HASP Payload Specification and Integration Plan

Payload Title: Solar Ultraviolet Radiation Measurement Apparatus (SURMA)

Payload Class: Large

Payload ID: 12

Institution: Louisiana State University

Contact Name: Victor Fernandez-Kim

Contact Phone: (225) 400-8644

Contact E-mail: vfernandezkim@gmail.com

Submit Date: 4/24/15

I. Mechanical Specifications:

Table 1: Weight Budget

Component	Material	Quantity	Mass (g)	Uncertainty (g)	Classification
Lazy Susan	Aluminum	1	240	±2.5	Measured
Rotating Platform plates	PVC	2	420	±5	Measured
Legs	Aluminum	2	205	±20	Calculated
Servos and mounting	Assorted	2	80	±5	Measured
Foamular insulation	Polystyrene	1800 cm^3	80	±15	Calculated
Arduino[s] (stacked)	Assorted, PCB	5	174.5	±0.5	Measured
Interior Insulation	Aluminum/mylar	1200 cm^2	20	±10	Estimated
Bearing	Brass	2	17.5	±2.5	Measured
Axle Pipe	PVC	1	12	±1	Measured
Vertices structural supports	Aluminum	4	400	±80	Estimated
Box adhesives	Gorilla glue, tape, etc.	N/A	5	±3	Estimated
Camera and mounting	Assorted	1	6	±5	Estimated
Clear window	Cast Acrylic	1	10	±5	Estimated
Totals:			**1455**	**±151.5g**	

Mechanical Drawings

The payload has two main mechanical systems: the electronics bay and the mounting/orientation hardware. The electronics bay is composed of rigid polystyrene and will be equipped with thin aluminum bracers along the edges (**Figure 1**). The electronics bay will serve as the housing and thermal insulation for the internal electronic components. The internal electronics will be mounted to PVC paneling to prevent the components from shifting during flight. A servo motor will be mounted on the inside of the electronics bay box to provide elevation angle control. The models created are still incomplete and certain design aspects are still in progress. For example, the servo that will be mounted on the rotating plates (provides horizontal orientation control) (**Figure 2**) must be insulated within a small enclosure. This small enclosure that provides thermal protection is still in design and its specifications are still unknown.

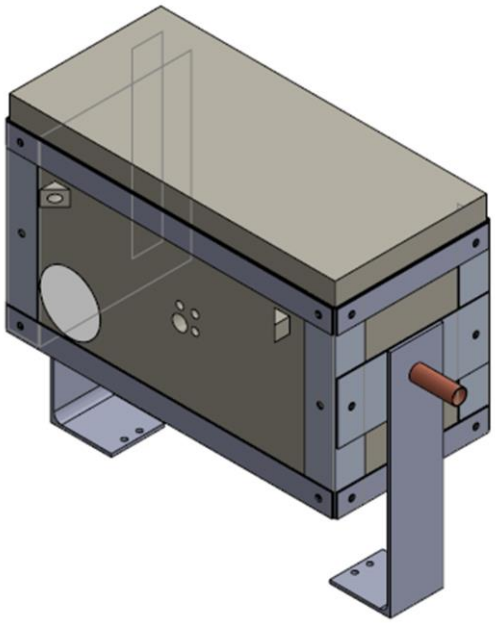


Figure 1: Electronics Bay Assembly with attached mounting. All scientific equipment will be located within the electronics bay and the servo motor controlling elevation orientation will be connected to an axle (on the other side).

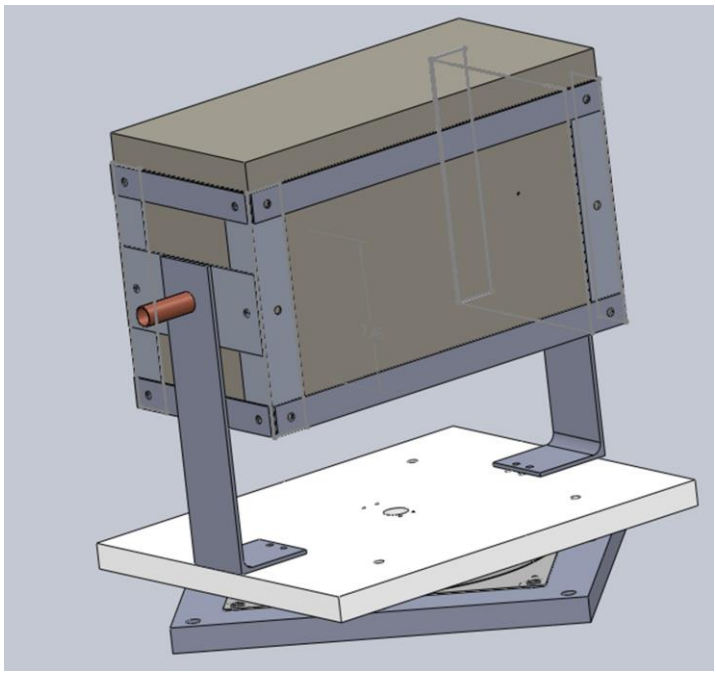
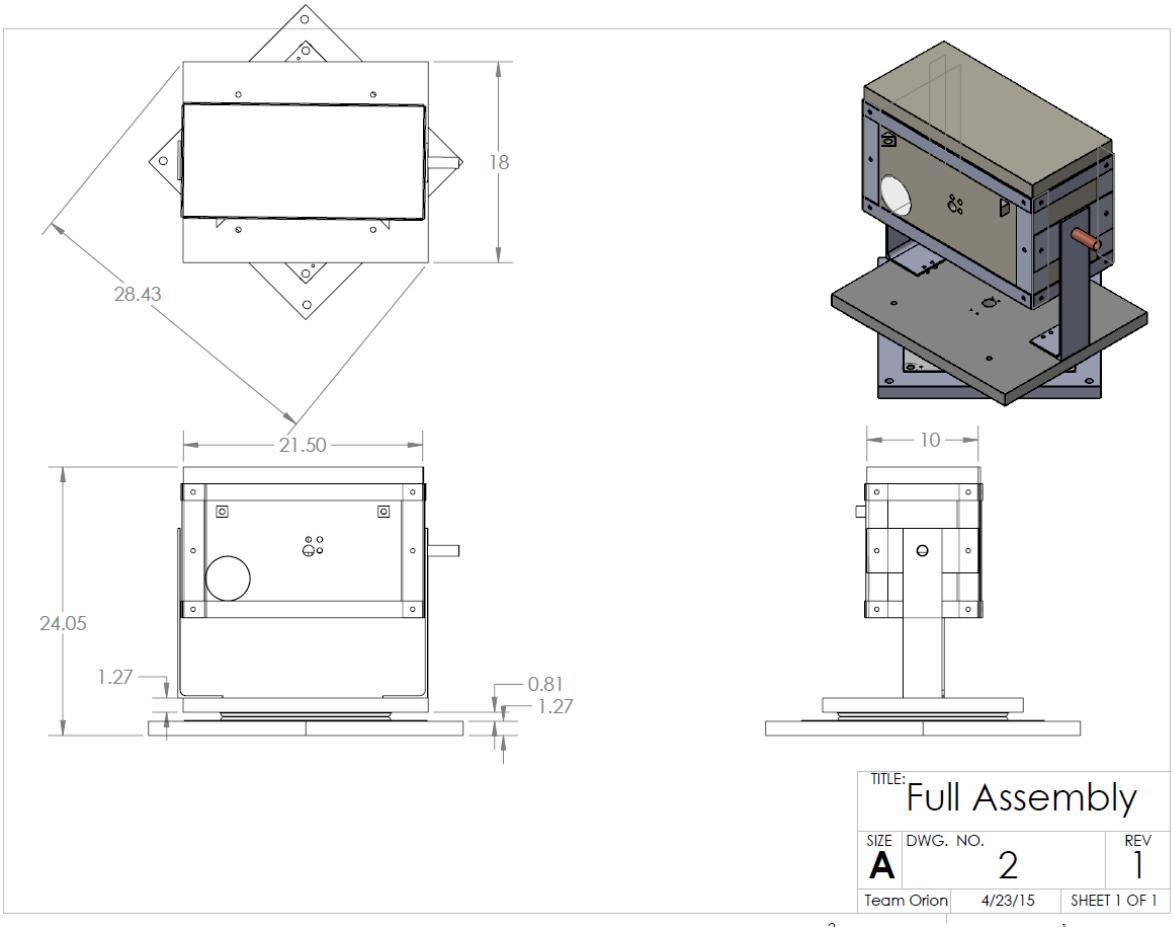
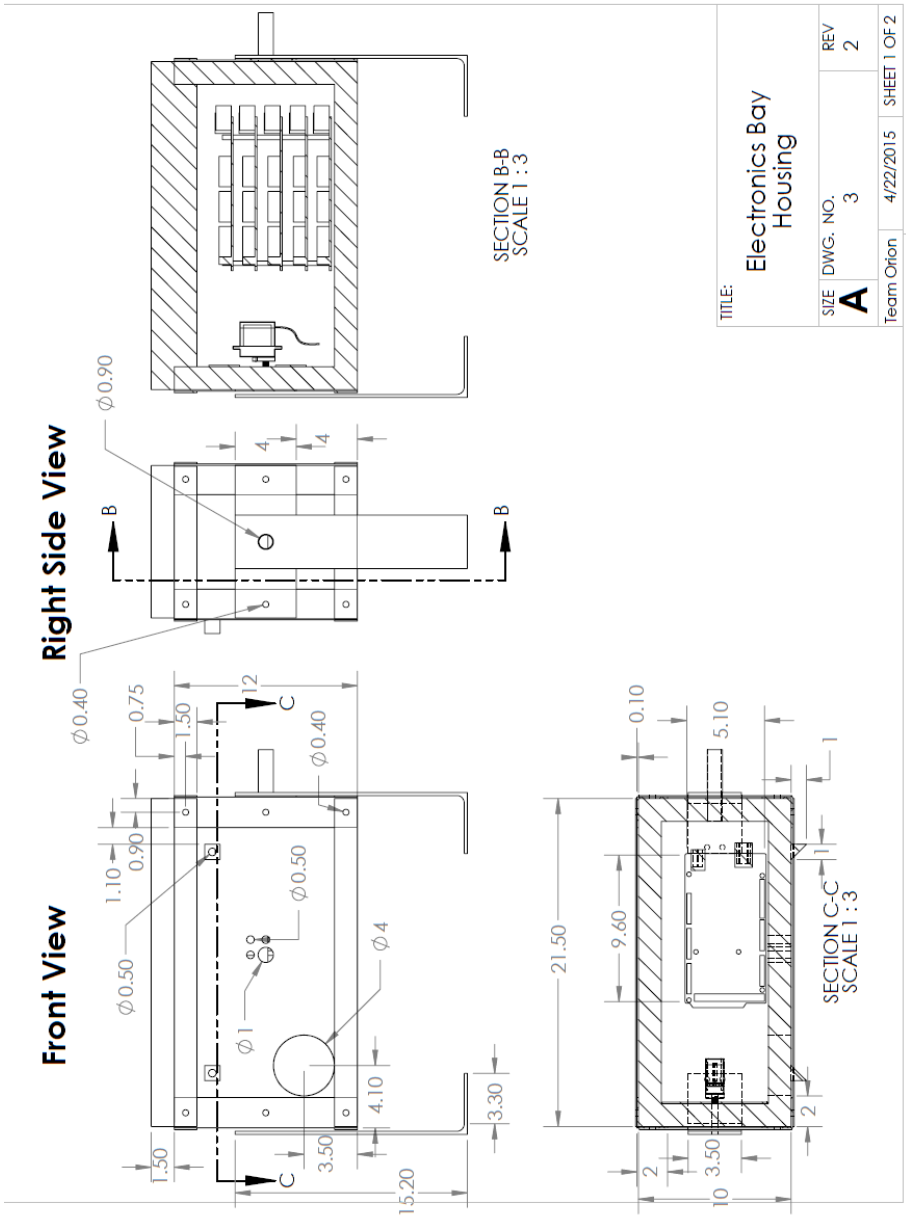
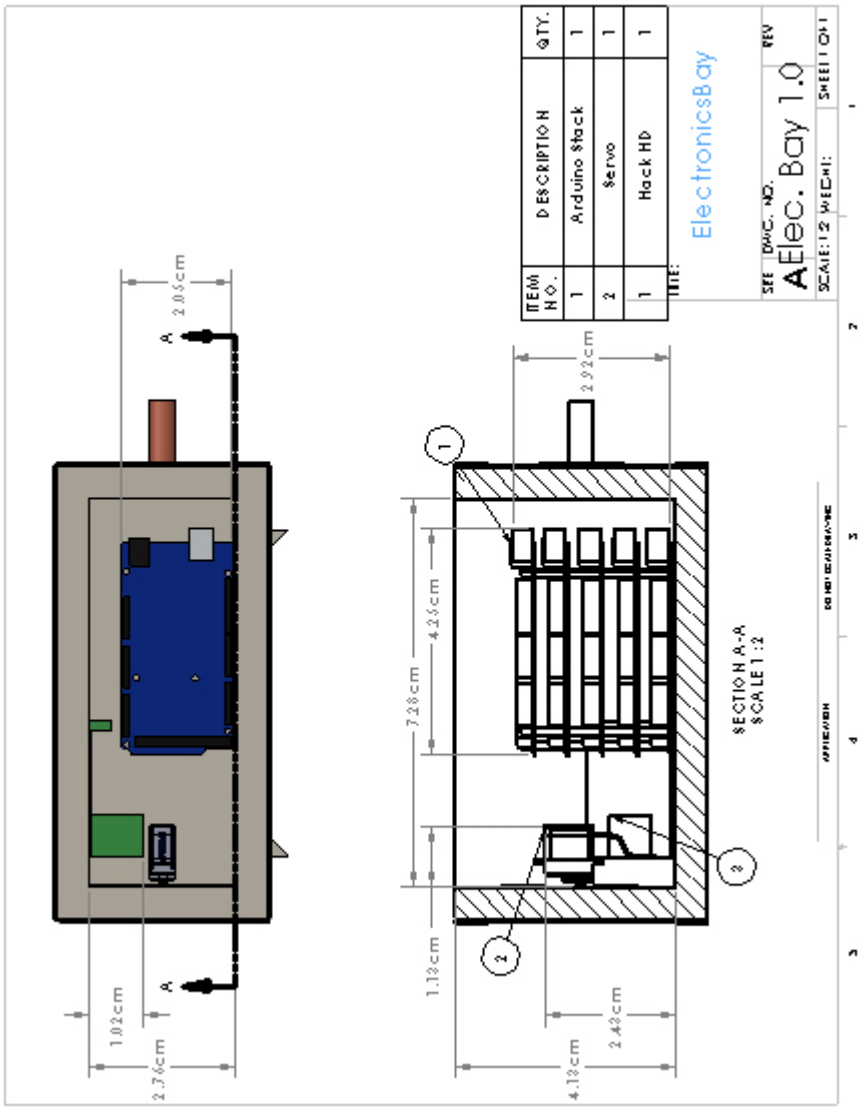


Figure 2: Electronics Bay mounted onto two rotational plates. A servo motor controlling horizontal orientation will be located directly under the electronics bay.

The holes in the bottom plate below the lazy susan will bolt through the mount. The power will be supplied by wires that will go through the middle of the lazy susan, then around the right side through the pipe axle into the box. The corners will be secured with sheet metal screws for structural support. There will be a hole in the front of the box filled with a clear plastic lens glued in for the use of the gnomon, with a camera behind it inside the box along the back wall. The internal structure housing will also be covered in mylar aluminum. The Arduino stack will be placed onto a thin layer of foam in the box so that it fits secured between the lid and the bottom of the box. **SURMA will not be flying anything potentially hazardous to HASP or the ground crew between or after launch.**




Right Side View
Front View
SECTION B-B
SCALE 1 : 3
SECTION C-C
SCALE 1 : 3
TITLE:
Electronics Bay
Housing
SIZE
A
DWG. NO.
3
REV
2
4/22/2015
SHEET 1 OF 2
Team Orion


ITEM NO.
DESCRIPTION
QTY.
1
Arduino Stack
1
2
Servo
1
1
Hack HD
1
ElectronicsBay
SIZE
DWG. NO.
REV
Elec. Bay 1.0
SCALE: 1:2 WEIGHT:
SHEET 1 OF 1
SECTION A-A
SCALE 1 : 2
2.92cm
7.26cm
4.23cm
1.13cm
2.48cm
4.13cm
2.06cm
1.02cm
2.76cm

II. Power Specifications:

Current and Power Calculations

The SURMA payload will use a 30W Murata UHE-12/2500-D24-C DC/DC Converter that will convert the 30 VDC to 12 VDC for the Arduino and other power components. The UHE converter is able to provide up to 12 VDC at 2.5 A while only drawing a maximum of approximately 1.4 A from the HASP Interface's 30 VDC supply. The expected current draw of the converter is shown in **Table 2**.

Table 2: Power Draw (UHE DC/DC Converter)

	Voltage (V)	Current (mA)	Power (W)
No Load	30	70	2.1
Expected	30	180	5.6
Full Load	30	1390	41.7

The expected load was extrapolated from the no load and full load conditions using the expected current draw of SURMA's electronic equipment. The current load on the UHE converter will be, on average, 324 mA, resulting in a 180 mA draw from the HASP Interface. The power budget for the various electronics is shown in **Table 3**.

Table 3: Power Draw (Components)

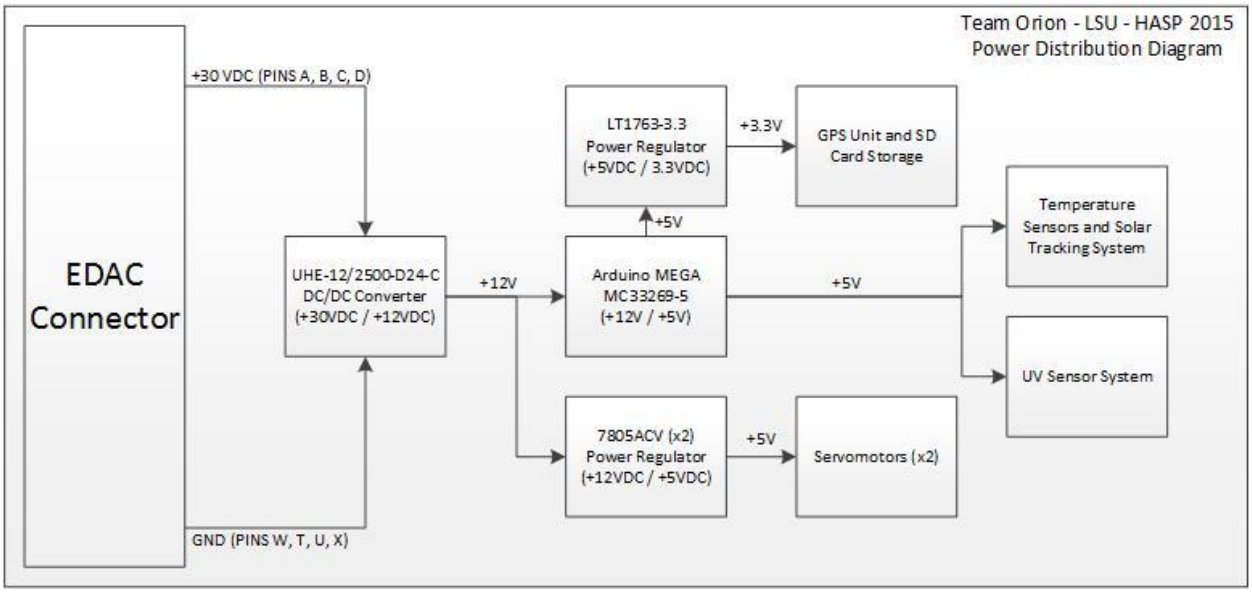
Component	Voltage	Current Draw (mA)	Duty Cycle	Duty Cycle Current (mA)	Power (mW)	Status
Arduino Mega	12	25	1	25	300	Measured
UV Acquisition System	5	10	1	10	50	Measured
Temperature Sensors	5	5	1	5	25	Calculated
GPS and Storage Board	5	20	1	20	100	Estimated
Solar Tracking Electronics	5	10	1	10	50	Estimated
Horizontal Servo (No Load)	5	80	1	80	400	Measured
Horizontal Servo (Full Load)	5	1200	0.02	24	120	Calculated
Vertical Servo (No Load)	5	80	1	80	400	Measured
Vertical Servo (Full Load)	5	1200	0.02	24	120	Calculated
HackHD Camera	4	650	0.06	39	156	Estimated
12VDC/5VDC Regulators	12	7	1	7	84	Calculated
Totals				324	1805	

SURMA uses two servomotors to rotate the electronics bay. If a servomotor stall during the flight, 1.2 A of current will be drawn until the stall ends. Should both servos stall at the same

time, up to 3.1 A may be drawn at a particular moment. However, this case should not happen as SURMA will only operate on servomotor at a time. Various incident events and their respective current draws are shown in **Table 4**.

Table 4: Incident Event Current Draws		
	HackHD off	HackHD on
Both Servomotors Off	77 mA	727 mA
Single Servomotor On	154 mA	807 mA
Both Servomotors On	237 mA	887 mA
Single Servomotor Stalled	1357 mA	2007 mA
Both Servomotors Stalled	2557 mA	3128 mA

Power Distribution



SURMA will use the 8 power lines from the EDAC 516 connector on the HASP Interface to receive power. A single UHE DC/DC Converter will ramp down the 29-31 VDC to 12 VDC to power the Arduino MEGA and the Power regulators for the servomotors. The Arduino Mega has an internal voltage regulator that will provide 5 V to the various sensors in SURMA as well as the GPS and SD Card Board. The EDAC lines used are shown in **Table 5.**

Table 5: EDAC PINS	
PINS A, B, C, D	+30 VDC
PINS W, T, U, X	GND

III. Downlink Telemetry Specifications:

Serial data downlink will be **streamed**. Downlink will be set up as regular DATA transmissions, INFO transmissions (status updates sent down when a code from uplink command is received), and ERROR transmissions (when an error occurs). The serial downlink rate is 256 bytes per DATA transfer and 57 bytes for INFO and ERROR transfers. Large classification payloads on HASP have an allotted data rate of 4800 baud, which allows for 600 bytes of data transfer per second. This would give us maximum 2 data downlink transmissions per second with enough left over for an info or error transmission. **This would mean a serial downlink rate of 569 bytes per second (4552 bits/s).**

Table 6: Serial Data Record: DATA Transmission

ASCII Characters	**Name**	**Description**	**Format**
6 Byte String	Header- Data	Record type indicator for DATA transmission	DATA
10 Byte String	Timestamp	Current RTC Time	HH/MM/SS
9 Byte String	UVA	UVA flight data from ADC	UVA XXX
9 Byte String	UVB	UVB flight data from ADC	UVB XXX
9 Byte String	UVC	UVC flight data from ADC	UVC XXX
10 Byte String	UV-T	Broadband UV flight data from ADC	UV-T XXX
13 Byte String	UV-Temp	Records temperature of UV photodiodes	UV-TEMP XXX
9 Byte String	East Photodiode	Photodiode oriented east of the sun	PD1 XXX
9 Byte String	West Photodiode	Photodiode oriented west of the sun	PD2 XXX
9 Byte String	Servo Temperature Sensor 1	Records temperature of servo controlling azimuthal orientation	ST1 XXX
9 Byte String	Servo Temperature Sensor 2	Records temperature of servo controlling elevation orientation	ST2 XXX
125 Byte String	GPS	Requested data for position	See Format
13 Byte String	Status	Error status- CODE	See Format
12 Byte String	Checksum	Truncated check	X
4 Byte String	Footer	Record Type End indicator	XXXX
Total: 256 Bytes			

As shown in Table 6, the data sends the current timestamp, the recorded sensor data, GPS data, a flight UNIX timestamp, a status update, and a checksum. The sensor data includes all the UV

sensors and control sensors. Control sensors include the temperature sensors and the photodiodes. The GPS has a UNIX timestamp and SURMA will be equipped with an RTC timestamp to correlate the data.

Table 7: Serial Data Record: INFO Transmission			
ASCII Characters	**Name**	**Description**	**Format**
6 Byte String	Header- Info	Record Type indicator for INFO	INFO
10 Byte String	Timestamp	Current RTC Time	HH/MM/SS
12 Byte String	Trace	Function Trace in code	XXXX
13 Byte String	Status	Error Status- CODE	See Format
12 Byte String	Checksum	Truncated check	X
4 Byte String	Footer	Record Type End indicator	XXXX
Total: 57 Bytes			

The INFO transmission works the same as the DATA transmission but must be requested by command and will not be sent automatically. Its main purpose is to send a trace code, which indicates the method or module of the code the computer is currently running, and also takes in a status update.

Table 8: Serial Data Record: ERROR Transmission			
ASCII Characters	**Name**	**Description**	**Format**
6 Byte String	Header- Error	Record Type indicator for Error code	ERRR
10 Byte String	Timestamp	Current RTC Time	HH/MM/SS
12 Byte String	Trace	Function Trace in code	XXXX
13 Byte String	Status	Error Status- Code	XXXX
12 Byte String	Checksum	Truncated check	X
4 Byte String	Footer	Record Type End indicator	XXXX
Total: 57 Bytes			

The ERRR transmission is sent automatically when the Error Code changes. The Error code exists to handle abrupt changes or issues in the code. This may include restarts, servo jamming, inability to take data, or any other issues. Some errors will prevent the code from moving on until they are fixed and manually have the code reset to 0000, these errors may include the restart error, or a temporary servo jam that rectifies itself. The error transmission includes a code trace.

ERROR CODES:
0000- Default, No current errors
1111- During Flight Restart
2222- Servo1 not moving
3333- Servo2 not moving
4444- Azimuthal photodiodes not functioning

The potential error codes listed above are just examples of issues we may encounter. Through further testing more error causes and codes will be developed.

SURMA will be using **nine analog channels** as shown in Table_.
There will only be **two discrete lines being used (Received and Transmitted Data)** and are connected to pins 2 and 3 on the HASP serial interface.
SURMA will **not have any on-board transmitters**

Table 9: Analog Channels	
#	Description
0	UVA
1	UVB
2	UVC
3	Broadband UV
4	UV-Temperature
5	Servo1-Temperature
6	Servo2-Temperature
7	East photodiode
8	West photodiode

Downlink Formatting

The following show how SURMA is expected to downlink information and the format that will be displayed. Values marked with "*" are from the HASP Interface Manuel

DATA Formatting:
DATA, HH:MM:SS, UVa XXX, UVb XXX, UVc XXX, UV-T XXX, UV-TEMP XXX, PD1 XXX, PD2 XXX, ST1 XXX, ST2 XXX, (UNIX Timestamp**)*, $GPGGA*, (UTC Time)*, (Latitude ddmm.mmmm)*, (Latitude Hemisphere, N or S)*, (Longitude Hemisphere, E or W)*, (Position Fix Indicator)*, (Number of Satellites in use 00-12)*, Horizontal Dilution of Precision, .5 to 99.9)*, (MSL Altitude, -9999.9 to 99999.9 meters)*, (Geoidal height, -999.9 to 9999.9 meters*), (Differential GPS (RTCM SC-104) data age, number of seconds since last valid RTCM transmission, nu1l if non-DGPS)*, (Differential Reference Station ID, 0000 to 1023, null if non-DGPS)*, (GPS CHECKSUM)*, (Flight UNIX timestamp), Status XXXX, Checksum X, ENDD,

INFO Formatting:
INFO, HH:MM:SS, (TRACE a binary number 0000-1111), Trace XXXX, Status XXXX, Checksum X, ENDI

ERRR Formatting:
ERR, HH:MM:SS, (TRACE a binary number 0000-1111), Status XXXX, Trace XXXX, Checksum X, ENDE

IV. Uplink Commanding Specifications:

SURMA **shall require uplink command capabilities** and **will not be uplinked at regular intervals**. Uplinks should only occur in response to errors and necessary manual adjustments to the system. That being said, Team Orion plans to make no uplink transmissions unless something happens, however, the design of SURMA has extensive ability to handle and report errors, and respond to uplink commands. **SURMA has no on-board receivers.**

Table 10: Uplink Data Format

Byte:	**Hex Value:**	**Description:**
1	1	Start of Heading (SOH)
2	2	Start of Text (STX)
3	Command Byte 1	First byte of command transmitted from ground
4	Command Byte 2	Second Byte of command transmitted from ground
5	3	End of Text (ETX)
6	D	Carriage Return (CR)
7	A	Line Feed (LF)

The uplink data format used will be the default/suggested format listed in the Interface Manual: The command system shall come in to the on board Arduino on an RX receiving line. This line will be set to a word in the system code. This word shall take in each command byte and run the values into the system. They will be encoded for two sets of control, the first byte will be encoded for system control and error adjusting, and the second byte will be dedicated to control of the vertical rotational system.

Uplink Commands

Byte 1 XXXXXXXX will correspond to the error codes and will be used in case of a needed control of the horizontal system. The value will be between 0 and FF, however, shall also be encoded to certain binary values for control, a command byte of 00000000 will send no value and no response will come from the system for it, while a value of 0000XXXX will send a command to control the horizontal system by the amount held on the second nibble of the byte, this set up allows for command control coded to the nibble. There is an issue here of limiting the rotation of the horizontal system to only values less than a nibble, however this should not be an issue as horizontal corrections should not exceed the F hexadecimal value maximum, this is for positive rotations. The vertical system will use a planned .txt file for control of the rotational system. Table 11 provides a list of the optional codes that are planned to be included.

Table 11: Uplink Commands	
0001XXXX	This is the same as the zero for the first byte system, but will be a negative rotation of the servo
0001XXXX	This is the same as the zero for the first byte system, but will be a negative rotation of the servo
0010XXXX	System pause for a number of seconds equal to the XXXX nibble, not exceeding 15 seconds
0011XXXX	Stops the system until a command to continue code is sent
0100XXXX	Continues the paused code manually, this can be done from an error pausing the code or a manual pause
0101XXXX	Sets to a system restart command
0110XXXX	Requests for a trace downlink from the payload
0111XXXX	Runs to trace function that moves the code to the function traced at XXXX, the code trace stays less than a nibble and will ignore values for invalid traces
1000XXXX	Pings the payload, this will send an error transmission. The error transmission should send an error message of 0000, however, this will show if the system gets an error that it did not send us and confirm connectivity to payload.
1001XXXX	This sets the rotation of the vertical system to pause and ignore the planned rotations only taking in This pauses the rotation of the horizontal system the uplink rotation
1010XXXX	Pauses the rotation of the horizontal system
1011XXXX	Returns control to the Arduino

We retain values from 8-F which are still un-encoded for system commands, these may be used later.

Byte 2 XXXXXXXX will be a value between 0 and FF HEX, since the ground transmitted needs HEX values. A value of 0 will give no change, and the system will always rotate by an amount corresponding to the value on the second byte. This is because the system may run into errors on the flight and may need manual control rotation.

V. Integration and Logistics

Logistics	
Arrival date	Monday, August 3rd
Arrival time	12:00 PM
Time required for integration	4 hours
Participants	
Victor Fernandez-Kim (Team Leader)	vfernandezkim@gmail.com (225) 400-8644
Stephen Harb	stephenaliharb@gmail.com (337) 207-2149
Joshua Collins	josbluehill@gmail.com (985) 287-1093
Allen Davis	adav156@lsu.edu (251) 459-4578
Additional LSU Support	
Miscellaneous (i.e. lifting, moving equipment, hotel information/arrangements, any special delivery needs…)	None at the moment
Equipment	Multimeter, soldering station, oscilloscope, crimping tool, wire spools, UV lamp/detectors

Successful payload integration to HASP would be achieved if:

- SURMA systems are powered and are drawing expected current values
- All downlink and uplink communications are functioning
- SURMA experiences no mounting issues, meets regulations, and is stable
- SURMAs tracking system is functioning as designed
- All systems remain within operating temperature ranges thermal vacuum testing and unexpected errors are resolved

Due to the structure of the payload and its moving components, SURMA will be disassembled and packed to prevent damage to the components or structures. The electronics bay and electronics within it will be removed from the rotational platforms. The legs that stand from the rotational platforms will also be removable to pack separately. Team Orion plans to prepare pre-flight/integration test software to run through the entire system and signal for any errors at integration.

Team Orion will not be able to test the tracking system within the thermal vacuum chamber. Therefore, a separate software must be prepared specifically to ensure that the components and servos are within operating temperatures. The code will be designed to operate normally except the servos will rotate at set intervals (similar to what is expected at altitude) as opposed to being controlled by the tracking system. Lastly, the software will report if issues arise (such as servos stopping).

Integration Steps and Checks for Successful Integration

1. ___ Inspect individual components for damage or irregularities
2. ___ Assemble payload
3. ___ Weigh payload. SURMA makes up _____ kg of the allowable 20 kg
4. ___ Mount payload to HASP gondola and confirm no mechanical instability
5. ___ Confirm that SURMA does not sit or rotate beyond any restricted areas (including height). Maximum constraints: 38 x 30 x 30 cm
6. ___ Connect the EDAC 516 and RS 232 connectors to payload
7. ___ Power on and determine current draws across electronics and confirm with expected values
8. ___ Run pre-flight and flight software. Confirm telemetry communication is functioning
9. ___ Power off and remove SD card. Check saved values and check for irregularities

Thermal Vacuum Chamber Testing

1. ___ Mount SURMA within thermal vacuum chamber
2. ___ Power on and run thermal vacuum software
3. ___ Power down, remove payload, and check for issues