Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Measurement

Master's Thesis

Design of electronic fuse emulator with advanced functions

Bc. Josef Burda
Cybernetics and Robotics

Supervisor
doc. Ing. Jan Fischer, CSc.

Prague 2022

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Burda Josef** Personal ID number: **466347**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Measurement**

Study program: **Cybernetics and Robotics**

Branch of study: **Cybernetics and Robotics**

II. Master’s thesis details

Master’s thesis title in English:

Design of electronic fuse emulator with advanced functions

Master’s thesis title in Czech:

Návrh emulátoru elektronické pojistky s pokročilými funkcemi

Guidelines:

Design and realise a device that will emulate new functions of a power electronic fuse with integrated protection functions for automotive use. To emulate these new functions, use an automotive grade MCU from the SPC58xx family. Design the hardware and the software of the device. When designing the emulator pay attention to the possibility of integrating the emulator into a car’s power distribution system. Reaction time requirements to certain signals (e.g., emergency stop) should also be considered. During testing focus on carmaker requirements and functional safety.

Bibliography / sources:

1. STMicroelectronics: RM0403 Reference manual SPC58 2B Line, Rev 7, 2021
2. RIBBENS, William B.: Understanding automotive electronics. 7th ed. ELSEVIER, ©2012, ISBN 978-0-08-097097-4
3. VOLKSWAGEN, VW 80000 Electric and Electronic Components in Motor Vehicles up to 3.5 t General, Requirements, Test Conditions, and Tests, 2017
4. Vít Záhlava: Návrh a konstrukce DPS, Ben 2010

Name and workplace of master’s thesis supervisor:

doc. Ing. Jan Fischer, CSc. Department of Measurement FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **31.01.2022** Deadline for master's thesis submission: **20.05.2022**

Assignment valid until:
by the end of summer semester 2022/2023

doc. Ing. Jan Fischer, CSc. Supervisor’s signature	Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

. | Date of assignment receipt | Student’s signature

Declaration

I declare, that I have written this thesis independently, with help from the thesis supervisor doc. Ing. Jan Fischer, CSc. and that I have used only the literature quoted in the thesis. I further declare, that I have no objections to reproduction or presentation of this thesis or it's parts with consent of the faculty.

In Prague 20.5.2022

.....
Bc. Josef Burda

Acknowledgement

I would like to thank everyone that offered me support when writing this thesis. I would especially like to thank my supervisor, doc. Ing. Jan Fischer, CSC., for his patient support and guidance throughout the project. Thanks also go to all of my colleagues, who were willing to discuss ideas and solutions to problems. Last but not least, I would like to thank all of my family members who supported me throughout my life.

Abstract

The goal of this thesis is to design and realise an emulator of advanced functions of an electronic fuse currently in development. The emulator is built around an already existing electronic fuse device. The additional features are emulated by a microcontroller. Most important of these feature is the ability to control and diagnose the electronic fuse via SPI bus. An analysis of the parameters of the future device was performed and based on that the hardware and software was designed and implemented. In Application Validation measurements were performed on the finished device to validate that the requirements were met and determine it's parameters.

Key words: electronic fuse, automotive, emulator, microcontroller

Abstrakt

Tato práce se zabývá návrhem a realizací emulátoru pokročilých funkcí elektronické pojistky, která je ve vývoji. Emulátor je založen na již existující elektronické pojistce, která těmito funkcemi nedisponuje. Pokročilé funkce budou emulovány pomocí mikrokontroléru.

Mezi nejdůležitější z těchto funkcí patří možnost ovládat a diagnostikovat elektronickou pojistku prostřednictvím rozhraní SPI. Byl proveden rozbor parametrů budoucího zařízení, na jehož základu byl navržen a relizován hardware a software emulátoru. Na dokončeném zařízený byla provedena validační měření, jejichž cílem bylo ověřit zda byly splněny všechny požadavky.

Klíčová slova: elektronická pojistka, automotive, emulátor, mikrokontrolér
Překlad názvu: Návrh emulátoru pokročilých funkcí elektronické pojistky

Contents

List of Figures

List of Tables

List of Abbreviations

ADC	Analog to Digital Converter
e-fuse	Electronic Fuse
EMC	Electromagnetic Compatibility
FCS	Fuse Current Setting
FTS	Fuse Time Setting
FW	Firmware
GPIO	General Purpose Input Output
GSB	Global Status Byte
GUI	Graphical User Interface
HAL	Hardware Abstraction Layer
HW	Hardware
IAV	In Application Validation
I/O	Input/Output
I^2C	Inter Integrated Circuit
JTAG	Joint Test Action Group
LDO	Low Drop-Out Voltage Regulator
MCU	Microcontroller Unit
MOSFET	Metal-Oxide Silicon Field Effect Transistor
OLD	Open Load Detection
OTP	One Time Programmable (memory)
PCB	Printed Circuit Board
PIT	Periodic Interrupt Timer
PLL	Phase-Locked Loop
PWM	Pulse Width Modulation
RAM	Random Access Memory
RMS	Root Mean Square
ROM	Read Only Memory
RUL	Remaining Useful Life
SAR	Successive Approximation Register
SPI	Serial Peripheral Interface
SW	Software
UART	Universal Asynchronous Receiver-Transmitter

Chapter 1
Introduction

As modern cars get more complex, their power distribution systems are getting more complex as well. This increase in complexity requires a more sophisticated solution for power distribution. The standard melting car fuses have not changed for several decades, and they have several shortcomings, which are no longer acceptable in modern cars. Among these shortcomings is the fact, that melting fuses are not reusable. One way of addressing the growing requirements for reliability and safety is the use of electronic fuses. An electronic fuse is a device which replaces the traditional melting fuse as a circuit protection device while offering more features and improved performance. The adoption of electronic fuses in the automotive industry is relatively recent, and the developers of car power distribution systems need to get familiar with them before applying them to a car's systems.

This thesis aims to design and develop an emulator of an electronic fuse that is currently in development (VNF9D1M5), which will be used to provide the automotive developers with a way to familiarise themselves with the e-fuse before it is manufactured. The emulator will emulate advanced functions of the future device and will be built around an already existing e-fuse device (VNF7000AY) that does not provide these features. The most notable of these functions is the control and diagnostics of the e-fuse via SPI.

The emulator's development will involve the analysis of the future device's parameters, from which the requirements for the emulator's design will be derived. Based on these requirements, the emulator hardware and software will be designed and implemented. The HW design will include the selection of components and the design of a PCB. The SW design and implementation will involve implementing then features not available on the VNF7000AY. When the emulator's design is complete and the device is realised, validation measurements will be performed to confirm that the design requirements have been met.

Chapter 2
Assignment analysis

As mentioned in the introduction, the goal of this thesis is to design and implement an emulator of an electronic fuse with advanced functions. This emulator will be using an already existing electronic fuse. The purpose of the emulator will be to emulate advanced functions, which will be added to a new iteration of the e-fuse. These new functions include communication via SPI and basic diagnostic of the device as well as better configurability of the device and several other functions.

2.1 Emulator design requirements

Before the emulator is designed, the requirements of the new device need to be analysed in order to determine which functions can be emulated and what will be its limitations. This analysis will produce a list of achievable parameters of the emulator. Based on this, the emulator specification will be drawn up.

2.2 Emulator hardware design

The design of the emulator's hardware will be focused on meeting the requirements for the future e-fuse device. Some of these requirements will be fulfilled only partially or not at all. This is because it would be too complicated to implement them, and they are not key to the function of the device. HW design will be based on the block schematic, which was created as part of the requirement analysis.

The emulator will be designed as an evaluation platform at first. This means that the design will put emphasis on the accessibility of signals and generally making it easier to work with the device in a laboratory. Another more compact and simplified version might be designed when this version is properly evaluated, but that will not be a part of this thesis. The emulator will be designed to be able to integrate into a vehicle's power distribution system.

The first step of the design will be selecting the components from which the emulator will be assembled. This includes microcontroller selection, integrated circuits and connectors. E-fuse devices are already provided, which means I will not be selecting them. This process also includes checking the compatibility of the devices and their availability. Since the device is going to have automotive applications, it is also important to select automotive grade components where possible.

After the component selection comes the schematic design. The schematic will be derived from the block schematic. For the integrated circuits, the microcontroller and

the e-fuses, the reference designs will be followed. The schematic design also involves determining the pinout of the MCU. This will be partly done in component selection because the MCU will need to have enough pins to accommodate our needs. Before proceeding with the PCB design, the schematic will be reviewed.

Once the schematic has been finished and reviewed, I will be able to proceed with the printed circuit board design (PCB). During the design, basic design rules and device requirements shall be taken into account. The emulator is required to work with high currents (up to 64 A). This means that power losses on components, connectors and the board itself will need to be taken into account. Before manufacturing the board, a thermal simulation will be performed to determine whether the board and the components on it have sufficient cooling. A review of the PCB design will be performed as well.

When all this is done, the board design files will be sent to the chosen manufacturer. After receiving the board, the hardware part of the device will be completed by soldering the components onto the PCB. This will also include some testing to uncover design or soldering errors and prepare the board for software development.

2.3 Emulator software design and implementation

Software development for this project will involve the design and implementation of firmware (FW) for the MCU, which emulates the behaviour of the new e-fuse. Part of the development can be done before the hardware is ready. While waiting for the PCB to be manufactured, I shall familiarise myself with the MCU and its peripherals (SPI, ADC,...). Peripheral and core configurations will be examined, and the most appropriate ones will be selected.

Once the hardware is available, it will be possible to develop the code on it. The peripheral and I/O configurations shall be verified to work on the new hardware. The future e-fuse device behaviour is described using a finite state machine. A modified version of this state machine will be defined for the emulator. This modified state machine will be implemented. The FW needs to be designed to be indistinguishable from the real device by the user.

Since there are strict safety requirements concerning reaction times and speed in general, no Hardware Abstraction Layer (HAL) libraries shall be used. The MCU will be programmed using direct register access. This approach will result in more difficult development but also better FW performance. Important signals shall be handled by interrupts in order to minimise the reaction time as much as possible.

Alongside the FW for the microcontroller, software for an application platform will be developed. This software will be provided by STMicrocelectronics and its development is not part of this thesis. This application software will be used for the evaluation of the emulator.

2.4 Testing and validation

Once the emulator is finished, it needs to be tested and validated. This means it will be necessary to verify that the design requirements were met. It will also provide a more accurate view of the emulator's parameters. The testing and validation will be carried out according to a test specification which will have to be drawn up. This must

be done as a part of the emulator's specification. The test specification will be based on the requirement list and automotive industry standards. Because the emulator is built on an already existing e-fuse device, it will not be necessary to perform every test on the power part of the device.

Chapter 3
Introduction to fuses

In this chapter, I will give a brief overview of automotive fuses and their application. The reasons why fuses are needed and their parameters will be discussed. Then the concept of electronic fuses will be introduced and compared to the standard melting fuse. Applications and benefits (and disadvantages) of electronic fuses will be discussed as well.

3.1 Fuses in automotive applications

The purpose of fuses in power distribution systems is to protect individual components and the entire system. In standard melting fuses, this protection is achieved by the fuse melting under overcurrent conditions. When the fuse melts, the current flow in the circuit is interrupted, and the overcurrent condition is eliminated.

Wiring in cars is made from copper. Even though copper has excellent electrical conductivity, the wire resistance is not negligible. The wire resistance can be determined using the following formula:

$$R = \frac{\rho \cdot l}{A}, \tag{3.1}$$

where ρ is the electrical resistivity of the material (copper $\rho_{Cu} = 17 \times 10^{-9}\,\Omega\,\mathrm{m}$), l is the length of the wire, and A is the wire cross-section area.

When the passing current through a non-ideal conductor, Joule's law applies, meaning that a part of the power flowing through the conductor is turned into heat. The power turned into heat is proportional to the conductor resistance and square of the current as expressed in this formula:

$$P = I^2 R. \tag{3.2}$$

The conductor needs to dissipate the heat generated by the passing current. When the current is within the rating, the conductor manages to dissipate the heat into its surroundings. When the current exceeds the rating, the wire begins to overheat. This issue is exacerbated by the fact the wires in cars are bound into a wiring, where there are more wires generating heat and the heat dissipation is diminished. Wire overheating can lead to isolation melting which can lead to a short circuit between the wires in a wiring harness. It can even lead to the melting of the conductor itself. These are serious safety risks, and fuses are essential in reducing them.

In this section, an overview of fuse parameters and behaviour will be given, and the fault states against which fuses protect the circuits will be presented.

3.1.1 Types of faults in power distribution systems

Fuses generally protect against overcurrent faults. This means that the current flowing through the circuit exceeded the maximum rated current. This condition can be caused by both overloads and short circuits. When any of these faults occur, the fuse must open the circuit and interrupt the current flow. The fuse reaction must be fast enough in order to prevent any permanent damage to the circuit.

An overload in the circuit occurs when the current exceeds the rating of the wiring and the devices connected to it. In this condition, current is flowing only through the defined pathways (i.e. the wires). This usually happens when too many active devices are connected to the circuit at the same time or when a device experiences a malfunction. This can lead to wiring and/or device damage due to overheating.

Short circuit occurs when an unintended connection is made between the power source (battery, alternator) and the ground. This could happen as a result of wiring insulation damage or device malfunction. In this case, the current is flowing through an undefined path. Short circuits have very low resistance, which means that the supply voltage forces huge currents through them. Such large currents can damage the wiring, connected devices and even the power supply itself. Because the current passing through the wires during short circuit is so high, the chance of wiring insulation melting and catching fire is greatly increased.

3.1.2 Fuse parameters

When selecting an appropriate fuse for a circuit, it needs to have defined parameters. These parameters describe the fuse's maximum ratings and its behaviour during overcurrent. Fuse parameters can also be applied to e-fuses. Following is a brief overview of these parameters.

Voltage rating

As with most electric and electronic components, fuses also have maximum voltage ratings. This is the maximum voltage that can occur in the circuits. Fuses are designed so that they operate safely under this voltage. When voltage rating is exceeded, the fuse might not work as expected due to arcing. The circuit voltage must never exceed the fuse maximum voltage rating. Voltage rating is usually specified in volts DC. Sometimes AC value is given as well.

Breaking capacity

Breaking capacity or Interrupting rating determines the maximum current the fuse is able to handle safely. During an overcurrent event, the current through the fuse can be several times the nominal current. If this current is higher than the breaking capacity, it can result in fuse package damage. The damaged fuse might damage surrounding components (e.g. other fuses), or it may not break the circuit properly. Fuse breaking capacity is specified as maximum current at rated voltage.

Current rating

The current rating specifies the maximum current that the fuse can carry continuously. This rating can change depending on external conditions, especially temperature.

Under normal conditions (25 °C), the maximum current is usually derated to a lower value. This is done to avoid the fuse blowing prematurely due to manufacturing tolerances. Under normal conditions, the derating factor is typically 25%. For example a fuse rated for 20 A would be derated to just 15 A at 25 °C.

When ambient temperature increases, the fuse's ability to dissipate heat decreases. This means that the fuse will blow at a lower current than at 25 °C. In order to determine which fuse to use, rerating for increased ambient temperature needs to be performed. The rating for the defined temperature can be determined using the following formula:

$$I_{RAT} = \frac{I_{NOM}}{0.75 \cdot r_T}, \tag{3.3}$$

where I_{NOM} is the nominal circuit current, r_T is the temperature rerating factor and I_{RAT} is the required rating of the fuse. The rerating factor can be obtained from Temperature Rerating Curve from the fuses datasheet. An example of this curve can be seen in figure 3.1. For example if the circuit current is $I_{NOM} = 10$ A and the ambient temperature is $t = 80$ °C, the rerating factor from the curve is $r_T = 0.92$. The rerated current is then

$$I_{RAT} = \frac{10}{0.75 \cdot 0.92} = 14.5\,\text{A}. \tag{3.4}$$

This means that at least a 15 A fuse should be used for the circuit under these conditions.

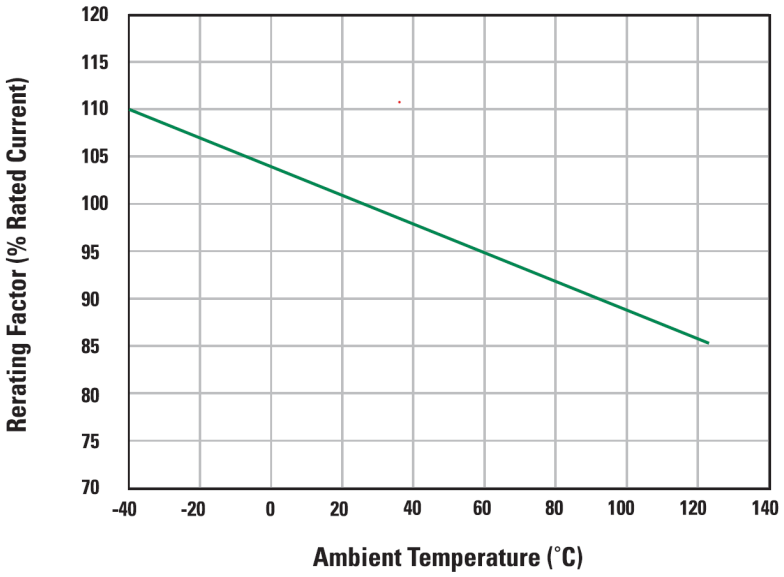


Figure 3.1: Example of fuse temperature rerating curve [1]

Time-current characteristic

All of the parameters mentioned so far described static parameters, but when selecting a fuse, reaction times are as important as the current rating. Time-current characteristic describes the response times to different overcurrent values. These characteristics are always declining, meaning that when the current gets higher, the time needed for the fuse to break the circuit is shorter. Knowing the time-current characteristic helps to select the right fuse when the fault needs to be cleared in a specified

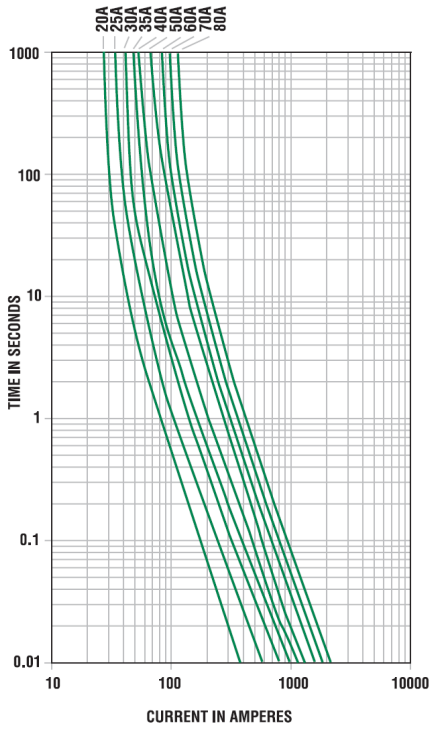


Figure 3.2: Average time-current characteristic of differently rated fuses [1]

time. It can also be used to choose a fuse that will allow a temporary overload. In figure 3.2 you can see a comparison of average,characteristics of fuses with different current ratings. In figure 3.3 characteristics of two fuses with identical current ratings and different constructions are compared.

I^2t rating

I^2t expresses the amount of thermal energy available from the current. The unit is A^2s. It is also used for rating electric motors or wiring. There are three kinds of I^2t rating. Melting I^2t rating expresses the amount of energy needed to melt the fuse. Arcing I^2t expresses the amount of heat dissipated during arcing. Total Clearing I^2t expresses the total amount of heat dissipated in the fuse from the start of the overcurrent event to its end when the fuse clears the fault by interrupting the current.

I^2t rating is important when selecting fuses. The total clearing I^2t of a fuse needs to be higher than the I^2t rating of the wiring of the circuit it protects.

3.1.3 Fuse selection process

All of the parameters mentioned above need to be considered when selecting a fuse. Another complication in the selection process comes from the conditions in which the fuse operates, such as ambient temperature. In order to compensate for these, the fuses need to be rerated according to specified characteristics.

When using melting fuses, another issue is introduced into the selection process, and that is the variance of the parameters. As with essentially every electrical and electronic component, no two identically rated fuses are the same. The parameters in the fuses datasheet are only the average values. As you can see in figure 3.4, taken

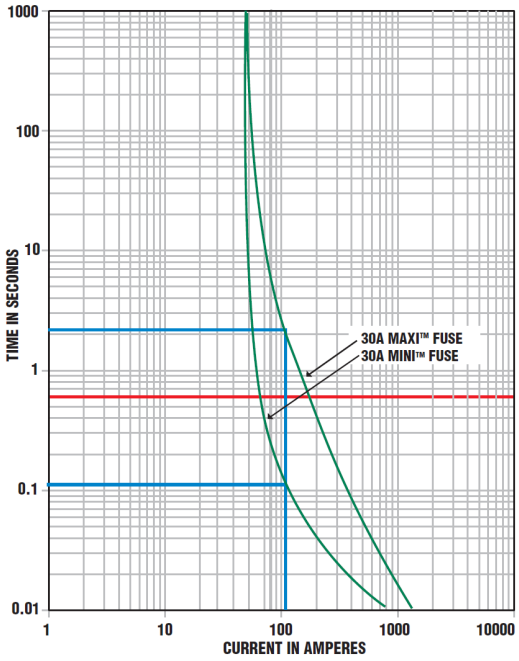


Figure 3.3: Comparison of time-current characteristics of fuses with different construction [1]

from a LittelFuse catalogue [2], the maximum and minimum opening times of a fuse can differ by several orders of magnitude. This means that in the event of an overload, it is difficult to determine for how long will the overcurrent flow. When rating the wiring and devices in the circuit, the longest time the fuse takes to open needs to be considered. This usually leads to overrating of the wiring in order for it to handle the longer overcurrent event. The overrating of the wiring leads to increased mass and cost of the wiring. The increased mass also leads to increased fuel consumption of the car. The electronic fuse aims to resolve this problem by being much more accurate than the melting fuse.

Time-Current Characteristics

% of Rating	Opening Time Min / Max (s)
110	360,000 / ∞
135	0.75 / 120
200	0.15 / 5
350	0.08 / 0.25
600	0.03 / 0.1

Figure 3.4: Example of maximum and minimum opening time of a fuse [2]

3.2 Introduction to electronic fuses

As modern car systems get more and more complex, the power delivery systems need to keep up with the requirements. Many of these requirements focus on safety and reliability. Fuses are key safety components of a car's power distribution system safety. To make the system safer and more reliable, the fuses need to get more sophisticated. The advancements in areas of power transistors and integrated circuits in general lead to the concept of the electronic fuse (e-fuse), which is an ideal replacement for the traditional fuse.

In its most basic form, the electronic fuse uses a power switch, which is usually a MOSFET, some form of current sensing and control logic which controls the switch and evaluates the current sense data. This structure is illustrated in figure 3.5. When the current in the circuit is below the set threshold, the transistor remains switched on. Once the current exceeds the threshold (overcurrent event), the transistor switches off, interrupting the current through the circuit.

This is the simplest version of the electronic fuse possible. Thanks to the modern processes that allow integrating complex circuits onto the same chip as the power transistor, e-fuses can have many more functions. Different and more sophisticated protection mechanisms and even diagnostics of the fuse can be implemented in one device.

In this section, the principle of e-fuse will be introduced and the main protection mechanisms and other features of electronic fuses will be presented. After that, a comparison between the traditional melting fuse and the electronic fuse will be made. This will show the reasons for adopting electronic fuses into modern car power systems.

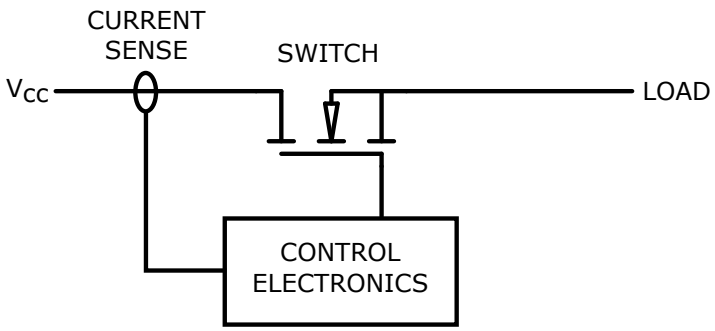


Figure 3.5: Basic schematic of an electronic fuse

3.2.1 Electronic fuse principle

An electronic fuse is an electronic device that can be used to protect a circuit in case of an overcurrent. The electronic fuse can also protect against other kinds of faults in the power grid. Its primary purpose in a system is the same as that of a standard melting fuse: interrupt current passing through it when it reaches a predefined level. However, the mechanisms used to interrupt the current are different. When the melting fuse experiences an overcurrent, the conductive element of the fuse heats up to the point of melting, which interrupts the current. E-fuses use electronic switches, transistors, and current sensing. When the e-fuse detects an overcurrent, it opens the switch, interrupting the current flow.

E-fuse can be realised either as a single integrated circuit or assembled from discrete

components (transistors, diodes, transils,...). The discrete solution allows the designer to select exactly which protections to implement. Unfortunately, this approach results in a much larger and less efficient device than the integrated solution. Modern manufacturing processes allow the integration of control logic onto the same silicon as the power switch. This tight integration allows for much more control over the transistor. For example, it is much easier to measure the transistor's temperature in the IC than to measure it on a discrete component.

Since the switching off of the transistor is controlled by the control logic of the e-fuse, the fuse parameters can be configured to required values instead of having preset parameters from the factory. This is very useful because the fuse can be configured precisely to the requirements of the circuit it protects. E-fuses are also more precise when it comes to current measurement and interrupt times. The VNF7000AY's current sense has accuracy of $\pm 7\%$ for currents from 10 A to 30 A [3], which is much lower than tolerances of melting fuses. The current sense accuracy is worse for current that are much lower than nominal, which is why, it is important to select the appropriate e-fuse device. Thanks to this, the wiring in the circuit does not need to be overrated, resulting in weight and cost reduction.

Since the fuse control logic can be realised as an integrated circuit, it is much easier and cheaper to implement more functions than standard overcurrent protection. These functions can include more protection mechanisms, diagnostics and configuration of the fuse's behaviour.

3.2.2 Current sensing in electronic fuses

For the electronic fuse to be able to perform its function, it needs to be able to monitor the amount of current passing through the circuit. In this section, the various methods of current sensing will be presented. Since there are many ways to measure current, we first need to determine the requirements for an automotive e-fuse.

The current measurement methods vary in complexity and accuracy and their principle of operation. One of the basic classifications is to direct and indirect methods. Direct methods need to interrupt the circuit in order to insert a measurement device (ammeter) through which the current will pass. Indirect methods do not need to interrupt the circuit because they measure the magnetic fields generated by the passing current. Since the e-fuse is already inserted into the circuit, the direct methods are the obvious choice.

Current Shunt

The most used direct current measurement method is using a current shunt. A current shunt is a resistor of a known resistance inserted into the circuit. When current passes through the shunt, a voltage drop appears on it.. By measuring the voltage across the shunt, the current can be determined using Ohm's law:

$$U = R \cdot I. \tag{3.5}$$

This method can be very accurate, but it is dependent on the accuracy of the voltage measurement and the accuracy of the resistor. Current shunts also have very good resistance against interference, depending on the measured current (frequency, amplitude).

Unfortunately, the principle upon which current shunts operate is also their greatest weakness. When current passes through the shunt, heat is generated (Joule's law):

$$P = U \cdot I = R \cdot I^2. \tag{3.6}$$

This causes the shunt to heat up, which can lead to its damage or destruction. The power loss on a shunt resistor also decreases the efficiency of the entire system. One shunt would probably not be a problem, but there can potentially be dozens of these in a single car. This can be mitigated by using lower resistance shunts. Using a lower resistance shunt also means lowering the voltage across the shunt. This can complicate the measurement and potentially make it less accurate. A lower resistance current shunts are also more expensive, as they require tighter manufacturing tolerances. A current shunt's price is comparable to the price of an entire electronic fuse.

Ratio Metric Sense MOSFET

When using MOSFETs, another current measurement method is available. Modern power MOSFETs are composed of many parallel cells with their sources, drains and gates connected. This is done to achieve a large current carrying capacity with low on-resistance. This is a very simplified explanation; for more detail, see a power electronics book. Some of these cells can have their sources electrically disconnected from the others and connected to a separate pin. This pin is usually called Current Sense or Mirror. If this pin is connected to the same potential as the source of the power transistor, the sense transistor acts as a current mirror. It consists of a large power transistor Q_{pow} and a small sense transistor Q_{sense} as illustrated in figure 3.6. In this case, an N-type MOSFET is used as a low side switch. For now, the R_{sense} resistor will be considered to be $0\,\Omega$.

The Shichman-Hodges model of the MOSFET provides an equation that can be used to calculate the current flowing through the transistor based on its parameters and voltages:

$$I_D = \frac{1}{2} K_p \frac{W}{L} (V_{GS} - Vth)^2 (1 + \lambda V_{DS}) \tag{3.7}$$

- I_D is the drain current
- K_p is the technological constant of the transistor
- W and L are the width and length of the transistor
- V_{GS} is the gate-source voltage
- V_{th} is the transistor threshold voltage
- V_{DS} is the drain-source voltage
- λ is the channel length modulation constant

Since both the transistors have their gates, drains, and sources connected and are made using the same technology, we can substitute into this equation and determine the ratio between I_{D_sense} and I_{D_pow}.

$$\frac{I_{D\_pow}}{I_{D\_sense}} = \frac{\frac{1}{2} K_p \left(\frac{W}{L}\right)_{pow} (V_{GS} - Vth)^2 (1 + \lambda V_{DS})}{\frac{1}{2} K_p \left(\frac{W}{L}\right)_{sense} (V_{GS} - Vth)^2 (1 + \lambda V_{DS})} = \frac{\left(\frac{W}{L}\right)_{pow}}{\left(\frac{W}{L}\right)_{sense}} \tag{3.8}$$

This means that the ratio between the drain currents is the same as the ratio between the sizes of the transistors. This ratio is given by the number of MOSFET cells used in the power part and the number of cells used in the sense part.

However, to measure the current passing through the sense transistor, a resistor R_{sense} needs to be connected to the source. This, unfortunately, causes the drain-source voltage of the sense transistor V_{DS_sense} to change and no longer equal to V_{DS_pow}. This causes the above equation to no longer apply. The higher the R_{sense} resistance, the worse the accuracy of the mirror. This makes the selection of the R_{sense} resistor a little complicated. In integrated circuits, the solutions are more complex, and an example of such a solution can be seen in figure 3.7. The transistor Q_1 and the operational amplifier U_1 are used to match the source voltages of transistors Q_{sense} and Q_{pow} [4].

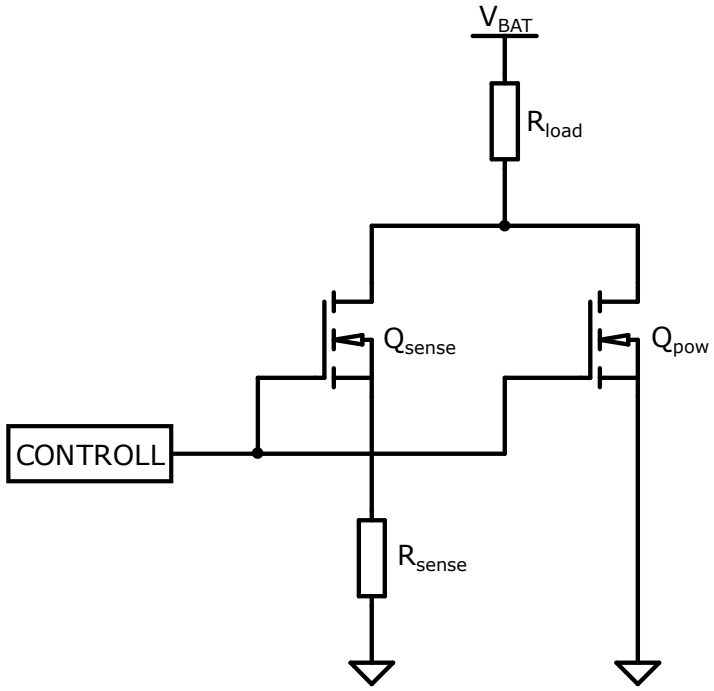


Figure 3.6: Current sensing mirror

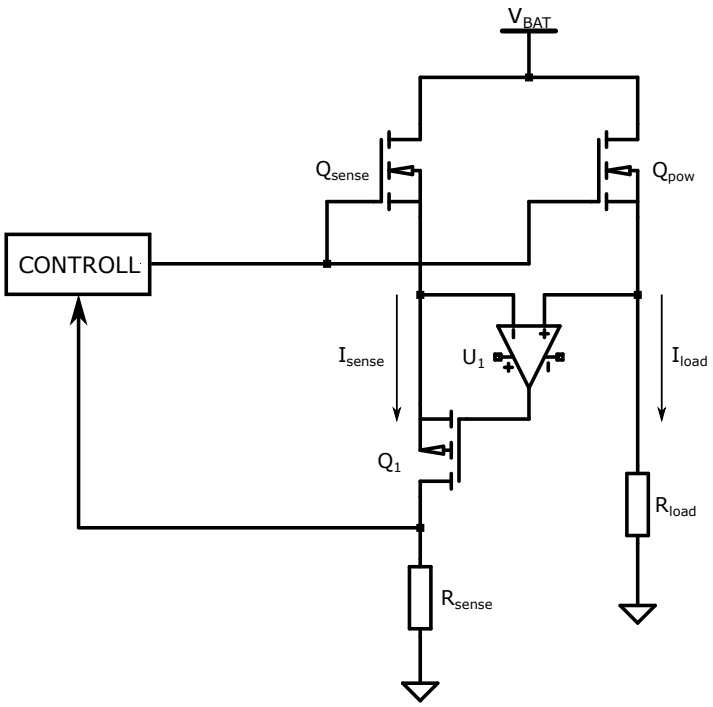


Figure 3.7: Improved current sensing mirror

3.2.3 Protection mechanisms integrated into electronic fuses

As mentioned in previous sections, thanks to the control logic being integrated into the e-fuse, it is possible to implement more protection mechanisms than just overcurrent protection. Some of these protections are also necessary to protect the e-fuse itself. This section will introduce the basic and the advanced protection mechanisms.

Short-circuit protection

During a short circuit, the current flows through a low resistance path from one terminal of the power supply to the other. The current is limited only by the resistance of the wiring and the voltage and output resistance of the power supply. That is why short circuit currents are usually very high.

One of the main functions of any fuse is to protect the circuit from short circuits by interrupting the current. This applies to electronic fuses too. Melting fuses, as the name suggests, interrupt the current by melting due to heating up by the passing current. E-fuses monitor the current passing through them, and when the current exceeds the preset limit, the control logic turns off the transistor, interrupting the current.

When a short circuit occurs, the current is so high that it can cause the power supply voltage to drop. If this voltage drop is significant and long enough, it can cause problems for systems connected to the same power supply. This is considered a safety risk. It is therefore important to interrupt the current as soon as possible. For reasons explained in the previous section, the melting fuse cannot react very fast. However, a well-designed e-fuse can respond much more quickly, as illustrated in figure 3.8.

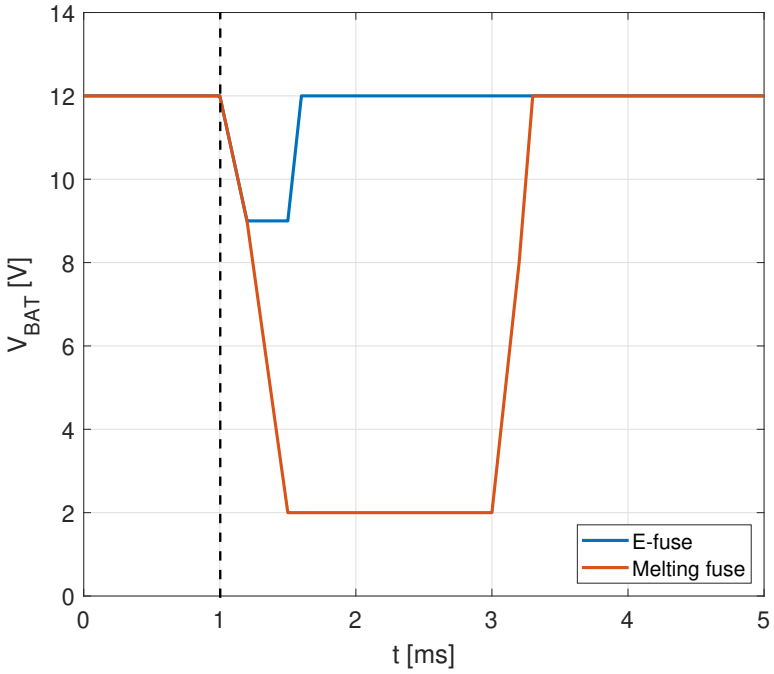


Figure 3.8: Voltage drop caused by short circuit at $t = 1\,\text{ms}$ (e-fuse vs. melting fuse)

Overload protection

During an overload, a current higher than expected is running through the circuit, usually due to a device fault. These currents are lower than during a short-circuit but can still cause damage. They are especially dangerous to wiring, as it can heat up to the point of melting the insulation. This can, in turn, lead to a short circuit. It is sometimes necessary to allow an overload for some time, for example, when turning a device on and charging its input capacitors. The I^2t rating of the wiring is the deciding factor here. It must not be exceeded.

Melting fuses have their own I^2t rating. A fuse with an I^2t rating lower than that of the wiring will protect it. However, the tolerances in fuse parameters mean that the wiring needs to be overrated in order to handle the worst-case scenario of the fuse's performance.

E-fuses can use logic and mathematics. One option is to have a set of predefined threshold currents and corresponding turn-off times. When the current exceeds a certain threshold, the fuse will wait for the defined amount of time, and if the current does not drop, it interrupts the circuit. This approach is very simple and, for many reasons, impractical. For instance, it does not really take repeating overloads into account.

Another option is to compute the I^2t of the current running through the fuse. By its principle, this approach requires some form of integrating the current over time. One option is to sample the current periodically and compute the I^2t using a microcontroller. Unfortunately, using an MCU would make the e-fuse too expensive. A simpler approach is to use a counter. When the current through the e-fuse exceeds the preset threshold, the counter starts counting up. The counter increment is proportional to the size of the overcurrent. When the current drops below the threshold, the counter starts to decrement. If the counter reaches a certain predefined value, which corresponds to an I^2t rating, the fuse interrupts the current. This process is illustrated in figure 3.9. The

second overload is long enough for the fuse to latch off at $t = 5\,\text{ms}$.

The I^2t computation allows the fuse to be closely matched to the wiring and devices protected by the e-fuse. E-fuses also offer greater precision than melting fuses, which reduces the need for overrating the wiring. This approach is used for fault detection in many other systems, as it helps to reduce false alarms. The way this system is implemented is key to the accuracy of the I^2t computation, but the exact implementation is kept secret by the manufacturers.

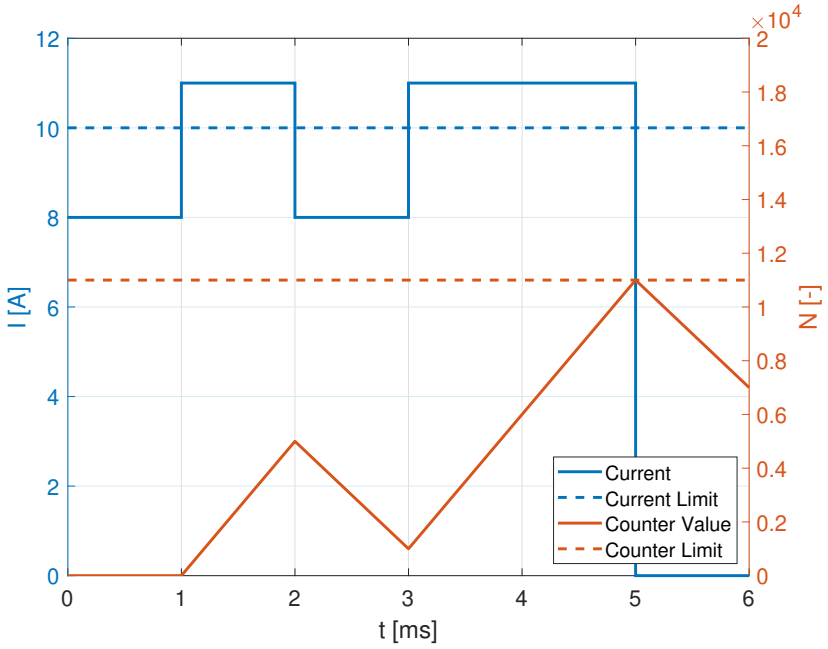


Figure 3.9: Overload protection illustration

Overvoltage and Undervoltage protection

So far, all of the mentioned protections are present in melting fuses as well as in e-fuses. Since the e-fuse already contains some form of an ADC, it is possible to monitor the supply voltage. The power supply voltage (V_{BAT}) is not constant. It can be higher or lower than the nominal voltage.

Undervoltage occurs when the V_{BAT} voltage drops a certain amount below the nominal level. It can be caused by a fault in the circuitry or in a device in the system. An undervoltage can cause the systems in the power grid to behave incorrectly or stop working altogether. The e-fuse can detect the undervoltage event and can send a warning to other parts of the system. This warning can be used to turn off devices safely before they are affected by the undervoltage.

During an overvoltage, V_{BAT} voltage exceeds the nominal level. This can cause unexpected behaviour and even damage to some systems. A melting fuse has no way of detecting an overvoltage; therefore, the protection needs to be built in into the devices. An e-fuse can monitor the V_{BAT} voltage, and when it detects an overvoltage, it can generate a warning and eventually interrupt the circuit. This means that the devices in the circuit protected by the e-fuse do not need to implement their own overvoltage protection.

Overvoltage protection becomes more important as new power distribution systems in cars start to include 48 V supply. A short between the 12 V and 48 V power rails could

cause the 48 V voltage to appear on the 12 V power rail. This would probably destroy most devices connected to that power rail. An e-fuse with overvoltage protection will reliably protect the devices.

Overtemperature protection

While the above-mentioned protection mechanisms focus on protecting the circuit wiring and devices connected to it, overtemperature protection primarily aims to protect the e-fuse itself. Like every electronic device, the e-fuse has a maximum operating temperature. When this temperature is exceeded, the e-fuse may not behave according to specification or be damaged. The e-fuse can overheat due to the high current passing through it (power loss in the switch) or due to high ambient temperatures. Usually, overheating is caused by the combination of these two factors.

In order for the power distribution to operate safely, the e-fuse needs to operate safely as well. This is the reason why e-fuses implement overtemperature protection. It is fairly easy to implement temperature monitoring in integrated circuits. A diode or a pair of diodes is used. The temperature dependence of these structures is well known ($-2\,\mathrm{mV\,K^{-1}}$ for an Si diode). The voltage of this structure is monitored, and once the voltage corresponding to the maximum junction temperature is reached, the overtemperature protection engages.

The e-fuse can also generate a warning that its temperature is approaching the maximum possible level. This can help the system to prevent the e-fuse from overheating by lowering the current passing through it or engaging active cooling (a fan).

Power limitation

Another way the e-fuse can protect itself is through power limitation. Sufficiently high currents can damage the e-fuse even before the overtemperature or short-circuit protections step in. The high currents can potentially melt the bonding wires between the silicon chip itself and the package pins. This would make the e-fuse completely unusable, and it would need to be replaced. This current limit (I_{LIM}) can be in excess of 150 A.

There is no need to interrupt the current passing through the e-fuse; it is enough to limit it. This can be done by changing the operating point of the power transistor. It is possible to change the gate voltage so that the transistor $R_{DS(on)}$ resistance increases. This is another advantage of the integrated circuit e-fuse. The increase in on-resistance limits the current that can pass through the e-fuse at the supply voltage (see Ohm's law).

The increased $R_{DS(on)}$ also means larger power loss and, therefore, heating of the e-fuse. If the current is limited for long enough, the e-fuse reaches its maximum temperature, and the overtemperature protection interrupts the current. The increased on-resistance also means a large voltage drop across the e-fuse, meaning the circuit voltage (power for devices protected by the e-fuse) will be lower.

There are various ways to implement this kind of protection. One way is to monitor the current and when it exceeds the I_{LIM} current, change the transistor gate voltage to increase the $R_{DS(ON)}$. This approach has the disadvantage of being fairly slow as the current measurements take time. Another option is to place two diodes into the IC. One close to the power transistor, the other on the edge of the die. When a large current passes through the transistor, it heats up, and so does one of the diodes. By comparing

the forward voltages of these diodes, a temperature gradient can be detected. If it is large enough, the power limitation is triggered.

3.2.4 Additional e-fuse features

Thanks to the e-fuse being an integrated circuit, it is possible to include additional features that extend the use of the device beyond just a fuse. These features can improve the safety and reliability of the power distribution system as well as provide greater control over it.

Modern integrated logic technology allows manufacturers to include complex, computationally demanding features into their devices. The result of this are so-called smart electronic fuses. The inclusion of some communication interface (SPI, I^2C) into the e-fuse gives the user more control and much more information.

Fault diagnostic

At their most basic, e-fuses do not differ much from ordinary melting fuses. They interrupt the current when it exceeds a certain level or when the I^2t rating is exceeded. It is impossible to distinguish the reason for the melting fuse to blow. However, the e-fuse can be equipped with dedicated diagnostic pins or diagnostic registers, which can be accessed through a communication interface.

This can be very useful in case of overloads, which are usually caused by malfunctioning devices. By latching off and then unlatching again, the e-fuse resets the device, which might clear the overload fault. Knowing that there is an overloaded device in the circuit, the controller may disable it, thereby removing the source of the overload. On the other hand, if the cause of the e-fuse latching off is a short-circuit, it is usually caused by wiring damage. The retry strategy can be defined by the car manufacturer to suit the application.

Open load detection

By monitoring the various currents and voltages available, the e-fuse becomes a source of valuable data. This data can be used to diagnose the state of the circuit. Open load detection (OLD) can detect if the load has been disconnected from the e-fuse. This mechanism is often implemented in high side switches and drivers. There are two kinds of open load detection. Off-state (offline) when the switch is open and On-state (online) when the switch is closed.

On-state open load detection monitors the current passing through the device. When it falls below the open load current threshold I_{OLD}, it is evaluated as an open load state. Off-state open load detection works as indicated in the schematic in figure 3.10. When the power transistor is off, a source of small diagnostic current (I_{DIAG}) is connected to the output. The control logic senses the power supply voltage V_{BAT} and the output voltage V_{OUT}. If the load is connected (R_{LOAD} is small), the diagnostic current is not high enough to cause a large enough voltage drop across the load. However, if the load is disconnected (R_{LOAD} is high), the output of the device is pulled high to the supply voltage V_{BAT}. This voltage can be measured by the control circuitry. The current source could be replaced by a resistor, but in an IC, it is simpler to use a current source.

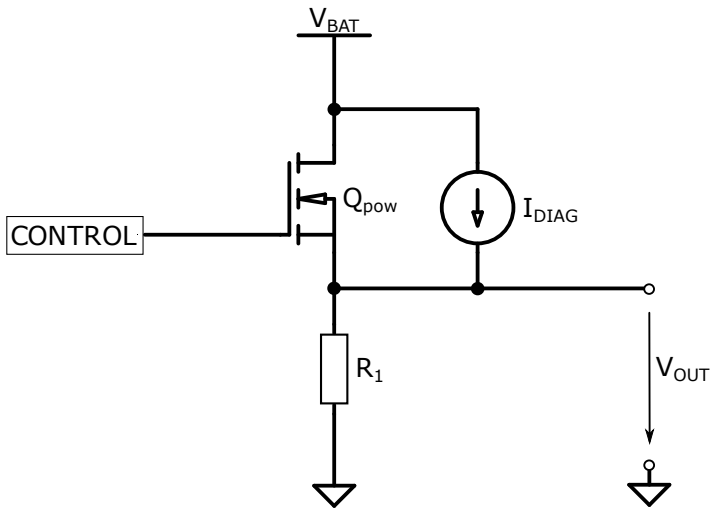


Figure 3.10: Open load detection mechanism

E-fuse parameter programmability

E-fuse properties are defined by the control logic. The logic of the e-fuse can be programmed to fit the application. The e-fuse's breaking current and time characteristics can be defined by setting one or more parameters. These settings can be based on the required I^2t of the e-fuse.

There are various ways of programming the e-fuse. For simpler e-fuses, it can be done by external resistors. If the e-fuse is equipped with a communication interface, SPI for example, it can be programmed by simply setting the values in appropriate registers. The programming can also be done by writing the one time programmable (OTP) memory in the e-fuse. The OTP is programmed when the e-fuse is installed into the system in the factory. Using OTP can reduce the risk of fault due to accidental rewrite of the e-fuse parameters.

The fact that e-fuses can be programmed to a specific application also relieves supply chain issues. Instead of several different types of melting fuses with different parameters, one type of integrated circuit (i.e. an e-fuse) is used and programmed as needed.

Automatic fuse reset

When a melting fuse clears a fault in the circuit, it is destroyed and needs to be replaced. That means that the systems protected by the fuse are inoperable until the fuse is replaced. This is problematic for systems responsible for the safety of the vehicle. To solve this problem in modern cars, there are two independent power supplies for such systems.

When an e-fuse clears a fault, it simply switches off a transistor. This transistor can be switched on again. When an overload in the system is caused by a device malfunction, it is possible that the device malfunction was only temporary, and the reset (because of the power supply loss) fixed the problem. That is the reason why e-fuses come with auto-reset functionality. When the e-fuse latches off, it way for a period of time and then unlatches. This process can be repeated. Most e-fuses have a maximum number of retries before requiring action by a master system.

Health monitoring of the wiring

Health monitoring of the wiring harness is a very advanced functionality proposed for the next generation of electronic fuses. These e-fuses operate with a model of the power distribution grid. Using the model of the wiring and the measurements of the current passing through it, they are able to estimate the temperature of the wiring. From this, it is possible to estimate the state of health of the wiring. This can be used to determine the remaining useful life (RUL) of the wiring.

Normally, the wiring is replaced only when it is damaged. This is unsatisfactory from the point of safety as the wiring could fail at a critical moment (i.e. emergency braking). When a reliable estimate of the remaining useful life is known, predictive maintenance can be applied. When the wiring is nearing the end of its RUL, it is replaced during regular service.

Capacitive charging mode

Modern cars are equipped with a large number of electronic devices which need to be powered. These electronic devices draw current in short pulses, which can be so large that the power supply can not deliver them, or they can be limited by the impedance of the power lines. This is why electronic devices require decoupling (or bypass) capacitors that are capable of delivering short high current pulses. Through the combination of many devices in the circuit or one device with very large capacitance, the capacity that needs to be charged when the circuit is turned on can become very large, in the range of millifarads. A car's battery is an excellent voltage source, and when a capacitor is connected to it, the capacitor will start to charge to the battery's voltage. The charging current is defined by the circuit resistance, battery voltage and capacitor voltage. The circuit resistance is a combination of factors: battery internal resistance, wiring resistance and the equivalent series resistance (ESR) of the capacitor. The blocking capacitors tend to have low ESR values, as that makes them more efficient and reliable. That means that the capacitor charging currents can become very large (in excess of 100 A), which could be damaging to the circuit. Capacitive charging tries to solve this problem by limiting the RMS value of the current to a manageable level. Meaning the capacitor will take longer to charge (milliseconds instead of microseconds), but the current will not damage the e-fuse or the circuit. One way of limiting the current is to use a pulse width modulation (PWM) during turning on of the e-fuse, which will charge the capacitor a little bit every time the e-fuse turns on, limiting the time the current is flowing, thus limiting the RMS value.

3.2.5 E-fuse vs Melting fuse comparison

Now that the electronic fuse has been introduced and the application of fuses in automotive has been discussed, a comparison between the electronic fuse and the melting fuse can be made. In this section, the advantages and disadvantages of using e-fuses instead of melting fuses will be discussed.

For these reasons and many more, the automotive industry has started to adopt electronic fuses instead of melting fuses. The introduction of e-fuses brings a reduction of costs during manufacturing. They also help to reduce the weight of the car (slightly), which in turn helps to reduce fuel consumption and CO_2 emissions. E-fuses also help manufacturers to keep up with the increasing demands on reliability and safety.

Reusability of electronic fuses

The obvious advantage of e-fuse over melting fuses is the reusability of the e-fuses. When a melting fuse clears an overcurrent event, it needs to be replaced. The e-fuse just needs to be switched back on. Since e-fuses are not user-replaceable, there is no need to place them somewhere accessible, like a fuse box. An e-fuse can be integrated directly into the system it is meant to protect, or it can be a part of a large power distribution module. The absence of a fuse box means more space for other systems and a lower overall weight of the car. The reusability of e-fuses also improves the safety and reliability of the car.

Improved accuracy of fault detection

As was already mentioned, the e-fuse relies on measuring the current, rather than on the effect of the current on the fuse, to protect the circuit from overcurrent. This helps to improve the accuracy of the fuse. Melting fuses have quite large manufacturing tolerance, which has a negative influence on the variance of their parameters. Integrated circuits are manufactured with much smaller tolerances. If the current measurement is done correctly, the e-fuse can react more accurately, which can lead to reduction of diameter of wiring connected to the e-fuse. This can, in turn, reduce weight, cost and emissions. The same applies to reaction times. The e-fuse has built-in logic that will react to overcurrent as fast as specified.

Variability e-fuse parameters

The fact that e-fuses are programmable also increases their variability. The designers of the power distribution system are no longer limited by the range of available fuses. They are able to define the e-fuse behaviour exactly as is required for the protected circuit.

Furthermore, some e-fuses can be reprogrammed during operation (those with a communication interface), which allows the system to change the e-fuse properties during an emergency (limp-home mode). One scenario when this can be used is when a vital system malfunctions and requires more power to continue working. The power distribution system may allow temporary overload of the circuit by changing the nominal current rating in order to keep the system working.

Control and monitoring

By controlling the logic of the e-fuse, it is possible to control the power MOSFET. With this, the e-fuse can be used as a high side switch. This allows the power grid controller to connect and disconnect the loads protected by the fuse. The monitoring of various voltages and currents provides a lot of information about the circuit. By combining these features, the e-fuse improves the management of the power distribution system.

Price and life-time

One of the disadvantages of electronic fuses is price. E-fuses are more expensive than melting fuses, due to the fact that they are complex electronic devices. The price

difference is however offset by savings in other areas, such as the reduced diameter of wiring.

Both melting and electronic fuses are designed for a specific life-time. Melting fuses have shorter life-times, due to the fact that the mechanical stresses caused by thermal cycling, weaken their structure. Electronic fuses are designed for the same life-time as most automotive electronics (around 80000 hours), which is much longer than a life-time of a melting fuse.

Failure conditions

When discussing functional safety of a device, its failure states need to be considered. This is a problematic topic when it comes to electronic fuses. When an electronic fuse fails (burns out), there are two failure modes of its power switch: fail open and fail closed. In the event the e-fuse fails open, the circuit remains protected, but nonfunctional. In case the e-fuse fails closed, the current can pass through it uncontrollably and the circuit is not protected. Currently, there is no way to guarantee that the switch fails open. This safety problem can be solved by using one central melting fuse to prevent catastrophic overcurrents. Another option is to use two e-fuses in parallel; therefore, if one fails open, the other can still interrupt the current.

Chapter 4
Emulator design requirements

Before starting to work on the design and implementation of the emulator, the parameters of the emulator need to be defined. The purpose of the emulator is to emulate the functions of a future device for the purpose of evaluating the additional features. This means that the specification of the emulator will be based on the specification of the future device. Unfortunately, not every feature can be implemented in the emulator, and some may be implemented with limitations. In this chapter, the specification of the future e-fuse will be analysed, and the possibility of their emulation will be discussed.

The emulated e-fuse device, called VNF9D1M5, is a dual-channel e-fuse with advanced features. This device is based on an already existing device VNF7000AY, which is a single channel e-fuse device. VNF7000AY does not have a serial communication interface. It is equipped with control and diagnostic pins. The parameters of the e-fuse are set using external resistors. The advanced features, such as control through SPI, will be emulated using a microcontroller.

The main limitations of the emulator will be related to two factors. First is the fact that the logic implemented using an MCU will be slower than the dedicated logic in the silicon. The second main limitation is the fact that the internal status of the logic already integrated into the VNF7000AY cannot be accessed. A simple block schematic of the emulator is in figure 4.1.

The requirements can be divided into two groups. Hardware requirements and software requirements. Hardware requirements specify the parameters of the device, such as maximum voltage and current. These requirements will be addressed during hardware design. Software requirements describe the required behaviour of the emulator. Some of these will be part of the hardware, but most will require a software solution.

4.1 Emulator hardware requirements

Hardware requirements are based on the specification of the VNF9D1M5. Some of the requirements are also a result of the emulator's purpose as an evaluation platform. These requirements include electrical specification, control and communication specification, analog to digital conversion and many other details. In this section, the most important parameters of the VNF9D1M5 will be evaluated in terms of the possibility of their implementation in the emulator.

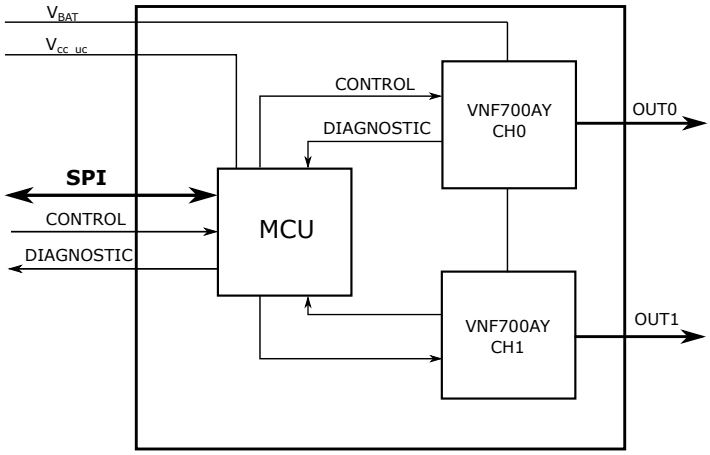


Figure 4.1: The basic structure of the emulator

4.1.1 Required electrical characteristics

Every electronic device has electrical characteristics that include absolute maximum ratings and characteristics of inputs and outputs, and electromagnetic compatibility (EMC). The emulator is required to have the same characteristics as the final device, or at least be as close as possible. Many of the characteristics are related to the power section of the device. Because the power section will be based on the VNF7000AY, which already fulfils most of the requirements, they will be met. For instance, the VNF7000AY is already equipped with a reverse battery protection mechanism, which means that there is no need to implement one in the emulator. Other such characteristics are related to supply voltage ranges (4 V to 28 V).

When it comes to supply voltage, there is a first limitation of the emulator. Because the emulator requires an MCU to perform the additional functions, it also needs a power supply for the MCU. Automotive MCU can work on a 5 V power supply. This voltage can be obtained by including a 5 V LDO (low drop-out voltage regulator). Including a regulator would increase the complexity and the size of the emulator as well as the power losses. The emulator will therefore require an external power supply $V_{DD} = 5\,\mathrm{V}$ for its logic. One characteristic which will have a significant influence on the design of the emulator is the maximum current rating. Each channel (out of two) needs to be able to carry 32 A of steady current. This means that the total current from the power supply can reach up to 64 A continuously. During overcurrents, these overloads can be much higher. As a result, great care needs to be taken when designing the PCB of the emulator. The traces carrying the high currents need to withstand them, and sufficient cooling of the VNF7000AY devices needs to be provided. The typical on-state resistance of the power transistor in the e-fuse is $R_{ON} = 1.5\,\mathrm{m\Omega}$. This means that the power loss on one e-fuse at maximum steady-state current of $I_{NOM} = 32\,\mathrm{A}$ is

$$P_{loss} = I_{NOM}^2 \cdot R_{ON} = 32\,\mathrm{A}^2 \cdot 1.5\,\mathrm{m\Omega} \doteq 1.5\,\mathrm{W}. \tag{4.1}$$

That is a significant amount of heat that will need to be dissipated by the PCB.

The current consumption of the device is another key characteristic. The VNF9D1M5 has maximum supply current in an active mode of 4 mA, and in standby mode, it is only 0.8 µA (at 25 °C) [5]. These kinds of currents can not even be approached with the emulator. The VNF7000AYs consume typically 4 mA each in active mode [3] with outputs turned off. The most significant part of the current consumption of the emu-

lator comes from the MCU, whose current consumption can reach 90 mA [6] and more when running at maximum frequency and peripherals are active. The consumption can be lowered in a standby mode of the emulator by switching the MCU into a low power mode. In these modes, the current consumption can be as low as 100 µA, but they have some drawbacks, such as long wake-up times (around 200 µs). However, it will not be possible to reach the performance of the real device, and the emulator's power consumption will need to be taken into account when integrating it into a power distribution system.

4.1.2 Communication and control interface

The emulator needs to have the same signal connections as the real device, meaning that the pins of the device need to have an equivalent on the emulator. This will be done using various connectors. There are two pins that will not be implemented in the emulator. These are the CP and the VREG pins. The CP pin is used to drive a gate of external reverse battery protection MOSFET. This feature is not required for the emulator. The VREG pin provides an output of the internal regulator of VNF9D1M5 [5]. Since the emulator does not include this regulator, implementation of this pin is not possible.

The fact that the first version is intended as an evaluation platform means that it is also required to allow access to all of the signals. To that end, there will be several headers that will serve as test points for the various signals. The emulator will be controlled through direct signals and through SPI. The master system needs to be able to connect to it. That will be done using a universal connector.

The power section requires robust connectors that will be able to handle the large currents going through the e-fuses. These connectors will need to have very low resistance, and their connection to the PCB will have to be low resistance as well. It is also required to be able to easily connect laboratory power supplies and loads to the emulator during evaluation.

4.1.3 Analog to digital conversion of voltages and currents

To perform their function, the e-fuses need to monitor the current passing through them. They are also able to monitor battery and output voltage. In the VNF7000AY, all of this is done through analog circuits. There is also the MultiSense pin which provides analog feedback of all of these values and of the e-fuse's temperature. The VNF9D1M5 provide these measurements digitally through an SPI interface. This means that the analog values need to be converted to digital. For that, an analog-to-digital converter (ADC) integrated into the MCU will be used.

VNF9D1M5 has an internal 10-bit ADC, and its parameters are specified in the datasheet. The MCU ADC will need to be configured to match the e-fuse's parameters. The errors of the measurement, such as integral and differential linearity errors, will be given by the parameters of the ADCs in the MCU. The sampling rate of the ADCs will be given by the capabilities of the hardware and the implementation of software.

4.1.4 Setting the e-fuse parameters via SPI

VNF7000AY uses external resistors to set the parameters of the VIP-Fuse protection. Resistor designated R_{FCS} to set the nominal current I_{NOM} and resistor desig-

nated R_{FTS} to set the nominal time t_{NOM} [3]. VNF9D1M5 contains the same protection mechanism, but it is programmed by writing a register via SPI. The emulator will need to be able to take the value received through SPI and set the corresponding parameters on one of the VNF7000AYs.

Digital potentiometers or rheostats will be used to set the appropriate resistance. The MCU will process the value received through SPI and set the potentiometer acting as either R_{FTS} or R_{FCS}. These digital potentiometers are controlled through a communication interface such as SPI or I^2C.

4.2 Emulator software requirements

Software for the microcontroller (the firmware) will define the behaviour of the emulator. It is the key part of the project because the MCU realises all of the new features of the e-fuse. The firmware (FW) also partially defines the hardware properties as it is used to configure the peripherals of the MCU. In this section, the main FW building blocks and requirements will be discussed.

4.2.1 VNF9D1M5 state machine

The behaviour of the VNF9D1M5 can be described using a finite state machine. Finite state machines operate in only one state at any given time and transition between these states in reaction to external signals. The best way to describe them is to use a state diagram. It is an oriented graph, where the nodes represent the individual states and the edges represent the transitions. Transition conditions are next to the edges representing the transitions. The state diagram of VNF9D1M5 can be found in its datasheet [5].

The state machine can be implemented in software, and that is exactly what is going to be done. Due to the limitations of the emulator, one state and some of the transitions will be left out. The state to be left out is the Prestandby mode. It is essentially identical to the Standby mode. It waits for the I^2t protection counters to reach 0. Unfortunately, the emulator does not have access to these counters. This state will be merged with the Standby mode state in the emulator's state machine. The transitions that are caused by regulator voltage V_{REG} falling below reset voltage V_{POR} will be left out as well because there is no regulator to provide the V_{REG} voltage.

4.2.2 Communication and control interface

Primarily, control of the VNF9D1M5 is done through a 24-bit SPI interface. In the emulator, this will be done by using the MCUs SPI peripheral in slave mode. The format of the SPI frame is in table 4.1. The opcode controls what operation is to be performed (read, write, ...), the address indicates the target register, and the remaining sixteen bits are the data. The problem is, that the device needs to respond based on the information in the first byte (opcode and address). In the MCU, the registers will be realised as an array in memory.

When the address byte is being received by the VNF9D1M5, it transmits the Global Status Byte (GSB) back at the same time. When data bytes are transmitted by the master, the slave transmits the contents of the accessed register. The emulator needs to load the data to be transmitted into the SPI transmit register before the data

transmission from the master starts (in between the first and second byte). The MCU will need some time (a few µs) to prepare the data. This needs to be taken into account when designing the master software, which will control the emulator. The development of this software is not a part of this thesis, but it will be developed alongside the emulator. Inside the real device, the SPI module is directly connected to the status and control register. Therefore it does not need any additional time to process the address and prepare the response.

MSB=23	22	21	20	19	18	17	16
OC1	OC0	A5	A4	A3	A2	A1	A0
15	14	13	12	11	10	9	8
D15	D14	D13	D12	D11	D10	D9	D8
7	6	5	4	3	2	1	LSB=0
D7	D6	D5	D4	D3	D2	D1	D0

Table 4.1: SPI frame used in VNF9D1M5. OCx - opcode, Ax - address, Dx - data [5]

4.2.3 Reaction times to control signals

Reaction times, especially switching delays, are an important part of the e-fuses safety. Switching delays and other timings are a part of the electrical characteristic of the e-fuse. In the emulator, these delays will consist of two parts. The first part are the delays of the VNF7000AY, which are specified in the datasheet [3]. The second part of the delay is the contribution of the MCU. When the microcontroller receives an external signal (or SPI message), it needs to process it before sending a signal to the VNF7000AY.

To make the delays caused by the MCU as short as possible, the signals requiring fast reaction times will be handled as interrupts. This approach will also greatly reduce the chances of the MCU not reacting to the signal. Even this will not guarantee that the emulator will have the same reaction times as the real device. The delays will always be slightly longer on the emulator, which will have to be taken into account when using the emulator.

Chapter 5
Design of Emulator Hardware

The development of the emulator involves hardware design as well as software design. The hardware design includes two main tasks. Component selection and design of the printed circuit board (PCB). This chapter documents the design process and the final implementation of the device hardware. The general requirements and guidelines for automotive electronics in [7] were followed during this process.

5.1 Selected components

Before starting with the design of the PCB, the hardware requirements need to be examined and appropriate components selected. This section contains an overview of the main components and presents the reasoning behind their selection.

5.1.1 Basic e-fuse device

The component upon which the emulator is based is an already existing e-fuse. The e-fuse to be used has been specified as a requirement on the emulator. It is the VNF7000AY [3]. VNF7000AY is a high-side driver with analog feedback and embedded VIP-Fuse protection intended for automotive applications. It contains all of the e-fuse functionality required for the emulator. Namely, it has short circuit and overload protection (VIP-Fuse), undervoltage and overvoltage detection and overtemperature protection.

The VNF7000AY uses a PowerSSO-36 package, which you can see in figure 5.1. This package allows the VNF7000AY to dissipate a significant amount of heat into the PCB under it and to handle the large currents passing through it. It has a large exposed pad (marked as TAB/Vcc) at the bottom which is connected to the V_{BAT} power supply. The exposed pad allows large current carrying capacity and better transfer of heat from the IC to the PCB. The output of the device is connected to 24 pins to accommodate the large output currents. The complete pinout of the device is in figure 5.2. Function description of the individual pins can be found in the datasheet [3]. The control pins are compatible with both 5 V and 3.3 V signals.

The key feature of the e-fuse functionality is the VIP-Fuse protection mechanism. This function is designed to provide flexible protection to the wiring harness and other parts of the power distribution grid (PCB traces, connectors,...). It uses an I^2t curve, which is used to define conditions under which the e-fuses latches off. The normalised version of this curve is in figure 5.3, and it is defined by the coefficients in table 5.1. K_t is the multiplier of nominal time t_{NOM} and K_I is the multiplier of nominal current

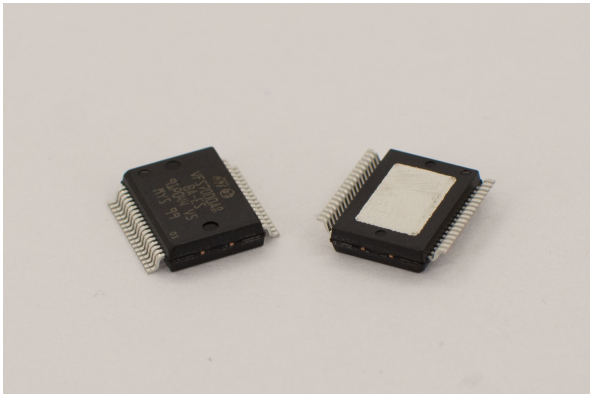


Figure 5.1: PowerSSO-36 package

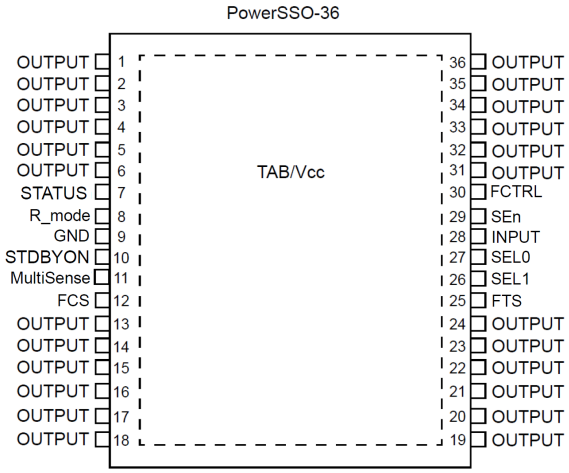


Figure 5.2: VNF7000AY pinout [3]

I_{NOM}. The area coloured in with blue denotes the time-current values for which the e-fuse remains on, and the red area is where the e-fuse latches off. For example, if the current is $5 \times I_{NOM}$, the e-fuses interrupt the circuit in $t_{NOM} \times 10^{-3}$. The values of I_{NOM} and t_{NOM} can be set using external resistors. Nominal current is set by the Fuse Current Set resistor R_{FCS} and nominal time is set by the Fuse Time Set resistor R_{FTS}.

The device also includes a multipurpose analog feedback signal called MultiSense, which provides information about the current passing through the e-fuse, the voltage of the power supply and the package temperature. MultiSense needs to be enabled by pulling the SenseEnable (Sen) pin and the information that is on the output pin is switched using the pins SEL0 and SEL1, that set up the internal multiplexer. When current monitoring is selected, the MultiSense pin outputs a current I_{SENSE} proportional (1 : 100000) to the fuse current. A current sense resistor is then used to produce voltage that can be measured by the MCU. When the output voltage feedback is selected, the MultiSense produces voltage V_{SENSE_VCC} proportional (1:8) to the battery voltage. When package temperature is selected, the MultiSense pin outputs a voltage V_{SENSE_TC} proportional to the temperature of the device package.

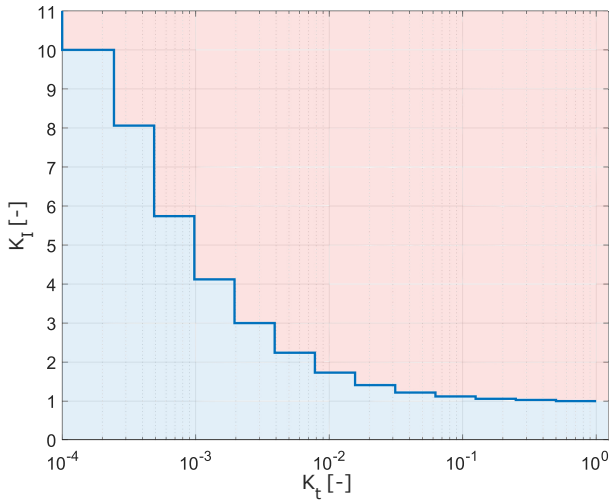


Figure 5.3: Nominal I^2t curve of the VIP-Fuse protection

K_I	1	1.03	1.06	1.12	1.22	1.41	1.73	2.24	3.00	4.12	5.74	8.06	10
K_t	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}	2^{11}	0

Table 5.1: Coefficients defining the shape of the nominal I^2t curve

5.1.2 Microcontroller used for emulation

Part of the assignment is the requirement to use an automotive-grade microcontroller. This requirement was made to make the entire emulator usable in a car during evaluations. As the emulation is not very computationally demanding, a low-end automotive MCU was chosen, the **SPC582B60E1** general-purpose microcontroller from the Chorus family. It is a single-core 32-bit MCU built on PowerPC architecture with 1MB of flash memory and 96kB of RAM. Because of the amount of flash, it is sometimes called "Chorus 1M". The maximum core clock is 80 MHz. The MCU contains many peripherals, of which the following are essential for the emulator; one 12-bit analog to digital converter, four deserial serial peripheral interfaces (DSPI), and an interrupt controller. The MCU can operate with either a 5 V or a 3.3 V power supply. The emulator uses 5 V for the logic part.

The SPC582B60E1 is manufactured in three different packages:

- QFN48 with 48 pins
- eTQFP65 with 64 pins
- eTQFP100 with 100 pins

An appropriate package needs to be chosen, with regard to the emulator requirements, such as the number of signals that need to be handled by the MCU or the maximum size of the PCB. The first prototypes of the emulator will be soldered by hand. This makes the QFN48 package unsuitable for the emulator as it is relatively difficult to solder by hand. The decision between the remaining two packages comes down to their size and the number of pins. The 64-pin package (eTQFP64) was chosen for a start because it has a smaller footprint. By assigning every required signal to a pin of the MCU, it was determined that there were enough pins on the 64-pin package. Out of the 64 available pins, only three pins remained unused. For the pinout, see section 5.2.

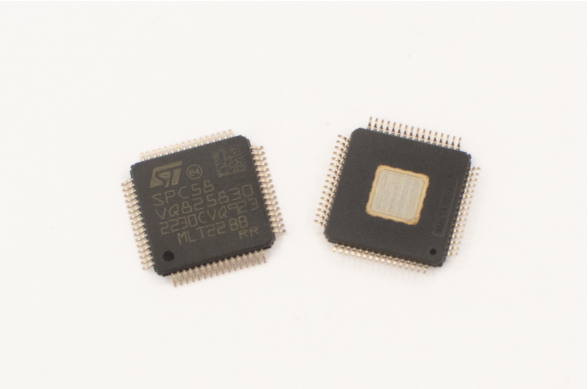


Figure 5.4: Emulation microcontroller - SPC582B60E1

5.1.3 Digital potentiometers for e-fuse parameter setting

The VIP-Fuse protection integrated into VNF7000AY can be programmed using external resistors. In the VNF9D1M5, the VIP-Fuse protection is programmed through SPI. This means that the emulator needs to be able to set R_{FCS} and R_{FTS} resistance to values corresponding to the settings in the registers. The ideal solution is to use digital potentiometers or rheostats. Although there are exceptions, most digital potentiometers are controlled via SPI or I^2C. The selected MCU has several SPI peripheral, which is why an SPI controlled digital potentiometer is required.

According to the VNF7000AY datasheet, the t_{NOM} is set by passing a defined current $I_{FTS} = 50\,\mu A$ through the R_{FTS} resistor and measuring the voltage across the resistor. The voltage levels and corresponding times are in table 5.2. To select an appropriate digital potentiometer, the maximum required resistance $R_{FTS_{MAX}}$ needs to be determined.

$$R_{FTS_{MAX}} = \frac{V_{FTS_{MAX}}}{I_{FTS}} = \frac{5.4\,V}{50\,\mu A} = 108\,k\Omega \tag{5.1}$$

However, most digital potentiometers currently on the market have a maximum resistance of $100\,k\Omega$. Luckily, this is sufficient as $R_{FTS} = 100\,k\Omega$ will produces $V_{FTS} = 5\,V$ which is still well within the range for $t_{NOM} = 1\,s$.

The VNF7000AY passes a current I_{FCS} proportional to the load current I_{LOAD} through the R_{FCS} resistor. The VIP-Fuse mechanism then senses the voltage V_{FCS} across this resistor and compares in to the reference $V_{ref0} = 475\,mV$. When nominal current flows through the e-fuse, V_{FCS} equals to V_{ref0}. Using this knowledge, a relation between R_{FCS} and I_{NOM} can be written as:

$$R_{FCS} = \frac{V_{ref0}}{I_{NOM}/K_F}, \tag{5.2}$$

where $K_F = I_{NOM}/I_{FCS} = 10000$ [3]. VNF9D1M5 can set eight different values of nominal current, which are 9 A, 11 A, 14 A, 16 A, 19 A, 25 A, 27 A and 32 A. The digital potentiometer needs to be able to provide a large enough resistance to set the lowest nominal current $I_{NOM_{MIN}} = 9\,A$. Therefore the maximum required FCS resistor is

$$R_{FCS_{MAX}} = \frac{475\,mV}{9\,A/10^5} \doteq 5278\,\Omega. \tag{5.3}$$

t_{NOM}	V_{FTS}
300 s	0.0 V to 0.5 V
257 s	0.8 V to 1.2 V
214 s	1.5 V to 1.9 V
172 s	2.2 V to 2.6 V
129 s	2.9 V to 3.3 V
86 s	3.6 V to 4.0 V
44 s	4.3 V to 4.7 V
1 s	5.0 V to 5.4 V

Table 5.2: VNF7000AY nominal time settings and corresponding voltage ranges

The best fitting commonly available value of maximum potentiometer resistance is 10 kΩ.

Each of the two channels of the emulator needs to be able to set the nominal current and time separately. This requires four separate potentiometers. Using four individual digital potentiometers would be impractical as it would complicate the PCB. That is why two-channel potentiometers were chosen.

Based on the analysis of the requirements presented above, the AD5162 Dual Channel Digital Potentiometer was chosen. It is a two-channel potentiometer with 256 positions controlled via SPI. One of the reasons for choosing this potentiometer was the fact that it has an operating temperature range from −40 °C to 125 °C, which is suitable for automotive applications. It is also manufactured in both required resistance ranges of 10 kΩ and 100 kΩ. The block diagram and pinout of this component is in figure 5.5. As you can see, the second 'potentiometer' is strictly speaking a rheostat but it does not matter for the emulator. AD5162 potentiometers are available in the MSOP-10 package, which is a relatively small 10-pin package.

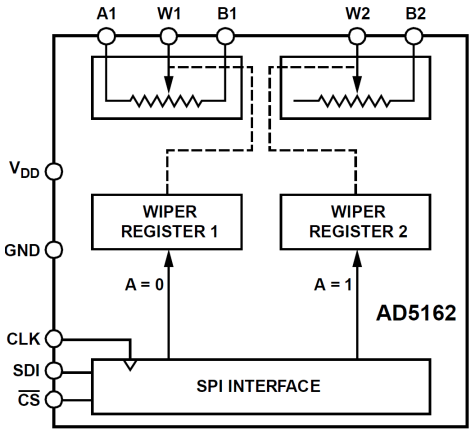


Figure 5.5: Block schematic and pinout of the AD5162 digital potentiometer [8]

5.1.4 Power and signal connectors

The emulator needs to be able to connect to the power supply and the loads that are connected to its outputs. Each channel of the emulator is required to provide up to 32 A of current indefinitely, meaning that the emulator might need to pull up to 64 A

from the power supply. These currents are relatively high, and there are not many PCB connectors rated for them. There is also the fact that the emulator is meant as an evaluation platform, and therefore the connectors need to be easily connected and disconnected. This leads to the selection of the REDCUBE Direct Plug Terminal by Würth Elektronik [9]. An image of these connectors is in figure 5.6.

The REDCUBE Plugs can handle currents up to 120 A, which is more than enough for the emulator. The contact resistance of these connectors is excellent; the maximum is only 1 mΩ. They also offer a very good connection to the PCB. It is achieved by using Press-fit technology. During this process, the connector is not soldered to the PCB, but it is pressed into it, which produces direct copper to copper contact. This produces much lower connection resistance because copper has much better conductivity than solder. The temperature rating of the REDCUBE Plug is up to 125 °C, which matches the requirements of the emulator. The connectors also come in three different colours (red, black and blue), which helps to differentiate positive and ground connections.

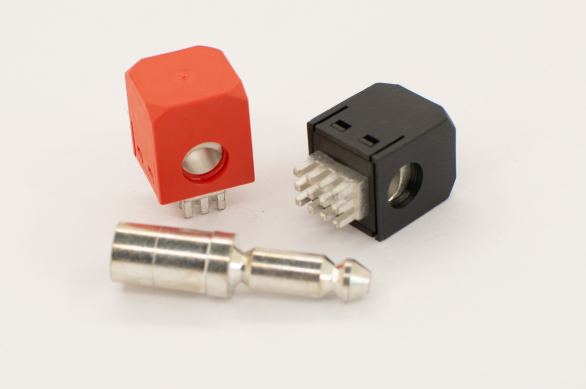


Figure 5.6: REDCUBE Plug connectors and plugs

Aside from the power connections the emulator needs to be connected to a controller. For evaluation purposes, a universal evaluation platform will be used - Chorus 1M Tiny Board. This board has a large 48-pin female connector which can be used to interface with the MCU on the Tiny Board. A matching 48-pin male connector will be used on the emulator. This connector can be seen in figure 5.7. This connection will provide all of the control and communication signals as well as the 5 V supply for the emulator microcontroller. The emulator MCU is programmed and debugged through JTAG, which means that a JTAG connector needs to be on the board. A generic 10-pin (5x2) header with 1.27 mm spacing was used, because it is compatible with the available debugger.

5.2 Emulator PCB design and realisation

The emulator needs to be assembled from the selected components. To connect these components, a printed circuit board (PCB) needs to be designed. The design of the PCB will be influenced by the requirements, especially the power section. During the design, the rules and guidelines in [10] will be followed. In the following section, the design of the PCB will be documented.

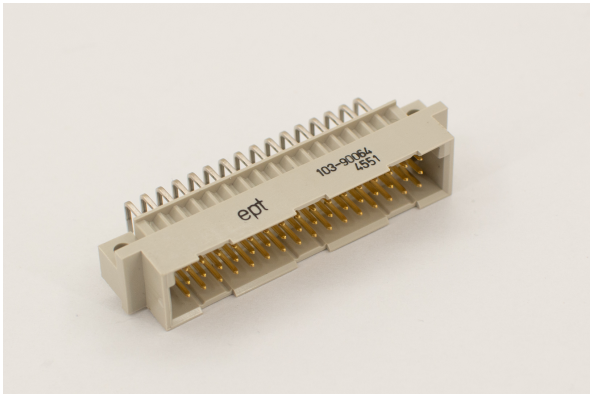


Figure 5.7: Universal 48-pin board to board connector

5.2.1 Design of the emulator's electrical schematic

The design of every PCB starts with the schematic. During schematic design, the connections between the individual components are defined. It is also the stage at which the parameters of passive components are determined. During the design of the schematic, reference schematics of the used components were consulted, and most of them were used. The complete schematic can be found in the appendix B.

The most complicated part of the schematic design was the definition of the pinout of the microcontroller. An overview of the functions of the individual MCU pins can be found in the IO definition table in [11]. Each signal was assigned to the selected pin until every signal was assigned. The resulting pinout is in figure 5.8, where supply pins are marked in red, assigned signal pins in blue and free pins are in green. Another important part concerning the MCU is the power supply decoupling and external ballast. In the emulator, the reference design from the MCU's hardware design guidelines [12] was used, although there were some minor modifications. The most notable modification is the addition of a 100 µF electrolytic capacitor C_{29}, which is intended to provide better filtering of large fluctuations in the $V_{CC\_UC} = 5\,\mathrm{V}$ power supply. The ballast's schematic can be seen in figure 5.9.

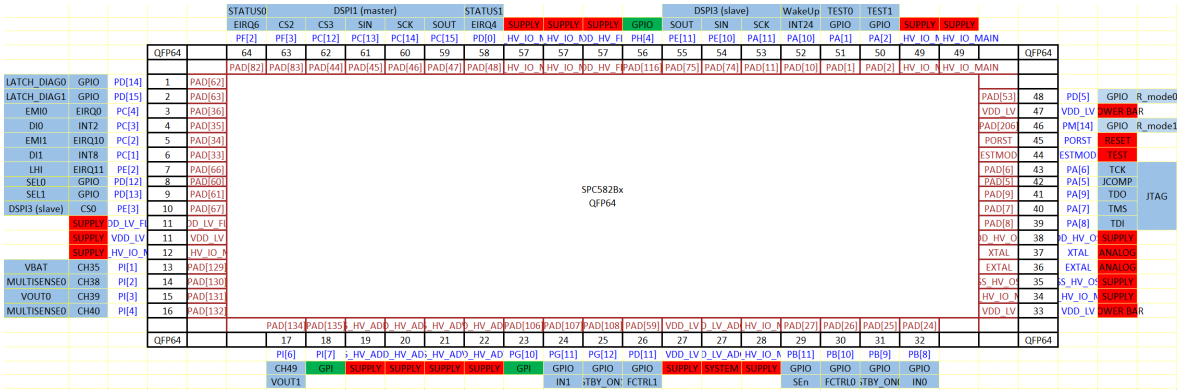


Figure 5.8: Signal assignment to the pins of the emulator MCU

Protection resistors were used to protect the microcontroller and other devices on the emulator against voltage transients. For regular control signals 15 kΩ resistors were used. For data signals, which require sharp edges, such as SPI and JTAG, only 1 kΩ resistors were used in order to maintain signal integrity. Another form of protection

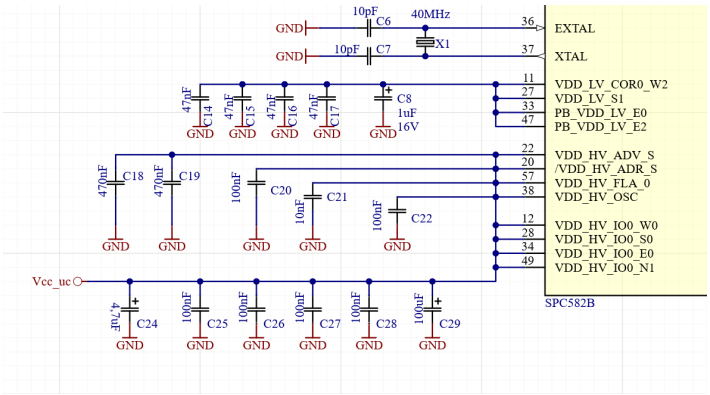


Figure 5.9: Microcontroller ballast schematic

are the transil and the reverse battery diode. The reverse battery diode (D_1) is placed between the board to board connector P_1 and the V_{CC_UC} power rail in order to protect the microcontroller and the digital potentiometers. The transil or a transient-voltage-suppression (TVS) diode TR_1 protects the VNF7000AYs against overvoltage on the V_{BAT} line. Indication LEDs were added to the most important signals to provide quick visual feedback. Green LEDs were added to the outputs, the V_{BAT} supply and the V_{CC_UC}. Orange LEDs were added to signals that indicate faults, which are the LATCH_DIAG signals and the FS signal. Test points were added to all of the internal and external emulator signals in order to make them easily accessible during the development of the emulator.

The R_{FTS} potentiometer has maximum resistance of 100 kΩ, which can produce maximum V_{FTS} voltage of 5 V. This is at the bottom of the range for the $t_{NOM} = 1$ s setting. A manufacturing tolerance could cause the set resistance to be lower than expected, causing the V_{FTS} voltage to fall below the 5 V threshold. For this reason, a series resistor has been added to each FTS connection between the potentiometers and the VNF7000AYs. Using these resistors, the R_{FTS} can be offset, causing an offset in the V_{FTS} voltage. In the schematic in figure 5.10 these offset resistors are marked R_{66} and R_{67}.

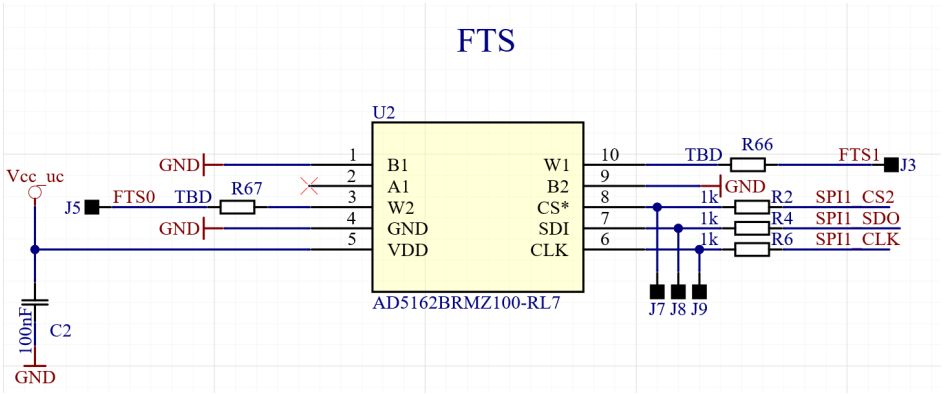


Figure 5.10: Schematic of the FTS potentiometer with offset resistors

5.2.2 Definition of physical properties of the PCB

Before starting with the layout of the PCB the physical properties need to be defined, along with the board stackup and design rules. There are no strict requirements on the PCB size, but it is required to be relatively compact. Based on the dimensions of the individual components and prior experience, the initial board size was set to 8 cm by 10 cm. The board stackup defines the number and the properties of the PCB layers. The power section of the emulator is required to carry currents up to 64 A, which means the connections on the PCB need to be designed to handle these currents. The current carrying capacity of a PCB trace is defined by its cross-section, which is a product of the trace width and thickness of the copper. Copper thickness for general application PCBs is usually 35 µm. At this copper thickness, the external traces would need to be at least 27 mm wide with allowed 75 °C temperature rise according to an online calculator [1]. Internal layers would need to be even thicker, at least 71 mm. These trace widths are impractical, which means the copper layers need to be thicker. Copper layer thickness selection depends on the capabilities of the manufacturer, but they usually offer at least 105 µm. Copper layer thickness also influences the minimum trace width (due to underetching), which is important for traces carrying signals connected to the MCU. If the traces connecting to the MCU are too wide, there may not be enough space to connect them all. Copper layer thickness that is high enough to allow large currents and at the same time thin enough traces needs to be selected. The chosen copper thickness is 70 µm.

Another way to improve the power connections is to use multiple traces in parallel. This is achieved by using a four-layer PCB. Adding another two layers also improved the thermal performance of the board, as it greatly improved the power dissipation from the high power components. Four-layer PCB also provides better electromagnetic compatibility (EMC), if properly designed, than two-layer ones. An illustration of the final PCB stackup is in figure 5.11. The final thickness of the board is 1.5 mm.

The design rules for the board, such as trace width, minimal isolation distance and minimal via diameter, are for the most part dictated by the capabilities of the manufacturer. In the case of the emulator, the chosen manufacturer is Pragoboard. Their recommendations and limits for 70 µm copper layers are following:

- Minimal trace width - 0.3 mm
- Minimal isolation distance - 0.2 mm
- Minimal via diameter - 0.15 mm

The minimal trace width and isolation distance are not limits of the manufacturer, but only recommendations with reference to the chosen copper thickness. These values were used during the design of the emulator.

5.2.3 Layout of the emulator's PCB

Once the schematic and the PCB parameters were finalised, the work on the layout of the PCB can start. PCB layout is designed in two stages. The first stage is component placement, where the components are placed on the board. The second stage is routing, where the placed components are interconnected according to the schematic.

[1] https://www.digikey.com/en/resources/conversion-calculators/conversion-calculator-pcb-trace-width

Figure 5.11: Stackup of the emulator PCB

The placement of the components follows the basic structure of the emulator outlined in the bock schematic in figure 4.1. The microcontroller is placed on the left side of the board, where it is far enough from the VNF7000AYs which can produce a lot of heat. The decoupling and ballast capacitors are placed on the bottom of the board close to the MCU. This is done in order to make the traces from the capacitors as short as possible, reducing their inductance. Long traces have large inductance, which creates a large impedance for the pulsed currents required by the microcontroller. The digital potentiometers are placed next to the MCU. The 48-pin board to board connector is placed on the left edge of the board, close to the MCU. Both of the VNF7000AYs are placed on the right side of the board, where they are close to the V_{BAT} connectors. The output connectors are placed at the top and the bottom of the board as close to the e-fuse ICs as possible to reduce the trace length, thus keeping their resistance low. Among all of the components, enough space was left to place the test point pins, which allow the connection of testing equipment. This component spacing also allows easy component replacement in a case one of the components is destroyed or damaged. The final layout can be seen on the 3D render of the device in figures 5.12 and 5.13.

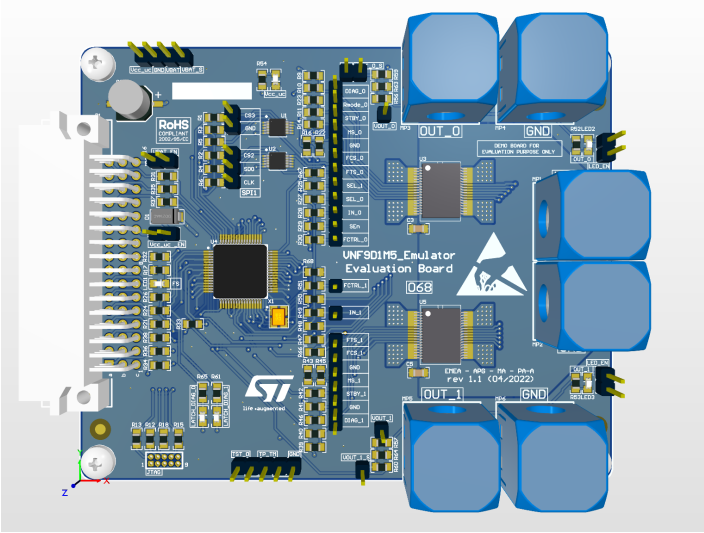


Figure 5.12: Top view of the PCB rendered in the design software

With the placement of the components finished, routing can begin. During routing, the components are connected electrically as defined in the schematic. Outer layers of the PCB are usually used for signal connections and inner layers for power distribution, one for ground (second inner layer) and one for supply voltage (first inner layer connected to V_{CC_UC}). Signals are routed in the inner layers only when necessary. This

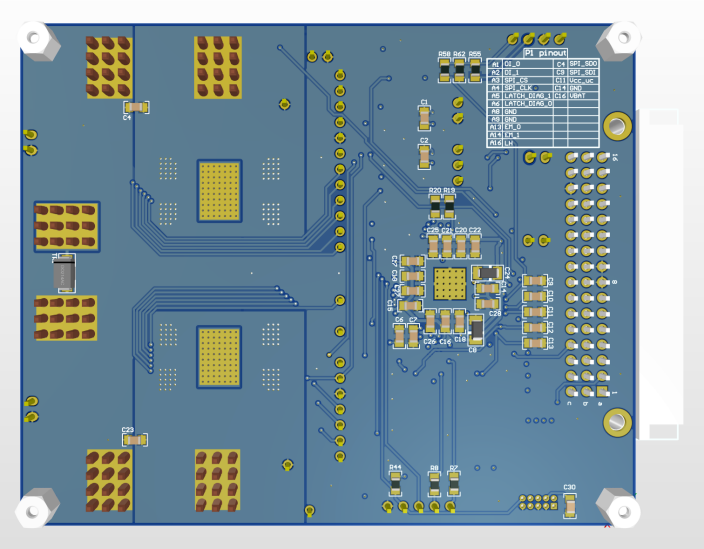
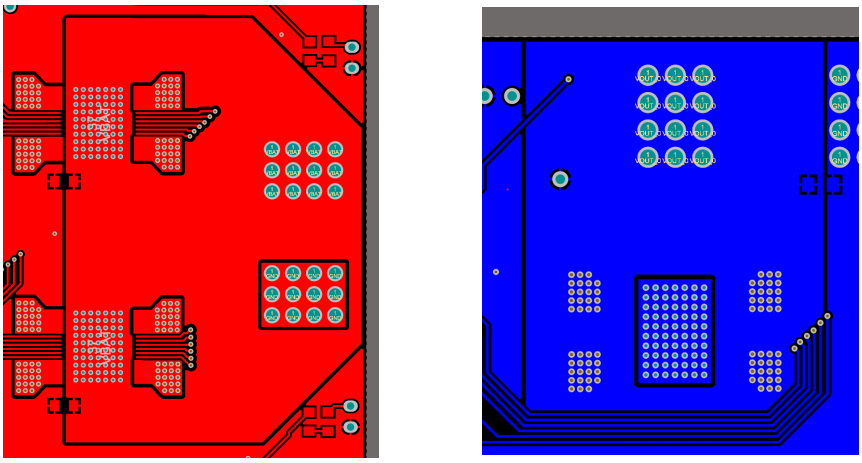


Figure 5.13: Bottom view of the PCB rendered in the design software

approach improves the board's EMC. The requirements on individual connections, such as maximum current through it or the voltage, need to be considered when routing the board. In the emulator, this was especially important when routing the power connection of the two VNF7000AY. The trace widths required to carry the e-fuse currents were too large to route efficiently; hence polygons were used. On the top and the first inner layer, there are polygons that include the V_{BAT} connector and the exposed pads at the bottoms of the VNF7000AYs. The two layers are connected by trough holes vias in the PCB pad for the exposed pad. This allows the current from the inner layer to flow into the device, as well as makes soldering much more straightforward and allows the e-fuse to dissipate its heat into other layers of the PCB. The output connections are routed by polygons in the bottom and the second inner layer. Because the output of VNF7000AY is connected to pins in all four corners of the package (see 5.2), four different polygons had to be made in the top layer to connect them. These top-level polygons were connected to the bottom two layers using vias. The connection polygons for the VNF7000AYs can be seen in figure 5.14. When using vias to carry large currents, great care needs to be taken of their current carrying capacity. When a current passes through a via, it heats up proportionally (Joule's law). Heating up of the via causes it to expand, which can cause the inner PCB layers to tear off from it. This is caused by the fact that the inner layers have a very fragile connection to vias. The ground connection of the outputs was made in a similar manner through copper pours in second inner and bottom layers. All of the other connections were routed with regular 0.3 mm traces. When all of the connections were routed a ground pour was added to the layers, meaning all of the free space was filled with copper connected to the ground (except for the first inner layer). To make sure these ground pours were properly connected to ground, vias connecting all of them were added at regular intervals throughout the board.

Thermal properties of the board needed to be considered during routing, as well as electrical characteristics. It is useless to have perfect electrical connections when the devices overheat when the device is turned on. The heat-generating components need to be identified and appropriate cooling implemented. Most heat is usually generated by components handling large power. That is true in the case of the emulator as well.



(a) Connection to V_{BAT} (b) Connection to V_{OUT0}

Figure 5.14: VNF7000AY connection to the PCB

The following components were identified to be the biggest sources of heat on the board:

- VNF7000AYs e-fuse devices
- REDCUBE Plug power connectors
- SPC582B60E1 microcontroller

The e-fuses are designed for a maximum steady current of $I_{FUSE(MAX)} = 32\,\text{A}$. This large current, combined with the on-state resistance of the e-fuses power transistor, causes a power loss in the e-fuse. According to the datasheet, the $R_{DS(ON)}$ of the e-fuse switch is typically $1.5\,\text{m}\Omega$ and it's maximum value is $3\,\text{m}\Omega$ at junction temperature of $150\,^{\circ}\text{C}$ [3]. Worst case scenario will be considered, meaning that $R_{DS(ON)} = 3\,\text{m}\Omega$ will be used in the calculation of the power loss. Each of the VNF7000AYs has the following power loss:

$$P_{e-fuse} = I^2_{FUSE(MAX)} \cdot R_{DS(ON)} = 32^2 \cdot 3 \times 10^{-3} \doteq 3\,\text{W}. \tag{5.4}$$

The VNF7000AYs package is designed with large power loss in mind, and that is why it includes a large exposed pad on its bottom. This exposed pad is then soldered to the PCB making a good electrical a thermal connection. Large copper areas of the PCB act as a plate heatsink which helps to dissipate the heat to the ambient air. The large currents passing through the e-fuses also pass through the connectors, causing a power loss in them as well. The maximum contact resistance of the REDCUBE Plug connectors is $R_C = 1\,\text{m}\Omega$ [9]. The V_{BAT} connectors can experience up to 64 A of current because they need to supply the current for both of the outputs. The power loss P_{C_VBAT} applies to the VBAT and GND connectors, that connect the board to the V_{BAT} power supply. The P_{C_OUT} applies to the output connectors, the OUT0, OUT1 and both GNDs.

$$P_{C\_VBAT} = (2I_{FUSE(MAX)})^2 \cdot R_C = 64^2 \cdot 10^{-3} \doteq 4\,\text{W} \tag{5.5}$$

$$P_{C\_OUT} = I^2_{FUSE(MAX)} \cdot R_C = 32^2 \cdot 10^{-3} \doteq 1\,\text{W} \tag{5.6}$$

Fortunately, the maximum rated current of these connectors is 120 A which means they are meant to handle power losses this high. However, some of the heat generated by the connectors will be conducted into the board, potentially heating the other components. Another component that generates some amount of heat is the microcontroller. According to its datasheet, it can consume up to $I_{MCU(MAX)} = 90\,\text{mA}$ at room temperature when all of its peripherals are running. Knowing that the microcontroller is powered by $V_{CC\_UC} = 5\,\text{V}$, the power loss on the MCU (P_{MCU})can be determined.

$$P_{MCU} = V_{CC\_UC} \cdot I_{MCU(MAX)} = 5 \cdot 90 \times 10_{-3} = 0.45\,\text{W} \tag{5.7}$$

Half a Watt is not a lot of power, but if the heat was not adequately dissipated, the MCU could overheat, which could, in turn, lead to the emulator malfunctioning. That is why it has a small exposed pad on the bottom of the package, which provides a good ground connection and heat dissipation into the ground planes of the PCB.

To verify that the heat design of the board is good before manufacturing, a thermal simulation was made in PICLS[2] programme. After inserting the Gerber files describing the PCB and the expected power losses on individual components determined above, the simulator provides a heat map of the PCB, which can be found in figure 5.15. In the simulation, the maximum temperature of the VNF7000AYs reached 71 °C, and the hottest components were the V_{BAT} power supply connectors, which reached 95 °C. None of the components on the PCB exceeded their maximum operating temperature, which means that the cooling of the components should be sufficient.

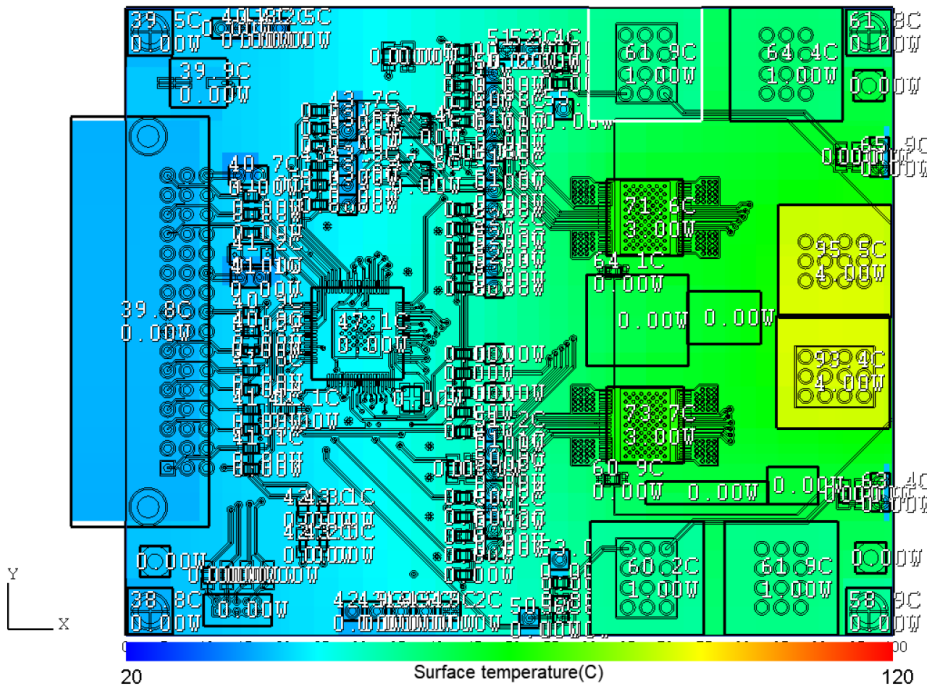


Figure 5.15: PICLS simulation of heat distribution across the emulator PCB

[2] https://www.mscsoftware.com/product/picls

5.3 Final emulator board assembly

With the placement and routing complete, the board manufacturing data was sent to the manufacturer. With the board finished it was time to solder the components to it by hand. Since all of the capacitors and resistors were in the 0805 or larger package, soldering them was quite straightforward. The VNF7000AYs had proven to be a bit more tricky. The large exposed pad at the bottom of the e-fuse package requires a very good connection to the PCB, which means making sure it is soldered properly. However, due to the robust thermal design of the board, it was somewhat difficult to heat up the PCB pad and the exposed pad on the VNF7000AY enough for them to solder together properly. This problem was solved by heating the vias in the PCB pad from the bottom with a soldering iron and heating the e-fuse device with a hot air soldering gun. This process resulted in good and reliable connections between the PCB and the VNF7000AY devices. A similar issue was encountered when soldering the microcontroller, but its exposed pad is much smaller, and the pad on the PCB had thermal reliefs, which allow the MCU to be soldered quite easily with just a soldering iron. Images of the top and bottom of the populated board are in figures 5.16 and 5.17. To make sure that the components were soldered correctly, a simple programme that flashes an LED was loaded into the MCU. After some issues with the JTAG connector, the programme was successfully loaded, and the LED started flashing, which meant it was time to start with software development.

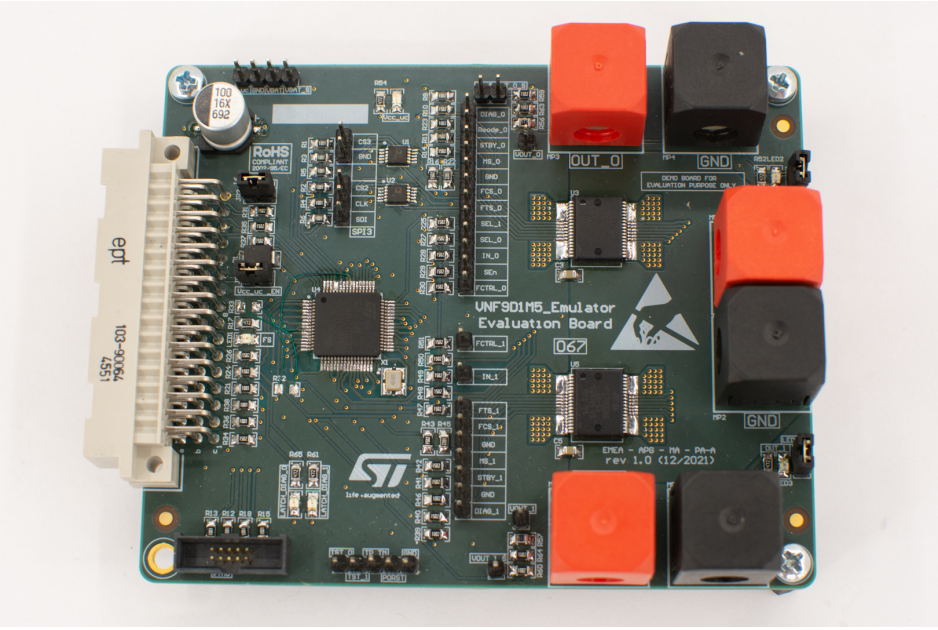


Figure 5.16: The top of the populated emulator PCB

5.4 Board and signal description

This section gives a brief overview of the PCB and the component placement on it, and the signals accessible on the PCB. The board description is in figure 5.18 and the legend for this description is in table 5.3. The signal description is available in appendix

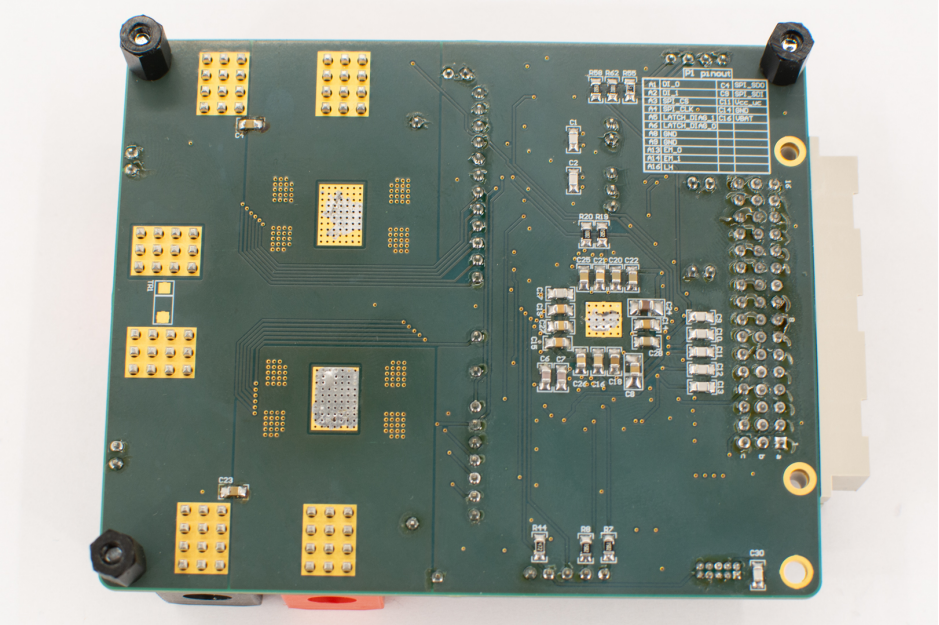


Figure 5.17: The bottom of the populated emulator PCB

Component Number	Component Description
1	Board to board connector
2	JTAG connector
3	Power supply connectors
4	Channel 0 output connectors
5	Channel 1 output connectors
6	Microcontroller
7	Digital potentiometers

Table 5.3: Legend for the board description in figure 5.18

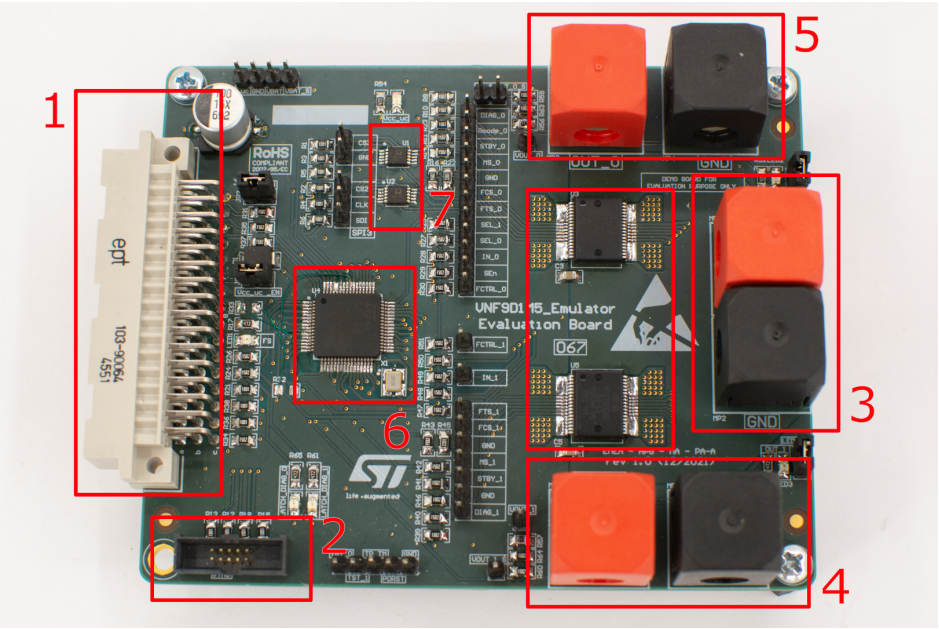


Figure 5.18: Description of the emulator's PCB

Chapter 6
Design and Implementation of emulator software

The emulator emulates the additional features of VNF9D1M5 using a microcontroller. The MCU chosen for this purpose is the SPC582B62E1 from the Chorus family with one megabyte of flash memory. The behaviour of the VNF9D1M5 device can be described using a state diagram in its datasheet [5]. This state machine needs to be implemented in code and programmed to the microcontroller. The MCU handles the SPI communication with the master system, controls the e-fuse devices and sets the digital potentiometers. It also takes measurements of all of the analog feedback channels. This requires the use of the microcontroller's peripherals, which need to be configured through software.

Hardware Abstraction Layer (HAL) libraries allow programming of the microcontroller without extensive knowledge of the MCU. However they limit the amount of control over the microcontroller. Instead, the programme is written as a register access code, meaning that all of the MCU's operations are defined by writing and reading registers. This approach produces a microcontroller firmware (FW) that does exactly what is required, is usually faster and requires less memory than a version made using a HAL library. The price for this is that the microcontroller's documentation [13] needs to be studied in great detail in order to know how to operate it. The result of this is a simple HAL-like library for controlling the MCU.

6.1 Microcontroller configuration

Before the programme can start emulating the functions of the VNF9D1M5, the microcontroller needs to be initialised to perform the functions required. This includes configuring the MCU's core and all of the required peripherals. In this section, the configuration of the microcontroller and it's peripherals is described.

6.1.1 Core and clock configuration

Before the peripherals can be configured, the microcontroller core and clocks need to be initialised first. This also includes the initialisation of the flash memory. Flash initialisation is straightforward; the only thing that needs to be done is to set `Read Wait State Control` bits in the `Platform Flash Configuration Register 1` which defines the number of wait cycles added to flash accesses. The value to be set is 2, and it is described in the MCU datasheet [6] for the given clock frequency of $f_{clk} = 80\,\text{MHz}$.

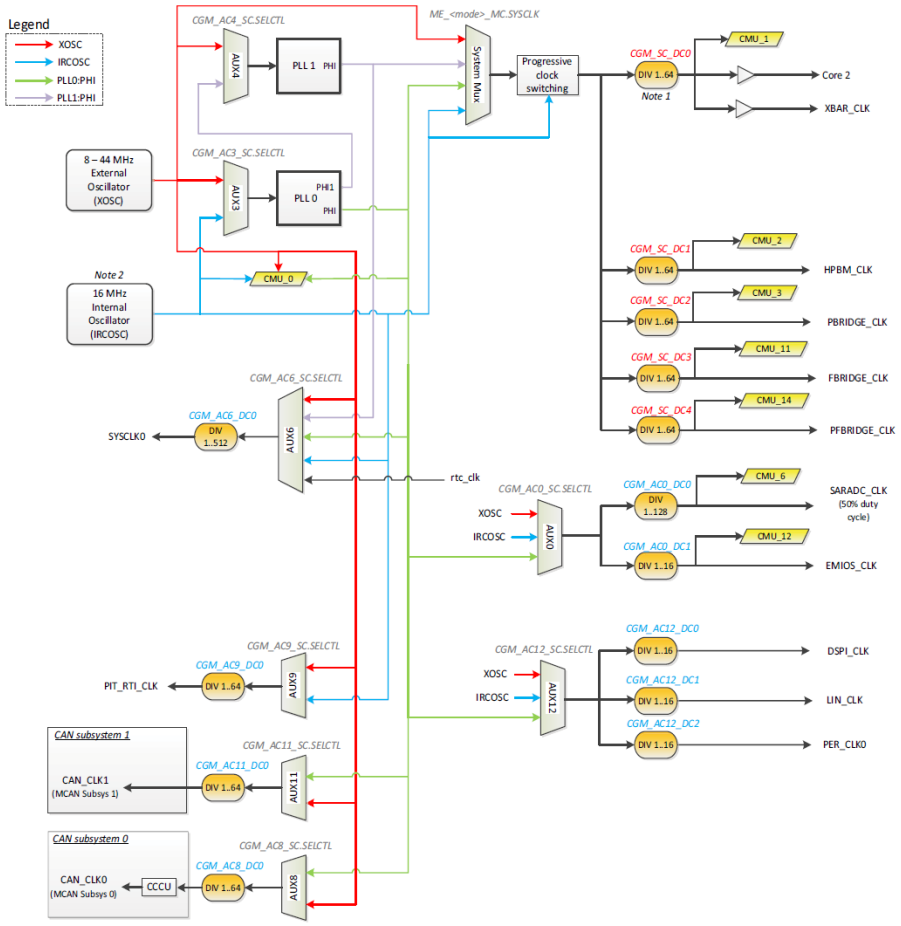


Figure 6.1: Clock distribution diagram for SP582B MCU [13]

The next stage is the initialisation of the various clocks in the MCU and the configuration of their distribution to the different peripherals. The emulator boards are fitted with a 40 MHz external crystal oscillator, which is the basic clock source for the microcontroller from which the other clocks will be derived. After starting, the MCU is switched into DRUN mode, in which it can use the external oscillator as a clock source for the phased-locked loops (PLLs). The PLLs are then configured to produce the final 80 MHz clock, which is the maximum rated operating frequency of the MCU. The PLL1 is then set to be the system clock. As you can see in the clock generation diagram in figure 6.1, clocks for the peripheral are all derived from the main system clock and are enabled and scaled by configuring the dividers through which they are routed. When a peripheral needs to be used, its clock needs to be enabled first, and that is done through the clock generation module (CGM). When the divider that generates the clock for the required peripheral is configured and enabled in the appropriate register of the CGM. For example, the ADC clock is generated by Auxiliary Clock 0 Divider 0 (`CGM_AC0_DC0`). To enable the clock for the ADC, the Auxiliary Clock 0 Select Control Register (`CGM_AC0_SC`) needs to be configured to use the PLL0 as a clock source by writing the appropriate value to it (0b0010). Then the divider itself is configured to produce the clock required for the ADC (max. 16 MHz). This is done by setting the dividing value in the `CGM_AC0_DC0` control register and setting the enable bit to one to start the clock.

6.1.2 SIUL2 configuration

In SPC582B, the connection between the core, the peripherals and the physical MCU pins are managed by the System Integration Unit Lite 2 (SIUL2) module, which provides the control over ten ports with up to 16 bits of bidirectional GPIO signals, supports up to 16 internal interrupts whose trigger event can be configured individually. SIUL2 module also carries out the multiplexing of the signals between the MCU pins and the peripherals. The configuration of these multiplexers is what decides the function of the associated I/O pin, and it is done in the I/O Pin Multiplexed Signal Configuration Registers (`SUIL2_MSCR_IO`). This register also controls the drive strength, pull up and down resistors and more. When configuring the pin as a peripheral output, the Source Signal Select (SSS) bits in the `SIUL2_MSCR_IO` registers are set to an appropriate value that can be found in the I/O definition tables, which are a part of [11]. When configuring inputs for the peripherals, the SSS bits in the Multiplexed Signal Configuration Register for Multiplexed Input Selection (`SIUL2_MSCR_MUX`) need to be set to the values from the multiplexing tables as well. The structure of the I/O multiplexers is in figure 6.2. This needs to be done for pins that are supposed to be connected to the MCU peripherals: ADC, DSPI modules and eMIOS module. For example, on the PCB, the SPI master serial clock signal is connected to pin 53 of the MCU, which corresponds to port PE[11]. To configure the pin to carry the signal to the DSPI module, the SSS bits in the `SIUL2_MSCR_MUX[890-512]` register need to be set to the value 0x02.

As already mentioned, the SIUL2 module also manages external interrupts from the I/O pins, which can be used to generate a core interrupt by an external signal edge. An external interrupt needs to be enabled by writing 1 to the appropriate bit of the DMA/Interrupt Request Enable Register 0 (`SIUL2_DIRER0`) and then selecting if a rising edge, a falling edge, or both will trigger the interrupt by writing the `SIUL2_IREER0` or `SIUL2_IFEER0` registers. If the interrupt is properly configured in the SIUL2, when the specified edge occurs on the pin, an interrupt signal is sent to the interrupt controller, which handles the interrupt according to its configuration. As you can see in the diagram in figure 6.3, the sixteen available external interrupts are grouped into four groups of four, meaning that an interrupt from a group could be caused by a single signal in the group. The source pad of the interrupt needs to be determined by the software. When configuring the GPIO signals, the most critical of them were also configured to trigger interrupts. Some trigger the interrupt through the SIUL2; some trigger them through the WakeUp Unit (WKPU). The configuration of interrupts in the WKPU is essentially the same as in SIUL2, but it produces different interrupt requests to the interrupt controller. The signals that produce interrupts are:

- Direct input 0 (DI0) - WakeUp External Interrupt 2
- Direct input 1 (DI1) - WakeUp External Interrupt 8
- Emergency 0 (EM0) - SIUL2 External Interrupt 0
- Emergency 1 (EM1) - SIUL2 External Interrupt 10
- Limp Home (LH) - SIUL2 External Interrupt 11

As you can see, the EM1 and LH signals are in the same interrupt group; therefore, they both produce the same interrupt request. When the interrupt service routine services the interrupt produced by these signals, it needs to check which signal generated the interrupt by reading the General Purpose Data Input (GDPO) of the pins.

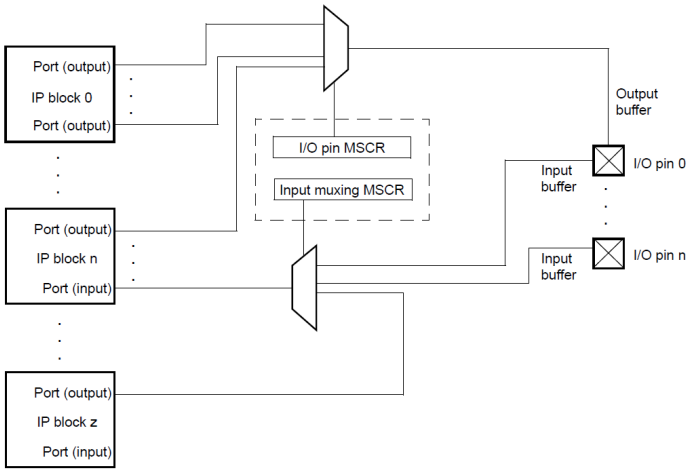


Figure 6.2: I/O pin multiplexing in the SIUL2 module [13]

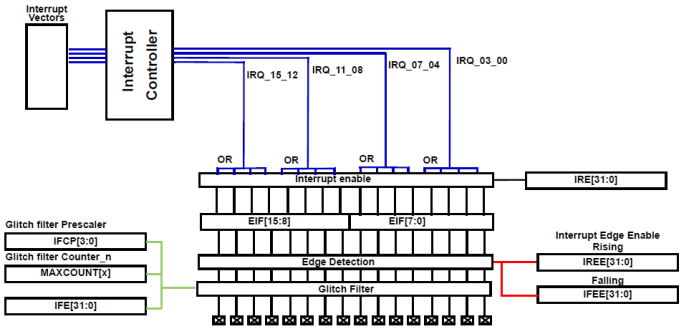


Figure 6.3: Pad to interrupt controller connection diagram [13]

6.1.3 Analog to Digital Converter configuration

The emulator needs to monitor the analog quantities on its input and outputs. The following quantities are being measured:

- Battery voltage V_{BAT}
- Output voltages on both channels V_{OUT0} and V_{OUT1}
- VNF7000AY MultiSense outputs MS0 and MS1

The battery and output voltages are measured via voltage dividers with a ratio of 1 : 6 so that the high voltages (up to 28 V) can be measured by the MCU with a 5 V power supply. This will need to be taken into account when interpreting the ADC outputs.

The VNF9D1M5 uses a 10 bit ADC to perform its measurements. In the emulator, these measurements are made with the ADC integrated into the MCU. SPC582B MCUs are equipped with a 12-bit successive approximation register analog to digital converter (SAR ADC) that is suitable for use in the emulator. This ADC offers internal channels that are connected directly to the I/O pins (the actual number of available pins depends on the package) and external multiplexed channels that can be mapped to internal channels. When sampling an external channel, external channel decode signals can be used to set an external multiplexer to connect the correct signal to the MCU pin. The external channels are very useful for measuring the MultiSense outputs of the VNF7000AYs because the external channel decode signals can be used to address

the MultiSense multiplexer inside the VNF7000AYs. Only two of the three available external decode signals are used as there are only two selection signals for the MultiSense. When deciding which external signal is assigned to which MultiSense signal, it is important to select the channels whose combination of external decode signals will select the correct MultiSense output. For example, external channel 128 needs to be used for current sense measurement because the external channel signals are both 0, which selects the I_{SENSE} output of MultiSense. The external channel 130 produces external decode signals 1 and 0, which corresponds to temperature sense selection on MultiSense. The assignment of signals to channels is in table 6.1. External channels 128 and 130 are mapped to internal channel 38, and external channels 136 and 138 are mapped to internal channel 40.

As the emulator only requires a 10 bit ADC, the SAR ADC is configured to run in a 10-bit mode instead of its usual 12 bit. This is part of sampling timing configuration, which is done by writing into a Conversion Timing Registers (`CTR`). When the emulator outputs are active, the ADC needs to make measurements of the required quantities continuously, and that is why it is configured to run in Scan Mode. Once the ADC is enabled in this mode it starts the conversion of the selected channels one by when. When all of these channels have been converted, the ADC start again with the first one and so on. This continues until the conversions are stopped again by the software. This helps to decrease the load on the MCU's core, as it does not need to keep monitoring whether the conversions are done and start them again.

As you can see in the structure in figure 6.4, the ADC does not contain any sort of supply independent voltage reference (such as bandgap reference), which could be used to correct for changes in the power supply of the MCU. This means that when interpreting the conversion results, the MCU supply needs to be measured externally, or a possibility of inaccuracies needs to be considered. The MCU on the emulator, fortunately, does not interpret any of the measurements (it returns raw ADC data), so this problem needs to be solved by the master device.

Signal	Assigned Channel
V_{BAT_SENSE}	Internal Channel 35
V_{OUT0_SENSE}	Internal Channel 39
V_{OUT1_SENSE}	Internal Channel 49
I_{SENSE0}	External Channel 128
I_{SENSE1}	External Channel 136
V_{SENSE_TC0}	External Channel 130
V_{SENSE_TC1}	External Channel 138

Table 6.1: SAR ADC channel assignment

6.1.4 Deserial Serial Peripheral Interface

The SPC582B is equipped with four Deserial Serial Peripheral Interface (DSPI) modules, which on this device are limited to only the SPI functionality. In the emulator, two SPI connections are required, one through which the emulator is controlled and one through which the emulator controls the digital potentiometers; therefore, two DSPI modules with different configurations are used.

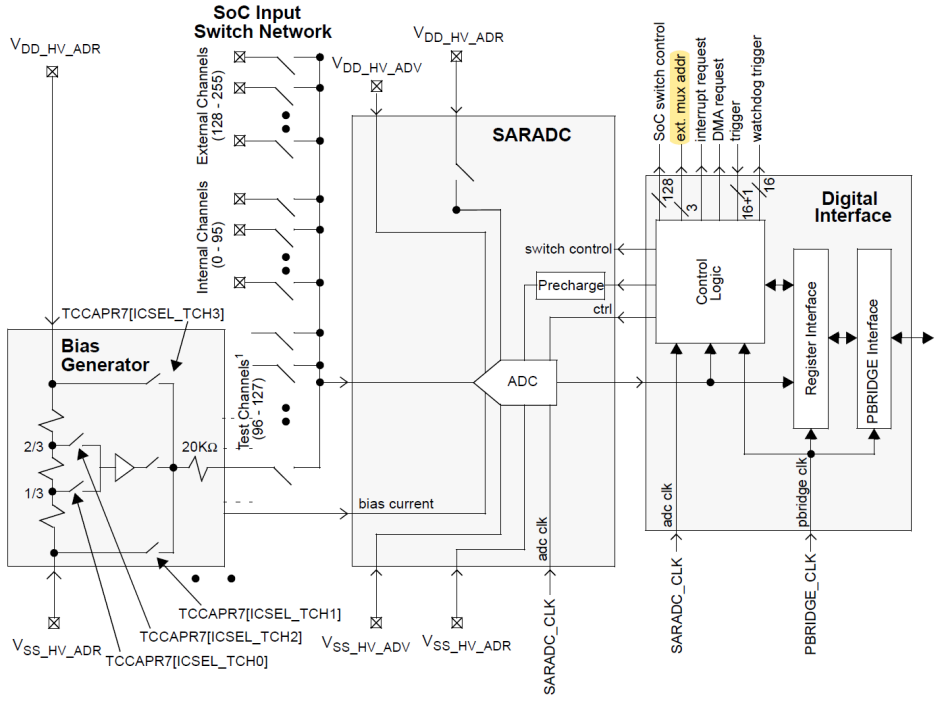


Figure 6.4: Structure of the SAR ADC in the SPC582B MCU [13]

The module chosen for the control and communication interface with the master application is DSPI3 configured in SPI slave mode. This module needs to be configured to behave the way the SPI interface on the VNF9D1M5 is described in the datasheet [5]. It is unfortunately not possible to implement the SPI communication exactly as specified in the datasheet because, unlike the real device, the emulator needs some time to process the first byte of the message to prepare a response which will be transmitted when receiving the second and third bytes, as indicated in the diagram in figure 6.5. Aside from that, the DSPI3 module is configured to work in the same way the SPI works in the device, which is:

- Clock Polarity (CPOL) = 0 - meaning that the clock signal is low when inactive
- Clock Phase mode (CPHA) = 0 - meaning the data signal is sampled with the rising edge of the clock signal
- Frame size = 8 bits

The clock phase and polarity configurations are done in the DSPI3 Clock and Transfer Attributes Registers (`DSPI3_CTARn`). The SPI on the VNF9D1M5 uses 24-bit frames, but if that were the case for the emulator, it could not process the first byte of the frame to prepare the response. That is why the DSPI3 module is configured to work with 8-bit frames. When in slave mode, the DSPI module only reacts to chip select 0 (CS0), and therefore there is no configuration required (only in SIUL2). Two MCU pins can be configured as CS0 for the DSPI3 module, which is used to eliminate SPI errors generated by the DSPI3 module cased by processing the 24-bit frames as 8-bit ones. The DSPI3 module's Receive FIFO Drain Flag (`RFDF`) is enabled by setting the corresponding bit in the Request Select/Enable Register (`DSPI3_RSER`), which allows the module to generate an interrupt request when a frame has been received. When

servicing this interrupt, the received data can be processed and the response loaded into the transmit register.

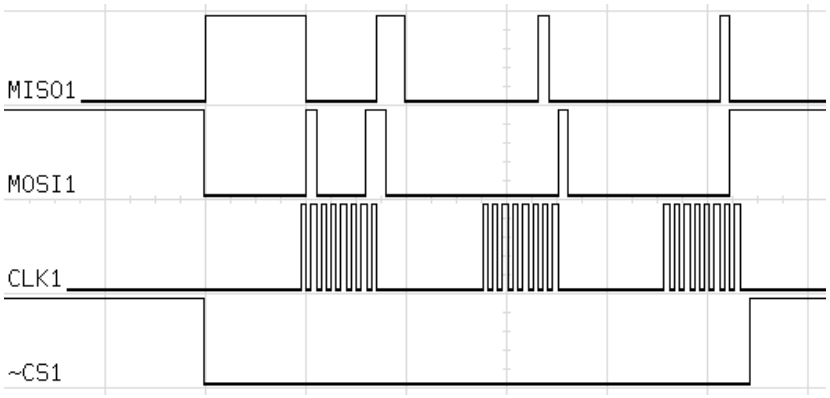


Figure 6.5: SPI frame format required by the emulator

The control of the two digital potentiometers is handled by the DSPI1 module, configured as a master. The module is configured to transmit the messages in the format required by the digital potentiometers, which can be seen in figure 6.6. The clock is idle low (CPOL=0), and the data is sampled on the rising edge of the clock (CPHA=0). The bit A0 determines which of the two potentiometers in the AD5162 is being set, and the bits D7-D0 determine the position of the slider. Binary value 0 corresponds to minimum resistance between the A terminal and the wiper terminal W, and the value 255 corresponds to the maximum resistance value between these terminals. The communication baud rate is set to 1 MHz, by setting the baud scaler and baud prescaler values in the `CTAR` register. As there are two of these devices, two chip select signals need to be used. The chip select signals chosen are CS1 and CS3. The AD5162 does not have a serial output; therefore, the communication is only from the MCU to the digital potentiometer, and there is no feedback from the device. The transmit buffer of the DSPI1 module is enabled, which allows the programme to load the setting for all four potentiometers at once (two dual-channel devices) without waiting for the transmission to finish.

6.1.5 PWM generation module configuration

When a channel of the emulator is in capacitive charging mode, the output is turned off and on again in an effort to charge the load capacitance over a longer period of time, thus lowering the RMS value of the current passing through the wiring. This could be done by toggling the INx pin through the SIUL2 GPDO register, but this approach takes up a lot of the core's resources. A better option is to use a dedicated

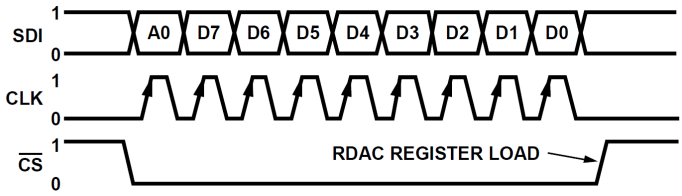


Figure 6.6: SPI communication format used by the digital potentiometers [8]

timer peripheral, which in the case of the SPC582B is called the Enhanced Modular IO System (eMIOS) and provides 32 16bit counters. These counters can be used to generate, among other things, a Pulse Width Modulated (PWM) signal, and that is what they are used for in the emulator.

To generate a PWM signal, two counters are connected together; one counter creates the PWM time base for the other counter. The counter that produces the timebase is configured in Modulus Counter Buffered (MCB) up mode with clear on match end. In this mode, the counter starts at 0 and counts up until it reaches a value in its compare register A, at which point its internal counter resets and starts again. The value in the internal counter register is used as a counter bus. The PWM generation counter is in Output Pulse Width Modulation Buffered mode and has two compare registers, A and B. When the counter bus reaches the value in register A, the output of the counter (the pin) is driven low; when the counter bus reaches the value in compare register B, the output is driven high again. By setting the values of the compare registers, the duty of the PWM signal can be defined.

The PWM signal required for capacitive charging has a frequency of $f_{PWM} = 250\,\text{Hz}$ and its duty cycle is $D = 0.5 = 50\%$. From this, the required settings of the counters and the eMIOS module are determined. In order to produce such low frequencies with 16-bit counters, the clock for the modulo f_{tim} timer needs to be slowed down. The maximum value that can be loaded into a 16 bit compare register is 65535, which dictates the maximum ratio f_{tim}/f_{PWM}. From this, the maximum usable timer clock frequency can be determined:

$$f_{tim\_max} = 65535 \cdot f_{PWM} \doteq 16.4\,\text{MHz}. \tag{6.1}$$

It was decided that $f_{tim} = 4\,\text{MHz}$ will be used, as it is still precise enough and leaves more room for flexibility. This was done by enabling and setting the global prescaler value through the eMIOS Module Configuration Register (`EMIOSMCR`). Based on this the values for the compare registers can be computed. The modulo counter compare register A is set to

$$A_{mod} = \frac{f_{tim}}{f_{PWM}} = \frac{4\,\text{MHz}}{250\,\text{Hz}} = 16000. \tag{6.2}$$

Since the required duty cycle is 50%, the compare register of the PWM counter are:

$$A_{PWM} = \frac{A_{mod}}{4} = 4000 B_{PWM} = \frac{3A_{mod}}{4} = 12000. \tag{6.3}$$

There are two output channels on the emulator. A PWM signal can be generated for each of them as there can be two (or more) PWM generating counters assigned to one modulo counter. The MCU pins controlling the outputs of the emulator can be connected to eMIOS channels 19 (output channel 0) and 27 (output channel 1). As you can see from the eMIOS diagram in figure 6.7, the only counter that can be used as a counter bus source for these channels is channel 27, which is connected to counter bus A. That is why this channel was used as the modulo counter.

6.1.6 PIT configuration

During its operations, the emulator needs to make precise measurements of time, such as the time between watchdog operations through SPI. Most microcontrollers contain timers for the express purpose of measuring time. The SPC582B contains

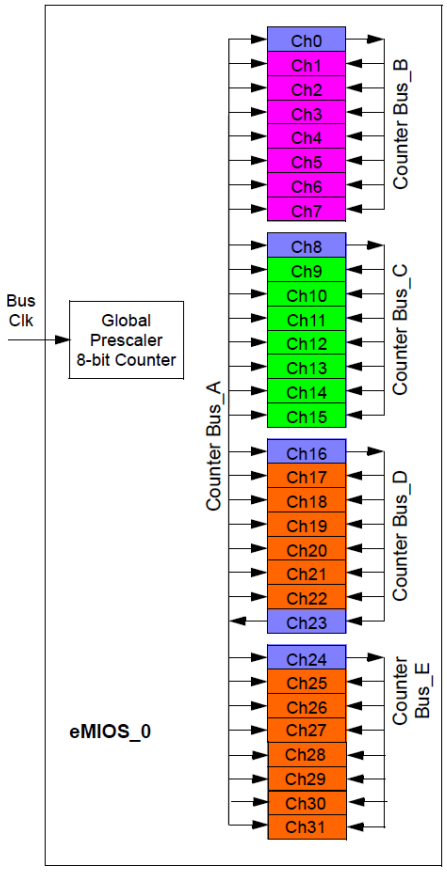


Figure 6.7: Structure of the eMIOS module in SPC582B [13]

several kinds of timers, one of which (eMIOS) was introduced in the previous section. There is also the Periodic Interrupt Timer (PIT) module, which can generate interrupts with a predefined period, for example, for a periodic sampling of a signal. The Timer Load Value Register (`LDVAL`) of the PIT's channel is loaded with a value corresponding to the required period. Once the PIT is started, it counts from this value down to 0; when the interrupt flag is set in the PIT Timer Flag Register (`TFLG`), the value from the `LDVAL` register is loaded into the counter and the PIT start counting down again.

When the emulator operates in Normal Mode, the watchdog is supposed to be serviced at least every 100 ms, else the emulator switches to Fail Safe Mode. When Normal Mode is entered, channel 0 of the PIT is set to generate an interrupt every 100 ms and started. When the watchdog is serviced, the counter is reset, meaning that the interrupt is generated only when the watchdog has not been serviced in more than 100 ms. The counter is reset by disabling and then immediately enabling the channel in the PIT Timer Control Register (`TCRL`). A similar approach is also used to time the capacitive charging mode, which is supposed to be on for 250 ms (PIT channels 1 and 2) and the switching from Fail-Safe Mode to standby mode when inputs are inactive for 14.1 ms (reconfigured channel 0). The PIT uses the core clock $f_{clk} = 80$ MHz for its counters, which are 32 bit. The `LDVAL` register values required for timings mentioned above are in table 6.2. All of the mentioned functions could be done using the other available counters, but the PIT has proven to be easy to work with and produces satisfactory results.

t [ms]	LDVAL [-]
14.1	1128000
100	8000000
250	20000000

Table 6.2: PIT LDVAL register values for different periods

6.1.7 Interrupt controller configuration

When an interrupt request is generated by a peripheral (or software), it needs to be scheduled to be handled by the processor. In SPC582B, this is done by the Interrupt Controller (INTC) module, which is responsible for scheduling the interrupts according to their priority, masking the maskable interrupts and more. In order for the required interrupts to work, two things need to be done. In the Interrupt Service Request Vector (ISRVec) table, the interrupt service routine (a function) needs to be assigned to the appropriate interrupt vector. The information about which interrupt vector produces which interrupt vector can be found in the reference manual [13]. After that, the interrupt vectors need to be enabled in the interrupt controller and their priority set. This is done through the INTC Priority Select Register n (`INTC_PSRn`), where n corresponds to an individual interrupt vector. The priorities of all used interrupts were set to the same value, with the intention of tweaking them when some problems were detected. Since no such problems were found, the priorities were left as they were.

6.2 Memory map implementation

The VNF9D1M5 contains Random Access Memory (RAM) and Read-Only Memory registers, which contain the information about the device and through which the device can be controlled. In the real device, these are physical registers connected to the SPI and other components of the device. In the emulator, the RAM and ROM are implemented as data arrays in the microcontroller's memory. For easier access, it is implemented as a union called `VNF9D1M5` composed of a structure `REG` and an array of 16 bit unsigned volatile integers `MEM`. The structure `REG` is designed to copy the register structure, and that is why this structure is composed of individual unions which represent the registers. Each of these register unions is composed from a structure of bit fields that correspond to the bits in the real register and a 16-bit volatile integer (`vuit16_t`) that represents the entire register. Thanks to this structure, every register of the emulator can be accessed as a whole or by its individual bits, which simplifies the programming and improves the readability of the code. The `MEM` array allows the programme to access the registers via their address, which is used when processing SPI operations, as, during these, the address of the requested register is provided. The size of the `MEM` array is defined by the number of registers that can be addressed through the SPI. The 24-bit SPI communication in the VNF9D1M5 uses six address bits, which means it is possible to address up to 64 registers, and that is how large the array needs to be.

A similar approach was chosen when implementing the ROM, which provides information about the device manufacturer, version of silicon and SPI configuration. As it is essentially a constant that is loaded once the program starts and can be only read, it is implemented as an array of 64 unsigned 16-bit integers. The array can be

addressed the same way the ROM registers would be as it is a copy. The VNF9D1M5 also has a Global Status Byte (GSB) that provides the most important information about the status of the device (fault reporting,...) and it is transmitted at the start of every SPI communication (when receiving the first byte). In the emulator the GSB is implemented as a union of a structure `B` and a `vuint16_t` variable `R`. The structure `B` consists of bit fields, each representing a bit of the GSB. This again allows easy access to every individual bit of the GSB as well as the entire byte at once. The union defining the GSB is in figure 6.8, and the RAM registers are defined in a similar way.

```
union GSB_TAG{
    struct{
        vuint8_t GSBN:1;
        vuint8_t RSTB:1;
        vuint8_t SPIE:1;
        vuint8_t OT_PL:1;
        vuint8_t ITLOFF:1;
        vuint8_t IPEAK_DETECT:1;
        vuint8_t BIT1:1;
        vuint8_t FS:1;
    } B;
    vuint8_t R;
};
```

Figure 6.8: Definition of the union representing Global Status Byte

6.3 SPI communication implementation

As already mentioned, the SPI communication implementation on the emulator has some specific challenges. The VNF9D1M5 uses a specific extended version of SPI called ST SPI. When the first byte of the SPI frame is being received, the Global Status Byte is being transmitted back to the master. In order for the most recent GSB to be prepared for transmission, the transmit register of the DSPI3 module is continually being updated with the most recent GSB (in Failsafe and Normal modes). When the first byte is received, the address of the register being accessed is known, and its contents can be prepared for transmission by loading it into the TX register. This is the reason why there needs to be a gap between the first and second byte of the SPI frame when communicating with the emulator. The MCU simply cannot prepare the information for transmission fast enough. This part of the SPI frame processing is done inside an interrupt routine which is triggered by the RXFIFO Drain Request flag. When the first byte is received, it is written into the address part of a buffer structure `spi_frame`, which is used to compose the SPI message for further processing. After receiving the address byte, the DSPI3 peripheral module is reconfigured (through the CTAR register) to work with 16-bit frames. This allows the data to be received and transmitted in a single block reducing the number of interrupts needed to process the frame. When the first byte has been received, the chip select signal needs to be pulled high again; otherwise, an SPI error is generated when receiving another byte. This is done by reconfiguring a pin that is connected to a pull-up resistor as a second chip-select 0 (CS0) for the SPI peripheral for the duration of the SPI message processing. When the data bytes are received, they are written into the data part of the `spi_frame` structure, the peripheral is reconfigured back to being 8-bit, and the latest GSB is loaded into the transmit register.

6.4 Finite state machine implementation

The behaviour of the VNF9D1M5 is controlled by an internal finite state machine, which can be described by a state diagram available in the device's datasheet [5]. The same state machine, with minor modifications, is implemented in the firmware of the emulator's microcontroller, and its state diagram is in figure 6.9. This section describes the implementation of the finite state machine in the emulator. First, the individual states (modes) are described and then the state transition mechanism. In the emulator's firmware, the modes are implemented as functions in which the appropriate functionality is implemented. Each of these functions also checks the transition conditions, and when they are met, a mode transition is initiated.

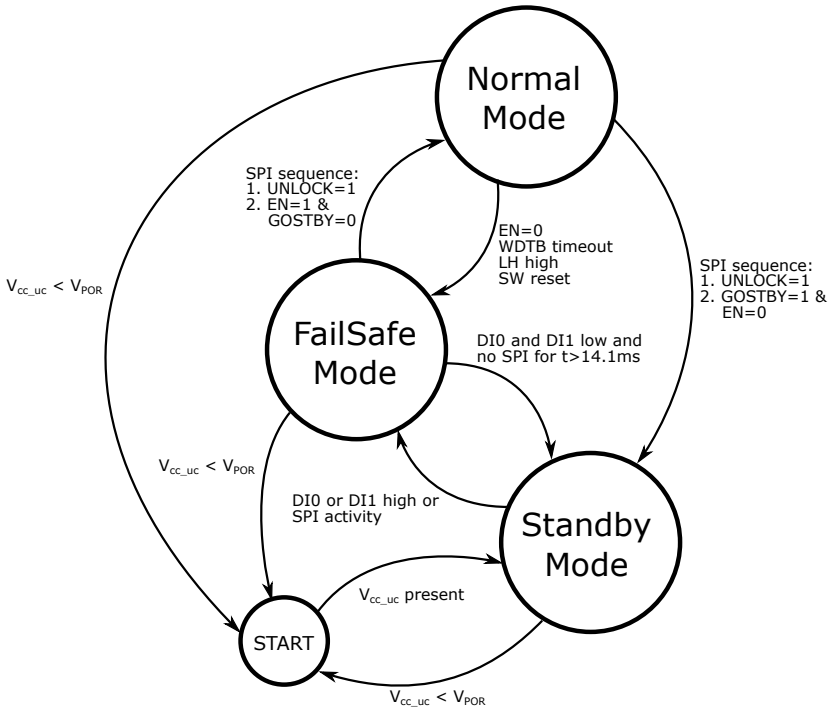


Figure 6.9: State diagram of the emulator state machine

6.4.1 Mode transition

In a state machine, the transitions between the states are as important as the states themselves. The transition conditions for the states are illustrated in the state diagram in figure 6.9. All of the transitions to the START states are caused by the microcontroller supply voltage V_{cc_uc} falling under the power-on reset level V_{POR}, which resets the microcontroller. There is nothing that can be done about these transitions in software. Once the MCU is powered, and the initialisation of its peripherals has been finished, the program transitions into the standby mode, where it waits in a loop. The information about in which mode is the emulator supposed to be is kept in a global variable `current_mode`. If the mode in this variable is different from the actual mode that is running, the loop of the mode is interrupted, and the correct mode is entered. The `current_mode` variable is changed when one of the state transition events occurs, such as one of the DIx pins going high while the device is in the standby mode. The transitions caused by control signal changes are triggered by changing the `current_mode` in

an interrupt generated by the signal change. When the transition is caused by a timeout (watchdog, no activity in failsafe mode), the contents of `current_mode` are done in the interrupt service routines of the PIT interrupts. If the transition is caused by a write to a bit (EN or GOSTDBY), the contents of `current_mode` are changed at the end of the mode cycle, when the SPI message has been properly processed. When a mode loop is exited due to `current_mode` variable change, the program returns to the main loop, where the next mode is entered according to the contents of the `current_mode` variable.

6.4.2 Standby mode

When no direct inputs are active, and no SPI communication is in progress for $t_{stdby} = 14.1\,\text{ms}$ the VND9D1M5 enters the standby mode. In this mode, the device outputs are not active, the ADC is stopped, the registers are cleared, SPI communication is inactive, and the device is in a low power state. In the emulator, the standby mode has the same functionality. Upon entering the standby state, the microcontroller disables both of the VNF7000AYs by setting the IN signals low and the STDBY_ON signal high, stops the ADC and disables all of the periodic interrupt timers. Originally, it was the intention to switch the MCU into a low power state such as HALT0 or STOP0 mode (for more information, see [13]), but unfortunately it has proven to be problematic to implement and the version of FW at the time of writing simply waits for and external signal to interrupt an infinite wait loop. The MUC was successfully programmed to go to the STANDBY0 low power state, but the wake-up time from this mode is too long and the emulator would not be able to meet the reaction time requirements. Upon leaving the Standby mode, the emulator sets the VNF7000AY STDBY_ON signals low to switch the devices back to active mode and sets the Reset (RST) bits in the GSB and the `ITCNTSR` register.

6.4.3 Failsafe Mode

The failsafe mode of the VNF9D1M5 is intended to assure the safe operation of the device in case of a failure. The switch to failsafe mode could be caused by failure in control logic, fault detection on the e-fuses output, and many other events. In this mode, the inputs are controlled through the direct input pins (DI0 and DI1), disregarding the control through the SPI. SPI registers are active, and the current sense is available. Most of the features of the device are available. When the emulator enters the Failsafe mode, the ADC measurements are started and the timeout timer set to $t_{stdby} = 14.1\,\text{ms}$ and enabled, regardless of the previous mode. During every cycle of the mode loop, the tasks described below are performed. At the end of each cycle of the Failsafe mode loop, the transition conditions are evaluated, and if a condition is met, the program leaves the loop and enters another mode. Before the program leaves the Failsafe mode, the ADC conversions are stopped, and the PIT is disabled.

Evaluating the status of the channels

The status of the emulator's outputs is updated by reading the pins connected to the diagnostic outputs (STATUS and MultiSense) of the VNF7000AYs. Based on this, the status bits in the Output Status Registers (`OUTSRx`) of each channel and the GSB are updated. The `IPEAKLSRx` (IPEAK fault latch off status) and `ITLOFFSRx`

(I2T latch off status) bits are set when the channel's STATUS pin goes low (it is an open-drain pin). These two faults are indistinguishable from each other; the only way to differentiate between them is to have access to the counters of the I^2t protection mechanism, which is not possible in the emulator. There are two more fault bits that can be set during the diagnostic process, the `OTSRx` (overtemperature fault) and `PLSRx` (power limitation). The VNF7000AY signals these faults by pulling the voltage on the MultiSense pin to $V_{SENSEH} = (5\,\mathrm{V} to 6.6\,\mathrm{V})$, which will be measured by the ADC and because this voltage is higher than the ADC's reference voltage the output is going to be at the maximum possible value of the 10-bit result 0x3FF=1023. If ADC detects this value on the MultiSense pin, the `OTSRx` and `PLSRx` are set for that channel. Based on these four bits (for each channel), the channel's FAULT_DIAG signal is set based on the logical OR product of these bits. Masking bits in the Control register (`CTRL`) can be used to prevent a fault from setting the FAUL_DIAG signal. As long as one of the `ITLOFFSRx`, `IPEAKSRx` and `OTSRx` fault bits is set, the output of the channel can not be enabled, and the bits can only be cleared by a read and clear (RC) operation. As the SPI might not be functional when the emulator is in the Failsafe mode, it is also possible to clear all faults on the channel by toggling the DIx pin. When the fault bits have been cleared, the faults need to be cleared in the VNF700AYs as well, by either toggling the INx signal if the fault was power limitation or overtemperature or by toggling the F_CTRLx signal in case of IPEAK or I_2t latch off. There are also some feedback status bits, but those are not very important, and their purpose can be found in the VNF9D1M5 datasheet [5].

Setting the digital potentiometers

Because the chosen digital potentiometers have volatile memory, they can not be relied upon to keep the set value indefinitely. They could, for example, be reset by a voltage drop on the power supply or interference in the SPI communication. To solve this problem and guarantee stable performance of the emulator, the setting of the potentiometers are periodically updated with the current settings from the I^2t Configuration Control Registers (`ITCFGCRx`). This is done by filling the transmit queue (TXFIFO) of the DSPI1 module with all four messages every time it is detected that the TX FIFO is empty.

Enabling the channel outputs

In the Failsafe mode, the outputs of the emulator are controlled via the direct input (DIx) pins and the emergency (EMx) pins. However, the output might be disabled by a set fault bit that has not been cleared yet. That is why the conditions for the channel to turn on are evaluated in every cycle, and if they are not met, the channel is turned off. When the channel is turned on, it is also evaluated whether the capacitive charging channel mode should be entered. In the real device, the capacitive charging in Failsafe mode is controlled by the Capacitive Charging Mode (`CAPCMx`) bit in one time programmable (OTP) memory; in the emulator, it is implemented as a constant that is hardcoded in the program. By default, capacitive charging in Failsafe mode is enabled on the emulator and can only be disabled by changing the constant and recompiling the code. In the real device, the capacitive charging in Failsafe mode can also be entered by toggling the DIx pin at least five times during a specified window; this feature is not implemented in the emulator.

Processing the ADC measurements

In the Failsafe mode, the analog feedback registers are also available, meaning that the ADC measurements need to be copied from the ADC's data registers to the register map representing the registers of the VNF9D1M5. The battery voltage and output voltage measurement results are simply copied from the ADC register to the appropriate parts of the memory map in every cycle. The current sense feedback is only available while the channel is on, so the values in those registers are updated only while the channel is off. Each channel has a Digital Current Sense register (`ADCISRx`), which contains current sense measurements and a Digital I^2t Current Sense register (`ADCI2TSRx`), which contains current measurements done by the I^2t protection block. However, on the emulator, only the first of these measurements are available, and therefore the two registers contain the same information. The Digital Case Thermal Sensor Voltage register (`ADCTVSR`) provides information about the case temperature of the device. As there are two output devices on the emulator, there are two sources of case temperature feedback. The value in the `ADCTVSR` register represents the maximum of the two measurements. Every analog feedback register also contains an update bit (`UPDTSR`) that is set when the data is written to the register and reset when the data is accessed via SPI.

6.4.4 Normal Mode

The normal mode is the main operating mode of the VNF9D1M5, in which all of the features of the device are available. The device is controlled entirely through the SPI, with the exception of the emergency (EMx) inputs, which can still be used to turn off the output of a channel. The Normal mode's behaviour is very close to the Failsafe mode, which is why in this section, only the parts that are implemented differently from the Failsafe mode are documented. When Normal mode is entered, the ADC measurements are started, and the PIT is set up to generate an interrupt if the watchdog timeout has passed ($t_{WDTO} = 100\,\text{ms}$). The FS bit of the GSB is set to 0.

Enabling the channel outputs

In normal mode, the outputs of the emulator can be controlled through the SPI by writing the `SOCRx` bits of the Channel Control Register (`SOCR`). By default, this is the only way to turn on the output of the channel; however, if the Direct Input Signal Enable (`DIENCRx`) bit is set, the corresponding channel can be controlled through the DIx signals. Same as in the Failsafe mode, there are fault bits that can prevent the channel from turning off, which need to be cleared by a read and clear operation before the output can be enabled. The control and status information is evaluated in every cycle of the loop, and the outputs of the emulator are configured accordingly. Capacitive charging mode can be enabled or disabled for each channel separately by setting the Capacitive Charging Mode (`CAPCRx`) bits in the `SOCR` register. If this bit is set when a channel is being turned on, capacitive charging mode is entered on that channel. In normal mode, capacitive charging can be interrupted by setting the `EXIT_CAPCRx` bit in the `SOCR` register.

Processing the ADC measurements

When the emulator is in normal mode, the measurements are carried out in the same way as in the Failsafe mode, but there are other options when it comes to processing the data. When the Current Sense Sampling Point (`SPCRx`) bit is set, the current sense data is passed through a digital low pass filter before being written into the digital current sense registers. This filter is described by the following equation:

$$y[n] = \frac{1}{16}(15 \cdot y[n-1] + x[n]), \tag{6.4}$$

where $y[n-1]$ is the filter output in the previous cycle, $x[n]$ is the value measured by the ADC and $y[n]$ is the filter output in the current cycle. When this filter is enabled, the first filtered measurement values are available after sixteen ADC samples and then it is updated after every eight samples.

Watchdog functionality

The VNF9D1M5 is primarily controlled through SPI, but when the SPI communications or the controller fail, there needs to be a safe state to which the device switches, and that is the Failsafe mode. To detect the communication failure, a watchdog is implemented in Normal mode. When the device is in Normal mode, the Watchdog Toggling Bit (`WDTB`) in the `SOCR` register needs to be toggled (switched from 0 to 1 or from 1 to 0) at least once every $t_{WDTO} = 100\,\text{ms}$. When this time runs out without the bit being toggled, the device is automatically switched to the Failsafe mode. In the emulator, the watchdog functionality is implemented using a Periodic Interrupt Timer (PIT) set to trigger an interrupt every 100 ms. In the Normal mode loop, the state of the `WDTB` is compared to its state from the previous cycle, and if they are different, the counter of the PIT is reset, allowing another 100 ms window. As long as the `WDTBis` being toggled, the PIT never runs out, and the interrupt is never triggered.

6.4.5 Capacitive charging mode

Capacitive charging mode is not a device state but a channel state because it is possible for only one channel to be in capacitive charging mode. When a channel of the emulator is turned on with capacitive charging mode enabled, the pin that controls the VNF7000AYs INx signal on that channel is reconfigured in SIUL2. Instead of being configured as a regular GPIO pin, it is set up as a PWM output of an eMIOS channel through the Source Signal Select (`SSS`) field of the `SIUL_MSCR_IO` register assigned to that pin. When the capacitive charging starts, the `CAPCSRx` bit in the `OUTSRx` register is set, and a PIT is set up to trigger an interrupt after $t_{ccm} = 250\,\text{ms}$, which is how long the capacitive charging is supposed to take. When the timer runs out, an interrupt is generated, and its interrupt service routine is called. In this routine, the interrupt flag is cleared, the `CAPCSRx` bit is cleared, and the INx pin is reconfigured back to GPIO mode. During capacitive charging mode, the I^2t functionality of the VNF7000AYs is disabled by pulling the F_CTRL signals low. This prevents the channels from latching off during capacitive charging

Chapter 7
Validation of the emulator

With the hardware and software development finalised, the performance of the emulator needed to be evaluated. The evaluation and validation also verified that the emulator met the requirements and helped to identify and specify the limitation of the emulator with reference to the real device. The testing was concerned mainly with the newly added features, as the VNF7000AY has already been thoroughly evaluated. The evaluation also helped to identify errors in hardware and software before the deployment of the emulator. The VW80000 [14] standard for testing electrical and electronic components for automotive was used as a guideline when creating the test specification. In this chapter, the tests are specified, and their results are presented and evaluated with respect to the requirements. The measurements conducted as a part of this validation were meant to verify the functionality of the device, not determine the exact characteristics. Unless otherwise specified, the $V_{BAT} = 13.5\,\mathrm{V}$ was used.

7.1 Evaluation platform and test setup

Before starting with the testing, an evaluation platform needed to be created and the testing setup prepared. The emulator, same as the device it emulates, cannot operate independently, but it needs a master device to control it. The emulator was designed with a specific master device in mind: the Chorus 1M Tiny Board universal motherboard, which you can see in figure 7.1. It is equipped with a microcontroller that can be used to control the emulator, and thanks to the USB/UART converter, it can also be controlled from a PC. Power for the MCU is provided by a 5 V voltage regulator, which is also used to power the logic part of the emulator. To be able to control the emulator, the MCU on the Tiny Board needs to be programmed with a compatible firmware (a software driver) that processes the commands received from the PC into commands for the emulator.

On the PC side, the whole platform can be controlled via an application with a graphical user interface (GUI), which allows the user to configure and control the emulator and monitor its status. The application provides control through simple buttons and sliders but also direct access to the emulator's registers, and it even allows the user to create and run scripts. In the figure 7.2 a screenshot of the main window of the application is available. The software driver and the PC application were provided by STMicroelectronics and were not a part of this thesis. It will be possible to use this software for the evaluation of the real device when it becomes available. Photography of the entire evaluation platform is in figure;7.3 the emulator board (on the right) and the universal motherboard (on the left) are connected via a testing interface board that

allows simple access to the signal between the emulator and the motherboard.

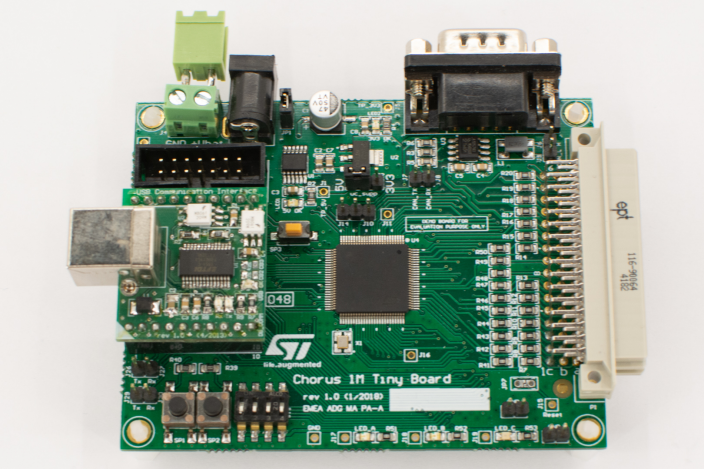


Figure 7.1: Chorus 1M Tiny Board v1.0 universal motherboard

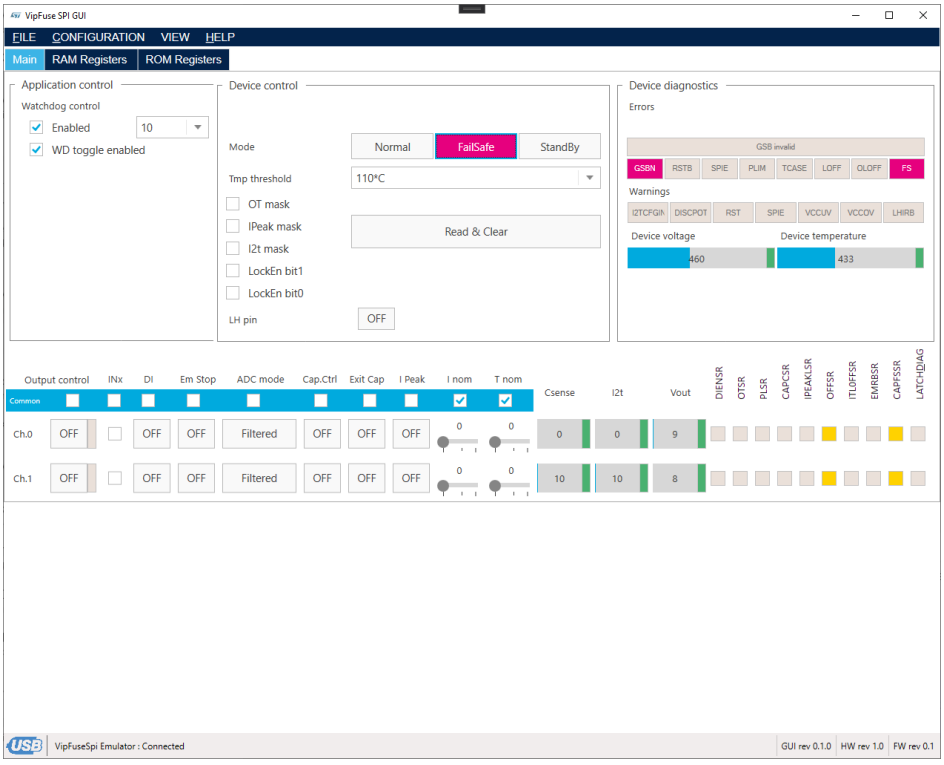


Figure 7.2: Main screen of the PC application for controlling the evaluation platform

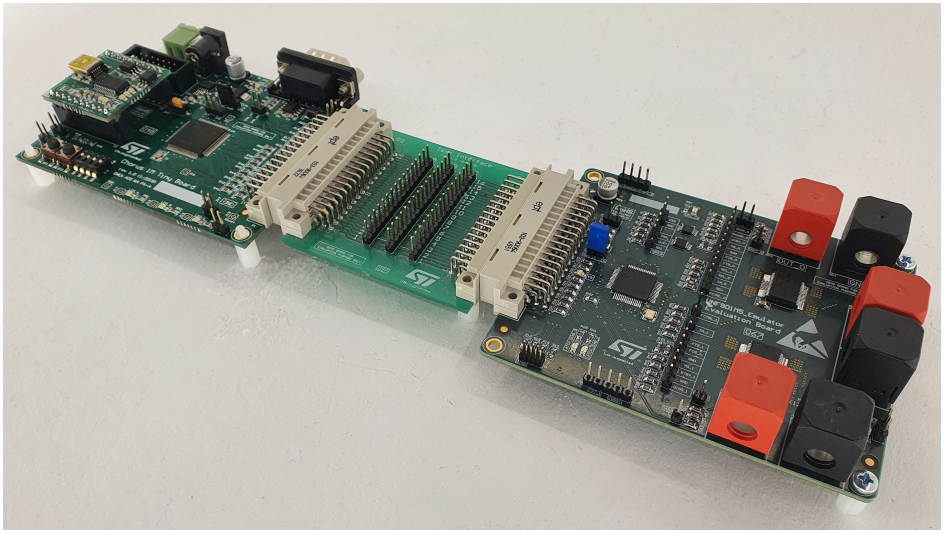


Figure 7.3: Complete evaluation platform with the emulator (right) and motherboard (left)

When performing evaluation tests, it is crucial to have suitable testing equipment, such as power supplies, oscilloscopes, loads and so on. Because the required output currents are so high, some specialised equipment is needed to provide (power supply) and sink (load) these currents. Fortunately, the testing was done in a very well equipped lab at STMicroelectronics, in which such equipment was available. The following equipment was used:

- Keysight MSOX4054A Mixed Signal Digital Oscilloscope
- Keysight N2781B 150A 10MHz Current Probe
- Elektro Automatic PSB 9080-240 3U 10000W Bidirectional Power Supply (2x)
- Statron 2225 Dual lab bench power supply
- Keithley 2000 6 1/2-digit Multimeter
- High power resistor decade (up to five 1 Ω resistors in parallel)

In figure 7.4, you can see a photo of the test setup. The bidirectional power supplies can operate in either supply mode or in sink mode, meaning that they can be used as electronic loads. In supply mode, these power supplies can deliver up to 240 A of current, which is more than sufficient for the evaluation of the emulator. When using the PSB 9080-240 as an electronic load, one problem needs to be addressed. Because the PSB 9080-240 is a power supply, it has large output capacitors to smooth out the output voltage. Unfortunately, when in load mode, these capacitors need to be charged, which causes a large current spike when the device under test (DUT) is connected to the load. The current spike can be reduced by setting the PSB 9080-240 as a voltage source, with output voltage slightly lower than the output of the connected device, which means that when the DUT's output is turned on, the capacitors charge only a little bit. Where possible, the high power resistor decade is used as a load in order to avoid this issue; however, the PSB 9080-240 allows more precise load current settings than the decade.

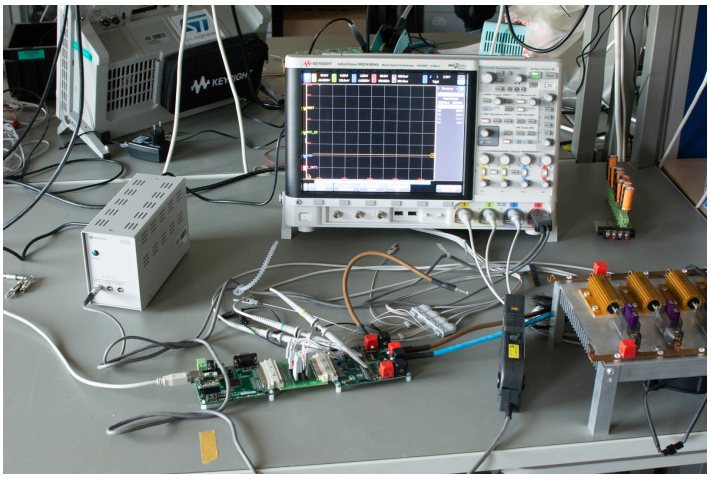


Figure 7.4: Measurement setup

7.2 Control of the emulator through SPI and pins

Before any performance measurements were made, the functionality of the emulator needed to be checked by stimulating the inputs and comparing the measured outputs with the expected ones. A large portion of this part of the evaluation revolved around the control and monitoring of the emulator through the registers accessible via SPI. When testing this, the PC application was used to give the control commands, and the behaviour was observed, or specific conditions were set up (such as an overload on the output), and the contents of the status registers were monitored. The first part of this testing was reading the ROM registers with the help of the ROM registers tab of the PC application. The received values were identical to the ones in the datasheet of VNF9D1M5, which meant this test was passed.

All of the control and status registers work as expected, and most work the same way they do on the VNF9D1M5, with the exception of the limitations of the emulator. For example, the Digital current sense register and I^2t digital current sense registers contain the same value for each channel as there is only one current sense available per channel. Some of the features are not implemented, and therefore they cannot be evaluated; this is the case of the SPI parity bit check or the I^2t counter registers.

7.3 Current consumption of the emulator

One of the limitations of the emulator is the increased current consumption in comparison with the real device, which is mainly caused by the use of a microcontroller. The theoretical range of the emulator's current consumption was discussed in 4.1, but it is important to validate the theoretical values by measuring them. The emulator utilises two different voltage supplies, the V_{CC_UC} used to power the microcontroller and the digital potentiometers and V_{BAT} which powers the VNF7000AYs. Current consumption from both of these supplies in all of the device modes needed to be measured. The emulator has jumpers that can be used to share V_{BAT} from the emulator to the Tiny Board and the 5 V power supply from the Tiny Board to the emulator logic, which were removed for this measurement. The emulator was powered from the Statron 2225 dual power supply, and the currents were measured with two Keithley 2000 multimeters in

ammeter mode. The motherboard was powered from a separate 12 V supply. There was no load on the output of either channel, and the indicator LEDs were disabled so as not to influence the results. The measurement setup indicated in the figure 7.5 was used, and the results of the measurements are in table 7.1. The emulator's output current was measured by the Keysight N2781B current probe clamped over one of the cables connecting the load to the output of the emulator. It is apparent that the emulator's current consumption is much higher than that of the real VNF9D1M5. In normal mode the VNF9D1M5's current consumption is typically $I_{BAT} = 2.5\,\text{mA} + 4\,\text{mA}$ for every active output, and $I_{DD} = 1\,\text{mA}$ (I_{DD} is the equivalent of I_{CC_UC}). The increased current consumption of the emulator was expected, and the measured values are actually lower than expected.

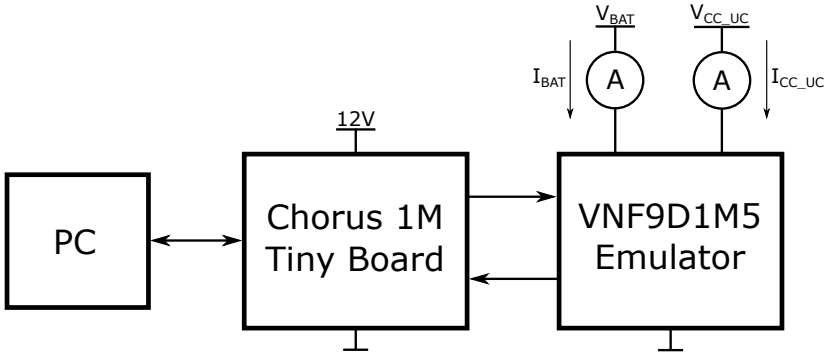


Figure 7.5: Current consumption measurement setup schematic

Mode	Active outputs	I_{BAT}[mA]	I_{CC_UC}[mA]
Standby	0	4.76	45.3
Failsafe	0	4.74	53.5
	1	6.42	53.5
	2	8.08	53.5
Normal	0	4.74	53.4
	1	6.42	53.4
	2	8.08	53.4

Table 7.1: Emulator's current consumption in different operating modes

7.4 Validation of the I^2t curve settings

One of the key features of the VNF9D1M5 is the ability to reprogram the wiring protection mechanism through SPI; therefore, it requires thorough validation. The I^2t curve is defined by two parameters, the nominal current I_{NOM} and nominal time t_{NOM}, each of which can be set to one of eight predefined values. That means that there are $8^2 = 64$ combinations of I_{NOM} and t_{NOM} settings that should be evaluated. It was decided to evaluate every FTS setting with a fixed FCS setting and every FCS setting with a fixed FTS setting, which resulted in 16 measurements. Both of the PSB 9080-240 power supplies were used, one in power supply mode providing V_{BAT} voltage and the other in load mode connected to output channel 1. The testing setup is illustrated in the schematic in figure 7.6.

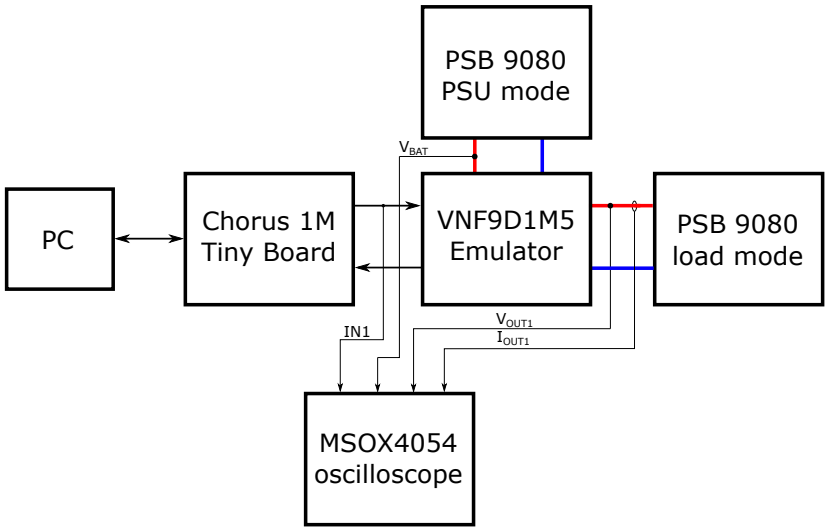


Figure 7.6: Testing setup for emulator's I^2t curve settings evaluation

When validating the FTS settings, the lowest nominal current setting was used $I_{NOM} = 9\,\text{A}$ (FCS=0). The load current used $I_{OUT1} = 13.6\,\text{A}$ was chosen to be on a specific step of the I^2t stairstep curve and should result in latch off time $t_{LOFF} = t_{NOM}/2^7$. With the emulator configured and the load connected, the output of channel 1 was turned on, and the output voltage V_{OUT1}, output current I_{OUT1} and the battery voltage V_{BAT} were recorded using the oscilloscope. An example of the recorded signals is in figure. The measured values are in table 7.2, where they can be compared against the expected ones ($t_{NOM}/2^7$). As you can see, the measured latch off times are close to the expected values, with slight deviations that are within tolerance of the VNF7000AY's integrated I^2t protection, which can be found in the datasheet [3]. This measurement proves that the emulator's nominal current time setting works as specified.

FTS [-]	t_{NOM}[s]	$t_{NOM}/2^7$[ms]	Measured t_{LOFF}[ms]
0	1	7.81	7.82
1	44	344	330
2	86	672	642
3	129	1008	964
4	172	1344	1284
5	214	1671	1600
6	257	2008	1910
7	300	2344	2230

Table 7.2: Latch off times t_{LOFF} for a fixed FCS and varying FTS settings

The FCS were validated in a similar way, with a constant nominal time setting $t_{NOM} = 1\,\text{s}$ (FTS=0) and varying nominal current settings I_{NOM}. Output channel 1 of the emulator was connected to the PSB 9080-240 in a constant current load mode, and the load current I_{OUT1} was increased with the FCS setting, always keeping the load current higher than the nominal current setting. The latch off times were measured and compared to the expected values according to the I^2t curve definition. The measured values are in the table 7.3. The measured values are again close to the expected ones, and the deviations are within tolerance of the protection mechanism.

FCS [-]	I_{NOM}[A]	I_{OUT1}[A]	Expected t_{LOFF}[ms]	Measured t_{LOFF}[ms]
9	1	23.5	7.81	7.80
11	2	23.8	15.6	15.2
14	3	23.8	31.2	29.8
16	4	33.3	15.6	15.2
19	5	49.5	7.81	7.84
25	6	49.5	15.6	15.3
27	7	33.75	62.5	60.1
32	0	33.75	500	453

Table 7.3: Latch off times t_{LOFF} for a fixed FTS and varying FCS settings

As part of the evaluation, the I^2t curve was measured for one FTS and FCS setting combination, specifically FTS=0 ($t_{NOM} = 1\,\text{s}$) and FCS=1 ($I_{NOM} = 9\,\text{A}$). The test setup remained the same, but the current load was gradually increased to cover every step of the I^2t curve of the VNF7000AY. For each of these steps, the output current I_{OUT1} and latch off time t_{LOFF} were measured, the results of which are in table 7.4. The measured values are also plotted in the graph in figure 7.7, together with the expected curve as it is defined in the VNF9D1M5's datasheet [5]. The graph indicates that the measured curve does not fit the specification exactly, as the latch off time are slightly shorter than they should be for the given currents. This could have been caused by several factors. The VNF7000AY has not yet been fully characterised, meaning that the device's performance might not match the datasheet exactly. The results could also have been influence by the measurement method, especially the electronic load and power supply.

In figure 7.8 you can see an image of the oscilloscope screen captured during one of the I^2t curve validation measurements. Recorded signals are battery voltage V_{BAT}, output voltage V_{OUT1}, control signal IN1 and the output current I_{OUT1}. In the output current waveform, the initial current peak caused by the output capacitors of the PSB 9080-240 can be observed. There is no apparent change in the output voltage, which is caused by precharging the output capacitors of the electronic load in order to reduce the initial current spikes.The main goal of this measurement was to verify that the I^2t protection works and that was done successfully.

I_{OUT1}[A]	t_{LOFF}[ms]
8.0	∞
9.2	412
9.4	246
10.2	118
11.5	60.6
14.0	30.9
17.0	16.4
23.0	8.54
32.0	4.82
49.5	2.13
57.5	1.28
85.6	0.74
94.3	0.76

Table 7.4: I^2t stairstep curve measurements for fixed FTS=0 and FCS=1 configuration

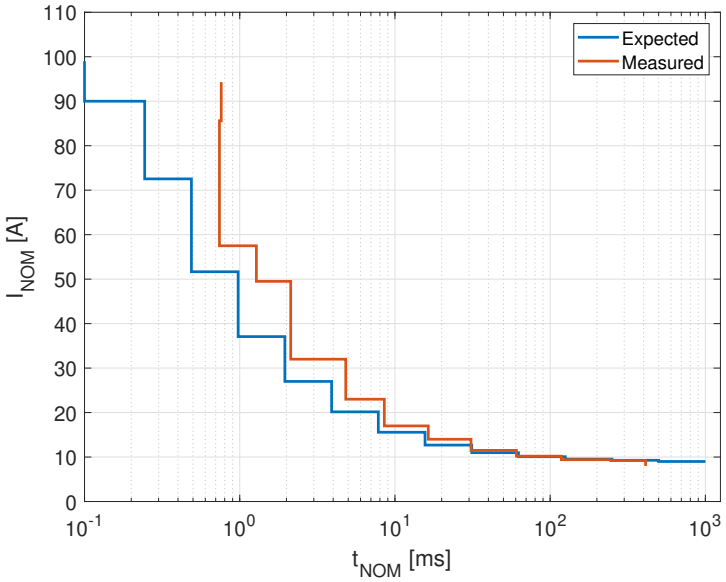


Figure 7.7: Comparison of the expected and measured I^2t curves

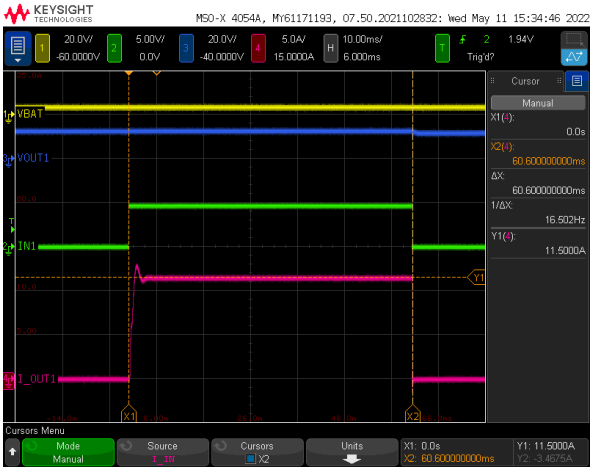


Figure 7.8: A captured image of one of the I^2t curve validation measurement

7.5 Validation of state machine implementation

One of the key parts of this project was the implementation of the VNF9D1M5's state machine into the emulator. In this section the evaluation of the state machine's implementation is documented. One of the main concerns during the design of the emulator was the impact of the MCU on the reaction times of the emulator with reference to the reaction times of the real device. The delays between the control and status signals changes and reactions to them were measured by generating the signals (or by inducing the conditions causing them) and observing the reactions of the emulator. The following signal reaction times were measured.

Output activation via direct input

The reaction time to direct input DI1 was measured by setting this signal high and observing channel control signal IN1 from the MCU to the VNF7000AY and the output voltage V_{OUT1} of channel 1. The recorded oscilloscope screen from this measurement is in figure 7.9. The measured delay between the direct input DI1 being activated and the MCU activating the output is

$$t_{DI} = 14.7\,\mu s.$$

This is an acceptable result, as the maximum turn off delay time of the VNF9D1M5 is specified at 50 µs when using direct input.

Figure 7.9: Reaction time to output activation via direct input

Limp home pin activation delay

The Limp Home (LH) pin is used to switch the device from normal mode to failsafe mode without needing to use the SPI communication or waiting for a timeout. The emulator was set up in normal mode with output channel 1 activated by writing 1 to the `SOCR1` bit of the `SOCR` register. When the device is in failsafe mode, the outputs are operated through the direct inputs only, regardless of the register content. When the LH pin was pulled high, the emulator switched to failsafe mode and therefore, the output was disabled. The measured reaction time was

$$t_{LH} = 40.2\,\mu s.$$

In figure 7.10, you can see the captured waveforms from the measurement.

Figure 7.10: Reaction time to Limp Home pin activation

Emergency stop pin activation delay

Emergency stop (EM) pins are intended to stop the output of the channel as fast as possible in case of an emergency. The emulator was set up in normal mode with output 1 activated via the `SOCR` registers, and then the emergency stop pin for channel 1 (EM1) was pulled high. The delay between the EM1 pin activation and output deactivation was measured to be

$$t_{EM} = 31.6\,\mu\text{s}.$$

The required maximum emergency stop pin reaction delay is not yet specified; however, the measured delay was deemed low enough for the emulator's operation. Screen capture from the measurement is in figure 7.11.

Figure 7.11: Reaction time to emergency stop pin activation

Emulator startup time

When using the emulator in the power distribution system, it is useful to know how long it takes for the emulator to initialise after powering up. The emulator FW was modified to generate a positive edge on the TEST0 pin of the emulator, once the

initialisation phase has been finished. The time between the V_{CC_UC} voltage being available and the edge on the TEST0 pin was measured. The measured startup time was

$$t_{STARTUP} = 8.2\,\text{ms}.$$

A screenshot of the measurement is in figure 7.12. There is no specific requirement for the emulator's startup time, but it is an important piece of information for the emulator's documentation.

Figure 7.12: Emulator startup time measurement

Fault propagation time

When one of the VNF7000AYs detects a fault, their STATUS pin is pulled low, or the MultiSense output voltage is set to V_{SENSEH}, depending on the type of fault. When one of these fault indicators is detected on a channel, the LATCH_DIAG signal for that channel is pulled low. The delay between the activation of the fault indicator and the activation of the LATCH_DIAG signal is the fault propagation delay. The emulator was configured in normal mode with output channel 1 active. A short-circuit was caused on the active output of the emulator, and the STATUS1 and the LATCH_DIAG1 signals were measured. The observed fault propagation time was:

$$t_{FAULT} = 31\,\mu\text{s}.$$

An oscilloscope screenshot from the measurement is in figure 7.13. The fault propagation time is within the expected range and should not affect the performance of the emulator in any meaningful way.

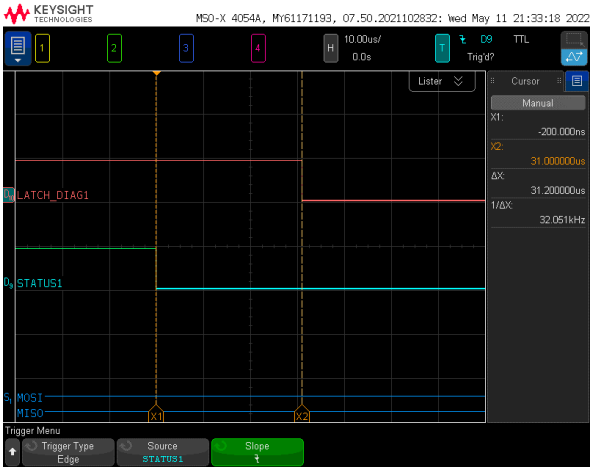


Figure 7.13: Fault propagation time measurement

Output activation delay after SPI activation command

In normal mode, the emulator's outputs can be controlled via SPI commands. The delay between the write SPI operation and the activation of the output was measured by monitoring the SPI bus, the output voltage of channel 1 V_{OUT1} and the control signal IN1 for the VNF7000AY with an oscilloscope. The measured delay between the SPI frame's end and the activation of the output was:

$$t_{SPI\_OUT} = 44.0\,\mu s.$$

An oscilloscope screenshot of this measurement is in figure 7.14.

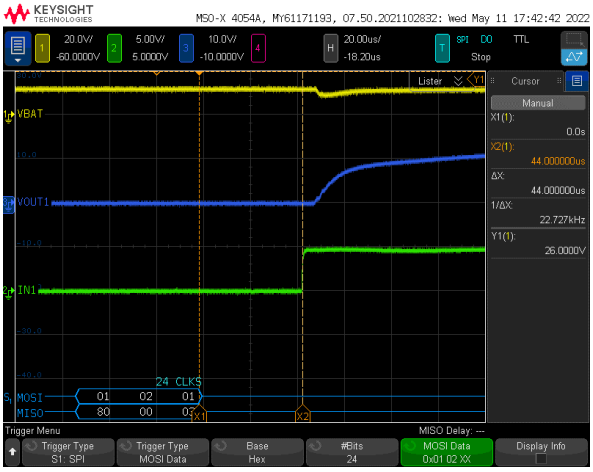


Figure 7.14: Delay between an output activation SPI command and output activation

7.6 Digital current sense and voltage monitoring performance

Another important feature of the VNF9D1M5 is the digital current sense and voltage monitoring. The emulator uses its internal ADC to convert the analog feedback signals to digital values that are then available in the appropriate registers. The datasheet of the real device defines the coefficients used to convert the raw digital values to current and voltage values. The emulator's design may have caused these coefficients to be different. During this part of the evaluation, the conversion coefficients were determined as well as the minimum and maximum measurable currents.

On the emulator, the output current of the channel I_{OUTx} is measured by measuring the output current from the MultiSense pin I_{SENSEx}, which is proportional to the output current ($I_{OUTx}/I_{SENSEx} = 25500$). The sense current is measured by measuring the voltage drop V_{SENSE} across a current sense resistor $R_{SENSE} = 1\,\mathrm{k\Omega}$. This voltage is converted to a digital value using the 10 bit ADC. The ADC reference voltage is $V_{CC\_UC} = 4.7\,\mathrm{V}$ (V_{CC_UC} should be 5 V, but there is a reverse battery protection diode, which causes a slight drop). From this, the expected current sense conversion factor k_I can be determined:

$$\begin{aligned} k_I = \frac{I_{OUT}}{\texttt{ADCISR}} = \frac{25500 \cdot I_{SENSE}}{\texttt{ADCISR}} = 25500 \frac{V_{SENSE}}{R_{SENSE} \cdot \texttt{ADCISR}} = \\ = 25500 \frac{V_{CC\_UC} \frac{\texttt{ADCISR}}{2^{10}}}{R_{SENSE} \cdot \texttt{ADCISR}} = 0.117\,\mathrm{A}, \end{aligned} \tag{7.1}$$

where `ADCISR` is the value contained in the Digital Current Sense register. The characteristic of the digital current sense was measured by connecting a load to the output channel 0 of the emulator and gradually increasing the load current (PSB 9080-240 in load mode was used), while monitoring the values in the digital current sense registers `ADCISR`. The measured values are in table 7.5 and also plotted in a graph in figure 7.15. A linear function was fitted to the data, with the following result:

$$\texttt{ADSISR} = 7.93 \cdot I_{OUT} + 5.34 \tag{7.2}$$

There is a slight offset that was probably caused by a measurement error. The inverse value of the linear term gives the conversion factor from `ADCISR` value to an output current

$$k_{Imeas} = \frac{1}{7.93} = 0.126\,\mathrm{A}.$$

This value is slightly different from the calculated one, but that could have been caused by tolerances in resistor value or in the I_{OUT}/I_{SENSE} ratio. This value will be listed in the emulator's documentation. The measured curve is close to the linear fit, with a slight deviation at the end. This was probably caused by saturation in the current sense mechanism of the VNF7000AY, or saturation of the ADC. The linearity of the current sense mechanism was evaluated, and the relative full-scale linearity error is $\delta_L = 1.25\%$.

I_{OUT}A	ADCISR [-]
0.27	7
0.9	11
1.3	14
2.2	22
5.3	46
10	86
20	165
30	246
40	317
60	485
80	645
100	802
120	965
122	982
128	1008
129	1023

Table 7.5: Results of the digital current sense evaluation measurement

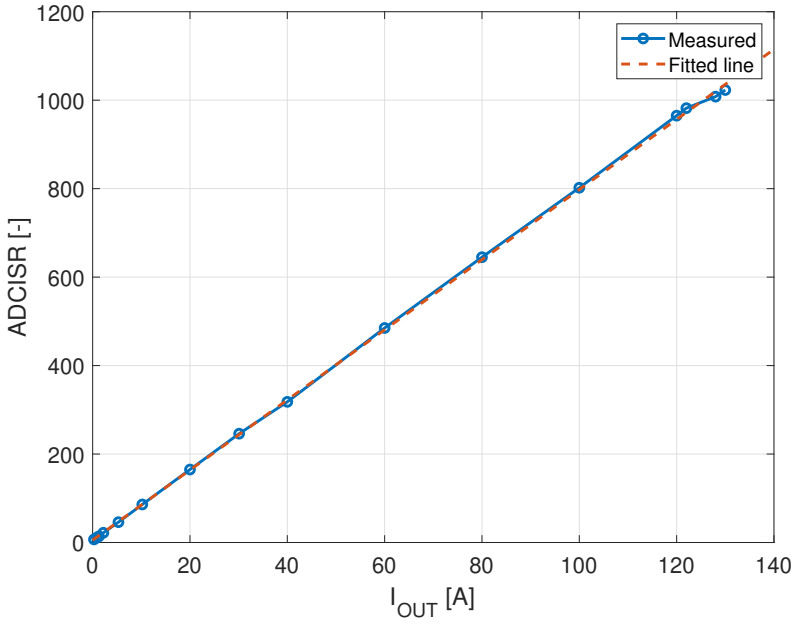


Figure 7.15: Output current vs. ADCISR value

Theoretically, the minimum current detectable by the emulator produces ADCISR=0x001 and therefore is $k_{Imeas} \cdot 1 = 0.126\,\text{A}$. However, due to noise and other interference, the first current value that produced a discernible change in ADCISR value was 0.27 A. The maximum measurable current produces ADCISR=0x3FF should correspond to output current of $k_{Imeas} \cdot 1023 = 128.9\,\text{A}$ which is corresponds to the measured values.

When it comes to voltage measurements, the battery and output voltages are connected to the ADC through resistor voltage dividers with a ratio of 1 : 6. The theoretical voltage conversion factor is:

$$k_V = 6 \cdot \frac{V_{CC\_UC}}{1024} = 27.5\,\text{mV}. \tag{7.3}$$

The voltage measurement characteristics were measured at one point for each measured voltage. The measured values are in table 7.6, together with the corresponding conversion factors. The measured conversion factors are close to the expected values, with slight deviations caused by component tolerances or measurement noise.

	Measured voltage [V]	Emulator output [-]	k_{Vmeas}[mV]
V_{BAT}	13.34	478	27.9
V_{OUT0}	13.38	477	28.1
V_{OUT1}	13.37	478	28.0

Table 7.6: Measured voltages vs. values given by the emulator

One of the required measurements was to determine the sampling rate of the ADC, or more precisely, the frequency with which the values in the digital current sense and digital voltage sense registers are refreshed. This measurement was carried out by modifying the FW of the emulator to toggle the output value on the TEST0 pin of the emulator every time the values in registers are updated. By measuring the time between the edges of the signal on the TEST0 pin, the sampling rate of the emulator was determined to be:

$$f_s = 38.5\,\text{kHz}.$$

An oscilloscope screenshot from this measurement is available in figure 7.16. This sampling frequency is much lower than the sampling frequency of the real device (400 kHz [5]), but it is sufficient for the emulator's purposes.

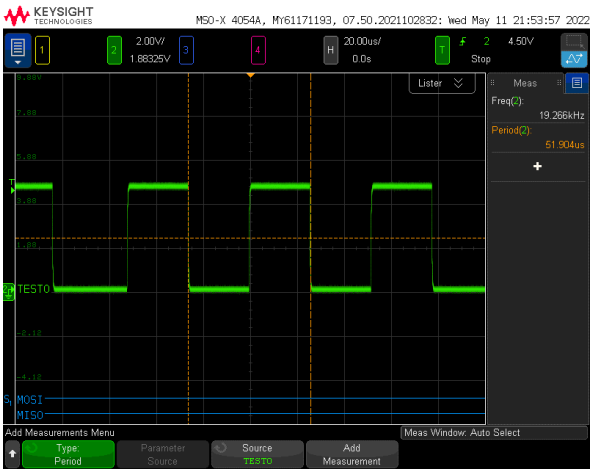


Figure 7.16: Measurement of emulator's ADC sampling rate

7.7 SPI frame processing time

As was extensively discussed in the design requirements 4 and software design 6 chapters, the emulator requires a short pause between the first and second bytes of the SPI message. During the evaluation, a modified version of the FW was loaded into the emulator's MCU, that pulls the TEST0 pin high while a byte of the SPI message is being processed. In figure, you can see an oscilloscope screenshot with the SPI message and signal from the TEST0 pin. The pause between the bytes of the message is $t_{PAUSE} = 10\,\mu s$ and can be changed in the master application's firmware. The measured time required for the MCU to process the byte is:

$$t_{SPI\_PROC} = 2.2\,\mu s.$$

An oscilloscope screenshot of this measurement is in figure 7.17. The information about the time required to process the SPI message will be a part of the emulator's documentation as it is important for the design of the software controlling the emulator.

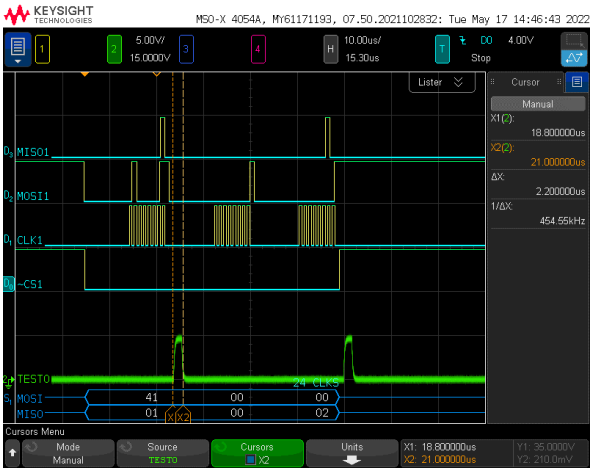


Figure 7.17: Processing time of the first byte of an SPI frame

Chapter 8
Conclusion

The main goal of this thesis was to design and implement an emulator of advanced functions of an electronic fuse currently in development (VNF9D1M5). The emulator is based on an already existing electronic e-fuse device (VNF7000AY), and the additional features are emulated by a microcontroller.

To determine the requirements for the emulator, the parameters of the VNF9D1M5 were analysed. These included the device's electrical characteristics, such as maximum supply voltage, maximum output current, and required behaviour, such as reactions to specific inputs or conditions on the output of the device. During this stage, the emulator's limitations were identified. Solutions to some of these were proposed, for instance, the need for a delay between the first two bytes of the SPI frame to allow the MCU to prepare a response was identified during the requirement analysis. This part of the project was slightly complicated by the fact that because the VNF9D1M5 is still in development, its specification changed several times during the design of the emulator. These changes were incorporated into the current design whenever they occurred.

Based on the requirements identified during the analysis of the VNF9D1M5, the hardware was designed. The first step of hardware design was to select the components that would make up the emulator. The components for the high power section were already selected as a part of the assignment. The hearth of the emulator is the microcontroller, which emulates the additional e-fuse features. The SPC582B from the Chorus family of 32-bit Power architecture automotive MCUs was chosen as it fulfilled all of the requirements for the emulator. Another essential component to be selected was the digital potentiometers that are used to configure the VNF7000AY e-fuse parameters. From the available devices, the AD5162 digital potentiometers were chosen because they had the required operating temperature range and were available in suitable voltage ranges. Appropriate connectors had to be chosen to carry the large currents to and from the board. It was decided to use the REDCUBE Plug connectors because they have sufficient current rating. An electrical schematic was drawn up from the selected components, and additional components needed to make everything work. Based on this schematic, the PCB was designed. The PCB design was influenced by the requirements, especially by the required current carrying capacity, which is why the board has four 70 µm-thick copper layers. This approach also helped to improve the thermal properties of the emulator as the additional layer helps to dissipate heat from the power components.

With the hardware finished, the work on the software for the microcontroller (firmware) was started. The VNF9D1M5's behaviour is controlled by a finite state machine that runs inside the device. This state machine was implemented in the FW for the emulator's microcontroller, with some limitations. The VNF9D1M5 can be controlled,

and its state is diagnosed via SPI access to its registers. These registers were also implemented in the emulator, with some limitations caused by the emulator's construction. For example, the VNF9D1M5 provides access to the counter register of the I^2t protection mechanism. This feature is not available on the emulator because the MCU has no access to the counter inside the VNF7000AY device.

After the emulator's completion, it was necessary to perform an In Application Validation, which was meant to verify that the requirements were met and the emulator was performing as expected. Several measurements were made, and their results evaluated, concluding that the emulator performs within specification. This process also helped to uncover several errors in the design of the software and firmware, which were fixed before the device development was finalised.

A device for emulation of the advanced functions of a future electronic fuse was developed and realised in this project. The emulator has the same characteristics as the real device with several limitations that are listed in its documentation. This was confirmed in the validation and evaluation measurements. This means that all of the thesis assignment goals were satisfied. In the future, some of the features currently missing from the emulator will be added to it by a firmware update. The hardware of the emulator might also be redesigned to be more compact to allow for easier integration into a vehicle.

Bibliography

[1] *Fuseology*. Littelfuse. 2021. URL: www.littelfuse.com.

[2] *Automotive Passenger Car Catalog*. Littelfuse. 2021. URL: www.littelfuse.com.

[3] *VNF7000AY Datasheet (pre-release)*. DS12798. Rev 2. STMicroelectronics. Mar. 2021. URL: www.st.com.

[4] *High Accuracy Current Sense of Smart High Side Switches*. SLVAE08. Texas Instruments. Nov. 2019. URL: www.ti.com.

[5] *VNF9D1M5 Datasheet (pre-release)*. DS13906. Rev 1. STMicroelectronics. Mar. 2022. URL: www.st.com.

[6] *SPC582Bx Datasheet*. DS11597. Rev 4. STMicroelectronics. Oct. 2020. URL: www.st.com.

[7] Ribbens, William B. *Understanding Automotive Electronics*. 7th ed. Elsevier, 2012. ISBN: 978-0-08-097097-4.

[8] *AD5162 Dual Digital Potentiometer Datasheet*. AD5162. Rev C. Analog Devices. Dec. 2010. URL: www.analog.com.

[9] *WP-PLUG REDCUBE PLUG Datasheet*. 7464000. Rev 002.000. Würth Elektronik. May 2021. URL: www.we-online.com.

[10] Záhlava, Vít. *Návrh a konstrukce desek plošných spojů*. 1st ed. BEN - technická literatura, 2011. ISBN: 978-80-7300-266-4.

[11] *SPC582Bx IO definition: signal description and input multiplexing tables*. TN1263. Rev 3. STMicroelectronics. Nov. 2020. URL: www.st.com.

[12] *SPC58xx hardware design guideline*. AN4880. Rev 8. STMicroelectronics. Oct. 2021. URL: www.st.com.

[13] *SPC58 2B Line 32 bit Power Architecture automotive MCU Reference Manual*. RM0403. Rev 7. STMicroelectronics. June 2021. URL: www.st.com.

[14] *Electric and Electronic Components in Motor Vehicles up to 3.5 t General, Requirements, Test Conditions, and Tests*. VW80000. Rev 3. Volkswagen. Oct. 2017.

[15] Letor, Romeo and Crisafulli, Roberto. "Smart Power devices and new electronic fuses compliant with new E/E architecture for autonomous driving". In: *2019 AEIT International Conference of Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*. 2019, pp. 1–6. DOI: 10.23919/EETA.2019.8804538.

Appendix A
Signal Description

Signal Name	Signal Description
SPI1 CLK	SPI clock signal for digital potentiometers
SPI1 SDO	SPI data output signal from the MCU to the digital potentiometers
SPI1 CS1	SPI chip select for the FTS digital potentiometer
SPI1 CS3	SPI chip select for the FCS digital potentiometer
SPI3 CLK	SPI clock signal for digital potentiometers
SPI3 SDO	SPI data output signal from the emulator MCU to the master
SPI3 SDI	SPI data input signal from the master to the emulator MCU
SPI3 CS0	SPI chip select of the MCU
DI0, DI1	Direct inputs for controlling channel outputs, active high
EM0, EM1	Emergency inputs that disable the channel ouptuts, active high
LH	Limp Home pin switches device to failsafe mode, active high
LATCH_DIAG0, LATCH_DIAG1	Latch off diagnostic, pulled low when channel latches off
IN0, IN1	VNF7000AY output control pin, active high
R_mode0, R_mode1	VNF7000AY high resistance mode select pin, active high
F_CTRL0, F_CTRL1	VNF7000AY fuse control pin, enables e-fuse functionality, active high
STDBY_ON0, STDBY_ON1	VNF7000AY standby pin, switches device to low power mode, active high
STATUS0, STATUS1	VNF7000AY diagnostic pin, indicates a fault, active low (open drain)
Sen	VNF7000AY Sense enable pin, enables output of the MultiSense pin, active high
SEL0, SEL1	VNF7000AY MultiSense output selection pins
MS0, MS1	VNF7000AY MultiSense analog feedback output pin
FCS0, FCS1	VNF7000AY Fuse Current Set pin, connected to R_{FCS}
FT0S0, FTS1	VNF7000AY Fuse Time Set pin, connected to R_{FTS}

Table A.1: Description of signals available on the emulator

Appendix B
Emulator's electrical schematic and PCB layout

In this appendix the complete schematic and individual PCB layer drawings are available.

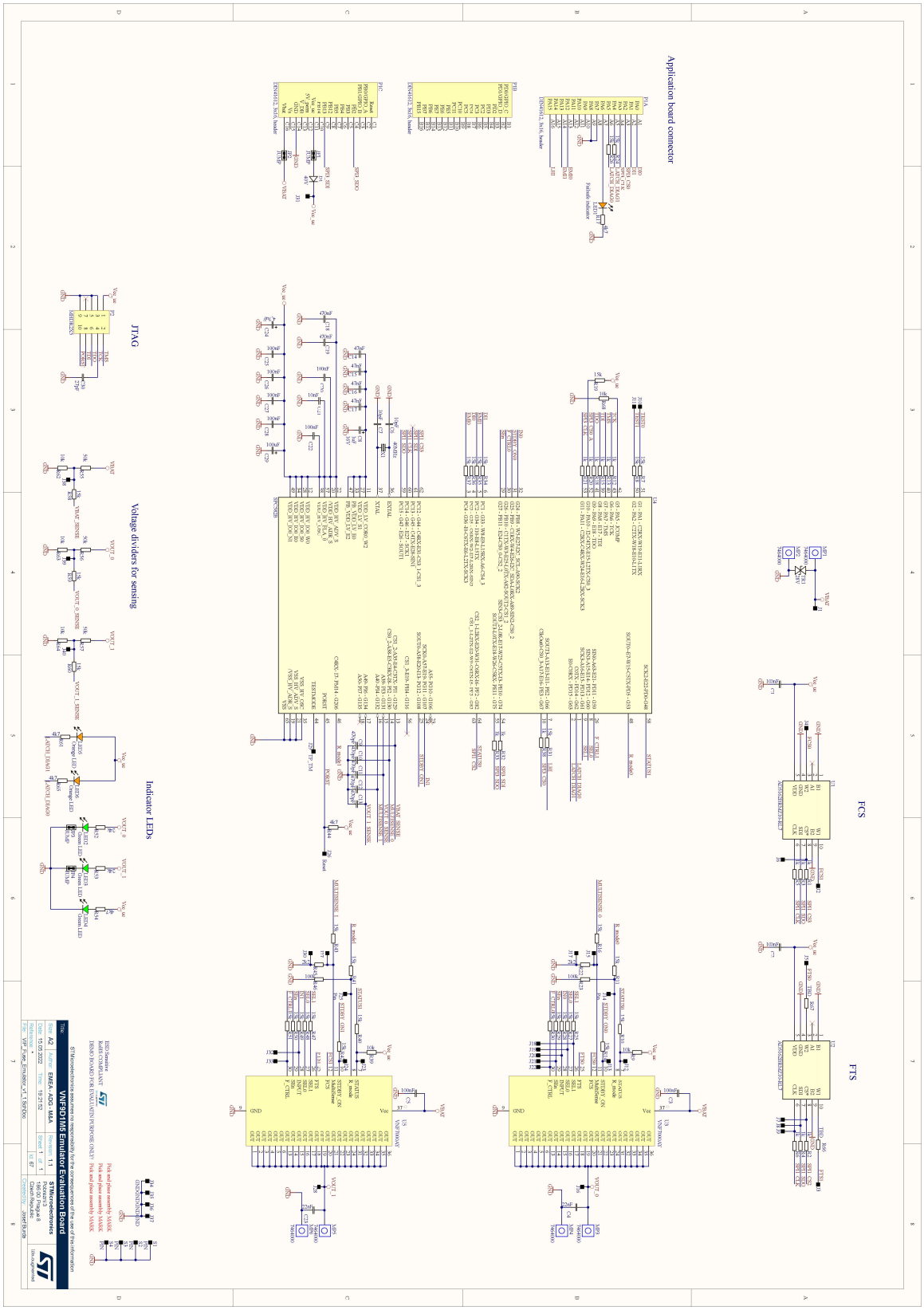


Figure B.1: Complete schmematic of the emulator

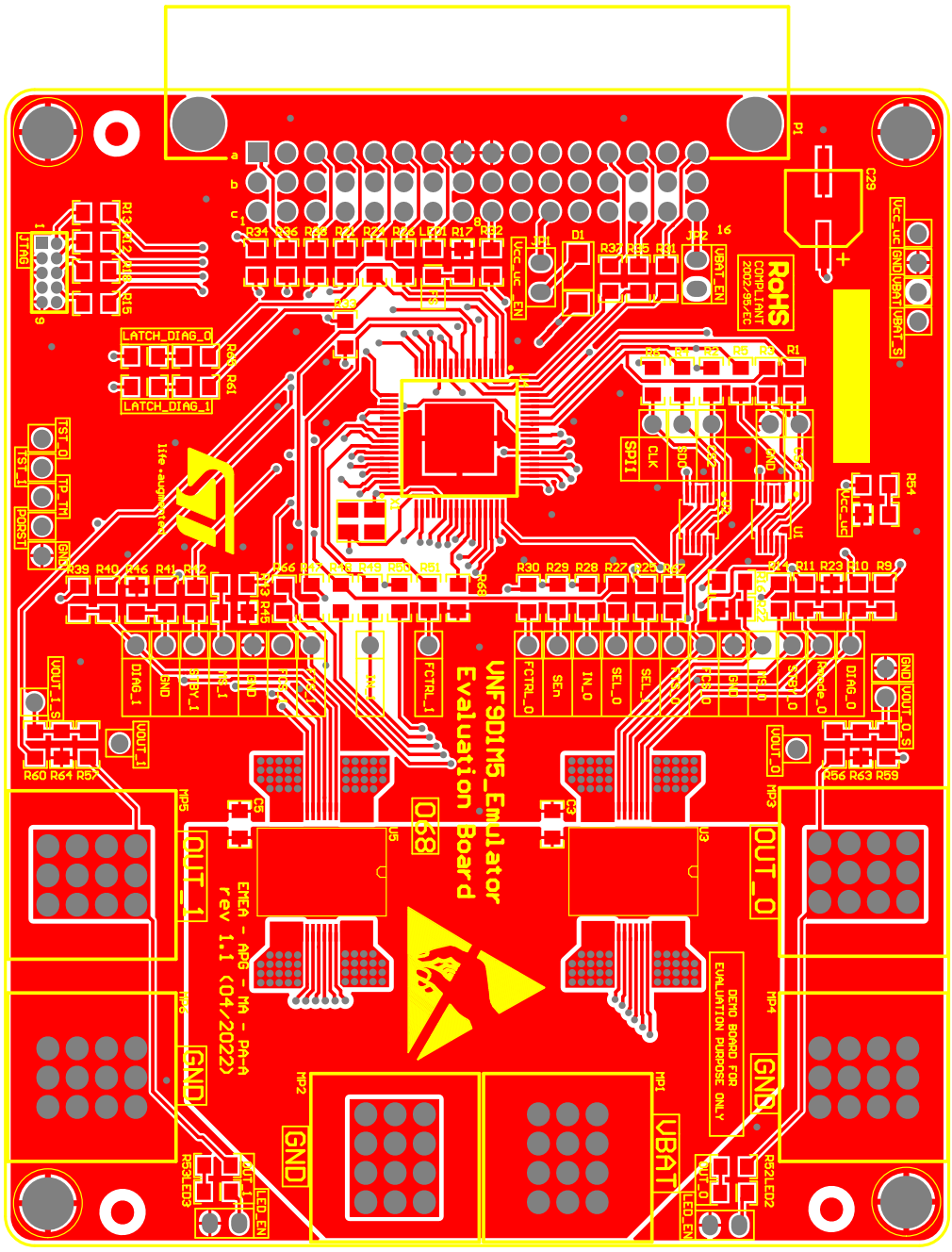


Figure B.2: Layout and silkscreen of the top layer of the PCB

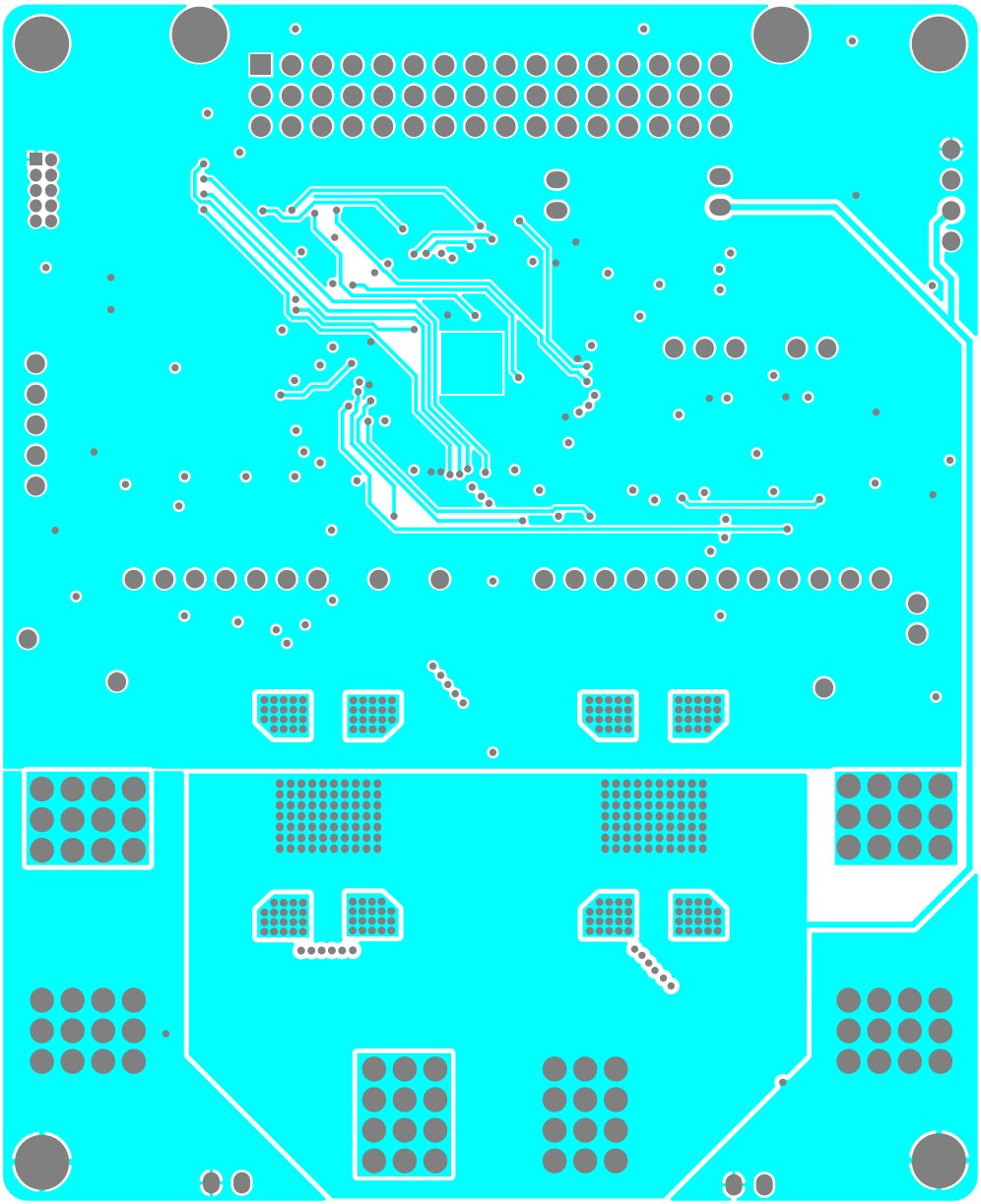


Figure B.3: Layout of the first inner layer of the PCB

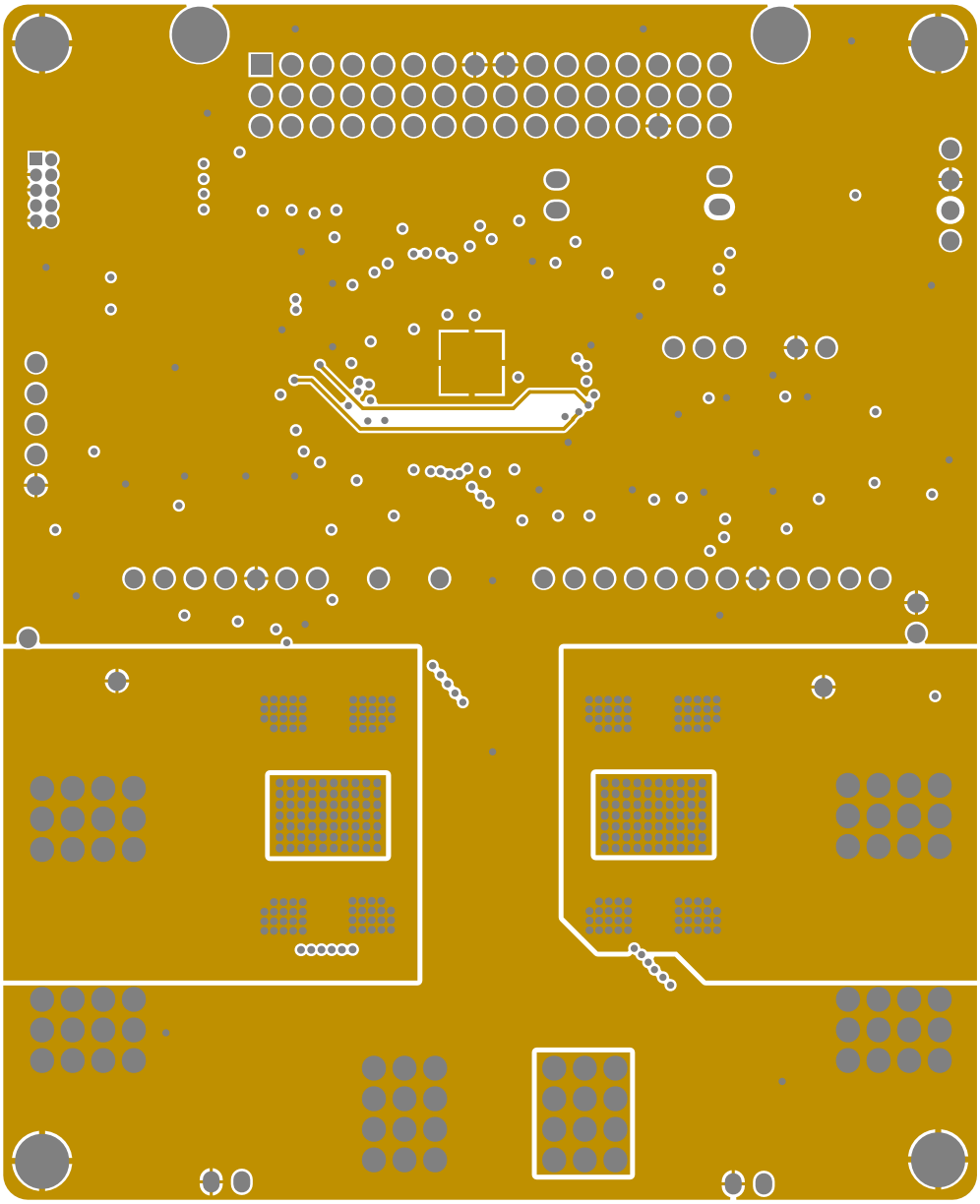


Figure B.4: Layout of the second inner layer of the PCB

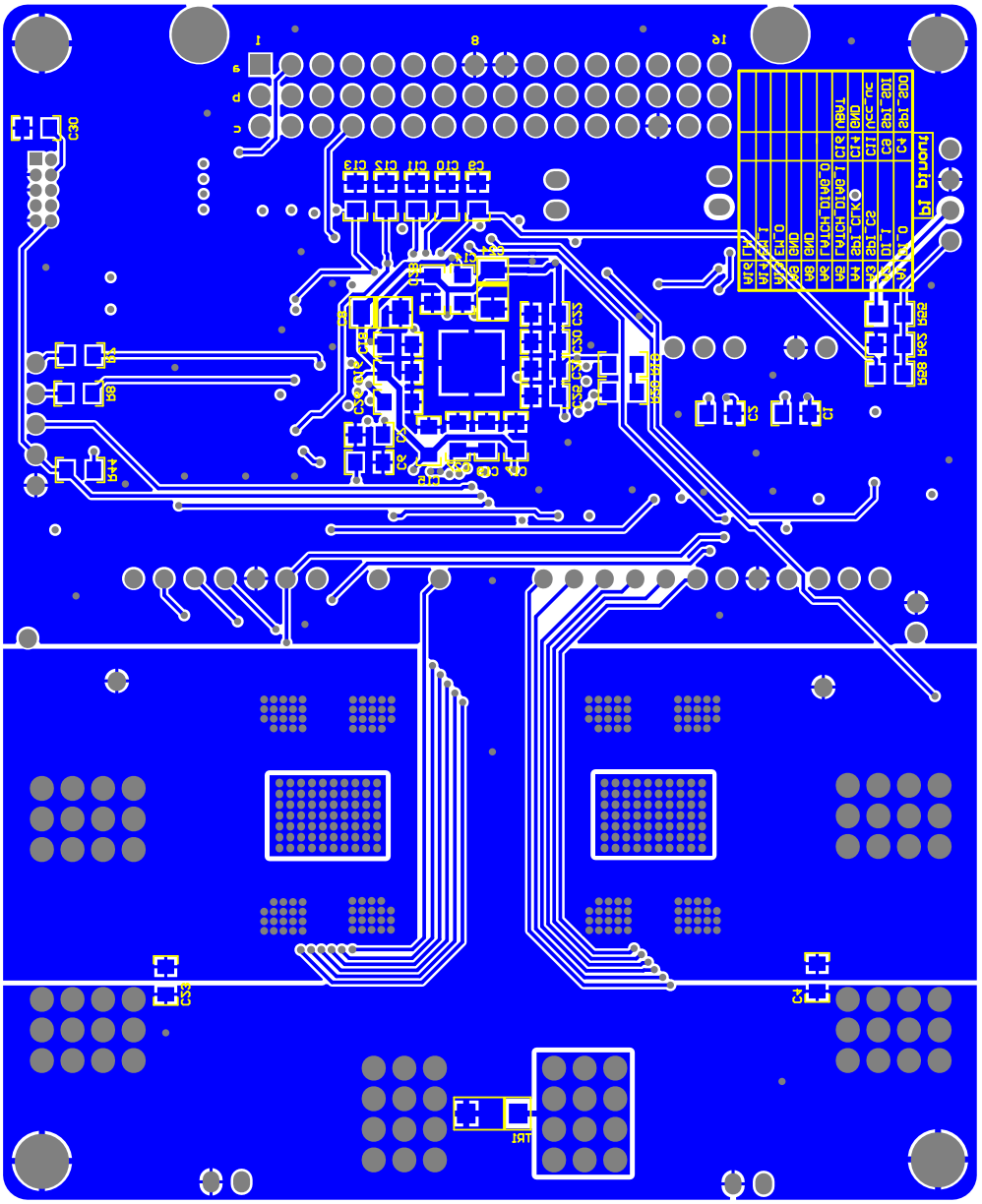


Figure B.5: Layout and silkscreen of the bottom layer of the PCB

Appendix C
Contents of the attached CD

/
- `Hardware` ..... PCB design files for Altium
- `Thesis` ..... Text práce
  - `sources` ..... LaTeXsource files and images
  - `DP_Burda_Josef_2022.pdf` ..... Thesis in PDF format