University of Bridgeport
June HASP Status Report
Submitted June 24, 2016
Submitted: Maheshwari Kumar Rakkappan Team Member

Team Activities:

PSIP:

As requested, the team revised the body of the PSIP report and diagrams. A revised PSIP was submitted on Monday, May 30th. The final PSIP has been submitted today, Friday, June 24, 2016.

Build:

Mechanical Progress

1. Chassis nearly complete (Figures 1 and 2).
2. All walls fabricated, foam and heaters added, heaters tested.
3. Top surface (Lexan) fabricated, foam and heater added, heater tested.
4. Equipment Bay Drawer fabricated, heater added and epoxied and tested.
5. Worked out a passive mechanism for drawer retention.
6. Assembled entire chassis with flanges, attached to base-plate-substitute useable for Lab testing.
7. Interior of Test Bay walls and ceiling painted black. Entire chassis exterior painted gloss white.
8. Mechanical Drawings updated to reflect as-built hardware.
9. Servo tray design complete. 3D print either today or early next week.
10. Robot arm designed and fabricated, needs slight modification then ready to bolt in and wire up. (Figure 3)
11. Need to cut mouse hole in Test Bay floor, drill hole in Stern side for HASP wiring, attach chassis to the HASP Mounting Plate, and mount the outside Temperature sensor

Electrical Progress

1. Servo voltage ripple too high when driving servos. Servos require 10-msec current pulses, and the switcher isn't fast enough to respond. Turned switcher PS2 up to 8 volts, and wired in a Low-Dropout Linear Regulator behind it – set for 6 volt output. This remains stable, and will provide better power measurements, at the expense of a little power.
2. Breadboarded circuit for receiving data from multiple digital temp sensors – works well.
3. Fabricated small wiring distribution board for temp sensors.
4. Breadboarded DIGITAL 1, 2, and 3 controls and automatic Equipment Bay Overtemp heater shutoff.
5. Built / Tested Servo Drive board
6. Redesigned Current Sensor amplifiers to be driven from a higher voltage supply, stay within their linear range, and have higher gain.
7. Now ready to wire the chassis.

Software Progress

1. Defined *TESTING MODES* versus *FLIGHT PHASES* for the control of the software, as follows:

   MODE
   - 0 = Running Power-on tests (POT)
   - 1 = POT complete. Check for time_sync
   - 2 = Standby. Time Sync Completed or bypassed
   - 3 = Diagnostic Mode
   - 4 = Flight Mode, Testing active

   PHASE
   - 0 = Ground operations
   - 1 = Wait for Launch (not testing)
   - 2 = Ascending (testing)
   - 3 = At Altitude, Fixed Temperature (testing)
   - 4 = On Temperature profile (testing)
   - 5 = Testing Stopped (hold present temperature).

2. Defined Task timeline as follows:

   TESTING SCHEDULE - seconds after each minute (Mission Time, seconds locked to GPS time)

   00 - 01 Nothing scheduled
   02 - 05 GPS-related Processing
   05 Temp. Measure / Heater Control
   06 - 10 Test Servo1
   11 Temp. Measure / Heater Control
   12 - 16 Test Servo2
   17 Temp. Measure / Heater Control
   18 - 22 Test Servo3
   23 Temp. Measure / Heater Control
   24 - 28 Test Servo4
   29 Temp. Measure / Heater Control
   30 - 34 Test Servo5
   35 Temp. Measure / Heater Control
   36 - 40 Test Servo6
   41 Temp. Measure / Heater Control
   42 - 46 Test Servo7
   47 Temp. Measure / Heater Control
   48 - 52 Prepare and send Test Message
   53 Temp. Measure / Heater Control
   54 - 58 Log data to SD card
   59 Temp. Measure / Heater Control

3. Partitioned the Software as follows:
    a. Executive - manage time, test schedule, modes, and diagnostics. Call other routines based on time flags from Time_update. Receive commands from HASP
    b. Time Update – Keep Mission and Flight Times. Sync Mission Time to either OWN_GPS or HASP_GPS
    c. OWN_GPS – receive and parse data every second. Two types of sentences are parsed: $GPGGA and $GPRMS. Calculate course/speed once per minute. Compare our own GPS data to the HASP, and trigger a failure message if GPS fails.
    d. HASP_PROCESS - receive and parse $GPGGA data every minute. Also Receive and parse Commands, then pass them to the Executive for Processing.
    e. Thermal Management – every 6 seconds, read the wall temperatures and activate or deactivate the Heaters, keeping the wall at the Target temperature set by the Executive. IF all heaters need to be ON and we are in the servo testing part of the timeline, pick one heater to turn off, at random.
    f. Servo Drive – manage the driving of all the servos, one at a time, and set flags for Servo Measurement
    g. Servo Measurement – when a servo is started, watch its position 100 times per second and read the current draw, calculate power and energy until the servo stops and holds for a short interval
    h. Prep and Send – Gather data processed by all other routines and prepare outgoing messages to HASP. Frame them properly. Send Alarms or Status messages as directed by Executive. Also copy both GPS data messages to the HASP, so they will be recorded in our data stream.
    i. Logfile – Manage the writing of all test data, alarm messages, etc. to the onboard SD card. Manage files on this card when directed by command. (Open new logfile, close, clear/erase files, etc.)
    j. Miscellaneous Processing – Outside Air temperature and Pressure, monitor the Arduino in the Test Bay, and other miscellaneous functions.

4. The Executive is 35% coded. Time management has been set up but not fully tested. MODE and PHASE processing needs work
5. Time Update is 80% coded and unit tested.
6. OWN_GPS is 90% coded and unit tested.
7. HASP_Processing is 98% coded and tested. HASP GPS messages can be received and parsed, and can be used to synchronize time. Command messages are received and parsed and checked for validity. Now, they just print. But the will be passed to the Executive soon.
8. Thermal Management is not yet coded, but we *have* written code to get the data from the six sensors. This was tested with breadboarded hardware. Once we set up the framework, we can drop this code in. Would like to do this next week and wire the heater controls, so we can begin temperature testing.
9. Servo Drive – 60% coded and unit tested.

10. Servo Measurement – We have made several tests in this area to validate concepts, and we can read the Potentiometers, but we don't have much coded.
11. Prep and Send – not coded.
12. Logfile – 50% coded and unit tested.
13. Miscellaneous – not coded.

Integration and Test

The Executive, Time_update, OWN_GPS and HASP Processing are all working together in one program. The other processes have not been integrated.

We have programmed an Arduino Mega to emulate the HASP system for us. It receives its own GPS signals (3 messages per second) and sends one $GPGGA message to our Project every minute. When you type 2 letters and RETURN on a PC keyboard, this Arduino properly frames an uplink message and sends it to the Project computer. This piece of STE "Special Test Equipment" will be used for system test as our box gets integrated, and will hopefully simplify the HASP integration process.

Vacuum Test: Individual components have been placed in 1-hour long vacuum tests.

Thermal Control Model: A new team member, Abd Elfatah Karkory, is developing a 1-D analytic heat transfer model for thermal control for the testbed. Faculty have agreed that this model will be enhanced by other grad students under future masters projects. Subsequently a CFD model will be created and after the HASP flight, temperature data will be available for a "proof of concept" validation.

HASP Group Meetings: Maheshwari Kumar Rakkappan, Dr. Neal Lewis, Advisor Mr. Larry Reed, Xuang (Sam) Zhang and Dr. Jani Pallis participated on the June 3rd conference call meeting.

Issues:

No issues. A question was submitted regarding the possibility of a professional photographer from the University of Bridgeport being allowed to photograph the launch activities in New Mexico. Dr. Guzik is addressing the question with NASA.

Milestones Achieved:

Final PSIP submitted.

Current Team Leaders/Members, Demographics:

Student Project Manager: Bashar Alhafni (Undergraduate Student – Computer Science)
Leader Flight Computer, Data: Bashar Alhafni (Undergraduate Student – Computer Science)
Arduino Gesture Programming: Wayne Teto (Undergraduate – Electrical Engineering)
Structure Lead Arjun Kumar (Graduate Student – Mechanical Engineering)

While Arjun just graduated this semester, he will be in the Bridgeport area for another month and is continuing to assist. Rochen Krishna Thashanath Sajeevan (ME graduate student) will be taking over for Arjun. Karan Kakanur Patel (ME graduate student) has also joined the team and will be responsible for 3D printing.
Thermal Control: Maheshwari Kumar Rakkappan - (Graduate Student – Mechanical Engineering). Graduate Mechanical Engineering student Abd Elfatah has joined the team in this area.
Robot Gesture Range of Motion and Fabrication: Phillip Carroll (Undergraduate Student – Industrial Design); Team Member: Josh Hauge
Power and Communications: Xuan (Sam) Zhang (Graduate Student – Computer Science/Electrical Engineering)
Lead Faculty Advisor: Dr. Jani Macari Pallis (Mechanical and Aerospace Engineering)
Faculty Advisor Dr. Neal Lewis (Technology Management - Project Management)
Faculty Advisor Dr. Sarosh Patel (Computer Science and Engineering)
Education Partner: David Mestre (Discovery Museum and Planetarium: Director of Space Sciences, responsible for Challenger Center Mission Control)
Education Partner: Lawrence Reed: (Discovery Museum and Planetarium: Electrical Engineering, Communications).
Engineering Volunteer: James J. Pallis (United Airlines – Mechanical Engineering Design)

Figure 1 The UB Testbed container.

Figure 2: The UB Testbed container opened and with the Electronics Bay drawer puller out. Nichrome wire patterns are visible on the walls.

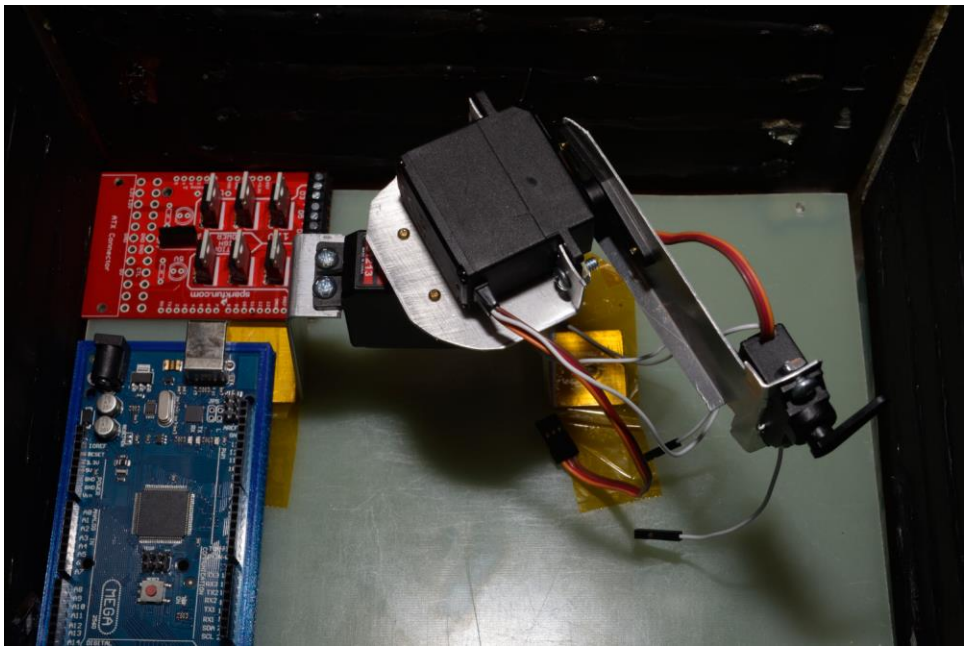

Figure 3 – The “mini” 3-servomotor robotic arm in the container (just taped down at this point).