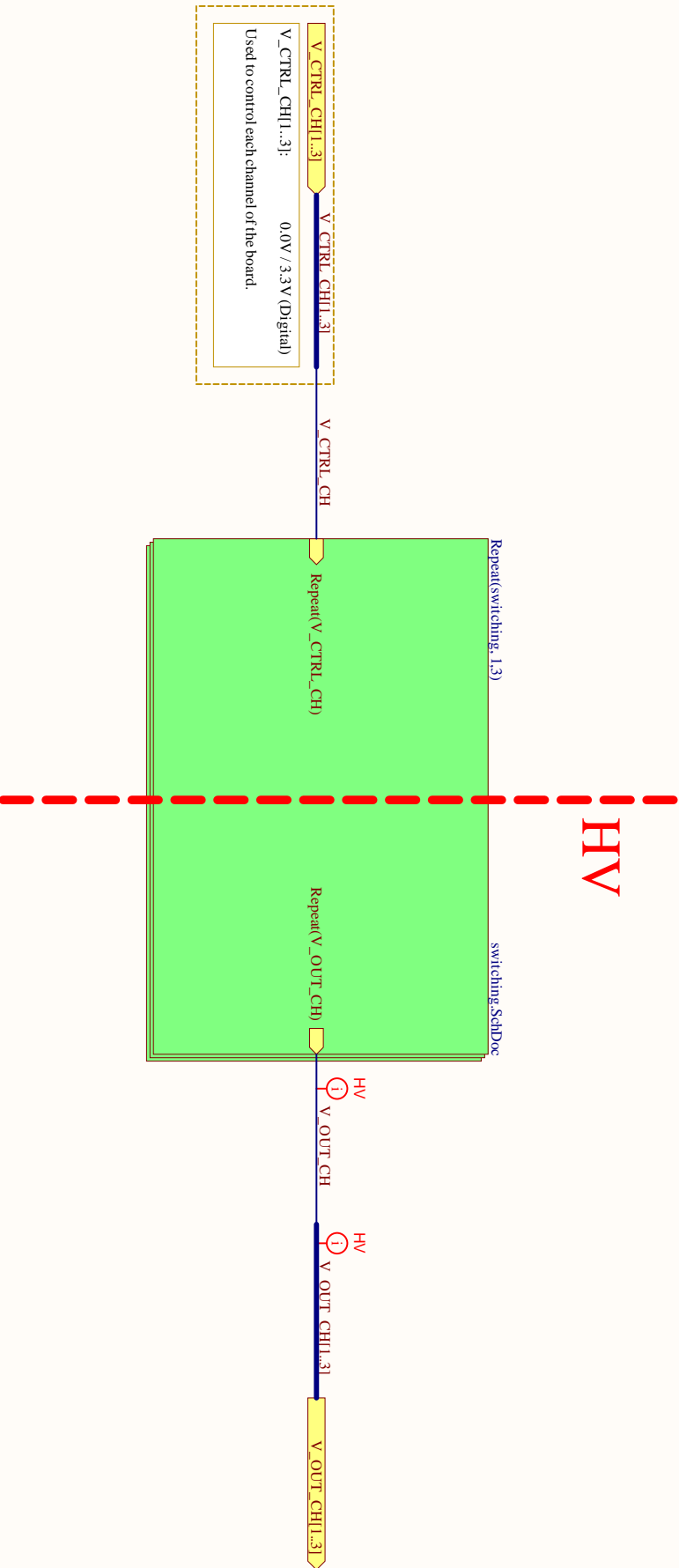
Date: 2022-07-28


Author: Albert Jansson

Based on schematic documents provided by:

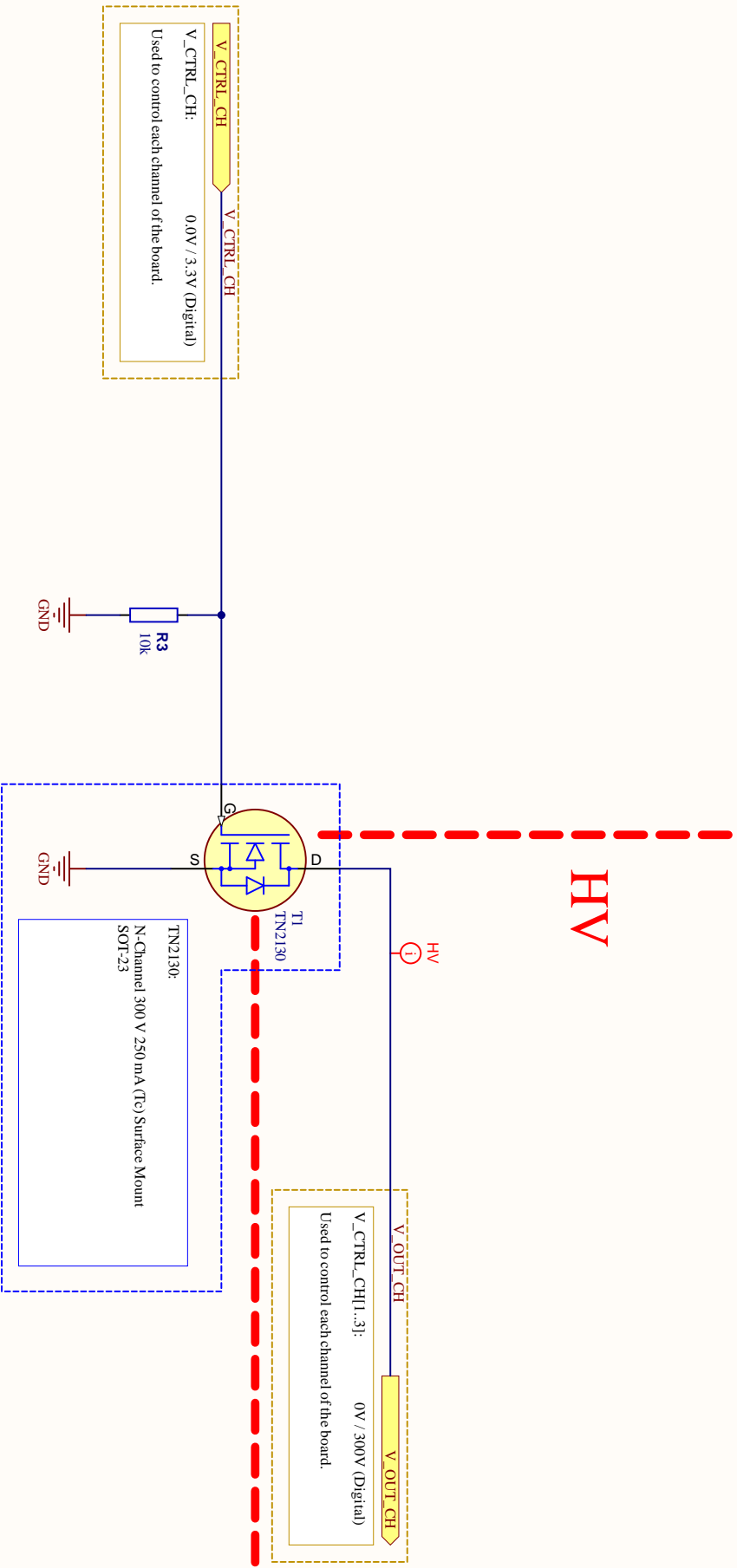
MYCRONIC

Melchi Benboudia  
EPFL STI IMT LAMTS  
Rue de la Maladière 71b  
2000 Neuchâtel  
Switzerland



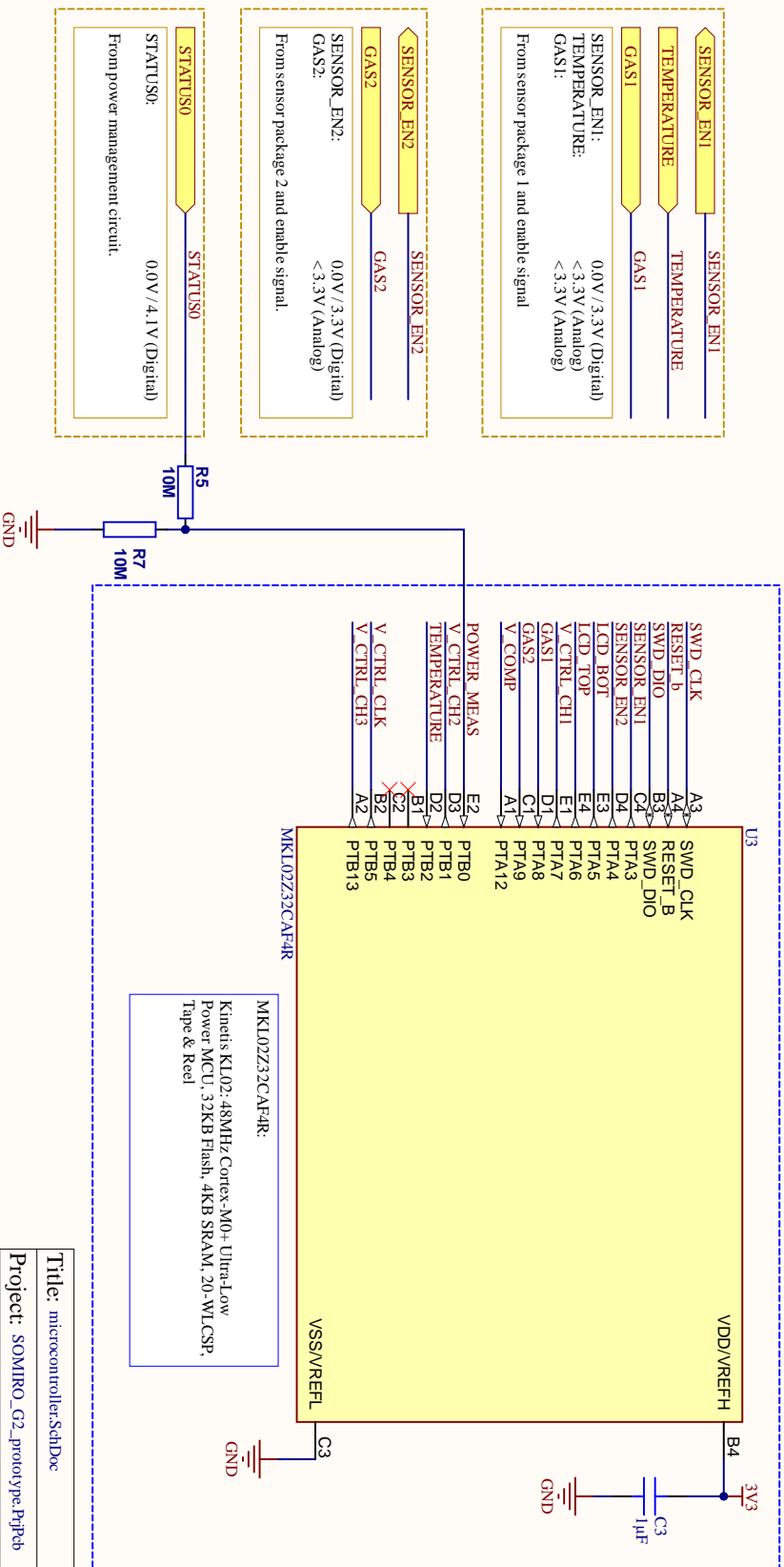
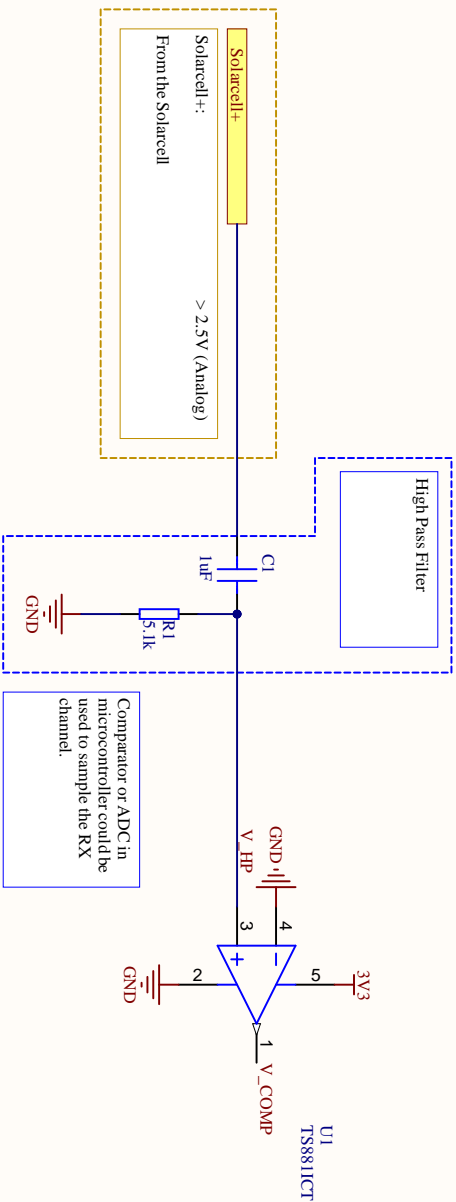
Title: switch_assembly.SchDoc		Author: <i>Albert Jansson</i>	Based on schematic documents provided by: <i>Mehdi Benbredda</i> <i>EPFL STI IMT LMTS</i> <i>Rue de la Maladière 71b</i> <i>2000 Neuchâtel</i> <i>Switzerland</i>
Project: SOMIRO_G2_prototype.PjPeb			
			
Time: 14:11:06	Date: 2022-07-28		

MYCRONIC



Title: switching_SchDoc		Author: <i>Albert Jansson</i>	Based on schematic documents provided by: <i>Mehdi Benbredda</i> EPFL STI IMT LAMS Rue de la Maladière 71b 2000 Neuchâtel Switzerland
Project: SOMIRO_G2_prototype.PjPeb			
Time: 14:11:06	Date: 2022-07-28		





LCD:  
0.0V / 3.3V (Digital)

Used to control the VLC shutter for TX channel.

LCD\_BOT  
LCD\_TOP

V\_CTRL\_CH1.3i  
0.0V / 3.3V (Digital)

Used to control each channel of the board.

V\_CTRL\_CH1.3i  
V\_CTRL\_CH1.3i

V\_CTRL\_CLK  
V\_CTRL\_CLK

0.0V / 3.3V (Digital)

From the microcontroller, used to control the LT3484-0.  
(Driving low puts the part in shutdown)

SWD\_CLK  
RESET\_b  
SWD\_DIO

SWD\_CLK  
RESET\_b  
SWD\_DIO

0.0V / 3.3V (Digital)

Physical interface used for production Flash programming

Title: microcontroller:SchDoc

Project: SOMIRO\_G2\_prototype:RjPb

Time: 14:11:06

Date: 2022-07-28

Author: Albert Jansson

Based on schematic documents provided by:

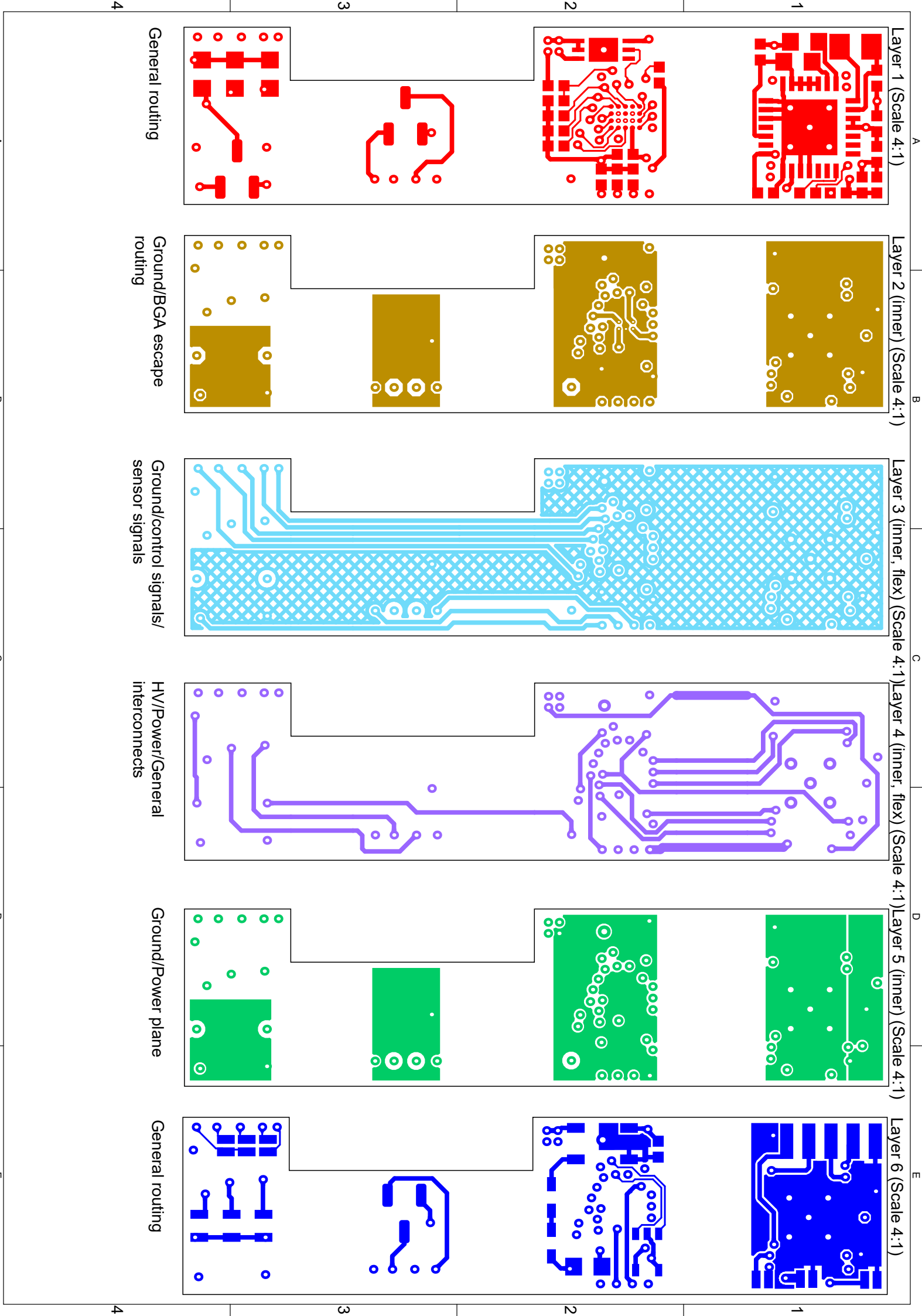
MYCRONIC

Melodi Benboudia  
EPFL STI IMT LMTS  
Rue de la Maladière 71b  
2000 Neuchâtel  
Switzerland

# **Appendix B**

## **Layout**

This appendix contains the layout produced together with some notes on the layer usages, connector locations and measurements.



Layer Stack Legend

Material	Layer	Thickness	Dielectric Material	Type	Gerber
Top Overlay					
Surface Material	Top Solder	0.02mm	WE-SM-0001	Legend	GTO
WE-ED-0008	Layer 1	0.03mm		Solder Mask	GTS
Prepreg		0.07mm	WE-PP-0034	Dielectric	GTL
WE-ED-0003	Layer 2 (inner)	0.02mm		Signal	G1
Core		0.15mm	WE-CO-0043	Dielectric	
Prepreg		0.17mm	WE-PP-0048	Dielectric	
WE-ED-0003	Layer 3 (inner, flex)	0.02mm		Signal	G2
Core		0.05mm	WE-PI-0103	Dielectric	
WE-ED-0003	Layer 4 (inner, flex)	0.02mm		Signal	G3
Prepreg		0.17mm	WE-PP-0048	Dielectric	
Core		0.15mm	WE-CO-0043	Dielectric	
WE-ED-0003	Layer 5 (inner)	0.02mm		Signal	G4
Prepreg		0.07mm	WE-PP-0034	Dielectric	
WE-ED-0008	Layer 6	0.03mm		Signal	GBL
Surface Material	Bottom Solder	0.02mm	WE-SM-0001	Solder Mask	GBS
Bottom Overlay					Legend
Total thickness: 0.99mm					GBO

Rigid sections:

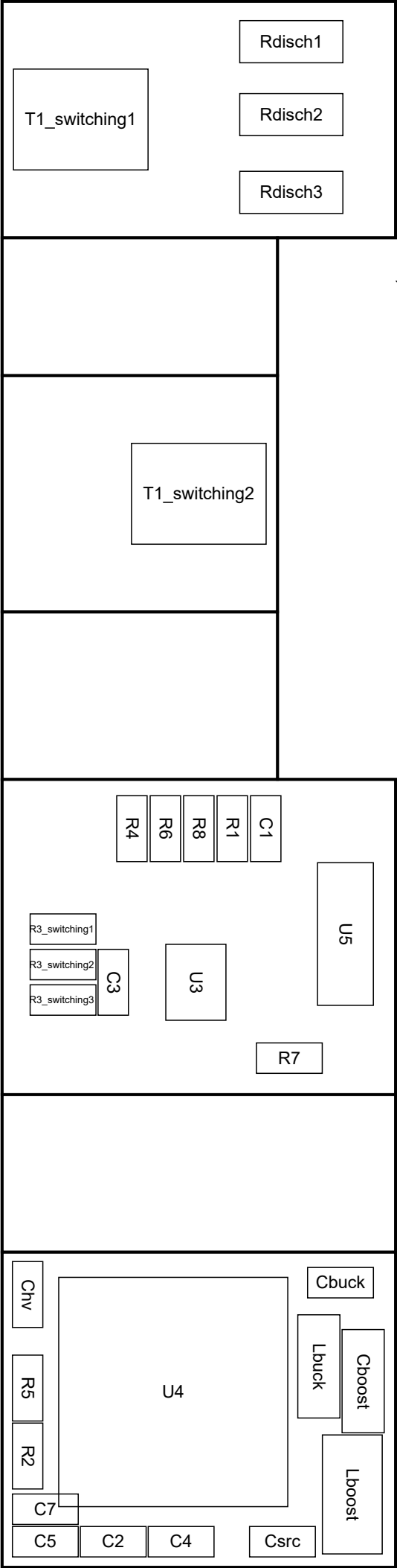
Layer Stack Legend

Material	Layer	Thickness	Dielectric Material	Type	Gerber
Surface Material	Flex Top Coverlay	0.04mm	WE-CL-0007	Solder Mask	GCT2
WE-ED-0003	Layer 3 (inner, flex)	0.02mm		Signal	G2
Core		0.05mm	WE-PI-0103	Dielectric	
WE-ED-0003	Layer 4 (inner, flex)	0.02mm		Signal	G3
Surface Material	Flex Bottom Coverlay	0.04mm	WE-CL-0007	Solder Mask	GCB2
Total thickness: 0.16mm					

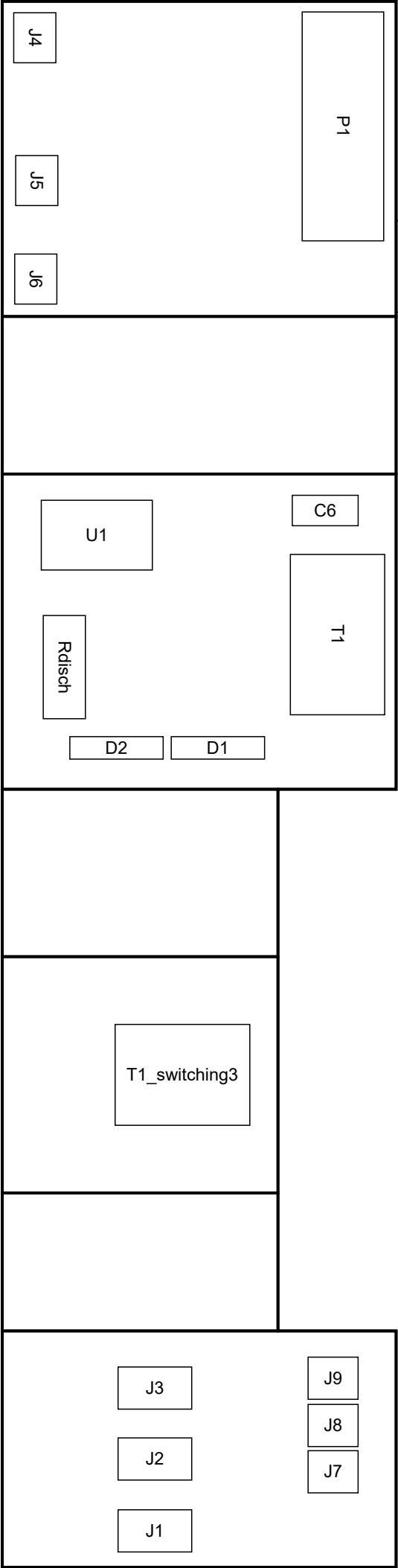
Flexible sections:

# Assembly drawing

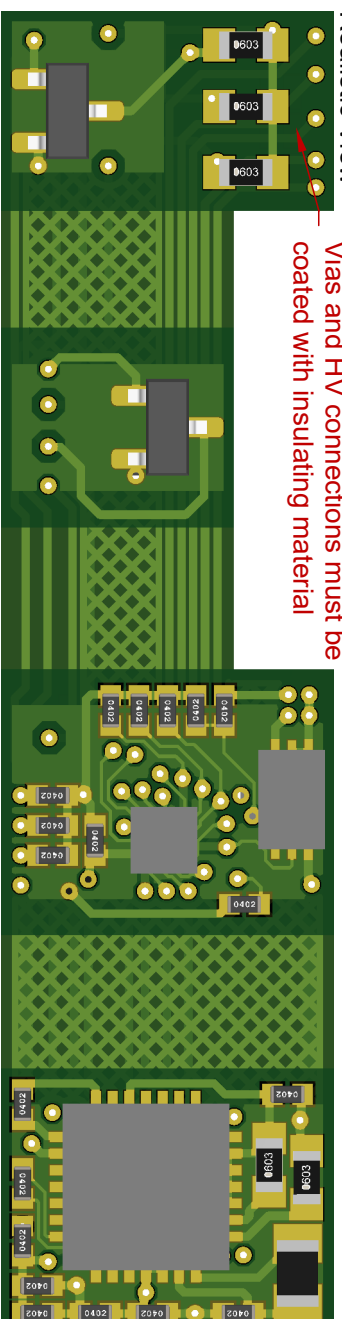
View from Top side (Scale 7:1)



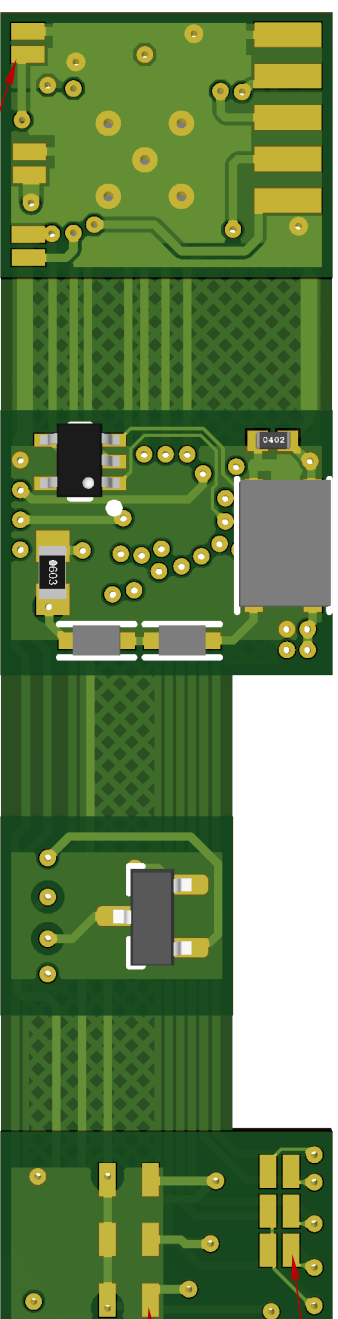
View from Bottom side (Scale 7:1)



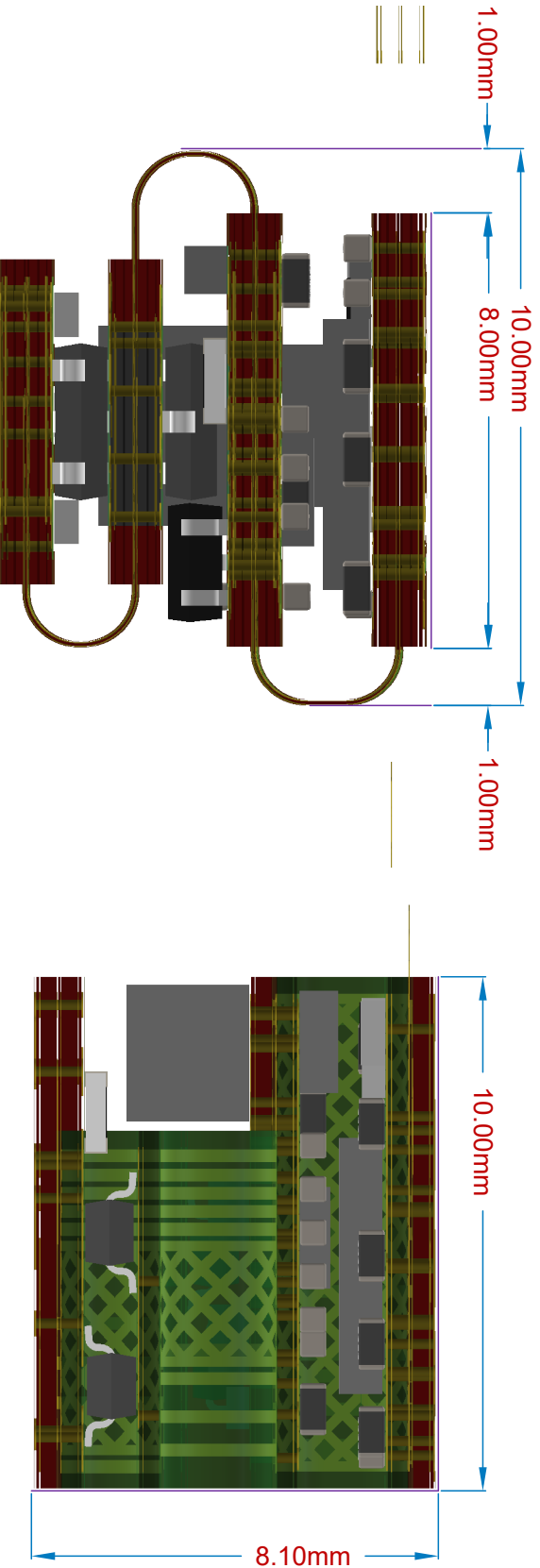
Vias and HV connections must be coated with insulating material



1. MOSFETs are SOT23 devices.
2. Spacing of rigid sections can be slightly reduced.
3. PCB design rules have been verified with the manufacturer.
4. Clearance rules are set up according to IPC-2221B 300-500V assuming internal or coated conductors.
5. Creepage distances are 0.5 mm minimum. Clears IPC-2221B 250-300V with coating.



Solar, Battery, LCD Shutter connections



# Appendix C

## Software

This appendix contains the software used for testing the PCBs. The code builds on the CMSIS driver framework available for the microcontroller and is mainly meant for testing the hardware.

Parts of the code is not fully tested and should not be expected to have the implied function without further verification.

### C.1 Main file

```

1  /*
2  *   Test code for SOMIRO prototype PCBs
3  *   Based on demo code from Martin Kojtal (0xc0170)
4  *   https://github.com/0xc0170/kinetis_klxx_gcc
5  */
6
7  // #define FRDM_KL02Z
8  // #define SOMIRO_G1
9  #define SOMIRO_G2
10
11 #ifdef FRDM_KL02Z
12 #include "KL02Z.h"
13 #endif
14
15 #ifdef SOMIRO_G1
16 #include "G1.h"
17 #endif
18
19 #ifdef SOMIRO_G2
20 #include "G2.h"
21 #endif
22
23 #include "MKL02Z4.h"
24 #include "fsl_tpm.h"
25 #include "fsl_cmp.h"
26 #include "fsl_gpio.h"
27 #include "fsl_adc16.h"
28 #include "fsl_smc.h"
29 #include "fsl_pmc.h"
30 #include "fsl_clock.h"
31
32 #define SW_LCD_CTRL
33

```

```

34 #if defined(SW_LCD_CTRL) && defined(LCD_TOP)
35     #define OUT_PIN          LCD_TOP
36     #define OUT_PIN_PT       LCD_TOP_PT
37     #define OUT_PIN_PORT     LCD_TOP_PORT
38     #define OUT_PIN2        LCD_BOT
39     #define OUT_PIN2_PT      LCD_BOT_PT
40     #define OUT_PIN2_PORT    LCD_BOT_PORT
41 #else
42     #define OUT_PIN          GPIO_LED_GREEN
43     #define OUT_PIN_PT       GPIO_LED_GREEN_PT
44     #define OUT_PIN_PORT     GPIO_LED_GREEN_PORT
45 #endif
46
47 #define COMM_BUF_LEN        50
48 #define RCV_MSG_LEN         10
49 #define SENSOR_DATA_LEN    50
50
51 #define FREQ_TOLERANCE      30
52 #define FREQ_INIT           330
53 #define FREQ_ONE            150
54 #define FREQ_ZERO          75
55
56 #include "task_functions.h"
57
58 volatile uint32_t TPM_counter = 0U;
59
60 // Timer interrupt
61 // Do something small on interrupt trigger
62 void TPM1_IRQHandler(void){
63     // Clear overflow flag
64     TPM_ClearStatusFlags(TPM1, 256);
65     ++TPM_counter;
66 }
67
68 void setup_timer1_delay(tpm_config_t* tpmInfo)
69 {
70     // Timer config
71     CLOCK_SetTpmClock(3U); // MCGIRCLK as source
72     TPM_GetDefaultConfig(tpmInfo);
73     tpmInfo->prescale = kTPM_Prescale_Divide_128;
74     TPM_Init(TPM1, tpmInfo);
75     // Sets time in cycles, 4M/128/32 = 976.5625 Hz
76     TPM_SetTimerPeriod(TPM1, 32);
77     // Overflow interrupt
78     TPM_EnableInterrupts(TPM1, 256);
79     EnableIRQ(TPM1_IRQn);
80     TPM_StartTimer(TPM1, 1);
81 }
82
83 void set_gpio_out_pin()
84 {
85     // Turn LED off
86     OUT_PIN_PT->PSOR = (1U << OUT_PIN);
87 #ifdef OUT_PIN2
88     OUT_PIN2_PT->PCOR = (1U << OUT_PIN2);
89 #endif
90 }
91
92 void clear_gpio_out_pin()
93 {
94     // Turn LED on
95     OUT_PIN_PT->PCOR = (1U << OUT_PIN);
96 #ifdef OUT_PIN2
97     OUT_PIN2_PT->PSOR = (1U << OUT_PIN2);
98 #endif

```

```

99  }
100
101  int main(void)
102  {
103      SMC_SetPowerModeProtection(
104          SMC, kSMC_AllowPowerModeAll);
105      SystemInit();
106      CLKCFG_Boot();
107
108      tpm_config_t tpmInfo;
109      setup_timer1_delay(&tpmInfo);
110
111      // Enable clock for PORTs
112      CLOCK_EnableClock(kCLOCK_PortA);
113      CLOCK_EnableClock(kCLOCK_PortB);
114
115      gpio_pin_config_t gpioDigitalOutput =
116      {
117          kGPIO_DigitalOutput,
118          0,
119      };
120
121      // Set port mux to Alt1
122      OUT_PIN_PORT->PCR[OUT_PIN] = PORT_PCR_MUX(1U);
123      GPIO_PinInit(OUT_PIN_PT,
124                  OUT_PIN,
125                  &gpioDigitalOutput);
126  #ifdef OUT_PIN2
127      OUT_PIN2_PORT->PCR[OUT_PIN2] = PORT_PCR_MUX(1U);
128      GPIO_PinInit(OUT_PIN2_PT,
129                  OUT_PIN2,
130                  &gpioDigitalOutput);
131  #endif
132
133  #ifdef LCD_TOP
134  #ifndef SW_LCD_CTRL
135      LCD_TOP_PORT->PCR[LCD_TOP] = PORT_PCR_MUX(1U);
136      LCD_BOT_PORT->PCR[LCD_BOT] = PORT_PCR_MUX(1U);
137      GPIO_PinInit(LCD_BOT_PT,
138                  LCD_BOT,
139                  &gpioDigitalOutput);
140      GPIO_PinInit(LCD_TOP_PT,
141                  LCD_TOP,
142                  &gpioDigitalOutput);
143  #endif
144  #endif
145
146      actuate_state_t actuate_state = {
147          .iterations = 0,
148          .state = 0,
149      };
150      setup_actuator_pins(&gpioDigitalOutput);
151      setup_read_sensors(&gpioDigitalOutput);
152
153      comm_state_t comm_state = {
154          .sending = 0,
155          .bit = 8,
156          .byte = 0,
157          .length = 0,
158      };
159  #ifdef SW_LCD_CTRL
160      comm_device_t comm_devices = {
161          .enable = &set_gpio_out_pin,
162          .disable = &clear_gpio_out_pin,
163      };

```

```

164 #else
165     comm_device_t comm_devices = {
166         .enable = &comm_lcd_activate ,
167         .disable = &comm_lcd_deactivate ,
168     };
169 #endif
170     uint8_t comm_buffer[COMM_BUF_LEN];
171     uint16_t test_data = 0x0A0D;
172     uint32_t test_data32 = 0x0A0F0A0A;
173
174     comm_rcv_state_t comm_rcv_state = comm_rcv_idle;
175     uint8_t comm_rcv_data = -1;
176
177     uint16_t batteryVoltage = 0;
178
179     sensor_state_t sensor_state = {
180         .index = 0,
181     };
182     uint32_t sensor_data[SENSOR_DATA_LEN];
183
184     // Task timing variables [~ms]
185     uint32_t time_send_communication = 0;
186     uint32_t period_send_communication = 100;
187     uint32_t time_test_data = 0;
188     uint32_t period_test_data = 5000;
189     uint32_t time_read_battery_voltage = 0;
190     uint32_t period_read_battery_voltage = 1000;
191     uint32_t time_read_sensors = 0;
192     uint32_t period_read_sensors = 1000;
193     uint32_t time_receive_comm = 0;
194     uint32_t period_receive_comm = 500;
195     uint32_t time_activate_actuators = 0;
196     uint32_t period_activate_actuators = 100;
197
198     // Turn off active low LED connected to OUT_PIN
199     set_gpio_out_pin();
200
201     CLKCFG_VLPR();
202     SMC_SetPowerModeVlpr(SMC);
203
204     /* All features have to be set up by this point */
205     while (1)
206     {
207         /* The mainloop runs all tasks once, then enters
208         VLPW until TPM sends an interrupt to wake the
209         MCU. Each task needs to keep time and state ,
210         helped by TPM_counter that has a given update
211         period */
212         if ((TPM_counter - time_send_communication)
213             > period_send_communication)
214         {
215             time_send_communication = TPM_counter;
216             send_communication(&comm_state ,
217                             comm_buffer ,
218                             &comm_devices);
219         }
220
221         receive_communication(&comm_rcv_state ,
222                             &comm_rcv_data);
223
224         if (batteryVoltage > 100)
225         {
226             actuate_pattern(&actuate_state);
227         }
228     }

```

```

229     if ((TPM_counter - time_read_battery_voltage)
230         > period_read_battery_voltage)
231     {
232         time_read_battery_voltage = TPM_counter;
233         batteryVoltage = read_battery_voltage();
234     }
235
236     if ((TPM_counter - time_read_sensors)
237         > period_read_sensors)
238     {
239         time_read_sensors = TPM_counter;
240         read_sensors(&sensor_state, sensor_data);
241     }
242
243     // Trigger tasks
244     if ((TPM_counter - time_test_data)
245         > period_test_data)
246     {
247         time_test_data = TPM_counter;
248         comm_buffer[0] = 0xFE; // Start bit
249
250         // Send received data
251         // comm_buffer[1] = comm_rcv_data;
252
253         // Send test data
254         write_16to8(&test_data, &comm_buffer[1]);
255         // write_32to8(&test_data32, &comm_buffer[1]);
256         force_send_comm(&comm_state, 2);
257
258         // Send sensor data
259         // if (sensor_state.index > 0)
260         // {
261         //     write_32to8(
262         //         &sensor_data[sensor_state.index-1],
263         //         &comm_buffer[1]
264         //     );
265         //     force_send_comm(&comm_state, 6);
266         // }
267     }
268
269     if ((TPM_counter - time_receive_comm)
270         > period_receive_comm)
271     {
272         comm_rcv_state = comm_rcv_init;
273     }
274
275     if ((TPM_counter - time_activate_actuators)
276         > period_activate_actuators)
277     {
278         time_activate_actuators = TPM_counter;
279         actuate_state.iterations = 10;
280     }
281     SMC_SetPowerModeVlpw(SMC);
282     // VLLS for testing current draw of external
283     // components, requires reset to wake
284     // smc_power_mode_vlls_config_t vllsCfg =
285     // {
286     //     .subMode = 0U,
287     //     .enablePorDetectInVlls0 = false,
288     // };
289     // SMC_SetPowerModeVlls(SMC, &vllsCfg);
290 }
291 }

```

## C.2 Tasks

```

1  #ifndef TASK_FUNCTIONS_H
2  #define TASK_FUNCTIONS_H
3
4  #include "MKL02Z4.h"
5  #include "fsl_tpm.h"
6  #include "fsl_cmp.h"
7  #include "fsl_gpio.h"
8  #include "fsl_adc16.h"
9  #include "fsl_smc.h"
10 #include "fsl_pmc.h"
11
12 typedef struct
13 {
14     uint16_t index;
15 } sensor_state_t;
16
17 void setup_read_sensors(gpio_pin_config_t* pinCfg)
18 {
19     #ifdef BOARD_HAS_SENSORS
20         SENSOR_EN1_PORT->PCR[SENSOR_EN1] = PORT_PCR_MUX(1U);
21         SENSOR_EN2_PORT->PCR[SENSOR_EN2] = PORT_PCR_MUX(1U);
22         GPIO_PinInit(SENSOR_EN1_PT, SENSOR_EN1, pinCfg);
23         GPIO_PinInit(SENSOR_EN2_PT, SENSOR_EN2, pinCfg);
24     #endif
25 }
26
27 void read_sensors(sensor_state_t* state,
28                  uint32_t* sensor_data)
29 {
30     #ifdef BOARD_HAS_SENSORS
31         #define SENSOR_AVG_SAMPLES      5U
32         #define SENSOR_VOLTAGE_OFFSET    0U
33         #define SENSOR_MASK              0x03FF
34         #define SENSOR_DATA_SHIFT0       0
35         #define SENSOR_DATA_SHIFT1       10
36         #define SENSOR_DATA_SHIFT2       20
37         uint32_t value;
38         // Enable ADC
39         adc16_config_t adc16Config;
40         adc16_channel_config_t adc16ChannelConfig;
41         ADC16_GetDefaultConfig(&adc16Config);
42         ADC16_Init(ADC0, &adc16Config);
43         // Use software trigger
44         ADC16_EnableHardwareTrigger(ADC0, false);
45         adc16ChannelConfig.enableInterruptOnConversionCompleted = false;
46
47         // Enable sensors
48         GPIO_PinWrite(SENSOR_EN1_PT, SENSOR_EN1, 1);
49         GPIO_PinWrite(SENSOR_EN2_PT, SENSOR_EN2, 1);
50
51         // Wait for sensors to stabilize?
52
53         // Read first sensor
54         adc16ChannelConfig.channelNumber = GAS1_ADC_CH;
55         value = 0;
56         for (uint8_t i = 0; i < SENSOR_AVG_SAMPLES; ++i)
57         {
58             ADC16_SetChannelConfig(ADC0,

```

```

62             0U,
63             &adc16ChannelConfig);
64     while (0U == (kADC16_ChannelConversionDoneFlag
65                 & ADC16_GetChannelStatusFlags(ADC0, 0U)))
66     {
67     }
68     value += ADC16_GetChannelConversionValue(ADC0, 0U);
69 }
70 value /= SENSOR_AVG_SAMPLES;
71 sensor_data[state->index] = (value & SENSOR_MASK)
72                             << SENSOR_DATA_SHIFT0;
73
74 // Read second sensor
75 adc16ChannelConfig.channelNumber = GAS2_ADC_CH;
76 value = 0;
77 for (uint8_t i = 0; i < SENSOR_AVG_SAMPLES; ++i)
78 {
79     ADC16_SetChannelConfig(ADC0,
80                             0U,
81                             &adc16ChannelConfig);
82     while (0U == (kADC16_ChannelConversionDoneFlag
83                 & ADC16_GetChannelStatusFlags(ADC0, 0U)))
84     {
85     }
86     value += ADC16_GetChannelConversionValue(ADC0, 0U);
87 }
88 value /= SENSOR_AVG_SAMPLES;
89 sensor_data[state->index] |= (value & SENSOR_MASK)
90                             << SENSOR_DATA_SHIFT1;
91
92 // Read third sensor
93 adc16ChannelConfig.channelNumber = TEMPERATURE_ADC_CH;
94 value = 0;
95 for (uint8_t i = 0; i < SENSOR_AVG_SAMPLES; ++i)
96 {
97     ADC16_SetChannelConfig(ADC0,
98                             0U,
99                             &adc16ChannelConfig);
100    while (0U == (kADC16_ChannelConversionDoneFlag
101                & ADC16_GetChannelStatusFlags(ADC0, 0U)))
102    {
103    }
104    value += ADC16_GetChannelConversionValue(ADC0, 0U);
105 }
106 value /= SENSOR_AVG_SAMPLES;
107 sensor_data[state->index] |= (value & SENSOR_MASK)
108                             << SENSOR_DATA_SHIFT2;
109
110 // Increment array position (wrap-around)
111 ++(state->index);
112 if (state->index == SENSOR_DATA_LEN)
113 {
114     state->index = 0;
115 }
116
117 // Disable ADC and sensors
118 ADC16_Deinit(ADC0);
119 GPIO_PinWrite(PTA, SENSOR_EN1, 0);
120 GPIO_PinWrite(PTA, SENSOR_EN2, 0);
121
122 #endif
123 }
124
125 volatile uint32_t comm_rcvd_edges;
126 volatile uint32_t comm_rcv_timer;

```

```

127  volatile bool comm_msg_received;
128
129  void PORTA_IRQHandler(void)
130  {
131      // Clear Interrupt Flag
132      GPIO_PortClearInterruptFlags(PTA, 1U << V_COMP_OUT);
133
134      if ( comm_rcvd_edges == 0)
135      {
136          tpm_config_t tpmInfo;
137          TPM_GetDefaultConfig(&tpmInfo);
138          tpmInfo.prescale = kTPM_Prescale_Divide_64;
139          tpmInfo.enableStopOnOverflow = true;
140          TPM_Init(TPM0, &tpmInfo);
141          TPM_StartTimer(TPM0, kTPM_SystemClock);
142          ++comm_rcvd_edges;
143      }
144      else if ((comm_rcvd_edges > 0)
145              && (comm_rcvd_edges < RCV_MSG_LEN))
146      {
147          ++comm_rcvd_edges;
148      }
149      else
150      {
151          TPM_StopTimer(TPM0);
152          comm_rcv_timer = TPM_GetCurrentTimerCount(TPM0);
153          comm_rcvd_edges = 0;
154          comm_msg_received = true;
155      }
156  }
157
158
159  typedef enum _comm_rcv_state
160  {
161      comm_rcv_init ,
162      comm_rcv_start ,
163      comm_rcv_loop ,
164      comm_rcv_end ,
165      comm_rcv_idle ,
166  } comm_rcv_state_t;
167
168
169  void receive_communication(comm_rcv_state_t* state ,
170                             uint8_t* data)
171  {
172      switch (*state)
173      {
174          case comm_rcv_init:
175              comm_rcvd_edges = 0;
176              comm_rcv_timer = 0;
177              comm_msg_received = false;
178              EnableIRQ(PORTA_IRQn);
179              ++(*state);
180              break;
181
182          case comm_rcv_start:
183              // One TPM wait period later ,
184              // a message should be recieved
185              if (comm_msg_received)
186              {
187                  // Check for a resonable frequency
188                  uint32_t frequency = comm_rcv_timer/RCV_MSG_LEN;
189                  if ((FREQ_INIT-FREQ_TOLERANCE < frequency)
190                      && (frequency < FREQ_INIT+FREQ_TOLERANCE))
191                  {

```

```

192             // Frequency "init" received
193             *data = 0xA;
194         }
195         else if ((FREQ_ONE-FREQ_TOLERANCE < frequency)
196             && (frequency < FREQ_ONE+FREQ_TOLERANCE))
197         {
198             // Frequency "one" received
199             *data = 1;
200         }
201         else if ((FREQ_ZERO-FREQ_TOLERANCE < frequency)
202             && (frequency < FREQ_ZERO+FREQ_TOLERANCE))
203         {
204             // Frequency "zero" received
205             *data = 0;
206         }
207         else
208         {
209             // Some other frequency received
210             *data = -1;
211         }
212     }
213     *state = comm_rcv_end;
214
215     case comm_rcv_end:
216         DisableIRQ(PORTA_IRQn);
217         ++(*state);
218
219     default:
220         break;
221 }
222
223 }
224
225 typedef struct
226 {
227     uint8_t sending;
228     uint8_t bit;
229     uint8_t byte;
230     uint8_t length;
231 } comm_state_t;
232
233
234 typedef struct
235 {
236     void (*enable)();
237     void (*disable)();
238 } comm_device_t;
239
240
241 void send_communication(comm_state_t* state,
242     uint8_t* buf,
243     comm_device_t* dev)
244 {
245     if (state->sending)
246     {
247         // Send one bit per invocation
248         uint8_t bit = ((buf[state->byte] >> --(state->bit))
249             & 1U);
250         if (bit)
251         {
252             // Idle high
253             dev->disable();
254         }
255         else
256         {

```

```

257         // Active low
258         dev->enable();
259     }
260     if (state->bit == 0)
261     {
262         ++(state->byte);
263         state->bit = 8;
264     }
265     if (state->byte == state->length)
266     {
267         state->sending = 0;
268         // Clear buffer
269         for (uint8_t i = 0; i < state->length; ++i)
270         {
271             buf[i] = 0;
272         }
273     }
274 }
275 else
276 {
277     // Disable sending device
278     dev->disable();
279 }
280 }
281
282 void force_send_comm(comm_state_t* state, uint8_t length)
283 {
284     state->sending = 1;
285     state->bit = 8;
286     state->byte = 0;
287     state->length = length;
288 }
289
290 uint16_t read_battery_voltage()
291 {
292     #define AVG_SAMPLES 5U
293     #define VOLTAGE_OFFSET 0U
294     uint32_t value = 0U;
295     // Enable ADC
296     adc16_config_t adc16Config;
297     adc16_channel_config_t adc16ChannelConfig;
298     ADC16_GetDefaultConfig(&adc16Config);
299     ADC16_Init(ADC0, &adc16Config);
300     // Use software trigger
301     ADC16_EnableHardwareTrigger(ADC0, false);
302
303     adc16ChannelConfig.channelNumber = PWR_MEAS_ADC_CH;
304     adc16ChannelConfig.
305         enableInterruptOnConversionCompleted = false;
306
307     // Read and average samples
308     for (uint8_t i = 0; i < AVG_SAMPLES; ++i)
309     {
310         ADC16_SetChannelConfig(ADC0,
311                                0U,
312                                &adc16ChannelConfig);
313         while (0U == (kADC16_ChannelConversionDoneFlag
314                      & ADC16_GetChannelStatusFlags(ADC0, 0U)))
315         {
316         }
317         value += ADC16_GetChannelConversionValue(ADC0, 0U);
318     }
319     value /= AVG_SAMPLES;
320
321     // Apply calibration

```

```

322     value += VOLTAGE_OFFSET;
323
324     // Disable ADC
325     ADC16_Deinit(ADC0);
326
327     // Return corrected value
328     return value;
329 }
330
331 void write_16to8(uint16_t* in, uint8_t* out)
332 {
333     /* Uses little-endian byte order */
334     out[0] = ((*in) >> 8) & 0xFF;
335     out[1] = (*in) & 0xFF;
336 }
337
338 void write_32to8(uint32_t* in, uint8_t* out)
339 {
340     /* Uses little-endian byte order */
341     out[0] = ((*in) >> 24) & 0xFF;
342     out[1] = ((*in) >> 16) & 0xFF;
343     out[2] = ((*in) >> 8) & 0xFF;
344     out[3] = (*in) & 0xFF;
345 }
346
347 void setup_actuator_pins(gpio_pin_config_t* pinCfg)
348 {
349     V_CTRL_CLK_PORT->PCR[V_CTRL_CLK] = PORT_PCR_MUX(1U);
350     V_CTRL_CH1_PORT->PCR[V_CTRL_CH1] = PORT_PCR_MUX(1U);
351     V_CTRL_CH2_PORT->PCR[V_CTRL_CH2] = PORT_PCR_MUX(1U);
352     V_CTRL_CH3_PORT->PCR[V_CTRL_CH3] = PORT_PCR_MUX(1U);
353     GPIO_PinInit(V_CTRL_CLK_PT, V_CTRL_CLK, pinCfg);
354     GPIO_PinInit(V_CTRL_CH1_PT, V_CTRL_CH1, pinCfg);
355     GPIO_PinInit(V_CTRL_CH2_PT, V_CTRL_CH2, pinCfg);
356     GPIO_PinInit(V_CTRL_CH3_PT, V_CTRL_CH3, pinCfg);
357     V_CTRL_CLK_PT->PCOR = (1U << V_CTRL_CLK);
358     V_CTRL_CH1_PT->PCOR = (1U << V_CTRL_CH1);
359     V_CTRL_CH2_PT->PCOR = (1U << V_CTRL_CH2);
360     V_CTRL_CH3_PT->PCOR = (1U << V_CTRL_CH3);
361 }
362
363 typedef struct
364 {
365     uint8_t state;
366     uint16_t iterations;
367 } actuate_state_t;
368
369 void actuate_pattern(actuate_state_t* state)
370 {
371     if (state->iterations)
372     {
373         switch (state->state)
374         {
375             case 0:
376                 /* Initialize */
377                 ++(state->state);
378                 break;
379
380             case 1:
381                 GPIO_PinWrite(V_CTRL_CLK_PT, V_CTRL_CLK, 1);
382                 GPIO_PinWrite(V_CTRL_CH1_PT, V_CTRL_CH1, 1);
383                 GPIO_PinWrite(V_CTRL_CH2_PT, V_CTRL_CH2, 0);
384                 GPIO_PinWrite(V_CTRL_CH3_PT, V_CTRL_CH3, 0);
385                 ++(state->state);
386

```

```

387         break;
388
389     case 2:
390         GPIO_PinWrite(V_CTRL_CLK_PT, V_CTRL_CLK, 0);
391         GPIO_PinWrite(V_CTRL_CH1_PT, V_CTRL_CH1, 1);
392         GPIO_PinWrite(V_CTRL_CH2_PT, V_CTRL_CH2, 0);
393         GPIO_PinWrite(V_CTRL_CH3_PT, V_CTRL_CH3, 0);
394         ++(state->state);
395         break;
396
397     case 3:
398         GPIO_PinWrite(V_CTRL_CLK_PT, V_CTRL_CLK, 1);
399         GPIO_PinWrite(V_CTRL_CH1_PT, V_CTRL_CH1, 0);
400         GPIO_PinWrite(V_CTRL_CH2_PT, V_CTRL_CH2, 1);
401         GPIO_PinWrite(V_CTRL_CH3_PT, V_CTRL_CH3, 0);
402         ++(state->state);
403         break;
404
405     case 4:
406         GPIO_PinWrite(V_CTRL_CLK_PT, V_CTRL_CLK, 0);
407         GPIO_PinWrite(V_CTRL_CH1_PT, V_CTRL_CH1, 0);
408         GPIO_PinWrite(V_CTRL_CH2_PT, V_CTRL_CH2, 1);
409         GPIO_PinWrite(V_CTRL_CH3_PT, V_CTRL_CH3, 0);
410         ++(state->state);
411         break;
412
413     case 5:
414         GPIO_PinWrite(V_CTRL_CLK_PT, V_CTRL_CLK, 1);
415         GPIO_PinWrite(V_CTRL_CH1_PT, V_CTRL_CH1, 0);
416         GPIO_PinWrite(V_CTRL_CH2_PT, V_CTRL_CH2, 0);
417         GPIO_PinWrite(V_CTRL_CH3_PT, V_CTRL_CH3, 1);
418         ++(state->state);
419         break;
420
421     case 6:
422         GPIO_PinWrite(V_CTRL_CLK_PT, V_CTRL_CLK, 0);
423         GPIO_PinWrite(V_CTRL_CH1_PT, V_CTRL_CH1, 0);
424         GPIO_PinWrite(V_CTRL_CH2_PT, V_CTRL_CH2, 0);
425         GPIO_PinWrite(V_CTRL_CH3_PT, V_CTRL_CH3, 1);
426         ++(state->state);
427         break;
428
429     default:
430         GPIO_PinWrite(V_CTRL_CLK_PT, V_CTRL_CLK, 0);
431         GPIO_PinWrite(V_CTRL_CH1_PT, V_CTRL_CH1, 0);
432         GPIO_PinWrite(V_CTRL_CH2_PT, V_CTRL_CH2, 0);
433         GPIO_PinWrite(V_CTRL_CH3_PT, V_CTRL_CH3, 0);
434         --(state->iterations);
435         state->state = 0;
436         break;
437     }
438 }
439 }
440
441 void comm_lcd_activate()
442 {
443     TPM_Deinit(TPM0);
444
445     tpm_config_t tpmInfo;
446     tpm_chnl_pwm_signal_param_t tpmParam[2];
447
448     tpmParam[0].chnlNumber = (tpm_chnl_t)0U;
449     //0U no output, 1U LowTrue, 2U HighTrue
450     tpmParam[0].level = 1U;
451     tpmParam[0].dutyCyclePercent = 50U;

```

```

452
453     tpmParam[1].chnlNumber = (tpm_chnl_t)1U;
454     //0U no output, 1U LowTrue, 2U HighTrue
455     tpmParam[1].level = 2U;
456     tpmParam[1].dutyCyclePercent = 50U;
457
458     CLOCK_SetTpmClock(3U); // MCGIRCLK as source
459     TPM_GetDefaultConfig(&tpmInfo);
460     tpmInfo.prescale = kTPM_Prescale_Divide_128;
461
462     TPM_Init(TPM0, &tpmInfo);
463     TPM_SetupPwm(TPM0,
464                  tpmParam,
465                  2U,
466                  kTPM_EdgeAlignedPwm,
467                  4000U,
468                  32768U);
469     TPM_StartTimer(TPM0, kTPM_SystemClock);
470     TPM_UpdatePwmDutycycle(TPM0,
471                             (tpm_chnl_t)0U,
472                             kTPM_EdgeAlignedPwm,
473                             50U);
474     TPM_UpdatePwmDutycycle(TPM0,
475                             (tpm_chnl_t)1U,
476                             kTPM_EdgeAlignedPwm,
477                             50U);
478 }
479
480 void comm_lcd_deactivate()
481 {
482     TPM_UpdatePwmDutycycle(TPM0,
483                             (tpm_chnl_t)0U,
484                             kTPM_EdgeAlignedPwm,
485                             0U);
486     TPM_UpdatePwmDutycycle(TPM0,
487                             (tpm_chnl_t)1U,
488                             kTPM_EdgeAlignedPwm,
489                             0U);
490     TPM_Deinit(TPM0);
491 }
492
493 #endif

```

## C.3 G1 header file

```

1  #ifndef G1_H
2  #define G1_H
3
4  #define CPU_MKL02Z32CAF4
5
6  #include "fsl_clock.h"
7
8  /* Port A */
9  #define PTA3          3U
10 #define PTA4          4U
11 #define PTA5          5U
12 #define PTA6          6U
13 #define PTA7          7U
14 #define PTA8          8U
15 #define PTA9          9U
16 #define PTA12         12U
17
18 #define V_CTRL_CH1     PTA5
19 #define V_CTRL_CH1_PT  PTA
20 #define V_CTRL_CH1_PORT PORTA
21 #define V_CTRL_CH2     PTA4
22 #define V_CTRL_CH2_PT  PTA
23 #define V_CTRL_CH2_PORT PORTA
24 #define V_CTRL_CH3     PTA3
25 #define V_CTRL_CH3_PT  PTA
26 #define V_CTRL_CH3_PORT PORTA
27 #define V_CTRL_CLK     PTA6
28 #define V_CTRL_CLK_PT  PTA
29 #define V_CTRL_CLK_PORT PORTA
30 #define V_COMP_OUT     PTA12
31
32 /* Port B */
33 #define PTB0          0U
34 #define PTB1          1U
35 #define PTB2          2U
36 #define PTB3          3U
37 #define PTB4          4U
38 #define PTB5          5U
39 #define PTB13         13U
40
41 #define PWR_MEAS       PTB1
42 #define PWR_MEAS_ADC_CH 5U
43 #define IR_IN          PTB3
44
45 #define GPIO_LED_BLUE  PTB3
46 #define GPIO_LED_BLUE_PT PTB
47 #define GPIO_LED_BLUE_PORT PORTB
48 #define GPIO_LED_GREEN PTB3
49 #define GPIO_LED_GREEN_PT PTB
50 #define GPIO_LED_GREEN_PORT PORTB
51 #define GPIO_LED_RED   PTB3
52 #define GPIO_LED_RED_PT PTB
53 #define GPIO_LED_RED_PORT PORTB
54
55 /* Port features */
56 #define TPM0_CH0       V_CTRL_CLK
57 #define TPM0_CH1       V_CTRL_CH1
58
59 #define TPM1_CH0       V_COMP_OUT
60 #define TPM1_CH1       SWD_DIO
61

```

```

62 #define BOARD_BOOTCLOCKRUN_CORE_CLOCK 47972352U
63 #define BOARD_BOOTCLOCKVLPR_CORE_CLOCK 4000000U
64 /*!< Oscillator 0pF capacitor load */
65 #define OSC_CAP0P 0U
66 /*!< Disable external reference clock */
67 #define OSC_ER_CLK_DISABLE 0U
68
69 extern uint32_t SystemCoreClock;
70
71 const mcg_config_t mcgConfig_BOARD_BootClockRUN =
72 {
73     /* FBI – FLL Bypassed Internal */
74     .mcgMode = kMCG_ModeFBI,
75     /* MCGIRCLK enabled, MCGIRCLK disabled in STOP mode */
76     .irclkEnableMode = kMCG_IrclkEnable,
77     /* Fast internal reference clock selected */
78     .ircs = kMCG_IrcFast,
79     /* Fast IRC divider: divided by 1 */
80     .fcrdiv = 0x0U,
81     /* FLL reference clock divider: divided by 1 */
82     .frdiv = 0x0U,
83     /* Mid frequency range */
84     .drs = kMCG_DrsMid,
85     /* DCO is fine-tuned for maximum frequency with
86        32.768 kHz reference */
87     .dmx32 = kMCG_Dmx32Fine,
88 };
89
90 const sim_clock_config_t simConfig_BOARD_BootClockRUN =
91 {
92     /* SIM_CLKDIV1 – OUTDIV1: /1, OUTDIV4: /2 */
93     .clkdiv1 = 0x10000U,
94 };
95
96 const osc_config_t oscConfig_BOARD_BootClockRUN =
97 {
98     /* Oscillator frequency: 0 Hz */
99     .freq = 0U,
100    /* Oscillator capacity load: 0pF */
101    .capLoad = (OSC_CAP0P),
102    /* Oscillator low power */
103    .workMode = kOSC_ModeOscLowPower,
104    .oscerConfig =
105    {
106        /* Disable external reference clock, disable
107           external reference clock in STOP mode */
108        .enableMode = OSC_ER_CLK_DISABLE,
109    }
110 };
111
112 const mcg_config_t mcgConfig_BOARD_BootClockVLPR =
113 {
114     /* BLPI – Bypassed Low Power Internal */
115     .mcgMode = kMCG_ModeBLPI,
116     /* MCGIRCLK enabled, MCGIRCLK disabled in STOP mode */
117     .irclkEnableMode = kMCG_IrclkEnable,
118     /* Fast internal reference clock selected */
119     .ircs = kMCG_IrcFast,
120     /* Fast IRC divider: divided by 1 */
121     .fcrdiv = 0x0U,
122     /* FLL reference clock divider: divided by 1 */
123     .frdiv = 0x0U,
124     /* Low frequency range */
125     .drs = kMCG_DrsLow,
126     /* DCO has a default range of 25% */

```

```

127     .dmx32 = kMCG_Dmx32Default,
128 };
129
130 const sim_clock_config_t simConfig_BOARD_BootClockVLPR =
131 {
132     /* SIM_CLKDIV1 – OUTDIV1: /1, OUTDIV4: /5 */
133     .clkdiv1 = 0x40000U,
134 };
135
136 const osc_config_t oscConfig_BOARD_BootClockVLPR =
137 {
138     /* Oscillator frequency: 0Hz */
139     .freq = 0U,
140     /* Oscillator capacity load: 0pF */
141     .capLoad = (OSC_CAP0P),
142     /* Use internal clock */
143     .workMode = kOSC_ModeOscLowPower,
144     .oscerConfig =
145     {
146         /* Disable external reference clock */
147         .enableMode = OSC_ER_CLK_DISABLE,
148     }
149 };
150
151 static void CLOCK_CONFIG_FllStableDelay(void)
152 {
153     uint32_t i = 30000U;
154     while (i--)
155     {
156         __asm__ ("nop");
157     }
158 }
159
160 void CLKCFG_Boot(void)
161 {
162     CLOCK_SetSimSafeDivs();
163     CLOCK_InitOsc0(&oscConfig_BOARD_BootClockRUN);
164     CLOCK_SetXtal0Freq(oscConfig_BOARD_BootClockRUN.freq);
165     CLOCK_SetInternalRefClkConfig(
166         mcgConfig_BOARD_BootClockRUN.irclkEnableMode,
167         mcgConfig_BOARD_BootClockRUN.ircs,
168         mcgConfig_BOARD_BootClockRUN.fcrdiv);
169     CLOCK_SetFbiMode(mcgConfig_BOARD_BootClockRUN.dmx32,
170         mcgConfig_BOARD_BootClockRUN.drs,
171         CLOCK_CONFIG_FllStableDelay);
172     CLOCK_SetSimConfig(&simConfig_BOARD_BootClockRUN);
173     SystemCoreClock = BOARD_BOOTCLOCKRUN_CORE_CLOCK;
174 }
175
176 void CLKCFG_VLPR(void)
177 {
178     CLOCK_SetSimSafeDivs();
179     CLOCK_BootToBlpiMode(
180         mcgConfig_BOARD_BootClockVLPR.fcrdiv,
181         mcgConfig_BOARD_BootClockVLPR.ircs,
182         mcgConfig_BOARD_BootClockVLPR.irclkEnableMode);
183     CLOCK_SetSimConfig(&simConfig_BOARD_BootClockVLPR);
184     SystemCoreClock = BOARD_BOOTCLOCKVLPR_CORE_CLOCK;
185 }
186
187 #endif

```

## C.4 G2 header file

```

1  #ifndef G2_H
2  #define G2_H
3
4  #define CPU_MKL02Z32CAF4
5
6  #include "fsl_clock.h"
7
8  /* Port A */
9  #define PTA3          3U
10 #define PTA4          4U
11 #define PTA5          5U
12 #define PTA6          6U
13 #define PTA7          7U
14 #define PTA8          8U
15 #define PTA9          9U
16 #define PTA12         12U
17
18 #define SENSOR_EN1     PTA3
19 #define SENSOR_EN1_PT  PTA
20 #define SENSOR_EN1_PORT PORTA
21 #define SENSOR_EN2     PTA4
22 #define SENSOR_EN2_PT  PTA
23 #define SENSOR_EN2_PORT PORTA
24 #define LCD_BOT        PTA5
25 #define LCD_BOT_PT     PTA
26 #define LCD_BOT_PORT   PORTA
27 #define LCD_TOP        PTA6
28 #define LCD_TOP_PT     PTA
29 #define LCD_TOP_PORT   PORTA
30 #define V_CTRL_CH1     PTA7
31 #define V_CTRL_CH1_PT  PTA
32 #define V_CTRL_CH1_PORT PORTA
33 #define GAS1           PTA8
34 #define GAS1_PT        PTA
35 #define GAS1_PORT      PORTA
36 #define GAS2           PTA9
37 #define GAS2_PT        PTA
38 #define GAS2_PORT      PORTA
39 #define V_COMP_OUT     PTA12
40 #define V_COMP_OUT_PT  PTA
41 #define V_COMP_OUT_PORT PORTA
42
43 #define GAS1_ADC_CH     3U
44 #define GAS2_ADC_CH     2U
45
46 #define BOARD_HAS_SENSORS
47
48 /* Port B */
49 #define PTB0           0U
50 #define PTB1           1U
51 #define PTB2           2U
52 #define PTB3           3U
53 #define PTB4           4U
54 #define PTB5           5U
55 #define PTB13          13U
56
57
58 #define PWR_MEAS        PTB0
59 #define PWR_MEAS_PT     PTB
60 #define PWR_MEAS_PORT   PORTB
61 #define V_CTRL_CH2      PTB1

```

```

62 #define V_CTRL_CH2_PT PTB
63 #define V_CTRL_CH2_PORT PORTB
64 #define TEMPERATURE PTB2
65 #define TEMPERATURE_PT PTB
66 #define TEMPERATURE_PORT PORTB
67 #define V_CTRL_CLK PTB5
68 #define V_CTRL_CLK_PT PTB
69 #define V_CTRL_CLK_PORT PORTB
70 #define V_CTRL_CH3 PTB13
71 #define V_CTRL_CH3_PT PTB
72 #define V_CTRL_CH3_PORT PORTB
73
74 #define PWR_MEAS_ADC_CH 6U
75 #define TEMPERATURE_ADC_CH 4U
76
77 #define IR_IN PTB3
78 #define IR_IN_PT PTB
79 #define IR_IN_PORT PORTB
80 #define GPIO_LED_BLUE PTB3
81 #define GPIO_LED_BLUE_PT PTB
82 #define GPIO_LED_BLUE_PORT PORTB
83 #define GPIO_LED_GREEN PTB3
84 #define GPIO_LED_GREEN_PT PTB
85 #define GPIO_LED_GREEN_PORT PORTB
86 #define GPIO_LED_RED PTB3
87 #define GPIO_LED_RED_PT PTB
88 #define GPIO_LED_RED_PORT PORTB
89
90 /* Port features */
91 #define TPM0_CH0 LCD_TOP
92 #define TPM0_CH1 LCD_BOT
93
94 #define TPM1_CH0 V_COMP_OUT
95 #define TPM1_CH1 V_CTRL_CLK
96
97 #define BOARD_BOOTCLOCKRUN_CORE_CLOCK 47972352U
98 #define BOARD_BOOTCLOCKVLPR_CORE_CLOCK 4000000U
99 /*!< Oscillator 0pF capacitor load */
100 #define OSC_CAP0P 0U
101 /*!< Disable external reference clock */
102 #define OSC_ER_CLK_DISABLE 0U
103
104 extern uint32_t SystemCoreClock;
105
106 const mcg_config_t mcgConfig_BOARD_BootClockRUN =
107 {
108     /* FBI - FLL Bypassed Internal */
109     .mcgMode = kMCG_ModeFBI,
110     /* MCGIRCLK enabled, MCGIRCLK disabled in STOP mode */
111     .irclkEnableMode = kMCG_IrclkEnable,
112     /* Fast internal reference clock selected */
113     .ircs = kMCG_IrcFast,
114     /* Fast IRC divider: divided by 1 */
115     .fcrdiv = 0x0U,
116     /* FLL reference clock divider: divided by 1 */
117     .frdiv = 0x0U,
118     /* Mid frequency range */
119     .drs = kMCG_DrsMid,
120     /* DCO is fine-tuned for maximum frequency with
121     32.768 kHz reference */
122     .dmx32 = kMCG_Dmx32Fine,
123 };
124
125 const sim_clock_config_t simConfig_BOARD_BootClockRUN =
126 {

```

```

127     /* SIM_CLKDIV1 - OUTDIV1: /1, OUTDIV4: /2 */
128     .clkdiv1 = 0x10000U,
129 };
130
131 const osc_config_t oscConfig_BOARD_BootClockRUN =
132 {
133     /* Oscillator frequency: 0 Hz */
134     .freq = 0U,
135     /* Oscillator capacity load: 0pF */
136     .capLoad = (OSC_CAP0P),
137     /* Oscillator low power */
138     .workMode = kOSC_ModeOscLowPower,
139     .oscerConfig =
140     {
141         /* Disable external reference clock, disable
142          external reference clock in STOP mode */
143         .enableMode = OSC_ER_CLK_DISABLE,
144     }
145 };
146
147 const mcg_config_t mcgConfig_BOARD_BootClockVLPR =
148 {
149     /* BLPI - Bypassed Low Power Internal */
150     .mcgMode = kMCG_ModeBLPI,
151     /* MCGIRCLK enabled, MCGIRCLK disabled in STOP mode */
152     .irclkEnableMode = kMCG_IrclkEnable,
153     /* Fast internal reference clock selected */
154     .ircs = kMCG_IrcFast,
155     /* Fast IRC divider: divided by 1 */
156     .fcrdiv = 0x0U,
157     /* FLL reference clock divider: divided by 1 */
158     .frdiv = 0x0U,
159     /* Low frequency range */
160     .drs = kMCG_DrsLow,
161     /* DCO has a default range of 25% */
162     .dmx32 = kMCG_Dmx32Default,
163 };
164
165 const sim_clock_config_t simConfig_BOARD_BootClockVLPR =
166 {
167     /* SIM_CLKDIV1 - OUTDIV1: /1, OUTDIV4: /5 */
168     .clkdiv1 = 0x40000U,
169 };
170
171 const osc_config_t oscConfig_BOARD_BootClockVLPR =
172 {
173     /* Oscillator frequency: 0Hz */
174     .freq = 0U,
175     /* Oscillator capacity load: 0pF */
176     .capLoad = (OSC_CAP0P),
177     /* Use internal clock */
178     .workMode = kOSC_ModeOscLowPower,
179     .oscerConfig =
180     {
181         /* Disable external reference clock */
182         .enableMode = OSC_ER_CLK_DISABLE,
183     }
184 };
185
186 static void CLOCK_CONFIG_FllStableDelay(void)
187 {
188     uint32_t i = 30000U;
189     while (i--)
190     {
191         __asm__("nop");

```

```

192     }
193 }
194
195 void CLKCFG_Boot( void )
196 {
197     CLOCK_SetSimSafeDivs ( );
198     CLOCK_InitOsc0(&oscConfig_BOARD_BootClockRUN );
199     CLOCK_SetXtal0Freq( oscConfig_BOARD_BootClockRUN . freq );
200     CLOCK_SetInternalRefClkConfig (
201         mcgConfig_BOARD_BootClockRUN . irclkEnableMode ,
202         mcgConfig_BOARD_BootClockRUN . ircs ,
203         mcgConfig_BOARD_BootClockRUN . fcrdiv );
204     CLOCK_SetFbiMode( mcgConfig_BOARD_BootClockRUN . dmx32 ,
205         mcgConfig_BOARD_BootClockRUN . drs ,
206         CLOCK_CONFIG_FllStableDelay );
207     CLOCK_SetSimConfig(&simConfig_BOARD_BootClockRUN );
208     SystemCoreClock = BOARD_BOOTCLOCKRUN_CORE_CLOCK;
209 }
210
211 void CLKCFG_VLPR( void )
212 {
213     CLOCK_SetSimSafeDivs ( );
214     CLOCK_BootToBlpiMode(
215         mcgConfig_BOARD_BootClockVLPR . fcrdiv ,
216         mcgConfig_BOARD_BootClockVLPR . ircs ,
217         mcgConfig_BOARD_BootClockVLPR . irclkEnableMode );
218     CLOCK_SetSimConfig(&simConfig_BOARD_BootClockVLPR );
219     SystemCoreClock = BOARD_BOOTCLOCKVLPR_CORE_CLOCK;
220 }
221
222 #endif

```

