1. Bonus Challenge

The bonus challenge requires the team to incorporate a 2.4 GHz XBEE Pro radio module (50mW to 63mW, one mile range) to establish uplink/downlink communications prior to and during flight. The team tested the XBEE system during the test flight, as detailed in the document below.

1.1. Bonus Challenge A: Command Uplink

The team will use a custom Arduino serial interface to setup the rocket. After the vehicle is vertically mounted on the launch pad, an operator will enter an access code to begin rocket configuration and select the flight profile, either Flight 1 or Flight 2.

If Flight 1 is selected, the vehicle will enter standby mode, a state where acceleration is used to determine liftoff. The vehicle will then follow the procedures for Flight 1, as documented in the main FRR document.

If Flight 2 is selected, the operator will then be prompted to enter roll commands in "target orientation, roll direction, hold time" format until all commands are entered. At any point in this process the user may choose to reset and restart, or skip to launch detection while retaining completed entries. Upon skipping or completion of entries the rocket will enter standby mode in the same manner as Flight 1.

The payload team has achieved communication between the XBEEs by implementing a small modification to the code since PDR. Software Serial now utilizes pins (0,1) as TX and RX, as opposed to (2,3), allowing the Arduino to receive data from the XBEE Shield. A new instruction to switch the XBee shield from "USB" to "XBEE" has been included in launch procedure to ensure successful communication. The team concluded that reason previous communication tests failed due to the combined errors in pin setup and shield configuration.

During test flight the command uplink process was not tested due to control code errors.

1.2. Bonus Challenge B: Data Downlink

Data from the 9-DoF – including vehicle orientation, angular velocity, and roll attempts – will be transmitted via the XBEE communication system to ground in the previously specified sequence. They will also be recorded on an onboard SD card. Data will be sent at 25ms intervals, with a new line distinguishing data between intervals, and the respective security code before each line.

Transmitted data would be recorded via the XCTU interface. This procedure was tested during test flight, with orientation, angular velocity, and roll control magnitude being transmitted to the ground computer from launch detection to landing without interruptions. Portions of the data were lost due to lack of real-time writing to a local file and the limited capacity of the XCTU interface, which will be fixed by simply selecting the "record session" option provided by XCTU.

1.3. Bonus Challenge C: Crosstalk

A segment of code will be activated after liftoff is detected to check for data uplink during each Arduino loop. If a string with the correct security code and dimensions is detected (two-character security code + 5-character equation) from XBEE readings, the string will be further broken down for analysis and processing. The results would then be sent back in string format with the security

code prefixed once more. This process was not verified during test flight, but may be easily examined during

2. Network Setting

The actual settings of the network remain as described in the Preliminary Design Review:

> Data transmission is achieved by two XBEE-Pro radio modules. One radio module is on the rocket, and the other one is connected to a computer through an Arduino on the ground. The onboard Arduino will collect data from the 9-DoF and send it down to the computer on the ground through the radio module. Communication between the two radio modules is setup by manipulating the settings of the radio module. Usually, a network needs a coordinator to work. In this case, however, XBEE Pro can accomplish peer to peer communication in a network without a coordinator. Thus, the two XBEE Pro radio modules are all set to be endpoint rather than a coordinator and a router. In addition, two XBEE pro radio modules should have the same PAN (personal area network) ID to pair up. Basically, the XBEE Pro module will first broadcast its information to find modules in the same network. The team set the PAN ID to 3591 in an effort to prevent having the same PAN ID as other teams. This number avoids many intuitive patterns such as the one in 5555(a repetitive set of numbers), 9909(someone's birth date), etc., that overall makes certain ID's more likely to appear. Moreover, the PAN ID can be reset at the moment of the launch if the same ID as another team through the XCTU software, a GUI used to set up the radio, provided by digit.

Both XBEE radio modules are set to endpoint. The PAN (personal area network) ID is set to 3591. Everything else remained under its default setting.

3. Configuration

Modified since the Preliminary Design Report, only the XBEE onboard is incorporated with an Arduino. The XBEE on the ground is connected to the computer simply through a USB adapter. The rest of the configurations have not changed. The Arduino onboard is connected to the XBEE shield to avoid wires disconnecting during the flight or simply cluttering the avionics bay. The ground XBEE and adapter reads and prints the signal sent from the onboard Arduino. The onboard Arduino reads and writes the signal sent from the ground to adjust the rocket's maneuvers. This system grants the ability to reset the maneuvering direction of the rocket before launch and cross-communication during flight.

4. Range

The XBEE hand book claims that the ideal working distance is one mile. During the test flight, the ground laptop continuously received data from the XBEE. However, during ground testing it was found that signal would be lost even if the two devices were separated by a wall. These results provide strong enough evidence that the XBEE Pro radio module is capable to carry on communication throughout flight, but measures must be taken to ensure clear line-of-sight from the ground XBEE to the rocket.