STM32 Based
Rocket Igniter Board

Table of Contents

Preface

Disclaimer

This document is written purely for guidance purposes. It is not written or verified by a professional. Everything shared in this document are of my own experience and observations. Therefore, the information might be incorrect or inaccurate. Reader must fully acknowledge this before moving forward.

About this document

The purpose of this document is to share the experience and observations that were gathered during the design of this board with public, in order to ease hardware development for future generations. There is no exact target audience; students, hobbyists, even professionals may benefit from this. However, this document assumes the reader has basic knowledge of PCBs and electronics.

Release date: September, 2018

Feedback

Any feedback related to the document or the design is welcome. Feel free to ask questions, suggest changes or improvement. Please direct your inquiries to *tozlu16@itu.edu.tr*.

Further reading

Most of the knowledge in this document is derived from manuals and application notes from various companies. Below are the essential manuals and datasheets.

- https://www.st.com/resource/en/application_note/cd00164185.pdf
- https://www.st.com/resource/en/datasheet/cd00161566.pdf
- https://www.st.com/resource/en/reference_manual/CD00171190.pdf
- https://www.alldatasheet.com/datasheet-pdf/pdf/880802/TEC/MS5611-01BA03.html
- http://www.ti.com/lit/ds/symlink/lm2596.pdf

PART A

Introduction to the board

A1) Purpose of this board

The Igniter Board was designed and developed for the Rocket Technologies Group of Istanbul Technical University. The board was designed to be included in the avionic bay of a rocket, and it eventually did fit inside the bay perfectly.

Its goal was to control and enable the deployment of the rocket's parachute. The deployment process begins by estimating the altitude using the on-board embedded barometer. Upon reaching 20 meters below the apogee point, the board ignites two ignition wires, which in turn ignite an amount of black powder which ejects the surrounding body of the rocket to allow the parachute to be deployed.

Aside from ignition, it features a UART bridge, which is used to transmit altitude data, to another board in the bay.

A2) Design process

The board consists of the following segments; MCU, barometer, ignition circuits, regulators and miscellaneous parts. All the drawings were done in Autodesk® Eagle 8.6.0. You might need 8.6.0 or later versions to open the board files.

The proper way of doing the schematic designs is to split the board up into modules like I did above, then work on modules, rather than working on the whole schematic all at once. For instance, I first drew the schematics of the MCU and did not move on to another part until I was done with it. After completing the MCU, I moved on to the regulators and so on. Modularizing the board makes it easier to maintain and manage the parts.

For this board, schematic design was basically reading and following the manuals. For this job, you will stick to the manuals that I shared in the preface section. Those manuals contain almost all the information you would need. The board layout, however, was somewhat trickier.

Due to budget limitations, the board had to be two-layered. Because of this, both layers had to contain a power plane and signal tracks. I decided to fill the top plane with GND and bottom plane with VDD. With this design, I ended up with a bunch of isolated sub-planes, as the signal tracks interrupt the power plane on that layer. Therefore, I routed lots of new, wide tracks that connect all the isolated planes with the main power plane. This is the problem with two-layered boards; dedicated layers for power planes are unavailable, instead power planes that are frequently interrupted by signal tracks must suffice. ST suggests to use at least a four-layered board to avoid this, though this two-layered board worked just fine.

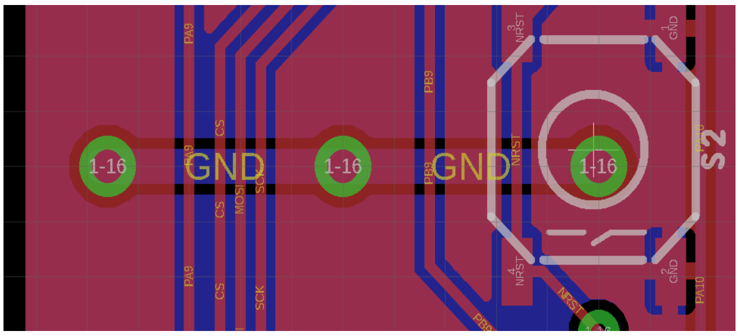

Figure 1: 0.8mm GND track routed to connect isolated planes

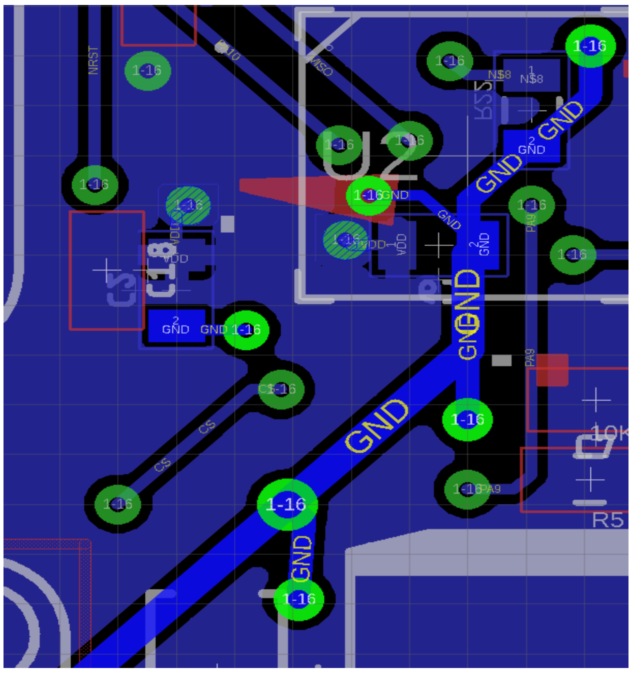

Figure 2: 0.7mm GND track routed to connect isolated planes

Layout of the parts was also challenging. Since the board was going to be housed in a rocket, I had very limited space. I had to keep the board as small as possible while also delivering the requirements. Finding the most optimized layout took several trials and errors, though it still is not perfect. My way of doing this was laying out and routing the MCU parts first, then moving on to ignition circuit and other miscellaneous parts. Throughout this process, I occasionally found myself having routing issues, this forced me to place parts on the bottom layer, as well as redesigning entire segments of the board. Ultimately, I managed to develop an optimal layout.

All the resistors and capacitors are of 0805 package. The only reason for this is that 0603 package is difficult to solder by hand, whereas 0805 is easier and also small enough for this board.

A3) Board properties

Board size	69.50mm x 85.00mm
Board thickness	0.8mm, two-layered.
Material	FR-4
Soldermask	Green
Track thickness	1 oz (35um)

A4) Post-manufacturing

Following images represent the three main phases of this board's production in order; computer drawing, manufactured board, soldered board.

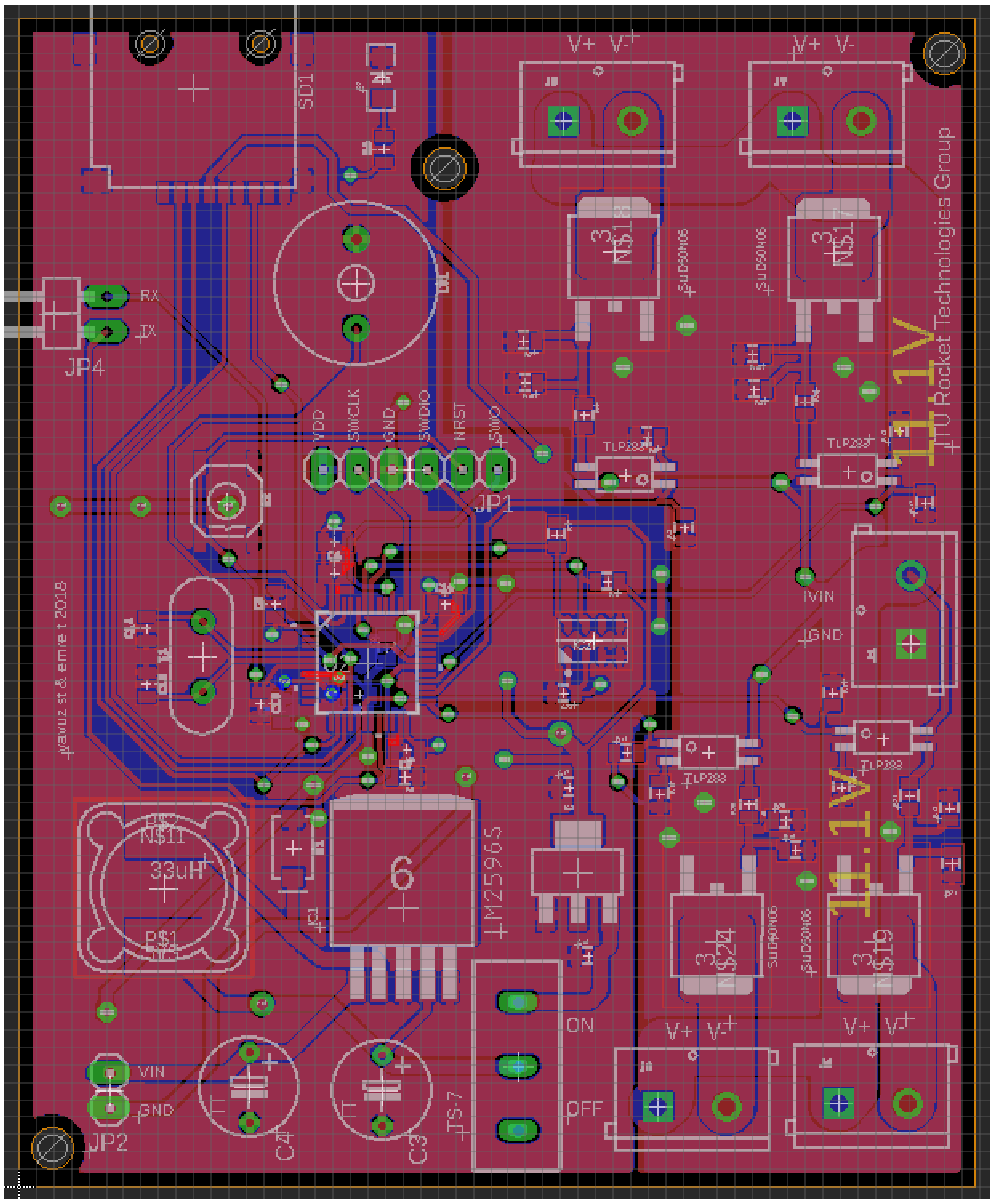

Figure 3: Computer drawing. Eagle 8.6.0

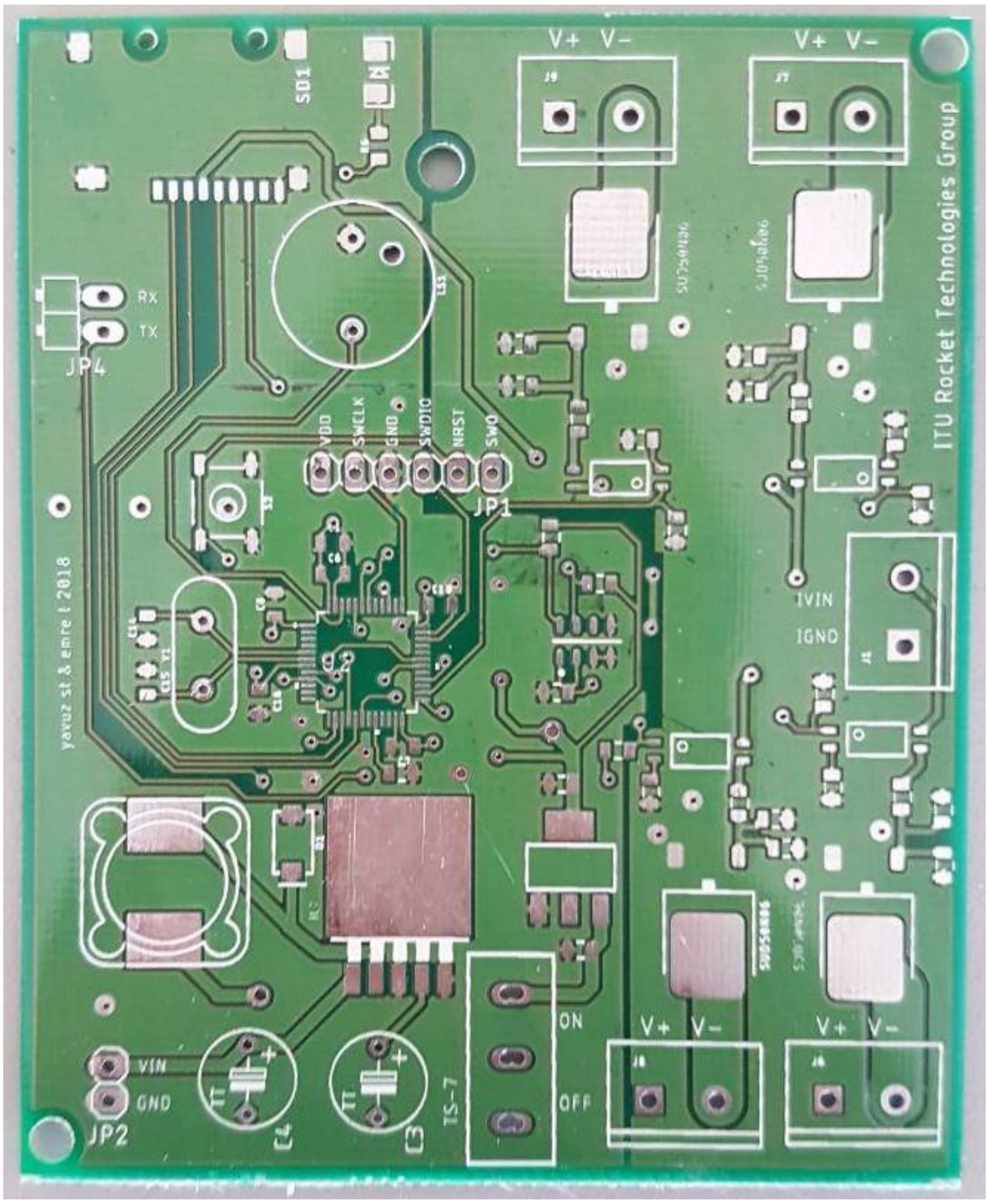

Figure 4: Manufactured board (top)

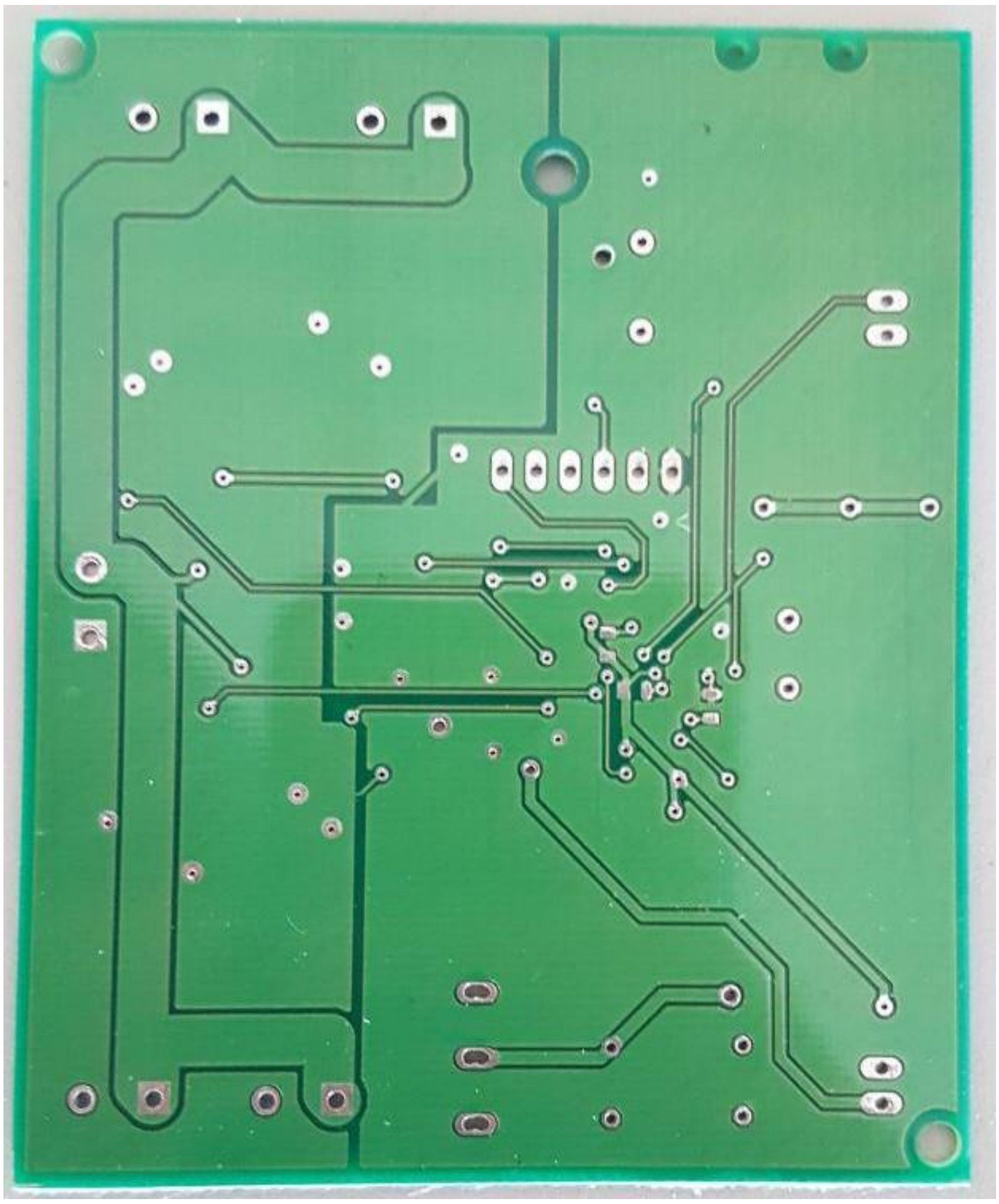
Figure 5: Manufactured board (bottom)

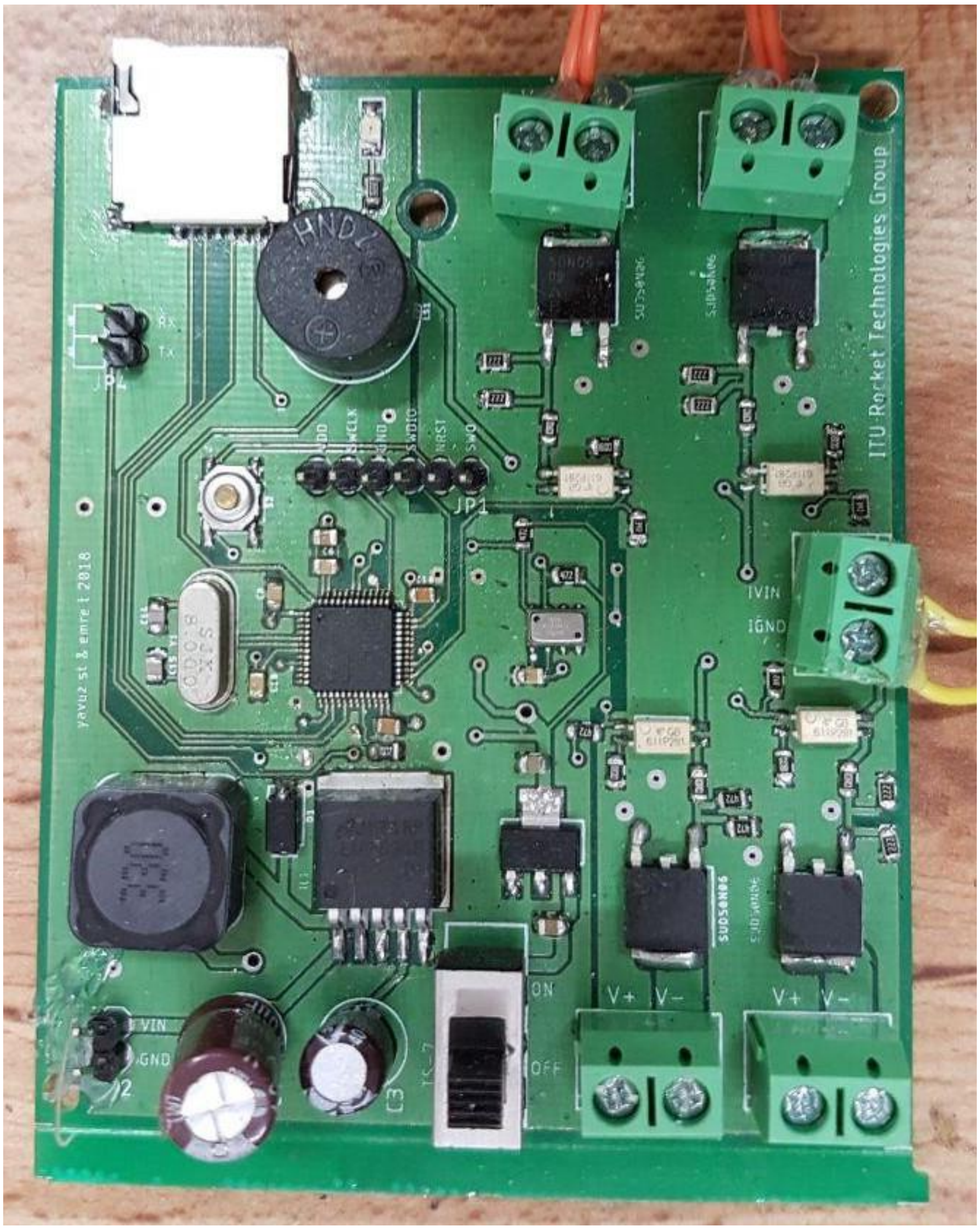

Figure 6: Soldered board (top)

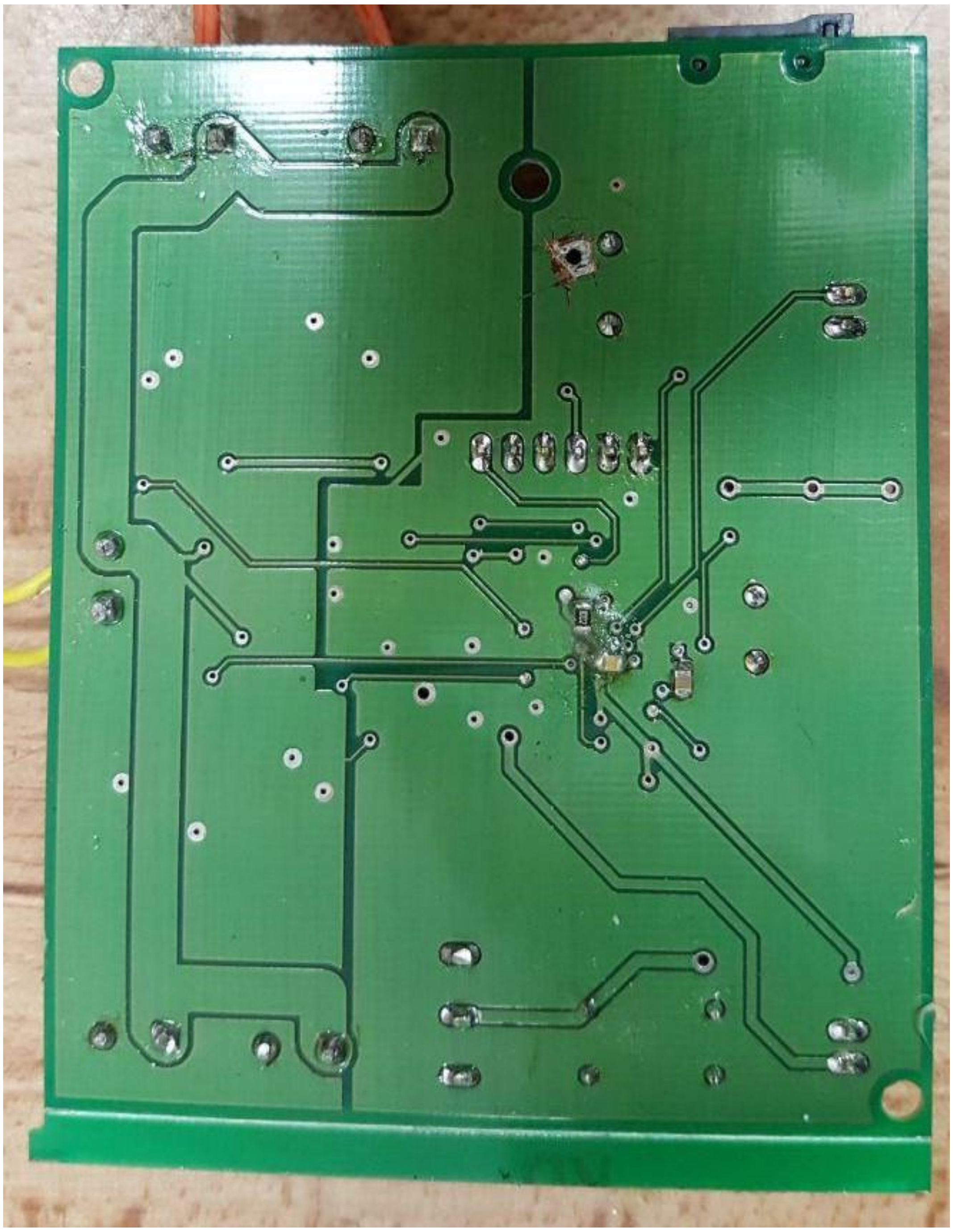

Figure 7: Soldered board (bottom)

This board is hand-soldered. There is visible flux residue left, especially on the bottom. I used Isopropyl alcohol to clean the top layer, perhaps due to the ingredients of my IPA, it left visible residue all over the board. An ESD safe brush was used to do the cleaning.

Reader may have noticed the hole with a scratched surrounding. That hole is a manufacturing flaw, it does not exist in the design. It is essentially a via that forms a conductive path between the top and the bottom layer, but around that area of the board, top layer is GND and bottom layer is VDD. This means that hole creates a short, this is why I had to scratch its surrounding copper to isolate it. I verified the short with a multimeter, checked all 6 boards that I received from the PCB house, all of them were suffering from the same flaw. This is a rare scenario where the PCB house manufactured defected boards.

PART B
Embedding the STM32 MCU

B1) MCU Selection – Why STM32F103?

The main concerns during the MCU selection phase were; price, availability, software support, processing power, memory and physical size. At this point, options were quite clear; ATmega or STM32 – most common and easy to use MCUs in the market. Using ATmega was the easiest and the most reliable choice, however that would contradict with the idea of this project – which is to learn and experience new concepts, technologies. Today, developing projects with ATmega a.k.a. Arduino MCUs are incredibly simple, so much so that it merely gives people the illusion of actually developing something, which rarely is the case. This is mainly why the STM32 was chosen for this project.

The processing power that came with the STM32 was certainly not needed, neither was the memory, but the learning difference was immense.

Tools and libraries provided by ST incredibly eased up the software development phase, which placed STM32 above the other MCUs in the market.

B2) External Circuitry

Minimum external parts to get the STM32 MCU working are; decoupling capacitors, a crystal, reset circuitry, boot pin configuration and SWD output. Note that, this is what ST suggests, indicating the MCU *may* work without some of those components. Also, an external crystal is not mandatory since the MCU offers an internal clock source. The drawbacks are that internal oscillator is less reliable, has more jittering and also less accurate than an external crystal.

Detailed information on decoupling caps can be found in the manual. To sum it up, for STM32F103C8T6 MCU, you would need; 5x 100nF, 1x 10uF, 1x 1uF decoupling capacitors to filter the power input. The MCU has 3 main power pairs, 1 analog power pair and 1 external battery input which should be connected to the VDD rail if not used.

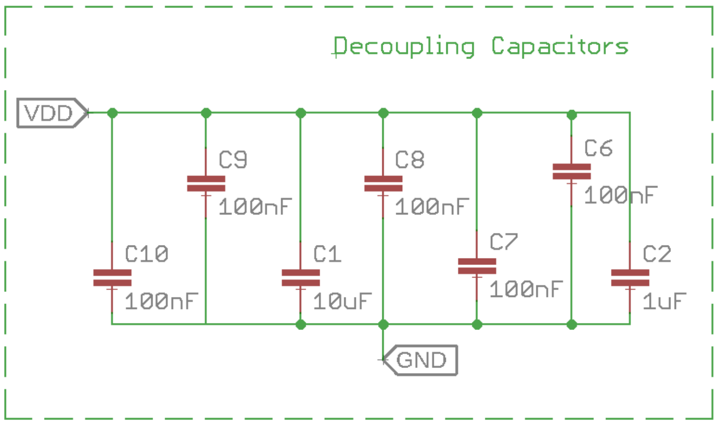

Figure 8: Decoupling capacitors for STM32F103C8T6

The memory block used to boot the MCU is determined by configuration of boot pins. Main flash memory is sufficient for most purposes, this project is no exception. Correct boot pin configuration for main flash memory is achieved when both boot pins are in low state, in other words, pulled to low. The manual suggests to use 10kOhm resistors as pull-downs.

In order to give the user the option to alter the boot mode between different memory blocks, designer can attach jumpers to the boot pins. Since this would take extra space on the board, I decided to strictly pull the pins low. Detailed information on boot pins can be found on the manual.

Igniter board houses an external crystal for the STM32 MCU. There is no guidance as to what the proper frequency would be, therefore an 8MHz crystal is used. The important thing here is to match the crystal's frequency value with the value in your code during software development. Any value from 15pF to 22pF for load capacitors would be fine.

Reset circuitry is also described in the manual. Implementation for this board is shown in the following image.

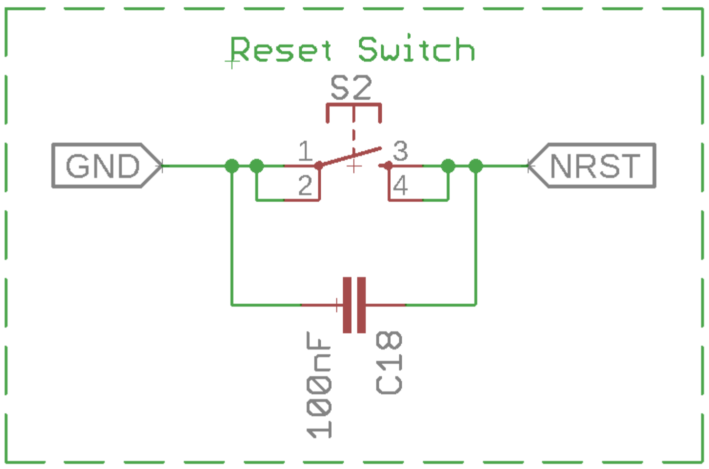

Figure 9: Reset Circuitry. NRST is tied to NRST pin of the MCU

As an interface between the MCU and the computer, Serial Wire Debug is used. Normally, SWD requires 3 pins; SWCLK, SWDIO, RST, aside from a common ground. On the Igniter Board, there are 2 additional pins; VDD and SWO.

VDD pin is added so that the board could be powered without having to use the main power supply of the board. This way I could power, program, debug the MCU without having to provide an external power source, which was really helpful during the programming and the testing phase.

SWO pin is an extra debugging feature offered by the SWD interface. Although it exists on the board, SWO pin was never used.

B3) Board Design

There are lots of guides on placement of decoupling capacitors. Following image is taken from ST's manual and suggests the most ideal placement, which I attempted to follow.

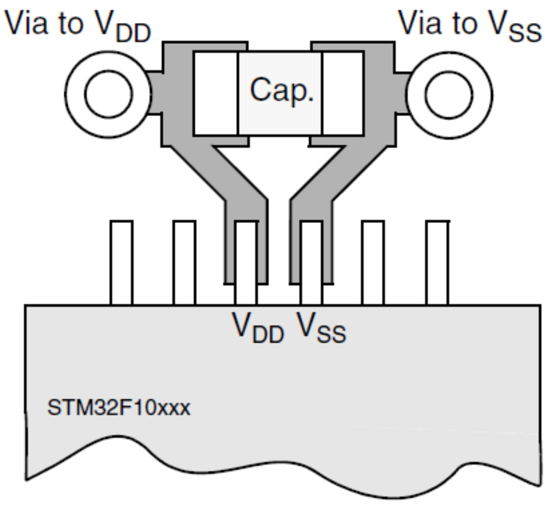

Figure 10: Ideal capacitor placement

To achieve this placement for each and every capacitor on a two-layer board is simply impossible, it is not required either. The idea is to keep the tracks as short and wide as possible, maintaining a low impedance. Also, both layers can be used to place those capacitors. On the Igniter Board, all the tracks entering the MCU are 0.25mm wide.

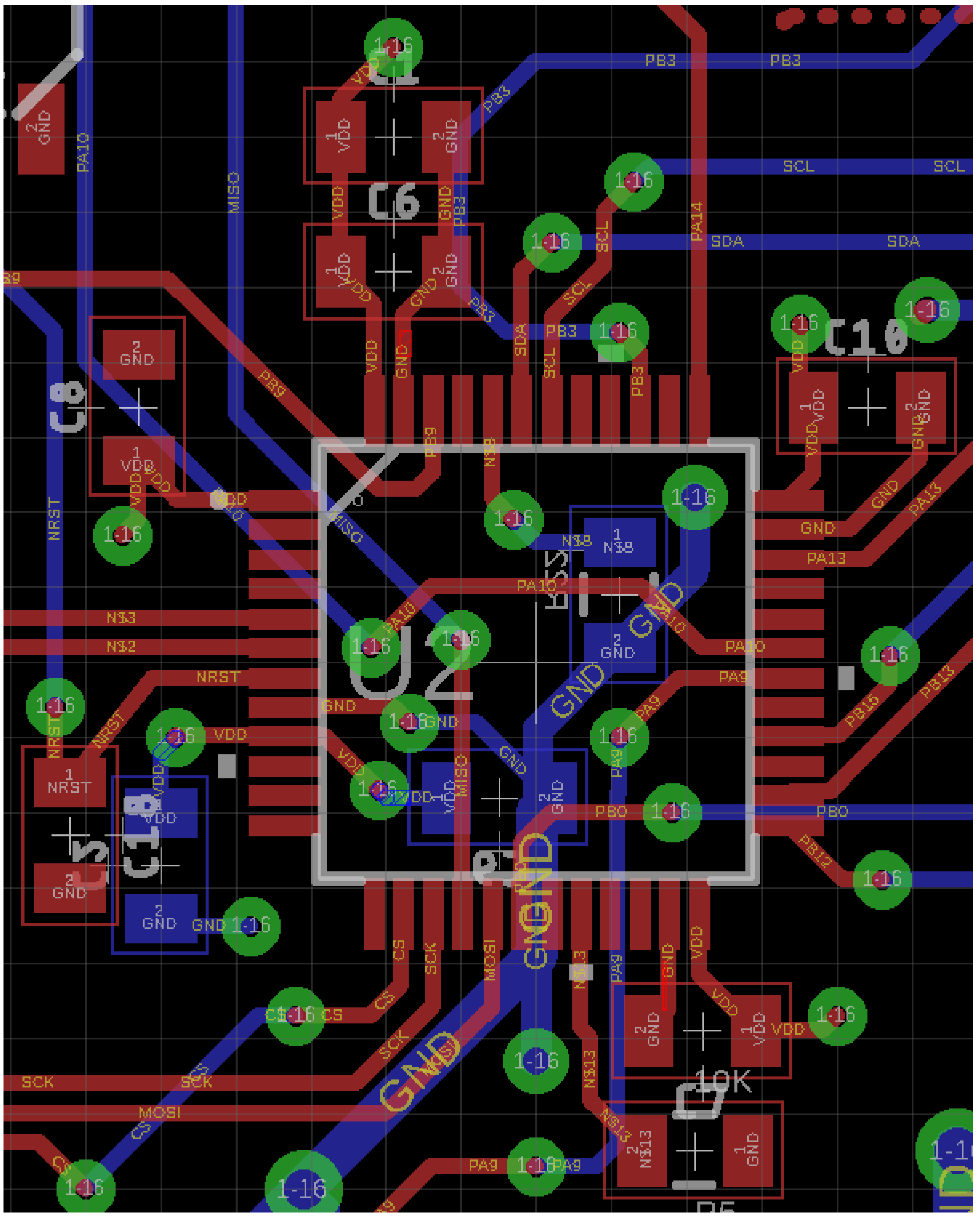
Figure 11: Circuitry around the STM32 MCU. Capacitors and resistors.

For the crystal, I kept it as close as possible to the MCU, while keeping the tracks and the load capacitors symmetric.

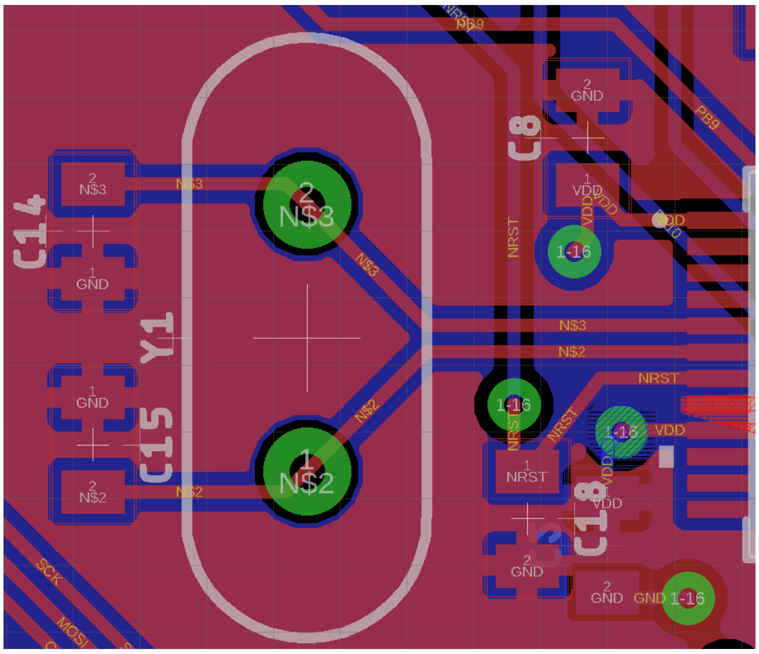

Figure 12: Symmetric tracks and capacitors for the crystal

It can be stated that the STM32 MCU was embedded properly and successfully on this board. No stability issues, timer inaccuracy or any other unexpected behavior were observed.

PART C

Ignition Circuit

C1) About the circuit

Ignition circuit is the electronic system that controls the ejection of the rocket. Due to the nature of its task, it is crucial that the system is fail-safe. Thus, it is equipped with an array of components to attain the level of safety required. The last thing needed on the field is a rocket ejecting at the wrong time, in the wrong place i.e. on the ground. It must be noted that this circuit has an enormous flaw that will be addressed soon.

Following is an image of the schematic of the circuit.

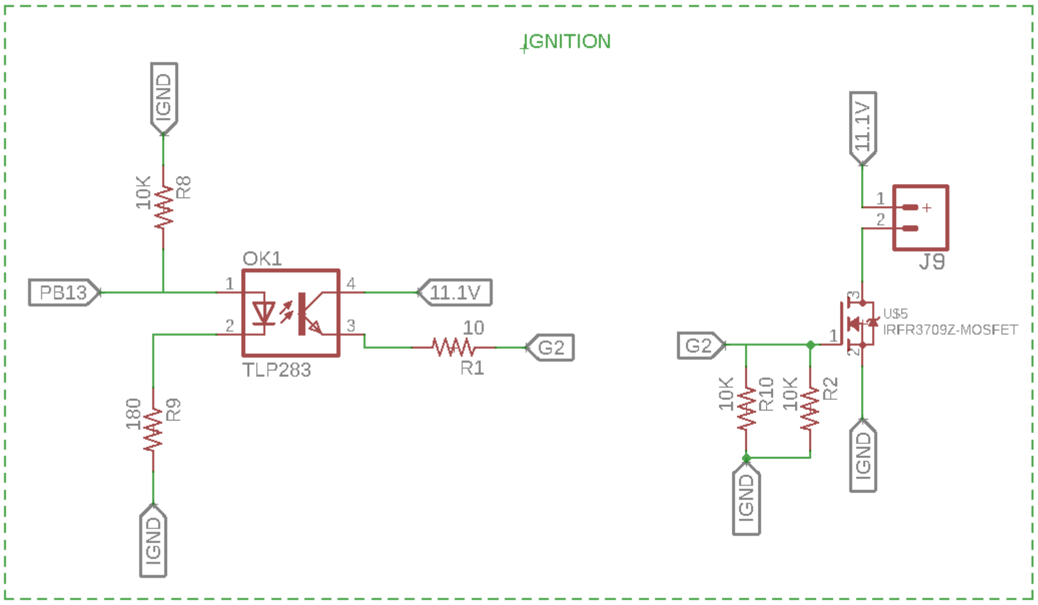

Figure 13: Ignition Circuit schematic

The circuit consists of two main stages: The opto-coupler and the N-MOSFET. Input of the opto-coupler's LED is connected to the MCU's pin PB13 in this case. The chain of events is triggered as the MCU drives the associated pin high, which activates the opto-coupler and ties 11.1V to the MOSFET's gate. Applying voltage to the MOSFET's gate allows current to flow from the drain to the source. Finally, this current flows through the terminal placed in between the drain and the source. Ignition wire that is attached to the terminal is ignited by this very current.

The circuit is powered directly from a 11.1V Li-Po battery, and has a separate ground(IGND). Cables for power input are attached to a terminal, since common pin headers cannot handle the power delivered by the battery.

During the ignition of the wire, current draw can go up to 5A. Thus, MOSFET used in this circuit must be able to handle currents up to 5A, voltages up to 12V. The MOSFET used in this board is SUD50N06 from Vishay.

C2) Safety Measures

The ignition circuit is required to be electrically isolated from the rest of the board, as noise might cause the opto-coupler or the MOSFET to get activated or prevent them from getting activated, despite this is unlikely. Any possible kind of electrical interference between the circuit and the rest of the board is unwanted. This requirement led to the use of an opto-coupler. This way, the ignition circuit and the rest of the board do not have to share a common ground.

The input of the opto-coupler and the gate of the MOSFET are pulled to low. These pins cannot be left floating as this could lead to unintended activation of the components. Those pull-down resistors cannot be omitted. Notice, not one, but two pull-down resistors are attached to the gate of the MOSFET for further safety.

C3) The Flaw

The design of this circuit contradicts with one of the main requirements discussed above, which is electrical isolation between the circuit and the rest of the board. Careful eyes may have noticed this just by looking at the schematic given at the beginning of this part.

The output of the diode inside the opto-coupler is tied to IGND through a resistor, whereas it should have been tied to GND. The reason can be obvious, the MCU's 3.3V logic level is defined with respect to GND, not IGND. Then, only way to make this circuit work is to tie GND and IGND together, which makes the entire circuit pointless and redundant. Unfortunately, there is no alternative solution, and that is the work around I followed with my circuit.

Also, notice the pull-down resistor attached to the input of the opto-coupler pulls the pin to IGND, not GND –which is another flaw.

Nevertheless, this circuit did work with no problems, and successfully ejected the rocket's body at the desired altitude.

Following is the corrected version of the circuit.

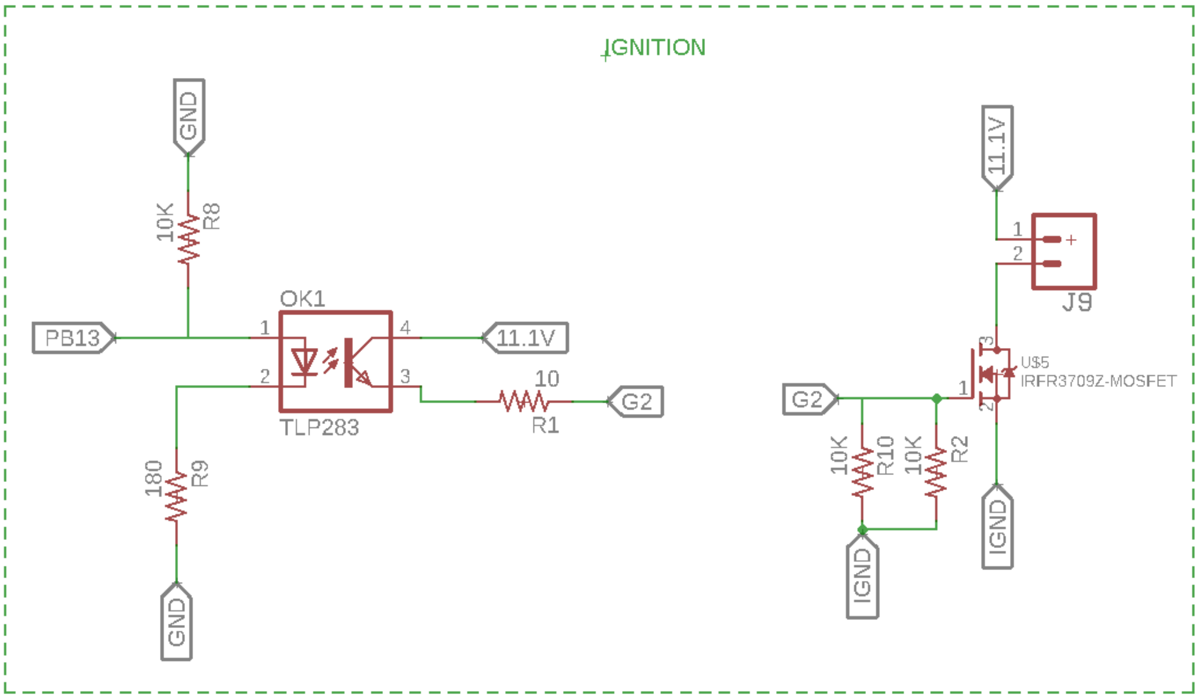

Figure 14: Corrected circuit.

PART D

POWER SUPPLY

D1) Step Down Regulator

The input voltage for the circuit is planned to be 7.4V from a Li-Po battery. Target voltage is 3.3V. This leaves us two options, a linear regulator or a step down regulator. Regulating 7.4V to 3.3V with a bare linear regulator is very inefficient and slightly risky. Instead, I embedded a step down regulator to drop the 7.4V to 5V and use a linear regulator thereafter.

The step down regulator is based on the LM2596 from TI. The circuitry around it consists of an inductor, two aluminum capacitors and a schottky diode. This is also the suggested circuit in the datasheet.

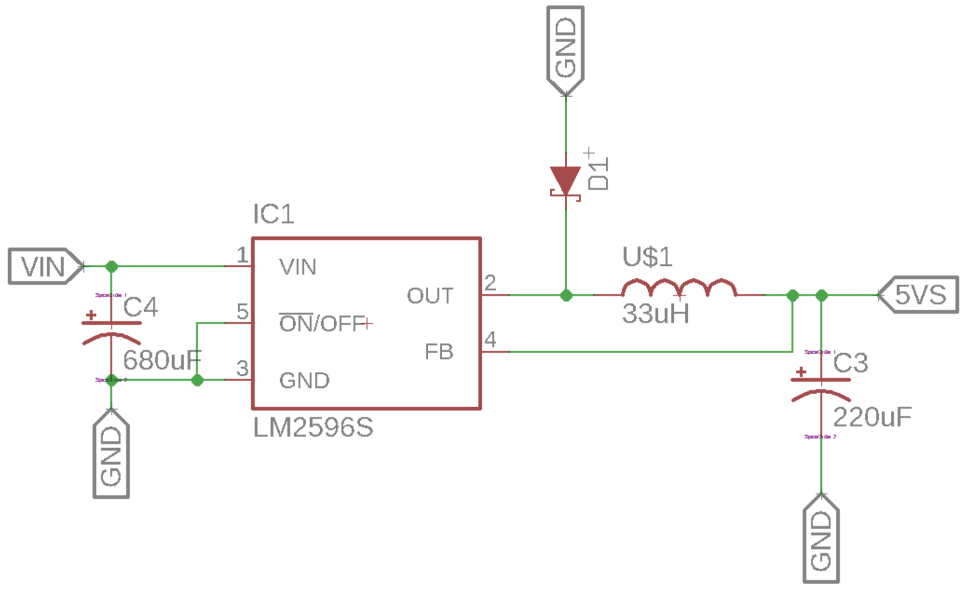

Figure 15: Step down regulator circuit, based on LM2596

There is no significant reason to use aluminum capacitors for this circuit, however, they are cheap and easily available.

As for the layout, TI pointed out some information in the datasheet, the rule of thumb is to keep the traces as wide as possible and the external circuitry close to the LM2596 IC.

D2) Linear Regulator and Power Switch

To achieve the target voltage, a linear regulator is used to drop 5V to 3.3V. The linear regulator used in this board is AMS1117 from ams-semitech.

There is also a slide switch added to the board. This switch controls the whether the MCU side of the circuit is powered or not, it has nothing to do with the ignition circuitry.

There is a minor, yet silly design flaw about the switch’s placement. The current to the circuit enters the switch after it goes through the step down regulator. This means, the regulator is powered on and is working no matter what the state of the switch might be. This is why the switch should have been placed right after the power jack to function as it was intended.

It can be seen that the holes for the slide switch are not circular, as they usually are. They are rather oblong holes. This is a very important detail about the package of the switch. I drew the board symbol of the switch myself, mainly because there were no matching libraries. To draw oblong holes in Eagle, one has to use the “milling layer” in Eagle. The oblong hole should be drawn on the milling layer, and must be included in the gerber files that will be sent to the PCB house.

PART E
MISCALLENOUS PARTS

E1) MS5611 Barometer

The board houses an MS5611 digital barometer from TE Connectivity. Its only task is to measure the air pressure, then convert it to an altitude value.

Schematic of the barometer includes a mere capacitor, and 2 I2C pull-up resistors, as is suggested by its datasheet.

The primal challenge with the barometer is, in fact, in the software side. That is, implementing C code for the communication between the STM32 MCU and the sensor. Enough information on software development with MS5611 is given on the datasheet. However, to realize a properly working interface takes some effort. A working library based on the HAL drivers from ST is shared in the repository.

E2) SD Card Socket and Buzzer

One of the requirements of the igniter board was to save flight data to a micro-SD card attached to the board. However, despite many attempts, I never managed to establish a fully working interface between the MCU and the micro-SD card. Therefore, it is unknown whether the socket circuitry works at all. In spite of this, since it imitates working designs, I would expect it to function properly if a working interface is supplied.

The buzzer is a very specific component required due to the board’s purpose being related to the rocketry. Basically, when the rocket successfully lands on the ground, it needs to be located and retrieved by the team. Buzzer is added so that the sound it makes can be helpful to locate the parts.

There is a personal remark I would like to add about the buzzer’s circuitry. The buzzer is controlled and powered directly by the STM32 MCU. This is not the best way to do this, as the voltage supplied by the MCU is very low, 3.3V. Note that, the volume of the sound the buzzer makes is directly proportional to the voltage applied to its pins. Thus, a higher voltage is preferable in this case. To achieve this, a simple BJT can be used, with its base pin connected to the MCU.

Final Words and Remarks

The development of this board involves months of research, a manufactured but unsuccessful board design, and lots of trials and errors. The very beginning of this process goes back to February of the year 2018. The board was successfully manufactured around July, 2018. This document is written during September, 2018.

It can be concluded that the board is successful. It was programmed, debugged, powered on and off many times, and has worked in a stable manner. It successfully gathered pressure and altitude data, ejected the rocket properly, as was intended. Nevertheless, reader should pay attention to the flaws mentioned throughout this document.

Most of the omitted information about the design can be deduced by the reader by carefully examining the board files.

Last but not least, the software that was used on this board for parachute deployment is shared in the repository.

About the author

I am Yavuz Selim Tozlu, at the time of writing this paper, I am studying Electronics and Communication Engineering at Istanbul Technical University. E-mails in any regard can be sent to tozlu16@itu.edu.tr.