Master Controller Design Continuation

Formula Student Project 1 - EK-FSP1

Report by

Sebastian Rud Madsen

semad17 - 11/07/97

University of Southern Denmark

Supervisor

Gourinath Banda

Project Period

01/02-2022 to 01/06-2022

Date of Hand-in

01/06-2022

Sebastian Rud Madsen

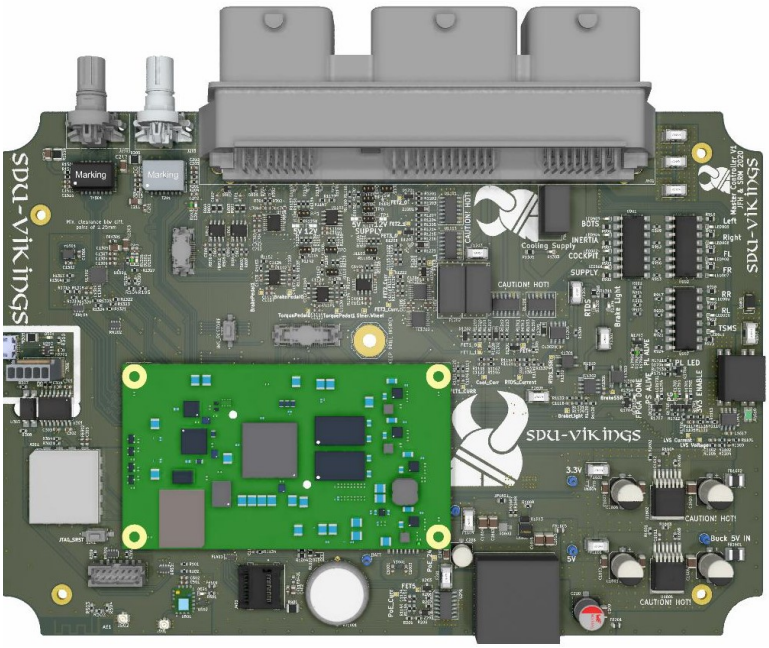


Table of Contents

1 Purpose

This project is a continuation of the work performed of [1]. In the project, the SDU-Vikings Master Controller requirements were analyzed, and hardware and PCB designed in KiCad 5. Most circuits were tested, during which multiple programs were written for the Zynq-7020 in Xilinx Vivado and Vitis.
Some circuits were not tested originally due to time constraints. This projects aims to complete the tests and improve the hardware and PCB design. Additionally, a few circuits failed testing, and the designs are to be revised.
To further test the software capabilities, part of the Zynq-7020's functionality is then to be added. This is the Embedded Linux-based liveview, which hosts a webpage containing car data, accessible over Wi-Fi. When fully operational, only Zynq core 1 will be used for Embedded Linux, and will initialize the bare-metal software on core 2, which handles sensors, state machines, communication, and data logging. However, this is outside the scope of this project, which will simply host the liveview using both cores 1 and 2 as proof-of-concept. The Embedded Linux will be booted from an SD card.

2 Reading Guide

The report is split into testing, software, and hardware sections, containing both information from the bachelor thesis [1] as well as new design. The bachelor thesis will be referenced heavily, and while it is assumed that the reader knows its contents, the important terminology has been repeated in Section 10. Additionally, the Master Controller's functionality will be briefly described in the next section.

Both the hardware and software files are available on the private SDU-Vikings GitLab in the following repositories:
https://gitlab.com/sdu-vikings/future/master-controller
https://gitlab.com/sdu-vikings/future/master-controller-software

Both are also temporarily hosted in the public GitHub repository below:
https://github.com/SR-Madsen/VikingsMasterController

3 Master Controller V1

A block diagram of the Master Controller's responsibilities is shown in Figure 1. This shows that the Master Controller has many responsibilities, among the most important being sensor measurement and actuator handling. These are all centered around the Zynq-7020 SoC, which is placed on the PicoZed SoM daughterboard.

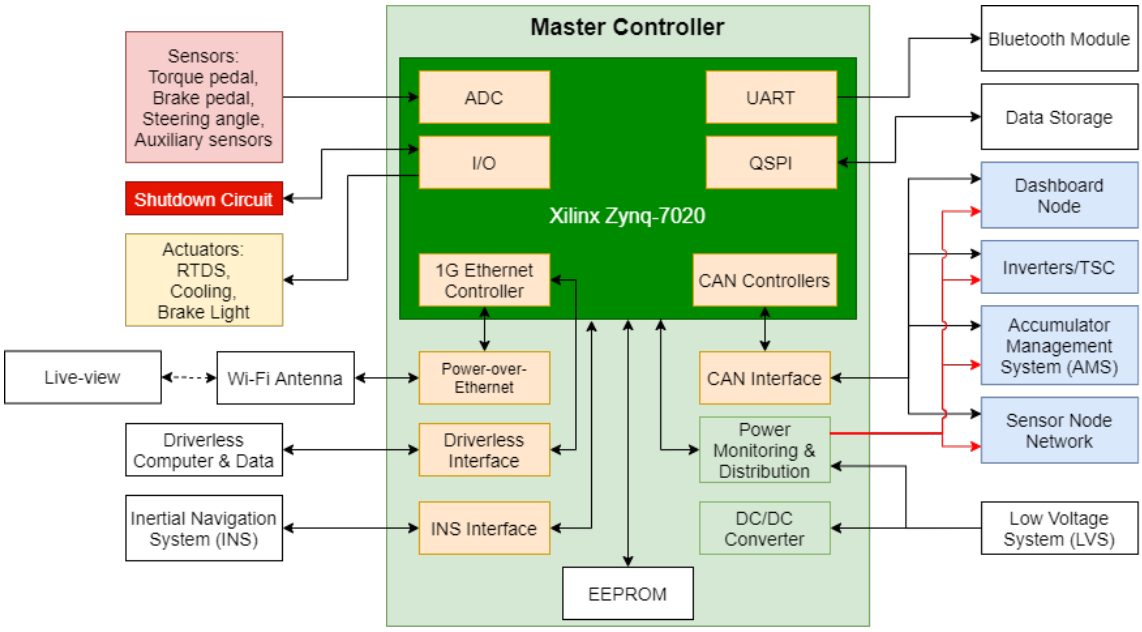


Figure 1: Block diagram of the Master Controller, showing external modules

During the design process, a large focus was put on modularity and flexibility. For this reason, it is possible to supply multiple voltages to sensors, and thus various voltages may be returned. Pin headers with jumpers are used to select the output voltages and input voltage dividers.

The Zynq I/O is used to monitor the car's shutdown circuit, consisting of a large amount of switches and interlocks, for errors. An overview of the shutdown circuit is shown in Figure 2. The signals are powered by the 24 V low-voltage battery, and thus have been isolated and level-shifted to 3.3 V, which is compatible with the Zynq.

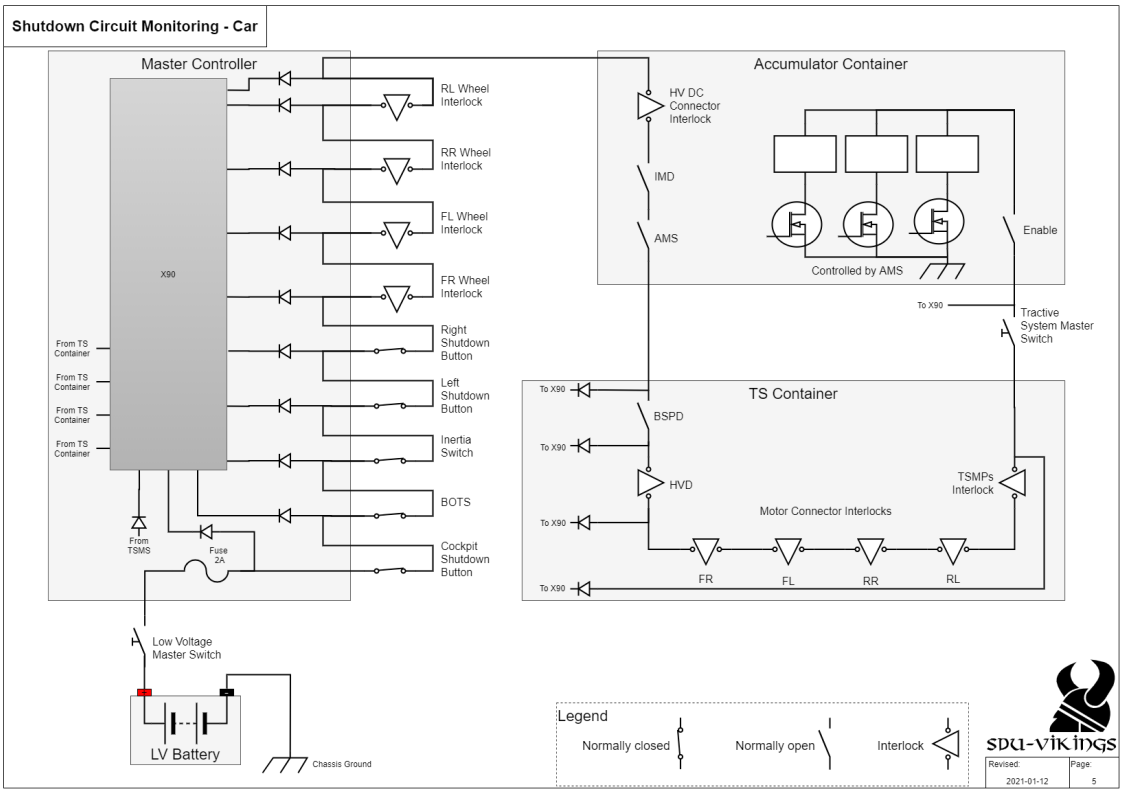


Figure 2: Overview of shutdown circuit for X90-based car setup

The Master Controller has hardware for communication through CAN, UART, Bluetooth, Wi-Fi through the Wireless Access Point which it also powers, and Ethernet. It is also responsible for distributing the power to other nodes, such that it will always be the first to power up, as well as can disable other nodes when an error occurs.

Some focus has been put on EMC when designing the PCB. Ferrite beads have been used for most IC supplies, and a common-mode choke is used to filter the low-voltage supply. Additionally, capacitors have been placed to minimize current loops by switching supplies and current transients in long traces.

The Master Controller fits in a highly IP-rated case, and thus the connectors are also rated as such. For Ethernet, circular connectors are used, while a Molex CMC connector is used for all other signals entering or leaving the Master Controller. These can be seen in the top of the PCB shown in the figure on the front page.

4 Testing

Neither Bluetooth UART to a PC, nor Wi-Fi through the Wireless Access Point has been tested. Additionally, the UART communication with the INS has not been tested, but as the INS is still unavailable, this will not be part of the testing section.

As such, the Bluetooth and Wi-Fi communication will be tested before any design changes are made.

4.1 Bluetooth

A Bluetooth test is setup in Vivado 2020.1 by enabling the Zynq UART and the Bluetooth GPIOs, being pin D19 and pin D20. The block diagram is shown in Figure 3. During initialization in Vitis 2020.1, using the UART and PL GPIO drivers written in [1], the GPIOs are both set to logic '1', enabling the RN4871U Bluetooth module in application mode as described in the datasheet [2].

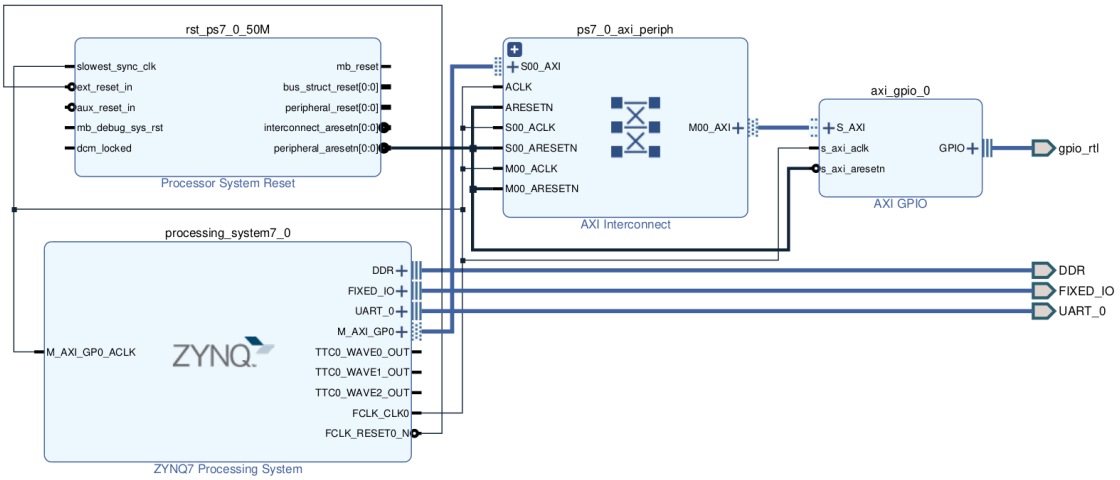
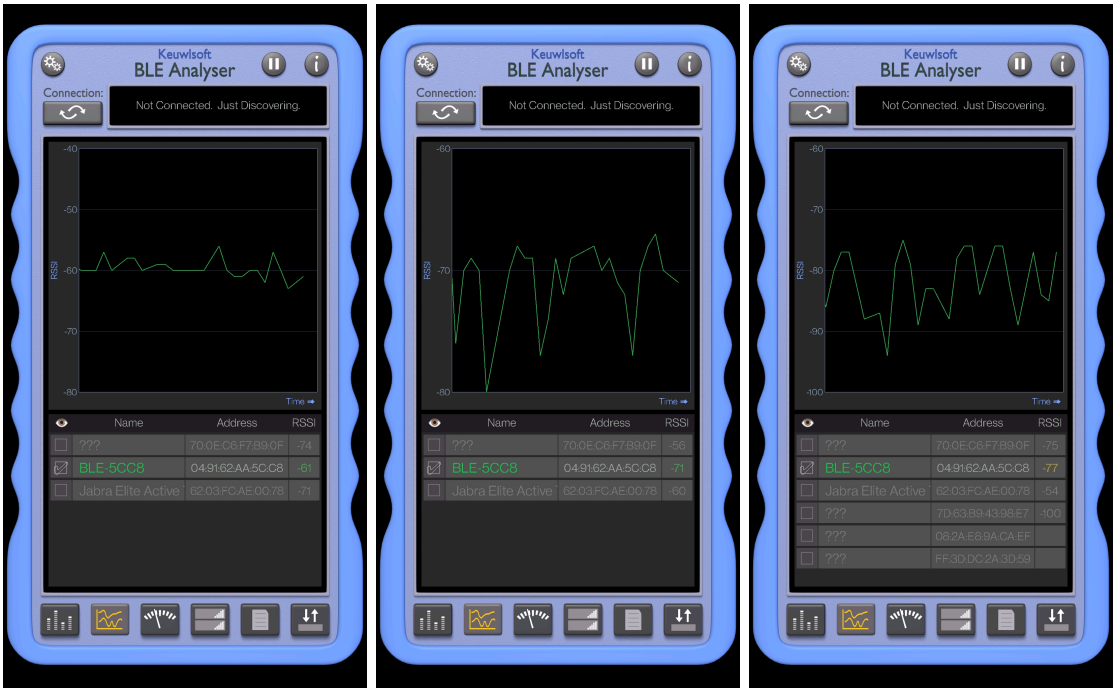


Figure 3: Vivado block diagram used for Bluetooth test setup

Initially when powering on, the red LED D501 indicating On/Off flashes once every few seconds, indicating that the module is powered.
Using the BLE Analyzer phone application[1], it is noted that the module advertises every second by default. The Received Signal Strength Indicator (RSSI) is measured at various distances with the app, roughly 30 cm above the module. The resulting graphs are shown in Figure 4. At longer ranges, it is noted that there is much larger variance in the signal.

[1] https://play.google.com/store/apps/details?id=com.keuwl.ble&hl=da&gl=US



(a) RSSI at 30 cm (b) RSSI at 5 m (c) RSSI at 11 m

Figure 4: Graphs of the measured advertisement signal strength at 30 cm, 5 m, and 11 m

The estimated averages at 30 cm, 5 m, and 11 m is -60 dB, -75 dB, and -83 dB respectively. It thus appears that the PCB antenna functions well.

It is found that pin 13 of the RN4871U, UART RX Indication, is left floating and pulled up internally. This means that the module is in low-power mode, and cannot be communicated with through UART. The same is true for pin 8, UART_CTS, which means that the Bluetooth module will not read the UART RX input. A temporary fix is implemented by soldering wires to ground, but PCB fixes must also be implemented.

Occasionally, the Zynq pin D20 cannot be pulled low, and thus the RN4871U cannot be forced to enter test mode. This may be due to a bad connection in the JX header, which should be pushed down firmly. Alternatively, the pin can be used as input with pull-down resistor, pulling the voltage to 0.5 V, which will be registered as low [2, p. 11].

Using a PC to successfully pair with the module, the LED indicates the new status by blinking twice in rapid succession every few seconds.
Even with the pairing made and all fixes implemented, it is not possible to open a UART connection with the module. The connection has been tested with multiple operating systems. Additionally, the module cannot be configured by the Zynq, despite pulling pin D20 low and the UART TX signal being transmitted correctly. Using Microchip's

Bluetooth Smart Discover app[2], it is revealed that the Bluetooth module does not advertise as Microchip BLE with the necessary device information, and as such is not recognized as a communication module.

Based on the testing, as well as posts on the Microchip forum[3] [4] [5], it appears that the module firmware, v1.18, is unreliable and the source of the issue. The firmware cannot be updated to the newest v1.42, as no header has been placed on the PCB for this purpose. As such, Bluetooth data transfer cannot be tested with the current PCB, but the advertisement strength hints that it would be successful.

Due to the chip crisis, it is not possible to purchase a new Bluetooth module. Additionally, the RN4871U cannot be found with a newer firmware version than v1.18. For this reason, no further testing can be completed.

The fixes mentioned in the section must be implemented on the PCB. These are:

- Connect UART CTS and UART RTS through 0 Ω resistors to the Zynq
- Connect UART RX Indication to the Zynq, such that low-power mode can be utilized
- Add header connected to necessary signals [2, p. 27] for firmware programming

4.2 Wi-Fi

The Wireless Access Point used is the Ubiquiti NanoStation AC loco [3], which is powered with 24 V Power-over-Ethernet activated by the Zynq. The Zynq hardware for the test is like in [1, p. 78], but with a PL GPIO used for activating the 24 V switch. Instead of connecting the PC to the M12 connector, a secondary WAP is used as transceiver for the PC, which is wirelessly connected to the Master Controller WAP as shown in Figure 5.

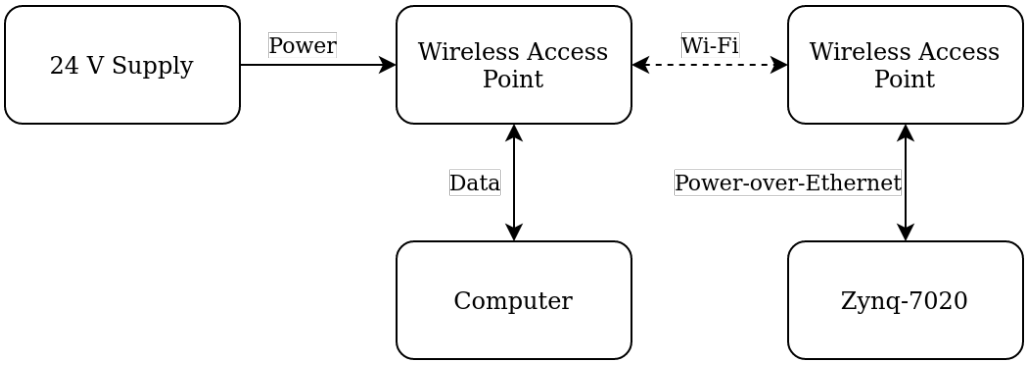


Figure 5: Setup used for testing of Wi-Fi

[2] https://apps.apple.com/us/app/bluetooth-smart-discover/id1004015467
[3] https://www.microchip.com/forums/m1116057.aspx
[4] https://www.microchip.com/forums/m987220.aspx
[5] https://www.microchip.com/forums/m987036.aspx

Using the lwip echo server, the Zynq can be pinged over Wi-Fi as shown in Figure 6, as well as complete the echo test in Figure 7. Evidently, the hardware is functional and performs as expected.

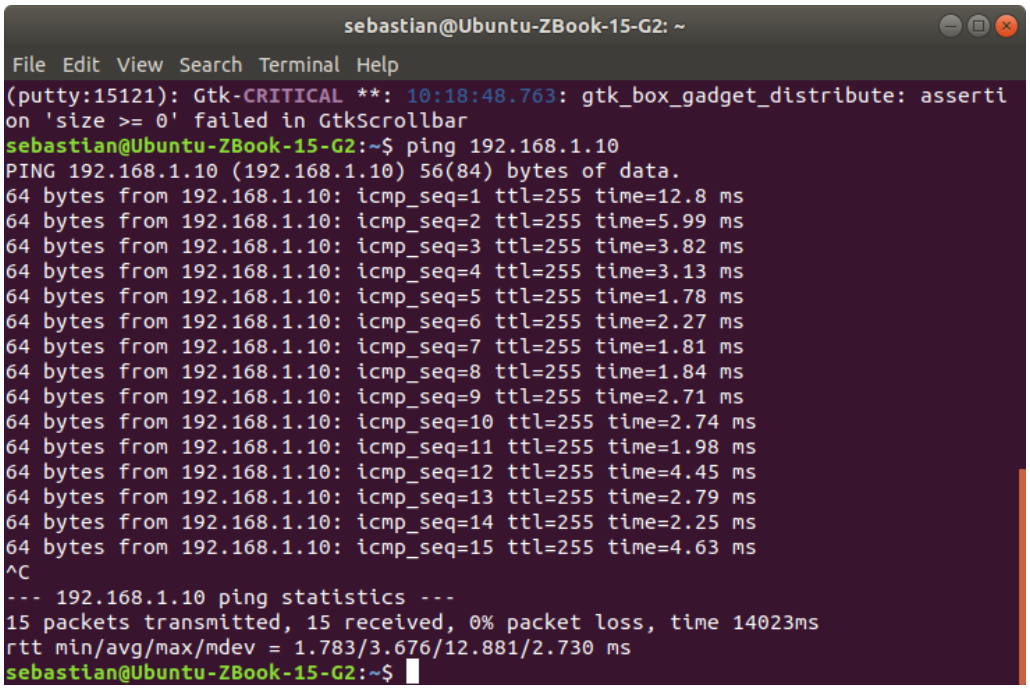


Figure 6: Screenshot of terminal showing pinging of Zynq over Wi-Fi

Figure 7: Screenshot of terminal showing echo test completed over Wi-Fi

5 Embedded Linux

This section will describe the process of compiling and booting Embedded Linux with the required components for the Master Controller. Additionally, establishing of the liveview demo page will be documented.

5.1 PetaLinux

For the Xilinx SoCs, a set of tools called PetaLinux is available, which simplifies the process of getting Embedded Linux on the Zynq. As Vivado 2020.1 has been used for designing the PL hardware, PetaLinux 2020.1 will be used, which requires Ubuntu 18.04.4 but functions for 18.04.6. PetaLinux is documented in the Xilinx guide UG1144 [4].

The PetaLinux tools are installed and setup as described in chapter 2 of the guide. In order to create a project for the liveview demo, a Vivado hardware file must be created. The Zynq-7000 requires the following hardware to be enabled [4, pp. 19-20]:

- A Triple Timer Counter (TTC)
- External memory controller with at least 32 MB memory
- UART for serial console
- Flash memory (SD, QSPI)
- Ethernet (optional)

The used interfaces are UART0, SDIO1, and ETH0. The pins used for Ethernet must be set to the "fast" speed setting to function correctly. Additionally, the MIO GPIO is activated in order to use the PS_ALIVE LED, and an AXI GPIO is placed for activating the Power-over-Ethernet relay. This is shown in Figure 8.

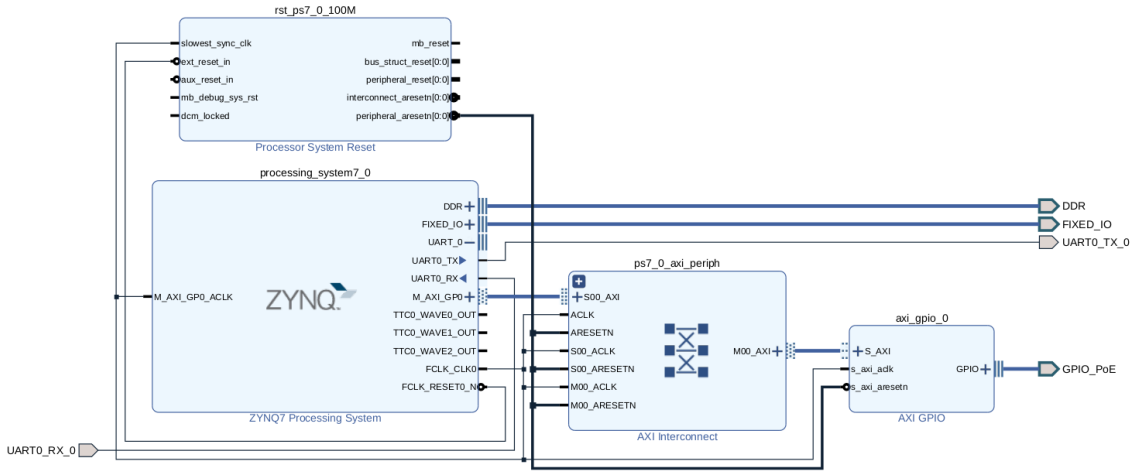


Figure 8: Vivado block diagram used for liveview test setup

The Vivado bitstream is generated, and the hardware platform exported as an XSA-file with bitstream included.

To use PetaLinux, the settings scripts must be sourced with the command [4, p. 15]

```
$ source <PATH TO PETALINUX INSTALLATION>/settings.sh
```

An empty project based on the Zynq template is then generated by going to the desired folder and writing in the terminal, [4, p. 21]

```
$ petalinux-create --type project --template zynq --name <NAME>
```

The hardware is then imported with [4, p. 24]

```
$ petalinux-config --get-hw-description <PATH TO XSA FOLDER>
```

In order to make necessary changes for booting from an SD card, the Root File System EXT4 is to be selected in the configuration menu. Additionally, ensure that the SD card is used as primary boot device in the "Advanced Bootable Images Storage Settings" menu [4, p. 78]. To reduce potential headaches later, the ETH0 IP address is chosen to 192.168.1.10 in the same menu.

For hosting the webpage, the virtual server Apache2 will be used. Thus, the package must be included by customizing the file system. Additionally, for ease-of-use over the default vi text editor, nano will also be included. This is accomplished by adding the line CONFIG_nano to the file "user-rootfsconfig" in the folder <PROJECT>/project-spec/meta-user/conf. Both Apache2 and nano are then enabled with [4, p. 133]

```
petalinux-config -c rootfs
```

nano is included under "user packages". Apache2 is included under "Filesystem Packages" > "Misc". The system image can then be built with the command [4, p. 26]

```
$ petalinux-build
```

This takes a while as the Linux kernel must be compiled.
Finally, the boot image can be generated for the Zynq with [4, p. 30]

```
$ petalinux-package --boot --fsbl <FSBL IMAGE>
                    --fpga <BITSTREAM> --u-boot
```

In case of errors, and before pushing to Git, the workspace can be cleaned to remove the built kernel, images, and temporary files with

```
$ petalinux-build -x mrproper -f
```

5.1.1 JTAG Boot

In order to ease debugging during initial booting, the JTAG boot may be used. This will transfer the images to memory over JTAG.

To ensure that the Zynq boots off the JTAG, the correct boot configuration must be selected. This is simply done on the PicoZed board with the switch SW1 by setting both taps to low, as specified in the PicoZed user guide [5, p. 27].
The files may then be transferred to the PicoZed and allow booting it.
Both the bitstream and kernel are transferred with [4, p. 55]

```
$ petalinux-boot --jtag --fpga --bitstream <BITSTREAM>
                 --kernel
                 --hw_server-url <TCP:localhost:3121>
```

The command may fail due to no connection. The connection must first be made in Vivado by auto-connecting in the hardware manager, after which the command will work, assuming Xilinx USB drivers are installed.

The root filesystem may be excluded from transfer, causing the boot to fail. This is a bug in PetaLinux occurring for unknown reasons. The approach to fixing this requires first outputting the commands to a tcl file by appending "--tcl output.tcl" to the boot command.
Afterwards, the commands below are added to the output.tcl file, before the final three lines "con", "exit", and "puts stderr [...]".

```
targets -set -nocase -filter {name =~ "arm*#0"}
puts stderr "INFO: Loading filesystem"
dow -data "<PROJECT>/images/linux/rootfs.cpio.gz.u-boot" 0x04000000
after 2000
```

The tcl script must then be run through the Xilinx System Debugger, which is opened with the command

```
$ <VIVADO PATH>/2020.1/bin/xsdb
```

Finally, in the debugger, the script may then be run with

```
xsdb% source <PROJECT>/output.tcl
```

Which will correctly transfer and boot the bitstream and Linux kernel, if the terminal is in the project folder. If an MMU error is given due to RAM lock, the processor must be reset by pressing the on-board JTAG_SRST button, after which the script is re-run.

5.1.2 SD Boot

The SD card is formatted as described in [4, p. 77], with a BOOT and a ROOTFS partition. The images and bootloaders (BOOT.bin, image.ub, boot.scr) are moved to the BOOT partition, while the file system (rootfs.tar.gz) is extracted to ROOTFS.

To ensure that the Zynq boots off the card, SD card boot configuration must be selected by setting both taps of SW1 on the PicoZed to high. The PicoZed will then boot off SDIO.

It is, however, found, that this approach will not work. In the Zynq reference manual's documentation of the BootROM [6, p. 189], it is described that fixed pins are used to check for an SD card. These are pins MIO40:45, which is SDIO0, and not SDIO1 as used on the Master Controller. For this reason, it is not possible to boot directly off the SD card. This must be fixed for PCB V2 by using the specified pins.

5.2 Linux Bring-Up

For initial bring-up and testing of the Linux application, it will be booted through JTAG. For serial communication, a UART connection with 115200 baudrate is used. The default username and password is 'root'.

To indicate that the processor has not locked up, the PS_ALIVE LED will be blinked with a shell script. The used pin is C17, internally called MIO41. First, the GPIO chip root value must be found. This is stored in /sys/class/gpio, and can be deduced by using

```
$ cat gpiochipXXX/label
```

where XXX is a GPIO chip's value. The PS GPIO will return zynq_gpio, while PL GPIOs will return other values. In this case, the PS GPIO has a value of gpiochip905. Thus, the PS_ALIVE LED is connected to GPIO $905 + 41 = 946$. The script is then written as

```
while :
do
echo 946 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio946/direction
echo 1 > /sys/class/gpio/gpio946/value

sleep 1

echo 0 > /sys/class/gpio/gpio946/value
echo 946 > /sys/class/gpio/unexport

sleep 1
done
```

The script is made executable with

```
$ chmod +x led_script.sh
```

After which it can be run in the background with

```
./led_script.sh &
```

The Power-over-Ethernet relay must be activated for the Wireless Access Point to be supplied. The GPIO chip in /sys/class/gpio for the PL GPIO is gpiochip1023. With the same approach as for the PS_ALIVE LED, it is activated. It should be noted that the chip values may change depending on hardware setup from Vivado. As experienced in the Bluetooth testing, the JX header may have a loose connection, and should be pushed down firmly.

Linux automatically configures the Ethernet PHY used for Wi-Fi, and thus no setup is required for it. The IP address is found with the terminal command

```
$ ifconfig
```

which for eth0 is 192.168.1.10 as set earlier. To communicate correctly, it may be required to set a fixed IP address and mask for the PC, as well as disable IPv6.
Apache2 sets up a folder containing the index.html file hosted on the webpage. This folder is /usr/share/apache2/default-site/htdocs. By adjusting the html file, the webpage is changed to show a descriptive header, shown in Figure 9.

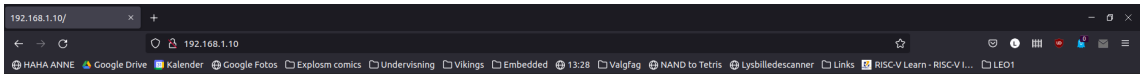


Figure 9: Screenshot showing the default Apache2 webpage hosted by the Zynq over Embedded Linux

Another way of testing the Ethernet is using the included Dropbear SSH. This can be accessed from the PC with the command

```
$ ssh root@192.168.1.10
```

This also allows transfer of files from the PC over SSH with SCP using the command

```
$ scp <PATH TO SOURCE> root@192.168.1.10:<PATH TO DESTINATION>
```

Folders can be transferred by adding '-r' after 'scp' and setting a folder path instead.

5.3 Liveview Page

The liveview pages' looks are inspired by the X90-based Master Controller's pages, which are documented in [7, pp. 89-92]. The pages are the main overview page, shutdown circuit page, cooling system page, inverter overview and control pages, and AMS overview and details pages. A mockup of the main overview is shown in Figure 10. This is used as concept for the general page setup and design of all the pages.

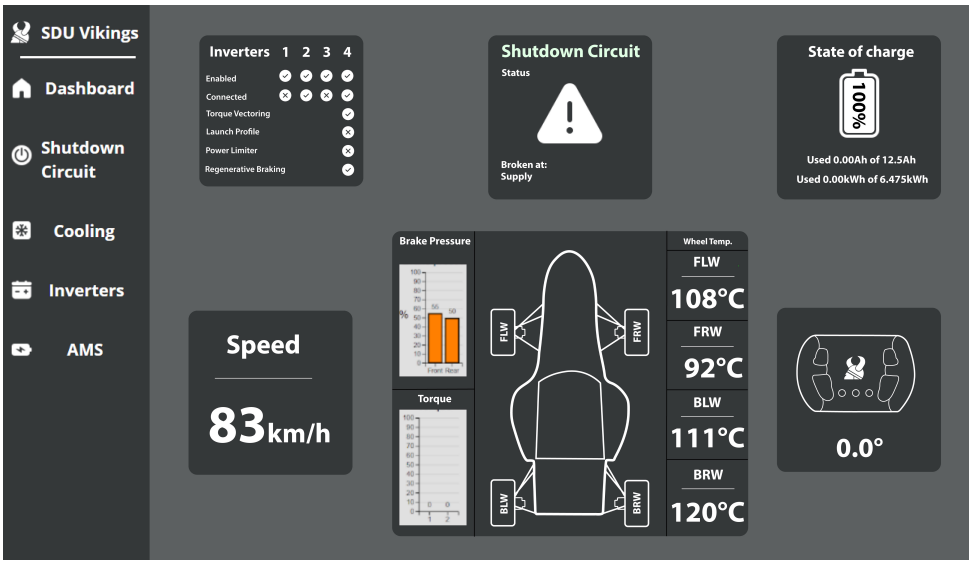


Figure 10: Mockup of the liveview's main overview page

The pages are created in HTML with CSS. In order to update values at certain intervals, JavaScript is implemented, which automatically calls a CGI script on the Zynq at specified intervals. Due to limited time, the webpage has not been finished. The main overview at time of writing is shown in Figure 11, as it looks when accessed over Wi-Fi from the PC.

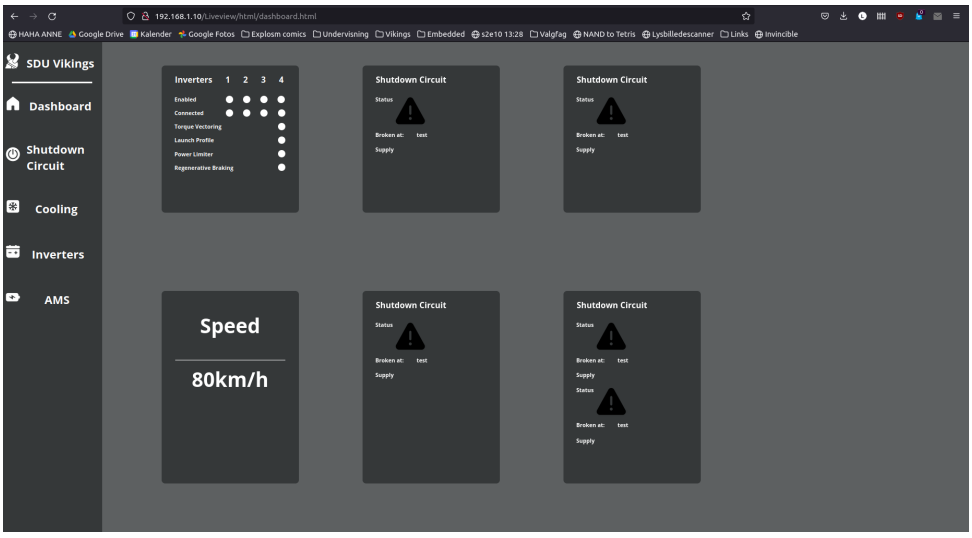


Figure 11: Screenshot showing the liveview webpage hosted using Apache2

6 Hardware

Design changes will be made to the Master Controller based on test results from the bachelor thesis, Section 4, and Section 5.

6.1 Design Errors

Referring to the test summary section of [1, p. 90], the design failures are listed in Table 1. The 12 V converter does not activate, as the enable signal must be 5 V, while it is currently 3.3 V. The torque pedal measurement circuit saturates, as it is designed for 12 V, but the op amp is only supplied with 3.3 V. The chosen op amp cannot be supplied with 12 V, meaning the design must be revised. SD card detection fails due to using a common DATA3/CD signal, routed to both the DATA3 and CD pins. A connector with dedicated CD signal should be used.

Test Specification	**Result**
12 V converter output	Failed
Sensor signal voltages	Failed
SD card detection	Failed

Table 1: Summary of failed and unfinished tests from the original report

Additionally, it was found that some shutdown circuit signals were missing. These connections must be added to the CMC connector, as well as an optocoupler and Zynq connections. Finally, the rechargeable battery does not fit in the PCB footprint, and the charging circuit hence has not been tested.

A summation of the tasks to be completed is shown in Table 2

Circuit	**Task**
12 V converter	Change enable voltage to 5 V
Sensor circuits	Change torque sensor measurement circuit
Shutdown circuit	Add missing connections
Battery charging circuit	Fix battery footprint
SD card detection	Find connector with split DATA3/CD

Table 2: Hardware tasks to be completed for this project

6.2 Design Corrections

With the tests completed in Section 4, as well as the knowledge from Section 5, the necessary design changes can now be made to all non-functioning circuits.

6.2.1 Bluetooth

As described in Section 4.1, changes were necessary for the Bluetooth module. The UART hardware flow control signals have been connected, through 0 Ω resistors, to the Zynq. P1_6, used for enabling and disabling low-power mode, has been connected to JX2.14 for future use. A header has been added to allow for connecting to a firmware updater in future iterations.

6.2.2 12 V Converter

A level-shifter is added to change the open-drain 3.3 V enable signal to 5 V. The chosen level-shifter, SN74LV1T34 [8], is also used for some of the actuator connections and known to function well.

6.2.3 Torque Sensor Measurement Circuit

The operational amplifier MCP6H02-E/SN [9] is footprint compatible with the MCP6L02-E/SN [10], and can be supplied with up to 16 V. Additionally, the offset voltage is lower, improving conversion accuracy. It can thus easily be swapped with the current buffer, with the only drawback being increased price.

6.2.4 Shutdown Circuit

The missing signals from the accumulator container, BSPD, HVD, and motor connection interlocks are added to the brown CMC pins A1:4 and B1:4, at the cost of expansion connections. The signals are routed to an optocoupler identical to the existing ones, the LTV-847S [11], and connected to PicoZed pins JX1.87:90.

6.2.5 SDIO Interface

A connector with dedicated CD connection is found, the Molex 502570-0893 [12]. To use the bootable SDIO0 interface, the signals are moved to JX3.34, JX3.36:39, JX3.41, and JX3.43, being CMD, DATA, CD, and CLK respectively. The existing signals on the pins are swapped with SDIO1, and their dedicated level-shifters removed, as these pins are supplied by 3.3 V.
As the SDIO0 pins are 1.8 V, they must be translated to the SD card's 3.3 V. By using the PicoZed FMC Carrier Card as reference [13, p. 4], a circuit is designed using the level-translator TXS02612 from Texas Instruments [14].

6.3 Minor Improvements

Along with the major design changes listed above, many minor changes have been made to design and PCB to improve performance and stability of the Master Controller. These have been listed below.

Circuit design:

- Placed a high-frequency RC filter by the XADC for connection to expansion board
- Placed 1 µF capacitors by sensor supply outputs
- Placed 100 nF capacitors by 3.3 V and 5 V converter outputs to reduce HF EMI
- Series termination resistors for Ethernet changed to 50 Ω to match trace impedance
- Moved on-LEDs for low-current actuators to before current measurements
- Added decoupling capacitors for the MIC826 voltage monitors

PCB design:

- TX_EN shortened to match other TXD Ethernet signals
- Ethernet TX signal traces impedance-matched to 50 Ω
- SDIO signal traces impedance-matched to 50 Ω
- Current loops reduced at all converters by reducing distance to capacitors
- Copied wide layer 1 traces for Shutdown Circuit to layer 4 for reduced resistance
- Optimized copper pours to improve coverage and reduce noise
- Added exposed copper connected to ground for easy oscilloscope connection
- Added silk screen in various locations for clarification
- Changed mounting holes on JTAG connector to corrected size

A 3D render of the PCB V1 is shown in Section 11.1, while the finished PCB V2 is shown in Section 11.2.

7 Conclusion

In this project, Bluetooth and Wi-Fi on the Master Controller PCB has been tested. The Bluetooth circuit prompted hardware design changes, while Wi-Fi functioned as expected.

By using PetaLinux, an Embedded Linux distribution was built for the Zynq, which allows hosting of the liveview webpage using Apache2. Due to using the SDIO1 peripheral instead of SDIO0, it was not possible to boot off an SD card, and instead only JTAG boot through USB was used. The Linux system was tested and both GPIO, UART, and Ethernet worked. Additionally, files could easily be transferred using SCP from the PC. Mockups and vector graphics for the liveview pages were created, and work was begun on the webpages in HTML with CSS and JavaScript, but not yet finished.

Using the testing results, hardware changes were made to fix non-functional circuits. These include the 12 V converter, torque sensor measurement circuit, shutdown circuit, battery charging circuit, SD card connection, and Bluetooth circuit. Additionally, many minor improvements were made to both schematics and PCB, ultimately reducing EMI and noise while improving performance and usability.

8 Future Work

The first task to be completed is ordering and soldering the PCB V2, which also encompasses testing of all connections and circuits anew. When this has been completed, the electromagnetic emission and conduction of the PCB can be measured under load to find whether the components used for EMC are sufficient. The EMC can be further improved by placing ferrite beads of over 500 Ω at 100 MHz on all single-ended CMC pins, and by adding common-mode chokes to differential signals like Ethernet and CAN [15, p. 24].
Further testing of the filtering accuracy is also required in order to optimize the Sallen-Key filters. The performance may be improved if the filters are placed closer to the JX headers, or alternatively by placing them before the voltage dividers. Finally, the operational amplifiers' characteristics may cause too large inaccuracies, in which case the more expensive, footprint-compatible MCP607T may be used.

More work is also required on the Master Controller software. Linux is only meant to be run on core 0, while a bare-metal real-time control application must run on core 1. The bare-metal application, which has yet to be created, must be initiated by core 0 as part of the boot process. The data communication between the cores must be defined, as the liveview back-end requires easy access to car data.

9 References

[1] Jonas Fuglsang Hansen & Sebastian Rud Madsen. Zynq-based master controller for formula student racecar, 2021. Accessible on GitLab and GitHub.

[2] Microchip Technology Inc. Rn4870/71 bluetooth low energy module, 2021. https://ww1.microchip.com/downloads/en/DeviceDoc/RN4870-71-Data-Sheet-DS50002489E.pdf, last accessed 01/06-2022.

[3] Ubiquiti. Nanostation ac and ac loco datasheet, 2019. https://dl.ubnt.com/datasheets/NanoStation_AC/NanoStation_AC_DS.pdf, last accessed 01/06-2022.

[4] Xilinx. Petalinux tools documentation reference guide, 2020. https://www.xilinx.com/support/documentation/sw_manuals_j/xilinx2020_1/ug1144-petalinux-tools-reference-guide.pdf, last accessed 01/06-2022.

[5] AvNet. Picozed with zynq-7020. rev. 2.1, 2020. https://www.avnet.com/wps/wcm/connect/onesite/4b3419d3-df8f-4a7f-96d2-c987732acb4a/5282-UG-PicoZed-7010_7020-v2_1.pdf?MOD=AJPERES&CACHEID=ROOTWORKSPACE.Z18_NA5A1I41L0ICD0ABNDMDDG0000-4b3419d3-df8f-4a7f-96d2-c987732acb4a-nSCkRWT, last accessed 01/06-2022.

[6] Xilinx. Zynq-7000 SoC Technical Reference Manual, April 2021. https://docs.xilinx.com/v/u/en-US/ug585-Zynq-7000-TRM, last accessed 01/06-2022.

[7] Jacob Offersen. X90 master controller documentation, 2020. Accessible on GitLab and GitHub.

[8] Texas Instruments. SN74LV1T34 Single Power Supply Single Buffer GATE CMOS Logic Level Shifter, June 2017. https://www.ti.com/lit/ds/symlink/sn74lv1t34.pdf, last accessed 01/06-2022.

[9] Microchip. MCP6H01/2/4, December 2011. https://ww1.microchip.com/downloads/en/DeviceDoc/22243D.pdf, last accessed 01/06-2022.

[10] Microchip. MCP6L01/1R/1U/2/4, February 2009. https://www.mouser.dk/datasheet/2/268/22140a-71218.pdf, last accessed 01/06-2022.

[11] LITE-ON Optoelectronics. Photocoupler - LTV-8x7 Datasheet, September 2017. https://optoelectronics.liteon.com/upload/download/DS-70-96-0016/LTV-8X7%20series%20201610%20.pdf, last accessed 01/06-2022.

[12] Molex. 5025700893 1.10mm Pitch microSD Memory Card Connector, April 2022. https://www.molex.com/webdocs/datasheets/pdf/en-us/5025700893_MEMORY_CARD_SOCKET.pdf, last accessed 01/06-2022.

[13] AvNet. PicoZed FMC Carrier Card V2 Schematics, March 2016. https://www.avnet.com/opasdata/d120001/medias/docus/178/SCH-AES-PZCC-FMC-V2-G-v1.pdf, last accessed 01/06-2022.

[14] Texas Instruments. TXS02612 SDIO PORT EXPANDER WITH VOLTAGE-LEVEL TRANSLATION, February 2009. https://www.ti.com/lit/ds/symlink/txs02612.pdf?ts=1650794800320, last accessed 01/06-2022.

[15] NXP. i.MX 6ULL Layout Recommendations Hardware Development Guide for the i.MX 6ULL Applications Processor, User's Guide, November 2016. https://www.nxp.com/files-static/soft_dev_tools/doc/user_guide/IMX6ULLHDG.pdf?&fasp=1&WT_TYPE=Users%20Guides&WT_VENDOR=FREESCALE&WT_FILE_FORMAT=pdf&WT_ASSET=Documentation&fileExt=.pdf, last accessed 01/06-2022.

10 Terminology

PicoZed: The primary computing board for the Master Controller, containing a Zynq-7020 with relevant power supplies, volatile and non-volatile storage, and peripheral circuits.

JX: The three large 100-pin receptacles used for mounting the PicoZed board to the Master Controller motherboard. Designated JX1, JX2, and JX3. A full overview of their connections is shown in Appendix L of [1, pp. 142-144].

CMC: The large 112-pin Molex connector used for almost all connections to other modules in the car. A full overview of its connections is shown in Appendix M of [1, p. 145].

WAP: The Ubiquiti Nanostation AC Loco Wireless Access Point is used for Wi-Fi communication between the Master Controller and a PC. It is powered by 24 V Power-over-Ethernet from the Master Controller, through an M12 connector.

AUX: Auxiliary XADC channel, used for 12-bit conversion of 0-1 V signals.

INS: Inertial Navigation System, an external module containing gyros, GPS, and many sensors used for estimating car movements.

PL: Programmable Logic. The FPGA part of the Zynq, which can be programmed with Vivado, as well as connect to specific I/O banks.

X90: The currently used Master Controller, a heavy-duty X90 from B&R Automation.

11 Appendix

11.1 Master Controller V1 PCB

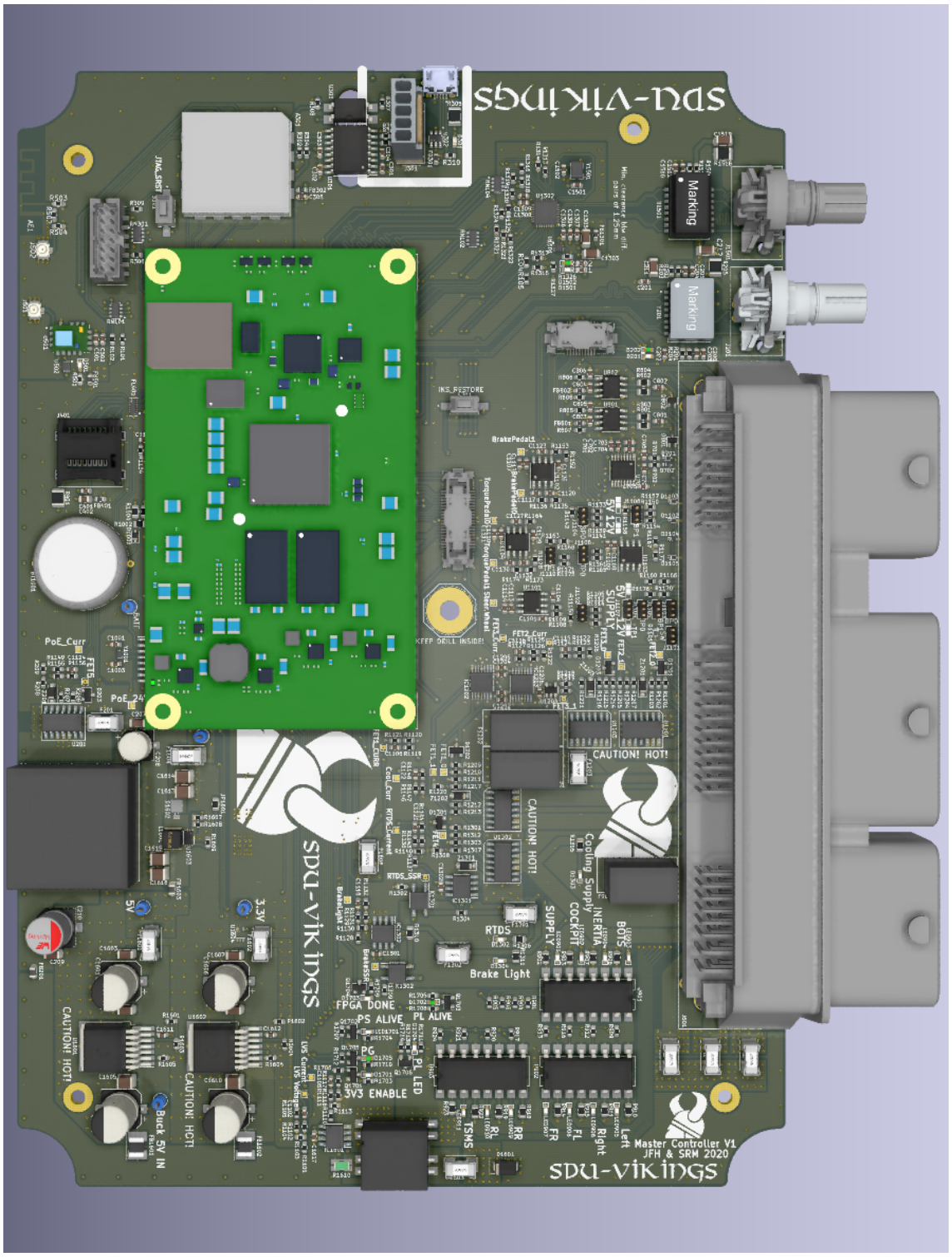


Figure 12: KiCad Render of the Master Controller V1 PCB

11.2 Master Controller V2 PCB

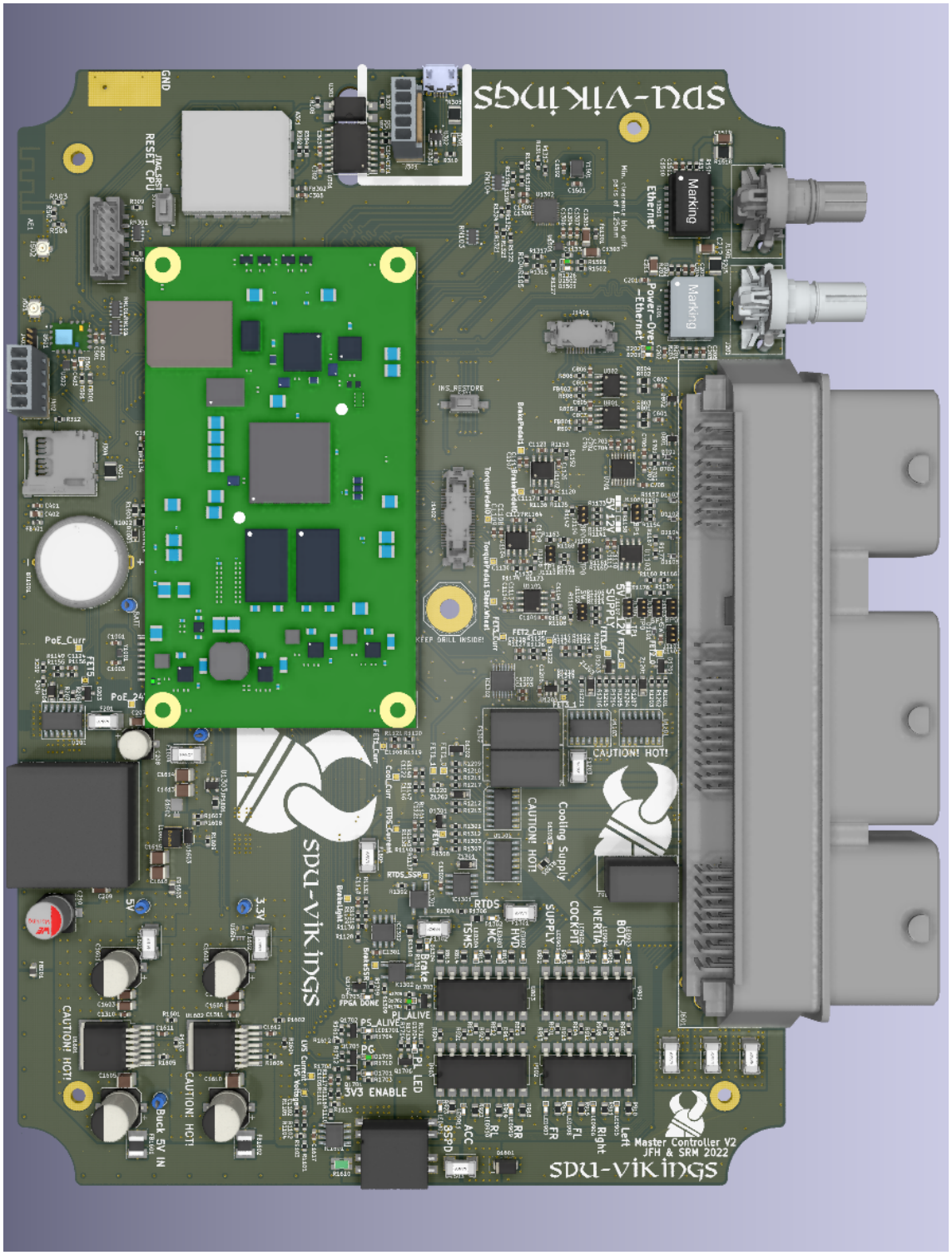


Figure 13: KiCad Render of the Master Controller V2 PCB