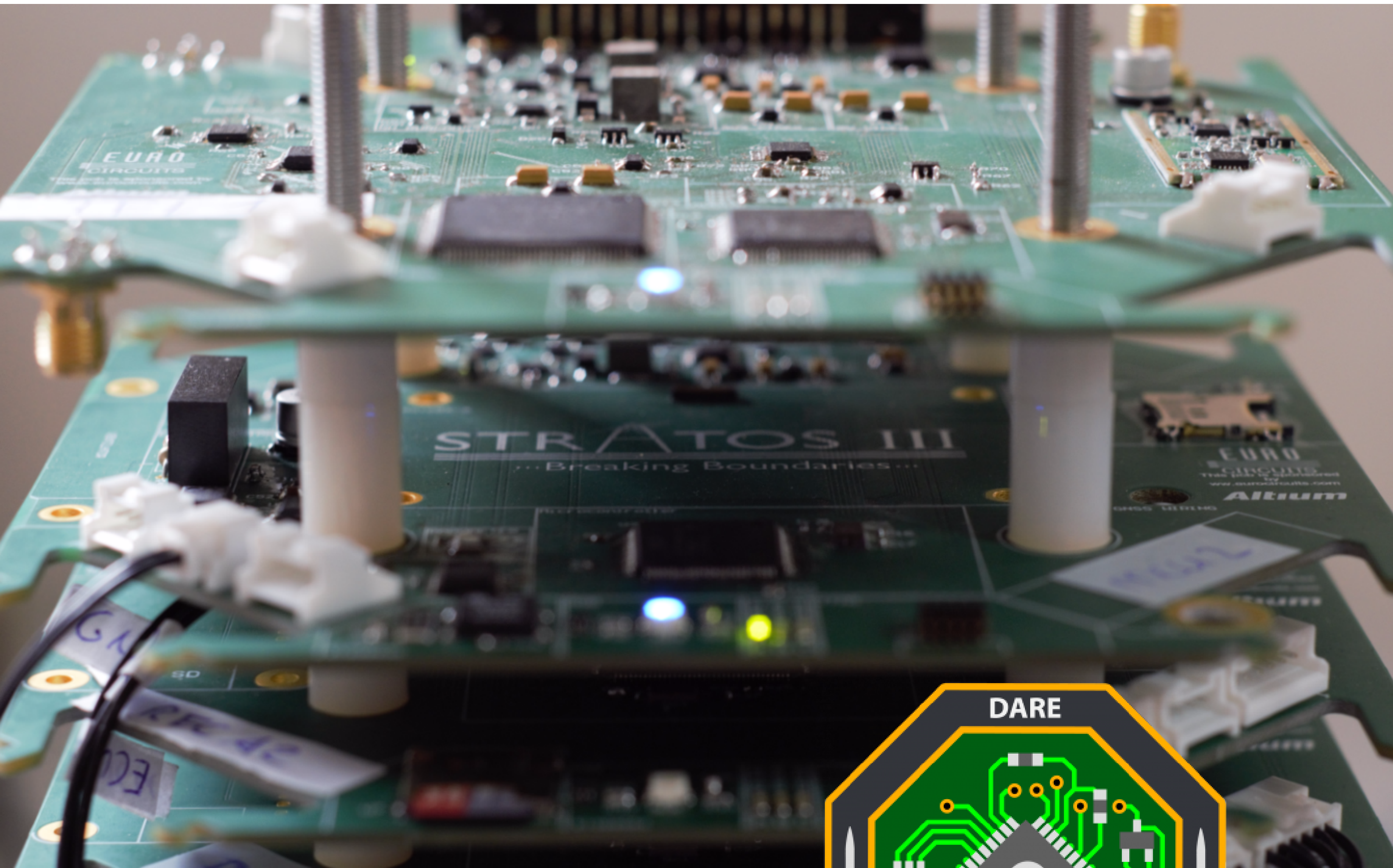
Electronics Manual

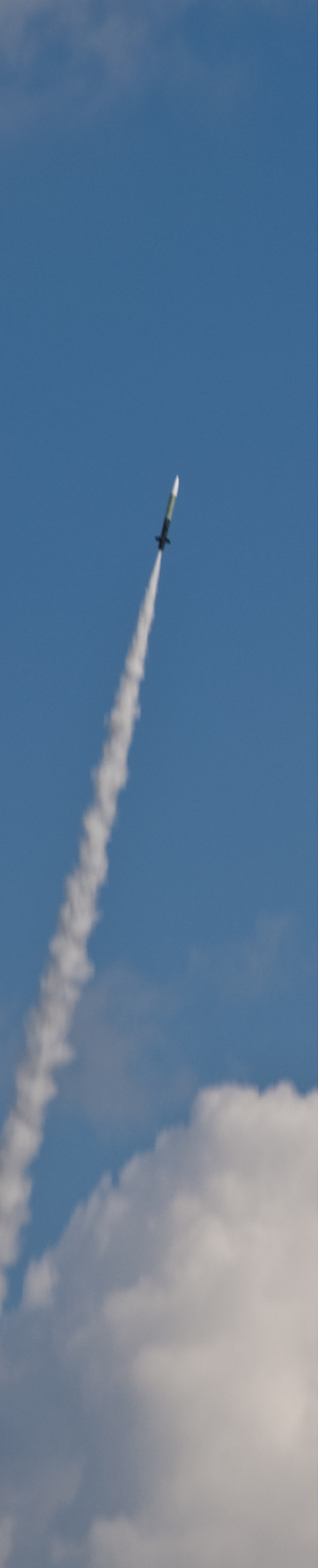
Small Rocket Project



Authors:
Manfred Josefsson
Maurits van Heijningen
Jun Feng

Version:
V 3.5.0

February 16, 2019



Contents

Table 1: SRP Electronics Manual changelog.

Version	Date	Comments
1.0	8 February 2010	First draft
1.1	8 February 2010	Second draft
1.2	8 February 2011	First publication
1.3	9 February 2010	Publication during workshop 2010
1.4	18 March 2010	Update adding information about servo programming mode.
1.5	24 September 2011	Updated introduction and micro-controller text.
2.0	17 December 2011	Added sections on safety and integration, updated images.
3.0	4 November 2014	Complete overhaul of electronics with new printed circuit boards.
3.1	15 February 2016	Updates for V2 of PCB and general improvements.
3.2	13 March 2016	Added chapters on firmware, interfacing and fixed several errors.
3.3	11 March 2017	Brought the manual further up to date and fixed several errors.
3.4	December 2018	Replaced the Firmware part with an updated version using Atmel Studio.
3.5	February 2018	Improved the manual by reshuffling and (re)writing chapters.

For questions about to this document, contact any of the Electronics Team members. For mail contact you can reach the DARE Electronics Team at electronics@dare.tudelft.nl.

For questions pertaining to safety regulations, please email the DARE Safety Board at safety@dare.tudelft.nl.

1 Introduction

Welcome to DARE and the SRP - Small Rocket Project! This manual shows you how to put your rocket electronics together to make a successful rocket. The electronics are used to deploy the parachute and perhaps add some additional functionality depening on *your* mission.

1.1 Project Approach

As SRP is open to students with a wide range backgrounds, the DARE Electronics team provides SRP teams with a predesigned piece of SRP Electronics. The design consist of a readily available circuit board and software: the board, components and tools for programming are readily available.

For a simple SRP mission, little electronics design will be required. However, you will experience that you do need to actually understand the given system in any case for flying it or customizing it. Additionally, the hands-on experience will be priceless: there is still plenty of room for mistakes–these mistakes will prepare you for potential future DARE projects.

To this end, we suggest the workflow depicted in Fig. 1. It is essential to understand the presented system as it is the very first step; do not hesitate to ask any of your mentors or Electronics team members when in need of advice, explanation or examples during the project!

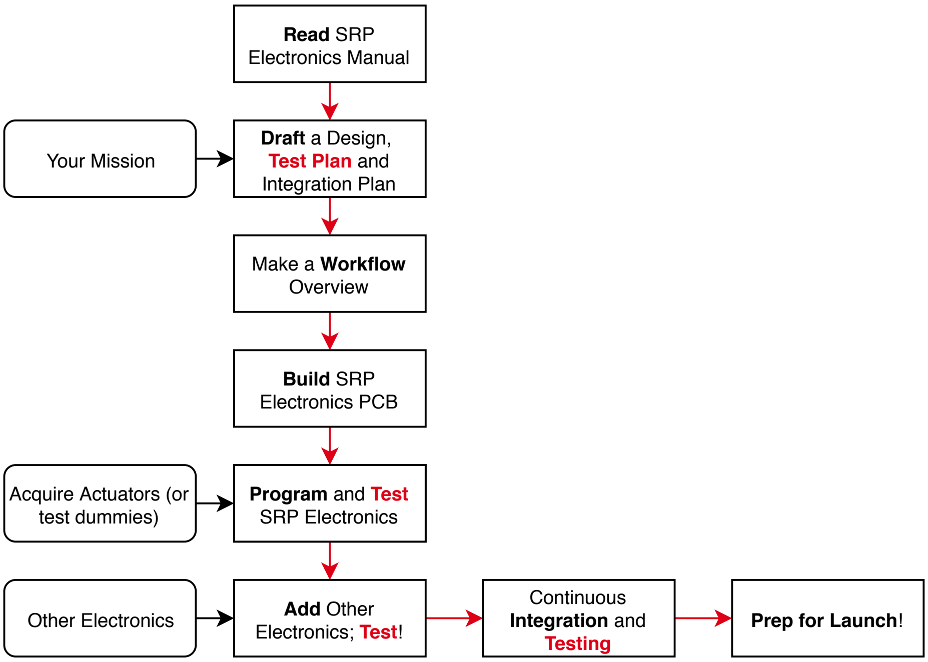


Figure 1: Suggested approach in realizing the electronics systems of your SRP Electronics. Testing is very important as it will allow for noticing problems at the right stage and prevent the need to redo the final product.

1.2 In This Manual

This manual provides a guide into building and using the electronics and is written assuming that the reader is an SRP participant without much (any) experience working with electronics. Before the actual description of the electronics system, critical safety aspects are covered in Chapter 2. These aspects are applicable to all readers, but especially to those who build their own electronics or modify the existing designs.

The remainder of the manual is organized as follows. Chapter 3 overviews the electronics from a high-level perspective and provides fundamental understanding. Then, Chapter 4 covers a somewhat deeper electronics knowledge by explaining the components that will be encountered when building the circuit board. By following Chapter 5, you will be able to solder the circuit board as it contains the design and build instructions. Chapter 6 presents information about the software: what a typical launch sequence looks like, how to program and how to use the electronics. Finally, Chapter 7 gives insight on integration of the electronics in the rocket.

1.3 System Customization

The design offers two strategies to customize your electronics beyond the standard configuration.

Customizing Firmware The SRP board has several extra pins available that do not have any specified function with the default software. You can customize the SRP software to use these pins for your own purposes.

Customizing Hardware While building your own electronics is certainly possible, it is not recommended as it takes significant amount of knowledge and is rather time consuming. The presented design is designed to be flexible enough to fit virtually any setup. However, a less time consuming strategy is augmenting your SRP electronics using ready-made boards like an Arduino Nano.

A few customization ideas are described in Appendix A.

2 Safety

Safety is the most important element when working with rockets. The electronics of the actuation systems may give rise to safety issues when handled without care. The most obvious actuation system with safety implications are pyrotechnical charges, since the igniters and black powder used for these are highly energetic explosives. However, even when not working with explosives, care should be taken.

Typically the electronics are turned on before the rocket is put in the tower. If the electronics inadvertently decides to trigger the parachute release mechanism while carrying the rocket with loaded motor to the tower or inserting the igniter in the motor, unsafe situations may arise. These situations pose a danger to the people handling rocket, so safe design, implementation and operation are critical.

Designing a fail-safe electronic system is based on the principle that there should never be a single event that can trigger the electronics. The design should aim for multiple switches that need to be triggered to prevent activation when a single component fails in the on-state.

2.1 Regulations

The Safety Board has imposed a set of regulations in order to ensure that all electronics that go into a rocket are safe. The regulations are as follows:

2.1.1 Power

1. Electronics should be able to be easily turned off (Unless previously agreed upon with the safety board).
2. Electronics for SRP rockets should contain a power switch on the outside of the rocket
3. Use of batteries or any other power supply without polarity protection is not allowed.

2.1.2 Handling Electronics

4. Unless previously agreed upon with the safety board, use of a breakwire[1] is mandatory.
5. Electronics should provide a visual or audible signal when it is turned on or its state is changed.
6. Use of transmitters should be reported to a person designated by the safety board at least one month before launch.

2.1.3 Electronic Triggering Explosive Devices

7. All electronics that trigger an explosive device should contain a mechanical safe/arm switch or plug that mechanically disconnects the explosive device from the power source.
8. The safe/arm switch should be located such that the safety officer can easily toggle it while the rocket is in the launch tower.
9. Toggling of the safe/arm switch by the safety officer should be the last action to be performed on the rocket.
10. Toggling the safe/arm switch should provide a visual or audible signal to indicate the state change.
11. The schematics of an electronic system that triggers an explosive device, should be reviewed at least a month before launch by a person designated by the safety board .
12. The hardware of an electronic system that triggers an explosive device, should be reviewed at least a week before launch by a person designated by the safety board .
13. The safety board has the final authority to disallow use of certain electronic systems if they deem them to be unsafe, in violation of the safety regulations, or for any other reason, even if the system is in full compliance with the regulations above.

[1] Breakwire = small wire bridge that is connected with a rope to the launch tower. During the launch the breakwire will disconnect, signaling the rocket that lift-off has occurred.

2.2 Safe Design in Practice

The electronics systems described in this manual are designed in compliance with the regulations above. Therefore you should only have to deal with regulation 12. Should you however decide to design and build your own system, or buy a pre-built system, it is important to take all these regulations into account.

To help complying with the regulations, a list of practical implementations follows. These have already been considered for the SRP board.

- **Polarity protection** The best way to implement this is to put a diode directly behind the power connector. Alternatively, it is possible to only use components that are inherently protected against connecting them wrong, but unless there is a good reason for doing this, the diode-solution is preferred.
- **Visual or audible signal** This can be done by providing a beeping sound when the state is changed, or have multiple LEDs at the outside of the rocket, which indicate the state.
- **Transmitter use** The reason for this regulation is the fact that transmitters transmit energy, which may be received by the ignition lines, causing premature explosion. Transmitters may also have influence on on-board sensors.
- **Mechanical safe/arm** This can be a switch on the outside, or a plug that needs to be inserted and stay inserted during flight.

3 SRP Electronics Overview

Electronics can be experienced as complex, but it is certainly not magic[2]...! As described earlier, the core "SRP Electronics" is already designed and ready to be built. The SRP Electronics is a flexible package that can be used for simple rocket launches, enabling different kinds of recovery options under safe operations: a the critical mission objective for many launches. In addition, the design allows for easy integration with other electronics.

The design takes the form of a small 5×5 cm hexagonal printed circuit board (PCB) that can be mounted inside a designated space inside a rocket. An example of SRP electronics mounted for usage in a bigger sized rocket is shown in Fig. 2.

To offer a good basis for the remainder of the SRP project and to avoid the thought that electronics draws from some mysterious force, this chapter describes the high-level design of the SRP PCB and illustrates the logical reasoning that went into it. Whether you end up doing electronics later on does not matter–in order to really become a good rocket engineer, it is essential to understand what the electronics subsystems are capable of and how the functionality is realized.

In short, in this chapter we give a high-level decomposition of the subsystems that are present on the PCB. After that we highlight the physical interfaces that allow for the main and customizable functionalities of your specific implementation. Finally, we give more detail on possible actuator options for parachute deployment.

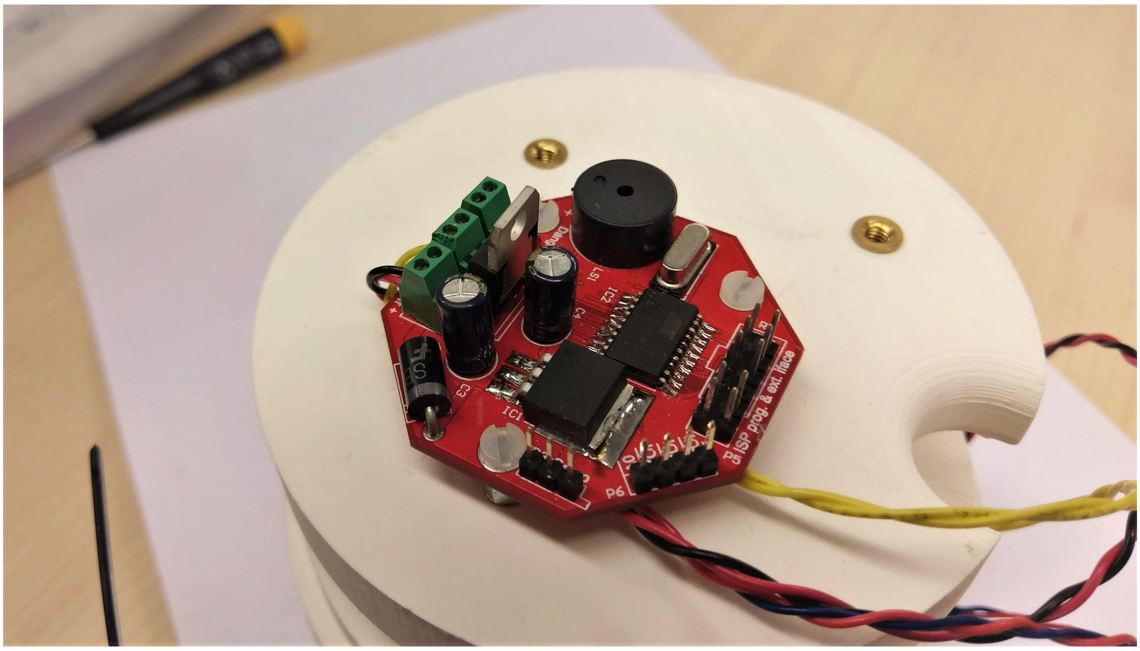


Figure 2: SRP Electronics PCB (without wiring!) mounted on a 3D-printed structure for the side-project "Stratos 3 Minus" (Fall 2018).

3.1 Design Motivation

The safety requirements, mission requirements and the predicted scope of use-cases drive the design for a set of desired functionalities. We can list the main functionalities of the SRP PCB in a simple manner:

- The SRP PCB can be used for typical SRP rockets.
  - The SRP PCB can run from different power sources (e.g. 9V battery, LiPo battery).

[2]Except for antenna design, that's dark magic.

  - The SRP PCB can fit in a cylinder roughly the size of a SRP engine and be easily mounted (not a real "function", but added for completeness).

- The SRP PCB can perform parachute deployment.
  - The SRP PCB can actuate a pyro charge (more details later in this chapter).
  - The SRP PCB can actuate a regular servo.
  - The SRP PCB can track time in order to trigger deployment at the right moment (X seconds after breakwire release, depending on your configurations).

- The SRP PCB can be safely operated.
  - The SRP PCB can be armed and only actuate after that moment.
  - The SRP PCB can detect whether a pyro is attached or not.
  - The SRP PCB has audible and visible ways to alert its state.

- The SRP PCB can be easily configured and expanded.
  - The SRP PCB can be configured for a specific mission without needing full reprogramming.
  - The SRP PCB can take an external signal and act depending on it.
  - The SRP PCB can output a signal to trigger other electronics for the flight sequence.
  - The SRP PCB's software can be modified to output custom signals.

Would you be satisfied with the set of above functions for *your mission*? These functions are the driver for the designed SRP Electronics you will be building and testing as you follow the manual. Keep them in mind, you will encounter them continuously.

3.2 System Decomposition

Fig. 3 shows the high-level breakdown of the resulting SRP PCB's subsystems and the connections to the "off-board" world. Have a good look and read the paragraphs in this section explaining the motivation and functions of the system's design.

PCB As you will recognize later on, the PCB is simply put a piece of glass-fiber with a network of copper traces (flat, narrow paths of copper foil) on either side to connect different components. Connected together, these components form an electrical circuit with specific functions: the green and red boxes in Fig. 3.

Aside from circuits, the PCB encompasses connectors to connect to the outside world like actuators or its power source. We cover these "external interfaces" in more detail in the following section.

Microcontroller The microcontroller is the brains of the SRP electronics as can be deduced from its many connections. For now, it is sufficient to know that the microcontroller is similar to a very small computer and implements the behaviour that we want.

Without behaviour, we can't really use the PCB (or other electronics): it is an integral part of any rocket that we fly. As a result, a huge amount of work actually needs to be spent on writing *firmware* (code for the microcontroller), which is very often overlooked.

Luckily, like the SRP PCB hardware design, the firmware has already been written. As the firmware and the behaviour it implements is of such critical importance, we focus our attention on the "software side" of the SRP functionality in Chapter 6.

Power By down-regulation of the voltage at the input of the circuit, we can allow for an input voltage range from roughly 7 to 10 V. The internal circuitry on the PCB runs mainly on 5V. The microcontroller monitors the voltage level and alerts the user with annoying beeps from the buzzer if the level is too low. The "raw" voltage taken directly from the source triggers the actuator on the Pyro port, but is fed through some safety circuitry first.

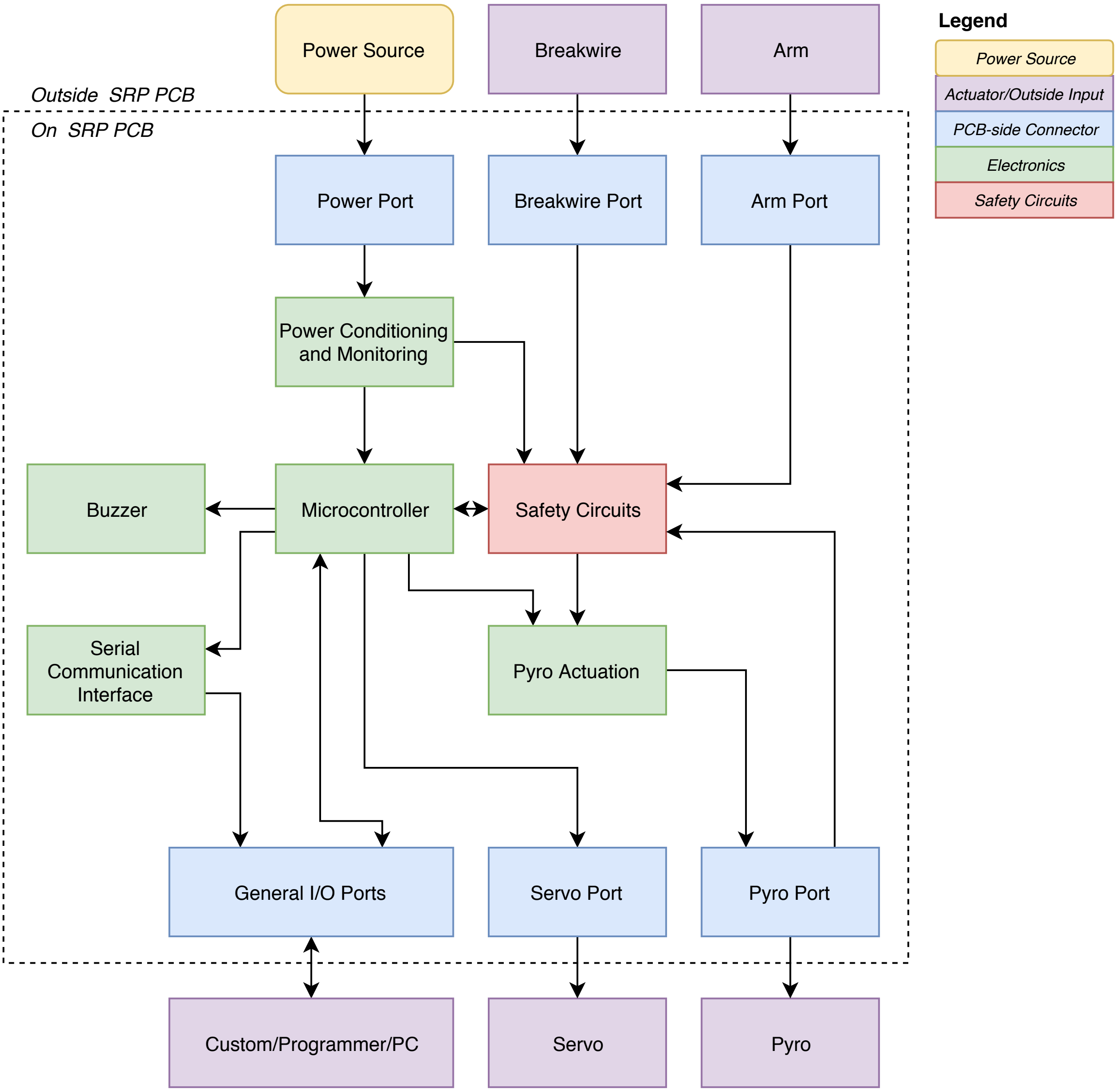


Figure 3: Pinout of the SRP Board external interface connector.

Safety Circuits While the behaviour implemented in the firmware explained later inherently implements safe operation, the SRP PCB also needs some hardware-implemented safety. Without going into too much electrical engineering details, it should be known that these circuits mostly rely on electrical switches (transistors) or mechanical switches (for example, the arm switch).

For SRP, safety must be guaranteed when working with a pyro charge. So, the safety circuits implement the following:

- Allows for current to flow to the pyro port if and only if
  - the arm switch is closed AND
  - the breakwire has been released AND
  - the microcontroller enters the deployment mode during (flight).

- An LED is activated when the microntroller activates its signal that is to trigger the pyro actuation circuit. This prevents attachment of the pyro when there might be a voltage on the pyro port.
- Measurement of whether the pyro is actually attached and relay this info to the microcontroller (continuity detection). This is more a feature for the mission, not so for safety, but included for completeness.

Other Subsystems The microcontroller is able to directly drive any typical 5V servo, more on that later in this chapter. Moreover, some pins can be used for custom signals, to trigger the microcontroller, or trigger external electronics by means of the microcontroller. Finally, the microcontroller needs a port for programming and for serial communication. You will be able to configure the microcontroller via this serial communication link at any time with a computer, as explained in Section 6.3.

3.3 PCB Interfaces

The next topic to be discussed are the interfaces between the SRP Board and the outside world. The SRP Board has several headers and terminals on it, each of which with a specific function. Fig. 4 labels the connectors of the SRP PCB. This section covers all the connectors/headers and what should be connected to them.

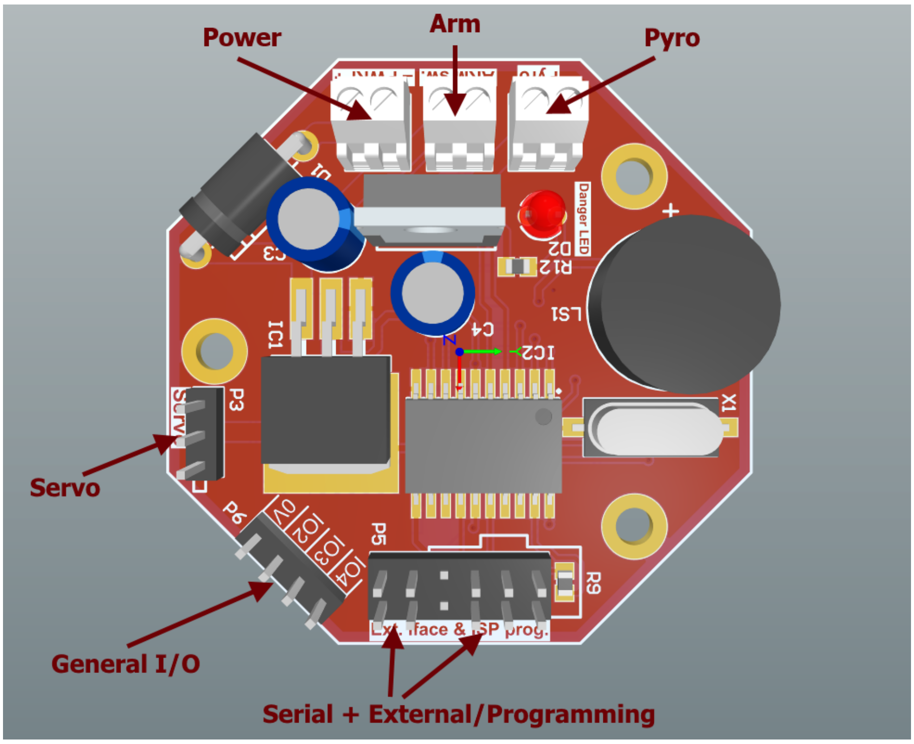


Figure 4: Connectors on the SRP PCB.

Terminals The SRP Board features three sets of screw terminals labeled “Power”, “Arm” and “Pyro” (often referred to as ports).

The “Power” set of screw terminals is where the battery will be attached. As stated before in Chapter 2, it is required that the rocket can be powered on/off using a switch. This switch can be attached to one of the lines between the battery and the power terminal. The power terminal also has + and - markings printed next to it to show how the battery leads should be connected to the terminal. Care should be taken that the wires between the battery and the SRP Board are thick enough for the current they are supposed to handle.

The “Arm” screw terminals are where the arm switch will be attached to the SRP Board. When the two poles of this screw terminal block are connected, the SRP Board will be armed. As stated in Chapter 2, arming the rocket should be performed using a switch mounted on the outside of the rocket. This switch should therefore be connected to the arm screw terminal block. Since these wires are also part of the power circuit they should be tick enough to handle the current as well.

Finally, the “Pyro” screw terminal block is where a pyro charge or wire cutter can be connected. Of course, thick enough wires must again be used to connect the pyro or wire cutter due to their current demands. If a servo is used instead, this screw terminal block should be left unconnected.

External Interface Connector On the other side of the board from the screw terminals, a 2x6 pin header is positioned. This header has multiple functions:

1. it is where a programmer has to be connected to program the microcontroller;
2. it contains connections for the breakwire and status LED;
3. it contains the communication interface over which the SRP board can be configured; and finally
4. it contains two pins which can be used to communicate with any additional electronics (serial communication over RS232).

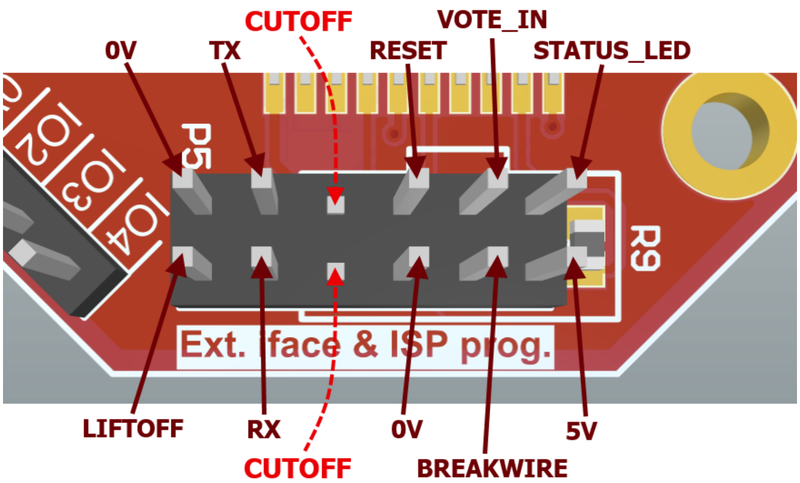


Figure 5: Pinout of the SRP Board external interface connector.

Fig. 5 overviews the functions of the different pins of the external interface connector. As will be shown later, a debug connector is available to test the SRP PCB and configure it during development. The description of the different pins is as follows:

- *Reset*: This is only used when programming the microcontroller. Under all other circumstances it should be disconnected.
- Cut off: These pins should be cut off to make space for the programming connector. They are of no use.
- *Breakwire* / 5 V: The breakwire should be connected between these pins. When the breakwire pin detects it is no longer connected to the 5V line, the timers on board start counting.
- *Status LED / 0 V*: A status LED can be connected between these pins. The status LED pin is powered to 5V through a 330 Ohm series resistor so most normal LEDs can be directly connected to these pins.
- *Data in (RX) / Data out (TX) / 0 V*: These pins are used for the RS232 interface over which the SRP Board can be configured. If it is desired that the board can be configured while the rocket is assembled, these pins should be connected to a three pin header on the outside of the rocket. These pins should be connected in the order 0 V, data out, data in.

- *Liftoff*: This pin is by default 0 V, but will be set to 5 V when the breakwire is released while the rocket is armed. Since it is directly coming from the microcontroller it cannot drive high currents.
- *Vote in*: This pin is normally kept high (5 V) with a pull-up resistor. If any external electronics want to deploy the parachute, they should pull this pin low at that moment. Together with the Liftoff pin, these two pins can be used by external electronics to deploy the parachute more accurately than the timer on the SRP Board.

GPIO Pins Next to the external interface connector a four-pin header is located, so-called "general purpose input and output" (GPIO). These pins are not normally used, but you can customize the firmware on the board to perform extra actions on these pins.

Servo Connector Next to the GPIO pins a three-pin header is located. This header is where a typical 5V servo can be connected. The pin closest to the GPIO pins carries the PWM signal for the servo. The centre pin carries the power and the third pin is the ground.

3.4 Actuation Hardware Options

The SRP electronics provide the option of igniting a pyro or controlling a servo mechanism. To be able to assemble the electronics, it must to be understood how they operate and under what circumstances.

E.g., your on-board power source must be able to supply the operating current! We provide short descriptions of these hardware with specifications that are important from an electronics perspective.

Pyro Pyro charges are triggered by passing a short, high current through the squib, setting it off.

Actuation Time	Quasi-Instant
Current	$\sim$ 0.5 amps
Voltage	Battery Voltage
Actuator	Transistor
Connection Point	2-Pin Screw Terminal

Dyneema Wire From an operation standpoint, Pyro charges and Dyneema wire triggers are very similar, the difference being that current for the Dyneema is turned on considerably longer. The Dyneema is burned through with a resistor, releasing whatever it is holding together.

Actuation Time	$\sim$ 4 seconds
Current	$\sim$ 1 amps
Voltage	Battery Voltage
Actuator	Transistor
Connection Point	2-Pin Screw Terminal

Servo Servo are in the simplest of cases electric motor packages that take three inputs–5 volts, ground and a pulsewidth-modulated signal.

They have internal circuitry to position the servo between 0 and 180 degrees. (There are more versions, but they are not covered here). This makes them very useful to do functions such as open a hatch of release something. The generally come in roughly one size, although both smaller and larger versions are also available.

Actuation Time	0.5 seconds
Current	0.5-2 amps depending on servo torque
Voltage	5 volts
Actuator	Signal from micro-controller
Connection Point	3-Pin Male Header

4 Components

Most SRP participants are new to electronics, so this chapter lists the components that will be soldered onto the PCB with a brief explanation, description or their properties.

It is definitely recommended to (at least) skim this chapter when you are unsure of your electronics understanding: when working in multidisciplinary teams or when asking advice from someone of a different expertise, it will be very useful to be able to understand their technical lingo.

At this point you might be curious how the real circuits of the SRP PCB look like. You can have a look at the schematic sheet provided in Appendix C. For more basic background on electronics (for example, basic relations on current and voltage) refer to this excellent article: `https://predictabledesigns.com/an-introduction-to-basic-electronics/`.

Battery The battery depends on what you are going to use to deploy your parachute. If you intend to use a servo, a dry cell 9V battery should be enough. On the other hand for a Dyneema wire, you will have to use a Lithium Polymer battery that can provide a higher current.

Buzzer The buzzer produces a sound when a current is passed to it. This makes it a good tool for providing feedback on for instance the system status. It also has the advantage of being more clear than the LED on a sunny day outside.

Capacitor A capacitor is a kind of energy storage. It will try to keep the voltage across it constant. Capacitors are used to filter out fluctuations in the voltage in order supply a constant voltage.

Diode A diode is a semiconductor that only allows current to pass through it in one direction: from the anode (+) to the cathode (-). It is used to ensure that no electronics is harmed or pyros detonate when the battery is connected wrongly. There are two types of diodes in use: a big and a small one. They are exactly alike except for their size and current throughput.

A light emitting diode (LED) is a diode that emits light when current is passing through it. For a LED the anode (+) is the long lead, while the cathode is the short one (-). It is used to send signals to the outside world. It will not be soldered on the circuit board, but connected with wires to a terminal socket and should be mounted on the outside of the rocket.

Microcontroller A microcontroller is a device that acts like a tiny computer by writing software for it. A program written in the C or assembly programming language can be put on the micro-controller. A microcontroller has several inputs that the program can read out, and based on that it can do something with its output.

The micro-controller will be soldered directly on the circuit board. It should be noted that both the microcontroller and the socket have a side with a circular cut out. They indicate the side of pin 1 and should correspond when soldering the microcontroller on the PCB.

Pin Headers Pin headers are pins to allow easy connection with other electronic subsystems. In this circuit they are only used on places that are not connected in flight: the programming button and the buzzer measurement pin.

Resistor A resistor is an element that transforms electrical power into heat. It has various uses: for LEDs it is used to limit the current through the LED; pull-up and pull-downs to define a line when it is not driven by the micro-controller or another source; or when used in multiples as a voltage divider. The value of a resistor can be read using the coloured stripes printed on it, starting from the strips which are close together. Resistors can be connected in any way.

Both polarized and unpolarized versions exist. Polarized versions have a (+) in the schematics. This corresponds to the longer lead. In most cases on the (-) side a series of dashes (- - - -) is printed along the length. Unpolarized capacitors can be connected in any way.

Switch A switch is used to easily make and break a connection. It is not indicated in the schematics. Wires can be soldered to the middle and either of the outer leads. It is used for switching the power; as such it should be connected to the red wire of the battery connector and have another wire soldered to the other lead. It is also used as ARM switch. Simply solder two wires to the switch.

Terminal Headers Terminal headers are used to connect wires to the circuit. Care should be taken that they be placed such that wires can easily be connected to them, even when the components on the other side of them (switches, LED and breakwire) are mounted on the rocket.

Transistor A transistor is essentially an electronic switch. When a predefined voltage is applied to its gate (G), it will conduct current from drain (D) to source (S). Otherwise it will not conduct. They exist in so-called P- and N- channel versions. N-channel versions will conduct if the gate is made positive with respect to the source, P-channel versions when the gate is made negative.

Voltage Regulator A voltage regulator converts a voltage to a certain stable voltage. The output voltage in this case is essentially fixed at 5V between a voltage range and up to a certain current. It is used because most electronics will only work at a prescribed voltage. The voltage used in the most of the board is 5 volts, which along with 3.3 volts are the standards in electronics.

5 Building the Electronics

To construct your SRP Board, you will be given pre-made PCB board onto which you will solder your components. It's a good idea to practise on a simple piece of prototype board with some LEDs, resistors or other cheap components before going loose on SRP electronics if it's your first time soldering. The total cost of all the parts are around 15 euros and there are only a finite number in stock in the lab, so it's a nice idea to be a little bit careful with them.

N.B.: multiple versions of the SRP boards exist. Only use the latest board that works with servos, wire burning and pyros as Fig. 6 shows. These are labeled accordingly.

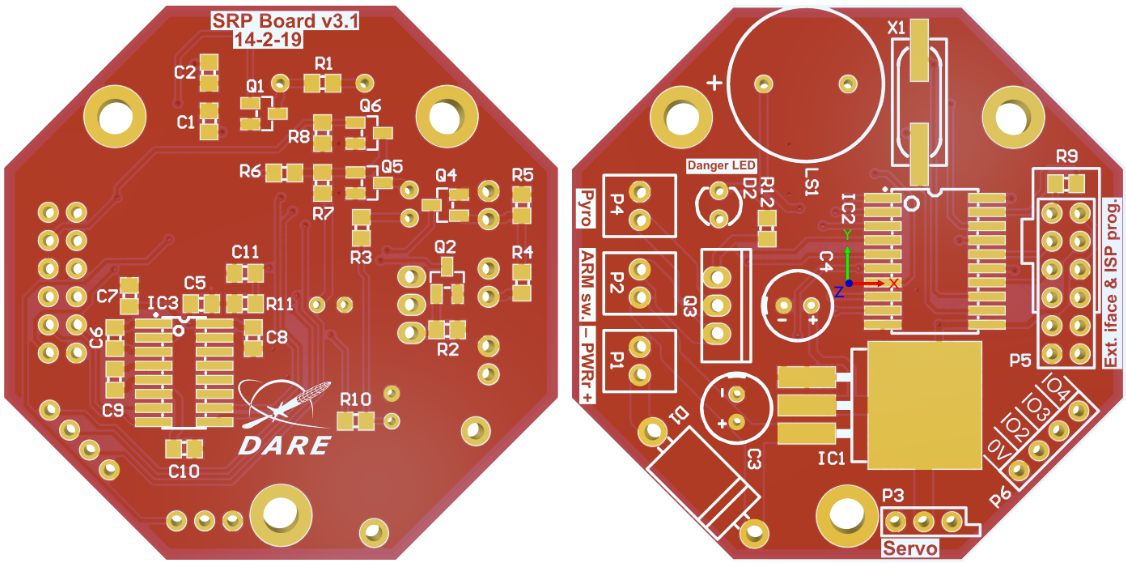


Figure 6: Bottom and top side of a bare SRP PCB. The date will vary, but the DARE logo is on the latest version only.

5.1 Preparation

- Get soldering tin and a soldering iron and put them on a clean table.
- Get a PCB and collect components as detailed in the appendix in a bowl or similar.
- Put the instructions close at hand.

5.2 Surface Mount

The SRP Board contains a few surface mount components which are just soldered onto the surface. If you open up any modern gadgets, they'll have the same design- they are smaller, lighter and cheaper than through-hole components. They are also a bit harder to solder.

How to: Apply solder to one exposed pad of the component–just a little! Hold the part with the tweezer and carefully heat the just-soldered pad while gently pushing the component towards it. As the solder melts, the part will gently float down, locking down one side. Then apply heat and solder to the other side to attach it as well. Rework the result if needed and make sure that the component is aligned and lying flat.

5.3 Through-Hole Components

Through hole components have long leads that go through the circuit boards and out the other side. At the intersection between the board and leads, solder is melted to form an electrical connection.

How to: Hold the solder on one side of the lead, and the hot soldering iron on the other. As the solder melts, it will flow towards the hotter lead and will form a connection. Apply sufficient solder until you achieve a little mountain.

5.4 Construction

Here we provide a step-by-step building instruction for the SRP boards. Be sure to ask help when you are in doubt, and to follow the electronics soldering instruction. Many errors can occur from improper soldering. Follow the order of the components in below tables closely. The order allows for optimal comfort in handling the soldering iron.

Finally, ground yourself when soldering to avoid electrostatic discharge (ESD): any surplus of charge stored in your body can destroy delicate electronics. The soldering tables have ESD protective wristbands, ask for instructions on usage.

Part 1: Bottom Side SMD Parts: Flip the board and start with soldering the bottom side. Do the following parts in order:

Board Code	*Value*	*Description*	*Format*
C1, C2	22pF	Capacitors	0805 footprint
IC3	MAX202	Chip	16 pins, slim
C5, C6, C7, C8, C9, C10, C11	100nF	Capacitors	0805 footprint
Q1, Q2, Q4, Q5, Q6	NMOS	Transistors	SOT-23 (T-Shape)
R2, R4, R5, R7, R8, R10, R11	100k Ohm	Resistors	0805 footprint ('104')
R1, R3, R6	1k Ohm	Resistors	0805 footprint ('102')

Part 2: Upper Side SMD Parts: The upper side is the one with the large rectangular pad. Do the following parts in order:

Board Code	*Value*	*Description*	*Format*
X1	7.372 MHz	Crystal	4HSMX
IC2	ATTiny4313	Chip	20 pins, wide
R12	100 Ohm	Resistor	0805 footprint
R9	330 Ohm	Resistor	0805 footprint
IC1	5V	Voltage Regulator	3 pins

Part 3: Through-Hole Parts Finally, solder the through hole components. You'll need to clamp it or attach the screws to prevent it from wobbling. Do the following parts in order:

Board Code	*Value*	*Description*	*Format*
LS1	Piezoelectric	Buzzer	14mm diameter
P1, P2, P4	Double	Screw Terminals	100 mil
P3	3x1	Pin Header	100 mil
P6	4x1	Pin Header	100 mil
P5	6x2	Pin Header	100 mil
Q3	PNP	Transistor	TO-225
D2	Red, 3 mm	LED, Red	w/ polarity notch
C3, C4	100uF	Capacitor	⌀6.3mm H11mm
D1	5A/60V	Diode	DO201AD Footprint

Do not forget to cut off the pins of the external interface connector as described in Chapter 3!

Part 4: Little Extras: Other things your rocket will need.

Descrption	*Value*	*Description*	*Format*
Breakwire	30 cm	Two wires	two-pin female crimp header on the end.
Switch Wire	30 cm	Two wires	Switch soldered on the end.

5.5 Testing

Testing is usually done with a multimeter. When a voltage needs to be measured, make sure that the battery is connected. Hold the red probe to the point to be measured and the black probe to the minus of the battery, the ground, unless instructed otherwise. When done measuring disconnect the battery.

You can test/check the following to make sure that your electronics work:

1. Put the entire board under the microscope and make sure that all connections are soldered. Check that all the required parts have been mounted on the board.
2. Power the board. Take a multimeter and measure the two pins on the servo connection on the other side of the little footprint indicator. There should be 5V between these pins.
3. Measure pin 2 and pin 15 of the MAX202 chip (IC3) on the back to check that it is powered as well.

6 Establishing Functionality

Now that the SRP PCB is successfully assembled and we are sure that the parts are properly powered, we have a piece of rocket hardware! However, it still needs to start to behave as we require for a successful mission: the SRP PCB requires software to run on the microcontroller. Similar to the PCB design, this software (also often referred to as *firmware*) has been pre-written to handle all the requirements on the SRP Board. The software has to be flashed onto the microcontroller in order for the board to function as expected.

This section is structured as follows. First the way software is flashed on to the microcontroller will be explained, followed by an elaboration on the functionality of the software. After this a list of configurable options will be provided as well as how these can be configured. Finally, the way the software communicates with the outside world will be shown.

6.1 State Machine

The functionality of the SRP Board software is implemented as a state machine. In short, this means that the board has a set of states to be in with, e.g. *preparation* or *systems_check*, with special conditions that allows fror transitioning into a different state. Fig. 7 describes the state machine in the form of a "finite state diagram". Be sure to understand the flow of the states and the below general remarks about this state machine:

- Actions relating to the pyro system will only be taken if the board is configured to actuate the pyro system. The same holds for the servo system.
- Since the presence of a servo cannot be detected electronically, this is not checked by the state machine.
- The board will only check if the battery is charged when it boots up. This is to prevent situations where the board could enter the error mode while it is already in the launch tower.
- Whenever the state progresses from SYSTEMS_CHECK to ARMED, the board will make a short beeping noise twice. If the state moves in the opposite direction a single long beep will be heard.
- If an error is made in the activation sequence, the software will go into the ERROR state. It will only move back into another state once it has been disarmed.
- The board will toggle the status LED every time a state transition happens along the expected path in the procedures.
- The board will deploy the parachute when a maximum time after lift-off is reached, or when a minimum time after lift-off has been reached and an external board sends a signal that it wants to deploy the parachute.

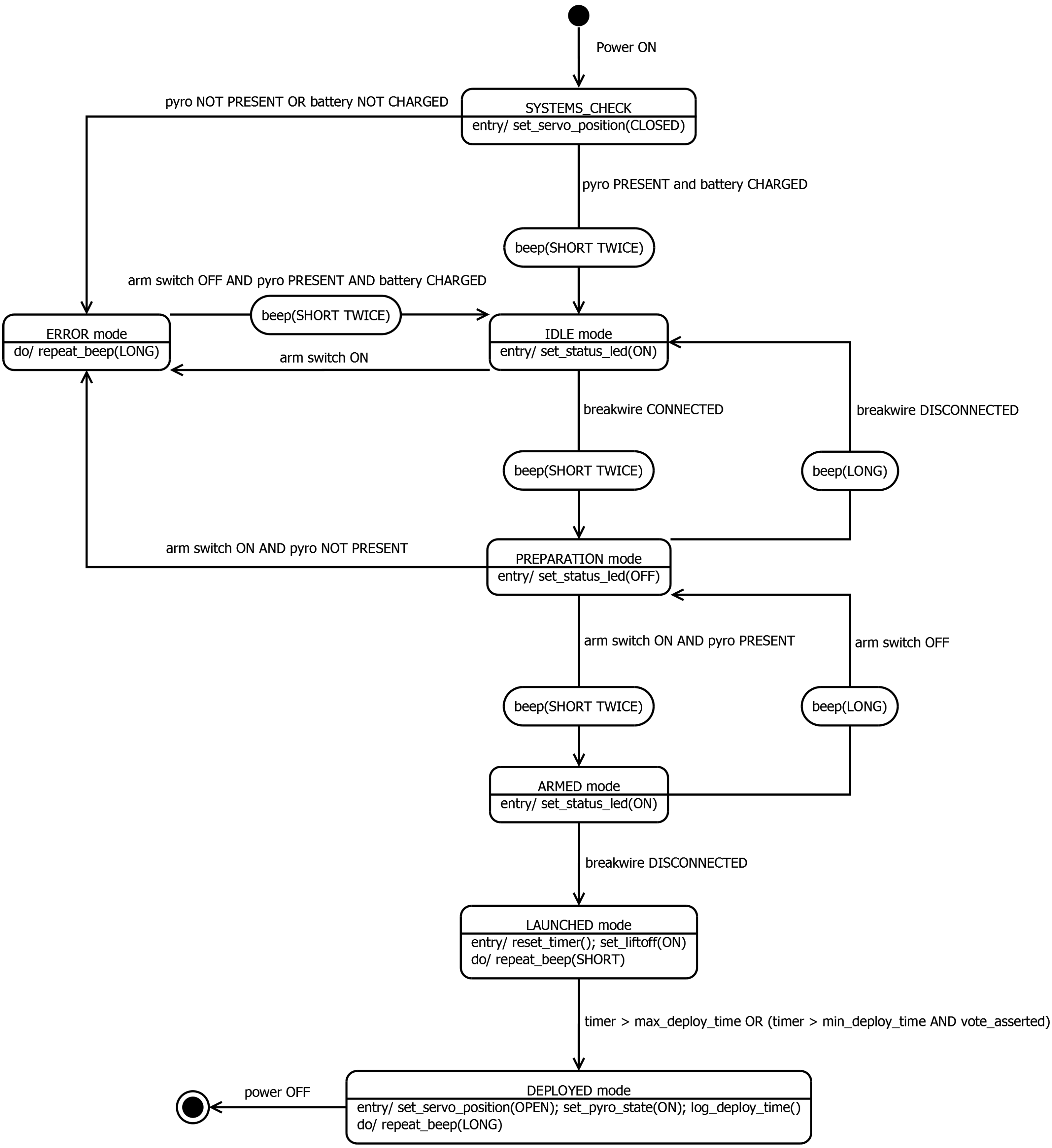


Figure 7: SRP microcontroller state diagram.

6.2 Programming

Depending on your experience, programming hardware like a bare SRP PCB might seem quite intimidating. The below steps will guide you through the process: if followed closely it should be smooth process.

First, we will set up the programming environment, get the firmware and prepare the physical working environment. After that we hook up the SRP PCB to the PC and flash the firmware. As a final step, some thorough testing is needed to ensure that the PCB is ready for configuration and further integration and testing. Please note that the following procedures have only been tested on a machine running Windows 10.

Computer Setup On the PC side we need to install the IDE[3] that will be used and load in the default SRP firmware as a project.

1. Go to `https://www.microchip.com/mplab/avr-support/atmel-studio-7` and download the *Atmel Studio 7.0 installer* (see the *downloads* tabs halfway on the webpage). We are assuming you use the web installer, but the offline installer should work the same.
2. Proceed with installation of Atmel Studio 7.0 by running the installer.
3. During the install, when prompted to pick your architecture, tick the *AVR 8-bit MCU* option. This is the architecture of the SRP microcontroller: of course, if you want you can tick more boxes. Installing the *Atmel Software Framework* and *Example Projects* is not required for SRP programming.
4. Let the installation finish and launch Atmel Studio to confirm the successful installation.
5. Navigate to `https://github.com/JunDARE/srp-electronics` and download the GitHub repository as a .zip file (press the *Download ZIP* button which is located under the "Clone or Download" drop down).
6. Unzip the folder to a convenient place on your computer: the *src* directory contains the firmware and will be your Atmel Studio project folder for SRP. The *configuretool* directory will be used later on.
7. In Atmel Studio, go to *File > Open > Project Solution...* and navigate to the *src* directory of the unzipped folder. Select the `SRP.cppproj` file to load the SRP project into Atmel Studio.
8. In the most-right panel, the *Solution explorer*, the SRP project has appeared. Right-click on "SRP" and click *Build*, see Fig. 8. This command compiles the SRP firmware and outputs the necessary files that will be used to flash the current firmware onto the microcontroller in an automatically generated *Debug* subfolder.

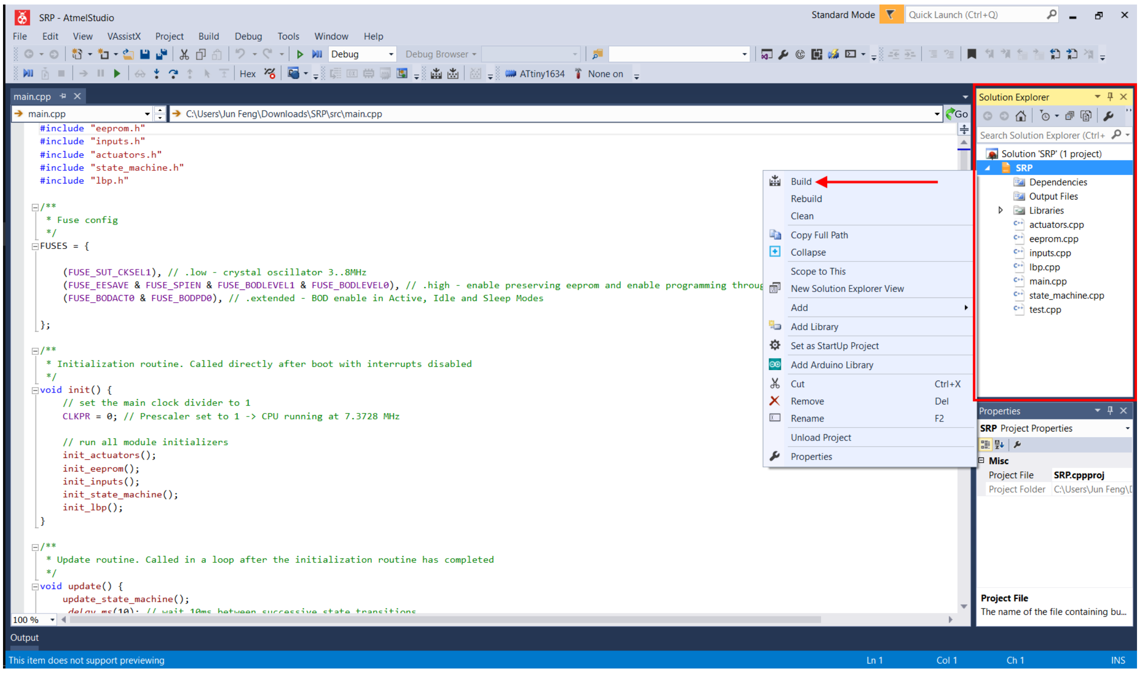
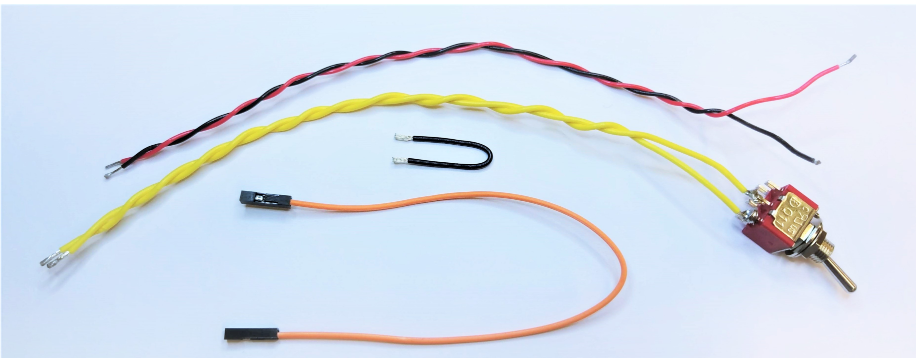


Figure 8: Building the SRP project in Atmel Studio.

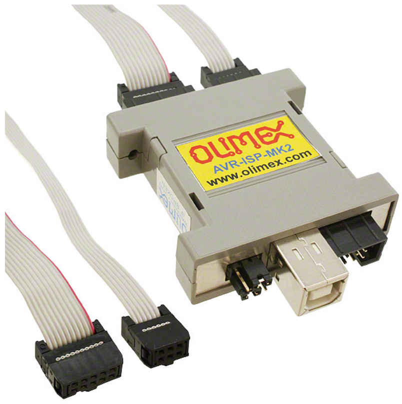
[3] Integrated Development Environment or simply put a programming software package with graphical interface.

Hardware Setup Follow the below steps to setup the SRP PCB, the programmer and wiring correctly, so that the microcontroller can be flashed via Atmel Studio and tested right after. It strongly recommended to create a clean and spacious work area so you don't lose components or end up damaging your hardware by accident. In addition, wear an ESD wristband if possible and if you are not able to find some of the mentioned parts, ask for help!

1. Make the following wires (Fig. 9a):
    - a pair of red a black wires stripped on both ends to power your SRP board;
    - a short stripped wire to emulate a pyro for the SRP board; and
    - a pair of stripped wires to emulated the ARM switch.
2. Collect the following items:
    - a simple wire with connections on either side that can go on the breakwire connector;
    - an Olimex programmer (Fig. 9b) with its USB and programming cables; and
    - a power supply with clips to connect to your pair of power cables (or a battery that can also power SRP).
3. Make sure that the Olimex programmer is correctly configured: refer to the yellow sticker on the programmer and set the TARGET jumper in OFF mode.
4. Connect the programmer to your computer. Necessary drivers should automatically install.
5. Check the settings of the power supply: set it at 9V, and limit the current to 0.5 A (this current limit is a little bit conservative for normal operation, but for programming it will be definitely sufficient).
6. Turn off the power supply and connect your self-made power cables to the "Power" port of the SRP board and attach clips of the power supply to the other end of the wires.



(a) Wires.



(b) Programmer.

Figure 9: Required (a) wiring as an example and (b) Olimex AVR-ISP programmer. On the left, the yellow wires are meant to test arming and the orange wire is for a breakwire connection.

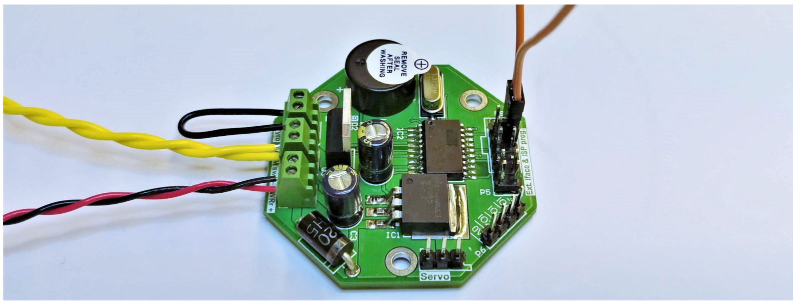


Figure 10: Typical SRP wiring setup for testing. Note that the Servo port would be connected when testing for a servo-actuated deployment rocket.

Firmware Flashing With everything well prepared, we are ready to flash the firmware.

1. In Atmel Studio, click *Tools* > *Device Programming*. A new window should pop up now.
2. With the programmer connected to your computer, change the settings at the top of the window as follows: Tool = *AVRISP mkII*, Device = *ATtiny1634* and Interface = *ISP*. Apply these settings by clicking *Apply*.
3. Click on *Interface settings* and change the clock frequency that the programmer will use to 125 kHz. Click on *Set* to apply this setting.
4. Connect the ribbon cable of the programmer to the SRP PCB's "External interface" port as indicated by the white marking on the top of the PCB. Now, power on the SRP PCB via the power supply.
5. To check if the connection is good at this point, you can check if you are getting a voltage and/or device signature reading in the top part of the device programming pop-up window.
6. Click on the *Fuses* tab–we will first set the clock settings of the microcontroller by adjusting its so-called fundamental "fuse bits". Scroll down and make sure that the fuse `LOW.CKDIV8` fuse is disabled. In addition, set the `LOW.SUT_CKSEL` fuse to *Ext. Crystal Osc. 3.0-8.0 MHz...*. See Fig. 12.
7. Click the *Program* button of the *Fuses* menu to apply the fuse settings. Atmel Studio should verify the fuse settings afterwards and let you know it's executed OK.
8. Click on the *Memories* tab and ensure that all *Erase device...* and *Verify...* boxes are ticked. The selected flash file should be the .elf-file that was generated when you built the firmware. Likewise, the selected EEPROM file should be the .eep-file from the same directory as the .elf-file. See Fig. 13.
9. Click on the upper *Program* button. The firmware will now be flashed to the PCB.
10. Click on the lower *Program* button. The default memory configuration (e.g. threshold voltage for low battery voltage and deployment mode) will now be flashed to the PCB.
11. You can remove the Olimex programmer connection to the SRP PCB.

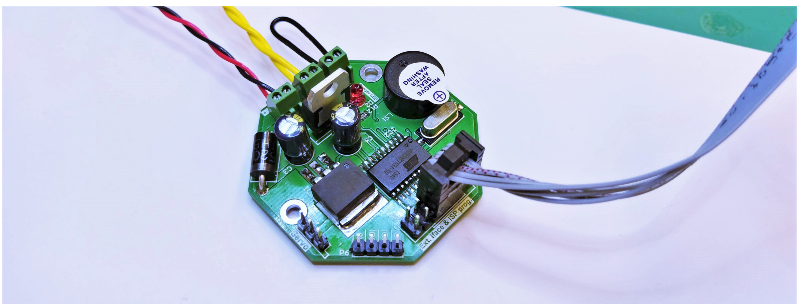


Figure 11: SRP PCB connected at the programming port for programming.

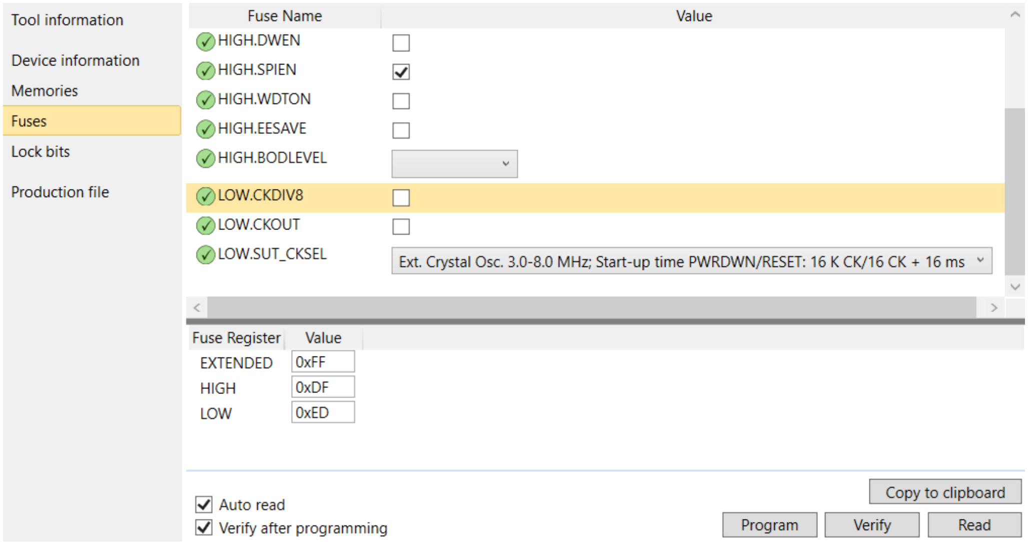


Figure 12: Fuse settings in the device programming menu.

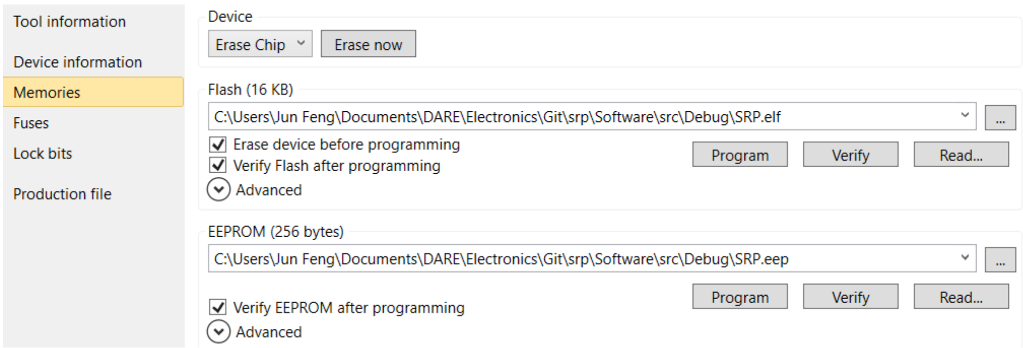


Figure 13: Memories tab in the device programming menu.

Testing Now, we must test whether programming was successful and if the intended behaviour can be executed by the SRP PCB. Have a look at the below questions that will help you in testing. It will be most instructive to have a good look at the state machine overview and write a small test plan for yourself. You are always encouraged to ask a mentor or someone from the electronics team for help or advice.

- Looking at the state diagram, what is the normal order of events? Which scenario's can you test? How can you determine in what state the SRP currently is?
- Reset the SRP PCB state machine by toggling the power supply. Did the PCB go into `SYSTEMS_CHECK` state? What state is the PCB in now, is it as expected?
- Can you trigger the ERROR mode? How do we get out of this state?
- What are the three conditions that can trigger ERROR mode? Write or mark them down, it will be very useful in the future to know these conditions by heart.
- With the previously made wires during preparation, can you test getting all the way to the LAUNCHED mode?
- Can you test disarming the system, i.e. going from ARMED back to IDLE?
- Why does the PREPARATION mode have a transition to the ERROR state? Can you test this transition?

6.3 Configuration

After the board has been programmed, it is possible to configure certain settings of the SRP Board without reprogramming it. Instead, you can communicate with the board using a simple software tool on your laptop and a connection to the PCB. A list of the configurable settings is provided in Table 2 below. Depending on your rocket and mission, you will very likely need to change these settings, e.g. if you are planning to use a pyrotechnic deployment method.

Table 2: SRP microcontroller configuration options.

Option name	**Default value**	**Explanation**
min_deploy_time	10 s	The time after lift-off after which the parachute is allowed to be deployed.
max_deploy_time	14 s	The time after lift-off at which the parachute should always be deployed.
measured_deploy_time	0 s	The measured time after lift-off after which the parachute was actually actuated. This value is read-only.
battery_voltage	- V	The voltage that is used to power the SRP Board. This value is read-only.
battery_empty_limit	6.5 V	If the battery voltage is below this voltage, the board will boot up into error mode.
deploy_mode	servo	The current configuration of this SRP Board. It can be "servo" or "pyro".
servo_closed_position	0	The servo position which should be considered "closed" for the SRP Board. This value can be between 0 and 255.
servo_open_position	255	The servo position which should be considered "open" for the SRP Board. This value can be between 0 and 255.
servo_position	-	This value is write-only. Writing to it sets the servo to the specified position. This value can be between 0 and 255.
address	0	This value represents the name of the SRP Board. It is supposed to be set to the serial number of the board. This value can be between 0 and 255.

Follow the step-by-step guide in the following paragraphs if you want to configure your board. Tip: the necessary connections can also be exposed outside the rocket to allow for the SRP Board to be reconfigured without taking the rocket apart.

Setup It is expected that you still have the folders downloaded from the SRP GitHub repository.

1. If you do not have Python installed yet, install Python to you machine (version 3 or higher will do). Install the `serial` package for Python(`pip install serial`).
2. Collect the following items:
   - Your SRP board with power cables (and the other wire for immediate testing, i.e. ARM, pyro and breakwire wires);
   - the SRP debug connector;
   - an USB-RS232 serial converter (see following figure or ask someone from Electronics); and
   - a power source for your PCB.
3. Plug the debug connector onto the external interface pins of the PCB.
4. Connect the USB converter to the blue-green-black wires of the debug connector (RX, TX and ground pins, respectively in Fig. 5). These are for serial communication. The USB-side of the connector can go into your PC, some drivers might need to be installed automatically.

SRP Configuration Now we power on the PCB and start configuring.

1. Check if the connection the ports of your PCB won't make it boot into the ERROR mode.
2. Set the power supply to 9V, 0.5A, turn off the power supply.
3. Connect the SRP PCB to the power supply and turn on the power supply.
4. Turn to your PC and run the python file `SRP.py` provided in the *configuretool* folder of the downloaded SRP files.
5. Use the `get` and `set` commands to read or configure your configurable SRP PCB options. See Fig. 15.
6. When you're done, close Python and disconnect the serial connection to the PCB. Be sure to test the new settings of your board!

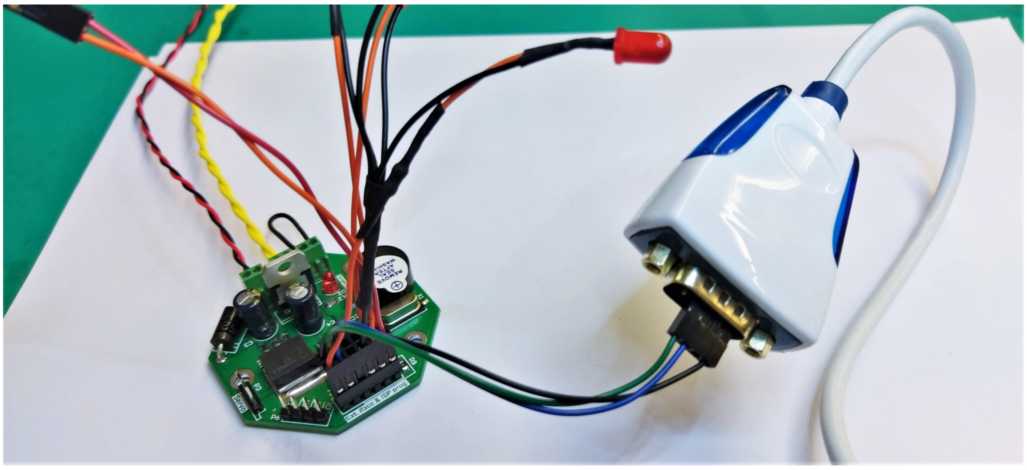


Figure 14: SRP PCB connected at the External interface port for custom configuration using a debug connector and an USB-RS232 serial converter.

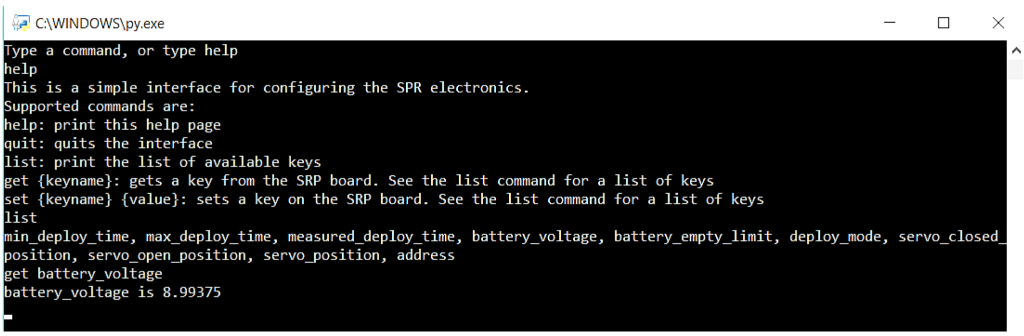


Figure 15: SRP configuretool usage.

7 Rocket Integration and Launch

An often underestimated part of the process is integration of the electronics with the rest of the rocket. Integration consists of very simple things how to mount the electronics in the rocket, to more complicated things, such as the electrical interface with the release mechanism.

- The board has 3 holes that can fit an M3-size bolt
- These holes are located in a T shape. The distance between the top two holes is 30 mm, with the third hole being 36 mm perpendicular to these two holes.
- The board has a length and width of 50 mm, which is within the radius of the engine. The diagonal are 57 mm.
- The board should be placed horizontally to avoid shear forces acting on the components when it lifts off.
- Care must be taken that the vibration and g-forces experienced during launch must not shake loose any of the connectors.
- The board should be mounted chip-upwards, to push the components into the board rather than to pull them out.
- Needless to say, check that bulkheads and other parts above and below the board cannot cause a problem during launch.

7.1 External Interfaces

There are several interfaces between the electronics, the rest of the rocket and the outside world. First, there is the mechanical interface between the electronics and the rocket. This is called the mounting and further discussed in that chapter. Secondly there are electrical interfaces as defined in section 3.3.

On/off switch: The launch procedure requires the rockets to be assembled and activated for some time. Therefore, there needs to be a way to turn off the rocket without disassembling it. This means that an on/off switch is required.

Status LED: The status LED will show whether the electronics is on and should therefore be visible from the outside.

Arm/safe switch: The launch procedure requires the final arming to be done by the safety officer while the rocket is in the tower, so a switch for this should be present.

Breakwire: The breakwire needs a female connector at the outside of the rocket.

Launch Box: If necessary, the rocket can be connected to the launchboxes via a data cable. This allows communication with the rocket while it is in the launch tower.

7.2 Launch procedure

Before further describing integration, it is important to describe the launch procedure. The launch procedure starts about 30 minutes before launch. The rocket should by then fully assembled and the engine already integrated (engine handling performed by designated DARE Safety Officers). The launch procedure starts with turning on the electronics:

1. **Turn on electronics.** A switch for turning on the electronics should be located at the outside of the rocket. This switch should be oriented such that acceleration and airflow during launch does not switch off the electronics.
2. **Insert Rocket into Launch Tower.**

3. **Connect Breakwire.** The breakwire is a small wire bridge that is connected with a rope to the launch tower. This should be connected to a corresponding connector sticking out of the rocket. During the launch the breakwire will disconnect, signaling the rocket that lift-off has occurred.
4. **Run away in a controlled and distinctive manner.** After connecting the breakwire the safety officer will connect the igniter to the launch box. During this procedure all non-essential personnel, including the rocket owner, should have left the launch pad.
5. **Toggle Arm Switch.** The arm switch mechanically connects the pyro with the electronics. It also signals the electronics that it can start looking at the breakwire lift-off detection. This operation will be performed by the safety officer.
6. **Ignite Motor**. Back up at the command post the button to ignite the motor can be pressed. Lift-off will disconnect the breakwire.

A Customization Ideas

Here are a few ideas which you may implement next to the normal SRP electronics functionality. Generally, this is only limited to your creativity, assuming that it's safe. It also gives you bonus points towards the SRP rocket award!

Needless to say, you need to validate any type of sensor to make sure it doesn't generate nonsense and will vote to activate the parachute prematurely. This is a (very) important exercise in real rockets as well.

Accelerometer An accelerometer can be used integrating it over time can give you the velocity. In an ideal case, when the velocity is equal to 0, you've reached apogee! This is not entirely true if you launch at an angle like most SRP rockets do however.

If you integrate the acceleration every loop, you can work out the difference in speed.

$$v = v_0 + at$$

where v is the velocity, v_0 is the original velocity, a is the acceleration and t is the time step. More mathematically, this can be expressed as

$$v = v_0 + \int_{t_1}^{t_2} a dt$$

Pressure Sensor Another idea is to use a pressure sensor, such as the BMP180. These are usually quite accurate, although they might drift a little since the barometric pressure will change over time with the weather.

Aerospace Engineers will be familiar with the following formula from AE1110-I:

$$P_h = P_0 e^{\frac{mgh}{kT}}$$

Pitot Tube Rather than measuring the static pressure, it is a neat idea to try to create a pitot tube to measure Indicated Airspeed (IAS). If you interface it with static pressure, you can also get True Airspeed (TAS).

Black Box You might wish to collect some data on the rocket and make a Black Box (More correctly known as a Flight Data Recorder or a FDR). You could connect it to a GPS for instance. Data could be saved onto an SD card. You might want to compare your sensors (e.g. pressure or accelerometer) and compare their accuracy?

Telemetry A very nice challenge for the more seasoned electronics hobbyist would be an expansion of the electronics system with a (full) telemetry system. Data, video you name it...! Be aware that you need to add systems in the rocket, but also systems on the ground for signal reception and data decoding/recovery. So do not underestimate the effort: it is certainly possible (looking at the past) as long as you draft a good design and planning.

B Part List

Table 3: Part list for the SRP Board.

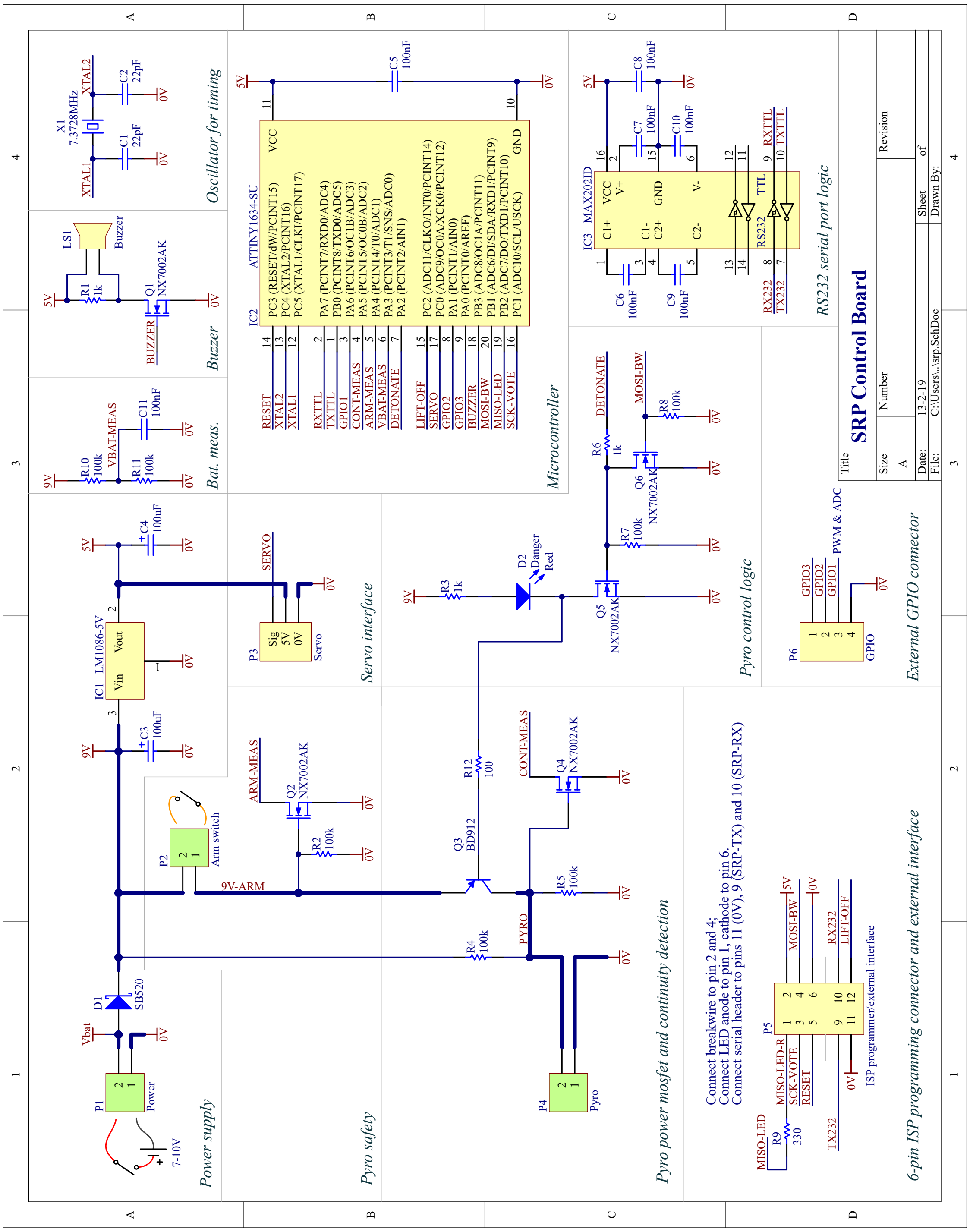
Type	Value	Footprint	Farnell code	Amount	Description
Capacitor	22pF	0805	1414676	2	22pF capacitor
Capacitor	100uF	-	1848439	2	100uF capacitor, radial leaded
Capacitor	100nF	0805	2407344	7	100nF capacitor
Diode	-	-	1467532	1	5A schottky diode
Diode	-	-	2112100	1	Through-hole 3mm diameter red LED
Integrated Circuit	-	DDPAK	1685485	1	5V regulator, 1.5A
Integrated Circuit	-	SOIC20	2443186	1	8-bit AVR Microcontroller, 1.8-5.5V, 16KB Flash, 256 Bytes EEPROM, 1kB SRAM
Integrated Circuit	-	SOIC16	2395901 or 9386378	1	Dual RS232 line driver using 100nF caps for charge pumps
Buzzer	-	-	2135931	1	Piezoelectric buzzer, 14mm diameter, through hole
Connector	-	-	2112482	3	Small 2-pin PCB-mount screw terminal
Connector	-	-	1022248	1	3-Pin servo header
Connector	-	-	1593443 (header), 1593524 (housing), 1593529 (terminal)	1	Header, 6x2-pin, ISP/external interface
Connector	-	-	1022250	1	Header, 4-Pin
Transistor	-	SOT23	2357126 or 2191746	5	Low current NMOS transistor
Transistor	-	TO-220	1084576	1	BD912 -15A -100V PNP transistor
Resistor	1kΩ	0805	2331796	3	1k resistor
Resistor	100kΩ	0805	2331823	7	100k resistor
Resistor	330Ω	0805	2331789	1	330 ohm resistor
Resistor	100Ω	0805	2502669	1	100 ohm resistor
Crystal	7.3728MHz	HC49	2449416	1	7.3728MHz crystal

Further Parts:

- External Switches (2x)
- M3 Screws/Bolts (3x)
- Wires for switches
- Shrinkwrap for switches, breakwire
- Breakwire connector

C Schematics and Render

C.1 SRP PCB Schematics



The text in red on the wires are so-called net labels. Connections with the same net label are physically connected to each other on the PCB too. If you would like to know more, someone from the Electronics Team is surely available for questions.

C.2 PCB Renders Top and Bottom