Operation Short Term Observational Radio Communication

James Murdock, John Wells, Kyle Hutchings, Joseph Grantham, Neil Davis

School of Computing, University of Utah

THE UNIVERSITY OF UTAH

Introduction

Operation STORC consists of a weather balloon with an attached payload for aerial weather observation and communication. The payload contains a software defined radio (SDR) transceiver for communication with a ground station and the transmission of weather sensing data. The hardware component is comprised of the SDR interfacing with a micro-controller (MCU).

The ground station consists of a SDR and a computer terminal. The SDR attached to the ground station will receive the data transmitted by the balloon payload and pass that data to the computer terminal, where the software component will be installed. The ground station runs a software application that predicts the approximate landing zone of the balloon based on available weather data.

Software Prediction (pystorc)

The software component of project STORC was designed to predict, track, and approximate the landing location of the balloon, and provide real-time path tracking to the team via Google Earth through the Keyhole Markup Language (KML).

Communication

Communication with the balloon is established using the Automatic Packet Reporting (APRS) over a variety of different mediums. Packets received from the balloon are decoded and sent to the plotting software through Serial, TCP, or UDP.

Weather Evaluation

Weather patterns are, by nature, unpredictable. Project STORC aimed to eliminate some of the guesswork associated with a balloon launch, and gather an approximate landing location. Knowing the landing location would allow the team to meet at the landing site while the balloon was descending, as well as pick a suitable launch location.

Eq 1: Lateral Wind Speed

$$V_a = \langle -\overline{u}\cos(\theta), -\overline{v}\sin(\theta) \rangle$$

Eq 2: In-flight Ascent velocity

$$V_a = \frac{8rg}{3C_d}\left(1 - \frac{3m}{4\pi\rho r^3}\right)$$

Eq 3: Air density at altitude

$$p_{air} = e^{\left(-\frac{Mgh}{RT}\right)}$$

Eq 4: Descent velocity

$$V_a = \sqrt{\frac{L_F}{.5C_d p_{air} A}}$$

Eq 5: In-flight Ascent velocity

$$V_a = \frac{8rg}{3C_d}\left(1 - \frac{3m}{4\pi\rho r^3}\right)$$

Eq 6: Balloon Radius at Altitude

$$r = \sqrt[3]{\frac{3p_1V_1}{4\pi p_2}}$$

Data Sources

Weather data is gathered from two separate sources: NOAA winds aloft and GFS wind data provided by the University of Wyoming. Both data sources are updated every 6 hours, and serve as a reasonable approximation for anticipating launch conditions.

Path Plotting and Correction

Once the initial prediction is calculated, the balloon is launched and the software waits to gather any information sent back from the balloon. Packets are gathered in triplets, and evaluated.
Erroneous packets are detected and discarded by calculating the variance of the triplet. Should the variance exceed a user-specified value, the packet is discarded. Valid triplets are sent to Google Earth and plotted as they occur.

Real-Time packet data plot from Oct 14th balloon launch

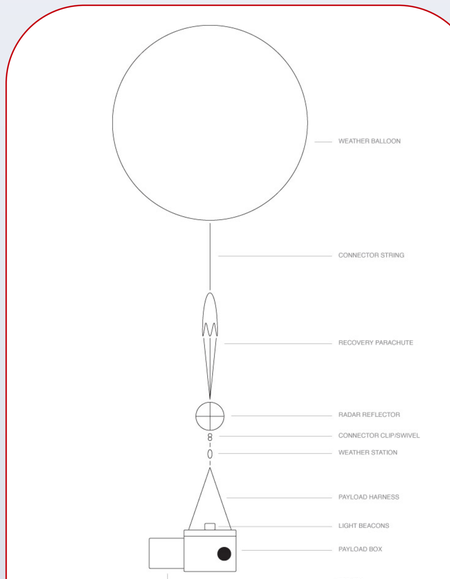
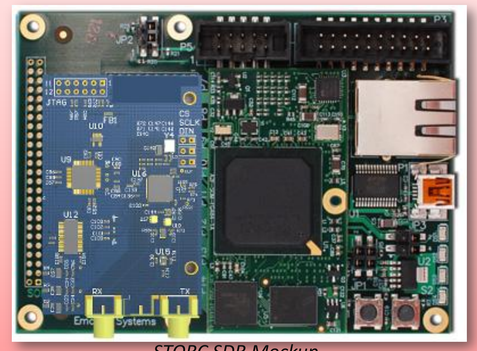


Illustration of balloon with attached hardware



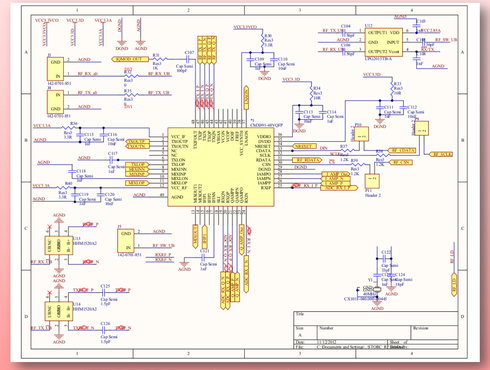
STORC SDR Mockup

STORC SDR

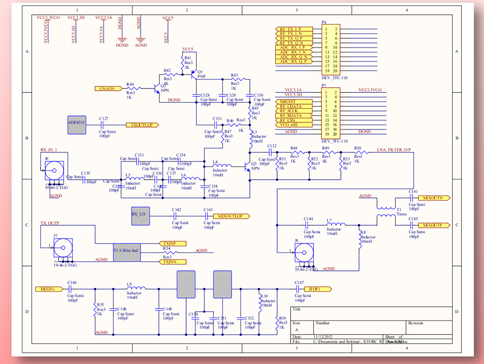
The STORC SDR originally used a design created by the University of Michigan and provided by professor Thomas Schmid. This design contained known errors, was too cumbersome, and did not operate at the frequencies STORC desired.

Professor Schmid later provided information about the Emcraft SOM that we could use to simplify our design. The main board is a modification of the Emcraft evaluation board to support an additional set of board connectors for the RF Daughter Card.

The new RF Daughter Card module consists of a CMX991 Quadrature module for operation from 100 MHz – 1GHz. The RF card is designed to both transmit and receive at the same time on two separate frequencies.



STORC RF Daughter Card Schematic (1)



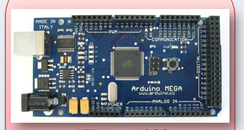
STORC RF Daughter Card Schematic (2)

Automatic Release Mechanism

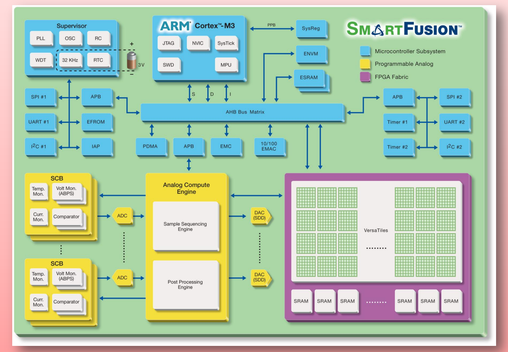
In order to keep the balloon from drifting and landing in these inaccessible areas, we decided to implement an automatic balloon disconnect circuit so that we could control just when the balloon would disconnect. The circuit did not need to be that complicated. It would consist of an Arduino processor and a heating element to be triggered by the processor. We wanted to be able to define a Maximum Altitude for the balloon to achieve and at that point a heater would be turned on to melt the zip tie attaching the payload and parachute to the balloon.

Arduino Setup

The setup of the Arduino consists of a serial input coming from the GPS transmitter, a software parser to separate out the information, and an output pin to act as the disconnect signal. We found that not only could we define a maximum altitude, but a latitude and longitude as well. In this way, we were then able to define a geographical box for the balloon to operate within. If the balloon leaves this box, the circuit disconnects the payload from the rest of the balloon.



ATMega1280
Used for the Oct 14th balloon launch



SmartFusion block diagram
A microprocessor/FPGA to be attached to the main board for future launches.

SmartFusion SoM (System-on-Module)

The key part of our embedded software component is the software written for the SmartFusion SoM. The SmartFusion has an Arm Cortex-M3 processor as well as an FPGA fabric. This enables us to integrate both software and digital logic into our setup. The main purpose of the SmartFusion SoM is to communicate with the sensors and transmit that information back to our base station. The built in SPI and I^2C bus interfaces allow us to integrate our system with the sensors more easily.

The software on the SmartFusion is set up as a web server while the ground station is set up as the client. In this way the ground station can make requests and send commands to the SmartFusion for sensor data or possibly to tell it to disconnect the payload from the balloon.