RockSAT-X 2014 Final Report

Northwest Nazarene University

Department of Physics and Engineering

Mission

1. To fly and deploy proof-of-concept deployable sensors mounted on 3D printed airfoils
2. To test the survivability of Flexible Electronic Microcontroller Units from American Semiconductor Inc.
3. To test a fully flexible sensor array from NASA’s Jet Propulsion Laboratory and XEROX PARC

Faculty Advisers: Dr. Dan Lawrence and Dr. Steve Parke

Student Team: Drew Johnson, Lukas Rieke, Alex Hanson, Devon Ellis, Connor Back, Erik Anderson, Jaron Hush, Brenton Peck, Matthew Spires, Cassie Wade, Lucas Schamber, and Dakota Martin
Mission Statement
The mission for the Northwest Nazarene University RockSAT-X 2014 (NNU RSX14) payload was a threefold mission. The first mission involved custom 3D printed airfoils designed by NNU students containing optical sensors. These airfoils would be deployed using a revolving mechanism for smooth deployment, and the airfoils were designed to orient in the atmosphere and descend in a smooth and controlled manner. These sensors would relay information back to the payload using XBEE RF modules, and serve as proof-of-concept for high altitude deployable atmospheric sensors.

The second mission involved long time NNU RSX partner American Semiconductor Inc. (ASI) and their cutting edge FleX Microcontroller Units (MCU). As the first fully flexible MCU, ASI desired data on the survivability of the FleX MCU. NNU students designed custom 3D printed airfoils for the FleX MCU, and ASI designed a flexible stamp with the FleX MCU and an LED connected to the crystal clock oscillator. This LED would blink back a signal similar to an electronic heartbeat that would indicate continued functionality of the FleX MCU. All of the airfoils were to be recorded by a Hack HD camera directly above the point of deployment for the airfoils. This camera would record video of the electrical heartbeat of the FleX MCUs and also indicate the stability of the airfoils.

Lastly, the third mission involved a fully flexible sensor array made by XEROX Palo Alto Research Center (PARC) through a contract with NASA’s Jet Propulsion Laboratory (JPL). This flexible sensor array is a prototype of a project headed up by Kendra Short at JPL called the STructure of the Atmosphere: Network Lander Experiment, or STANLE. This prototype contained circuitry printed on a flexible Kapton sheet, and would demonstrate the usability of similar flexible sensor arrays for deployment over extraterrestrial bodies, similar to confetti being tossed in the air. With thousands of these STANLE sensors, large scale data reconnaissance could be achieved without flying a larger, more expensive satellite.

Mission Requirements
In order to deploy the airfoils integrated with XBEE sensors, the XBEE Airfoils, and also the FleX MCU airfoils, the ASI Airfoils, a simple deployment mechanism was mandated. After contemplating several design strategies, including an extendable arm and a magazine-cartridge style mechanism, the final design was chosen to be a revolving system. This revolving system, to be expanded upon later, would allow for continuous deployment using a single motor. Each airfoil would have to be designed to fit inside one of the six chambers. The motor was activated by and powered off Timer Event 2, set to activate after the skin of the rocket had blown off.
The XBEE Airfoils would transmit back to the payload using the XBEE proprietary ZigBee mesh protocol. This allowed a single XBEE to remain on the payload, dubbed the mothership XBEE, and all the deployed XBEE airfoils would communicate with the mothership through radio frequencies. The Arduino Mega on the payload would read the data coming in from the deployed XBEEs and, after parsing the data, save the data onto an internal SD card.

Since the data from the ASI Airfoils would be a visual signal and the stability of the airfoils was also a mission metric, a Hack HD camera was placed right above the point of deployment. The Hack HD was set to record one minute video clips and save onto an internal SD card after each completed minute long video. The Hack HD (HHD) was controlled by the Arduino Mega MCU that would connect the HHD switch line to ground for 100 milliseconds to activate the camera. The HHD was powered by Timer Event 1 and the Arduino used Timer Event 1 as a signal to activate the HHD.

For the STANLE aspect of the experiment, the sensor array and waterproof housing were designed and machined by JPL and PARC. The footprint for the design was created in communication with the NNU team, in an effort to make sure the payload still had room for all necessary equipment. The STANLE system was powered by the NNU payload’s Timer Event 1 that would activate at launch.

**Payload Design**

The most constraining component for this payload was the revolving deployment system. This system would determine the size of airfoils and constrain the motor. In order to maximize airfoil size, the revolving system was designed to occupy the form factor of the payload bay and thus constitute the first floor of the payload. All the electronics would be housed above this revolving system. Several views of the final deployment system are given in the following figures.
Figure 2: SolidWorks Rendering of Revolving Deployment System

Figure 3: Fan Blades for Revolving Deployment System
The fan blades in Figure 3 would rotate around the central axis in a counter-clockwise motion from this angle. There are two fixed guide tracks, on top and on bottom of the deployment chamber, that will create a scissor effect and deploy the airfoils that come in contact with the guide tracks. This system allows for continuous smooth deployment of the airfoils.

The airfoils had to be designed with orientation in mind. In other words, the center of mass had to be located towards the bottom of the airfoil, so that the airfoil would orient in a controlled manner. The airfoils were originally modeled after maple seeds, but were scaled up in order to incorporate larger electronics. The airfoils also had to remain symmetrical to avoid tumbling upon deployment. The first style, the Plant Style airfoil for the ASI airfoil is shown in Figure 5.

Figure 6, with the FleX MCU mounted on a leaf in Figure 6. The FleX MCU stamps were adhered using a UV-cured adhesive by ASI after NNU students printed the airfoils on a 3D printer.
The large cylinder at the base of Figure 5 was a container for the batteries, located at the bottom so the blinking LED opposite the battery would face the camera. This design was tested by students tossing and dropping the airfoils at various intensities and orientations to find the optimal balance so the LED would point opposite the direction of travel.

The XBEE airfoils were more of a hassle to design. As with the ASI airfoils, batteries had to be installed so that the electronics would last 10 days, with six days being the longest period the airfoils would be left alone and active in the launch pad. Since the FleX MCUs required minimal power, roughly 1 mA, a small Li-ion CR13N 3V battery with 160 mAh was selected. This battery would have half the
capacity needed and would be wired in parallel with a second battery in the cylinder towards the base of the ASI airfoils, allowing for double capacity.

The XBEEs required substantially higher power than the ASI airfoils, around 35 mA, putting the required capacity for 10 days at 8400 mAh. The battery found was a larger ER26500 Ultralife battery with a 9 Ah capacity at 3.6 V. This in turn forced the airfoils to accommodate a larger battery and a larger sensor, as the XBEEs were roughly the size of a quarter compared to the tiny FleX MCUs.

![Figure 7: XBEE Airfoil Top View](image)

![Figure 8: XBEE Airfoil Side View](image)
The HHD box was positioned directly above the deployment region and was comprised of two parts: a back shell and a face plate. The face plate had a one inch diameter hole positioned for the camera lens and a lens cover that sealed a two inch diameter borosilicate glass lens over this hole. The HHD camera was controlled through six leads connected to the screw terminals in the HHD. The six leads consisted of power (3.3V), three ground lines, a switch line, and an LED line. These lines all originated from a 19 pin Amphenol connector that went to several different locations. An exploded view of the HHD box is shown below.

![Figure 9: Hack HD Camera Enclosure Rendering](image)

The HHD firmware allows the user to control the length of video files down to one minute increments as well as an override feature. Since the HHD would be powered on from T+01 seconds to T+320 seconds, a length of five minutes and 20 seconds, the HHD was set to record videos for a minute and then save each video file at the end of the minute. The rear of the HHD is shown in Figure 10, with the front view shown again in Figure 11.

![Figure 10: Back of Hack HD Enclosure](image)
The HHD was controlled by an Arduino Mega 2560 microcontroller, as was the entire payload. The Arduino had a wireless SD shield that allowed the Arduino to not only save data onto the SD card but also communicate with the airfoil XBEEs. XBEEs use a non-standard pin pitch so the Arduino shield had a designated spot and pin outs for an XBEE to reside on the shield securely. The serial output of the XBEE was sent to the Serial1 line on the Mega, and a string of 19 hexadecimal doubles, and the parsing of those values will be discussed in subsequent sections.

The Arduino had a series of digital inputs and outputs all integrated into a digital signal bus at the end of the Arduino. These lines included a remove-before-flight (RBF) digital in/out line, HHD switch line, TE1, TE2, and lines designated for telemetry. This digital bus was connected via a 26 pin rainbow cable to the 32 pin circular Amphenol connector shown in Figure 12. The Arduino was housed inside the red anodized box in Figure 13. The Arduino would mount directly to the aluminum deck and the enclosure would drop in over the electronics onto a layer of quarter inch rubber gasket material. This gasket material would create a waterproof seal and protect the Arduino.
The last critical element of the payload was the STANLE enclosure, shown in Figure 14. This was designed by JPL to fit a platform given by NNU as to not exceed the payload region and mass budget. The enclosure included a four wire Glenair Mighty Mouse connector, chosen for its small size, through which the NNU team would supply a 3.3V signal to power the STANLE sensor array. The full ideal STANLE would consist of the 25x30 cm system detailed in Figure 15, however the sensor system that flew was closer to a form factor of 4x6 inches. The circuitry included an optical sensor, memory, and a MCU.
The following image shows the functional block diagram for the payload. A power distribution board was designed to decrease the NASA-supplied 30V to 12, 5, and 3.3V for various electronics on the payload. The first distribution board was a custom PCB, but the design proved far too clumsy and cramped to be truly effective. Voltage regulators were not able to bend over due to close quarters and were at high risk of being sheared from the rotational forces. If the regulators were epoxied, then they were difficult to remove or adjust later on. For the final payload, the voltage regulators were heat sunk and grounded to the side of the Arduino enclosure, as seen in Figure 17. This method proved thermally and structurally sturdy, but was difficult to fix and adjust when regulators were blown.
**Student Involvement**

The organizational chart for the NNU RSX14 team is given below.

![Organizational Chart](image-url)
Testing Results

As the XBEE’s data transmission was a crucial aspect, this was a priority for the team. As mentioned earlier, the XBEEs relay data into the Arduino as 19 hexadecimal doublets, and their definitions are given below. The Arduino reads these doubles as integers, so a hex pair 0xC3 would be read into the Arduino as 195 as shown in Figure 19. The Arduino would read and save the values according to the process shown in Figure 20.

![Figure 19: XBEE Parsing Legend for Arduino](image-url)
Given this method, several tests were performed to make sure the XBEEs would operate as desired. First, photodiodes were connected to the router XBEEs, those that would be deployed, and the coordinator XBEE was installed on the Arduino. The photodiodes were put through a voltage divider circuit to create a voltage signal between 0-1.2V, the upper threshold of XBEE analog input values. Two router XBEEs were used for this test: one with a constant 1 V supply on the input pin and the other with the photodiode and an alternating light source flashing above the XBEE. The output from the Arduino was parsed according to the origin XBEE and then plotted in the following MATLAB figure. The MATLAB code was sufficient at parsing the single data file containing all test data into separate arrays for each XBEE router and plotting the data from each router concisely.
A concern was raised with the decision to house the mothership XBEE inside an aluminum enclosure, essentially creating a Faraday cage and blocking all signals for the XBEE. In order to test this, the payload was powered up with a coordinator XBEE inside a fully machined and bolted Arduino enclosure, set to read the incoming XBEE data. Two XBEEs were powered with a 1V signal voltage on a moving cart that would walk a 100+ foot hallway while the router XBEEs transmitted their signal. The coordinator XBEE would record zero incoming data if the Arduino enclosure was a true Faraday cage. The results for this test are shown in Figure 22.
As the above figure would suggest, the XBEEs were able to communicate through the aluminum Arduino enclosure. This could be due to a high frequency of communication, 2.4 GHz, or a not-perfect seal and therefore a less-than-perfect Faraday cage allowing for radio communication to slip through the cracks.

In order to create a second level of redundancy for the data storage, the team investigated downlinking the XBEE data through telemetry lines. This is where the rainbow cable in Figure 12 became incredibly useful. The grey male header bus snapped into the Arduino digital bus and allowed for several lines to be sent outside the Arduino enclosure. The downlinking was broken up into two data segments. The shorter segment was a binary doublet corresponding to the router XBEE, while the longer segment was composed of 12 binary bits that would be binary conversions of the XBEE data. These signals were programmed to output when the Wallops telemetry read strobe line, a 1 KHz signal, was high and the lines would remain active for 1 ms for one read cycle.

![XBEE Downlinking Test](image)

The HHD was a fairly simple aspect of this project to control. Requiring a 5V signal to power the camera and a grounded switch, programming took only a few simple lines. The 5V power signal would be sent two places, the camera itself and a digital input on the Arduino. One second after reading the HHD power signal, the Arduino would connect the HHD switch line to a common ground shared by the Arduino and the HHD. This switch was grounded for 100 ms and then put high and this process was repeated to turn off the camera.
Mission Results

Unfortunately, a series of small communication gaps and miscalculations resulted in little to no data being recorded by the NNU payload. The HHD camera was loaded with a 2 GB memory card, under the calculation that roughly 200 one minute long video files, each at 9 MB, would fill the 2 GB memory card. The HHD enclosure was also difficult to access the memory card, as the front lid would have to be removed and the screw terminals unscrewed in order to remove and clear the memory card. For these reasons, the HHD memory card was not cleared in the days before launch. However, each minute long video on the HHD was stored in a 90 MB video file and only 20 videos filled the SD card. The overwrite setting on the HHD was also left off, meaning the last videos on the SD card were of testing in the Wallops integration chamber.

The XHED payload from Colorado was able to record video of several experiments by extending four cameras looking down the length of the rocket from the nose cone. These four cameras were able to momentarily capture video of the airfoils being deployed. Due to the rocket’s ACS being vented prior to launch, the rocket did not fully de-spin to less than 1 Hz as expected. The picture below shows the first ASI airfoil being deployed.

Upon further deintegration at Wallops Flight Facility, it was discovered that no XBEE data had been saved on the Arduino’s memory card. The only lines of code saved on the SD card were the headers printed at the start of every code, but no mission data was saved. Subsequent footage showed each of the airfoils being ejected in a tumbling manner, likely due to the constant spinning of the rocket. The fourth airfoil being deployed, an XBEE airfoil, had all of its internal circuitry ejected, as shown in Figure 26.
Upon recovery, every enclosure of the payload had taken on water. The JPL enclosure had a few gaps in its sealing, shown in pictures below. The STANLE array was rinsed with isopropyl alcohol to prevent any erosion to the components. The water damage in the HHD enclosure likely came from only a single layer of O-ring sealing, with a gap between the lens and the metal holder going unsealed. As for the Arduino enclosure, the screws holding in the Amphenol connector were barely long enough to reach into the enclosure, let alone create a good tight seal through the thick gasket material. It is believed that these two reasons caused water leakage in the enclosure.
Figure 27: NNU Payload Recovered from Mission

Figure 28: STANLE Enclosure Post-Launch
Figure 29: Gasket Seal for STANLE Enclosure Post-Launch

Figure 30: Arduino Connector Post Launch
Conclusions

Each of the five airfoils, two with the FleX MCU and three with XBEE circuitry, were deployed from the NNU payload, as determined from XHED video. Due to a miscalculation, the NNU camera did not record any mission video directly over the deployment region. The spinning of the rocket made it very difficult to determine if any of the airfoils oriented and stabilized after deployment. This condition also made it very difficult to determine if the ASI FleX MCU survived launch, as the airfoil tumbles through the XHED video for only a few seconds before leaving view.

As of May 22\textsuperscript{nd}, 2015, the STANLE enclosure has made it safely to JPL and is in route to PARC for further testing. There has been no news of data as of this date.

The Arduino memory card was also absent of mission data. As seen in Figure 26, the circuitry was ripped from one of the deployed XBEE airfoils. The battery and the XBEEs were held inside the airfoil with a combination of electrical tape and epoxy, creating a structure that was not very strong and likely failed due to the high accelerations of the launch or before. If the XBEEs had maintained connection up to launch, then the data files would have three minutes of time stamped data prior to launch. This is not the case, and it is concluded that the XBEE circuitry did not maintain a good connection.

Potential Follow-On Work

Currently, the NNU RockSAT-X 2015 team is working with ASI again to test a flexible analog-to-digital converter (FleX ADC) in more contained experiment. There will be no deployables, but the team is including a proof-of-concept CubeSAT motherboard that be extended outside the payload area and then relay information back to a personal ground station via amateur transmitter radio.
Benefits to the Scientific Community

The STANLE project, headed by Kendra Short out of JPL, has huge impacts to the scientific community. She envisions flying a satellite above an extraterrestrial body, such as Mars, and deploying thousands of flexible sensor arrays, like confetti being thrown into the wind. These arrays would have sensors like the block diagram shown in Figure 15 and relay information back to the satellite. This method would allow for non-invasive data reconnaissance of a planetary body if the sensor arrays were printed on UV degradable materials.

American Semiconductor Inc. has been pioneering flexible electronics for the past decade and revealed the first ever fully flexible microcontroller in the NNU RSX14 project. In working through this project, students were allowed to give valued feedback to the company on the MCU. In the past few months, ASI revealed a development kit for their FleXform-ADC, utilizing technology used in this project. The FleXform-ADC comes with a custom PCB and a flexible sensor region with the FleX ADC on board. Developers can obtain one of these kits and test their own flexible sensors in conjunction with the ASI FleX ADC to increase the production and implementation of flexible electronics into everyday life.

Lessons Learned

- Create secure connections for all aspects of the project
- Check all calculations regarding storage and power ad nauseam
- Test all sealing mechanisms
- Incorporate timeout procedures to note if system stalls or doesn’t read mission data
- Detailed procedures, while tedious at times, help when tensions are high and in high pressure situations
- The fastest way to get something done is to take your time and think carefully
- Don’t be afraid to request help when venturing into unfamiliar territory
- Make sure your team is adequately staffed and prepared at all aspects of the project
- Take advantage of all the time available