**Overview**

This document shows how each component in and outside of the Rocket communicates with each other.

Radio Serial SCM

GPS

Flight Computer

Transceiver

ECU

Sensors / Valves

USB – Serial NEMA-0183

Two Wire Serial -  
SCM

Two Wire Serial - SCM

Analog/  
Digital IO

Command Box

Two Wire Serial - SCM

**Simple Checksum Messaging – SCM**

Our 2 Wire protocol is ASCII based.

ID | Data | Checksum

XX , XXXXX , 00 ;

**ID** (2 Characters): is the ID for each packet, IE HB for heartbeat or OK for ok  
**Data** (5 Characters): What data is contained in the packet, varies based on packet  
**Checksum** (2 Characters): The sum of all ASCII codes of the message modulo 100

We decided on our self-designed SCM, because the signals we send are heavily impacted by the noisy environment which can lead to a corruption of the message. With SCM, each component calculates the checksum when sending and receiving a package, therefore comparing the received package’s checksum with the self-calculated one. If they don’t match, a request is sent to repeat the package. Since we are taking modulo 100, there is 99 percent chance that a corrupted package will get noticed and corrected.

**Package Types:**

VS: Valve State, we send 5 Digital H/L one in each bit of data   
HB: Heart Beat, repeat back data received / create random data on master  
T0-9: Thermocouple 0-9, 10 bit value in data field  
P0-9: Pressure Transducer 0-9, same   
RP: Repeat Package  
ER: Error  
OK: sent every time a package is received and executed

**Breakdown components**

**Command Box (Arduino Uno)**

The Command Box can toggle valves manually and receives data through radio.

VS - Send only  
HB - Send and Receive  
T0-9 - Receive Only  
P0-9 - Receive Only  
RP: Receive Only  
ER: Send and Receive  
OK: Receive Only

**Engine Controller Unit – ECU (Arduino Mega)**

The ECU reads sensors, valve states and ECU Battery Voltage and forwards them to the Flight Computer in SCM format. It also receives valve states from the command box and flight computer and executes them.

VS - Only Receive  
HB - Only Reply  
T0-9 - Send Only  
P0-9 – Send Only  
RP: Send only  
OK: Send only

**Flight Computer (Raspberry Pi)**

The Flight Computer does the bulk of the calculations:

* Decode NEMA from GPS
* Packet Generation for ECU and Transceiver
* Create Heartbeat for ECU
* Create Heartbeat for Transceiver
* Calculate velocity from GPS Altitude
* Deploy Parachute at Apogee (from velocity)
* Shut off Engine at certain point.
* Report Position and Cross Track Error to Transceiver
* Reading Battery Voltage

The ID affiliations equal the ones of the Command Box.

**GPS (design based on Holme)**

The GPS implements NEMA packet generation, serial sending of NEMA packets and integration of NEMA with the rest of the GPS.

NEMA codes we are using:

* GGA, //Global Positioning System Fix Data (This is the important one)
* GLL, //Geographic position, latitude and longitude (and time)
* GSV, //Satellites in view
* HDT, //Heading, true north
* VTG, //Track made good and ground speed
* XTE //Cross-track error

**Transceiver (Arduino Pro Mini)**

The transceiver sends all the data via Radio to the Command Box.