

2015

**The Melbourne Amateur Radio And Technology Group (MARTG)
Entry Into The
2015 Global Space Balloon Challenge (GSBC)**

By

Julie VK3FOWL and Joe VK3YSP

Topics

- Global Space Balloon Challenge
- CASA Regulations
- HAB Flight Profile
- Hardware & Software Development
- HAB Flight Simulation
- Launch, Tracking & Recovery Operations
- Observations and Conclusions
- Melbourne Amateur Radio And Technology Group

Global Space Balloon Challenge (GSBC)

- “Where people around the world can simultaneously fly high altitude balloons celebrating an age where anyone can reach the edge of space.”
- Promotes community, education and innovation
- In 2015 there were 298 teams in 47 countries



Civil Aviation Safety Authority Regulations

small balloon, light balloon, medium balloon and heavy balloon.

small balloon means a free balloon that can carry no more than 50 grams of payload. No approval is required.

light balloon means a free balloon that:

- (a) is no more than 2 metres in diameter at any time during its flight; and
- (b) can carry no more than 4 kilograms of payload.

Information to be provided to CASA if approval is required

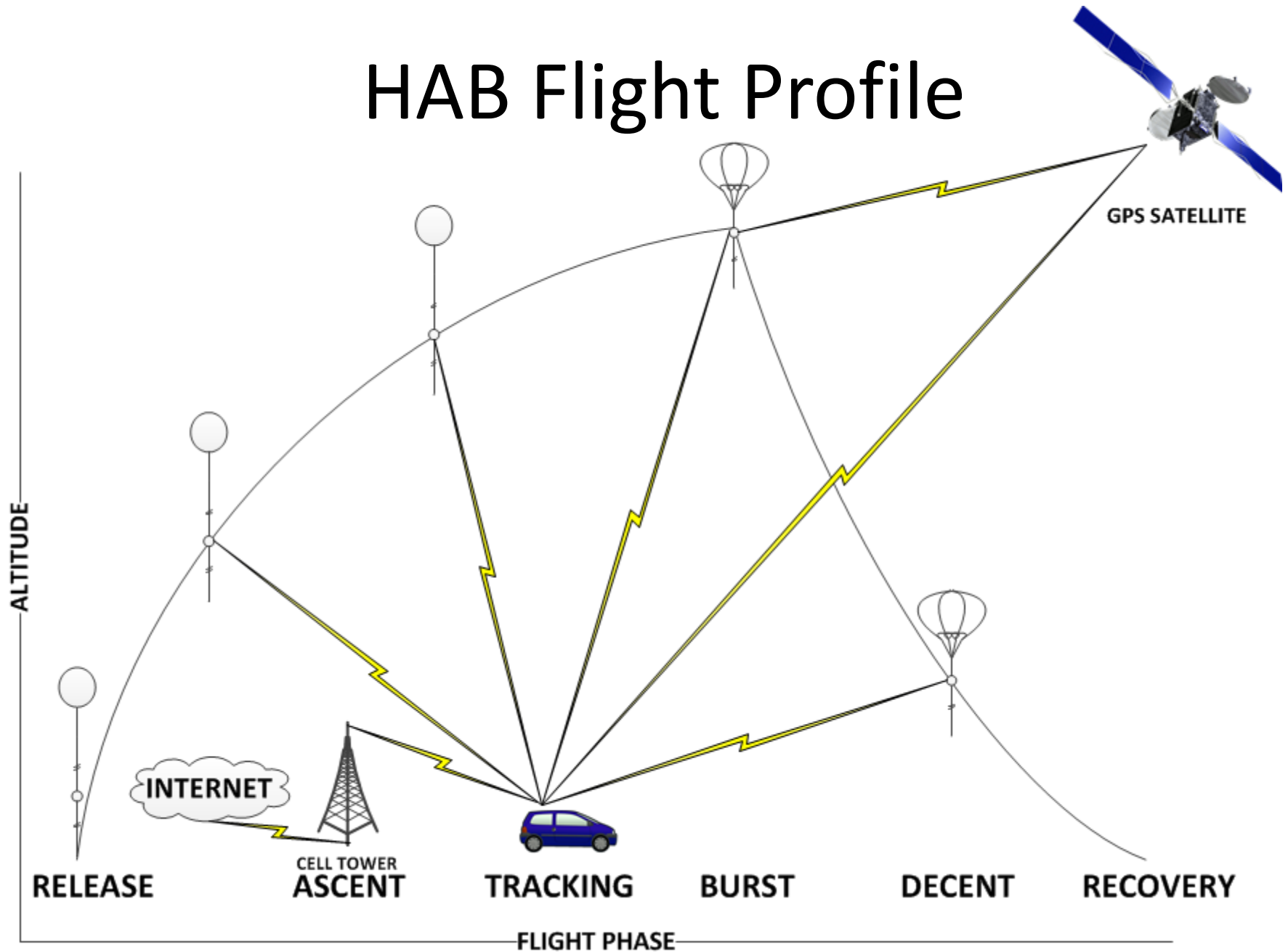
- 1 The name, address and telephone number of the person who will release the balloon (or, if several people will be involved, the name, address and telephone number of the person who will coordinate the release)
- 2 The date and time the release is to begin
- 3 Where it is to be carried out
- 4 The estimated size and mass of the balloon's payload
- 5 If more than 1 balloon is to be released at a time, how many balloons are to be released at the time

A person may operate a free balloon that carries a payload only if the payload has fixed to it a durable identification plate carrying sufficient information:

- (a) to identify the payload; and
- (b) to enable somebody who finds the payload to contact the person who released the balloon.

A person may operate a free balloon that carries a trailing antenna that requires a force of more than 230 newtons to break it only if the antenna has coloured streamers or pennants attached to it every 15 metres.

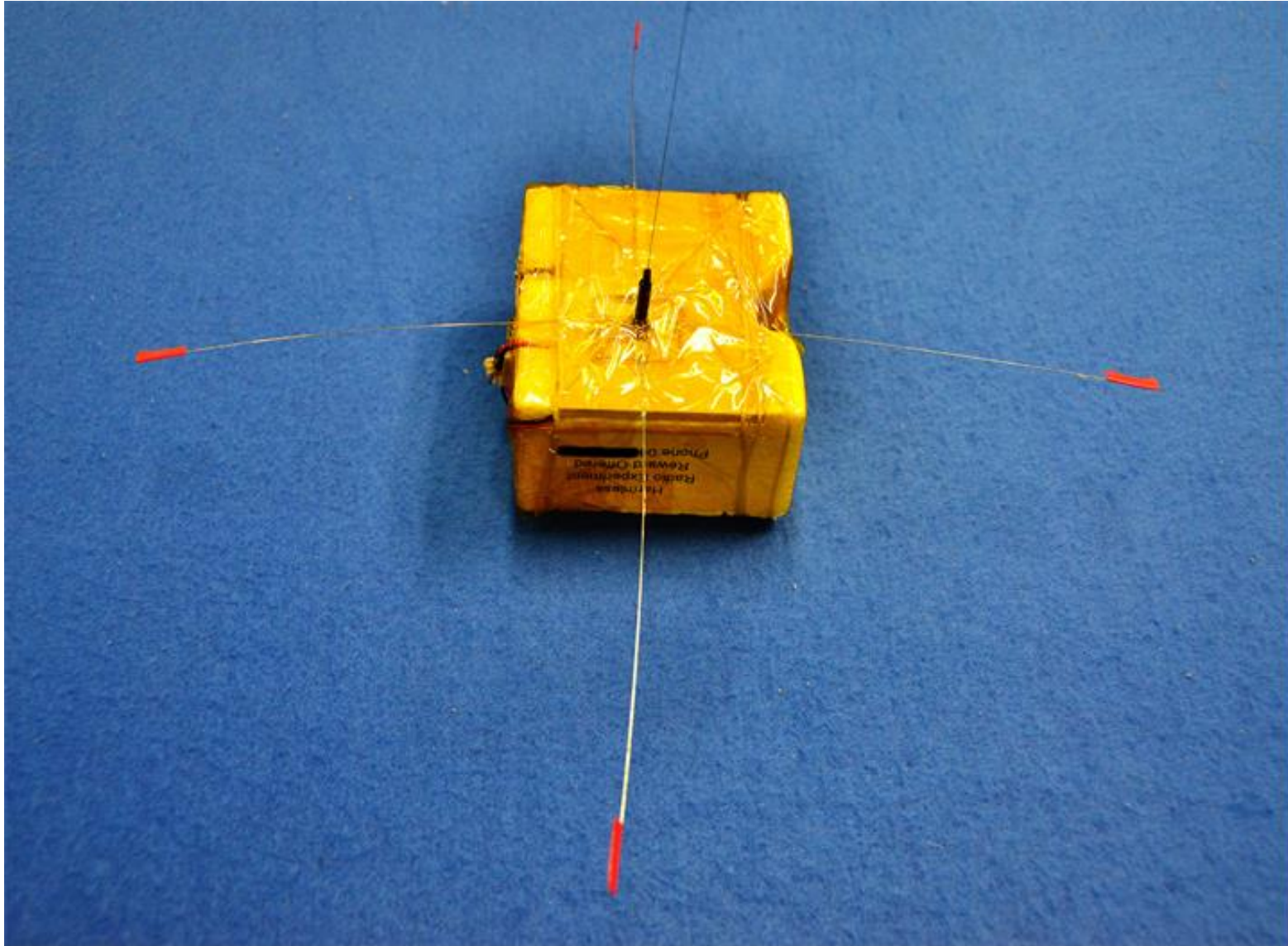
HAB Flight Profile



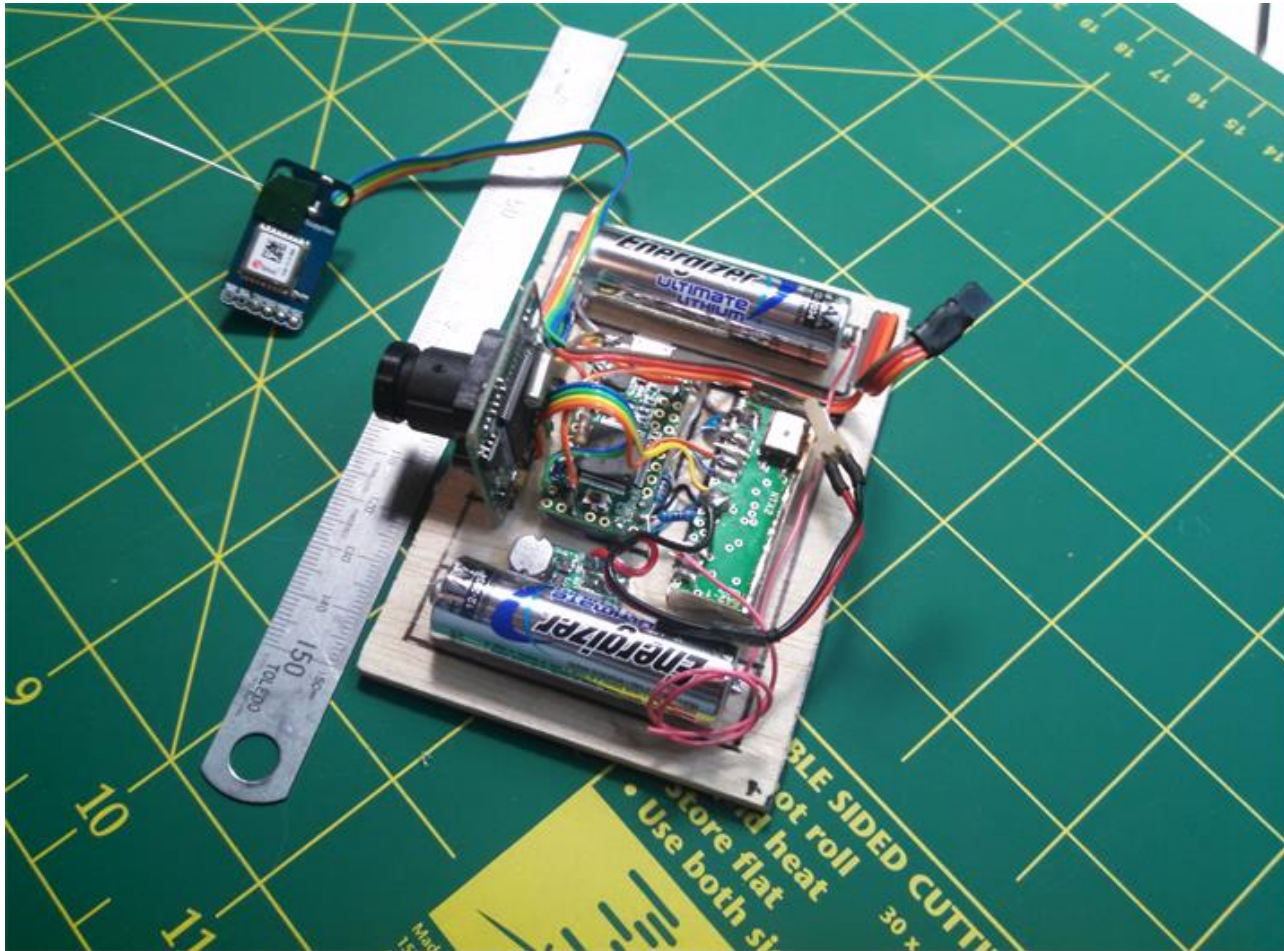
High Altitude Balloons

- Two HABs launched from Redesdale Vic.
- Both helium-filled, latex weather balloons
- MTG003:
 - 434.650MHz FM Telemetry BPSK63F
 - 434.650MHz FM Imagery BPSK1000F
- MTG004:
 - 10.139250MHz SSB JT65 Telemetry
 - 10.139750MHz SSB JT9 Telemetry

MTG003 PAYLOAD



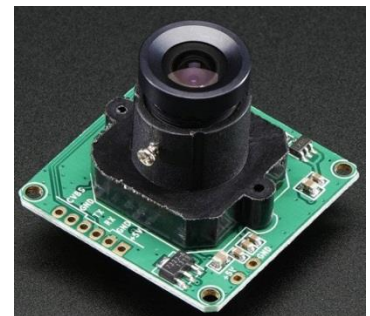
MTG003 ELECTRONICS PACKAGE



MTG003 Hardware Data

- Controller: Teensy 3.1: 3.3V, 32Bit, 72MHz
- Radiometrix RF Module: NTX2-434.650-10
- GPS Receiver: uBLOX MAX-M8C
- Camera: 640x480 Serial JPEG

Teensy 3.1



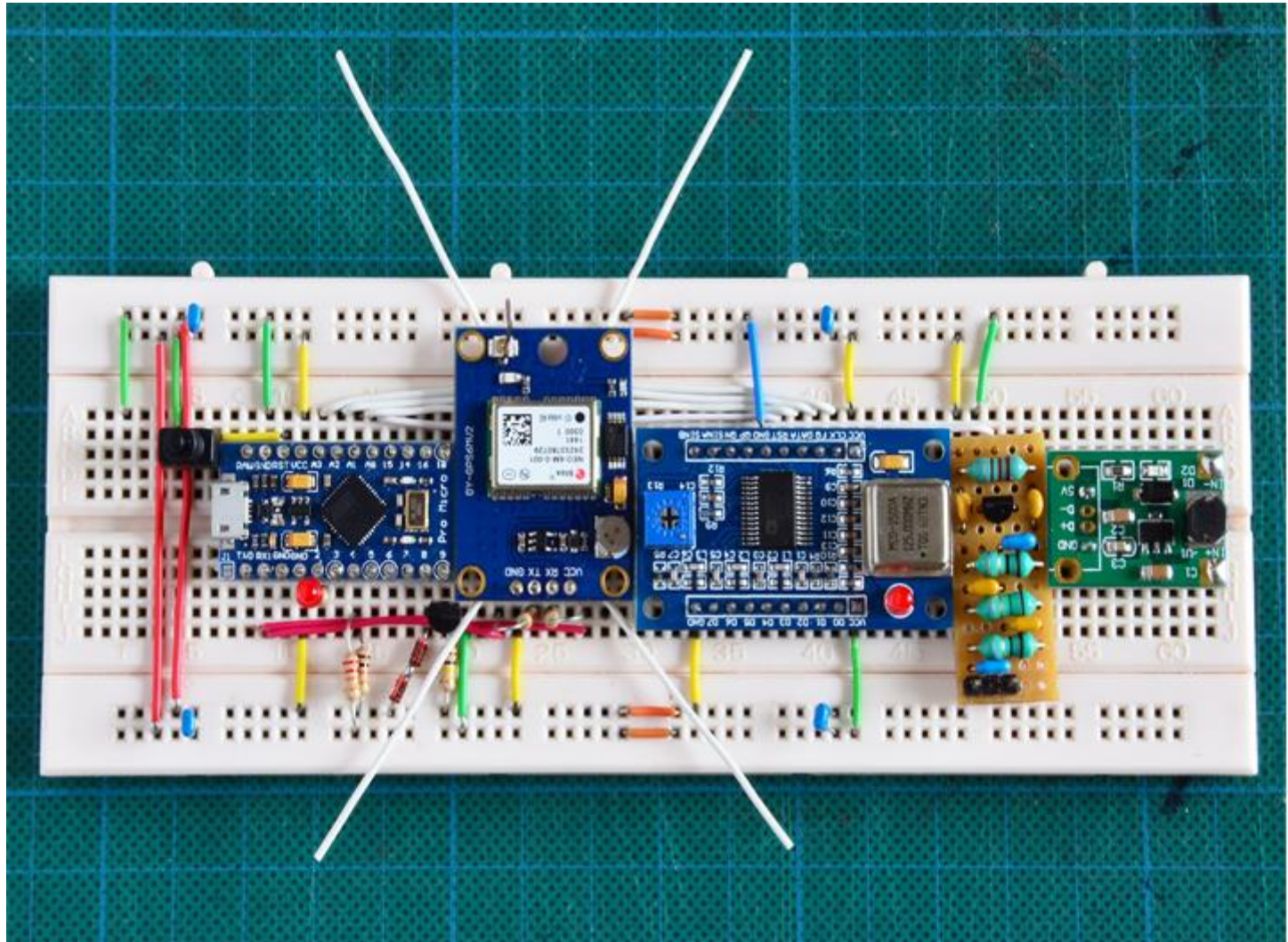
MTG003 Assembly and Rigging



MTG004 PAYLOAD

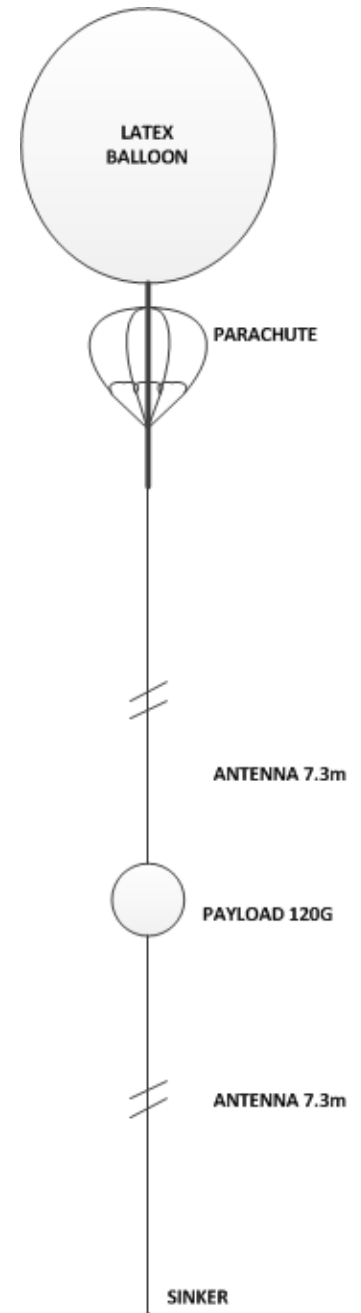


MTG004 ELECTRONICS PACKAGE



MTG004 Configuration

- Balloon diameter 1.6m
- Balloon weight 100g
- Parachute diameter 45cm
- Dipole length 14.6m
- Payload weight 120g
- Other weight 60g



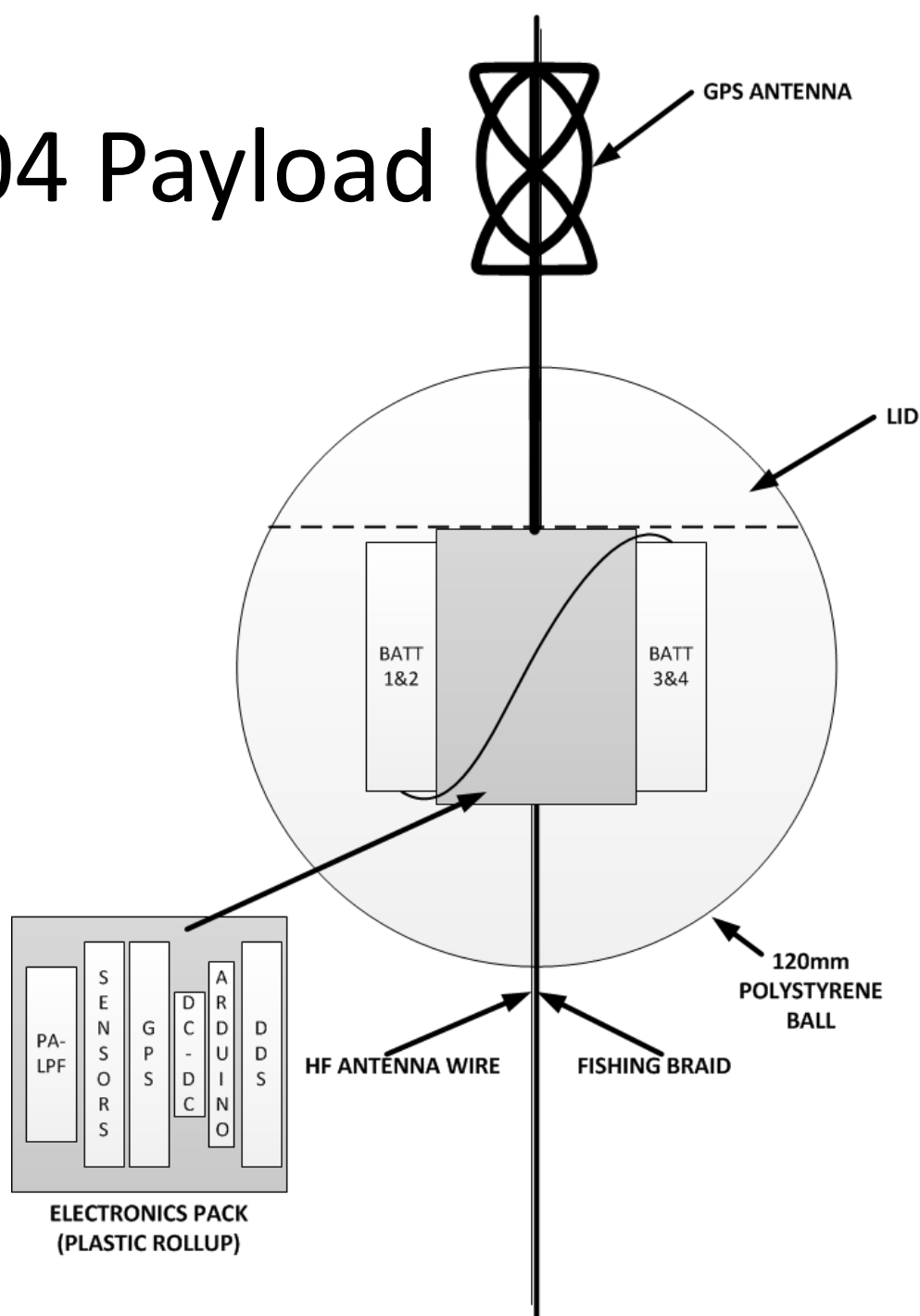
MTG004 Flight Data

- Flight Name: “MTG004”
- Payload Name: “Screwball 1”
- Operational Frequency and Modes:
 - 10.139250MHz JT65 USB
 - 10.139750MHz JT9 USB
 - Alternating every minute: JT65/JT9. Telemetry Only.
- Telemetry format:
 - Two line 13-Character, Base32-Encoded messages. 10 minute sequence.
- Message 1: Sent on minutes 0 and 5
 - Callsign, Internal Temp, Number of Satellites in View, Battery Voltage
- Message 2: Sent on minutes 1, 2, 3, 4, 6, 7, 8, 9
 - Latitude, Longitude, Altitude

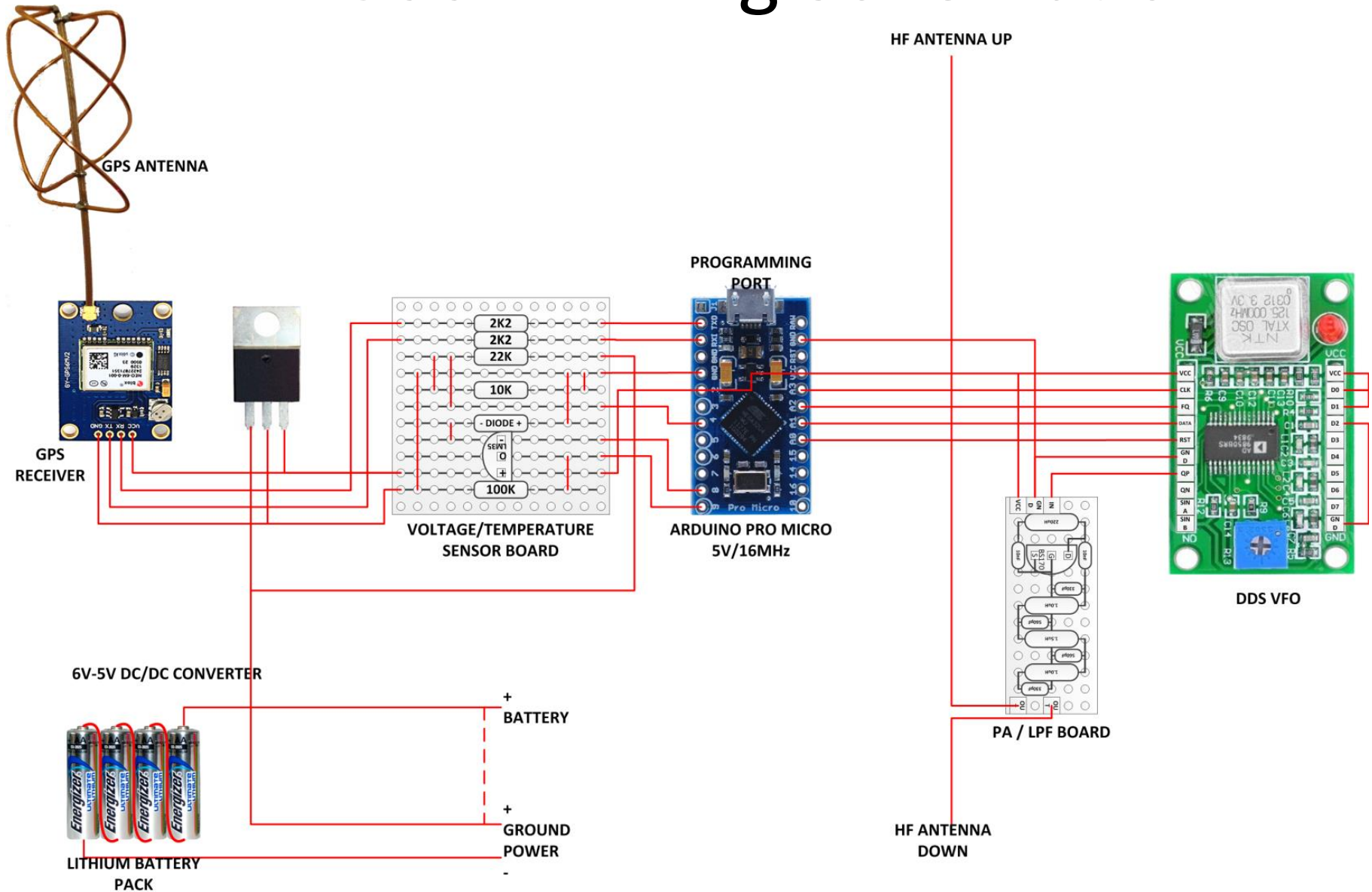
MTG004 Hardware Data

- MTG004 Configuration
 - Balloon – Pawan 100 latex weather balloon
 - Parachute – Estes 2267 Model Rocket 18 Inch Parachute
 - Payload – Spotlight 120mm Styrofoam Ball
- MTG004 Payload
 - GPS Antenna – 1.5GHz Quadrifilar Helix
 - GPS Receiver – uBLOX NEO-6MV2
 - Controller - Arduino Pro Micro ATmega32U4 5V 16MHz
 - DDS VFO - AD9850
 - 30m PA/LPF - BS170
 - Voltage/Temperature Sensor Board – LM35
 - 5V LDO Regulator – LM2940CT-5.0
 - Batteries – 4 x Energizer Lithium Ultimate AA

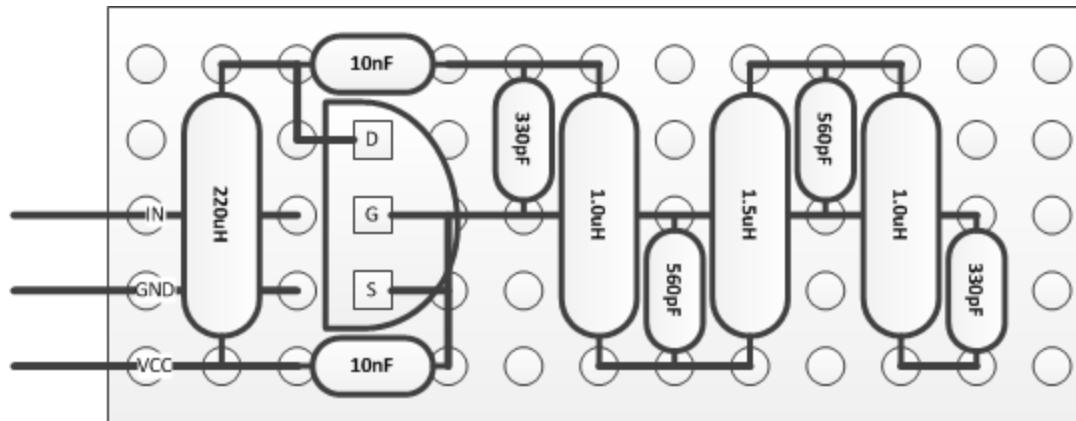
MTG004 Payload



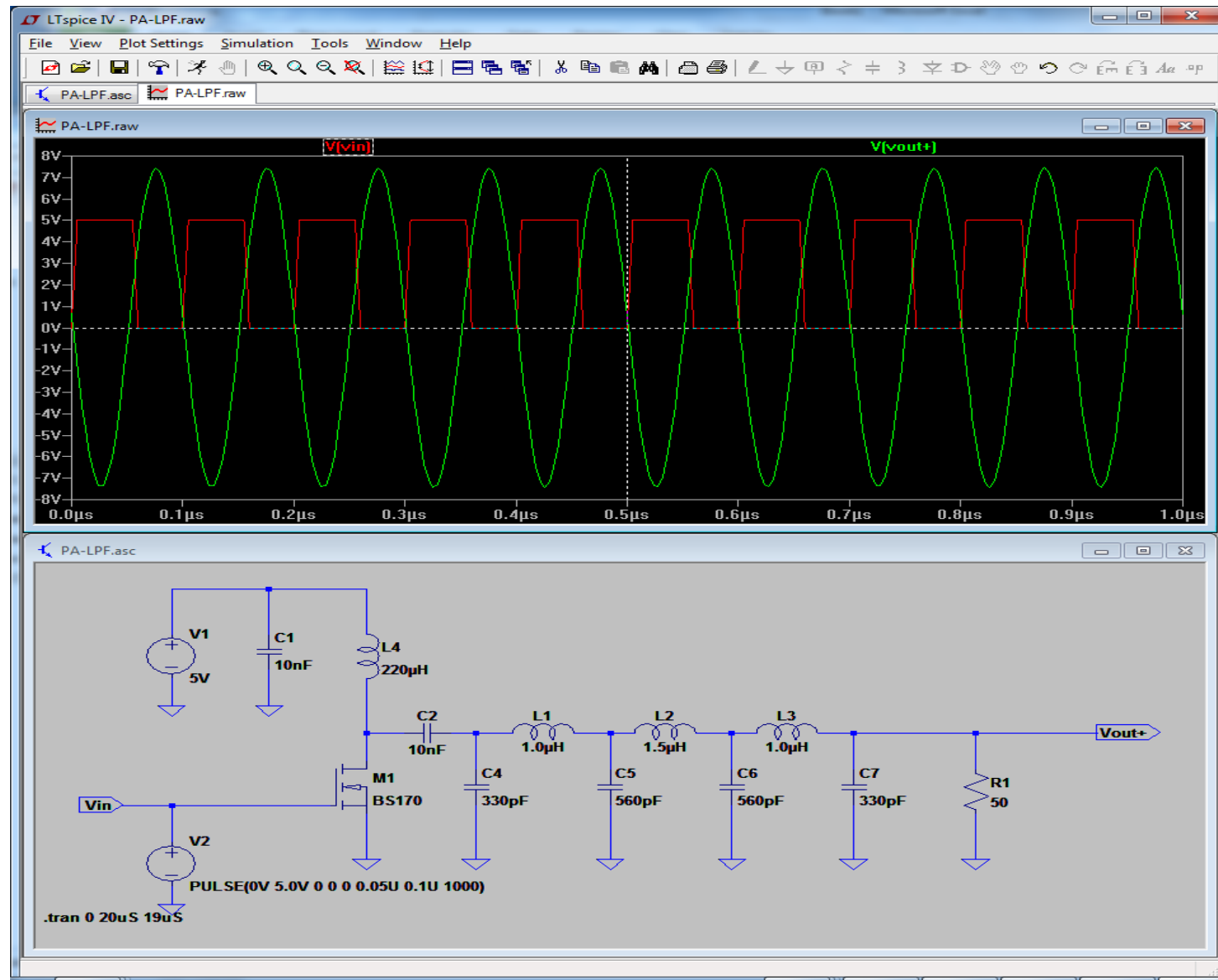
MTG004 Wiring Schematic



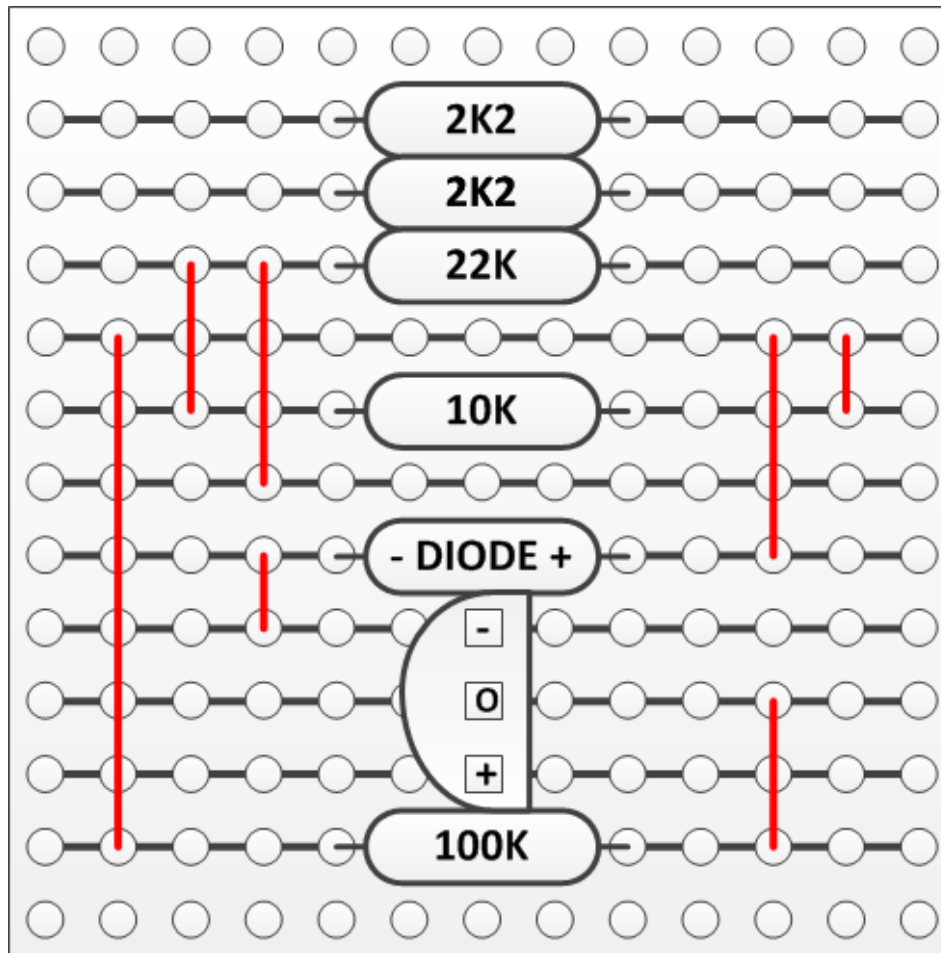
MTG004 30m Power Amplifier & LPF Board



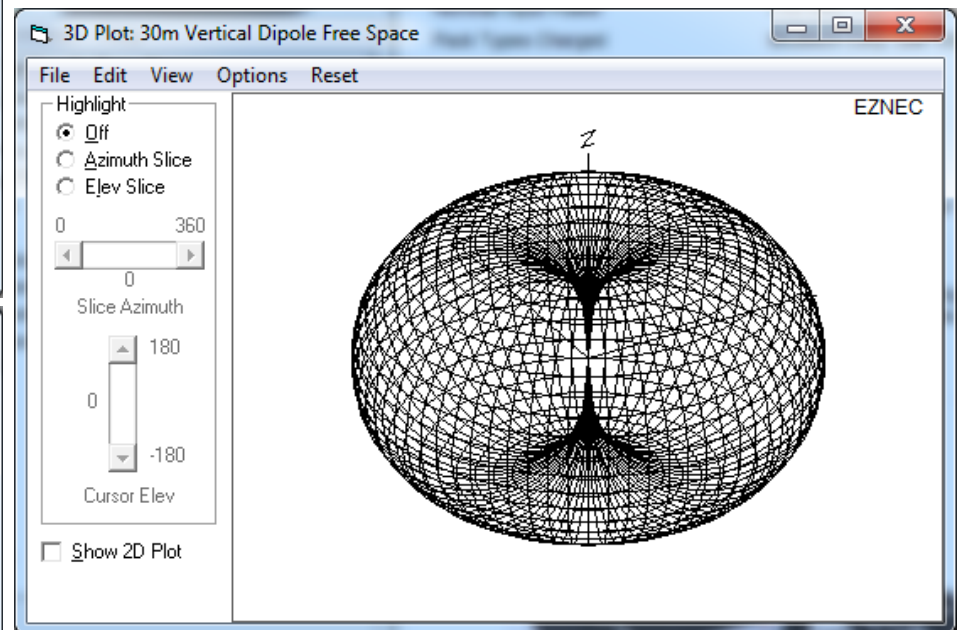
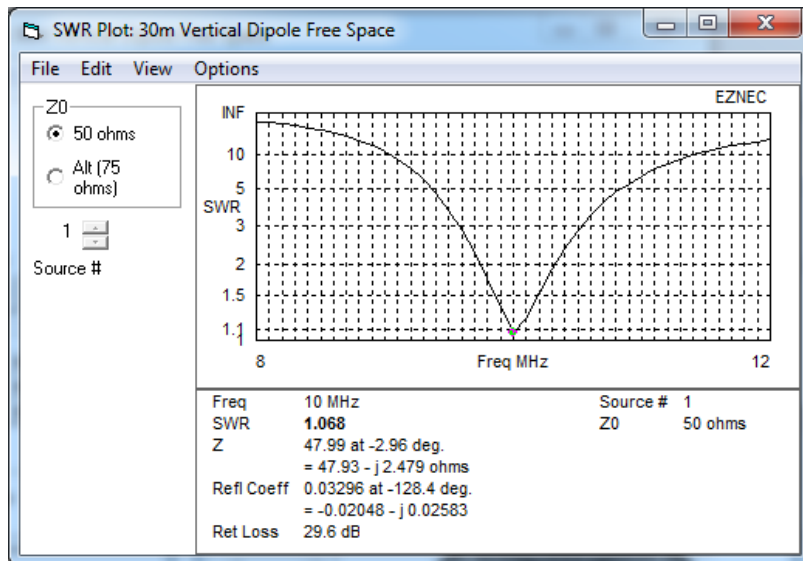
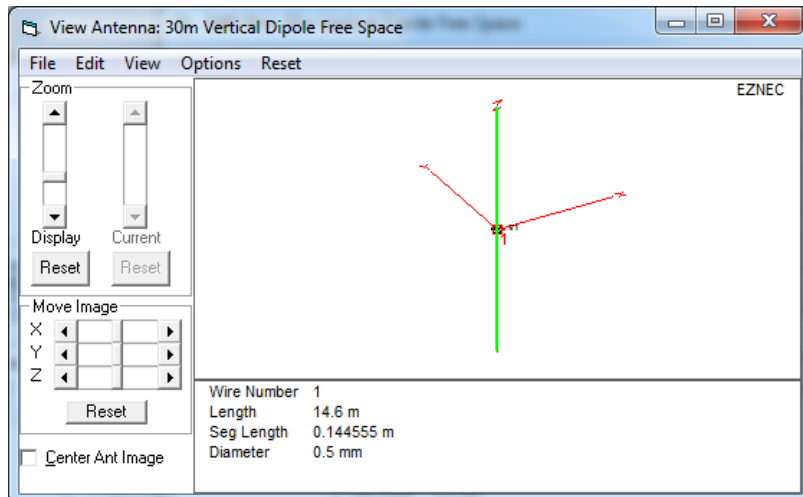
MTG004 PA-LPF Simulation



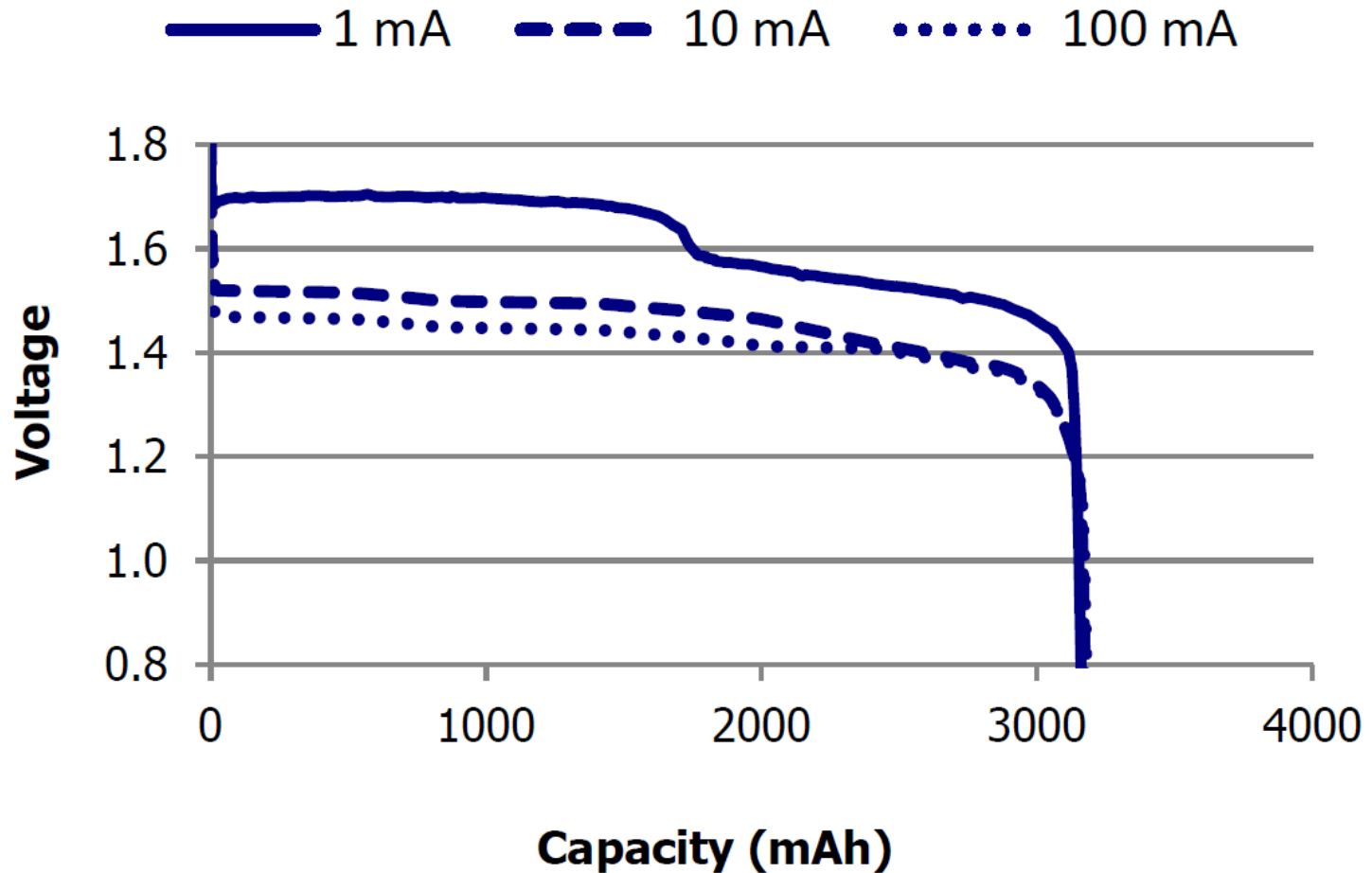
MTG004 Voltage/Temperature Sensor Board



MTG004 30m Antenna Simulation



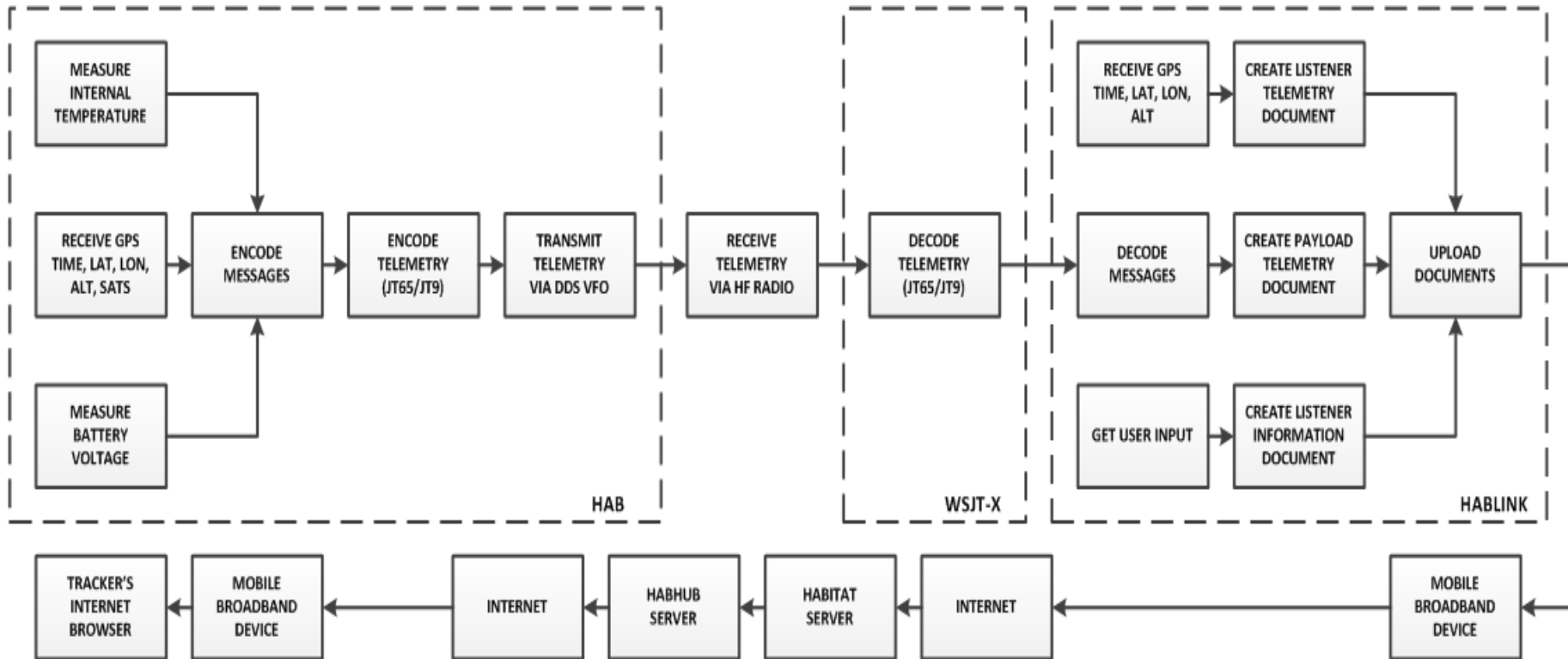
MTG004 Battery Capacity



Software Development

- Copyright (c) 2001, Dr. Joe Taylor K1JT
 - Fortran90 JT9/JT65 encoder see <http://physics.princeton.edu/pulsar/k1jt/index.html>
- Copyright (c) 2002, Phil Karn KA9Q
 - C++ Reed Solomon encoder used in JT65
- Copyright (c) 2015, Joe Gonzales VK3YSP
 - HAB: Arduino C++ application
 - HABLINK: Visual Basic application
- Released under the GNU General Public License.

Software Processing



Decoding GPS NMEA Sentences

The GPS receiver provides the following data every second at 9600bps:

\$GPRMC,**101059.00**,**A**,3754.45031,S,14505.53645,E,0.016,,**030315**,,,A*60

\$GPVTG,,T,,M,0.016,N,0.029,K,A*2F

\$GPGGA,101059.00,**3754.45031**,**S**,**14505.53645**,**E**,1,**09**,1.19,**78.5**,M,-1.9,M,,*60

\$GPGSA,A,3,32,22,18,27,04,19,24,11,14,,,,2.11,1.19,1.74*09

\$GPGSV,3,1,11,01,06,227,,04,33,228,36,11,22,228,36,14,79,019,36*79

\$GPGSV,3,2,11,18,28,106,31,19,44,272,37,21,10,048,21,22,62,136,40*7C

\$GPGSV,3,3,11,24,15,132,23,27,37,315,29,32,27,264,26*47

\$GPGLL,3754.45031,S,14505.53645,E,101059.00,A,A*7A

JT65 and JT9 Data

JT65 Data

NSPSEC = 11025 Number of samples per second

NSPSYM = 4096 Number of samples per symbol

NSPS = NSPSEC / NSPSYM = 2.7Hz Number of symbols per second

NSYN = 63 Number of sync symbols

NSYM = 126 Total number of symbols

TSYM = 1 / NSPS = 372ms Symbol period

TGAP = 60000 - NSYM * TSYM = 13128ms Transmission gap each minute

JT9 Data

NSPSEC = 12000 Number of samples per second

NSPSYM = 6912 Number of samples per symbol

NSPS = NSPSEC / NSPSYM = 1.7Hz Number of symbols per second

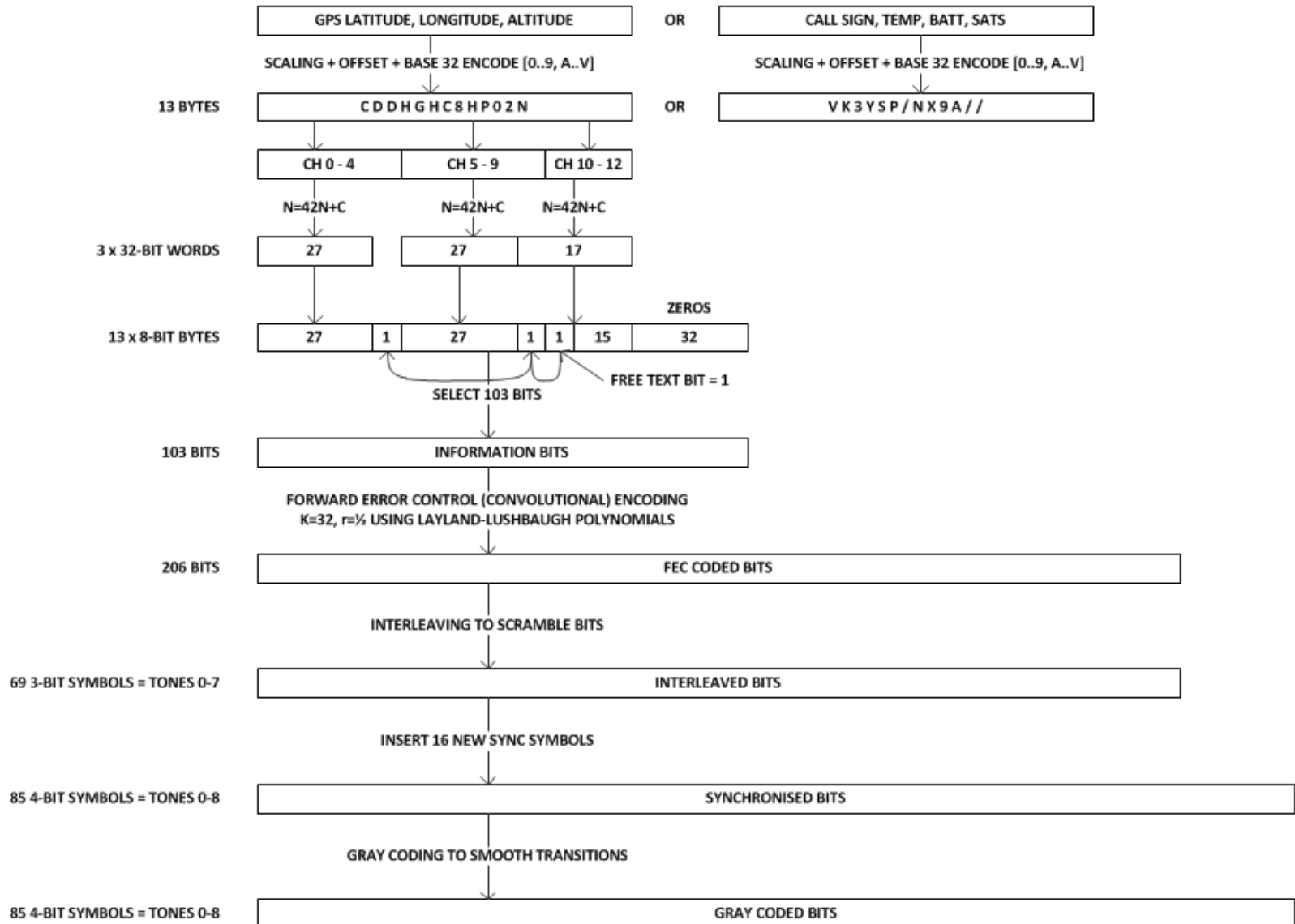
NSYN = 16 Number of sync symbols

NSYM = 85 Total number of symbols

TSYM = 1 / NSPS = 576ms Symbol period

TGAP = 60000 - NSYM * TSYM = 11040ms Transmission gap each minute

JT-9 ENCODING



Code Translation

FORTRAN

! Convolutional encoder for a K=32, r=1/2 code.

```
integer*1 dat(13)      !User data, packed 8 bits per byte
integer*1 symbol(500) !Channel symbols, one bit per byte
integer*1 i1
include 'conv232.f90'
nstate=0
k=0
do j=1,nsym
  do i=7,0,-1
    i1=dat(j)
    i4=i1
    if (i4.lt.0) i4=i4+256
    nstate=ior(ishft(nstate,1),iand(ishft(i4,-i),1))
    n=iand(nstate,npoly1)
    n=ieor(n,ishft(n,-16))
    k=k+1
    symbol(k)=partab(iand(ieor(n,ishft(n,-8)),255))
    n=iand(nstate,npoly2)
    n=ieor(n,ishft(n,-16))
    k=k+1
    symbol(k)=partab(iand(ieor(n,ishft(n,-8)),255))
    if(k.ge.nsym) go to 100
  enddo
enddo
```

C++

//Convolve 103 of these bits with Layland-Lushbaugh polynomials for a K=32, r=1/2 convolutional code to yield 206 bits

```
byte i4;
int i, j, k;
long m, n;
m = 0;
k = 0;
for (j = 0; j < 13; j++) {
  i4 = msg8[j];
  for (i = 7; i >= 0; i--) {
    m = ((m << 1) | ((i4 >> i) & 1));
    n = m & POLY1;
    n ^= n >> 16;
    //enc206[k++] = PARITY[((n ^ (n >> 8)) & 255)]; //Fast parity
    enc206[k++] = parity(((n ^ (n >> 8)) & 255)); //Compact parity
    n = m & POLY2;
    n ^= n >> 16;
    //enc206[k++] = PARITY[((n ^ (n >> 8)) & 255)]; //Fast parity
    enc206[k++] = parity(((n ^ (n >> 8)) & 255)); //Compact parity
    if (k >= BITS) break; //Stop after 206 bits
  }
  if (k >= BITS) break; //Stop after 206 bits
}
```

HABITAT CouchDB JSON Documents

Java Script Object Notation:

```
{"type":"listener_information","time_created":"2015-04-13T20:09:50Z","time_uploaded":"2015-04-13T20:09:50Z","data":{"callsign":"VK3YSP","radio":"ICOM IC-7200","antenna":"Dipole"}}
```

```
{"type":"listener_telemetry","time_created":"2015-04-13T20:09:51Z","time_uploaded":"2015-04-13T20:09:51Z","data":{"callsign":"VK3YSP","latitude":-37.9077017,"longitude":145.0922550,"altitude":43.7,"chase":true}}
```

```
{"type":"payload_telemetry","_id":"1a55c70410acda73091539e416897074a9cbc211cc1c9173fca917169bcc9055","data":{"_raw":"JCRNVEcwMDQsMSwyMDoxMDo1MCwtMzc1NC40NjMsMTQ1MDUuNTIsMTQsNCwxOC44LDUqMTY3MQo="},"receivers":{"VK3YSP":{"time_created":"2015-04-13T20:10:50Z","time_uploaded":"2015-04-13T20:10:50Z"}}
```


HABITAT Payload Document

The screenshot shows a web browser window with the URL `habitat.habhub.org/genpayload/`. The page title is "Sentence editor" with the subtitle "Parser configuration". The interface is divided into several sections for configuring a payload document.

Description: A text input field containing "Normal". A tooltip on the right says: "Optional but strongly recommended, e.g., Normal format, No-lock format (if it's different), Long format (i.e., more fields), and so on."

Protocol: A text input field containing "UKHAS".

Callsign: A text input field containing "MTG004" with a green checkmark to its right.

Checksum type: A dropdown menu showing "crc16-ccitt".

fields: A table of fields with their types and formats. Each row has a blue expand/collapse icon on the left and a green checkmark on the right.

Field Name	Type	Format
sentence_id	Integer	
time	Time	
latitude	Coordinate	ddmm.mmmmm
longitude	Coordinate	ddmm.mmmmm
altitude	Integer	
volts	Float	
temperature_internal	Float	
satellites	Integer	

Below the table are two buttons: "Add Normal Field" and "Add Custom Field".

intermediate filters: Two buttons: "Add filter" and "Add postfix".

post parse filters: A table with one row and two columns. The first column has a blue expand/collapse icon and the text "Filter". The second column has the text "Value".

Filter	Value
common.invalid_always	

Below the table are two buttons: "Add filter" and "Add postfix".

At the bottom of the page are two buttons: "Save" and "Cancel".

HABITAT Flight Document

The screenshot shows a web browser window with the URL <http://habitat.habhub.org/genpayload>. The page title is "habitat genpayload". The form is titled "Flight" and contains the following fields and sections:

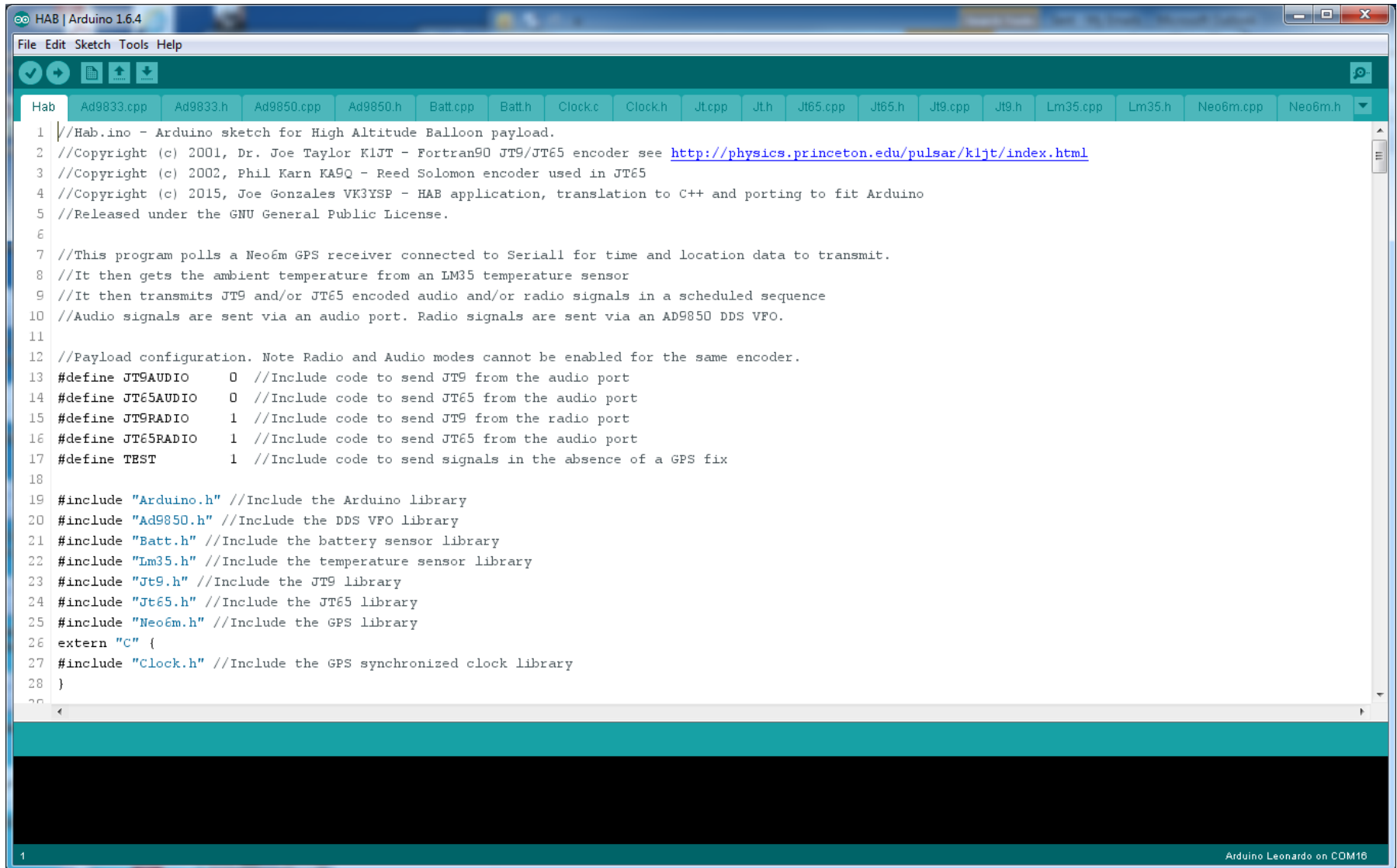
- Flight name:** MTG004 (with a green checkmark). Help text: "e.g., Delamion launch 32 or Shark Christmas launch."
- Project name:** GSBG. Help text: "Optional, but highly recommended."
- Group name:** MARTG. Help text: "Or your name, if you're working alone ->. Optional, also recommended."
- Timezone:** Australia/Melbourne (with a green checkmark). Help text: "Launch time below is interpreted in this timezone; daylight saving detected automatically from the date. Some clients/trackers may display times related to this flight in this timezone rather than the user's."
- Launch Date:** A calendar for April 2015. The date 14 is highlighted in yellow.
- Time:** 11:00 (with a green checkmark). Help text: "Launch time: 2015-04-19 11:00:00"
- Launch window:** Launch day (dropdown menu). Help text: "This period must cover the entire flight, and ideally exclude testing, so that only telemetry from the flight itself is archived. Launch window: 2015-04-19 00:00:00 - 2015-04-19 23:59:59"
- Launch location:** Reedsdale VIC 3444. Help text: "Optional. Suggested format: Place name, Area, Country"
- Latitude:** -37.02678847 (with a green checkmark). Help text: "decimal degrees"
- Longitude:** 144.546325 (with a green checkmark). Help text: "decimal degrees -180 <= 180"
- Altitude:** 100 (with a green checkmark). Help text: "metres. Optional: only enter it if it's significant."

payloads in this flight

Flight Name	Sentence Count	Transmission Count	Created
MTG004	1 sentence	1 transmission	2015-04-10T09:37:23+10:00

Buttons: Add, Delete, Save, Cancel

HAB App - Arduino IDE

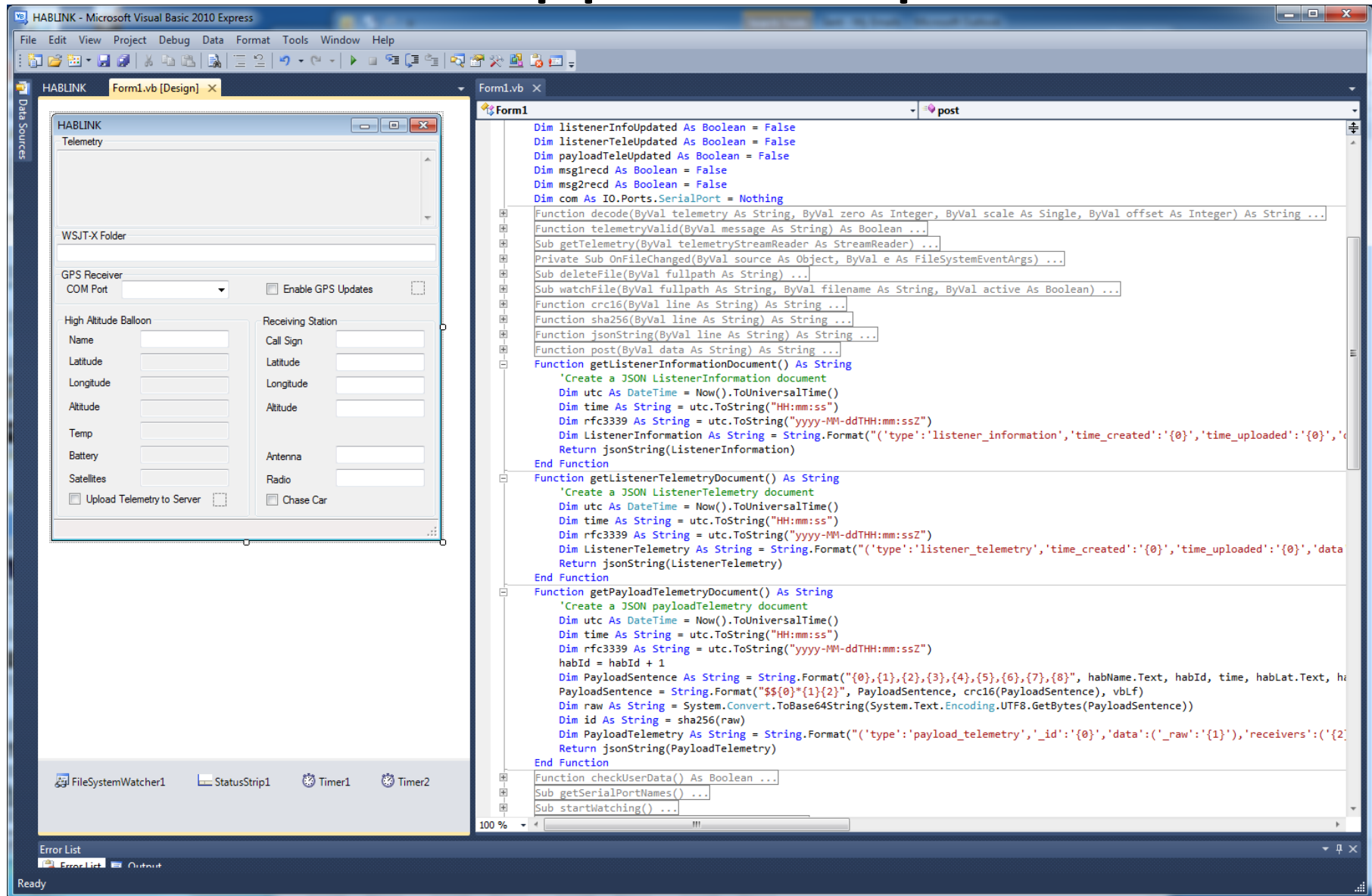


The screenshot shows the Arduino IDE interface with the 'HAB' sketch open. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. The toolbar contains icons for opening files, saving, and uploading. The file explorer at the top shows a list of files: 'Hab', 'Ad9833.cpp', 'Ad9833.h', 'Ad9850.cpp', 'Ad9850.h', 'Batt.cpp', 'Batt.h', 'Clock.c', 'Clock.h', 'Jt.cpp', 'Jt.h', 'Jt65.cpp', 'Jt65.h', 'Jt9.cpp', 'Jt9.h', 'Lm35.cpp', 'Lm35.h', 'Neo6m.cpp', and 'Neo6m.h'. The main text area displays the code for 'Hab.ino', which is an Arduino sketch for a High Altitude Balloon payload. The code includes comments about copyright and licensing, and defines various configuration options for the payload. The bottom status bar indicates 'Arduino Leonardo on COM16'.

```
1 //Hab.ino - Arduino sketch for High Altitude Balloon payload.
2 //Copyright (c) 2001, Dr. Joe Taylor K1JT - Fortran90 JT9/JT65 encoder see http://physics.princeton.edu/pulsar/k1jt/index.html
3 //Copyright (c) 2002, Phil Karn KA9Q - Reed Solomon encoder used in JT65
4 //Copyright (c) 2015, Joe Gonzales VK3YSP - HAB application, translation to C++ and porting to fit Arduino
5 //Released under the GNU General Public License.
6
7 //This program polls a Neo6m GPS receiver connected to Serial1 for time and location data to transmit.
8 //It then gets the ambient temperature from an LM35 temperature sensor
9 //It then transmits JT9 and/or JT65 encoded audio and/or radio signals in a scheduled sequence
10 //Audio signals are sent via an audio port. Radio signals are sent via an AD9850 DDS VFO.
11
12 //Payload configuration. Note Radio and Audio modes cannot be enabled for the same encoder.
13 #define JT9AUDIO 0 //Include code to send JT9 from the audio port
14 #define JT65AUDIO 0 //Include code to send JT65 from the audio port
15 #define JT9RADIO 1 //Include code to send JT9 from the radio port
16 #define JT65RADIO 1 //Include code to send JT65 from the audio port
17 #define TEST 1 //Include code to send signals in the absence of a GPS fix
18
19 #include "Arduino.h" //Include the Arduino library
20 #include "Ad9850.h" //Include the DDS VFO library
21 #include "Batt.h" //Include the battery sensor library
22 #include "Lm35.h" //Include the temperature sensor library
23 #include "Jt9.h" //Include the JT9 library
24 #include "Jt65.h" //Include the JT65 library
25 #include "Neo6m.h" //Include the GPS library
26 extern "C" {
27 #include "Clock.h" //Include the GPS synchronized clock library
28 }
29
```

1 Arduino Leonardo on COM16

HABLINK App - VB Express IDE



HAB Burst Calculator



burstcalc

balloon burst calculator

[About](#) | [Help](#)

Payload Mass (g)

140

Balloon Mass (g)

Pawan - 100



Target Burst Altitude (m)

Target Ascent rate (m/s)

2

Result

Burst Altitude: **15174 m**

Ascent Rate: **2.00 m/s**

Time to Burst: **126 min**

Neck Lift: **171 g**

Volume: **0.26 m³**

264 L
9.3 ft³

HAB Descent Rate Calculator

Descent Rate Calculator

For a rocket weighing 140 grams with a hexagonal parachute which is 45 centimeters in diameter, the descent rate is approximately 4.64 meters per second (16.71 kilometers per hour).

The descent time from 15174 meters would be about 3269 seconds.

The recommended parachute for a rocket of that weight is one with a diameter of about 40 centimeters.

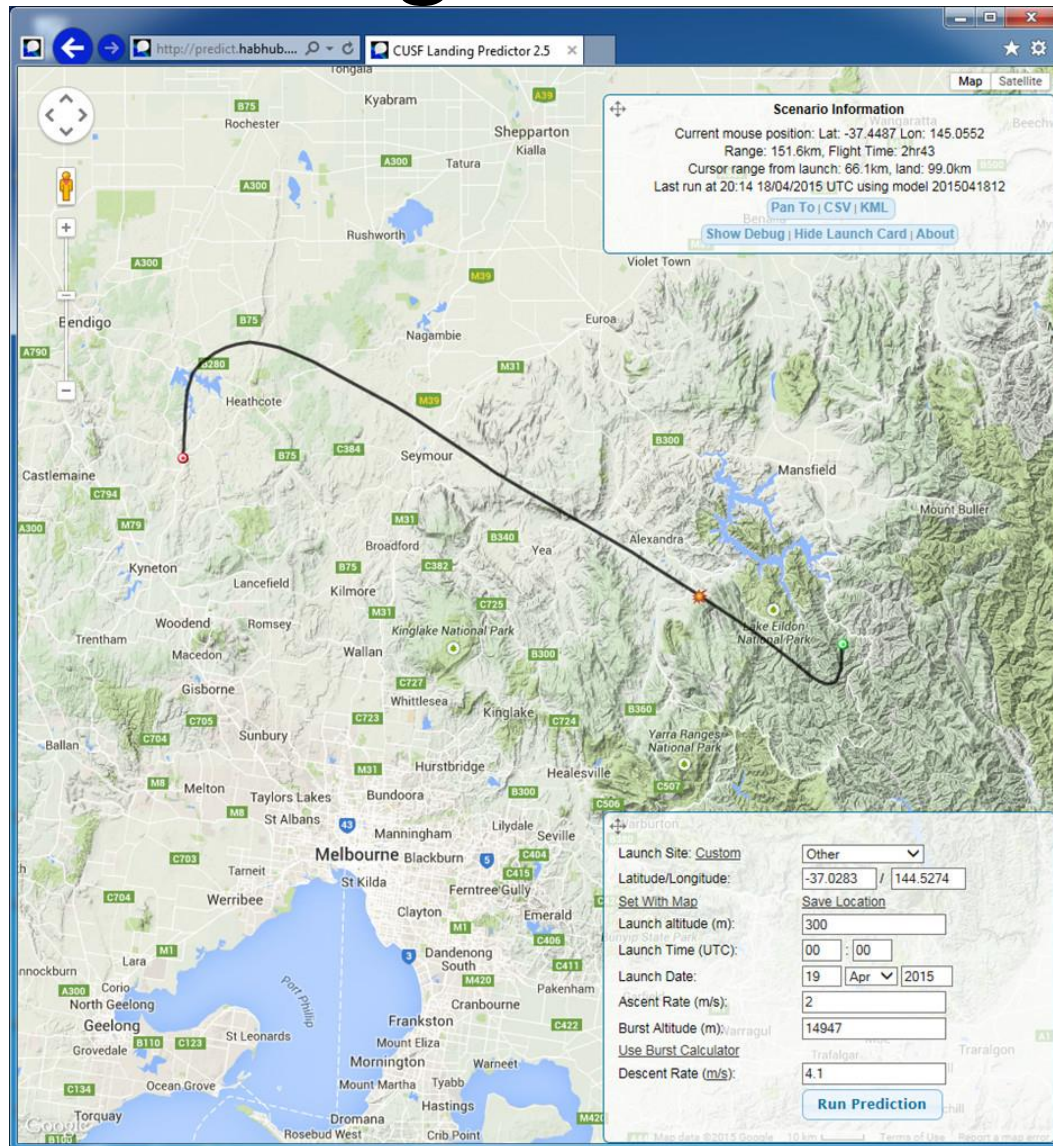
This tool estimates the descent rate for your rocket as it falls to the ground under its parachute. This calculator is based on the original EMRR calculator by Jordan Hiller.

Enter the weight of your rocket (don't forget to include the weight of the expended motor). Then enter the diameter (or maximum width) and choose the approximate shape of the parachute. Optionally, enter the altitude at which you expect the parachute to deploy.

Rocket Weight	<input type="text" value="140"/>	<input type="text" value="grams"/>
Parachute Diameter	<input type="text" value="45"/>	<input type="text" value="centimeters"/>
Parachute Shape	<input type="text" value="hexagonal"/>	
Altitude (optional)	<input type="text" value="15174"/>	<input type="text" value="meters"/>

Submit

HAB Flight Predictor



MARTG Mission Control



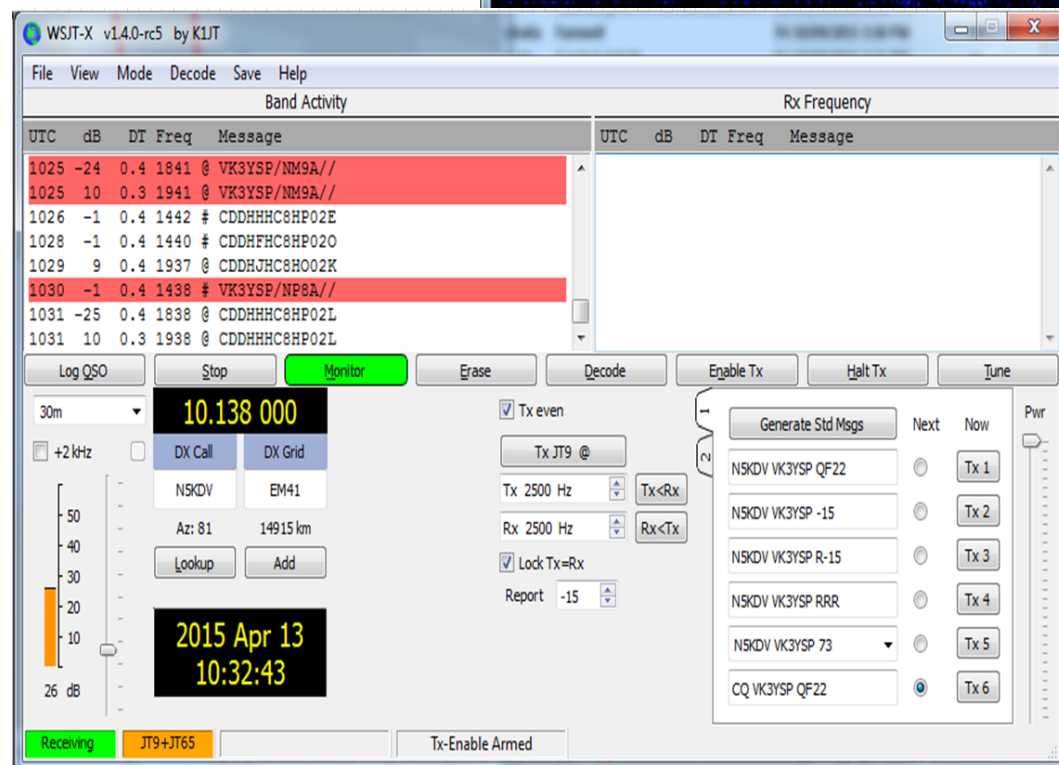
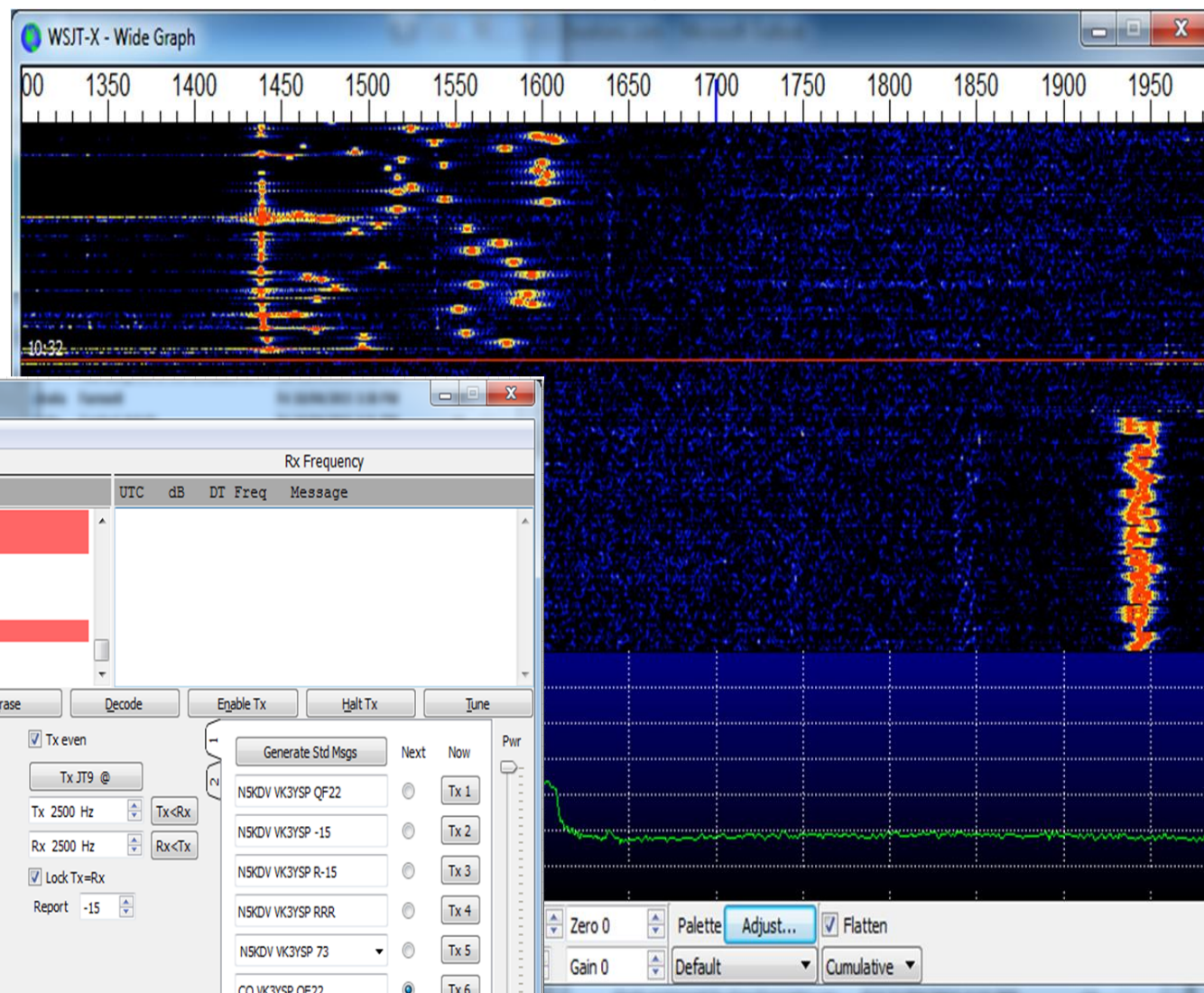
HAB Launch Team



HAB Launch



HAB Tracking



Uploading Telemetry to HABITAT

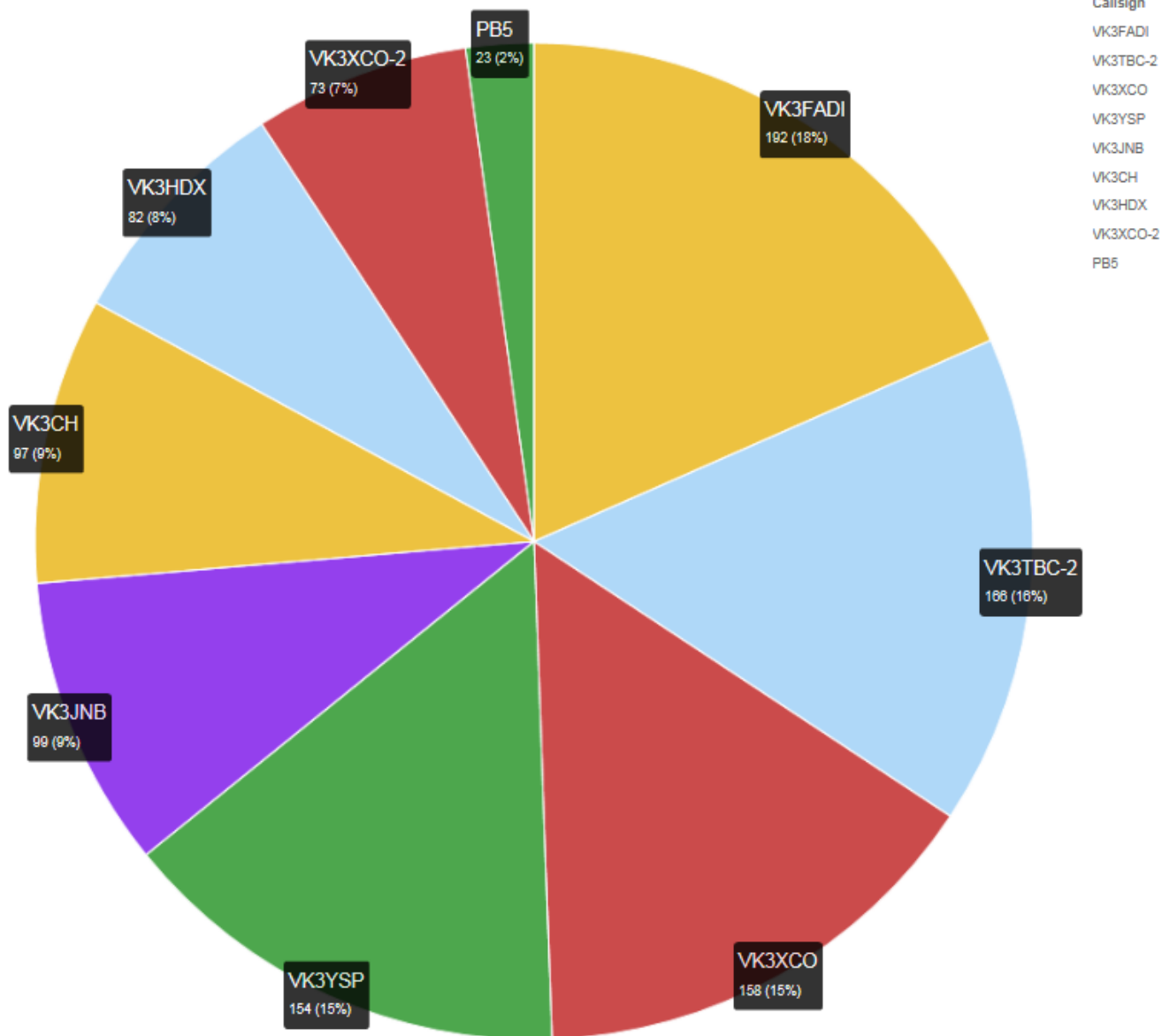
The screenshot shows the HABLINK application window. It has a title bar with standard Windows window controls. The main content area is divided into several sections:

- Telemetry:** A list box containing four lines of telemetry data:

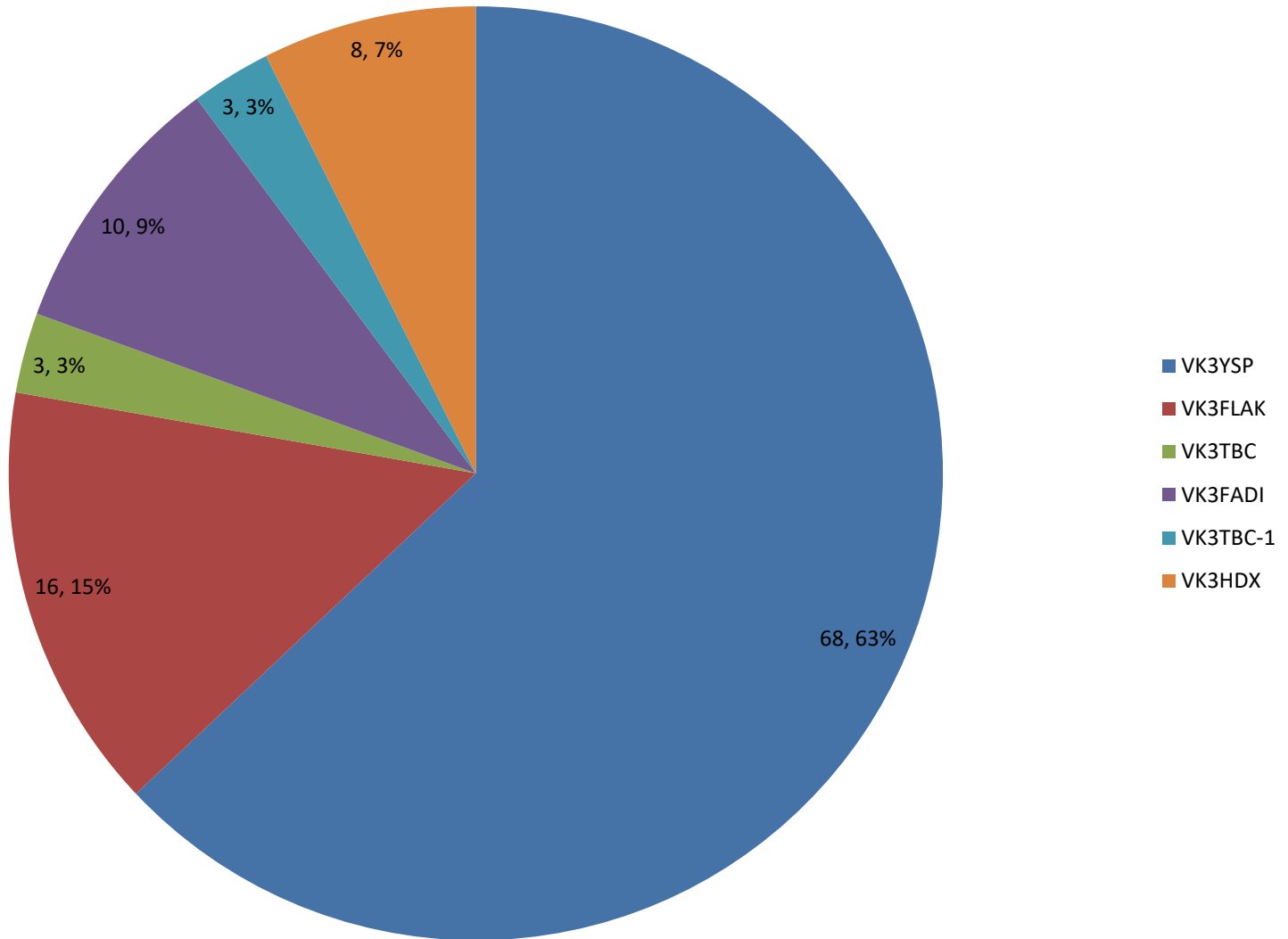
```
0510 -1 0.4 1389 # VK3YSP/YC8N//  
0512 -1 0.4 1398 # CEL8OHCDDHK0G4  
0516 -1 0.5 1389 # CELA3HCDHI0GC  
0523 27 0.3 1897 @ CEL9RHCDHI0FL
```
- WSJT-X Folder:** A text field showing the path `C:\Users\joeg\AppData\Local\WSJT-X`.
- GPS Receiver:** A section with a "COM Port" dropdown menu set to "COM25" and a checked checkbox for "Enable GPS Updates" next to a red status indicator.
- High Altitude Balloon:** A section with input fields for Name (MTG004), Latitude (-3713.733), Longitude (14556.66), Altitude (501), Temp (-53.2), Battery (6.3), and Satellites (8). It also has a checked checkbox for "Upload Telemetry to Server" next to a green status indicator.
- Receiving Station:** A section with input fields for Call Sign (VK3YSP), Latitude (-37.2329252), Longitude (145.9095687), and Altitude (239.6). It also has dropdown menus for Antenna (Dipole) and Radio (ICOM IC-7200), and a checked checkbox for "Chase Car".

At the bottom of the window, a status bar displays the text: "Telemetry received: 24:11s ago. Telemetry uploaded: 24:08s ago."

MTG003 Listener Statistics



MTG004 Listener Statistics



MTG004 Flight Data

Launch Weight: 179g

Flight Distance: 172km

Flight Time: 2 hours 13 minutes

Maximum Recorded Altitude: 11,871 metres = 38,946 Feet

Maximum Internal Temperature: 53.6°C

Minimum Internal Temperature: 2.9°C

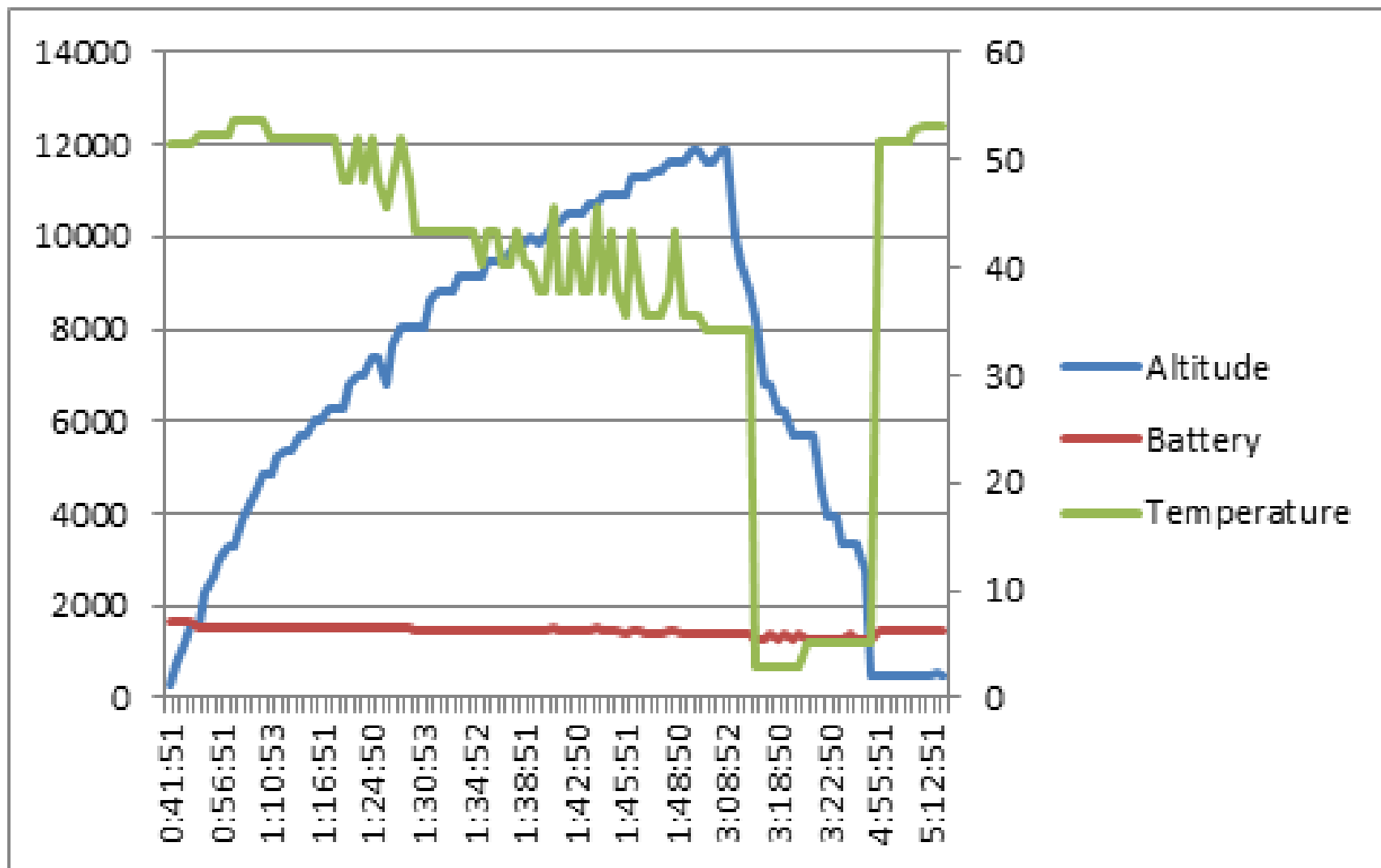
Maximum Battery Voltage: 7.1

Minimum Battery Voltage: 5.4

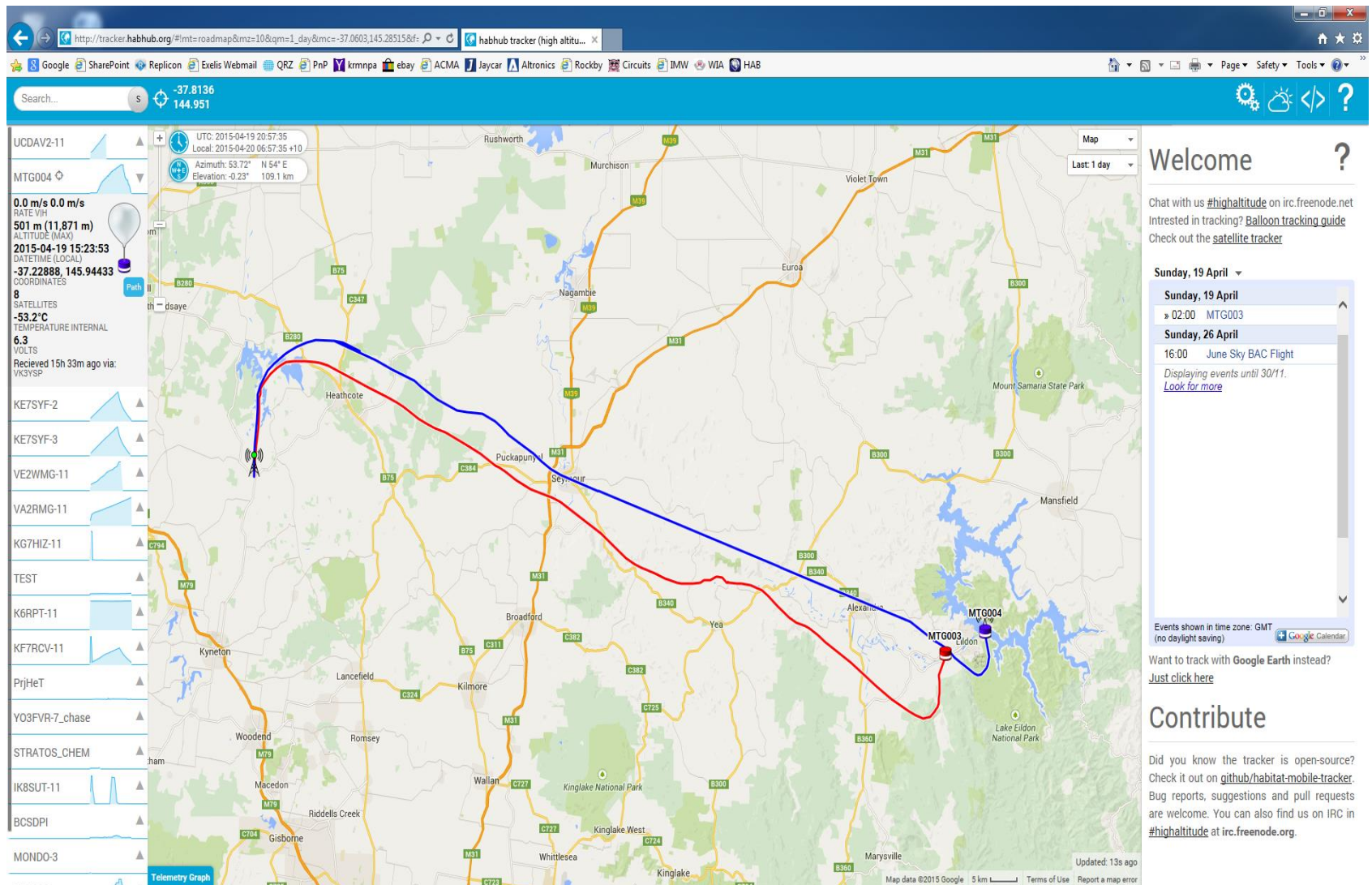
Ascent Rate: 3m/s

Decent Rate: 9m/s

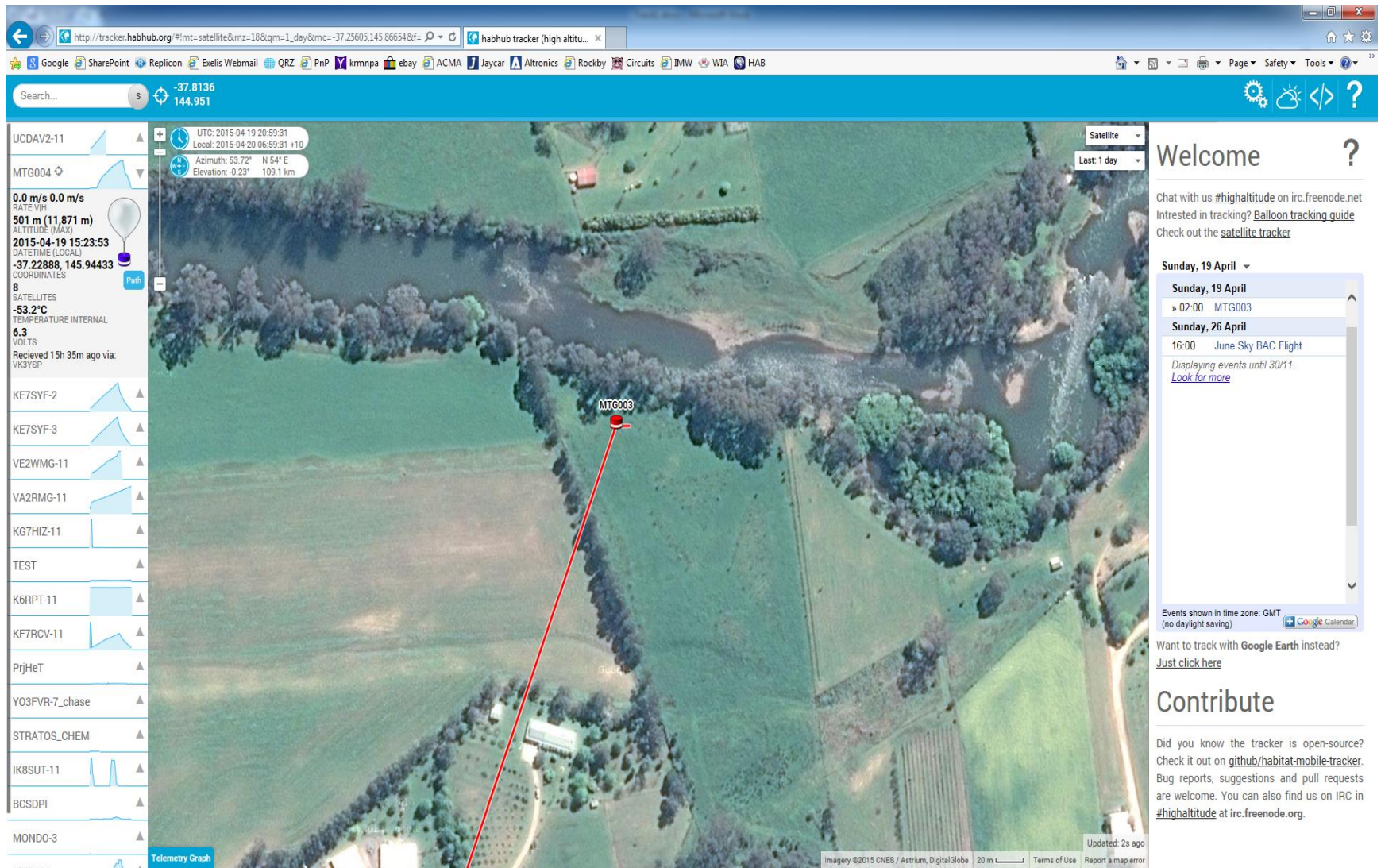
MTG004 Flight Data



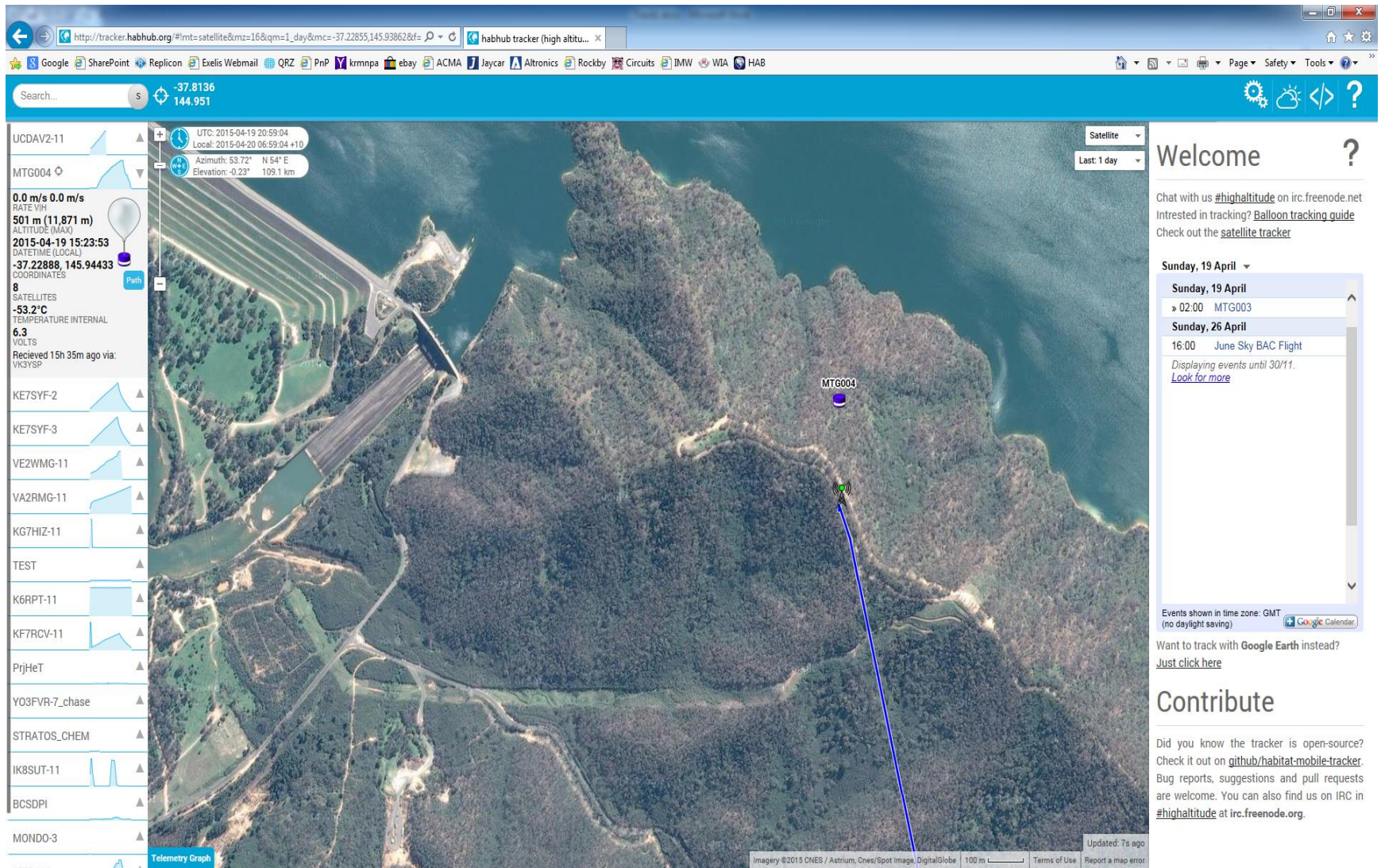
MTG003 and MTG004 Flight Track



MTG003 Landing Site



MTG004 Landing Site



HAB Recovery



HAB Recovery Team



MTG003 Pictures



Observations and Conclusions

General:

Project management, teamwork, roles and responsibilities

- Surprisingly, all aspects of the event were well executed with a minimum of planning and coordination
- Team members acted both autonomously and collectively, as required, with a high degree of professionalism and cooperation
- All effort, equipment, consumables and transport was freely donated by members

Hardware and software design

- Despite an aggressive development schedule the hardware and software actually worked with few exceptions

Accuracy of flight prediction

- Actual flight path and landing sites closely correlated with software predictions

Use of Forward Error Control coding

- Reception of telemetry and images was largely error free.

Telemetry link and battery endurance

- Telemetry was available during all phases of the flight profile.

Sensor performance

- The use and accuracy of the chosen sensors was satisfactory.

Observations and Conclusions

Burst altitude exceeded by large margin

- The actual burst altitude of 20km grossly exceeded the predicted burst altitude of 15km.
- High winds at launch precluded accurate helium fill and precise lift measurements.
- Launch in calm weather or perform pre-launch checks in a protected area.

MTG004 mishap prior to launch

- Strong winds caught the parachute and dragged the payload along the ground for 50m.
- Abort launch in windy conditions and secure payload while on ground.

Recovery team not in position for landing

- The recovery team arrived at the landing site 1-2 hours after landing.
- Luckily post-flight telemetry was available to facilitate recovery.
- Next time plan for recovery team to be on station.

Observations and Conclusions

MTG004:

Poor GPS Receiver sensitivity

- Desensitisation of GPS receiver due to DDS VFO. Shielding and filtering proved ineffectual.
- Use a high-gain QFD Antenna or a different DDS VFO.

Loss of GPS above 12km altitude

- Datasheet and application notes not sufficiently reviewed. Incorrect software implementation.
- Correctly initialize uBLOX GPS Dynamic Model at start up. Set Dynamic Platform = Airborne < 1g.

Excessive start-up time

- Incorrect scheduling of Housekeeping and Location telemetry frames at 1:4 ratio
- Change scheduling of Housekeeping and Location telemetry frames to 1:1 ratio

Inconsistent telemetry uploaded to HABITAT

- Incorrect scheduling of telemetry frames and incorrect use of legacy data.
- Change scheduling of telemetry frames and only upload fresh data.

Only one listener callsign displayed on HABHUB

- Use of UKHAS recommended SHA256 encryption to avert duplicate frames
- Abandon HABITAT recommendation and make all listener payload telemetry documents unique

Difficulty decoding JT9 on ground

- High signal strength on ground coupled with incorrect setting of RF and WSJT-X gains
- Analyse, document and practice correct equipment and software settings prior to flight

Observations and Conclusions

Over-dependence on valid GPS time

- GPS start up time adds to ground-initialization time.
- GPS may become unavailable in flight (due to RFI) or on landing (due to shielding).
- A valid GPS fix at precisely 1 second after the minute is required to initiate any transmission.
- Start the payload at the minute-rollover. Initialise a free-running, one-minute Arduino timer.
- Synchronise the Arduino timer to the GPS timepulse output when valid GPS time is available.
- Continue transmission of legacy data in the absence of a GPS fix, but indicate No Fix.

Transmit frequency drift due to temperature variation (400Hz – target 10Hz)

- DDS TXCO frequency is not stable over the wide temperature range on ground and at altitude.
- Attach a temperature sensor to the DDS TCXO and provide a frequency correction.
- Use GPS 1pps time pulse and an Arduino counter to measure/correct the DDS frequency.

Excessive payload assembly time (10 hours – target 1 hour)

- Multitude of PCBs and Veroboards require complex, time-consuming assembly.
- Design a single PCB motherboard incorporating all Veroboard components

Observations and Conclusions

Excessive internal temperature (53°C) at ground level

- Due to power dissipation, polystyrene insulation and duration of ground operation.
- Implement over-temperature protection: Power down GPS and DDS if temp too high.

Excessive payload cost (\$82 – target \$50)

- Batteries (\$20), GPS (\$15), Arduino Pro Micro (\$9).
- Try 2xAA instead of 4xAA. Look for other suppliers. Bulk purchase.

Excessive payload weight (150g – target 100g)

- Batteries (60g), Styrofoam (16g), GPS Antenna (15g).
- Try 2xAA instead of 4xAA. Use less Styrofoam. Try printed QFH antenna.

Excessive power dissipation (1W – target 0.5W)

- Telemetry transmission too frequent in high-altitude flight
- Reduce the rate of telemetry transmission in high-altitude flight.
- Turn off GPS and DDS, between transmissions. Use an external 6-minute power-down timer.
- Resume hi-rate telemetry transmission below 5000m.

Observations and Conclusions

HABITAT Flight Statistics not available

- HABITAT flight statistics only provided for approved flights
- HABITAT flight documents were submitted, but the irc approval request was not.

Unknown power output

- Too much variation in measurements using uncalibrated equipment.
- Perform measurement on calibrated equipment.

Fast, spiral decent

- Poor parachute design and asymmetrical attachment
- Re-design and test.

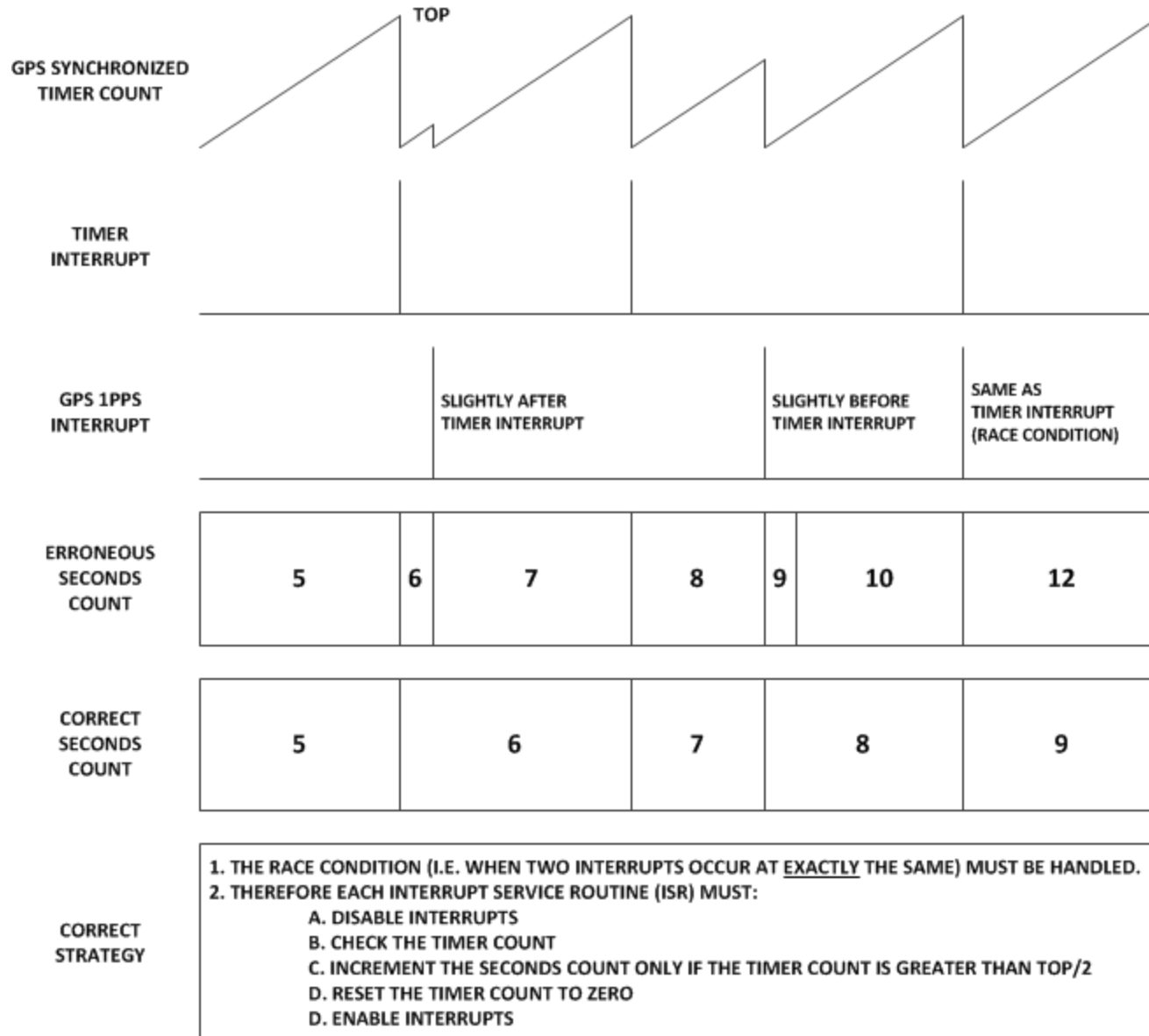
Inefficient DC power conditioning.

- A suitable configuration of step-up or step-down converters was not achieved.
- A linear LDO voltage regulator was used instead.
- Re-design and test.

No video recording facility.

- \$5 eBay spy cam did not arrive on time.
- Re-order another unit.

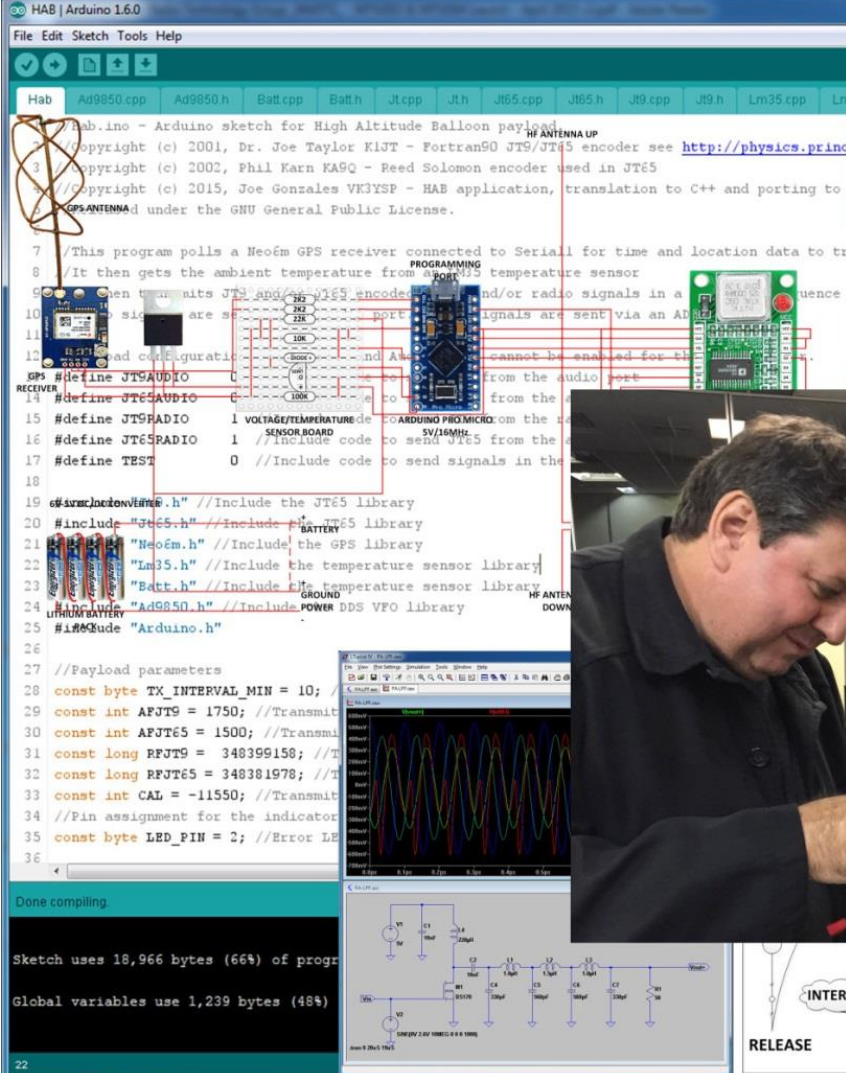
A GPS-SYNCHRONIZED ARDUINO TIMER



Melbourne Amateur Radio and Technology Group (MARTG)

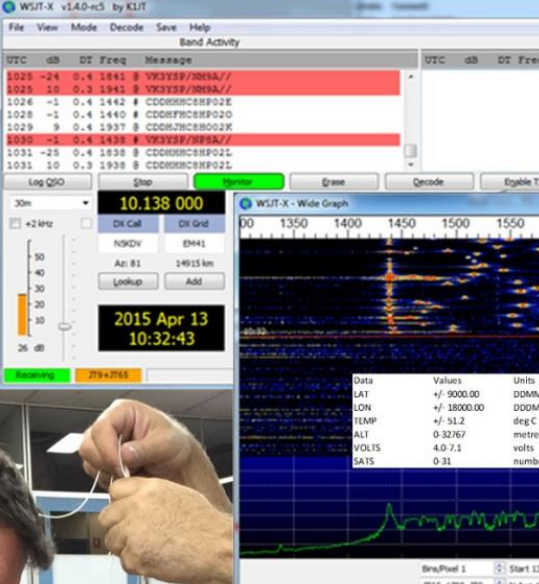


Finally - The End - Thank You




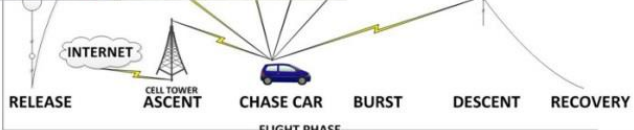
```
1 Hab.ino - Arduino sketch for High Altitude Balloon payload
2 //Copyright (c) 2001, Dr. Joe Taylor KI1JT - Fortran90 JTG5 encoder see http://physics.princeton.edu/~jtd/jtg5/
3 //Copyright (c) 2002, Phil Karn KA9Q - Reed Solomon encoder used in JTG5
4 //Copyright (c) 2015, Joe Gonzales VK3YSP - HAB application, translation to C++ and porting to
5 //GPS ANTENNA under the GNU General Public License.
6
7 //This program polls a Neofm GPS receiver connected to Serial for time and location data to tr
8 //It then gets the ambient temperature from an ADXL345 temperature sensor
9 //Then transmits JTG5 and JTG5 encoded data and/or radio signals in a
10 //sequence
11 //signals are sent via an ADXL345 temperature sensor
12 //signals are sent via an ADXL345 temperature sensor
13 //signals are sent via an ADXL345 temperature sensor
14 //signals are sent via an ADXL345 temperature sensor
15 //signals are sent via an ADXL345 temperature sensor
16 //signals are sent via an ADXL345 temperature sensor
17 //signals are sent via an ADXL345 temperature sensor
18 //signals are sent via an ADXL345 temperature sensor
19 //signals are sent via an ADXL345 temperature sensor
20 //signals are sent via an ADXL345 temperature sensor
21 //signals are sent via an ADXL345 temperature sensor
22 //signals are sent via an ADXL345 temperature sensor
23 //signals are sent via an ADXL345 temperature sensor
24 //signals are sent via an ADXL345 temperature sensor
25 //signals are sent via an ADXL345 temperature sensor
26 //signals are sent via an ADXL345 temperature sensor
27 //signals are sent via an ADXL345 temperature sensor
28 //signals are sent via an ADXL345 temperature sensor
29 //signals are sent via an ADXL345 temperature sensor
30 //signals are sent via an ADXL345 temperature sensor
31 //signals are sent via an ADXL345 temperature sensor
32 //signals are sent via an ADXL345 temperature sensor
33 //signals are sent via an ADXL345 temperature sensor
34 //signals are sent via an ADXL345 temperature sensor
35 //signals are sent via an ADXL345 temperature sensor
36 //signals are sent via an ADXL345 temperature sensor
```

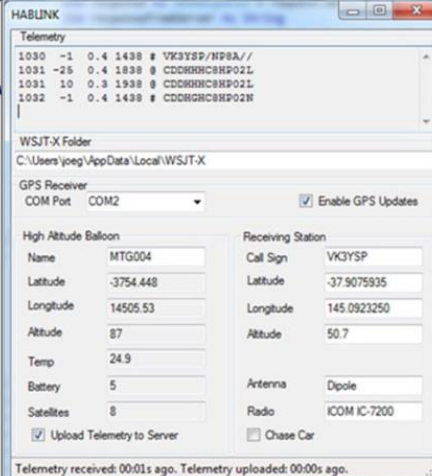
Sketch uses 18,966 bytes (66%) of program memory. Global variables use 1,239 bytes (48%) of dynamic memory.



Data	Values	Units	Resolution	Int Format	Characters	Symbol
EAT	+/- 9000.00	DDMM.mm	0.001 minute	+/- 900000	5	TTTT
LON	+/- 18000.00	DDMM.mm	0.01 minute	+/- 1800000	5	NNNN
TEMP	+/- 51.2	deg C	0.1 degree	+/- 512	2	DD
ALT	0-32767	metres	1 metre	0-32767	3	AAA
VOLTS	4.0-7.1	volts	0.1 volts	0-31	1	V
SATS	0-31	number	1 satellite	0-31	1	S







Telemetry	
1030	-1 0.4 1498 # VK3YSP/HBA//
1031	-25 0.4 1938 # CDDHHC8HP02L
1031	10 0.3 1938 # CDDHHC8HP02L
1032	-1 0.4 1498 # CDDHHC8HP02N

Receiving Station	
Call Sign	VK3YSP
Latitude	-37.9075935
Longitude	145.0923250
Altitude	50.7
Temp	24.9
Battery	5
Satellites	8
Antenna	Dipole
Radio	ICOM IC-7200