Minnesota Space Grant Consortium

UNIVERSITY OF MINNESOTA

Uplink, Downlink, and How NOT to Vent a High-Altitude Balloon

Adviser: James Flaten, MN Space Grant (at U of MN – Twin Cities)

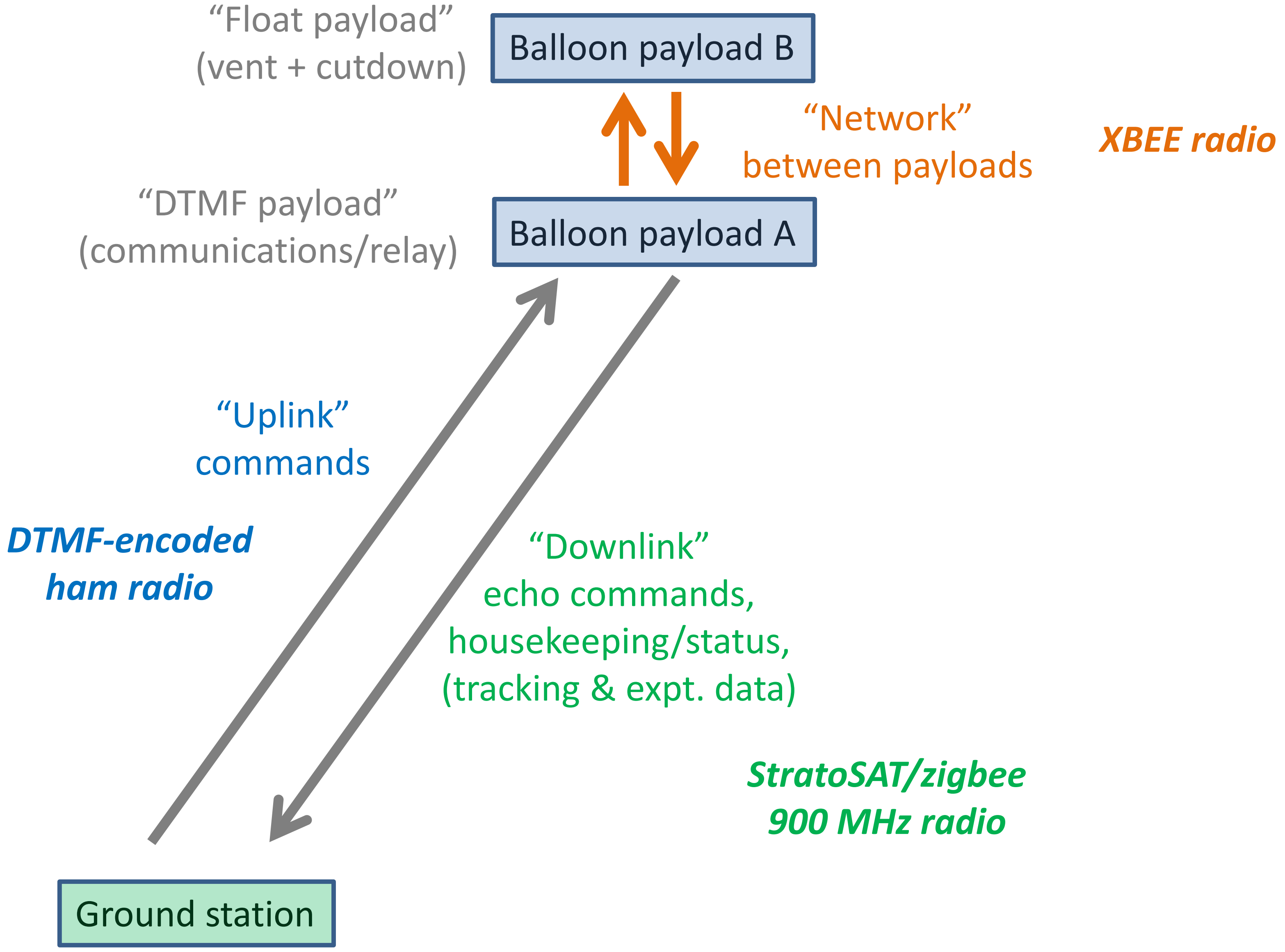
U of MN students: Seth Frick, Alex Ngure,
Hannah Weiher, John Jackson, and Monique Hladun

Abstract:

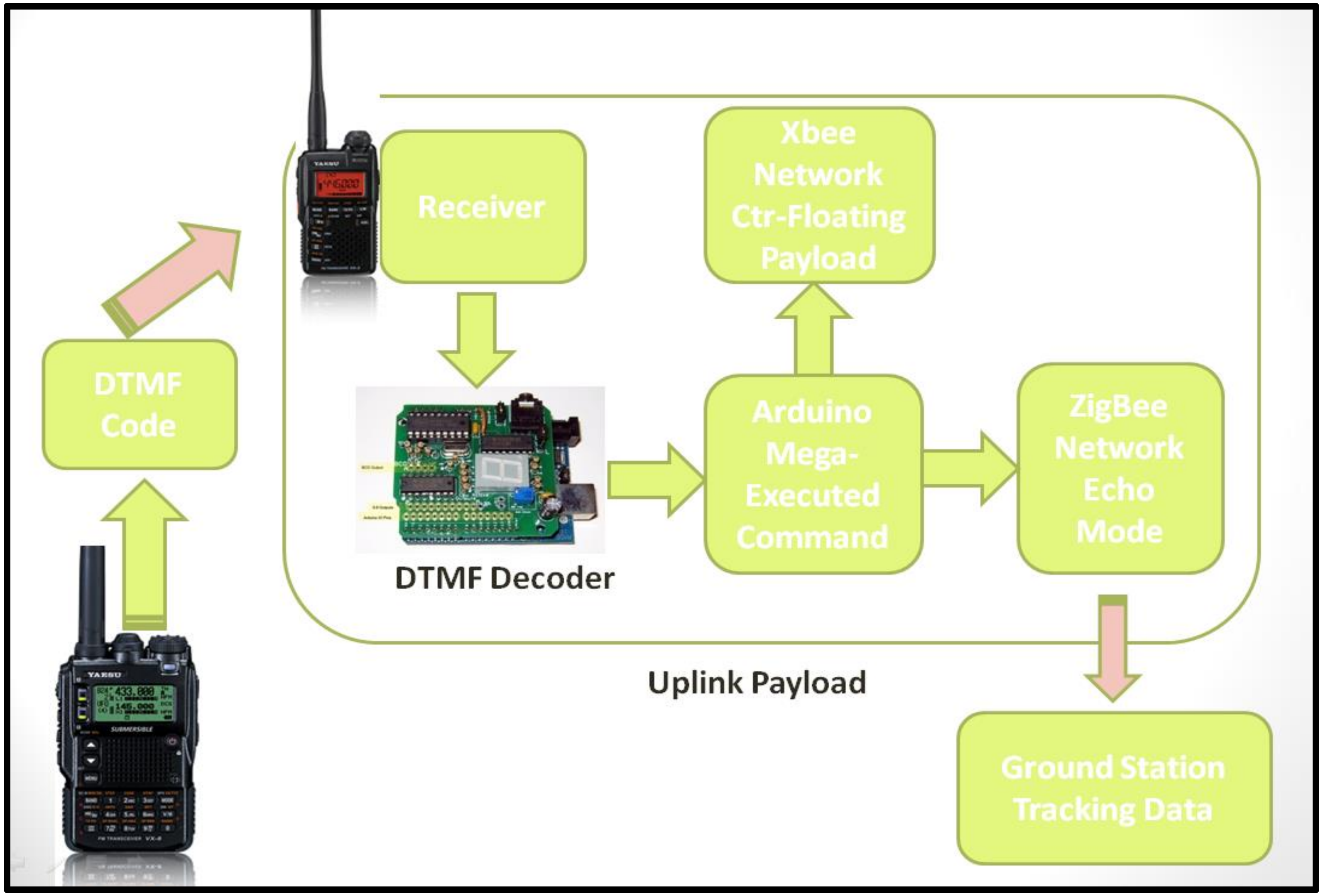
We uplink commands to high-altitude balloon missions using DTMF encoding on ham radio frequencies, interpreted by an Arduino microcontroller with a DTMF decoder shield. The commands are then circulated between multiple payloads using an XBee radio network. We echo uplink transmissions back to the ground, as confirmation of receipt, and downlink other housekeeping data using a Stratostar Zigbee monitoring Arduino output pins. We have also developed a radio-controllable vent mechanism that attaches directly to the neck of a weather balloon to vent helium in flight in an attempt to prolong time spent at altitude. This final project, controlled from the ground through the DTMF uplink and Xbee network (with on-board autonomous back-up capabilities as well), has been problematic but we will report on progress and results to date.

Minnesota Space Grant Consortium NASA UNIVERSITY OF MINNESOTA

The big picture.



Uplink


DTMF Code
Receiver
Xbee Network Ctr-Floating Payload
Arduino Mega-Executed Command
ZigBee Network Echo Mode
DTMF Decoder
Uplink Payload
Ground Station Tracking Data

Uplink of commands by DTMF.

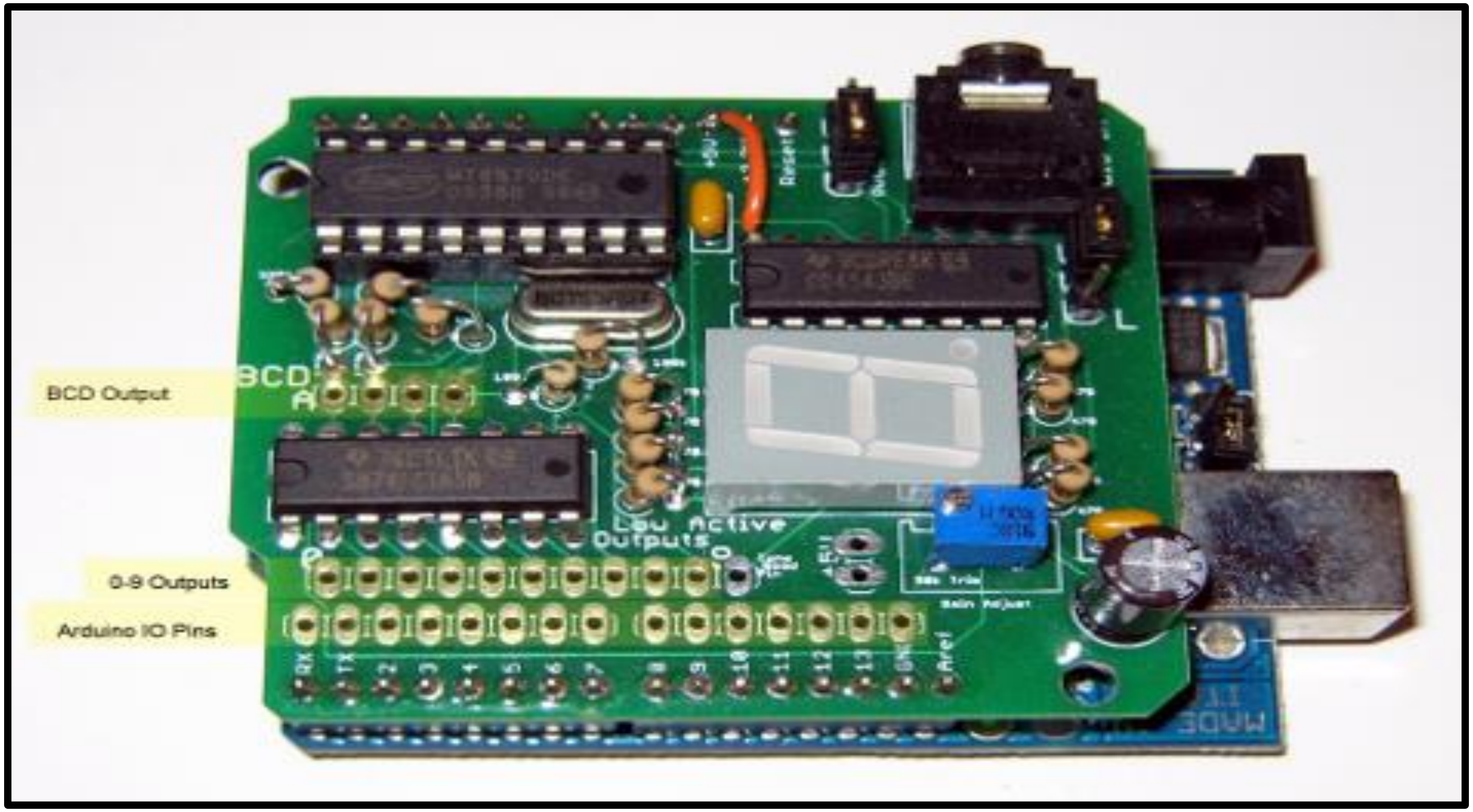
Dual-tone multi-frequency (DTMF) signaling (AKA “Touch-Tone” when used in telephones)

DTMF keypad frequencies

	1209 Hz	1336 Hz	1477 Hz	1633 Hz
697 Hz	1	2	3	A
770 Hz	4	5	6	B
852 Hz	7	8	9	C
941 Hz	*	0	#	D

Ref: http://en.wikipedia.org/wiki/Dual-tone_multi-frequency_signaling

DTMF decoder (Arduino shield)



Ref: http://w2mh.wordpress.com/2010/06/03/dtmf-shield/

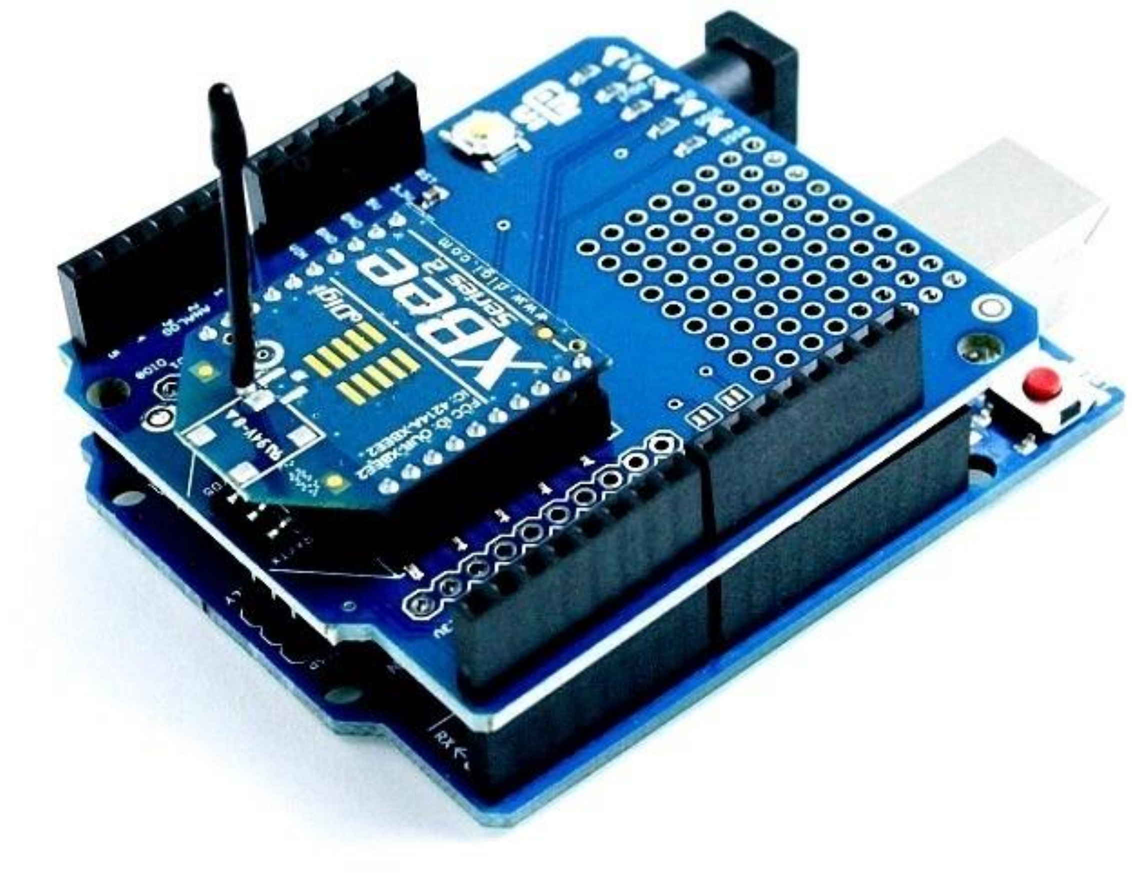
Downlink

Downlink by StratoStar zigbee $\rightarrow$ command pod. Approximate analog values with Arduino digital PWM (pulse-width modulation) and filter.

Network

Minnesota Space Grant Consortium

University of Minnesota



Inter-payload communication by XBEE network.
(Only implemented uplink part of this (so far)).

Minnesota Space Grant Consortium NASA UNIVERSITY OF MINNESOTA

The “Float” project

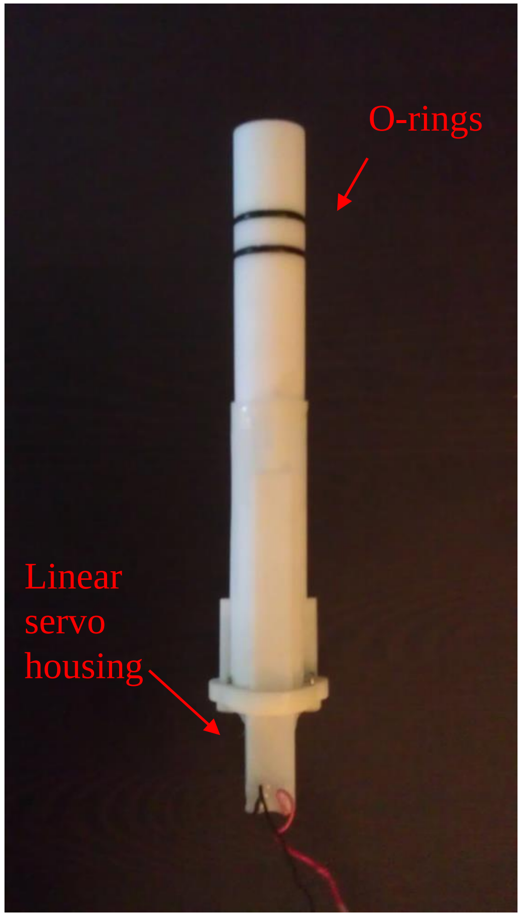
We want to be able to “float” balloons (i.e. keep them from ascending all the way to burst) because many experiments benefit from additional time at altitude.

Notes:

- Achieving float requires dumping excess lift, possibly by venting lifting gas and/or by cutting away a secondary excess-lift balloon(s), while in flight.
- Venting gas from a standard latex balloon requires not sealing the neck but instead inserting a “vent” mechanism in that neck that can be opened and closed in flight.
- Neck constriction (and the fact that helium wants to go up, not down) can make venting a slow process.
- Warning: Extending flight time will mean a longer chase (time-wise) and quite possibly a longer chase (distance-wise) as well.
- If you disrupt the standard go-to-altitude-then-burst mechanism for flight termin-ation then you need to provide an alternative **robust** flight termination mechanism.
- A cut-down can sever payloads, but recovering the vent from the neck can be hard.

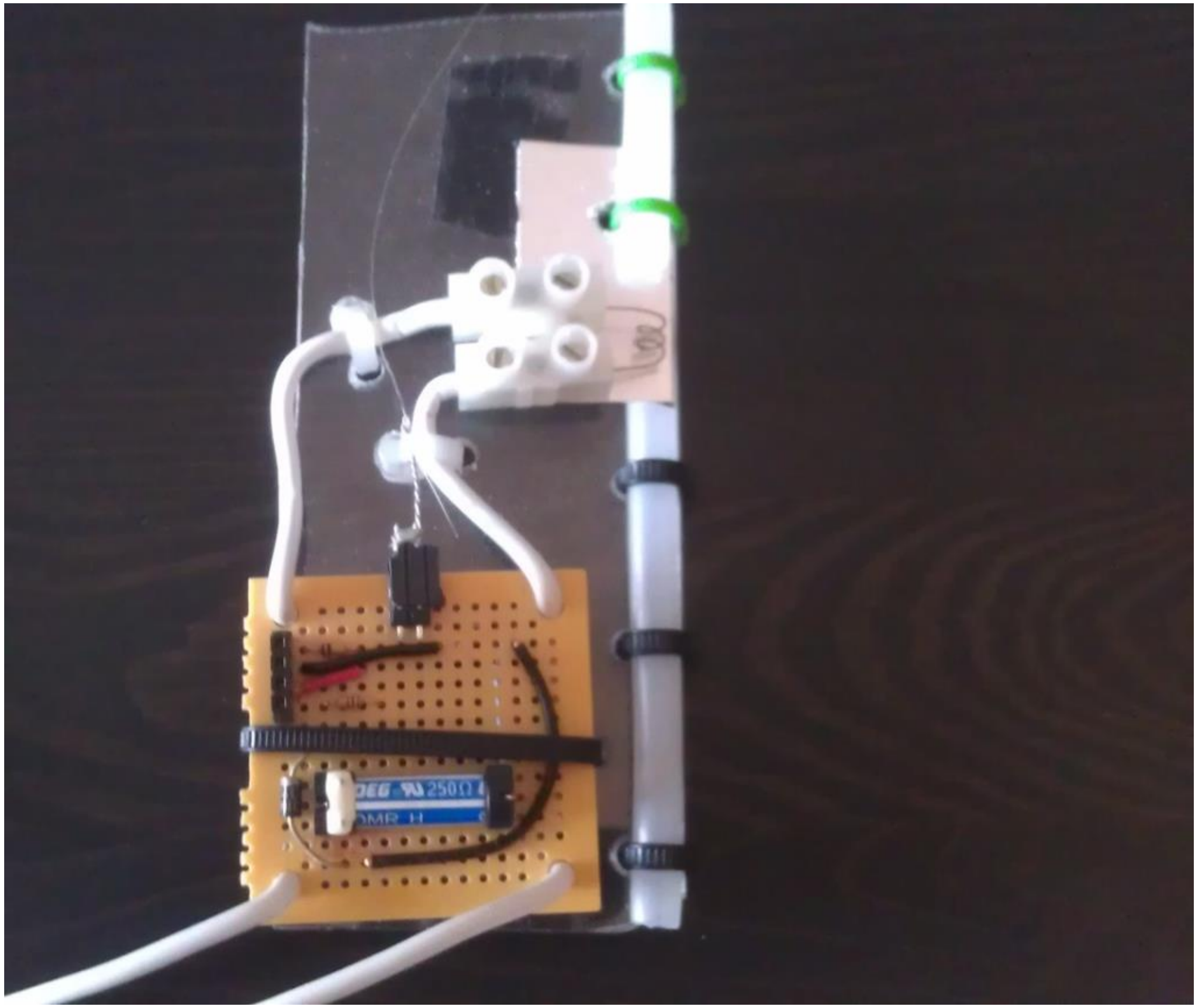
Project goals:

- Our aim is to command venting events using uplink from the ground and to have on-board microcrocontrollers to autonomously execute a mission profile if contact with the ground is not maintained.
- Downlink is used to keep the ground informed about venting status: vent open or closed, cumulative venting time, payload successfully cut down or not, etc.

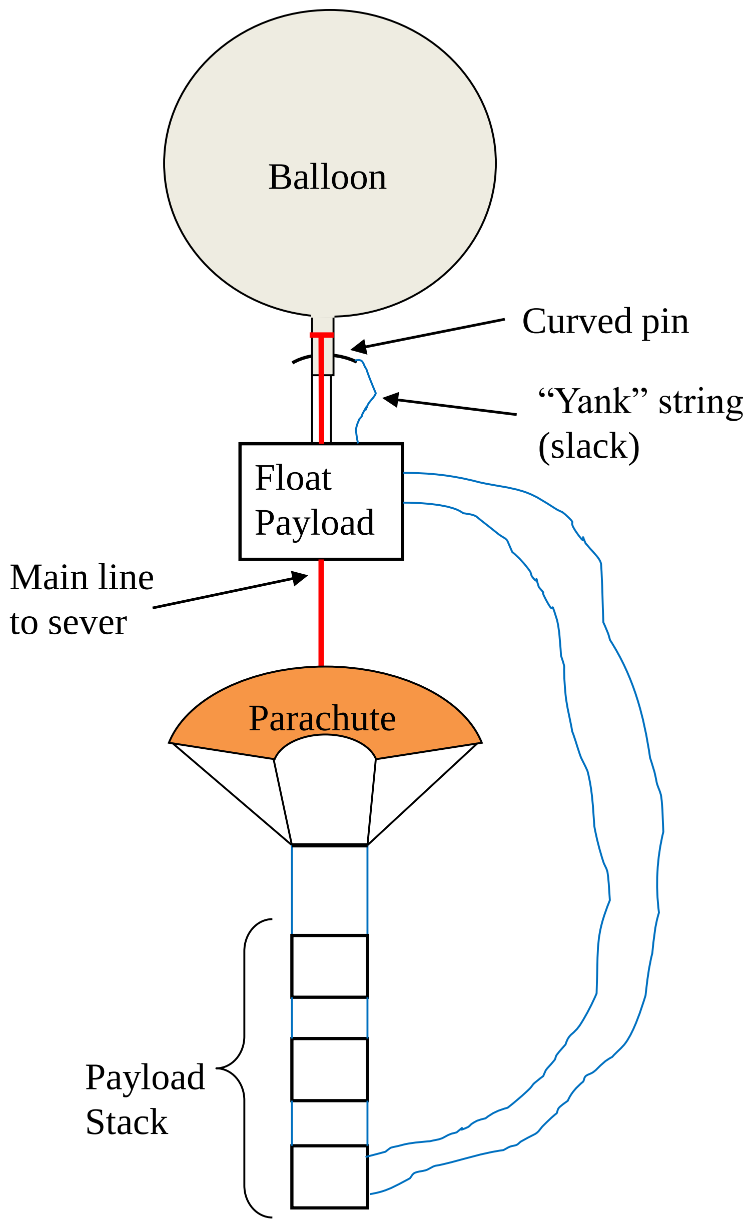


The Firgelli PQ12 linear servo at full extension (left) and the float mechanism main body (right) are shown.

Minnesota Space Grant Consortium NASA UNIVERSITY OF MINNESOTA

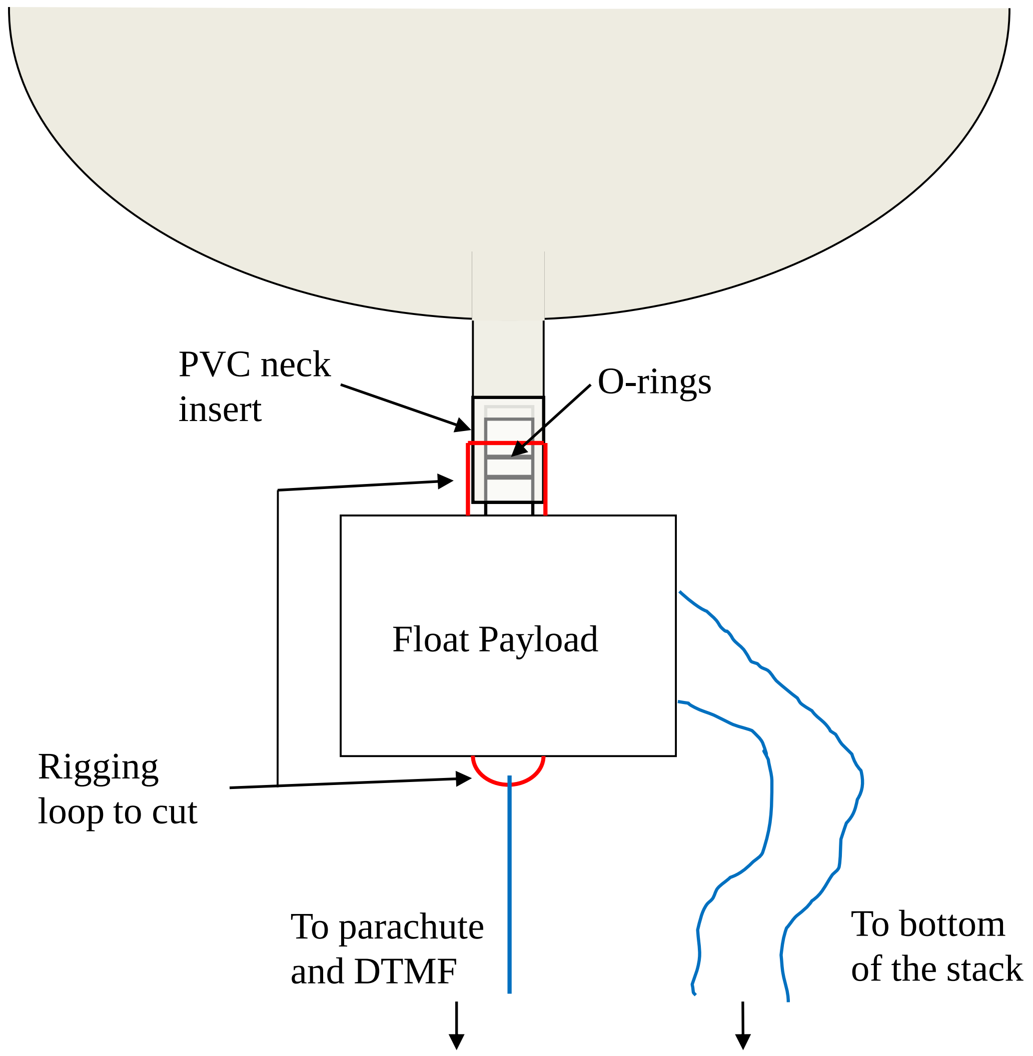


One of two nichrome burner cut-down mechanisms. Relay is controlled by local Arduino (commanded through XBEE with autonomous back-up). Fishing line is a separation-check pull-pin.

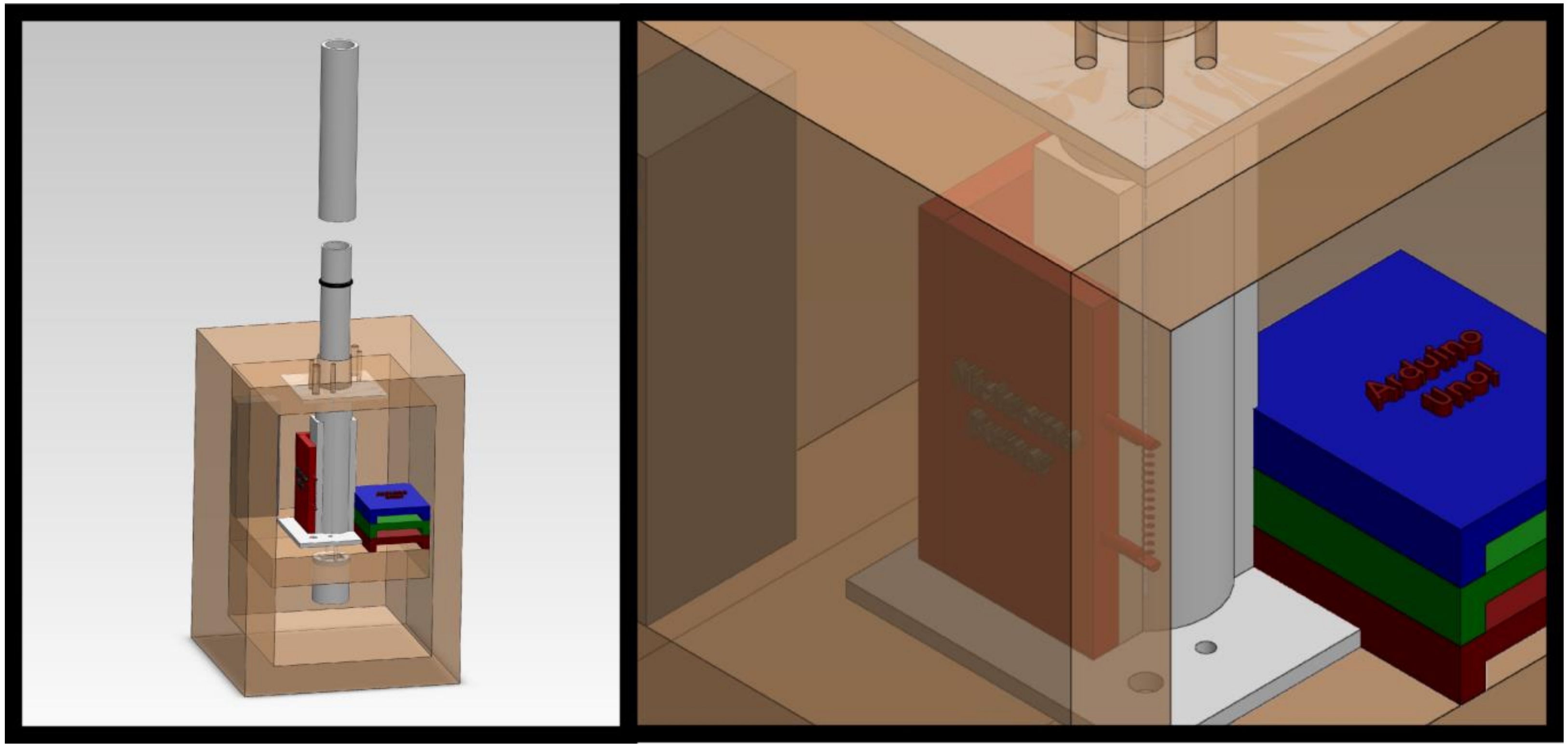


The "pull-pin" design for extracting the Float payload from the balloon neck (parts not to scale)

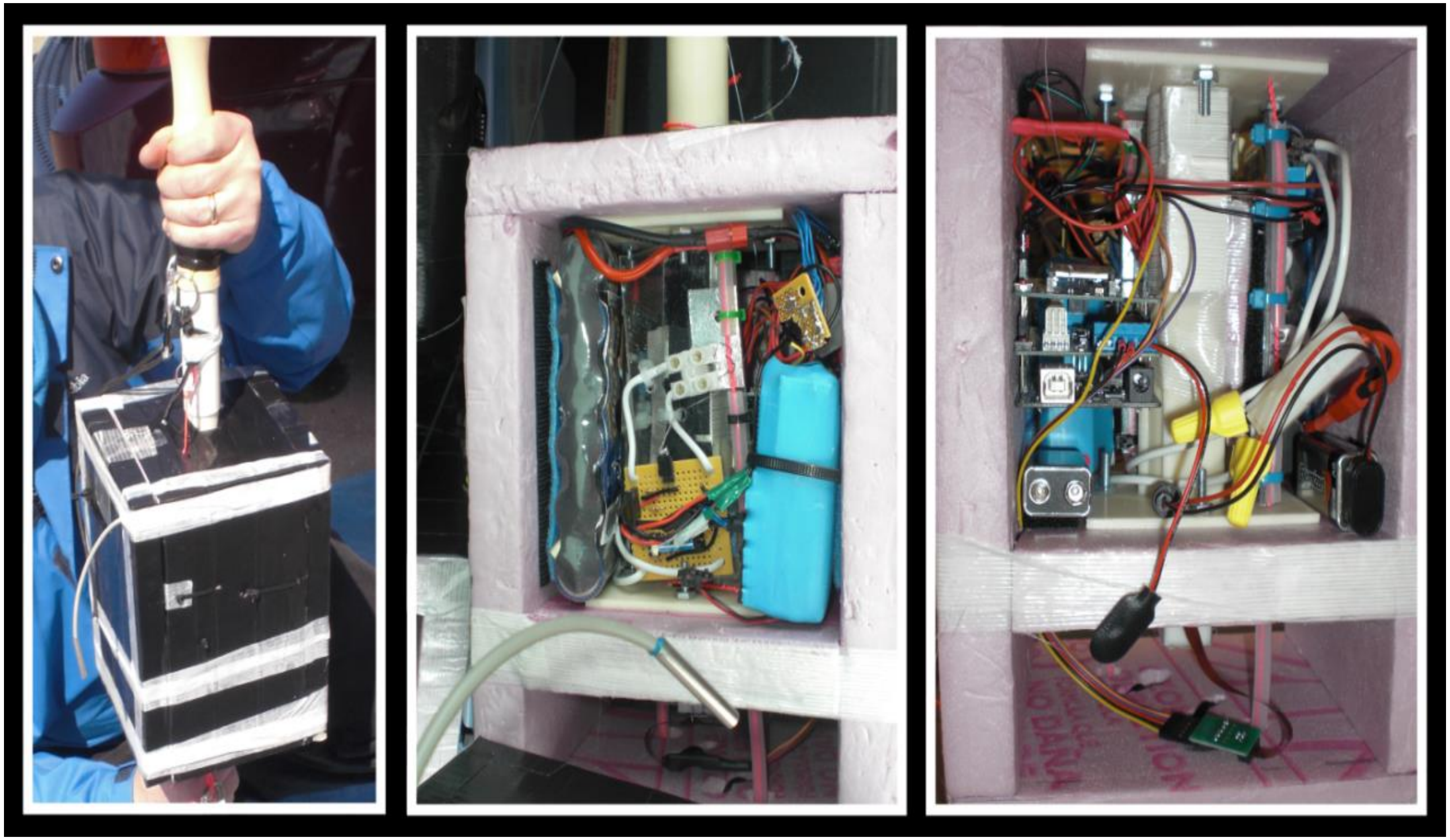
The "o-ring" design for extracting the Float payload from the balloon neck (parts not to scale)



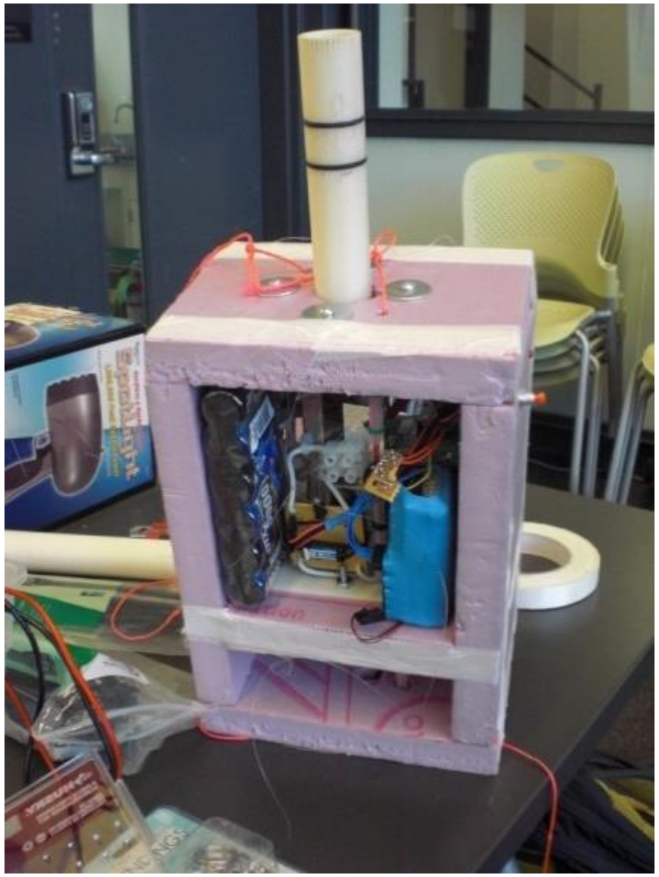
Minnesota Space Grant Consortium NASA UNIVERSITY OF MINNESOTA



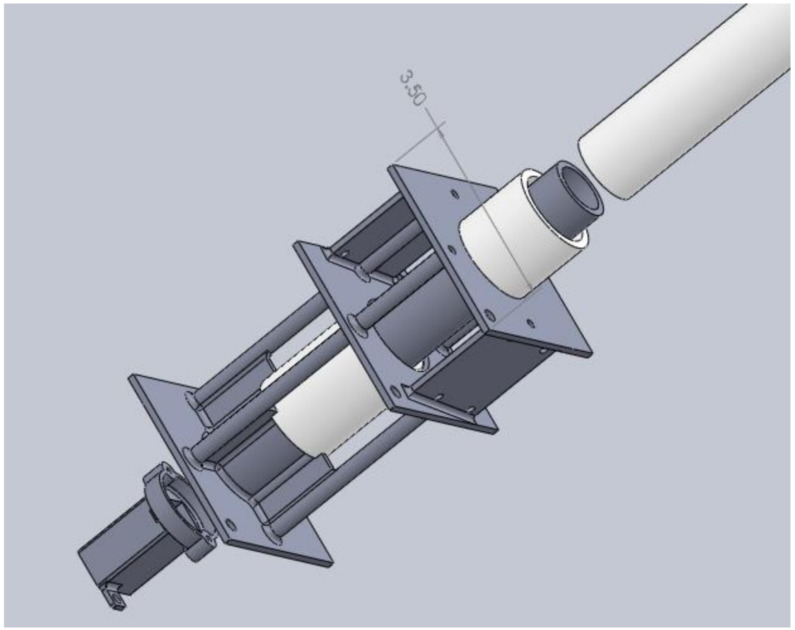
Two views of the CAD model of the Float: vent & cutdown payload with Arduino Uno, XBEE shield, motor shield and linear servo, nichrome burners (2), and battery pack.

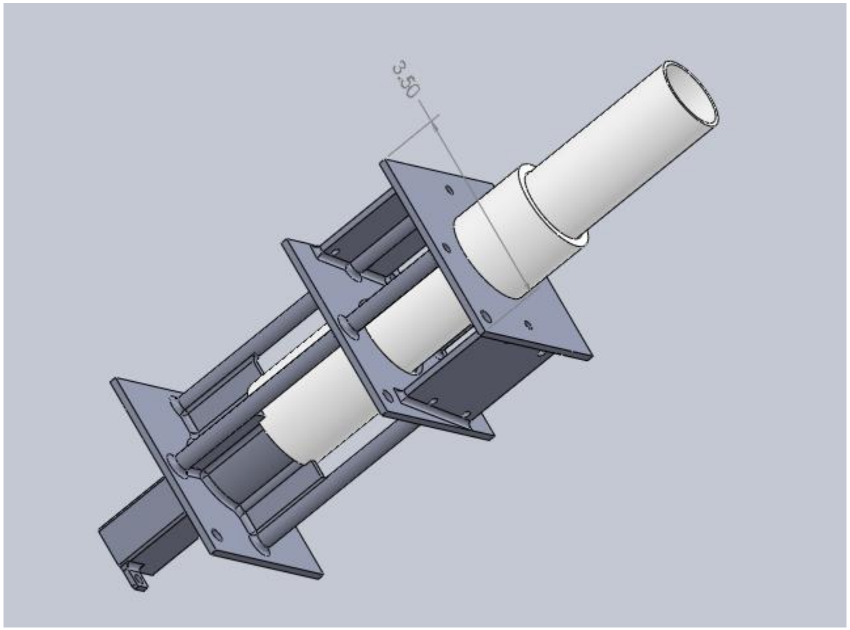


(left) A view of the Float payload rigged to a balloon
(center) Rear view of the open payload showing one nichrome burner, main battery pack (left), and StratoStar zigbee radio (right)
(right) Front view of the payload showing the Arduino stack (left) and the vent tube (center) with linear servo (at bottom of tube; partially hidden)



The Float payload during a venting test in which two pounds of excess lift were vented in under 5 minutes. The vent was successfully opened and closed by the DTMF/XBEE uplink. Vent status was monitored zigbee radio.





After breaking off the exposed neck of the vent column (twice!) where it was weakest (due to the o-ring grooves) the vent mechanism core was redesigned to support the tube in the neck of the balloon on both the inside and the outside simultaneously. Gray parts are 3-D printed. White parts are standard pvc tubing.

DTMF - Flost command set (early version – 1's reserved):

248: open vent for 1 second then close vent
289: open vent and close immediately
327: add 1 hour to 4-hour failsafe timer
369: open vent indefinitely
555: test the LED pin on the exterior of the payload
867: close vent indefinitely
999: fire the cutdowns

Note: All commands are echoed by the DTMF payload over the Xbee network, but the Float only responded to the ones listed. Other (dummy) commands were issued to test the downlink echo.

Note: Numerically, codes were selected which were different enough to be resolvable after being echoed through the downlink system.

DTMF command set (later version – more autonomous capabilities)

-2XX: open float valve for XX seconds, then close automatically.
-302: close float valve indefinitely.
-382: open float valve indefinitely.
-4XX: reserved (for competition payload)
-555: LED flash on Float payload.
-627: add one hour to fail-safe counters (both on DTMF payload and Float payload, assuming XBee link is up).
-678: hibernate (DTMF payload fail-safes inactive). Any new code received leaves hibernation.
-767: turn siren off.
-787: turn siren on.
-822: fire cut-down with pull-pin check.
-888: fire cut-down with no pull-pin check.
-940: subtract 5,000 feet from autopilot cut-down altitude.
-949: add 5,000 feet to autopilot cut-down altitude.
-990: reengage autopilot.
-999: disengage autopilot.

Autopilot configuration commands for Version 7a:
-900: subtract 5,000 feet from altitude at which autopilot opens float valve.
-909: add 5,000 feet to altitude at which autopilot opens float valve.
-920: subtract 100 ft/min from autopilot target ascent rate.
-929: add 100 ft/min to autopilot target ascent rate.
-960: subtract 5 minutes from autopilot float time.
-969: add 5 minutes to autopilot float time.

Autopilot configuration commands for Version 7b:
-900: set float valve open time for float tests to 30 seconds.
-909: set float valve open time for float tests to 1 minute.
-920: set float valve open time for float tests to 2 minutes.
-929: set float valve open time for float tests to 5 minutes.
-960: subtract 2,000 feet from altitude interval for float tests.
-969: add 2,000 feet to altitude interval for float tests.

Six zigbee downlink channels from the DTMF payload (late version):

1. Echo of the last DTMF code received
2. The number of DTMF digits received (to identify partial codes) and a special code to indicate a timeout monitor was called (if 3 DTMF tones were not received in close succession)

3. Health status: payload internal temperature (10 x) in degrees Celsius

4. Mission parameter: altitude at which autopilot venting will begin (in feet/100) OR altitude interval for float testing (in feet/100)
5. Mission parameter: Targeted ascent rate for autopilot venting (feet/min) OR float valve open duration during each float test (sec)
6. Mission parameter: Altitude at which autopilot automatically fires the cut-downs (feet/100) (i.e. target peak altitude)

Four zigbee downlink channels from the Float payload (late version) to report on status of the vent and the cut-downs:

Note: It is important to distinguish between what servo is commanded to do and whether or not the vent actually opened.
Note: The downlink updates only about every 30 seconds, so actions that complete in less than 30 seconds might be missed by the downlink system.

1. Cumulative (commanded) open time for float mechanism (min*10)
2. Number of times each cutdown had been called (default is zero), plus indication of successful separation (from pull-pin status).
3. Float push-button status – independent from servo potentiometer and from Arduino commanding
4. Actual vent position (open, closed, mid-way) according to potentiometer on the servo.

The "Failsafe" Timer

The Arduinos in the DTMF and Float payloads each monitored a "failsafe" timer to fire the cutdowns if too much time elapsed since turn-on. The default setting was 4 hours but this could be extended in 1-hour increments by DTMF commands, assuming the uplink was working, if additional time was desired. This might happen in the case of a launch delay, as opposed to a genuine extension of the time in flight – early flights were for proof-of-concept and were not planned to have extended float periods.

The Results (from multiple flights to test various aspects of the design):

- We inadvertently vented two balloons during ascent, while trying to perfect our connection to (and release from) the balloon neck. Hence we know that venting through the neck works, at least at low altitudes.
- DTMF tones could be transmitted (and correctly echoed) to balloons at altitude (though not always during post-burst chaos).
- The DTMF payload was finicky and had no reset capability. When it failed to relay ground commands and send along autonomous instructions the float wasn't commanded, so balloons simply burst as normal.
- We just missed testing the cut-down on two flights due to premature bursts.
- Most mysterious of all, on our last flight (last summer) we opened the vent at 60,000 ft (and got confirmation of that by downlink) but THE ASCENT RATE DIDN'T CHANGE AT ALL for 20 minutes, then burst!

Hence the title: "How **NOT** to vent a balloon."

At this conference we are looking for feedback and advice about our approach from other groups who have implemented venting solutions.

(Questionable?) Assumptions

- If you have x pounds of excess lift at launch, that is how much you have at altitude. (i.e. that is how much excess lift you need to vent to achieve neutral buoyancy)
- (A related statement, pertaining to a different way of dumping excess lift.) If you have an "excess-lift balloon" at launch, cutting it away at altitude will achieve neutral buoyancy.
- The efficiency of venting (i.e. the time it takes to vent a certain amount of excess lift) at altitude is similar to that on the ground.
- If an ascending balloon switches to neutral buoyancy at altitude there is adequate air friction to bring it to rest.

Potential future tests and future directions for hardware.

- We have built another DTMF payload with an encoder/decoder shield and remote-reset capability which we hope will be less finicky and will allow us to get away from StratoSAT downlink (which is overkill for this project).
- The Float project has been on hold (in hopes of getting feedback at this conference). We plan to move more autonomous capabilities to the Float Arduino (from the DTMF Arduino) so the Float can deal better with potential communication failures.
- In the absence of the ability to measure the gas flow through the vent (I'm interested in anyone has ideas about how to do that), we expect to try some flights where we vent repeatedly at different altitudes to try to characterize the efficiency of our venting mechanism. If high-altitude venting proves to be intolerably inefficient, we may need to start over.

I have both our DTMF payload and our Float payload with me (and will also have them at the poster session this evening), if anyone wants to look at them up close.

I welcome feedback on this project and advice about how we might push it forward.

THE END