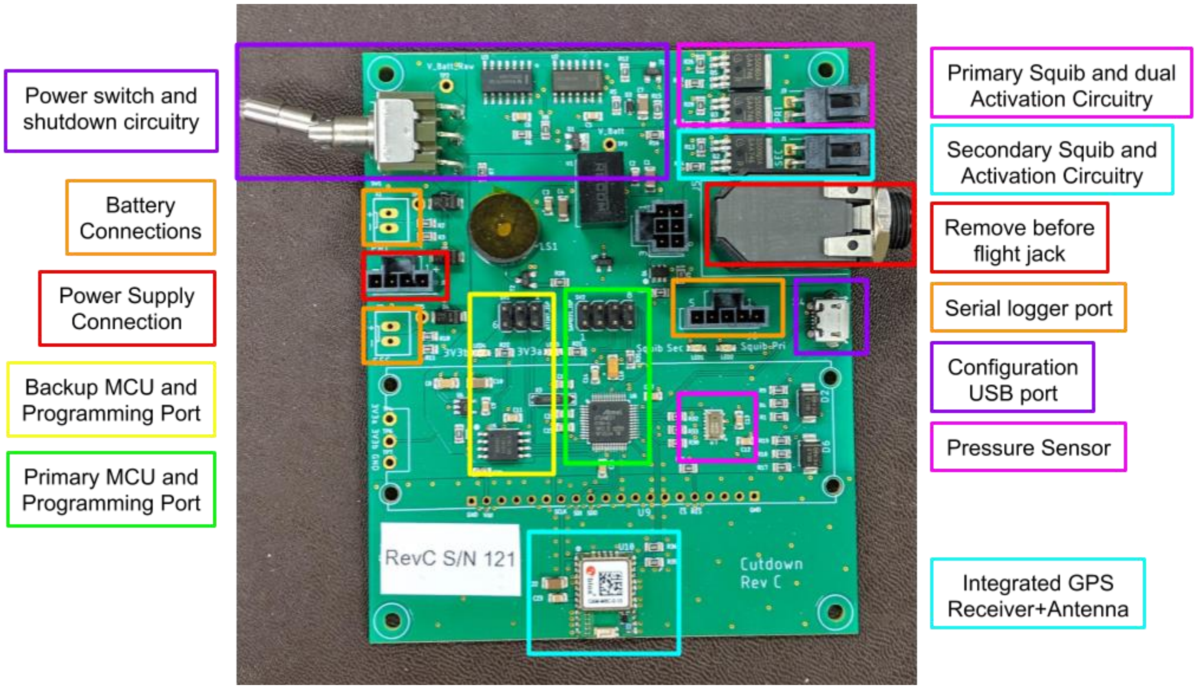
LASP Balloon Cutdown and Parachute Cutaway Controller

Design Engineer: Alex St. Clair



Power switch and shutdown circuitry
Battery Connections
Power Supply Connection
Backup MCU and Programming Port
Primary MCU and Programming Port
Primary Squib and dual Activation Circuitry
Secondary Squib and Activation Circuitry
Remove before flight jack
Serial logger port
Configuration USB port
Pressure Sensor
Integrated GPS Receiver+Antenna
RevC S/N 121
Cutdown Rev C

Figure 1 Functional Overview (with OLED display removed)

Contents

Hardware Overview

Power

The board has three separate power connectors for input power. The voltage range of a 3S (11.1V nominal) LiPo is expected: 10.4 – 12.8 V, though higher voltages are supported. There are two 2-pin headers for batteries (3S, 11.1V nominal LiPo), and one 4-pin nanofit for a power supply. The three power sources are combined via diode-or, which accounts for any difference in voltage. This combined rail runs through the power switch and shutdown circuitry. After the shutdown circuitry, the rail supplies all squib power switches and the redundant 3.3 V converters.

Microcontrollers

There are two microcontrollers: a primary SAMD21 and a backup ATtiny. The primary microcontroller is responsible for all triggers except the backup timer. It has non-volatile storage that contains configurations and that the timer is written to once per minute in case of reboot. The microcontroller interfaces with all of the sensors, the OLED, the buzzer, the squib detection, the battery voltage measurement, and the backup microcontroller. It runs the Adafruit Feather M0 bootloader to allow for USB Arduino programming and serial output. There is an SWD programming header to load the bootloader. The backup microcontroller only runs a backup software timer, and is powered from the backup 3.3 V converter. It runs baremetal software, and there is a SPI programming header.

Squibs

The board supports two squibs: one primary (marked PRI) and one backup (marked SEC). Detection circuitry is present for each squib and is fed to the primary microcontroller. The primary squib has a high-side power switch for each microcontroller so that it can by fired by either. The backup squib only has a high-side power switch for the primary microcontroller.

Remove Before Flight Jack

The remove before flight jack accepts a ¼" stereo jack with the barrel, ring, and tip shorted together. Inserting this plug will short out the squibs to prevent accidental activation on the ground. It also pulls an arming signal low. This signal is redundantly pulled high by both 3.3 V converters and runs to both microcontrollers. Once the plug is removed, both microcontrollers will arm and will ignore any commands until the plug is reinserted, at which point they will disarm. If either microcontroller detects that the arm is pulled low on bootup, they will arm and read stored timer values from EEPROM. This accounts for accidental resets (or watchdog timer resets) during flight.

Configuration

The Micro-USB serial port input to the primary microcontroller is the configuration interface. The primary microcontroller will then set backup microcontroller configurations using their shared SPI interface. The backup microcontroller ignores any configuration commands from the primary microcontroller while the system is armed.

The USB interface can be opened using any standard serial terminal on a support computer, and the baud rate is set to 115200. On bootup and for major events, the microcontroller will automatically output information over this interface. Otherwise it will not output unless it receives a command. There are two commands to know:

- "`MENU`": display the menu of all configuration commands, includes descriptions and units
- "`READ`": display the current non-volatile configurations, includes names and units

The rest of the commands set configuration values, and all of them are of the form "`[CMD],[value]`". A few critical configuration commands are listed in the Software Triggers section.

Software Triggers

Cutdown

The following list contains all of the active triggers in balloon cutdown mode, and their configuration commands over the USB interface in unarmed mode. Brackets denote the position of the new value, and enclose the expected unit.

- Primary microcontroller timer: a number of seconds have passed
    - Command: "`TPRI,[seconds]`"
- Backup microcontroller timer: a number of seconds have passed
    - Command: "`TBCK,[seconds]`"
- Altitude (above Mean Sea Level): the GPS reading exceeds the configured value
    - Command: "`HEIGHT,[km]`"
- Ground distance travelled: the GPS distance from the arming location (calculated using the Haversine formula on latitude and longitude) exceeds the configured value
    - Command: "`DIST,[km]`"
- Burst: two sequential GPS altitude differences exceed the configured fall rate
    - Command: "`BURST,[m/s]`"
- Sink: a configurable number of sequential GPS altitude differences show that the altitude is decreasing
    - Command: "`SINK,[number of samples]`"

Cutaway

The following list contains all of the active triggers in parachute cutaway mode, and their configuration commands over the USB interface in unarmed mode. Brackets denote the position of the new value, and enclose the expected unit.

- <u>Primary microcontroller timer</u>: a number of seconds have passed
  - Command: “`TPRI,[seconds]`”
- <u>Backup microcontroller timer</u>: a number of seconds have passed
  - Command: “`TBCK,[seconds]`”
- <u>Pressure trigger</u>: a pressure reading below the configured pressure value (cutaway ceiling) has been recorded, and the last ten pressure readings have been above the configured value and within 1 hPa of each other, suggesting that the payload has landed on the ground.
  - Command: “`CEIL,[hPa]`”
- <u>Low altitude timer</u>: after the balloon drops back below the cutdown ceiling (the pressure goes back above the value), a tighter timer starts.
  - Command: “`TLOWA,[seconds]`”

Flight Procedures

Day Prior to Flight

1. Charge the batteries
2. Set all configurations for the anticipated flight profile and for cutdown/cutaway mode
   i. Power the system on with the remove before flight (RBF) plug inserted
   ii. Open a serial terminal with the baud rate set to 115200
   iii. Use the “`READ`” command to see current configurations
   iv. Use the “`MENU`” command to see a command list
3. Plug the squib into the PRI squib connector <u>with the RBF plug inserted</u>
   i. If a secondary squib is to be used, insert it into the SEC connector
4. Power the system on
   i. Check that all displayed values are expected
   ii. Unplug the RBF connector
   iii. Check that both squibs register as “OK” (this can only be done when the squibs aren’t shorted out by the RBF plug)
   iv. Plug in the RBF connector
5. Power the system off and unplug the batteries
6. Attach the system to the load line

Pre-Rollout

1. Make sure that the RBF plug is inserted
2. Plug in the batteries

3. Power the system on
4. Check the displayed values
5. Tape the lid shut

Launch (Cutaway)

1. Check that the pressure reading is within the expected range for the launch altitude
2. Within five minutes of launch, remove the RBF plug
    a. Check that both timers are counting down
    b. Ensure that the display reads ready for flight. If not, check all individual display values.
3. Launch

Launch (Cutdown)

1. Within five minutes of launch, remove the RBF plug
    a. The primary timer will not start until a GPS lock is acquired
    b. Check that the backup timer is counting down
    c. If a GPS lock is not required within five minutes, reinsert the RBF plug and restart the procedure
2. Once a GPS lock is acquired:
    a. Ensure that the current distance is approximately zero
    b. Ensure that the current height is the expected altitude
    c. Ensure that the primary timer is counting down
    d. Ensure that the display reads ready for flight. If not, check all individual display values
3. Launch

Important Considerations

- Unplug the batteries for storage. When the power is switched off with the batteries plugged in, the system will still draw as much as 250 µA due to the battery voltage sense circuits.
- Be careful leaving the system armed—it will completely run down the batteries if it thinks it still needs to fire a squib.
- Be sure that the pressure value for the cutdown ceiling is below the pressure of the launch site and expected landing site.
- The temperature sensor is not very accurate, but should regulate well around 0° C.
- Unless in armed or firing mode, the system will shut down if both batteries fall below the critical battery voltage configuration. Choose this value carefully.

Display Information

Unarmed

In unarmed mode, the display cycles through a full set of system status information. It is important to note that it displays information for both cutdown and cutaway. The information is as follows.

- Primary timer
  - `PRI: [time remaining h/m/s]`
- Backup timer
  - `BCK: [time remaining h/m/s]`
- Low altitude timer
  - `TLOWA: [time remaining h/m/s]`
- Primary squib status
  - `PRI squib [OK/error!]`
- Backup squib status
  - `BCK squib [OK/error!]`
- GPS solution age
  - `GPS age: [time (s)] / No GPS solution`
- Horizontal distance trigger set point
  - `SetD: [distance (km)]`
- Current horizontal distance
  - `CurD: [distance (km)] / Origin not set`
- Altitude trigger (above MSL) set point
  - `SetH: [height (km)]`
- Current altitude (above MSL)
  - `CurH: [height (km)] / No GPS solution`
- Ambient pressure
  - `P: [pressure (hPa)]`
- Primary battery voltage
  - `PRI: [voltage (V)]`
- Backup battery voltage
  - `BCK: [voltage (V)]`
- Temperature
  - `[temperature (C)]`

Armed (Cutdown)

In armed mode, the first line displays exclusively the status of the board, either “`CUTDOWN READY`” or “`CUTDOWN FAULT`”. If the board is ready, it has two screens for the second line:

- Current GPS height and distance
  - `H:[height in km] D:[distance in km]`
- Timer and backup timer
  - `T:[primary timer (s)] BT:[backup timer (s)]`

In the case of a fault, there are three screens to show the status of each critical element:

- GPS and squibs
  - `GPS:[OK/FT] SQB:[OK/FT]`
- Timer and backup timer
  - `TPRI:[OK/FT] TBCK:[OK/FT]`
- Battery voltage and temperature
  - `BATT:[OK/FT] TEMP:[OK/FT]`

Armed (Cutaway)

In armed mode, the first line displays exclusively the status of the board, either “`CUTAWAY READY`” or “`CUTAWAY FAULT`”. If the board is ready, it has two screens for the second line:

- Pressure
  - `P:[pressure] hPa`
- Timer and backup timer
  - `T:[primary timer (s)] BT:[backup timer (s)]`

In the case of a fault, there are three screens to show the status of each critical element:

- Pressure and squibs
  - `PRESS:[OK/FT] SQB:[OK/FT]`
- Timer and backup timer
  - `TPRI:[OK/FT] TBCK:[OK/FT]`
- Battery voltage and temperature
  - `BATT:[OK/FT] TEMP:[OK/FT]`

Versioning

Hardware

Three hardware revisions were made: A, B, and C. Rev A is no longer fully supported by software due to pinout and interface changes. Revs B and C are electrically identical, with the only changes from B to C being mechanical and manufacturing related. The following public GitHub repository contains all three revisions. https://github.com/dastcvi/cutdown_hardware

Software

Public GitHub repositories were used for development of both the primary microcontroller software (in the form of an Arduino library) and the backup microcontroller software. These libraries are tagged with major releases, and contain branches for each hardware revision. The ‘master’ branch contains the most up-to-date release. The primary microcontroller software can be found at https://github.com/dastcvi/cutdown. The backup microcontroller software can be found at https://github.com/dastcvi/cutdown_backup.

Re-Flashing the Software

For the purposes of re-flashing the software, it is safe to assume that the bootloader is properly set up. In case of any bootloader issues, follow this tutorial: https://learn.adafruit.com/proper-step-debugging-atsamd21-arduino-zero-m0/restoring-bootloader.

Arduino Setup

- Install Arduino: I have been using Arduino 1.8.4 on Windows 10.
- In the Boards Manager (Tools > Board: “xxx” > Boards Manager…) install “Arduino SAMD Boards” and “Adafruit SAMD Boards”.
- Because the cutdown uses SERCOM5, and the board support package uses it as well, we need to alter the board support package to free it up
    - https://learn.adafruit.com/using-atsamd21-sercom-to-add-more-spi-i2c-serial-ports/muxing-it-up#freeing-up-sercom5-2-32
- Select the Adafruit Feather M0 (Tools > Board: “xxx” > Adafruit Feather M0).
- The Cutdown library must be placed in the Arduino Libraries folder.

Required Third-Party Libraries

- Adafruit_MPL3115A2: runs the pressure sensor
    - https://github.com/adafruit/Adafruit_MPL3115A2_Library
- Adafruit_SleepyDog: SAMD21 watchdog timer
    - https://github.com/adafruit/Adafruit_SleepyDog
- Adafruit_ZeroTimer: SAMD21 timer peripheral
    - https://github.com/adafruit/Adafruit_ZeroTimer
- FlashStorage: use flash on SAMD21 for non-volatile configurations
    - https://github.com/cmaglie/FlashStorage

Flashing Procedure

1. Ensure that a remove before flight jack is inserted (this is **critically** important).
2. Open and compile “cutdown_main.ino” in the Arduino IDE for the Adafruit Feather M0
    a. Ideally, change the serial number in the library’s “Cutdown_Commission.cpp” to be the board number you're flashing.

3. Power the cutdown on, insert the Micro USB cable, and select the board’s port in Arduino (Tools > Port).
4. Upload the code and immediately open up the serial monitor to look at the output.
5. The commissioning code should run and should:
    a. Print: "Config error, entering commission mode!"
    b. Proceed to write all of the configurations, displaying the status as it does so, and all of the written values. (20+ lines of serial info)
    c. Run some diagnostic tests using the onboard sensors. The OLED should display the alphabet. (20+ more lines of serial info)
    d. Enter an infinite loop and wait for a power cycle before it runs the normal code.
6. Restart the board and re-open a serial monitor. The cutdown should show unarmed.
7. Try running a “`READ`” command and making sure the configuration values are correct.