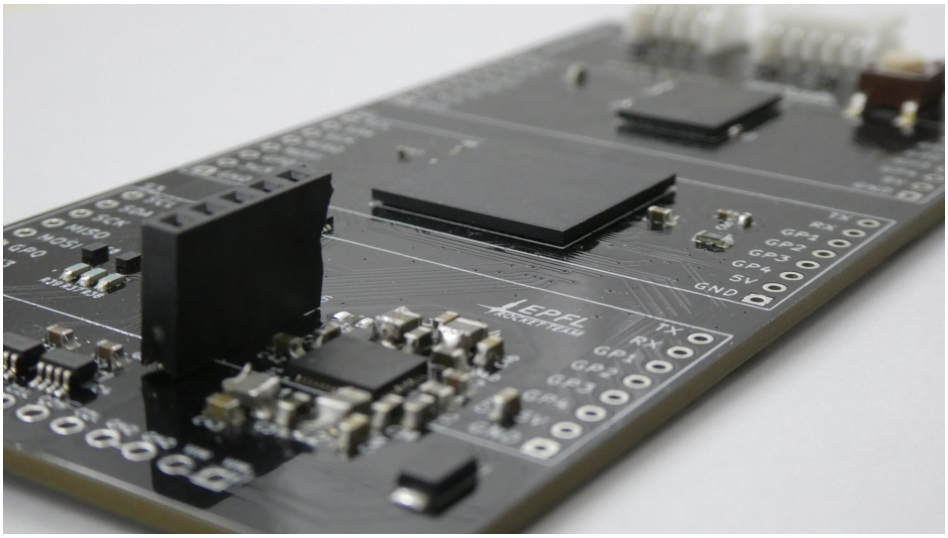
Semester Project

Design of a high speed microprocessor hostboard

Project acomplished at EPFL Rocket Team



The microprocessor hostboard assembled and ready to be used

Iacopo Sprenger

Under the supervision of Alexandre Schmid.

Fall 2021

Contents

Symbols and abbreviations

ACI	Atelier des Circuits Imprimmés
AV	Avionics
BOM	Bill of Materials
CA7	Cortex-A7
CAN	Controller Area Network
CANFD	Controller Area Network Flexible Data-Rate
CM4	Cortex-M4
DDR3	Dual Data Rate 3
EMI	Electromagnetic Interference
eMMC	Embedded MultiMediaCard
EPFL	Ecole Polytechnique Fédérale de Lausanne
ERT	EPFL Rocket Team
ESD	Electrostatic Discharge
GUI	Graphical User Interface
HB	Hostboard
I^2C	Inter-Integrated Circuit
LDO	Low Dropout Regulator
MPU	Microprocessor Unit
QFN	Quad-Flat no-Leads
RFI	Radiofrequency Interference
SDMMC	Secure Digital and MultiMediaCard
SDRAM	Synchronous Dynamic Random Access Memory
SPI	Serial Peripheral Interface
USB	Universal Serial Bus
UART	Universal Asynchronous Receiver Transmitter

1 Introduction

1.1 Purpose and motivation

For the past three years, the EPFL Rocket Team has been using the same STM32F446RE microcontrollers for it's avionics. Because the product has reached it's end of life and because of the increasing requirements for improved computing power. The choice has been made to develop a new hostboard with an improved microprocessor from the stm32mp1 serie.

The purpose of this report is to explain the choices, design and manufacture processes we used to develop the hostboard for such a high speed processor. This report will take you through the steps of circuit design followed by the board design and finally board manufacture and assembly.

1.2 Requirements

We start by defining some requirements for this project. These requirements will guide the development choices of the project.

- The hostboard shall run a powerful and future proof microprocessor.
- The hostboard shall have a debug indicator.
- The hostboard shall have three extension slots for peripherals
- The hostboard shall have easy and accessible debug and programming ports.
- The hostboard shall have two CAN busses.
- The hostboard shall be able to measure the battery voltage.
- The hostboard shall be protected from ESD.

2 Method

2.1 Preliminary design

The purpose of this preliminary design is to asses and choose what is necessary for the STM32MP157 microprocessor to properly work. We will then decide how the components are laid out on the board and finally how the board is secured inside the rocket.

2.1.1 Hostboard

The STM32MP157 MPU needs memory and power. The memory will be provided in the form of a microSD card for storage and DDR3 SDRAM for the working memory. The power will be provided by a power management IC which is made for the STM32MP157 MPU.

In addition to that, we will have external interfaces for sensors and actuators as well as a dual CANFD bus for inter hostboard communication.

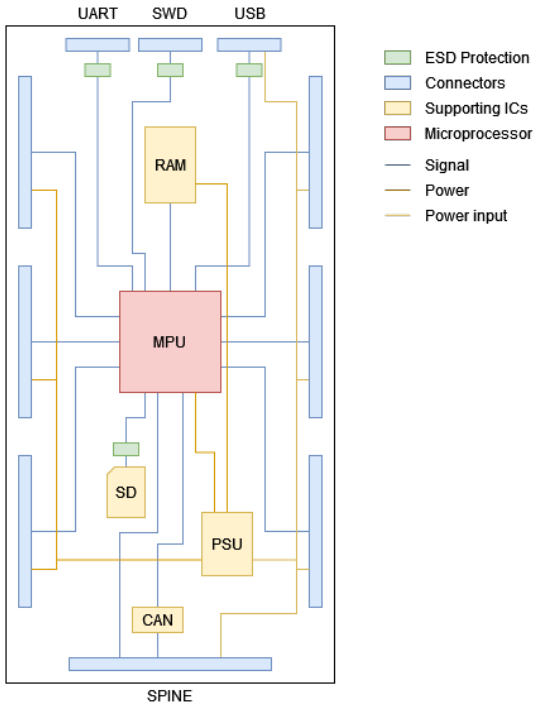


Figure 1: Block diagram of the hostboard's components and it's connections.

Figure 1 shows a schematic representation of the different components of the hostboard. This also shows how the components will be laid out on the PCB.

At the center, there is the microprocessor, which is connected to the DDR3 SDRAM memory as well as to the microSD card storage. The central location is ideal for the processor as it will be connected to nearly all of the components and interfaces. The SDRAM is right next to the processor but far enough such that the impedance controlled address and data buses can be properly routed.

The hostboard will also provide external peripheral connection points, which we will call sockets. There are three sockets on the front side of the board. They allow the attachment of three peripheral extensions per hostboard. The other connectors are for debugging on the top of the figure and for inter hostboard communication on the bottom in what we call the spine connector. The power input

is also provided through the spine interface from the batteries.

Finally, we have the 5V input which goes to the STPMIC1 and to the right hand side socket connectors. The STPMIC1 creates the necessary power supply rails for the MPU and it's peripherals. On the left hand side sockets, the power comes from the 3V3 output of the STPMIC1.

2.1.2 Structure

The structure serves to maintain the hostboards into place. The design of this structure is outside the scope of the project, however it is still relevant as the outline of the PCB must match this structure.

Figure 2 shows a shematic representation of how the hostboards are attached as seen from the side. The red and blue component is the hostboard with it's three extensions. It will be secured by two 3D printed rails on either of it's sides and by mechanical stops on the top and bottom. This relieves the connectors from any stress due to the rocket's accelerations.

Figure 2: The hostboard's structural attachment, seen from the side and in 3D.

2.2 Choice of components

The next step in our design process is to select the required components to support the microprocessor as well as our application. The choice of most components was motivated by their compatibility with our MPU, their availability and finally their price.

2.2.1 Microprocessor

The choice of microprocessor was motivated by a desire to stay in the STMicroelectronics ecosystem as we are already used to it. In the STM32 microcontrollers familly, there is only one series which combines the processing power of a dual core cortex-A7 processor with the real-time advantages of a cortex-M4 Microcontroller. It is the stm32mp157 serie. We specifically chose the stm32mp157dab1 due to its similarity with a relatively cheap development kit as well as the size and format of the BGA354 package which has the widest pitch. One big advantage of this processor is also its availability regardless of the current semiconductor shortage.

2.2.2 Power supply

As power supply we choose the recomended IC for this processor which is the STPMIC1APQR from stmicorelectronics. Sadly this exact model is not available, thus we buy the STPMIC1EPQR and

reprogram it to meet our needs. This will require an additionnal I^2C port to the hostboards, which is connected to the power supply.

2.2.3 Memory

The STM32MP157DAB1 microprocessor requires an external memory for it's cortex-A7 cores. This memory will be a DDR3 SDRAM located next to the MPU. The choice of this memory was motivated by it's similarity to the one used in the development kit as well as it's backward compatibility to the slower speed of the MPU DDR3 controller. The exact model we use, is the IS43TR16256B-125KBLI from ISSI.

2.2.4 Storage

The storage, also located outside the MCU will be handled by a microSD card. This is a temporary solution as unsoldered memory is not reliable enough in the high vibrations environment of a rocket flight. However it is easier to program than soldered memory as it can be plugged into a computer. The plan for a later version is to use a soldered eMMC memory.

2.2.5 Can transceivers

The CAN transceivers will transform the RX and TX signals from the MPU to the half duplex differential signal for the can bus.
The MAX3304 where chosen as they are some of the only which support 3V3 logic and are available.

2.2.6 Debug LED

To simplify the development of the avionics embedded software, the hostboards implement an RGB debug led. The LED we choose, is actually made of three leds covered by a opaque diffusing screen to mix the colors. We control these leds using PWM signals to mix the colors. To avoid using too much of the MPU's total output current, we drive the LEDs using a non inverting line buffer.

2.2.7 ESD Protection

In the past, we had some issues with the hostboards as they would often get broken by electrostatic discharge. This leads us to place ESD protections on the external connections of the new hostboards. We cannot place ESD protections everywhere as this takes to much space. However, we can place ESD protection where the user is most likely to touch exposed pads and connectors.

We use the RClamp0502A from Semtech which are vesratile ESD protection circuits which also work on differential pairs.

On the microSD card connector, we use the EClamp2357NQ as we want to reduce ESD as well as EMI and RFI. The protection has an internal low-pass filter which removes EMI and RFI.

2.2.8 Connectors

The connectors we use for debug are two JST connectors and one micro usb. The JST and usb connector have a polarity, which prevents the user to plug them backwards.

For the socket and spine, we use simple pin strips as they are cheap and easily available. These connectors do not need to be structurally solid as the forces on them are handled by the structure.

2.3 Choice of interfaces

To interact with the external world, the hostboards have three main interfaces. The debug interface is designed to help during the development of the software. The socket interface allows the attachment of peripheral extensions. The spine interface allows for inter board communication and power delivery.

2.3.1 Debug

The debug interfaces will be used to diagnose problems during the software development as well as to download programs in the MPU's memory. For software diagnostics, we have the UART8 port. To download programs to the memory as well as to run the debugger, we have the SWD port.

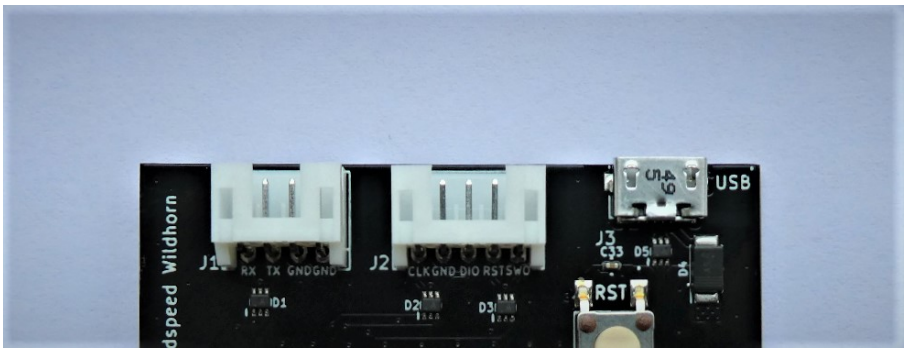


Figure 3: Picture of the hostboard's debug ports.

2.3.2 Socket

The Socket interfaces allow the connection to external peripherals such as sensors, telemetry or actuators. The hostboard has three socket interfaces and each interface is equipped with one USART, one I2C and one SPI port. Additionally, each interface has slightly different characteristics such as PWM channels, ADC inputs, DAC output or USB port.

The Figure 4 shows the precise pinout of the three sockets. There, we can see what each socket has as additional interfaces.
The first socket has four timer channels, which are usually used as PWM outputs. It also has a DAC output, which can be used in combination with a buzzer to play sounds. All of these general purpose pins can also be used as simple GPIOs.

The second socket has an USB port, this might be useful if we want to connect an USB device such as a camera to the hostboards. The other general purpose pins are analog input, which can also be configured to be GPIOs. The particularity for the second socket is that the USB pins cannot be used for something else.

The third socket is very similar to the first as it also has four timer channels. In place of the DAC, it has an analog input which can also be used as GPIO.

2.3.3 Spine

The spine connector is used to connect multiple hostboards together as well as to connect the hostboards to the batteries.

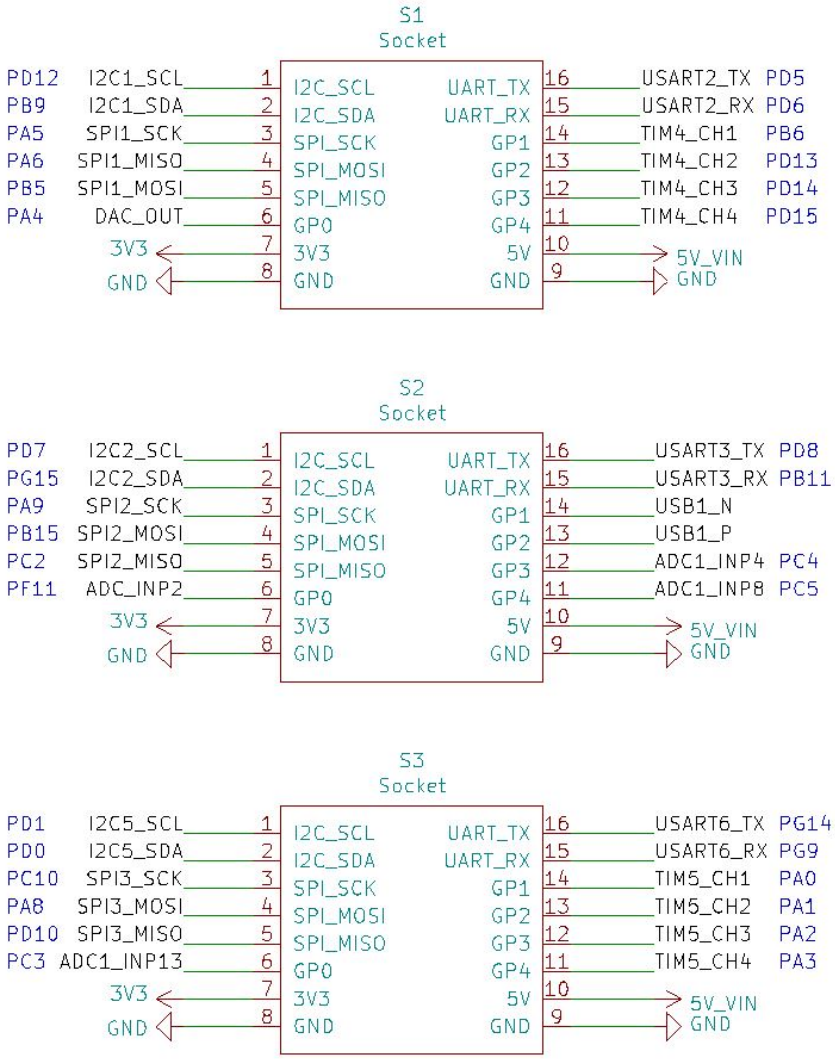


Figure 4: Schematics of the three sockets and their connection to the microprocessor.

The connection between the hostboards is done through two redundant FDCAN buses. This will allow multiple hostboards to easily share and check data that they are processing. The redundant bus architecture allows us to implement and experiment on some transmission redundancy. This also allows us a larger bandwidth at the expense of redundancy.

The spine has two power and two ground pins for power delivery. It also has two connections to analog inputs. These will be connected to the batteries directly through voltage dividers in order to be able to sense the battery level.

The Figure 5 shows how the spine is connected internally. Note that the battery voltage measure pins are clamped in the 0 - 3.3V range by the ESD protection diodes.

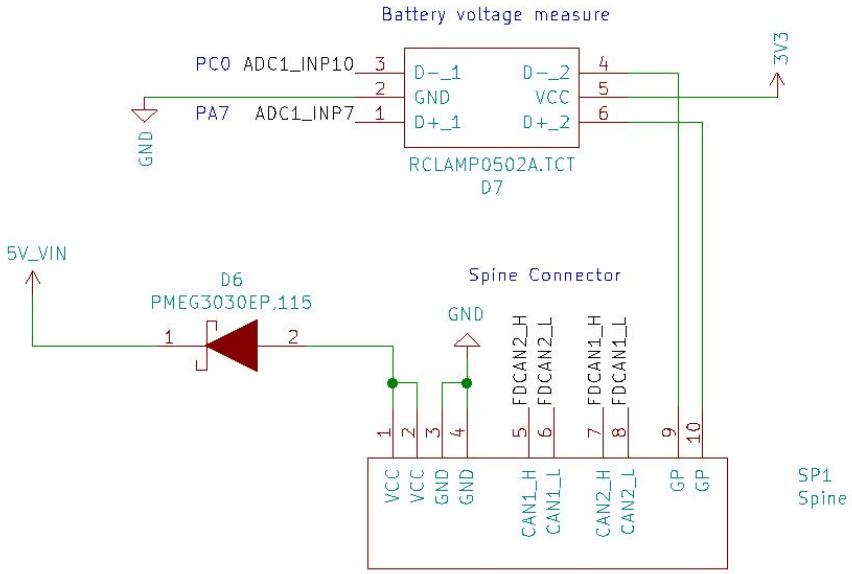


Figure 5: Schematics of the spine connector and its connection to the microprocessor.

2.4 Circuit design

To simplify the circuit design and make the result easier to read, we separated it into five sheets. This section will go through these sheets and explain the choices that where made for the design of this circuit.

2.4.1 MPU

The MPU sheet holds all the signal connections to the MPU as well as the peripherals and interfaces.

In order to select the proper boot mode for the MPU, we have hardwired two possible boot modes. These can be controller by a single dip-switch which shorts the BOOT 0 and 2 pins. In the shorted position, the MPU will boot from the USB or UART ports. In the open position, the MPU will boot from the microSD card.

Booting from USB or UART allows the user to load data into the memory from a computer or other host device.

To provide the user with a convenient feedback on the program's operation and state, we implemented an RGB debug led. The three channels of the led are connected to three PWM outputs through non inverting buffers. These allow the led to shine bright without using the limited current output capabilities of the MPU.

On this sheet are also the two FDCAN transceivers. They are connected to the MPU on one side and to the spine on the other side. Each one of them requires a decoupling capacitor. Their special functions shutdown and standby, which allow a lower power consumption, are not used in our application.

The debug ports are all connected through ESD protection circuits as these are the most likely to be touched or hot plugged. These are the UART8 serial port, the SWD debugger port and the USB2 port. The hostboard can be powered through the USB connector, but not the other debug interfaces.

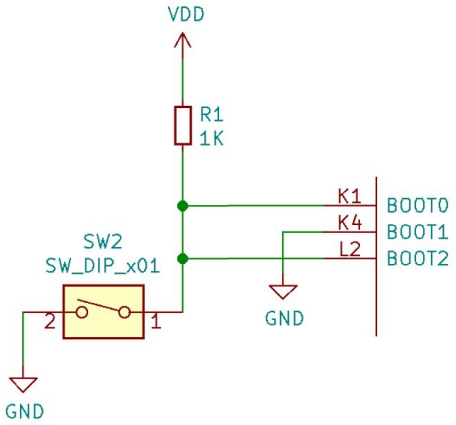


Figure 6: Schematics of the boot mode selection.

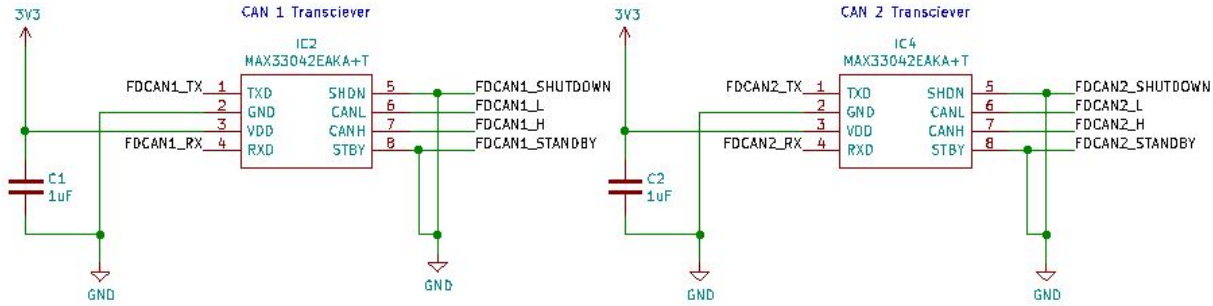


Figure 7: Schematics of the FDCAN transceiver and their connections to the microprocessor.

2.4.2 MPU Power

The MPU POWER also holds the MPU connections, but this time only the ones related to power and the main decoupling for the MPU. This is done on a separate sheet as the MPU has a lot of power pins.

The MPU is powered by three main supplies VDD CORE, VDD DDR and VDD. These must be properly decoupled by capacitors. There are also some auxiliary supplies like VDD USB which must be connected to the VDD3V3 USBHS and VDD3V3 USBFS pins.

In addition to those main supplies, some power ports must be connected in the right way to configure the internal regulators. BYPASS REG1V8 must be grounded as we do not have an external 1.8V regulator. VDD1V2 DSI PHY must be connected to VDD 1V2 DSI REG which in turn must be decoupled by a capacitor. All the regulator outputs must be decoupled by capacitors.

2.4.3 Memory

The DDR3 Sheet holds the connection between the MPU and the DDR3 SDRAM as well as the termination resistors network and it's decoupling. The Memory has three main connections to the processor, one address and two data buses.

These main connections are shown on the Figure 8. In green, there is the address bus. This bus

will be located on the bottom layer of the PCB and travel behind the memory chip. Additionally, it needs to have 56Ω termination resistors to the VTT DDR power supply. This power supply must be very stable and thus it has a huge amount of high frequency decoupling capacitors very close to the resistors.
In red, there are the two data buses byte 0 and 1. These will be located on the top layer of the PCB and travel on the left and right side of the memory chip. These data buses have the particularity of being swappable. The data bits inside of a byte are also swappable with the exception of the first bit which is used for calibration. In our connection, the two bytes are swapped for convenience as well as a number of bits inside of the two bytes. This allows us to route every signal on one layer without a single intersection.

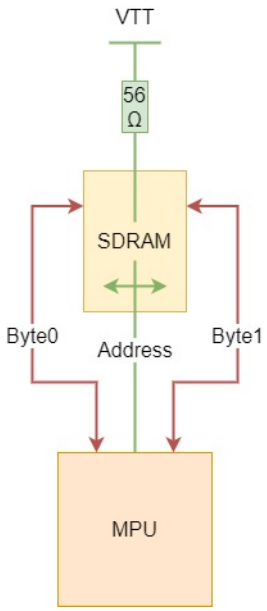


Figure 8: Block diagram of the connection between the microprocessor and the memory. The green is located on the back of the PCB, whilst the red is on the front side.

The last thing the memory needs is a power supply, which comes from VDD DDR. And we also have a reference power supply which is generated by the STPMIC1 at the half of VDD DDR.

2.4.4 Storage

The EMMC sheet holds the connection to the storage. For now, this is done using a microSD card which is simply connected through an ESD and RFI protection circuit. This is a quite straightforward directly to the SDMMC1 peripheral inside the processor.

2.4.5 Power

The POWER sheet holds the power supply subassembly. Here, everything revolves around the STPMIC1 power management IC.

The four main power rails are provided by four buck converters. These are VDD CORE at 1.2V, VDD DDR at 1.5V, VDD at 3.3V and 3V3 at 3.3V. The four synchronous buck converters need only an inductor and two capacitors as external components.
Three additional voltages are generated with LDO regulators. These are VREF DDR at 0.75V, VREF DDR at 0.75V and VDD USB at 3.3V.

All the inductors and decoupling capacitor values are recommended from STM for the use of the STPMIC1 with our MPU in its configuration.

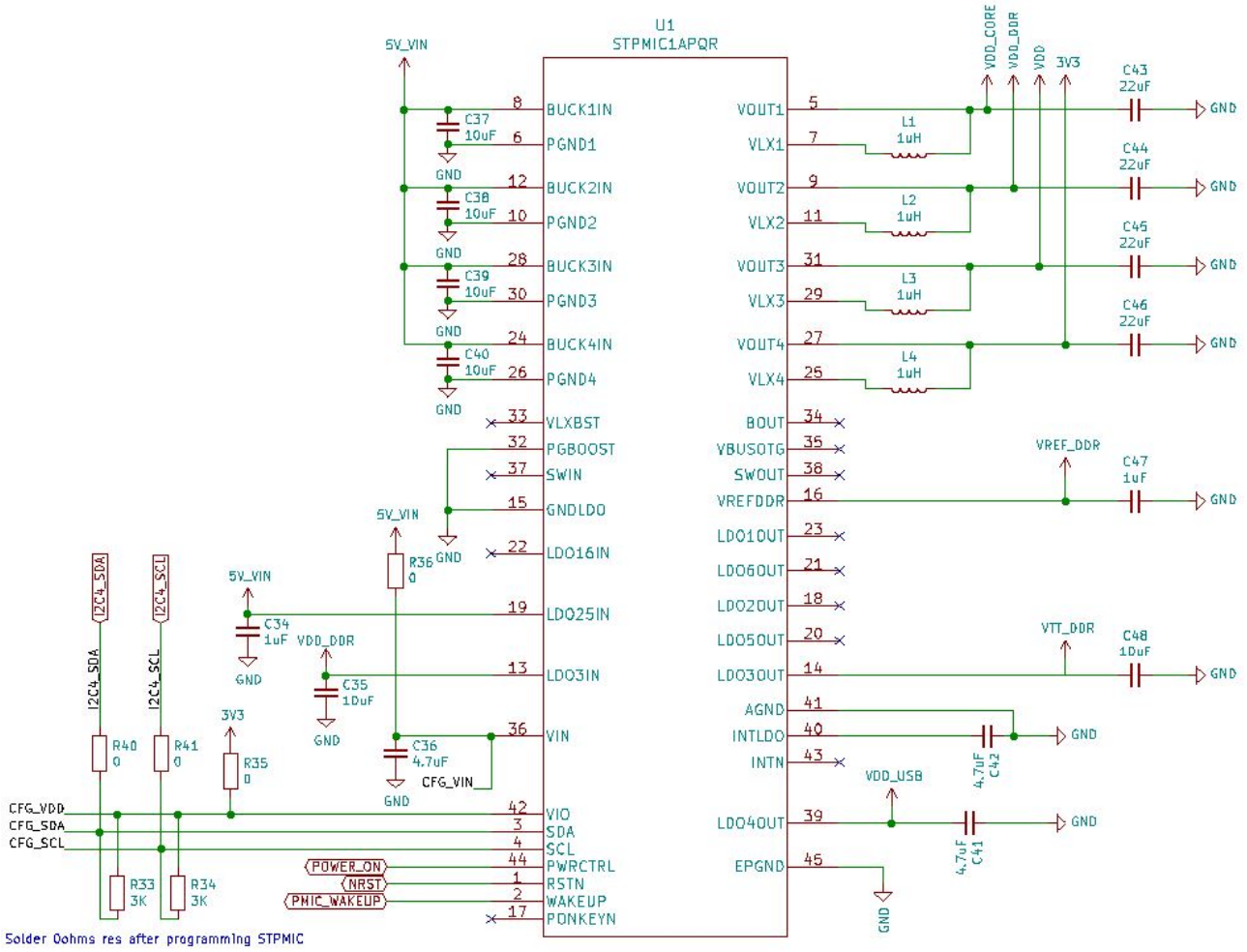


Figure 9: Complete schematics of the power supply and its passive external components.

The Figure 9 shows how the power management IC is connected to the power rails and its necessary external passive components. On the bottom left side of the figure, there is the configuration port of the power supply. When the 0Ω resistors are unsoldered, this is connected to test points which can be used to externally reprogram the power supply. For normal operation of the hostboard, the resistors must be soldered. This will allow the microprocessor to reprogram the power supply if needed.
This configuration is done through an I^2C interface. The following operations must be done in order to program the correct timings

- Write 0xEE at address 0xF8
- Write 0x92 at address 0xF9
- Write 0xC0 at address 0xFA
- Write 0x02 at address 0xFB
- Write 0xFE at address 0xFC

- Write 0xB0 at address 0xFD
- Write 0x02 at address 0xFE
- Write 0x01 at address 0xB9

It is then important to read those values in order to check that they where correctly registered as an incorrectly configured power supply might damage the microprocessor.

These operations can be made using any I^2C master. We use a simple python program, located in appendix A, which runs on a raspberry pi and uses it's GPIOs to connect to the STPMIC1. If we had a USB to I^2C dongle, we could also use STM's user friendly GUI configuration tool to do it.

2.5 Board design

The largest part of this project was the board design. Once the circuit is done, we must transfer it to a physical medium, which in our case, is a printed circuit board. This is quite challenging as all of the connections must fit on a limited number of planes without crossing each other. These signals must also be properly isolated from interfering with each other and they must sometimes follow specific guidelines in order to have the right electrical characteristics.

2.5.1 Design rules

The EPFL Rocket Team has a sponsoring agreement with PCBWay, thus they where chosen as manufacturer for the PCBs. We have to comply with their manufacturing capabilities, this will define the rules by which we design our circuit.

The first rule is the resolution of the copper etching. This gives us the minimum width and minimum space between two tracks. PCBWay can manufacture with this minimum at 4 mils for a reasonable price. This corresponds to 0.101 mm.
The second rule is the minimum size of the vias. PCBWay can manufacture vias with a diameter of 0.35mm and a bore diameter of 0.2mm.
These are the two main rules we followed during the board design.

2.5.2 PCB Stackup

The PCB stackup is the arrangement and allocation of the layers inside the circuit. For our use, we decided to go for a four layers PCB. This gives us a great compromise between ease of design, signal integrity and production cost.

Solder Mask	
1 Oz	Signal
0.11 mm	FR4
1 Oz	GND
Core	
1 Oz	Power
0.11 mm	FR4
1 Oz	Signal
Solder Mask	

Figure 10: Description of the PCB stackup for the hostboards. The orange layers are copper.

The details of our stackup can be seen on Figure 10. The top and bottom layers are used for the signals. The first inner layer is a ground plane and the second inner layer is a power plane. These allow us to easily distribute the power throughout the board. Additionally the power and ground planes provide references and shielding for the impedance controlled signals.

The stackup also defines the distances between the copper planes. In our case, we have a 0.11mm separation between the signal layers and their respective power plane. This is important as we will need it to compute the trace impedance as well as the required inter trace separation for an adequate signal integrity.

2.5.3 Controlled Impedance

Some of our high speed signals such as the DDR3 connection or the USB, need a controlled impedance in order to properly work. As we have a power and ground planes close to the signal layers, we route our impedance controlled tracks using microstrip lines and coupled microstrip lines. We use KiCAD to compute the parameters of the line in accordance with the desired impedance.

The DDR3 connection requires two types of signals with controlled impedance. The normal data and address lines, which are single ended, require a 55Ω impedance. This gives us a track with of W=0.159mm computed with KiCAD. The clock and data strobe lines, which are differential, require a 100Ω impedance. This gives us a track width of W=0.188mm and S=0.323mm of track separation.

The USB connection requires a single differential pair with an impedance of 90Ω. This connection thus requires a track width of W=0.204mm and S=0.167mm of separation.

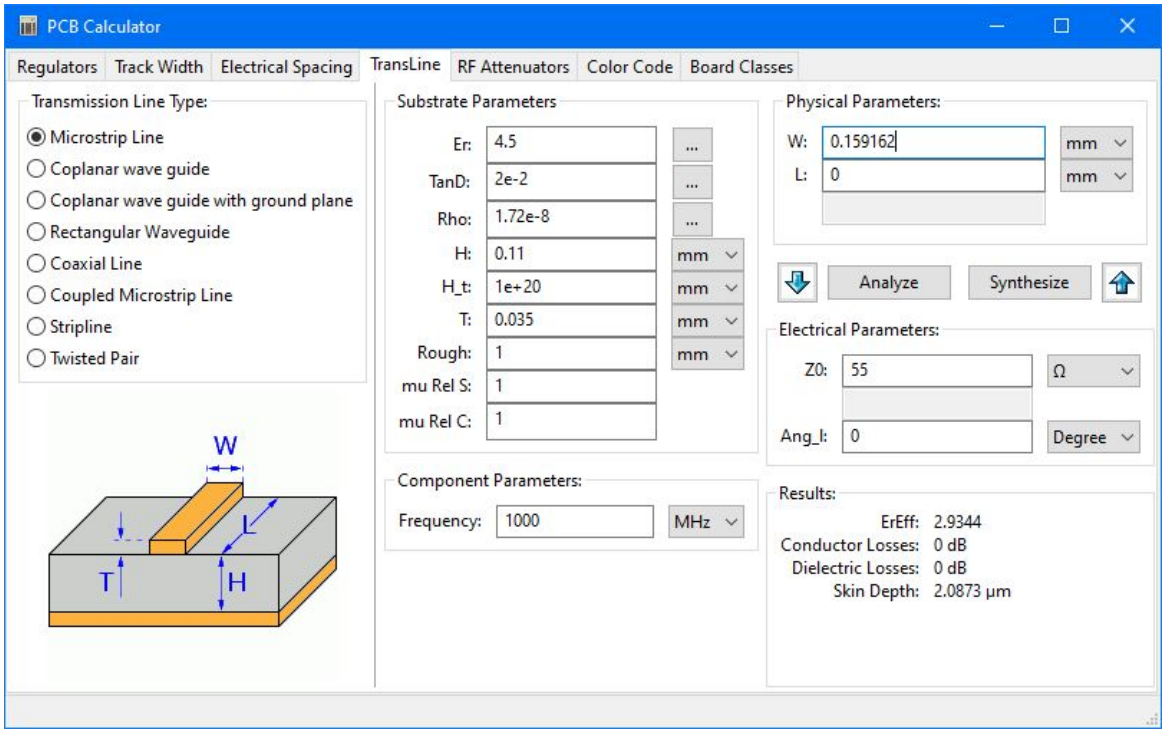


Figure 11: Screenshot of the KiCAD track impedance calculator. Currently being used in microstrip line mode.

The Figure 11 shows how the impedance calculator looks like as well as it's parameters for a microstrip line. It is very similar for a coupled microstrip line, with an additional output for the track

separation. In our case, the height to the ground plane is H=0.11mm and the thickness of the copper is T=0.035mm because we have one ounce copper layers. The dielectric material is FR4, which has a dielectric coefficient of ϵ_r=4.5.

2.5.4 Length matching

The second critical aspect for the DDR3 memory connection, is the correct propagation time for the signals. This will be handled by closely matching the lengths of the signals which connect the processor to the memory.

The manufacturer provides some guidelines for the relative lengths of the tracks. In general, the signals should be closely matched inside a signal group with respect to the signal group's clock or data strobe. The longest signal must be the differential clock. The address lines must be closely matched to the clock from -1.016mm to 0mm of length difference.
The data strobes for the data bytes must be shorter than the clock for up to -15mm. The data bytes must be closely matched to their respective data strobe for ±1.016mm.
The manufacturer also gives us guidelines concerning signal integrity. The tracks must be separated of at least three times the distances to the ground plane as much as possible in order to avoid crosstalk. This means that our tracks must be separated of at least 0.33mm. This is not possible everywhere, especially when approaching the fine pitch of the BGA. This guideline must however be respected as much as possible whenever it is possible. This guideline is called the 3S isolation rule.
To help the user respect these lengths and taking into account the internal pad to die lengths, the manufacturer provides a spreadsheet to check those lenght differences. This spreadsheet can be seen on the Figures 29 and 30 in the appendix B.

These spreadsheets only allow us to check that the track lengths are properly calibrated. However, we still need to route all those tracks between the processor and the memory. Additionally, we also need to manually tune the lengths by routing the tracks using serpentine and switchback patterns. The switchbacks are preferred over serpentine as they reduce the effect of orthogonal propagation. This effect makes the signals travel faster in a serpentine than in a straight line for the same track length, which is unwanted because we cannot control it accurately. The Figure 12 shows the difference between switchbacks and serpentines.

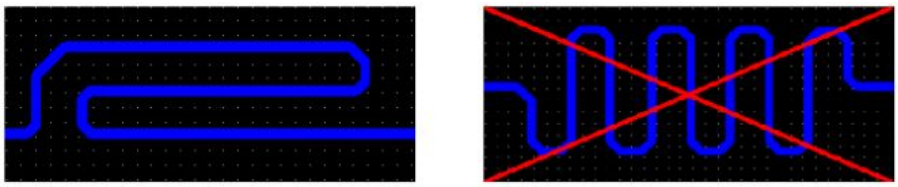


Figure 12: Comparison between the preferred switchback pattern and the serpentine pattern.

In order to route all of these tracks whilst respecting their desired lengths, the absence of crossings, their impedance as well as the 3S isolation rule. We need to come up with a methodology which allows us to tackle all of this constraints one by one to keep the routing manageable by a human.

The first step, as shown on the left of the Figure 13, is to perform an initial routing of all the tracks without any crossings. This allows us to define how we perform the BGA and what bits we swap to ease the routing. In this step, we do not care about the lengths of the traces, just their arrangement.

The second step, as shown on the right of the Figure 13, is to make each trace approximately at its final length. For now, we do not use the spreadsheet and only aim for tracks that are about 60mm long. This length was chosen because it corresponds to the longest straight trace, thus we cannot make the connection shorter. This step allows us to determine how much space is needed by all the traces. This is the most difficult step as it requires the user to estimate the required space, then route all the tracks and finally see if the space was enough or too much. If there is not enough space, this must be started over by changing the initial estimation and potentially the lengths of the longest straight trace.

This iterative process can take up to several days in order to resolve all these length and space issues on all the memory connection buses. The best way is to start by the data buses as they will define the minimal length of the address bus.

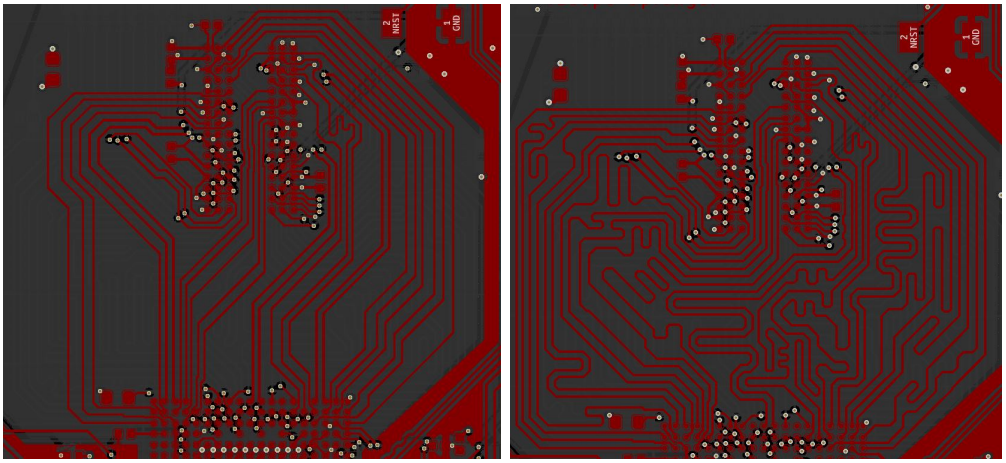


Figure 13: On the left, an initial routing without taking into account track lengths. On the right, an eye balled routing where the tracks are about 60mm long to determine the required PCB real estate.

The final step, as shown in the Figure 14, is to fine tune the track lengths using the spreadsheet. in this step, it is also good to check that the 3S isolation rule is respected as much as possible. This rule must also be respected inside the switchback and serpentine patterns to further reduce the risk of orthogonal propagation. This step is also quite time consuming as KiCAD only computes the track lengths and we also need to take into account the lengths of the vias as well as the lengths of the pad to die connections.

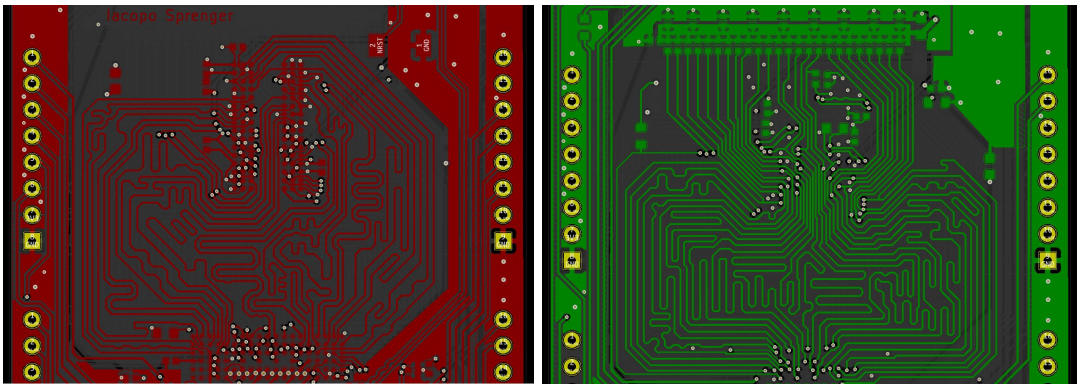


Figure 14: On the left, the finished routing of the data lanes. On the right, the finished routing of the address lanes.

2.5.5 Power supply

The power supply circuit needs to be designed by carefully keeping in mind that it needs to handle large currents. Since the power supply uses switching regulators, it also will also produce interference with sensitive digital signals.

The main workhorse of the power supply are it's four buck converters, they will handle the larger currents. This means that we must be particularly careful when placing their external components. Figure 15 shows a simplified schematic of the buck converter. It also shows the current paths when charging and discharging the inductor. The capacitors and inductor should be placed as close as possible to the STPMIC1 in order to reduce these current paths to a minimum. On the right side of the Figure 15, we show how the components where placed in order to reduce these paths.

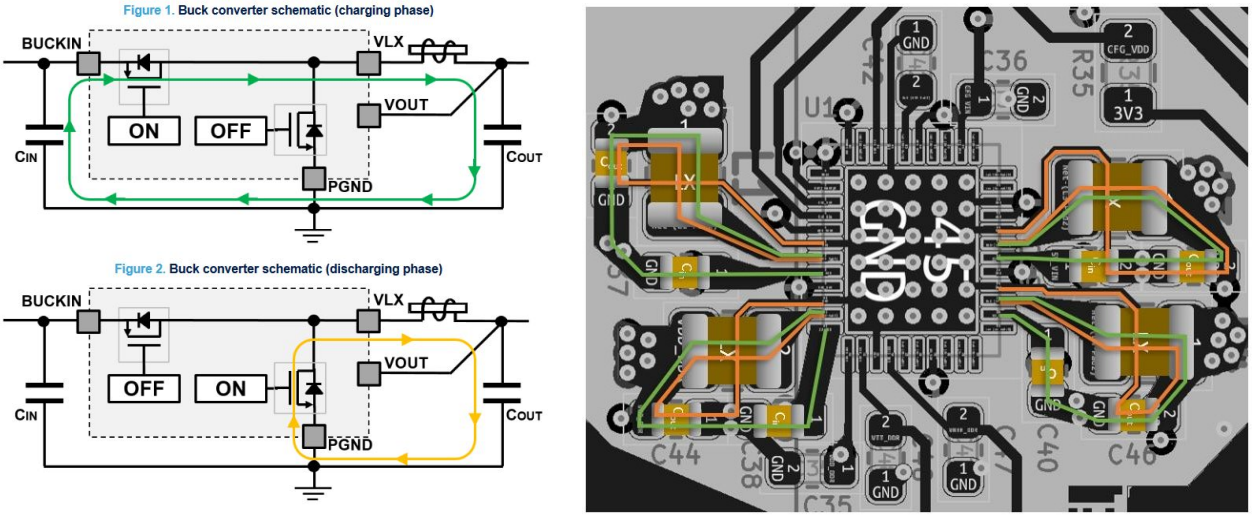


Figure 15: Critical current paths in the STPMIC1 buck converters and on the PCB.

The LDO regulators do not have a switching noise, but as they are close to the buck converters they might suffer from interference. Thus, we place decoupling capacitors close to their input and output.

2.5.6 Power delivery

To deliver power from the STPMIC1 to the rest of the board, we use the internal layer dedicated to power.
The Figure 16 shows the power delivery layer. We can see the three power supplies for the processor VDD, VDD CORE and VDD DDR, which go directly underneath its BGA package. The 3V3 supply has the largest plane as it power all the peripherals throughout the board. The VDD DDR supply covers the entire area beneath the memory connection as it serves as a reference for the differential signals.

The other supplies, generated by the LDO regulators, are routed through smaller tracks to their respective destinations. Finally, the power input, comes either from the spine or from the usb port on the right. It directly powers the right side of the sockets as well as the power supply.

The power delivery also includes the ground plane where the return currents flow back to the power supply or to the external power source. In order to avoid interference from the return currents, we divided the power plane into three sections that are connected together at single points. The most

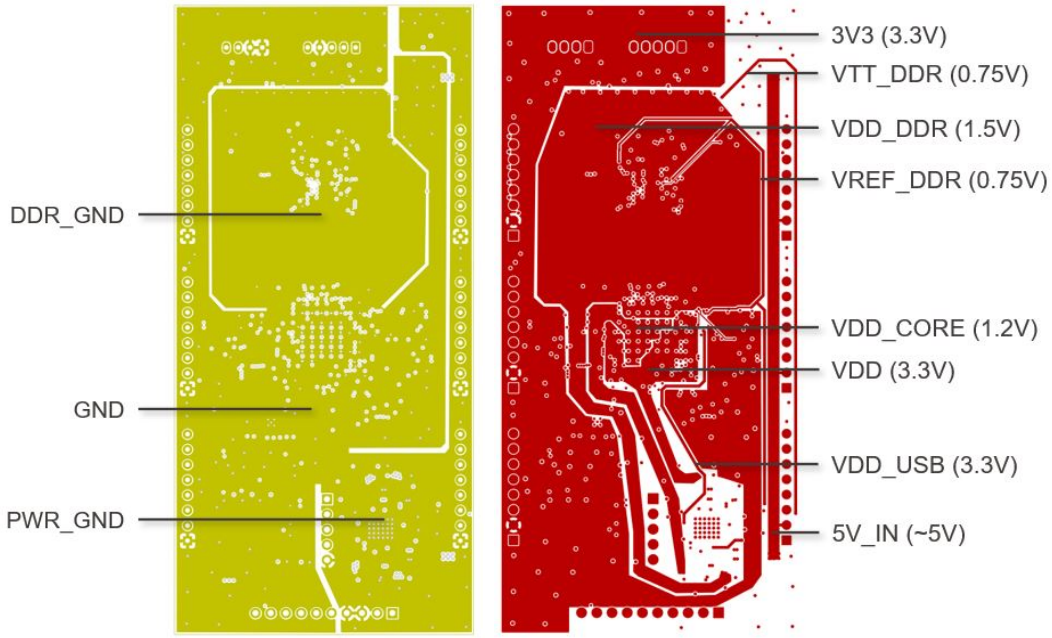


Figure 16: Description of the areas on the ground and power planes of the hostboard.

sensitive section is the memory connection reference plane and it is located below the impedance controlled memory tracks. Then comes the main ground for the peripherals and the MPU, which is only connected to the memory ground under the MPU. The last ground is the power ground, located around and under the power supply as well as the power inputs on the spine and the USB connector. This ground is only connected to the rest near the power supply in a way that no return currents from the power supply to the external power can flow near peripherals or memory.

2.5.7 Microprocessor

The 0.8mm pitch of the MPU's BGA package means that only one track can pass between two balls. This renders the connection to these balls quite complicated as they are three layers deep for the signals. The Figure 17 shows the complexity of the connections and wiring directly underneath the MPU. All the power pins are conveniently placed at the center of the BGA, which allows us to directly connect them to the power planes using vias between the balls. These same vias, which go trough the entire board, are also used to connect the appropriate decoupling capacitors on the bottom layer.

2.5.8 Peripherals

The connection to all of the peripherals as well as all of the external interfaces can be seen on the Figure 18. It was quite complicated to get all of the peripherals properly connected with minimal layer change to cross the tracks when required.

The design method to connect everything was similar to the one used for the DDR3 memory, except that here no length matching is required. We started by doing a preliminary routing trying at best to avoid crossing. Then, we did a second pass in which we eliminate all the crossing by performing layer changes with the signals. To be able to perform such crossing, one must be careful to keep the general direction of the tracks perpendicular between the two layers. This can be clearly seen on the left of the processor where we have separate vertical and horizontal areas both on the top and bottom layer. On the right of the processor, the bottom layer has almost only vertical traces when the top layer has only horizontal traces.

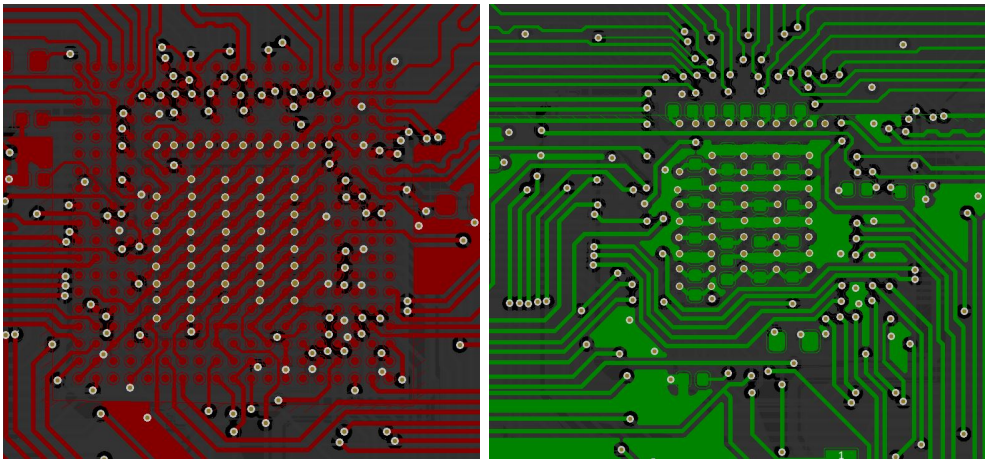


Figure 17: Closeup of the track connections to the microprocessor. In red the front layer and in green the back layer.

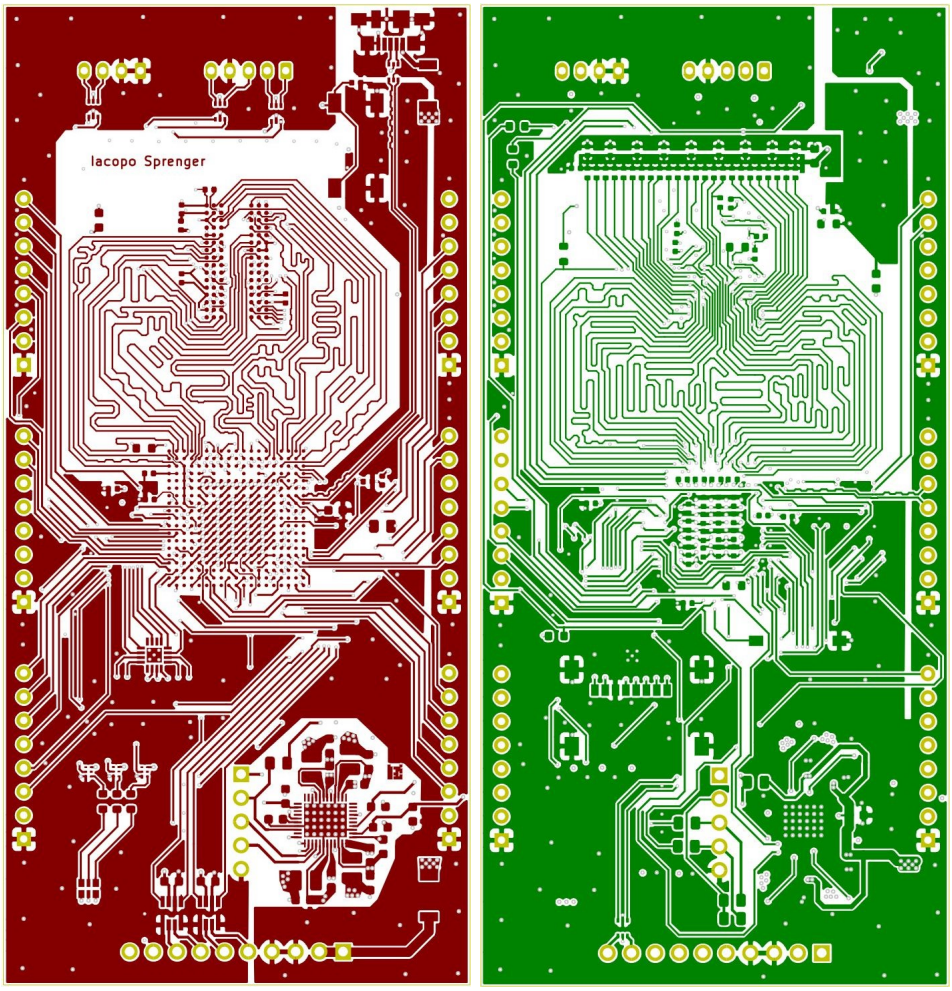


Figure 18: Pictures of the entire front layer in red and the entire back layer in green.

2.6 Board manufacture

The manufacture of the hostboard starts in China at the PCBWay factory where they etch the circuit onto a printed circuit board. The assembly and soldering of the components is done later in Switzerland at the ACI prototyping workshop.

2.6.1 PCB Manufacture

Multilayer PCBs are manufactured by sheets from the inside out. They start by coating the core with copper and etching the patterns for the power and ground planes. Then, the dielectric sheets are glued at the surfaces followed by a second round of copper and features etching. Once all the layers are done, the vias are drilled and their metal stubs are inserted. Finally, the finishing layers are applied, this is the solder mask, silkscreen and metal surface finish.

The PCB manufacturer also produces a metal stencil which has holes wherever the solder paste must be applied.

2.6.2 Solder paste application

The application of solder paste on our PCB is the first step in the assembly. The solder paste will melt in the oven and provide the bond between the components and the PCB.

The ACI is equipped with a simple and precise machine to apply this solder paste. We start by securing and tensioning the stencil on the top part of the machine. The left part of the Figure 19 shows the stencil secured in the metal clamps of machine.
Once the stencil is secured, it can be raised to clear the way for the attachment of the PCB. The PCB bust be secured using magnetic jigs as shown on the right side of the Figure 19. The PCB's position can be adjusted to closely match the stencil's position. To align accurately, we can move the PCB using the screws located on the side and bottom of the machine. Once everything is properly aligned, the stencil can be lowered and the paste can be applied using a spatula.

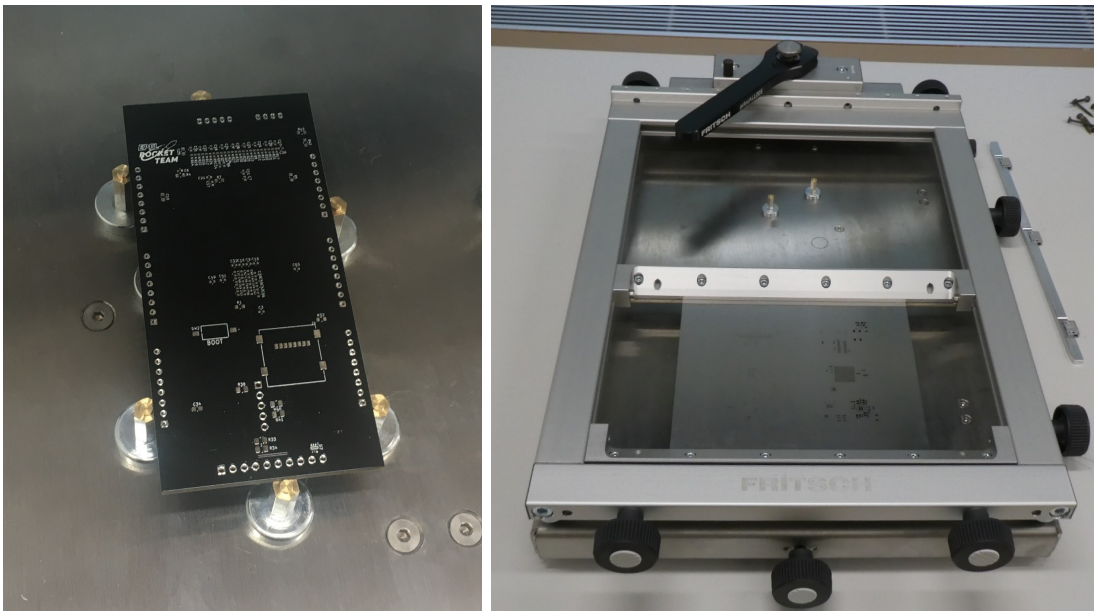


Figure 19: On the left, the hostboard secured and ready to receive a layer of solder paste. On the right, the stencil secured in metal clamps for solder paste application.

2.6.3 Pick and place

All the SMD components can be placed on the PCB with sufficient precision using the pick and place machine. The Figure 20 shows the setup we used to place our SMD components.

On the left of the image are the components located in feeding rails. These components rolls have a small piece of tape marking the values of the components to avoid errors.
In the middle of the image, there are two hostboards with the solder paste applied. These are the boards onto which we place the components. During the board design phase, all the components where clearly labelled on the silkscreen such that we can see the components code while using the pick and place machine.
On the right side of the image there is a print of the BOM, which allows us to quickly know what component is of what type.

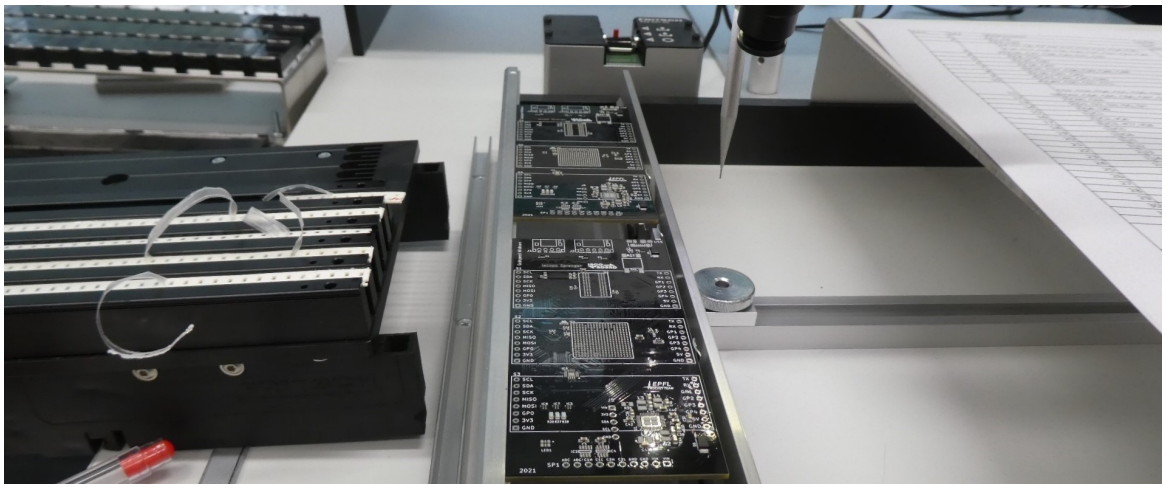


Figure 20: View of the pick and place machine with the components feeding rails, the hostboards and a printout of the BOM for the user.

The Figure 21 shows the machine in operation, where we are about to place one of the processor's decoupling capacitors.

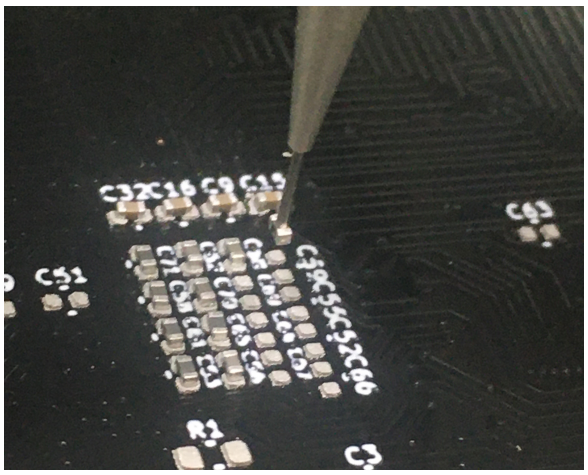


Figure 21: Closeup of the pick and place machine used to place a decoupling capacitor.

2.6.4 Microplacer

BGA and QFN components can only be placed with the help of the microplacer as there is no way to visually align them when looking from the top. This is because their leads do not appear on the side of the package.
The microplacer has an optical system which is capable of looking the PCB and the bottom of the component at the same time. This allows very precise alignment of components without seeing the leads.

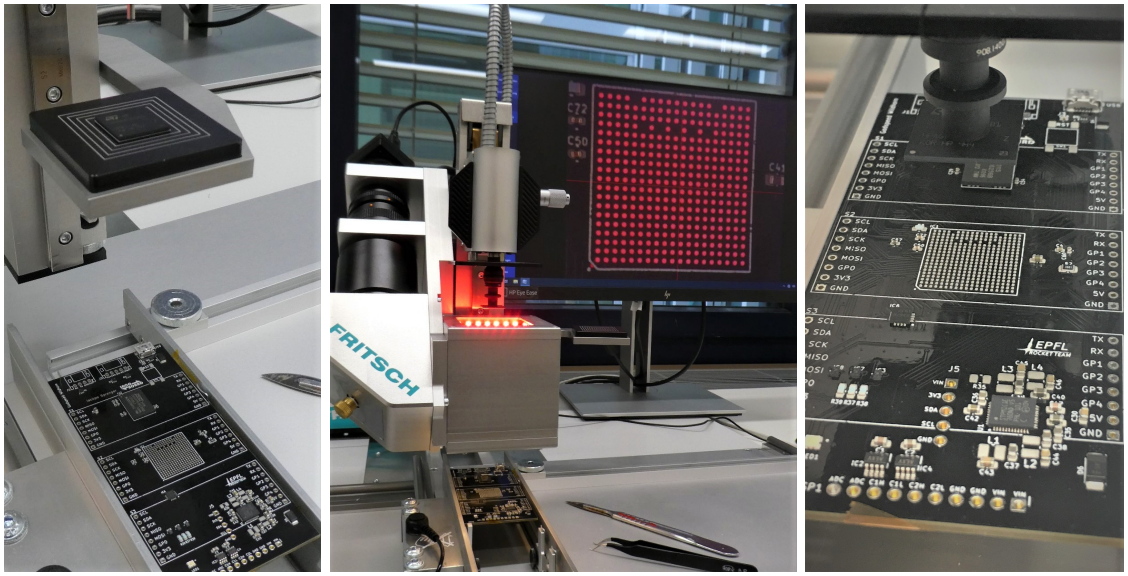


Figure 22: View of the microplacer. On the left, preparation of the component. In the middle, alignement of the component. On the right, deposition of the component.

We start by placing the BGA component on the receptacle as it is shown on the left of Figure 22. The component is then picked up with a vacuum and the optical system comes placed between the PCB and the component. This step can be seen in the middle of Figure 22. On the screen, we can see the superposition of the BGA pads in red with a picture of the PCB in colors. At this point, the PCB can be moved to perfect the alignment with the BGA.
The last step, visible on the right of the Figure 22 is to remove the optical system and lower the component in it's place.

2.6.5 Vapour phase oven

The last machine we used for soldering is the vapour phase oven. This is a special oven which allows a very uniform heating of the PCB and thus an uniform melting of the solder paste. This yields very good results for the soldering of SMD and BGA components.
The usage is straightforward, the oven needs to be preheated and then we simply need to place the PCB inside and start the cycle.

There is one special case that needs to be considered. If the PCB has components on both layers, it must be baked twice in the oven. However the side that gets baked a second time runs the risk of having the components fall down as it is usually facing downwards. To prevent this, we secure the components on the back with Kapton tape as show in Figure **??**. It is best not to do this on the side where the BGA components are located as their alignment is quite critical. Additionally, in case they do detach, it would be impossible to easily reattach them by reworking the boards by hand.

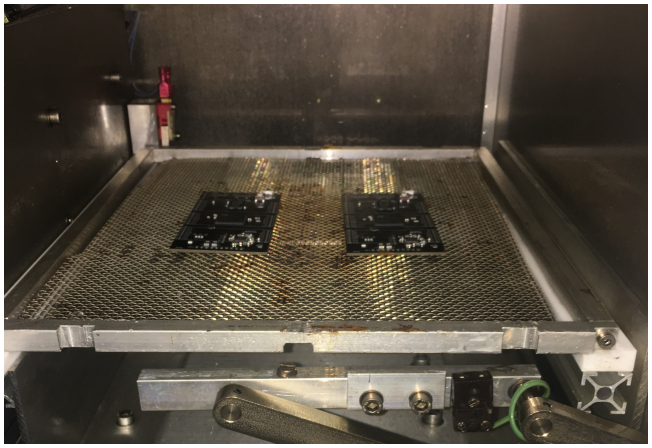


Figure 23: View of the upper chamber of the oven. The boards will descend in the lower chamber once the door is closed and the cycle is started.

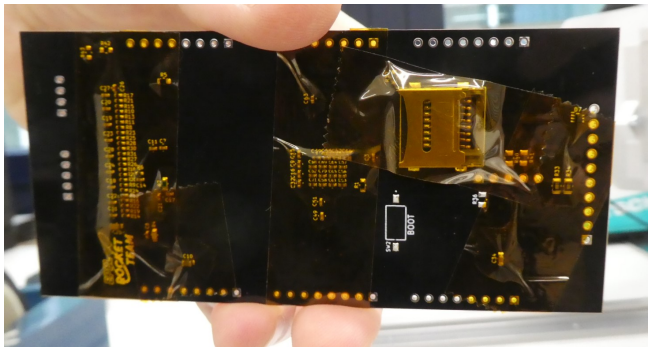


Figure 24: View of the back of a hostboard and its components secured with kapton tape for the second baking.

Once out of the oven, the PCBs simply need to cool down and they are ready for the next step.

2.6.6 Assembly steps

With all of these machines, we are able to assemble two hostboards in about eight hours by carefully following the steps highlighted here.

- Apply the solder paste on the bottom side of the PCB.
- Inspect the quality of the paste application, if it is not satisfactory, wipe it clean with isopropyl alcohol and start again.
- Use the pick and place machine to set every SMD component on the bottom side except R36, R40 and R41.
- Inspect the alignment of the components and adjust them with a scalpel if necessary.
- Place the PCBs in the oven to bake the bottom side and let them cool down.
- Secure all of the components on the bottom side with Kapton tape.
- Apply the solder paste on the top layer of the PCB.

- Inspect the quality of the paste application, if it is not satisfactory, wipe it clean with isopropyl alcohol and start again.
- Use the pick and place machine to set every SMD component on the bottom side except R35.
- Inspect the alignment of the components and adjust them with a scalpel if necessary.
- Use the microplacer to place the BGA components. Place the SDRAM and the MPU at the last moment to reduce the risk of moving them.
- If the BGA components are not well aligned remove and clean everything with isopropyl alcohol and go back to the top layer paste application.
- Place the PCBs in the oven to bake the top side and let them cool down.
- Remove the Kapton tape and inspect the result.
- Program the power supply using a raspberry pi and the python program in appendix.
- Solder the missing resistors and the THT connectors.

3 Results

3.1 Final product

The hostboard can be seen assembled and ready to use in Figure 25. All the components where successfully soldered at the ACI workshop. An inspection with the workshop's microscope allowed us to see that everything is soldered properly. We also ran some additional conductivity tests on key points to determine that there are no short circuits.

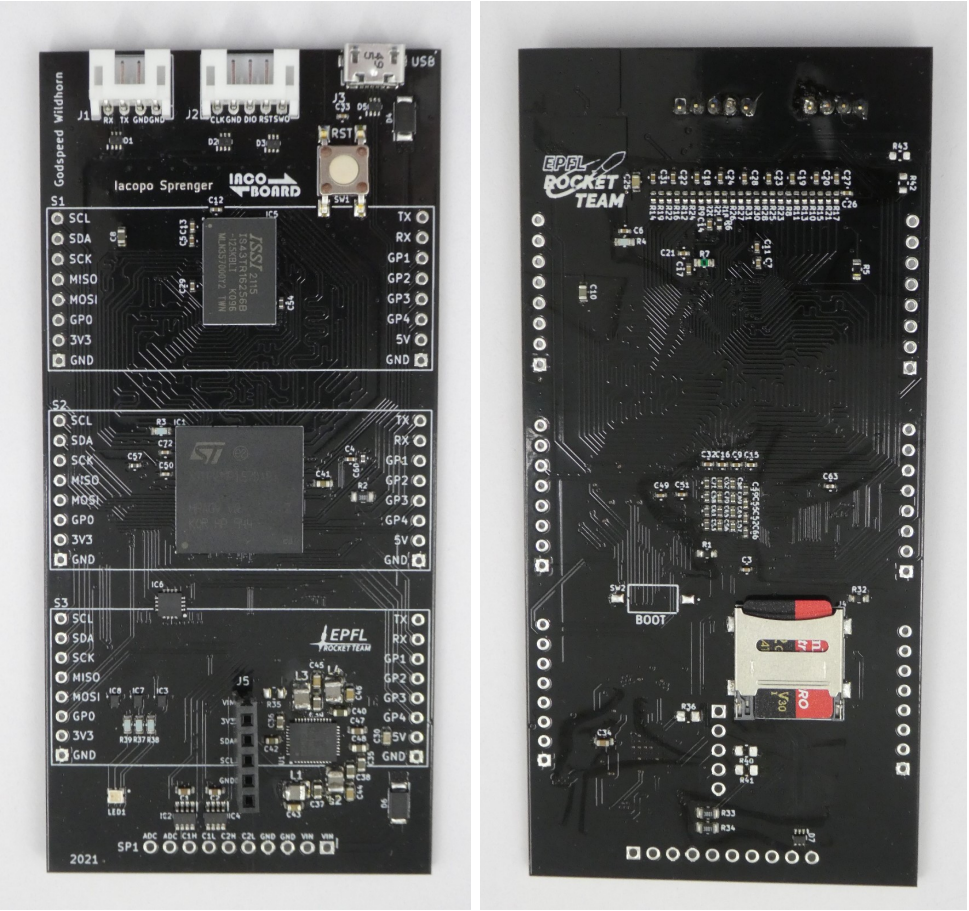


Figure 25: Pictures of the finished hostboard, front and back.

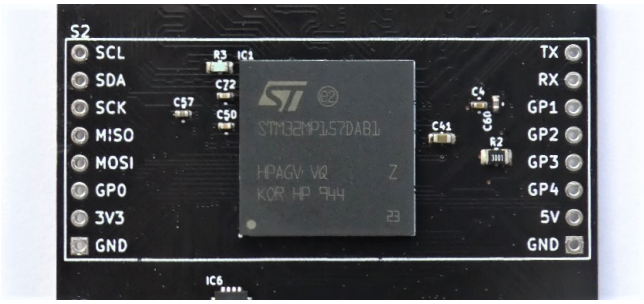


Figure 26: Detailed view of the microprocessor on the finished hostboards.

3.2 Power supply

Using a raspberry pi and the small python program, the power supply could be successfully reprogrammed to meet our needs.

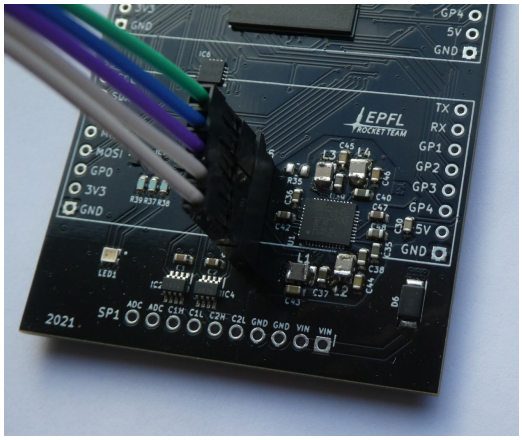


Figure 27: View of the power supply with the I^2C programming cables attached.

We are then able to inspect the quality of the power supply by measuring the outputs with a multimeter. When the board is powered, VDD CORE, VDD and 3V3 quickly reach their desired voltages. To inspect the VDD DDR supply, one must wait that the processor is ready to turn it on through the I^2C interface. Once this is the case, the supply reaches it's expected value quickly.

3.3 Computer connection

We are able to connect the hostboard to a computer using the three available debug interfaces. The device is able to boot from USB and UART when connected to the STM32CUBE programmer. We are also able to recognise the debugger by using an external st-link device connected to the SWD port.

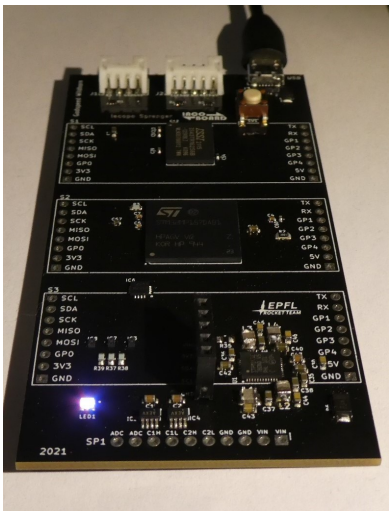


Figure 28: View of the hostboard connected to a computer by USB with its debug led turned on.

The Figure 28 shows how we are able to turn on the debug led with a very simple program on the Cortex-M4 core. However thorough testing of the Cortex-A7 and the memory subsystem is still necessary.

4 Discussion

4.1 Conclusion

The purpose of this project was to develop a new microprocessor hostboard for the EPFL Rocket Team. We decided to develop a powerful and future proof system by including the brand new STM32MP157 series of microprocessors on our hostboard. This came with it's design challenges going from the fine pitch BGA package to the required external memory and storage, passing by the complex multi rail power supply. This report outlines the design choices and methodology to include such a processor into a board design.

The most challenging part of this project was the implementation of the connection to the high speed DDR3 memory. This interface has many constraints due to the sensitive nature of high speed signals. This part allowed us to learn how to design with these constraints in mind. This was clearly the longest part of the project as it required many iterations to get the memory routing quite right.
The second challenging design task was the power supply. This subsystem taught us to design with higher currents in mind. It also required us to properly deliver all those power rails at their final location without interfering with sensitive components. And finally, all of these currents must also return to their source without interfering. This is done thanks to our carefully designed ground planes.
The last technical challenge of this project was to route all the connection of our board on only two signal layers.

Besides these technical challenges, we also had to face the semiconductor shortages. This lead us to buying slightly different components and reprogramming them during production. It also greatly reduced the range of available components from which we could choose for our designs.

In the end, this was an immensely rewarding project where I was able to learn a lot of new things by doing the hands on development of a product which will be very useful for the different projects inside the EPFL Rocket Team.

4.2 Future work

In the near future, the hostboards will be stress tested using the DDR3 test suite. This will allow us to tune the memory controller parameters for optimal performance.

Once the calibration is done, we will tackle the development of the software that will run on the hostboards during the rocket flights. This will be accompanied by the development and production of the extension boards which come on the hostboard's sockets.

In the more distant future, we plan to develop a second version of these hostboards with soldered storage memory and a better spine attachment system. The new version will also have minor improvements such a support for FDCAN low power modes.

4.3 Acknowledgements

I would like to thank Alexandre Schmid for his supervision of the project throughout the year. I would also like to thank René Beuchat and Andrea Guerreri who gave me precious advice on how to tackle the memory connection routing. The ACI and Marcel Groccia where of a great help for the assembly of the hostboards which was carried out in their workshop. Finally I would like to thank the Avionics team of the EPFL Rocket Team for their support and ideas for this project.

References

[1] Datasheet. *EClamp2357NQ.* SEMTECH, 2017.

[2] Datasheet. *DDR3 Memory IS43/46TR16256B.* ISSI, 2020.

[3] Datasheet. *+5V, 4Mbps CAN Transceiver with ±40V Fault Protection, ±25V CMR, and ±40kV ESD in 8-Pin SOT23.* ANALOG DEVICES, 2021.

[4] Datasheet. *DS12504 Arm® dual Cortex®-A7 800 MHz + Cortex®-M4 MPU, 3D GPU, TFT/DSI, 37 comm. interfaces, 29 timers, adv. analog.* STMicroelectronics, 2021.

[5] Datasheet. *DS12792 Highly integrated power management IC for micro processor units.* STMicroelectronics, 2021.

[6] KiCAD EDA. `https://www.kicad.org/`.

[7] Paul Horowitz and Winfield Hill. *The art of electronics.* Cambridge University Press, New York, 3rd. edition, 2015.

[8] Brian W. Kernighan and Dennis M. Ritchie. *The C Programming Language.* Prentice Hall Professional Technical Reference, 2nd edition, 1988.

[9] Reference Manual. *RM0436 STM32MP157 advanced Arm®-based 32-bit MPUs.* STMicroelectronics, 2021.

[10] Application Note. *AN5122 STM32MP1 Series DDR memory routing guidelines.* STMicroelectronics, 2019.

[11] Application Note. *AN5431 The STPMIC1 PCB layout guidlines.* STMicroelectronics, 2019.

[12] Application Note. *AN5440 The STPMIC1 I²C programming guide.* STMicroelectronics, 2020.

[13] Application Note. *AN5031 Getting started with STM32MP151, STM32MP153 and STM32MP157 line hardware development.* STMicroelectronics, 2021.

[14] Application Note. *AN5168 DDR configuration on STM32MP1 Series MPUs.* STMicroelectronics, 2021.

A Power supply programming

```
"""
small utlility used to program the stpmic1
for use with the ERT hostboards.
Connect the raspberry pi to the stpmic1:
RPI      HB
5V    to VIN
3V3   to 3V3
GPIO2 to SDA
GPIO3 to SCL
GND   to GND
run the following commands to install dependencies
and check presence of i2c slave:
    sudo apt-get install i2c-tools
    sudo i2cdetect -y 1
Modify the i2c device address accordingly
"""
import smbus
import time

DEVICE_BUS = 1
DEVICE_ADDR = 0x33
CONFIG = [  (0xF8, 0xEE),
            (0xF9, 0x92),
            (0xFA, 0xC0),
            (0xFB, 0x02),
            (0xFC, 0xFE),
            (0xFD, 0xB0),
            (0xFE, 0x02)]

bus = smbus.SMBus(DEVICE_BUS)
print("Writing to memory...")
for c in CONFIG:
    print("writing {:02X} at {:02X}".format(c[1], c[0]))
    bus.write_byte_data(DEVICE_ADDR, c[0], c[1])
bus.write_byte_data(DEVICE_ADDR, 0xB9, 0x01) #write to NVM
print("done!")
time.sleep(0.5);
print("checking memory...")
bus.write_byte_data(DEVICE_ADDR, 0xB9, 0x03) #read from NVM
time.sleep(0.5);
for c in CONFIG:
    print("reading at {:02X} ...".format(c[0]), end='')
    data = bus.read_byte_data(DEVICE_ADDR, c[0])
    if(data == c[1]):
        print(" read {:02X} PASS".format(data))
    else:
        print(" read {:02X} FAIL ({:02X})".format(data, c[1]))
print("done!")
```

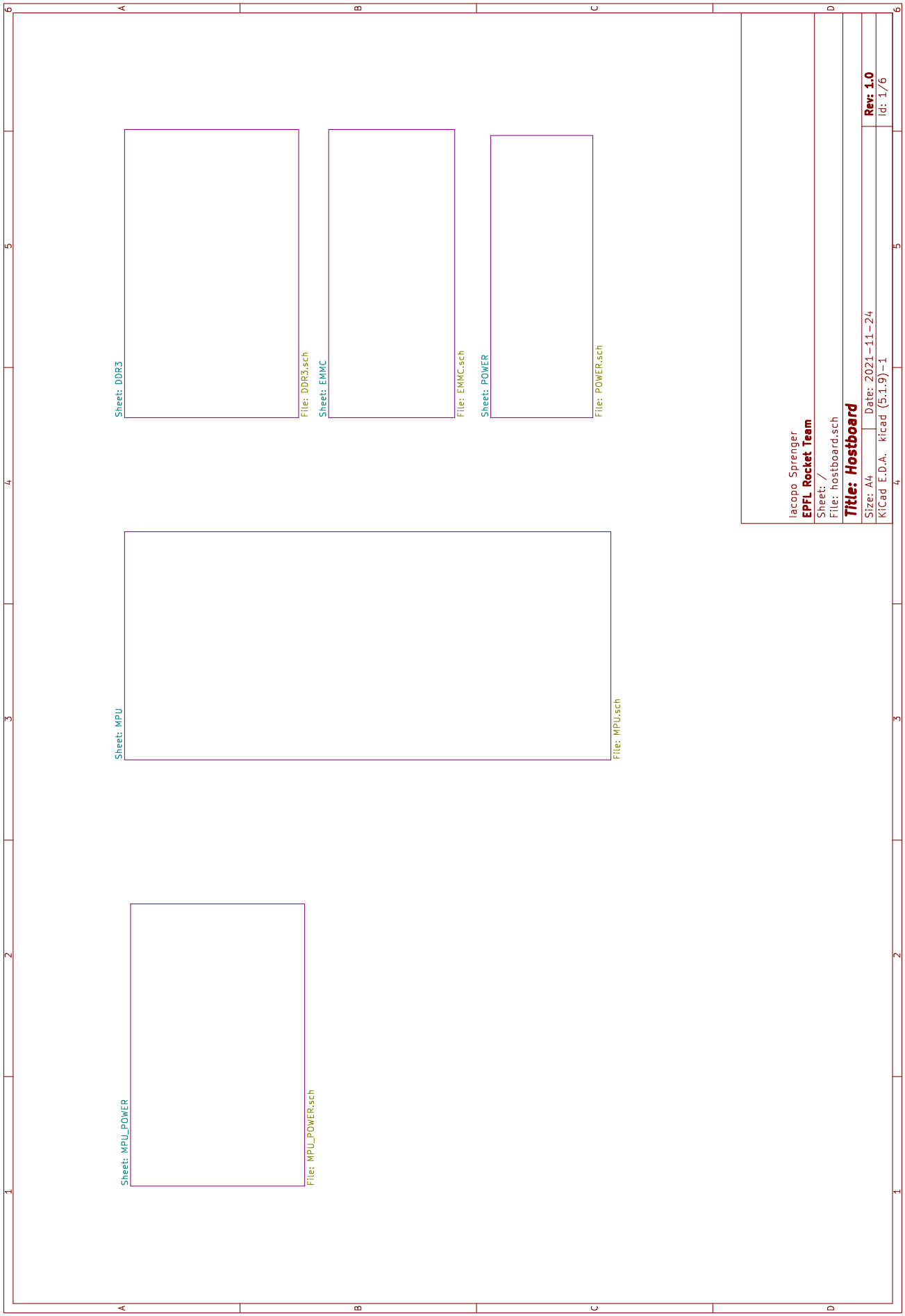
B Length matching spreadsheets

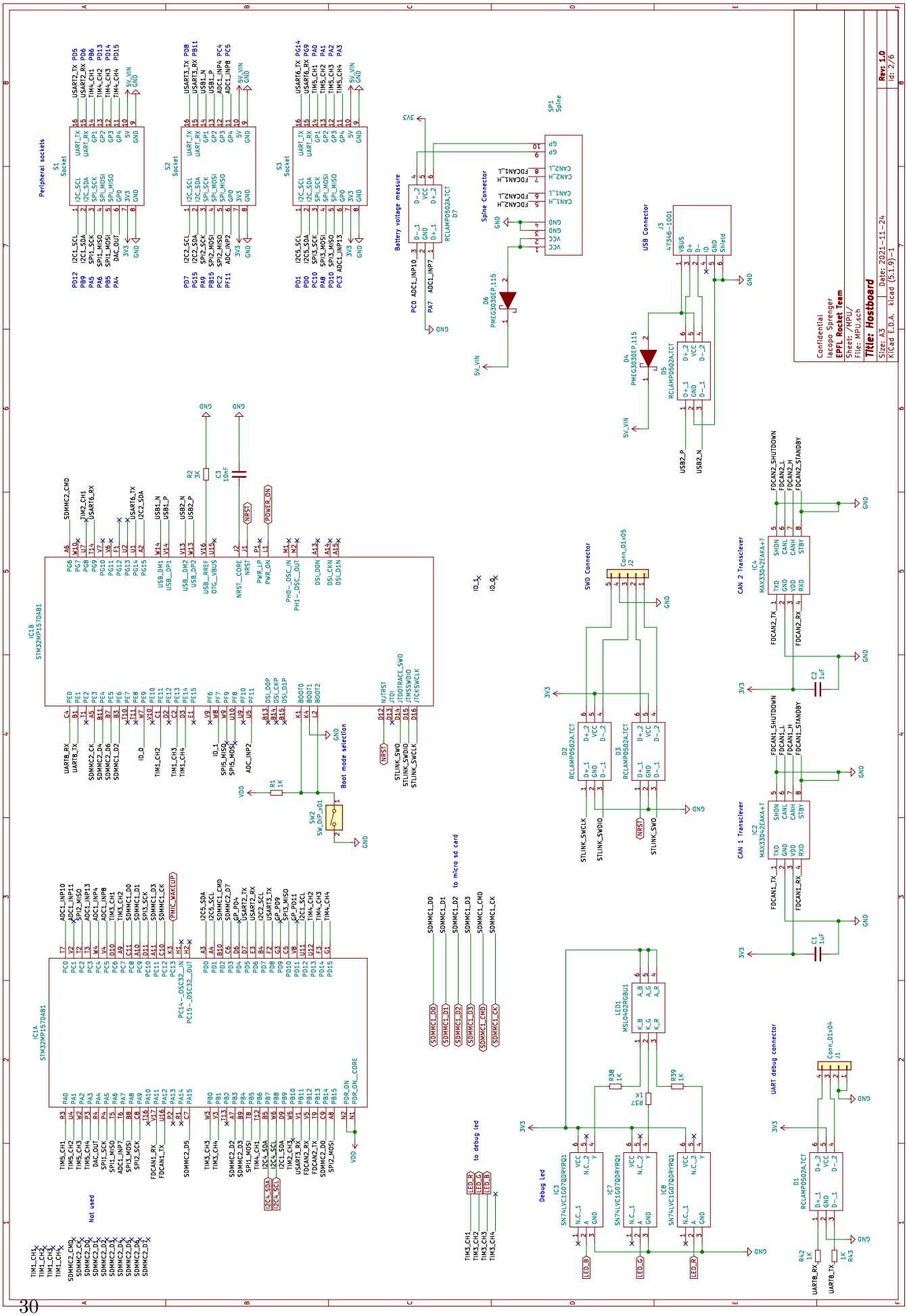
STM32MP15XXAB					
NET NAME	STM32MP15XXAB LENGTH (mm)	length without vias	STM32MP15XXAB to memory (mm)	TOTAL LENGTH (mm)	DELTA WITH ((CLK_P+CLK_N)/2): from -1.016 to 0 mm
A0	5.19	50.313	53.513	58.703	-0.612
A1	6.31	49.25	52.45	58.76	-0.555
A2	5.63	50.012	53.212	58.842	-0.473
A3	6.58	48.923	52.123	58.703	-0.612
A4	5.55	50.164	53.364	58.914	-0.401
A5	3.79	51.742	54.942	58.732	-0.583
A6	4.06	51.773	54.973	59.033	-0.282
A7	7.8	47.856	51.056	58.856	-0.459
A8	6.62	49.07	52.27	58.89	-0.425
A9	4.05	51.673	54.873	58.923	-0.392
A10	6.59	49.081	52.281	58.871	-0.444
A11	5.69	50.144	53.344	59.034	-0.281
A12	4.57	50.918	54.118	58.688	-0.627
A13	6.34	49.286	52.486	58.826	-0.489
A14	6.65	48.857	52.057	58.707	-0.608
A15	4.11		3.2	7.31	-52.005
BA0	6.01	49.56	52.76	58.77	-0.545
BA1	6.76	48.731	51.931	58.691	-0.624
BA2	5.81	49.787	52.987	58.797	-0.518
CASN	4.85	50.695	53.895	58.745	-0.57
CKE	6.66	49.104	52.304	58.964	-0.351
CLK_N	6.03	50.06	53.26	59.29	
CLK_P	6.09	50.05	53.25	59.34	
CSN	5.86	49.701	52.901	58.761	-0.554
ODT	6.47	49.22	52.42	58.89	-0.425
RASN	5.21	50.508	53.708	58.918	-0.397
WEN	4.69	50.79	53.99	58.68	-0.635

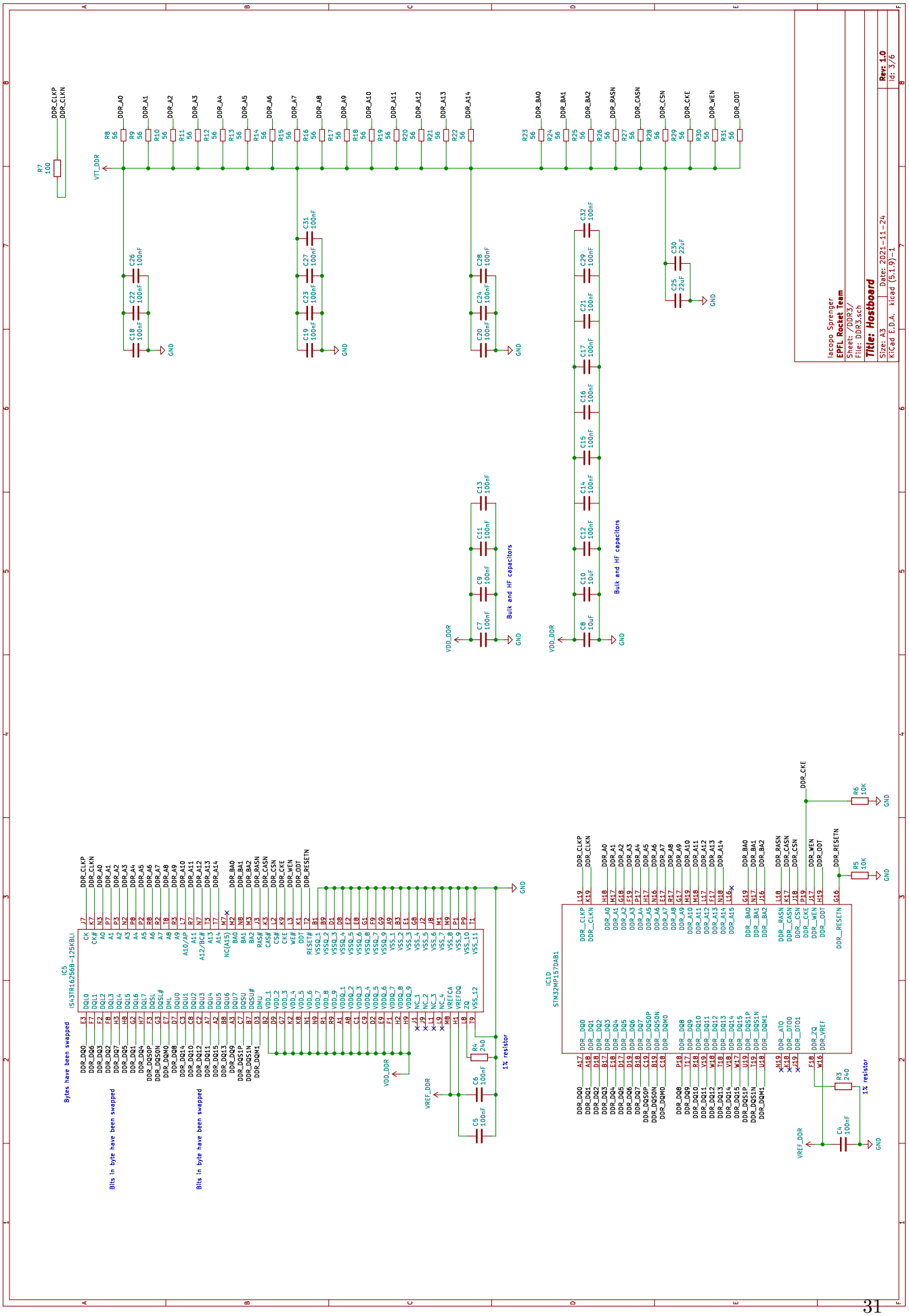
Figure 29: Spreadsheet used to compute the trace lengths for the DDR3 address bus. A14 is unused because we do not cover the entire address space.

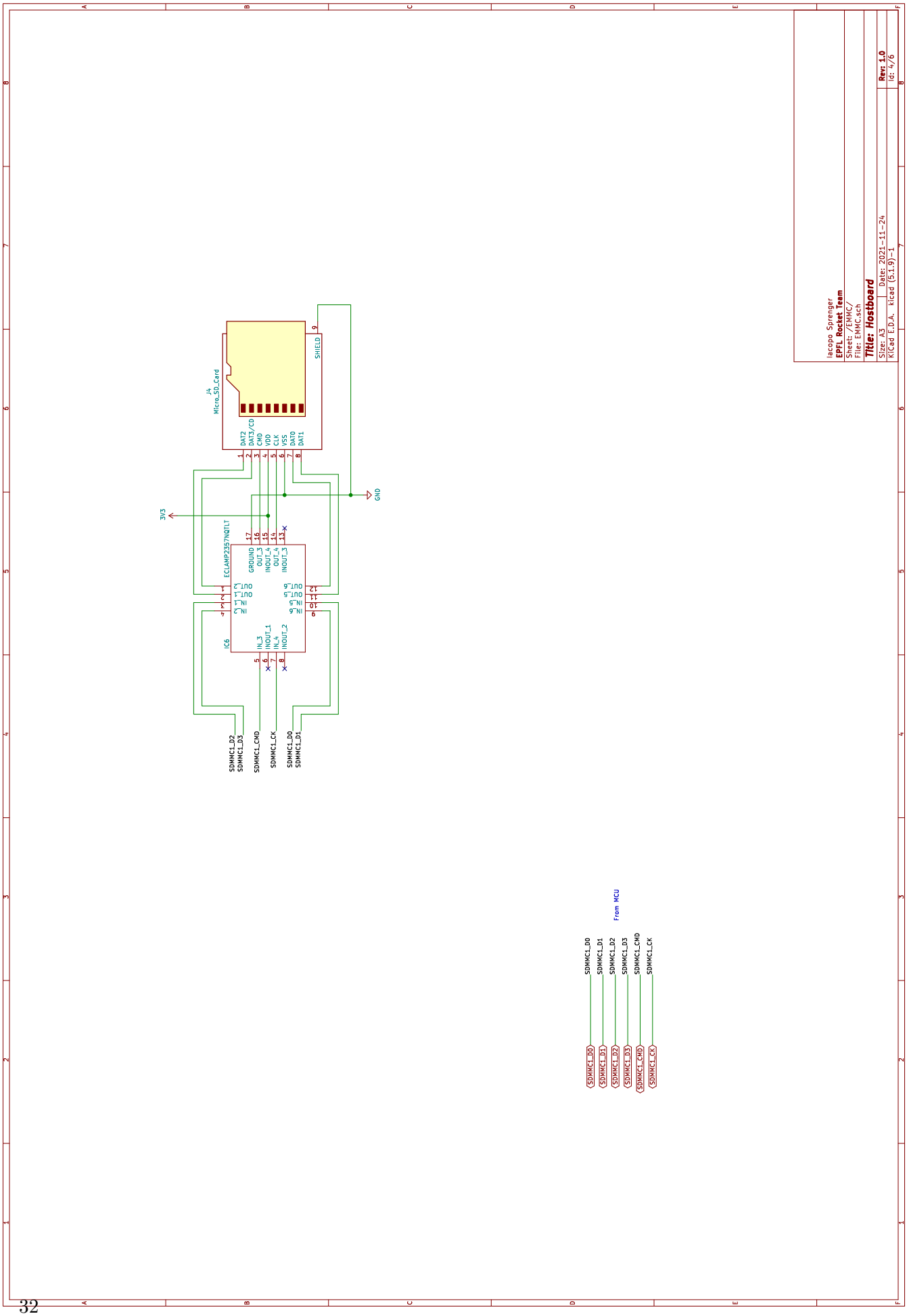
STM32MP15XXAB						
	NET NAME	STM32MP15XXAB LENGTH (mm)	STM32MP15XXAB to memory (mm)	TOTAL LENGTH (mm)	DELTA WITH ((DQSn_P+DQSn_N)/2) MAX: +/- 1.016 mm	DELTA WITH ((CLK_P+CLK_N)/2): from -15 to 0 mm
Byte 0	DQ0	8.17	49.933	58.103	0.061	
	DQ1	8.19	49.966	58.156	0.114	
	DQ2	6.92	51.179	58.099	0.057	
	DQ3	6.97	51.155	58.125	0.083	
	DQ4	6.7	51.411	58.111	0.069	
	DQ5	7.63	50.491	58.121	0.079	
	DQ6	7.33	50.728	58.058	0.016	
	DQ7	7.47	50.655	58.125	0.083	
	DQM0	6.76	51.379	58.139	0.097	
	DQS0_P	8.08	49.931	58.011		-1.273
	DQS0_N	8.09	49.983	58.073		
Byte 1	DQ8	5.36	52.431	57.791	0.011	
	DQ9	7.37	50.376	57.746	-0.034	
	DQ10	6.67	51.131	57.801	0.021	
	DQ11	7.66	50.141	57.801	0.021	
	DQ12	7.85	49.99	57.84	0.06	
	DQ13	6.8	50.993	57.793	0.013	
	DQ14	7.23	50.585	57.815	0.035	
	DQ15	7.9	49.935	57.835	0.055	
	DQM1	6.83	50.969	57.799	0.019	
	DQS1_P	7.26	50.52	57.78		-1.535
	DQS1_N	7.26	50.52	57.78		

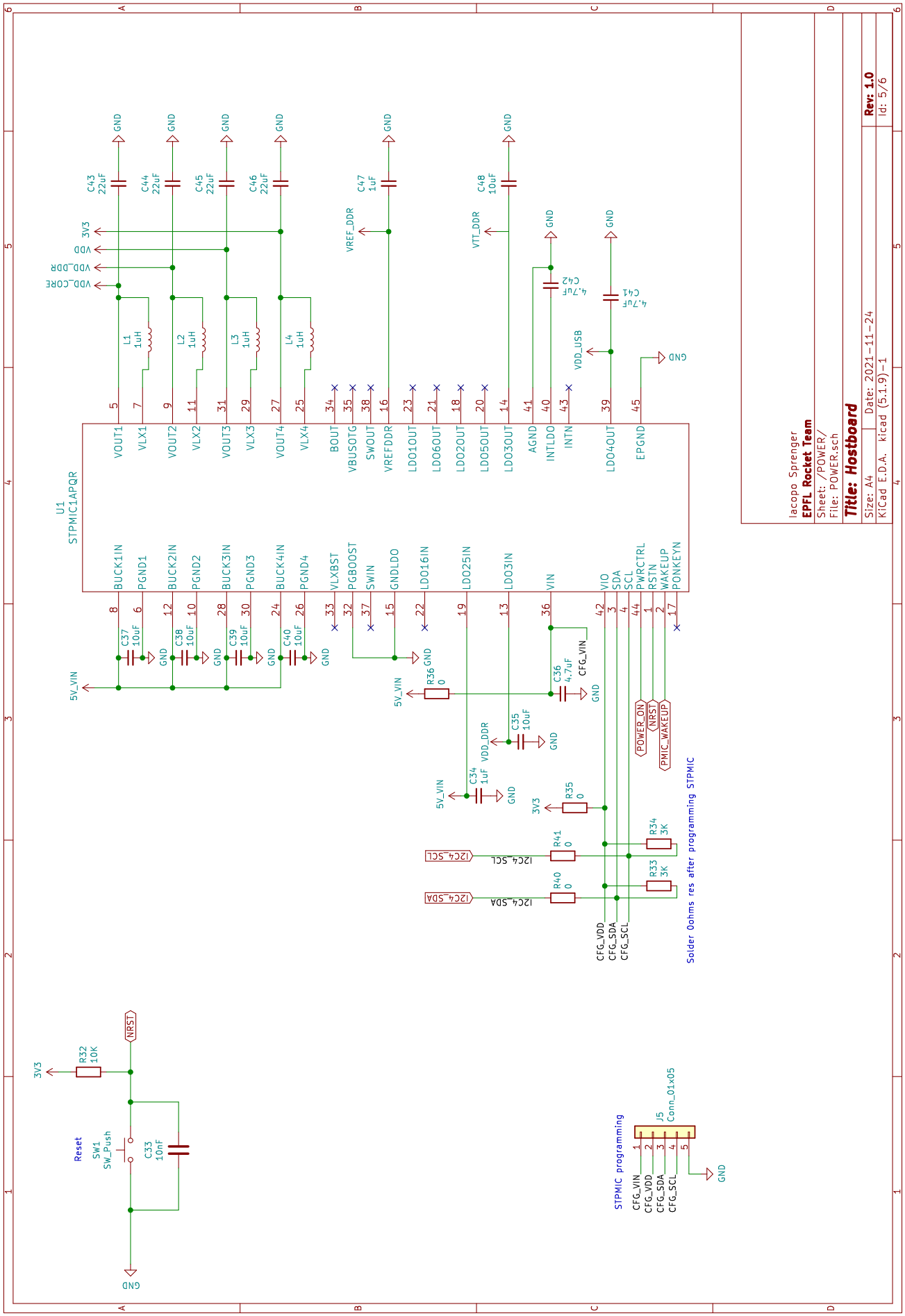
Figure 30: Spreadsheet used to compute the trace lengths for the DDR3 data buses.

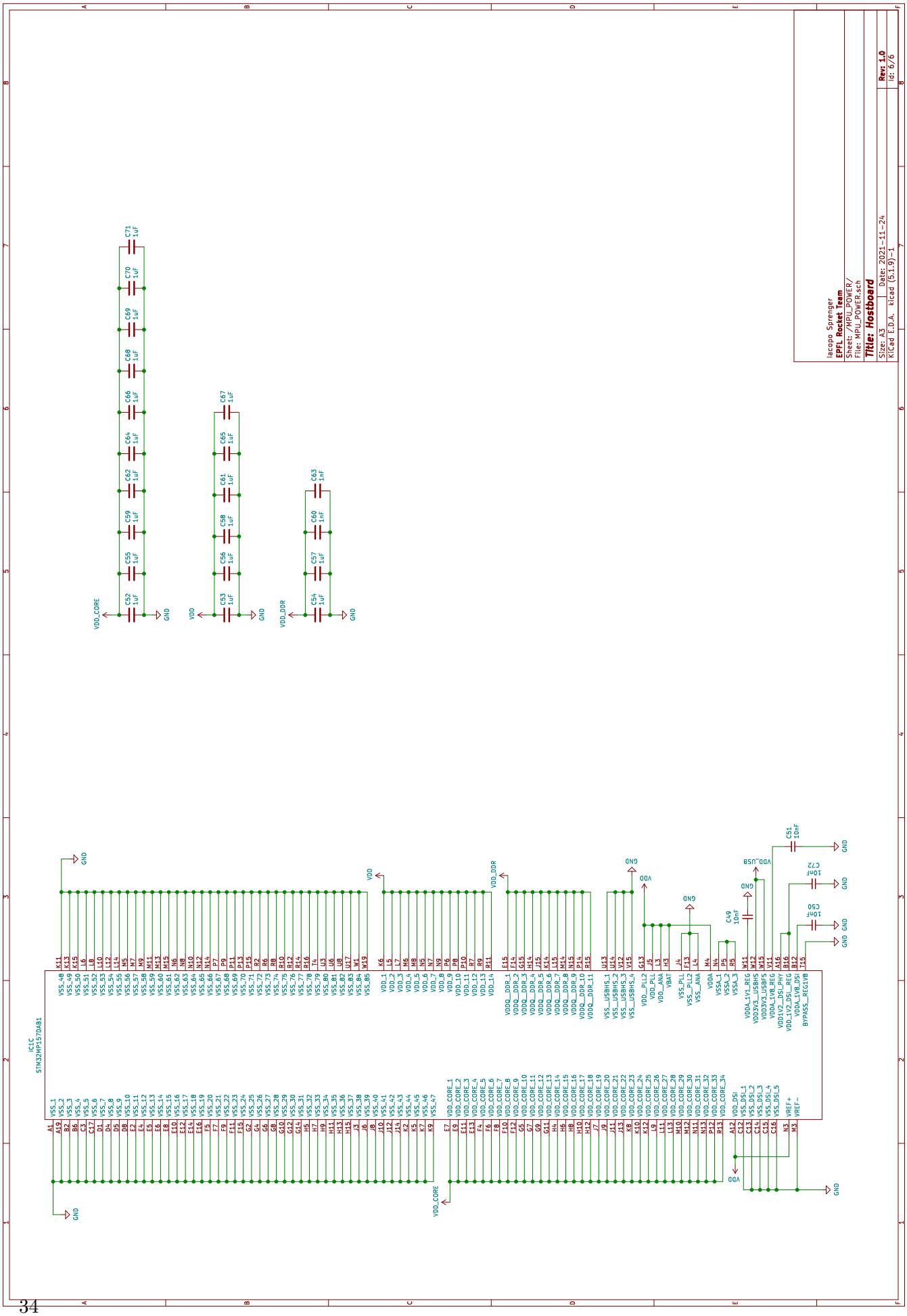

Sheet: DDR3
File: DDR3.sch
Sheet: EMMC
File: EMMC.sch
Sheet: POWER
File: POWER.sch
Sheet: MPU
File: MPU.sch
Sheet: MPU_POWER
File: MPU_POWER.sch
Iacopo Sprenger
EPFL Rocket Team
Sheet: /
File: hostboard.sch
Title: Hostboard
Size: A4
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 1/6

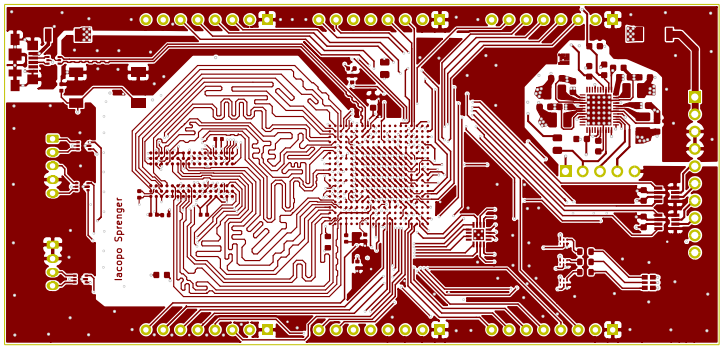

Peripheral sockets
S1 Socket
S2 Socket
S3 Socket
IC1A STM32MP157DAB1
IC1B STM32MP157DAB1
Not used
Debug led
to debug led
to micro sd card
Boot mode selection
SWD Connector
Spine Connector
Battery voltage measure
USB Connector
CAN 1 Transciever
CAN 2 Transciever
UART debug connector
Confidential
Iacopo Sprenger
EPFL Rocket Team
Sheet: /MPU/
File: MPU.sch
Title: Hostboard
Size: A3
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev 1.0
Id: 2/6


Iacopo Sprenger
EPFL Rocket Team
Sheet: /DDR3/
File: DDR3.sch
Title: Hostboard
Size: A3
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 3/6
Bulk and HF capacitors
Bits in byte have been swapped
1% resistor
VTT_DDR
VDD_DDR
VREF_DDR
GND

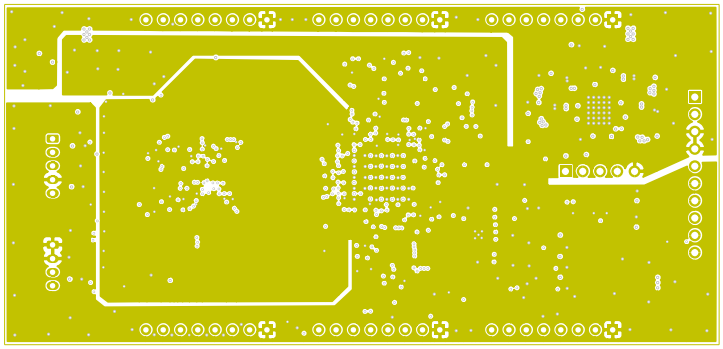

J4
Micro_SD_Card
SHIELD
DAT2
DAT3/CD
CMD
VDD
CLK
VSS
DAT0
DAT1
3V3
GND
IC6
ECLAMP2357NQTLT
GROUND
OUT_3
IN/OUT_4
OUT_4
IN/OUT_3
OUT_1
IN_1
OUT_2
IN_2
OUT_5
IN_5
OUT_6
IN_6
IN_3
IN/OUT_1
IN_4
IN/OUT_2
SDMMC1_D2
SDMMC1_D3
SDMMC1_CMD
SDMMC1_CK
SDMMC1_D0
SDMMC1_D1
From MCU
SDMMC1_D0
SDMMC1_D1
SDMMC1_D2
SDMMC1_D3
SDMMC1_CMD
SDMMC1_CK
Iacopo Sprenger
EPFL Rocket Team
Sheet: /EMMC/
File: EMMC.sch
Title: Hostboard
Size: A3
Date: 2021-11-24
Rev: 1.0
KiCad E.D.A. kicad (5.1.9)-1
Id: 1/6


U1
STPMIC1APQR
VOUT1
VLX1
VOUT2
VLX2
VOUT3
VLX3
VOUT4
VLX4
BOUT
VBUSOTG
SWOUT
VREFDDR
LDO1OUT
LDO6OUT
LDO2OUT
LDO5OUT
LDO3OUT
AGND
INTLDO
INTN
LDO4OUT
EPGND
BUCK1IN
PGND1
BUCK2IN
PGND2
BUCK3IN
PGND3
BUCK4IN
PGND4
VLXBST
PGBOOST
SWIN
GNDLDO
LDO1IN
LDO2SIN
LDO3IN
VIN
VIO
SDA
SCL
PWRCTRL
RSTIN
WAKEUP
PONKEYN
L1 1uH
L2 1uH
L3 1uH
L4 1uH
C43 22uF
C44 22uF
C45 22uF
C46 22uF
C47 1uF
C48 10uF
C42 4,7uF
C41 4,7uF
C37 10uF
C38 10uF
C39 10uF
C40 10uF
C36 4.7uF
C35 10uF
C34 1uF
R36 0
R35 0
R41 0
R40 0
R34 3K
R33 3K
R32 10K
C33 10nF
SW1 SW_Push
Reset
J5 Conn_01x05
STPMIC programming
CFG_VIN
CFG_VDD
CFG_SDA
CFG_SCL
GND
3V3
VDD
VDD_DDR
VDD_CORE
VREF_DDR
VTT_DDR
VDD_USB
5V_VIN
VDD_DDR
I2C4_SCL
I2C4_SDA
NRST
POWER_ON
PWIC_WAKEUP
Solder 0ohms res after programming STPMIC
Iacopo Sprenger
EPFL Rocket Team
Sheet: /POWER/
File: POWER.sch
Title: Hostboard
Size: A4
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 5/6

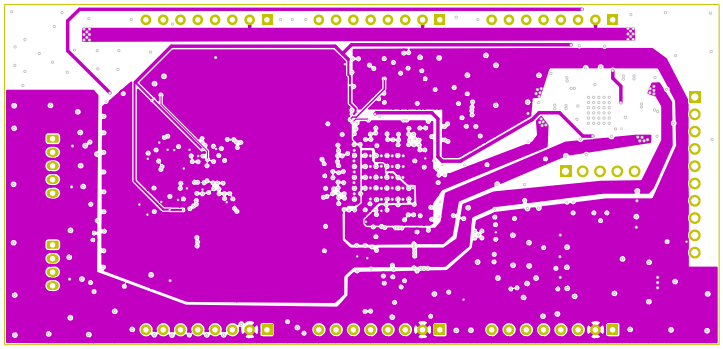

Iacopo Sprenger
EPFL Rocket Team
Sheet: /MPU_POWER/
File: MPU_POWER.sch
Title: Hostboard
Size: A3
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 6/6
VDD_CORE
VDD
VDD_DDR
GND
C52 1uF
C53 1uF
C54 1uF
C55 1uF
C56 1uF
C57 1uF
C58 1uF
C59 1uF
C60 1nF
C61 1uF
C62 1uF
C63 1nF
C64 1uF
C65 1uF
C66 1uF
C67 1uF
C68 1uF
C69 1uF
C70 1uF
C71 1uF
IC1C
STM32MP157DAB1
C49 10nF
C50 10nF
C51 10nF
C72 10nF
VDD_USB



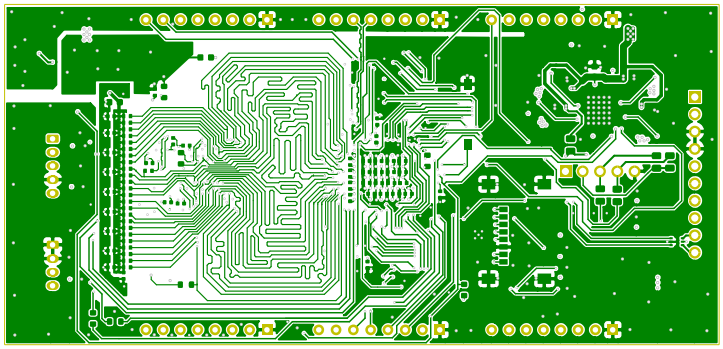
Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4 | Date: 2021-11-24 | Rev: 1.0
KiCad E.D.A. kicad (5.1.9)-1 | Id: 1/1



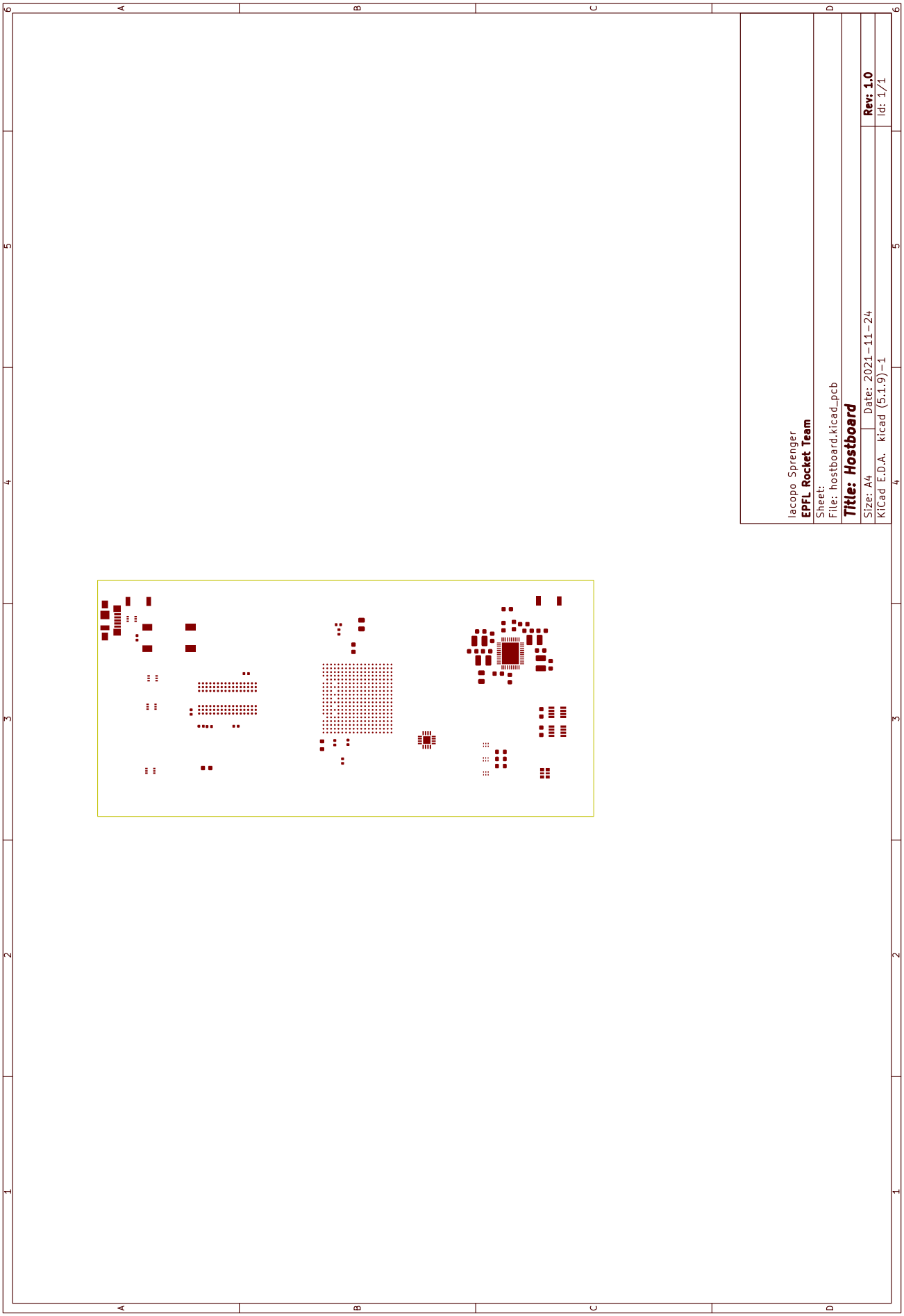
Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4 | Date: 2021-11-24 | **Rev: 1.0**
KiCad E.D.A. kicad (5.1.9)-1 | Id: 1/1

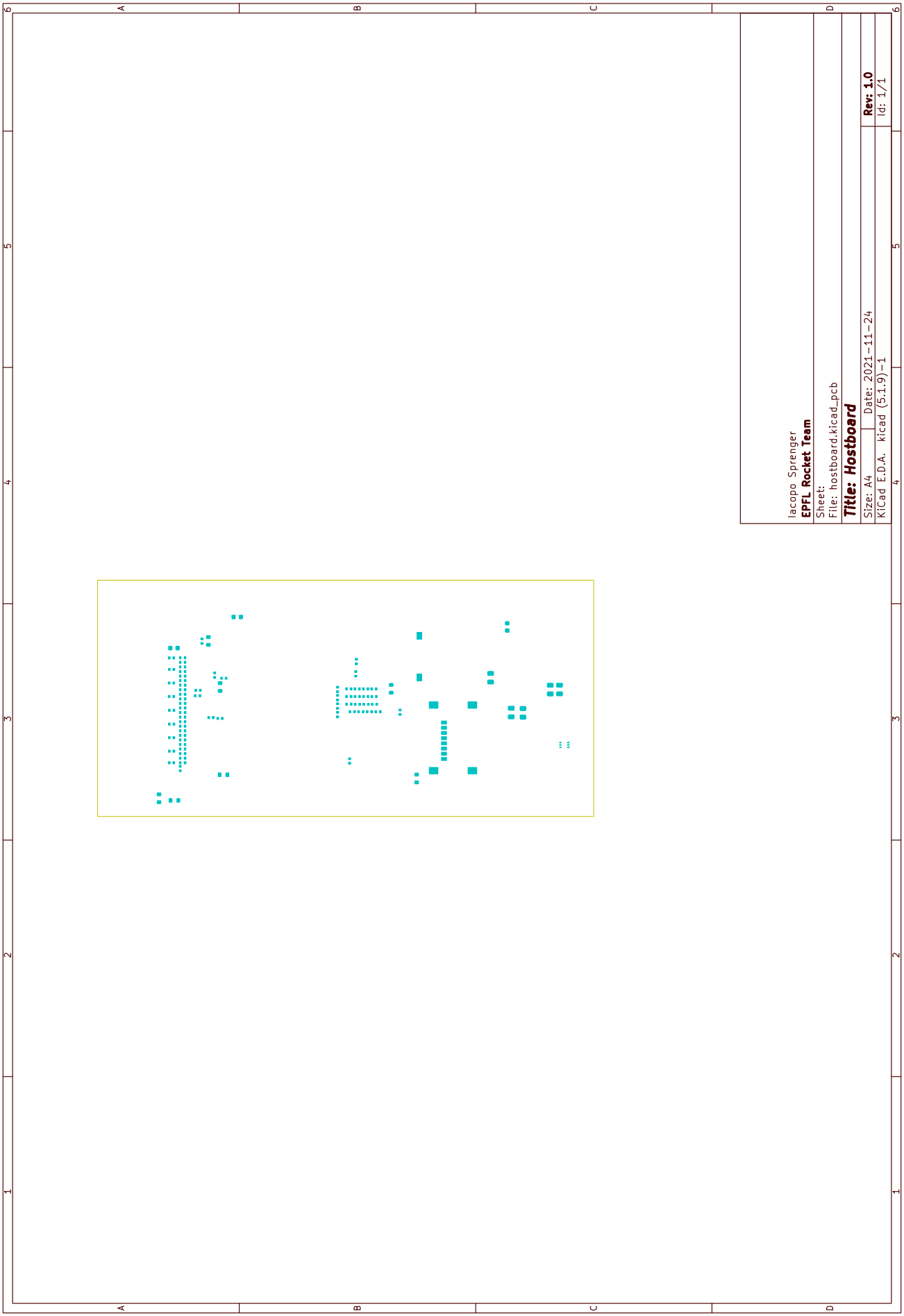


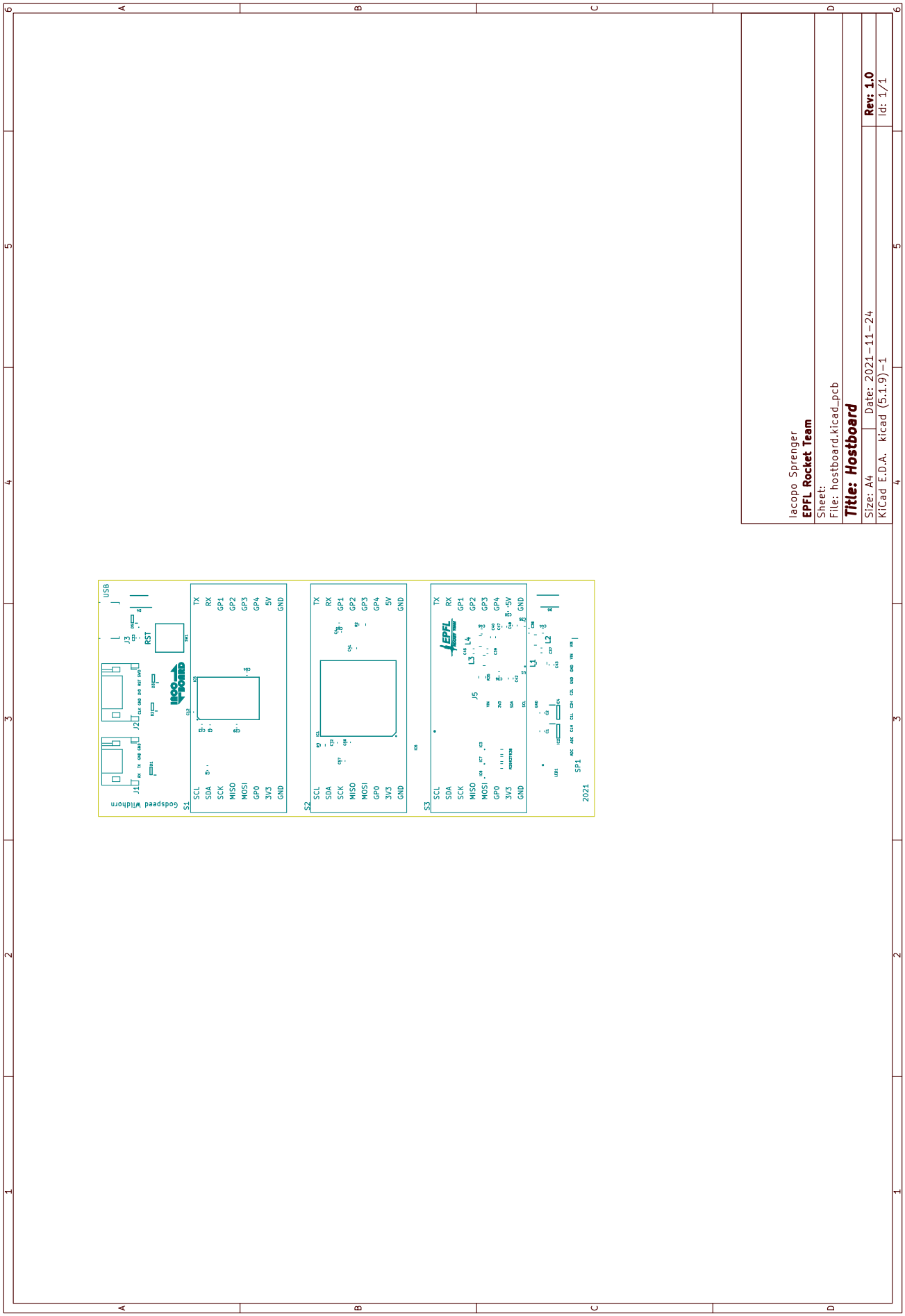
Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4 | Date: 2021-11-24 | **Rev: 1.0**
KiCad E.D.A. kicad (5.1.9)-1 | Id: 1/1

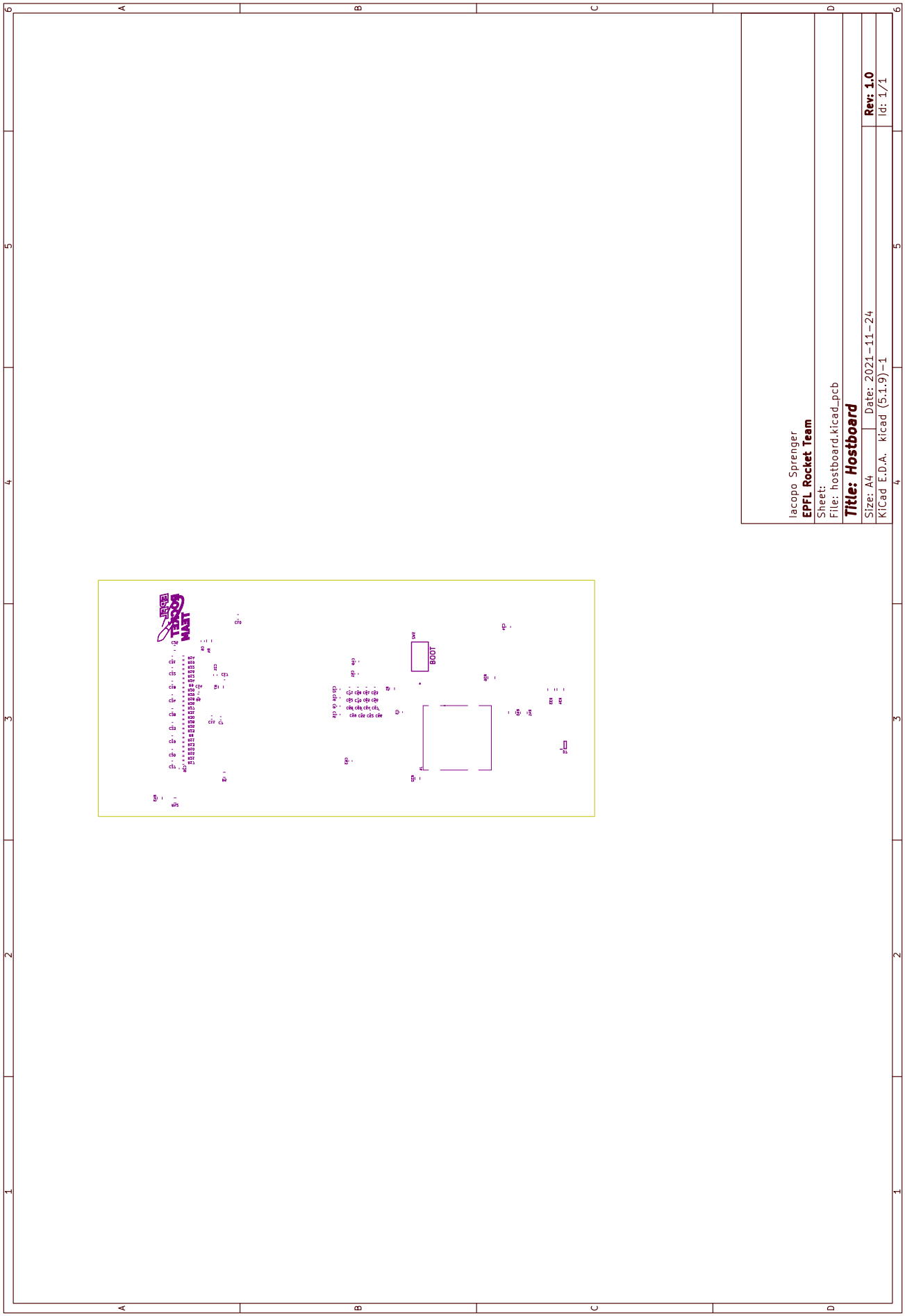


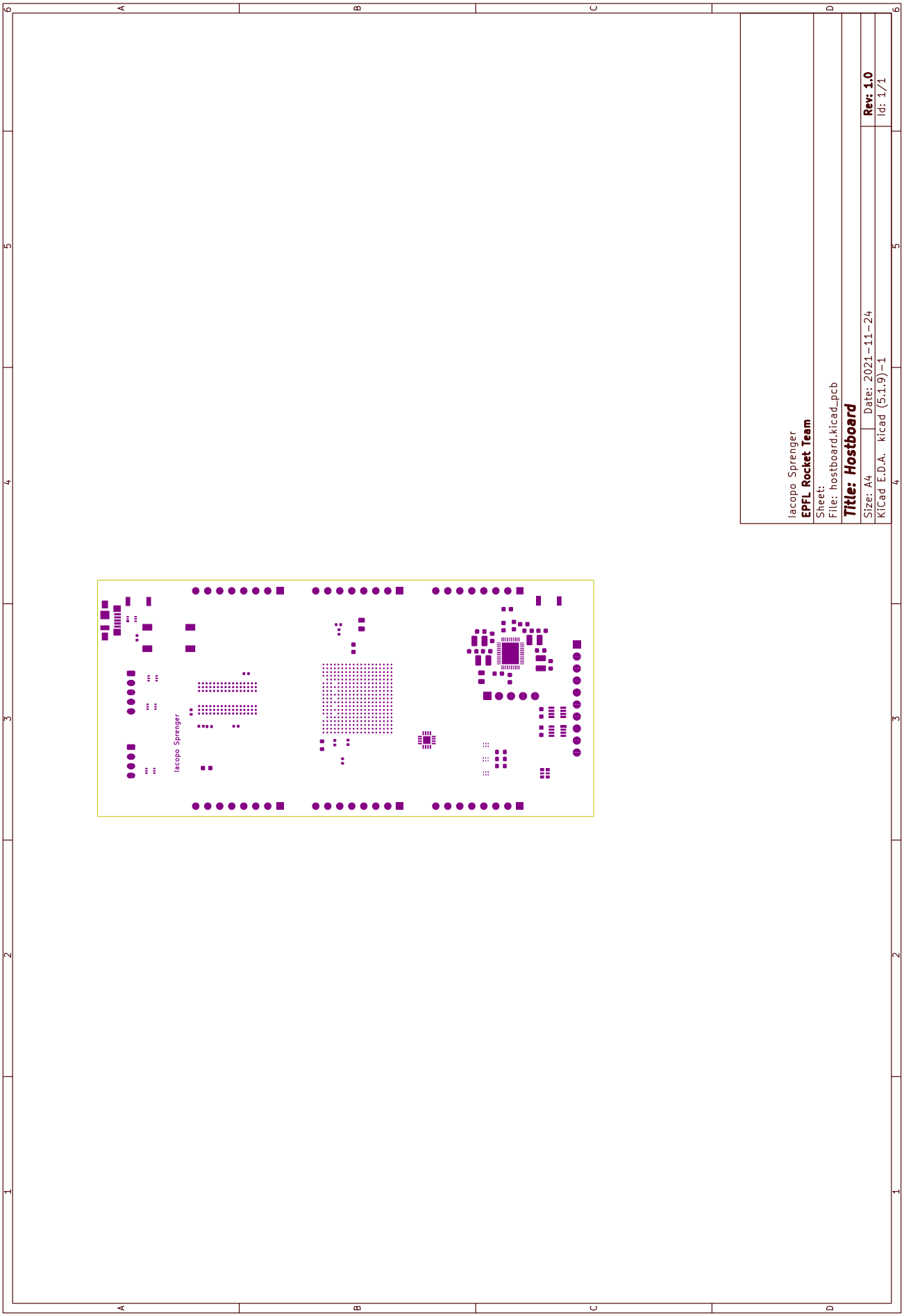
Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4 | Date: 2021-11-24 | Rev: 1.0
KiCad E.D.A. kicad (5.1.9)-1 | Id: 1/1

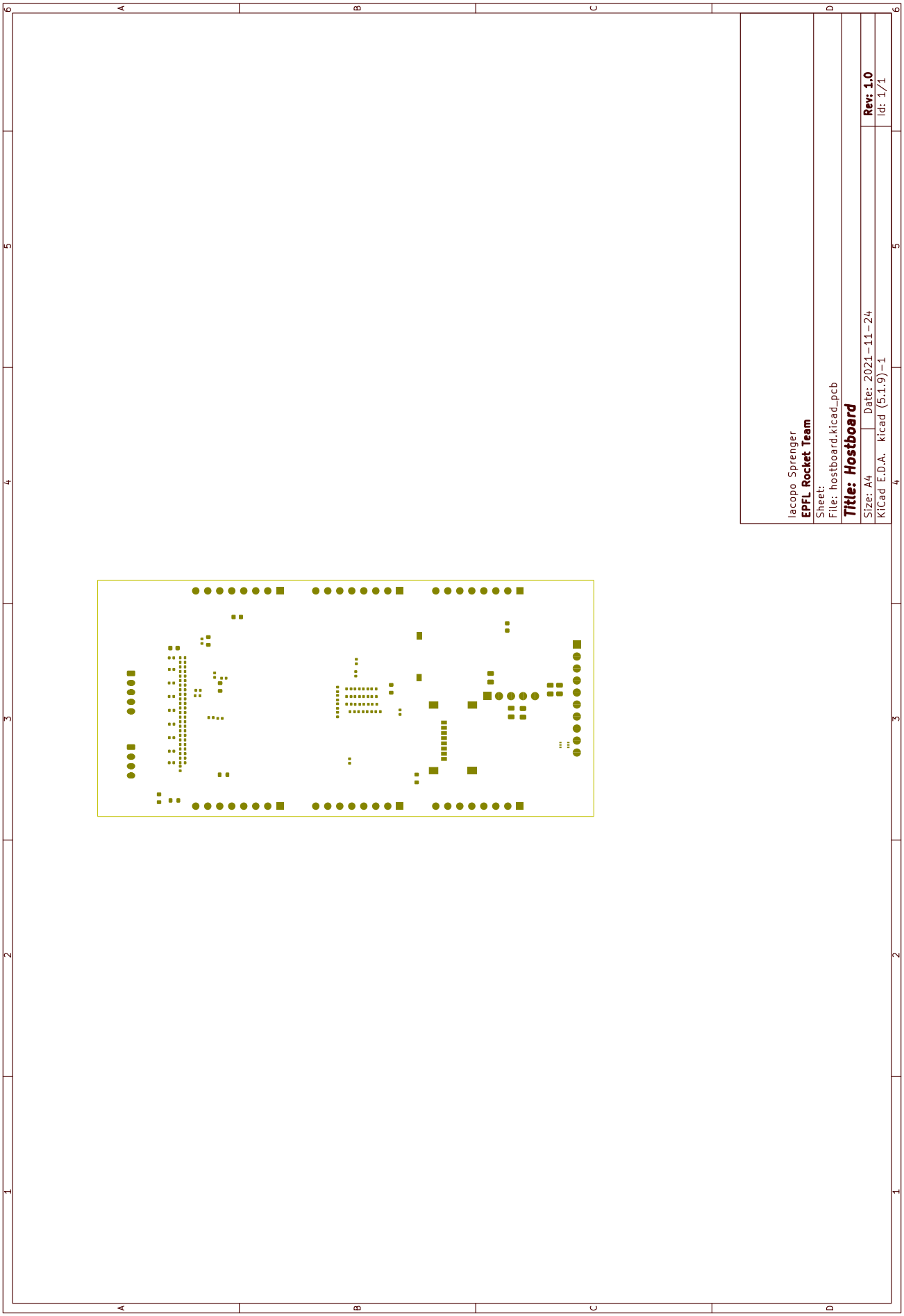

Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 1/1

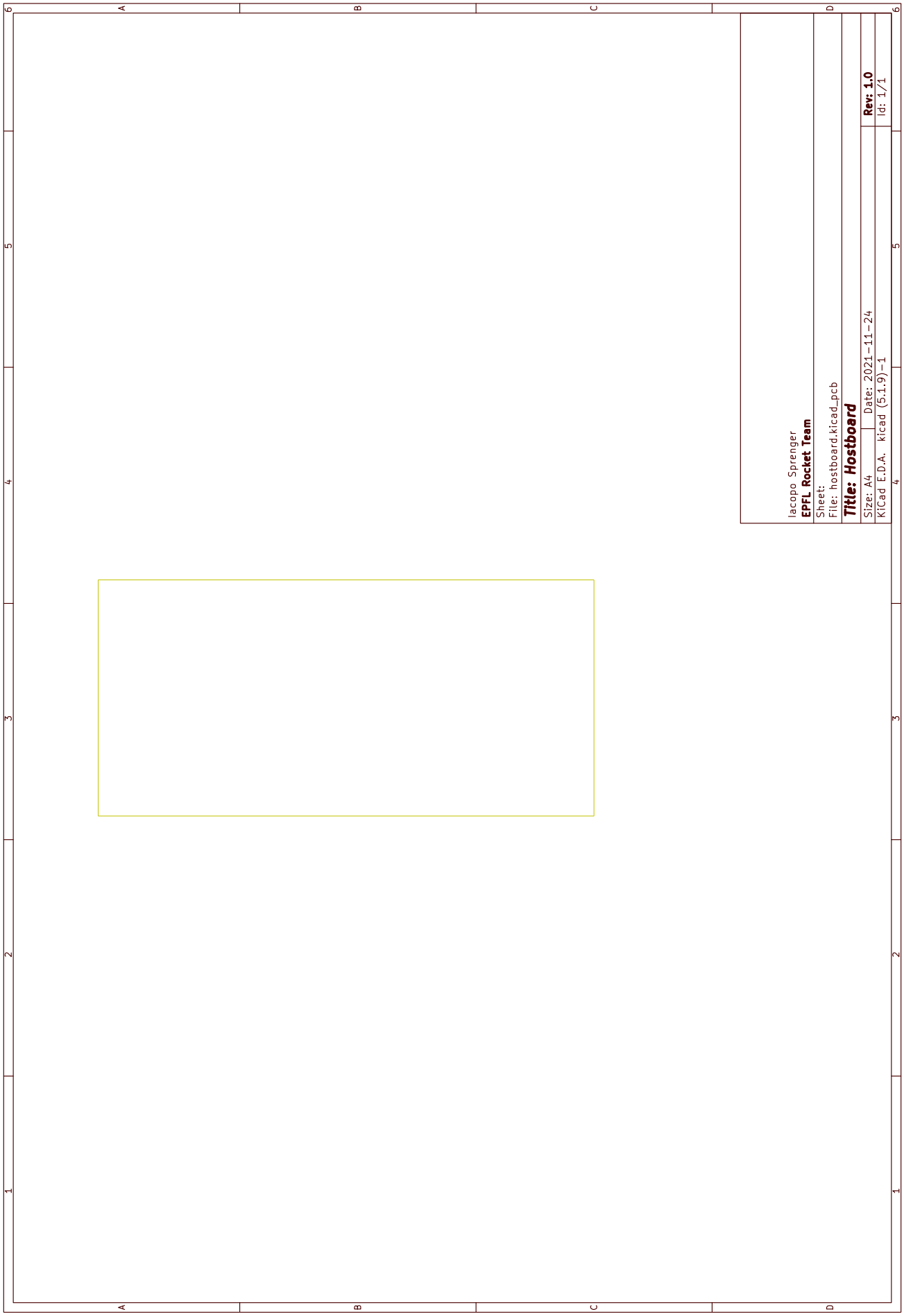

Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 1/1


Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 1/1
Goodspeed Wildhorn
USB
RST
S1
S2
S3
SCL
SDA
SCK
MISO
MOSI
GP0
3V3
GND
TX
RX
GP1
GP2
GP3
GP4
5V
-5V
SP1
2021


Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 1/1
BOOT


Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 1/1
Iacopo Sprenger


Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 1/1


Iacopo Sprenger
EPFL Rocket Team
Sheet:
File: hostboard.kicad_pcb
Title: Hostboard
Size: A4
Date: 2021-11-24
KiCad E.D.A. kicad (5.1.9)-1
Rev: 1.0
Id: 1/1