HASP Student Payload Application for 2018

Payload Title: Robotic Arm Manipulation and Materials Matching: RA(M^3) or RAM

Institution:Durham Tech, North Carolina Central University, and North Carolina State University

Payload Class (LARGE):

Submit Date: 12/15/2017

Project Abstract: The need for satellite maintenance is growing as established satellites run out of fuel and degrade in the harsh environments of space. NASA's Satellite Servicing Projects Division (SSPD) is developing unmanned craft to provide a lower-cost, lower-risk method of extending legacy satellites' lifetimes. While only two active satellites are designed for maintenance, the next generation, including the James Webb Space Telescope, plan to leverage robotic refueling and maintenance. Our experiment for HASP tests the limits of relatively low-powered, semi-autonomous robotic dexterity, running repeated iterations of tests that require precision actuation, computer vision, and force-torque sensors. A robotic arm will perform simple tasks such as toggling switches, twisting objects, and opening/closing velcro flaps as one might while refueling a satellite on-orbit. Our experiment evaluates the impacts of extended use in extreme conditions. Orbital missions must contend with traditional lubricants evaporating in low-pressure environments, survive direct and uninterrupted sunlight, and control heating/cooling in the absence of convecting atmosphere. Over the duration of the flight we will measure any degradation in performance, response time, and accuracy. We will also test the efficacy of computer-vision, both for autonomy as well as for closed-loop feedback during operation.

Team Name: The Unacceptable Risks

Team or Project Website: http://www.theunacceptablerisks.com

	Student Leader Contact Information:	Faculty Advisor Contact Information:
Name:	[redacted]	[redacted]
Department:	[redacted]	[redacted]
Mailing Address:	[redacted]	[redacted]
City, State, Zip:	[redacted]	[redacted]
e-mail:	[redacted]	[redacted]
Office Telephone:	[redacted]	[redacted]
Mobile Telephone:	[redacted]	[redacted]

Contents

Payload Description

Overview

There are over 1,000 active satellites in orbit around the Earth; that number grows yearly with no signs of slowing. Maintenance of these satellites is a long-term imperative, though only two satellites (Hubble and ISS) are currently designed to be maintainable. NASA is developing articulated robotic arms that can be sent to extend the lifespans of orbiting satellites. A prototype arm was unveiled at Goddard Space Flight Center in 2015. Meanwhile, a privately-owned British company announced in June of this year that they are developing a “space drone” with a non-invasive docking apparatus that will allow it to dock with a variety of satellites to refuel them, provide maintenance, or even adjust the satellite’s position. Our project, Robotic Arm Manipulation and Materials Matching: RA(M^3) or RAM, will contribute to the development of low-cost robotic actuation in a space environment, most immediately satellite servicing. RAM will also provide the opportunity to research the utility of robotic actuation driven by computer vision in a near-space environment.

Methods

To simulate the a satellite servicing mission, we designed a ‘Busy Box’ — much like a child’s toy — with a number of different tasks to perform, each requiring a unique set of kinematic instructions. We have chosen simple mechanical actions like twisting, pushing, pulling, and dragging in order to simulate the types of movement needed for simple satellite servicing missions. RAM will be a large payload to be mounted on the HASP gondola with the frame enclosed by Lexan panels, and will run a set of predetermined trials on various components of the Busy Box. We will measure completion time with the hopes of answering the following questions:

1. Are there particular techniques, patterns, or orders of operations that are quicker/more efficient than others?
2. Is there decay in performance of a low-cost robotic arm over a 12+ hour operation run under full solar exposure in near-vacuum conditions?
3. Is computer vision a viable component for counteracting any systematic drift due to system fatigue?

Science Goals

This experiment will provide a strong analysis of measured timing results for multiple patterns that will be repeated at intervals during flight; this involves measuring:

- total current draw
- time to complete individual and aggregated tasks
- percent-of-drift through repeated arm motions

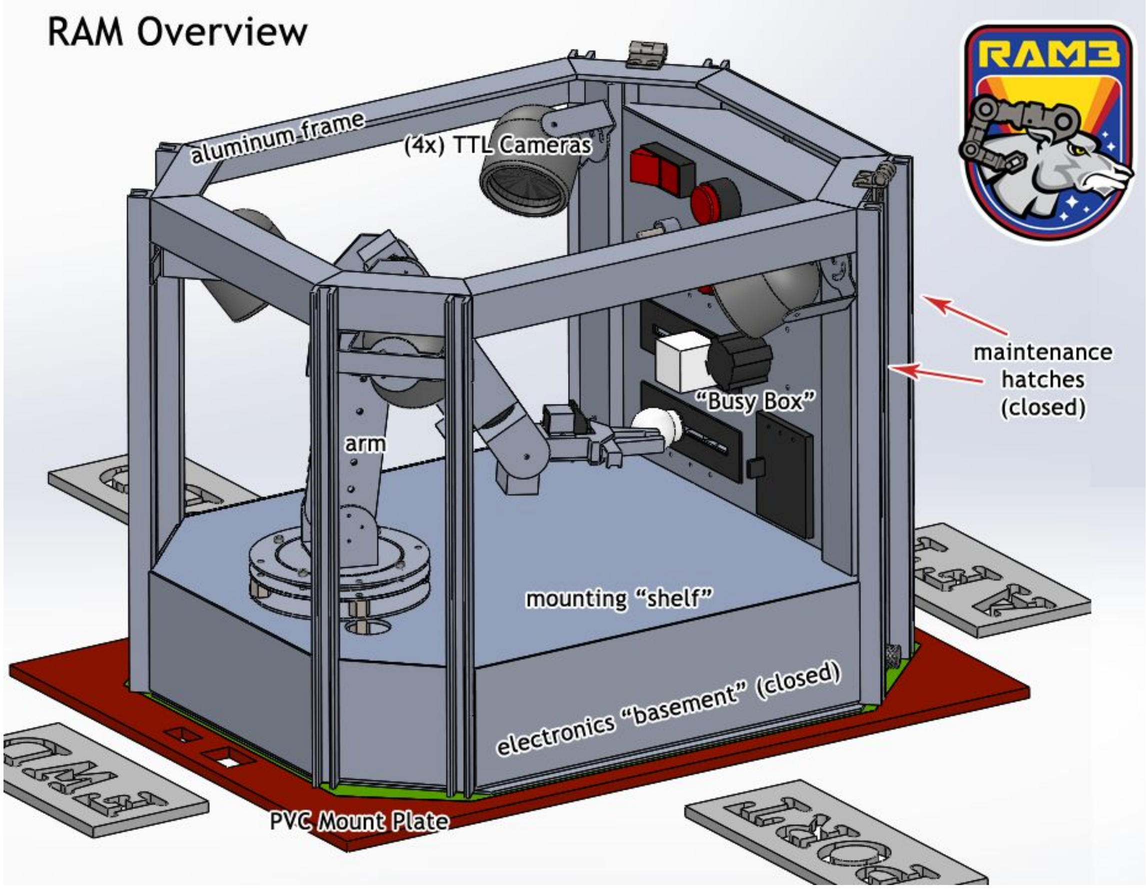
As a stretch goal, we may install Digital Multimeter chips on the supply lines for each servo to measure changes in individual current draw over the course of the entire flight. We may also install twin accelerometers — one fixed to the HASP baseplate as a calibrator and one on the arm — to interpret the effects of any external vibrations, though we do not expect significant shocks during float.

Concept of Operations

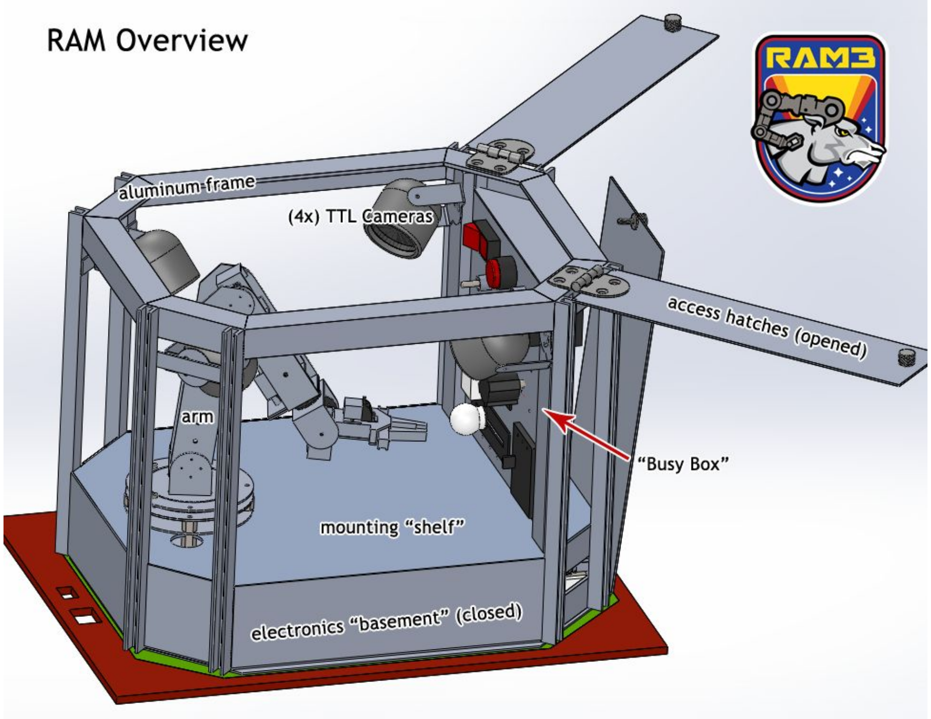
The ground team will send serial commands to RAM to initiate in-flight trials. While in transport and during ascent, the robotic arm of RAM will have a default storage position to ensure maximum durability and safety of the components for non-operating periods. The 'wrist' of the arm will rest in its 'cradle.' The geometry of the cradle will be finalized after testing, but should feature a self-centering feature and a switch to confirm proper docking during balloon ascent and descent.

- When the HASP gondola reaches its estimated flight altitude, the team will signal over the serial, from ground to HASP to RAM, to begin operations.
- At this time, RAM will begin a warm-up process that will consist of mostly self-diagnostics.
    - When it reports back that all systems are nominal, we will use the serial link to instruct RAM to begin a set of trials, sending recorded data to the ground.
- RAM will take all measurements automatically and will record video with Arduino-compatible TTL Serial cameras as long as the payload is powered on. This video will be used as a secondary method of corroborating trial data measured by the RAM system.
- In parallel with robotic operation, the payload will host a suite of sensors taking ambient temperature, pressure, and humidity readings as well as measuring temperatures of each servo. These measurements and trial data will be downlinked during flight and stored locally on RAM's local storage as well. The information recorded by our atmospheric sensors will further contextualize and inform our analysis of the arm's performance, particularly in the event of an issue.

The hull of RAM will have four cameras recording video, as well as a 5th arm-mounted camera. In addition to being used for post-processing, these cameras will be able to take pictures that can be downlinked during flight. The ability to receive live photographic feedback will help to inform the ground team in the event that any errors occur and provide immediate feedback. Having periodic imagery feedback gives a level of confidence to the ground team responsible for selecting which trials the arm performs. Each set of trials will take from 30 to 60+ minutes.



Please note that in all renders we have omitted the cradle and the arm-mounted PixyCam; we need more testing to determine their orientation, balance, and placement.



Kinematics & Electrical Design

RAM is designed to operate with almost no guidance from the ground team beyond "which pre-set trial am I running?". It is a robotic arm with six degrees of freedom (6DOF). A Raspberry Pi B+ (RPi) combined with a RoboPi expansion board that will control the arm directly. These components will drive the servos of the arm. The Raspberry Pi will make decisions based on input from the PixyCam, a robotic vision camera and microcontroller platform. In order to properly guide the arm based on camera input, we are implementing Robot Operating System (ROS) MoveIt!, the most popular open-source software for robotic actuation. MoveIt! can be used for both Forward and Inverse Kinematics, motion planning, control, and navigation. To use MoveIt! effectively, we will configure the software using Denavit-Hartenberg parameters specific to our robotic arm.

The primary goal of the current electronic component design is to alleviate single points of failure. It is entirely feasible that a lone RPi with the appropriate array of expansion boards could manage the entire system. However, the danger here is if that one RPi fails then the whole system could follow. In the current design, it is possible that the RoboPi expansion board can be controlled over the I^2C network (though our original setup doesn't reflect this, it is possible). In this sense, our design is a practice in high-level network design with the intention of cutting down on single points of failure.

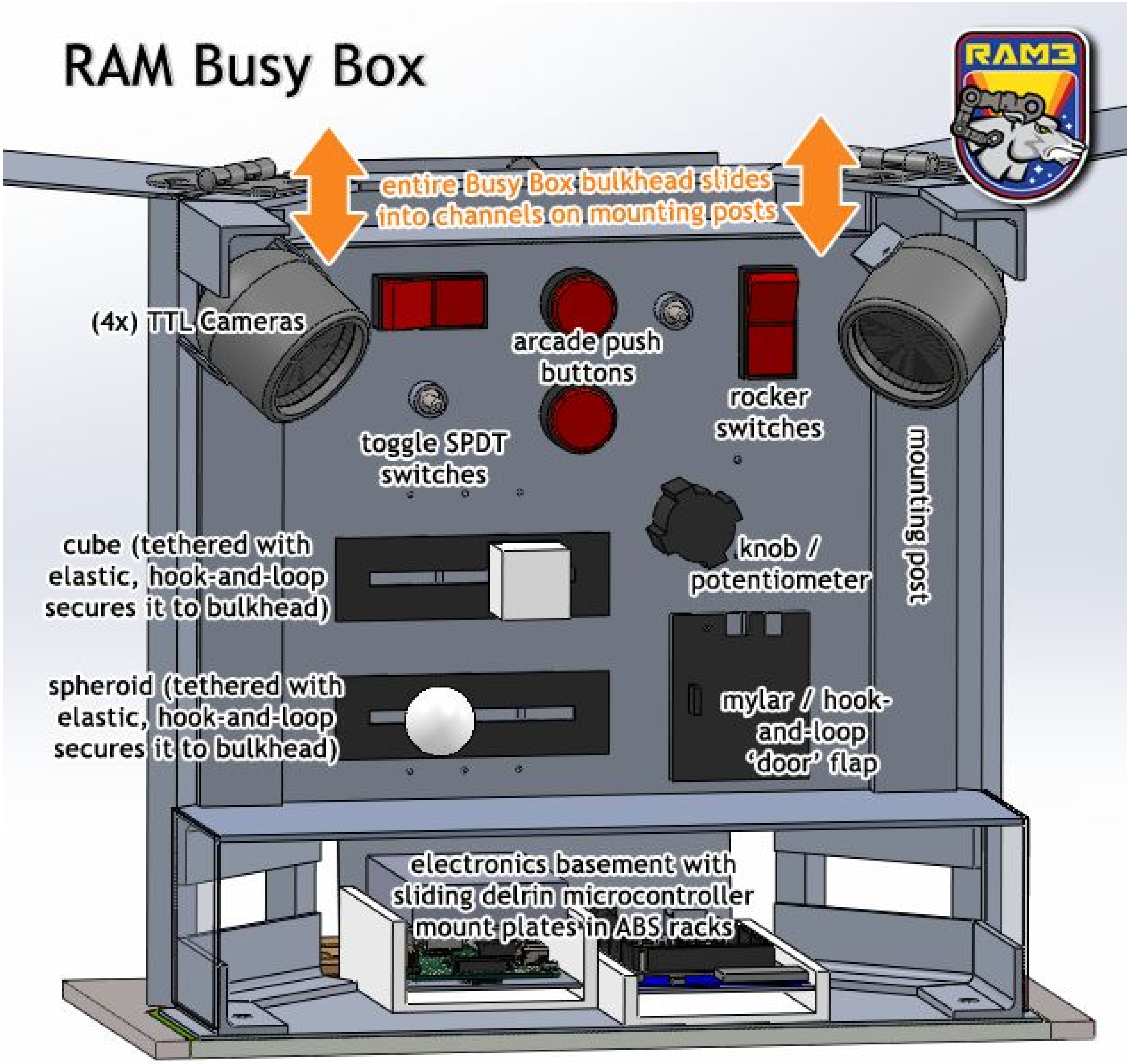
The secondary goal of the electronic design is to improve overall operational efficiency. By dedicating single Arduinos to simple and basic tasks, we free up each component to focus wholly on each objective. This also helps to simplify code implementation, which in turn increases robustness. A single program running on an Arduino needs to only focus on its task and not juggle other jobs. The code to manage multiple operations may create points of failure and slow down development time; the more complicated code is, the more it needs to be quality assured. By using multiple micro controllers, we are able to write more concise and efficient code keeps our system as robust as possible.

Operation

Each trial will consist of a series of actions to be performed by the arm. We can set the arm into one of at least three different modes for each trial. These modes will govern the starting position of the robotic arm as it performs each task in a trial. Each element in the Busy Box is attached to an actuator, which provides indication of success; the Arduino controlling the Busy Box will record button presses, switch toggles, etc. This feedback will dictate if the arm has completed the task and if it may proceed to the next action. The modes come into play between each action in a trial:

- Mode 1 will require the arm to return to its self-diagnostic position ("home") before starting its next action.
- Mode 2 will require that the arm move to a position halfway between "home" and the end position of its most recently completed action.
- Mode 3 will allow for the arm to move backward until it has acquired a lock on the actuator of interest for the next action in its current trial.

These modes allow for a few interesting scenarios. First, the starting position is a fixed (X,Y,Z) coordinate position: fixed values that will be sent to each servo. Since this position is not dependent on computer vision and will be a static value, it allows us to track any drift that develops over time. Modes 1 and 2 allow for two different ways to collect data on drifting. Mode 3 allows the system to operate without caring about drift, as it only uses feedback from the camera ("closed loop") and not from any static servo positions.

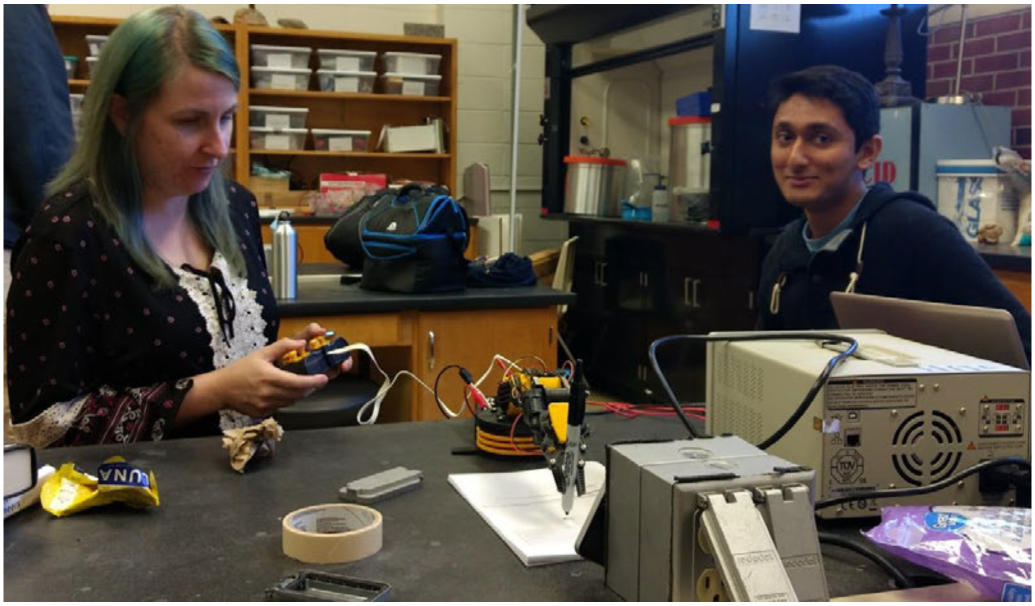

RAM Busy Box
RAM3
entire Busy Box bulkhead slides into channels on mounting posts
(4x) TTL Cameras
arcade push buttons
rocker switches
toggle SPDT switches
mounting post
cube (tethered with elastic, hook-and-loop secures it to bulkhead)
knob / potentiometer
spheroid (tethered with elastic, hook-and-loop secures it to bulkhead)
mylar / hook-and-loop 'door' flap
electronics basement with sliding delrin microcontroller mount plates in ABS racks

The Busy Box is a panel of toggles, buttons, and switches that will test the dexterity and functionality of our robotic arm. Electronic components like the switches and push-buttons will communicate directly to the Arduino. Purely-mechanical components like the cube and spheroid and the hook-and-loop flap will have limited feedback to the Arduino via magnetic proximity switches, but are primarily monitored via camera. Four cameras mounted around the frame provide a live view of the arm's operation. We expect our arm to perform the following tasks:

- switches: *toggle on/off*
- buttons: *press*
- knob: *turn in either direction to specific #s of degrees*
- cube: *grip, unstick from hook-and-loop, move, re-stick*
- spheroid: *grip, unstick from hook-and-loop, move, re-stick*

- door flap: *open and close velcro flap*

Data from these trials will provide interesting perspectives on operational efficiency. We may also see unexpected results; intuitively, it is assumed that the mode which stays as close as possible to the Busy Box (i.e. that doesn't return home after each task) would be the fastest. However, what if that is not the case? Comparing methods in this way allows for the discovery of interesting and compelling outcomes that could inform future experiments.



Success Criteria

MINIMUM SUCCESS CRITERIA [FLIGHT] (D-minus, "At least we tried.")

- ❏ payload powers on
- ❏ arm moves (at all) without blowing fuse
- ❏ arm completes one test 'blind', i.e. without computer vision
- ❏ system writes logs to SD card
- ❏ at least one camera records images and video

"PARTICIPATION AWARD" SUCCESS CRITERIA [FLIGHT] (C-minus, "We're adequate!")

- ❏ payload sends & receives telemetry
- ❏ arm de-cradles from its storage mode at float
- ❏ arm successfully completes at least one 'minimum' trial, autonomously using computer vision:
    - ❏ one button press recorded by Busy Box
    - ❏ one toggle switch in each direction recorded by Busy Box
    - ❏ one rocker switch toggle recorded
    - ❏ 45° or more of rotation of potentiometer/rotary encoder

    - ❏ cube removed from velcro
    - ❏ sphere removed from velcro
    - ❏ velcro opened
- ❏ payload records to SD:
    - ❏ servo position data
    - ❏ system logs

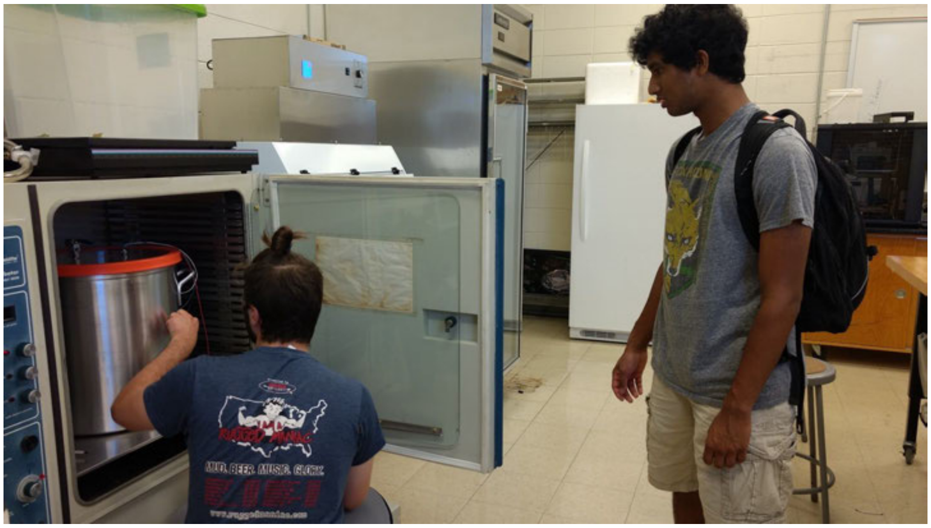
MODERATE SUCCESS CRITERIA (“B”, a good flight)

- ❏ payload sends & receives telemetry
- ❏ arm de-cradles from its storage mode at float
- ❏ arm re-cradles into storage mode before cutdown
- ❏ arm successfully completes *n* ‘standard’ trials autonomously, using computer vision (exact criteria and number still to be determined, but for example:
    - ❏ all buttons pressed within a certain time limit
    - ❏ all switches manipulated within a certain time limit
    - ❏ velcro opened and shut
    - ❏ potentiometer/rotary-encoder fully rotated 320° in both directions
    - ❏ cube removed and replaced three times
    - ❏ sphere removed and replaced three times)
- ❏ arm attempts *n* ‘difficult’ trials autonomously, using computer vision (exact criteria and number still TBD)
- ❏ payload records to SD:
    - ❏ servo position data
    - ❏ atmospheric data
    - ❏ system logs
    - ❏ video of flight
- ❏ payload sends usable still imagery to ground from arm and from 4 ‘overwatch’ cameras, periodically and on-demand

EXCELSIOR! SUCCESS CRITERIA (“A+”, an excellent flight, everyone gets ice-cream sundaes)

- ❏ payload sends & receives telemetry
- ❏ arm de-cradles from its storage mode at float
- ❏ arm re-cradles into storage mode before cutdown
- ❏ arm successfully completes *n* ‘standard’ trials autonomously, using computer vision, without missing a target, losing grip, etc., within a strict limit limit
- ❏ arm successfully completes *n* ‘difficult’ trials autonomously, using computer vision, without missing a target, losing grip, etc.
- ❏ payload records to SD:
    - ❏ servo position data

    - ❑ atmospheric data
    - ❑ system logs
    - ❑ video of flight
- ❑ payload sends usable still imagery to ground from arm and from 4 ‘overwatch’ cameras, periodically and on-demand
- ❑ payload transmits servo position to digital ground simulator
- ❑ payload transmits servo position to physical ground mimic arm (which echoes the motions of the on-float arm)
- ❑ Dr. Guzik makes a grunt of approval
- ❑ Extra credit: robot arm waves to HASP camera



Thermal Control Plan

We know from previous experience that the payload will have to withstand a wide range of temperatures. The challenge is to keep the payload warm during ascent while preventing our electronics from overheating in the lack of air and constant solar exposure at float:

- Mechanically, we have chosen materials proven to be resilient by past balloon projects.
- Electronically, the Raspberry Pi has a distinguished record of service in sub-orbital operation, having flown on multiple short- and long-duration flights by NC Near Space, University of Bridgeport, and CSBF. The family of Arduinos we are using has a similar record. Our servos have been flown by Bridgeport.
- We will use four layers of mylar and tulle to insulate the electronics bay.
- We will heat sink our heat-emitting electronics (i.e. voltage regulators / DC-to-DC buck converters) to the hull with blocks of aluminum, Kapton tape, and thermal paste. If our other microcontrollers prove, during testing, to be major heat accumulators, we will similarly heatsink them.

- We paint our electronics hull in white appliance enamel to protect our payload from solar IR load, reducing our absorptance.

Durham Tech has upgraded to a larger vacuum chamber with a more substantial pump. We have access to an incubator and an extreme temperature freezer that we can use for testing. We do not have a single unit like the Bemco that we can use for combined thermal testing but we have managed to rig the incubator so that we can sit the chamber inside of it. This works for endurance testing but it's not ideal. Like last year, we plan to visit industry partner Paul Mirel and borrow access to Goddard Space Flight Center's temperature-controlled vacuum chamber to simulate flight conditions roughly matching the following anticipated values for what we call "mini-Integration." This year, we have been tentatively invited to use the T/V facilities at NASA's Langley Research Center.

	Temp (˚C)	Pressure (Pa)	Expected duration
Early Sept. AM launch - Ft. Sumner, NM	15 to 25˚C	101,600	indefinite
Crossing tropopause	-55˚C	10,000	20 minutes
Float	-30˚C	500 to 2,000	10 to 16 hours
Crossing tropopause	-55˚C	10,000	20 minutes
Impact	20 to 30˚C	100,000	instantaneous, we hope
Awaiting recovery	0˚ to 40˚C	100,000	4 hours to 3 days

Team Management and Structure

The Team: Narrative Description

The team is sponsored by Julie Hoover, an instructor of geology and the Coordinator of Engineering. She is also the mentor of the Durham Technical Community College Science and Engineering Club, UNM NASA Swarmathon, and the NC Space Grant High Altitude Ballooning

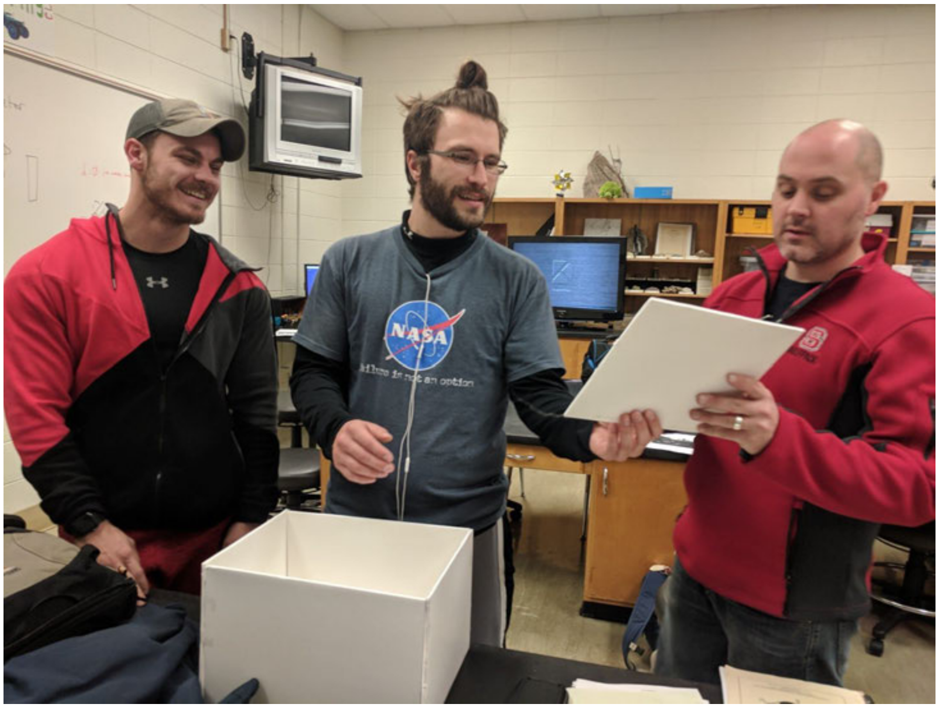
Competition. Ms. Hoover organizes travel, oversees the budget, orders supplies, corrals rowdy students, sets alarms, forces us to research and test, and keeps the ship from sinking. She has been the PI on ten NASA grants at Durham Tech.

The majority of the students on the team participated in HASP 2017 and have been working together on projects continuously since 2015. The team has a great rapport, knows each other's strengths and weaknesses, and genuinely loves doing these projects together. We were the rowdy group high fiving our way through our panic at Integration and hugging like lunatics at the launch. We're sure you remember how weird we are.

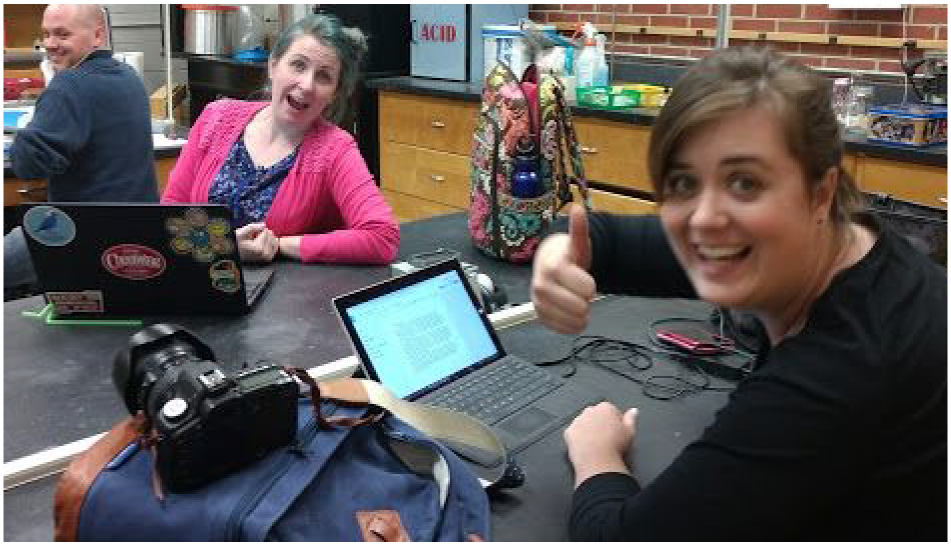
Jimmy Acevedo is returning as the student lead. He is a senior physics major at North Carolina Central University. Jimmy interned at Goddard Space Flight Center during the summers of 2016 and 2017 on the Primordial Inflation Polarization ExploreR (PIPER) balloon project. He has more scientific ballooning experience than most undergraduate students. He is also a NASA Community College Aerospace Scholar, a NC Space Grant STEM Community College Scholar, a NASA Space Public Outreach Team ambassador, and a Technician-class HAM radio operator.

We have three students from North Carolina State University. Dan Daugherty is a mechanical engineering senior who will be managing our mechanical engineering tasks for the second year. Munir Sultan and Kieran Valakuzhy were donated to us by IEEE and will return to support the electrical and software team.

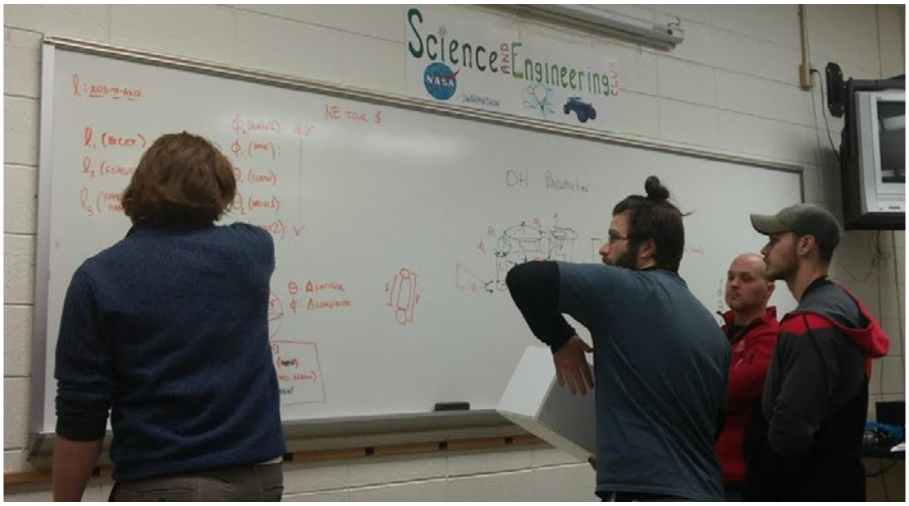
Javian Biswas is joining us from the American University in Washington, D.C. She has worked on public relations and fundraising for HASP 2017. This year she will continue representing our team in the aerospace lobbying circles of Washington.



The rest of the tUR team is from Durham Tech. Dan Koris is a sophomore at Durham Tech who will be the software/electrical team lead again this year. He pitched RAM to us during the HASP 2017 launch; the project draws on experience he gained at his NASA Swarmathon Research Fellowship this summer, where he researched state machines for multi-objective robots operating in dynamic environments with Dr. Jason Isaacs at University of California at Channel Islands. Spencer Boyd is formally joining the team, after consulting in 2017, as our draftsman, machinist, and treasurer of tUR. Noah Olson interned with HASP GOAT in the summer of 2017 during our grueling testing phase and missed most of the fun of the design and build stage. This year he will be supporting the team by helping with documentation, assisting with wiring, and assembling the arm. Meredith Murray and Laura Hagman are participating in their first year of HASP. Meredith is managing our fundraising, photography, and social media. Laura is attempting to keep our prose crisp, our documentation concise, and our technical explanations coherent.



We meet at least once a week for full team meetings at the Durham Scientific Ballooning Facility (the geology lab) where we have a tidy workshop of tools and materials. We have an email mailing list, group chat, and a massive Google Drive that we use to collaborate on work and stay organized. Team leads work together every day and hold teleconferences during busy periods.



We anticipate sending Ms. Hoover, Jimmy Acevedo, Dan Koris, Dan Daugherty and two students to Integration. We hope to send Ms. Hoover, Jimmy Acevedo, Dan Koris, and at least two students for the flight in Fort Sumner, NM.

Financial Support

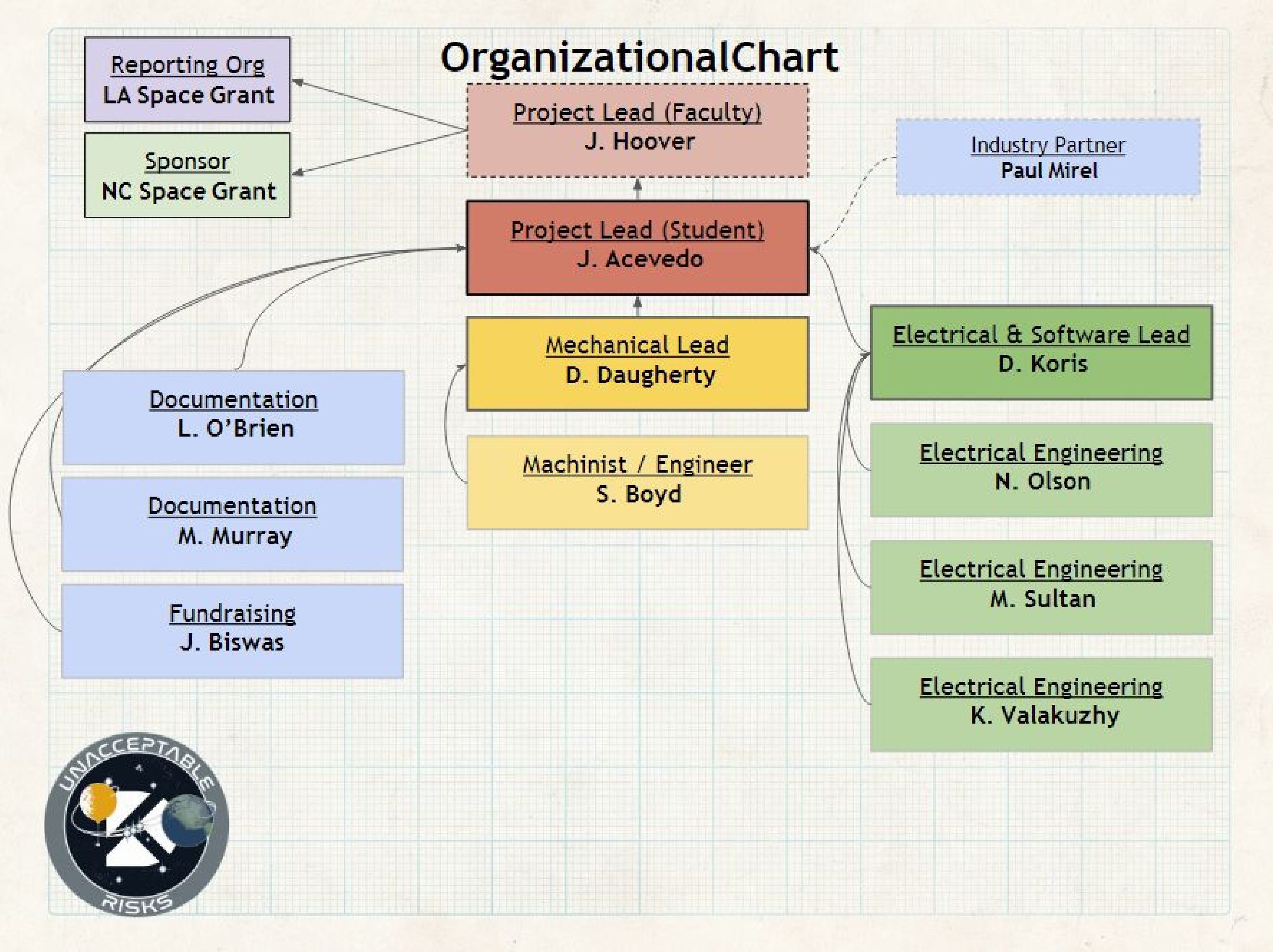
NC Space Grant has awarded tUR $5000 as part of the Team Initiative Grant. This grant is designed to help student teams succeed at opportunities like HASP that are unfunded. The Durham Tech Foundation has also pledged $5000. Our workshop, affectionately named the "Durham Scientific Ballooning Facility" has been outfitted through the generosity of the National Science Foundation's

(NSF) Centers of Research Excellence in Science and Technology (CREST) and various NC Space Grant awards. The team maintains a GoFundMe page and this year we have the added incentive of Dan Koris' much-lauded man bun on the chopping block if we raise $2000.

[redacted]

[redacted]

Organizational Chart



Timeline and Milestones

September 2017

27 All-Hands Meeting

October 2017

04 All-Hands Meeting
10 Engineering Team Meeting
11 All-Hands Meeting
18 All-Hands Meeting
25 Science Experiment Meeting
26 Leadership Meeting

November 2017

01 All-Hands Meeting
03 Leadership Google Hangout
03 Engineering Team Meeting
08 All-Hands Meeting
10 Q & A Teleconference
15 All-Hands Meeting
16 Engineering Meeting
17 Leadership/Engineering Meeting
19 All-Hands Meeting
27 Research Team Google Hangout
29 All-Hands Meeting/ Write-a-thon
30 Engineering Team Meeting

December 2017

01 **Final Draft Due to Jimmy Acevedo**
05 Engineering Team Meeting
06 All-Hands Meeting/ Order Supplies
07 Programming and Wiring
08 Leadership Meeting: Application Review / Order Supplies
13 Turn in Application
15 **Application Due**
14-15 **Tour of Langley, Goddard, & UMD**
18-31 Preliminary Build Days**

January 2018

01-07 Preliminary Build Days**
08 Classes Begin
~15 **Announce student payload selection**
17 All-Hands Meeting/Build Day
18 Programming and Wiring
22 All-Hands Meeting/Build Day
23 Engineering Meeting
24 All-Hands Meeting/Build Day
26 Leadership Meeting
31 All-Hands Meeting/Build Day
January TBD Monthly status reports and teleconferences

February 2018

07 All-Hands Meeting/Build Day
08 Software Team Call
12 Application Revisions Due
14 All-Hands Meeting/Build Day
15 Programming and Wiring
16 Revision Conclusion Event
21 All-Hands Meeting/Build Day
22 Engineering Team Meeting
23 Leadership Meeting
28 All-Hands Meeting/Build Day
February TBD Monthly status reports and teleconferences

March 2018

01 Cryo and Vacuum Testing begins
06 Engineering Meeting
07 All-Hands Meeting - Discuss PSIP
13 Programming and Wiring
14 All-Hands Meeting
22 Engineering Meeting
28 All-Hands Meeting
30 Leadership Meeting
March TBD Monthly status reports and teleconferences

April 2018

03 Engineering Meeting
04 All-Hands Meeting
11 All-Hands Meeting
13 PSIP Draft 1 due
18 All-Hands Meeting
18 **PSIP Final Draft due**
24 Engineering Meeting
25 All-Hands Meeting
27 Leadership Meeting
27 Apr 2018 Preliminary PSIP document due
April TBD Monthly status reports and teleconferences

May 2018
01 Engineering Meeting
02 All-Hands Meeting
09 All-Hands Meeting
16 All-Hands Meeting
23 All-Hands Meeting
29 Engineering Meeting
30 All-Hands Meeting
31 Leadership Meeting
May TBD Monthly status reports and teleconferences

June 2018
26 FLOP live for contribution
29 **Final PSIP document due**
June-July TBD Testing at Goddard Space Flight Center
June TBD Monthly status reports and teleconferences

July 2018
05 FLOP Draft 1 due
18 FLOP Final draft due
23 Payload Integration at CSBF
26 Final FLOP document due
27 End Payload Integration at CSBF
July TBD Monthly status reports and teleconferences

August 2018
13 Classes in session
August TBD Monthly status reports and teleconferences

September 2018
01 Sep - 05 Sep 2018 Flight preparation
06 Target flight ready
07 Target launch date and flight operations
08-11 Recovery, packing and return shipping
14 tUR GOAT PLAR
17 Science Team begins lab work
September TBD Monthly status reports and teleconferences

October 2018
30 Data Due
October TBD Monthly status reports and teleconferences

November 2018
01 Final report live for contributions
TBA SNCURCS Presentation
27 Draft Due
November TBD Monthly status reports and teleconferences

December 2018
07 Final Flight / Science Report due

**tUR team will be building our payload regardless of application acceptance, so we will begin preliminary builds in late December.

Payload Interface Specifications

Describe what HASP resources you will use and how your payload will fit within the constraints.

1. *Weight budget with uncertainties*

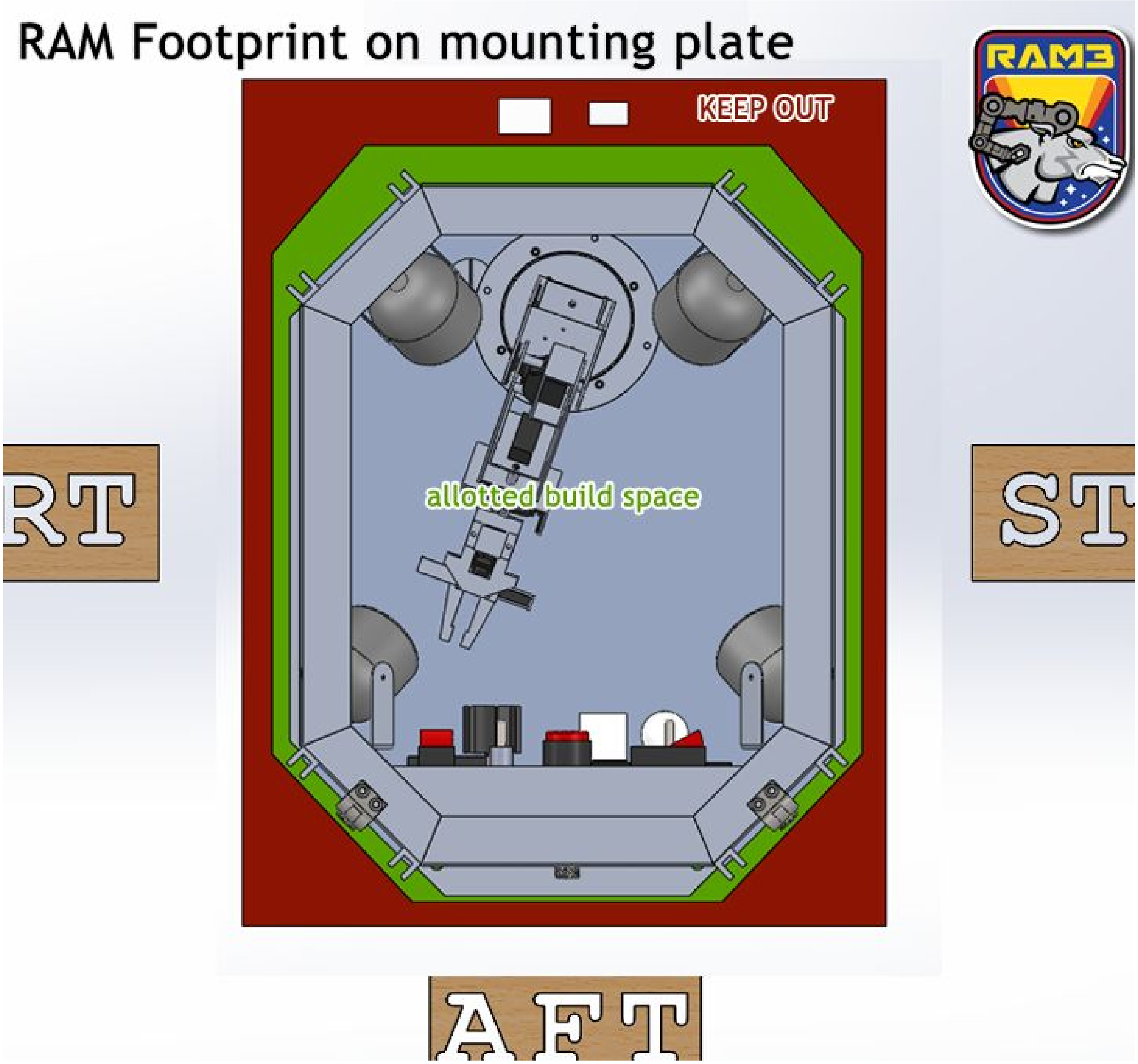
	Part File Name	Material	Quantity	**Mass (g)**	Error (±g)
Frame and Body					
	Right Aluminum (Panel)	6061 Al Sheet	1	58.8	0.1
	Mid Plate	6061 Al Sheet	1	427.3	0.1
	Right Aluminum Panel (EDAC and DB9) Ports	6061 Al Sheet	1	75.3	0.1
	Left Aluminum Panel	6061 Al Sheet	1	297.8	0.1
	Front Aluminum Panel	6061 Al Sheet	1	78.5	0.1
	Bottom Aluminum Panel	6061 Al Sheet	1	459.6	0.1
	Front And Rear Clear Panel	Lexan	2	88.9	0.1
	Top Panel	Lexan	1	128.8	0.1
	Frame Assembly	1"x1"x0.125" Al Angle	1	1834.4	0.1
	Panel Fasteners	18-8 Stainless	15	1.5	0.5
	Mount Plate Fasteners	1/4" 18-8 Stainless	8	1.7	0.5
	Washers/Misc Fasteners	Stainless		1.0	0.5
	subtotal			**3453.5**	**2.4**
Robotic Arm Base					
	28mm Hex Standoff	Stainless	4	64.3	1.0
	10mm Hex Standoff	Stainless	4	22.6	1.0
	Lower Plate	6061 Al Sheet	1	32.3	0.1
	Mid Plate Base Assembly	6061 Al Sheet	1	33.1	0.1
	Top Plate	6061 Al Sheet	1	15.6	0.1
	Top Plate Outer	6061 Al Sheet	1	16.2	0.1
	0.25" Thin Ball Bearing	Stainless Steel	1	27.0	2.0
	Bearing Fasteners	18-8 Stainless	4	0.4	0.5
	Bearing Hex Nut	18-8 Stainless	4	0.3	0.5
	subtotal			**211.8**	**5.4**
Robotic Arm (Lower Assembly)					
	Arm Bracket A	6061 Al Sheet	1	14.8	0.1
	Arm Plate A	6061 Al Sheet	1	22.6	0.1
	Arm Plate B	6061 Al Sheet	1	23.2	0.1
	Lower Arm Inner Bracket	6061 Al Sheet	1	10.1	0.1
	Servo Arm Bracket	6061 Al Sheet	1	7.7	0.1
	Servo End Bracket	6061 Al Sheet	1	9.3	0.1

	Small Screws	18-8 Stainless	8	0.1	0.5
	Pivot Fastener	18-8 Stainless	1	0.1	0.5
	Bracket Fasteners	18-8 Stainless	6	0.7	0.5
	subtotal			**88.6**	**2.1**
Upper Robotic Arm					
	Arm Bracket B	6061 Al Sheet	1	15.2	0.1
	Arm Plate C	6061 Al Sheet	1	19.2	0.1
	Arm Plate D	6061 Al Sheet	1	18.7	0.1
	Servo End Bracket	6061 Al Sheet	1	9.3	0.1
	Servo Arm Bracket	6061 Al Sheet	1	7.7	0.1
	Arm Bracket C	6061 Al Sheet	1	12.7	0.1
	Small Screws	18-8 Stainless	12	0.1	0.5
	Pivot Fastener	18-8 Stainless	1	0.1	0.5
	subtotal			**82.8**	**1.6**
Robotic Hand					
	Servo Bracket A	6061 Al Sheet	1	2.8	0.1
	CV and Servo Combo Bracket	6061 Al Sheet	1	9.8	0.1
	Left Jaw	Al Stock	2	6.8	1.0
	Gripper Base	Al Stock	1	22.3	1.0
	Gripper Cap	Al Stock	1	4.2	1.0
	Bracket Fasteners	18-8 Stainless Steel	4	0.2	0.5
	Cap Fasteners	18-8 Stainless Steel	6	0.1	0.5
	Track	Plastic	2	0.1	0.1
	subtotal			**46.3**	**4.3**
Electronics					
	Regulator Housing	6061 Al Sheet/Al Stock	n/a	59.2	0.1
	Arduino Mega/MK100 Rack	ABS	1	56.2	1.0
	Arduino Mega/MK100 Delrin Plate	Delrin	1	21.0	1.0
	Raspberry Pi/Robo Pi Rack	ABS	1	52.0	1.0
	Raspberry Pi/Robo Pi Delrin Plate	Delrin	2	27.0	1.0
	Wiring Allowance	various	n/a	750.0	225.0
	Arduino MK100	n/a	1	32.0	1.0
	Sparkfun R232 Shifter	n/a	1	30.0	1.0
	LM2596 DC to DC	n/a	3	59.5	1.0
	Arduino MEGA	n/a	1	37.0	1.0
	BME280	n/a	4	4.0	1.0
	Weatherproof TTL Camera	n/a	4	600.0	100.0
	Arduino Micro	n/a	1	65.0	1.0
	Raspberry Pi	n/a	1	31.0	1.0

	RoboPi Expansion Board	n/a	1	15.0	1.0
	Thermocouple Amp MAX31855	n/a	14	18.6	1.0
	Servo Batan B2122	n/a	7	110.7	1.0
	Pixycam CMUcam5	n/a	1	25.5	1.0
	Fasteners	various	8	2.0	1.0
	subtotal			**1995.7**	**341.1**
Busy Box					
	Busy Box Mounting Post	Al Stock	2	675.8	0.1
	Busy Box Mounting Plate	6061 Al Sheet	1	186.5	0.1
	Slider Block Brackets	6061 Al Sheet	2	51.9	0.1
	Push Button Actuators	n/a	2	48.0	5.0
	Toggle Switches	n/a	2	143.0	5.0
	Rocker Switches	n/a	2	76.0	5.0
	Potentiometer	n/a	1	22.0	5.0
	Potentiometer Knob	ABS	1	19.8	5.0
	Cube Slider Block (w/linkage)	ABS	1	21.7	5.0
	Sphere Slider Block (w/linkage)	ABS	1	13.4	5.0
	Velcro Strips for Slider Blocks		2	7.0	5.0
	Velcro Strap		1	45.0	5.0
	Elastic Strapping	CotS	3	85.0	5.0
	Fasteners	Stainless steel	17	20.0	5.0
	subtotal			**1415.1**	**55.3**
				Mass (g)	Error (±g)
		Mass Total:		**7293.7**	**412.2**

2. *Mounting plate footprint*

- Please note in the model below (where the 'Keep-Out' part of the plate has been colored red and the normal build plate colored green) that we fit within the allotted space. See the 'Drawings' section for more detailed illustrations of this point.


RAM Footprint on mounting plate

3. *Payload height*

- Our payload height is 30.0 cm, or 11.81 in. See the ‘Drawings’ section for more detailed illustrations of this point.

4. *Power Budget: Introduction*

A large payload on HASP provides a total power of 75 watts (30 volts @ 2.5 amps.) RAM will use this, and only this, to power four separate systems:

- network system,
- camera system,

- Busy Box,
- robotic arm.

RAM will take power from HASP through the EDAC cable and plug it directly into a power distribution block that will step it to appropriate voltages and distribute it to the four systems. Our expected peak draw will be around 52 watts and will not exceed the given 75 watts. To this end, RAM will be tested for its full operation with a cold fuse and a power supply that does **not** restrict itself to pushing only 75 watts.

- ***Power Budget: Design***

  RAM will consist of four major subsystems, each with their own power needs. To do this, RAM will have an easily-accessed power block that will step voltages to appropriate levels and distribute them to each system. This power block will receive its power directly from HASP via the EDAC connector on pins A,B,C,D for power and W,T,U,X for ground. Each of these four power and ground wires will come together before connecting to the power block. The subsystems and their components are as follows:

  1. **Network management system**
     a. <u>Components</u>: Arduino MKR-1000 and an RS232 shifter.
     b. <u>Purpose:</u> Send and receive data from the HASP gondola and manage the flow of data on RAM's local I^{2}C network.
     c. <u>Resources:</u> This network will run on an expected voltage of 5V and should only be a consistent and gentle power draw of 1.25 W.
  2. **Camera system**

a. Components: Arduino Mega, 2-4 space-ready Arduino-compatible Weatherproof TTL Serial JPEG Cameras, and 2-4 BME-280 environmental sensors.

b. Purpose:

   i. Take still images on request for troubleshooting

   ii. Film video of operations for post-flight review.

   iii. Each camera will have a simple environmental sensor located near it to take readings on ambient temperature, pressure, and humidity. The Arduino in this system will be responsible for managing and reporting this data.

c. Resources: This system will run on a consistent 12V. This system in particular will be tested for current spikes when the camera takes pictures. If there is a need, the team will design capacitors into the circuit to smooth out its power draw.

3. **"Busy Box" system**

   a. Components: Arduino Micro, various switches and potentiometers.

   b. Purpose: The robotic arm will be interacting with this system through its trials; the Busy Box records progress: tracking button presses, switch throws, etc.

   c. Resources: This system will run on a consistent 5V and expects to see a consistent and gentle power draw of less than 0.25W.

4. **Robotic arm**

   a. Components: Raspberry PI B+ and a Mikronauts RoboPi expansion board, seven Analog Feedback Micro servos (Adafruit PID 1450), a Pixy CMUcam5 Sensor, an Arduino-compatible servo controller, 14 K-type thermocouples with Thermocouple Amplifier MAX31855 breakout boards (two thermocouples for each servo[1]), and seven capacitors. This system is by far the most complicated.

   b. Purpose: Drive the operation of the robot arm through its pre-designed batteries of dexterity tests.

   c. Resources: This will be the largest power draw on the whole system. The team will likely use a capacitor on each servo to help smooth out the operation, but we may add more up the line to smooth out the power draw. This system will draw a varied amount of voltage from 4.8 to 5V depending on its current action and may see large shifts in power draw over the course of the flight.

[1] We may reduce this number if the cabling gets too cumbersome.

- ***Power Budget: Planned Testing***

  Electrical testing will be a major focus this year for the team, especially after our difficulties with HASP 2017. The entire payload will operate with a 2.5 A fuse inline immediately after the 30 V supply from the simulated HASP source. We will map a true power profile of payload operation, focusing especially on the arm, to more thoroughly address any potential areas of concern. In addition to the power profile, when performing thermal tests, the team will make sure that the fuse is also put through the same heat and cold cycles to make sure that it does not pose a hidden threat to the operation of the system through adverse thermal conditions.

Total Power Draw for Entire Payload		
System	**Voltage (V)**	**Peak Power (W)**
Network	5	1.25
Camera	12	2.14
Busy Box	5	0.10
Robotic Arm	4.8-5[2]	47.47
Total Expected Peak Draw:		**51.96**

Total Power Draw: Network System				
Device	**#**	**Voltage (V)**	**Peak Power (W)**	**Time On**
Arduino MKR 100	1	5.0	1.0	Always On
Sparkfun RS232 Shifter - SMD	1	5.0	0.25	Always On

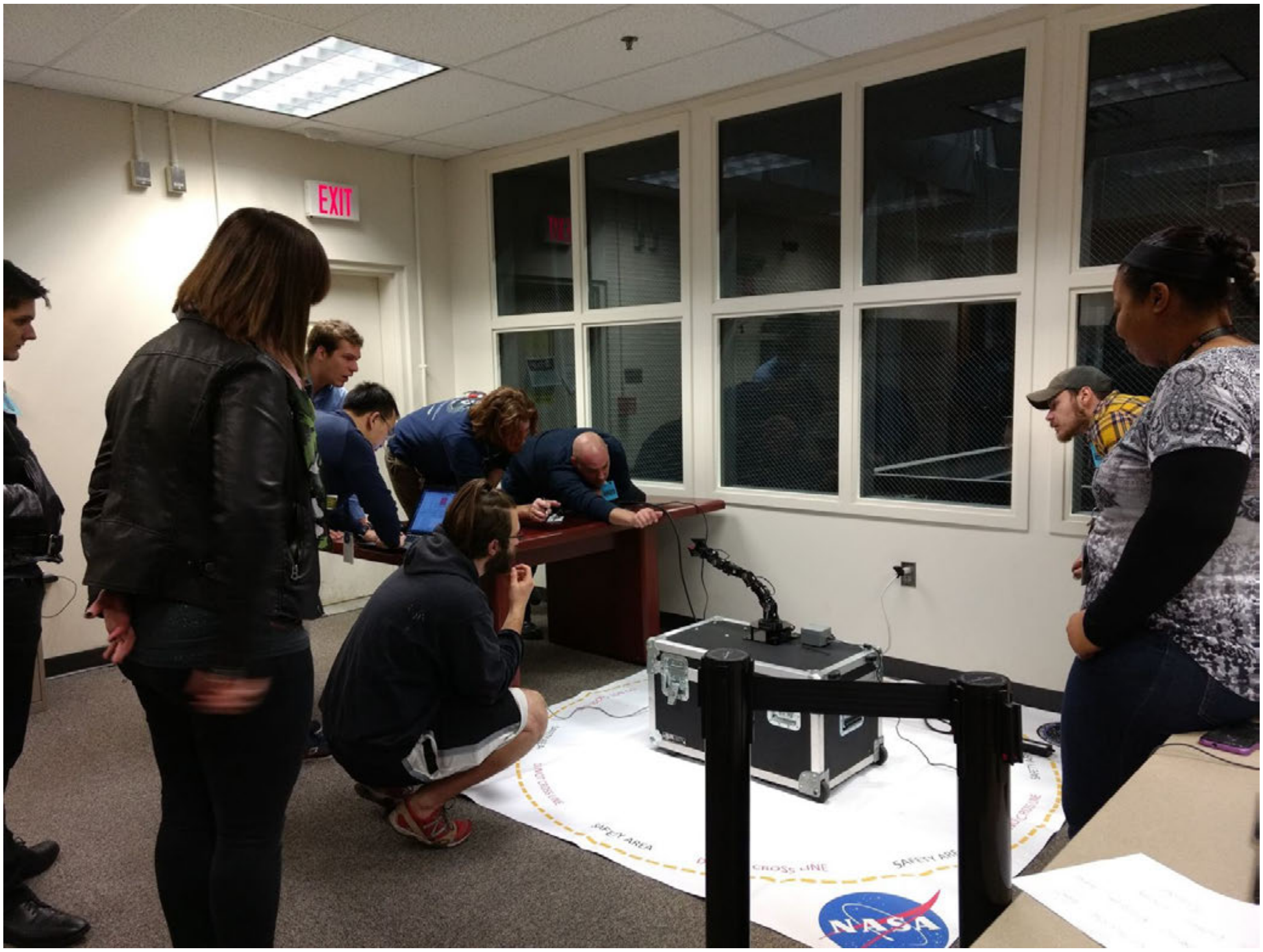
Total Power Draw: Camera System				
Device	**#**	**Voltage (V)**	**Peak Power (W)**	**Time On**
Arduino Mega	1	12.0	0.08	Always On
Adafruit BME280	4	3.0	< 0.01	Always On
Adafruit Weatherproof TTL Camera	4	5.0	1.30	Always On

[2] 5V for most of the system, 4.8V for servos.

Total Power Draw: Busy Box System				
Device	**Count**	**Voltage (V)**	**Peak Power (W)**	**Time On**
Arduino Micro	1	5.0	0.10	Always On

Total Power Draw: Robot Arm System				
Device	**Count**	**Voltage (V)**	**Peak Power (W)**	**Time On**
Raspberry Pi B+	1	5.0	6.70	Always On
RoboPi Expansion Board	1	5.0	5.00	During Trials
Thermocouple Amplifiers MAX31855	14	3.3	0.07	Always On
Servo Batan B2122[3]	7	4.8	35.0	During Trials
Pixycam CMUcam5	1	5.0	0.7	Always On

[3] System will use capacitors to smooth out operation and counteract increased power draw from inductance at the start of operation.



5. Downlink serial telemetry rate & uplink serial command rate

Serial Uplink and Downlink: Introduction

A large payload on HASP can send commands from the ground and data to the HASP gondola for downlinking at a baud rate of 4800. This Serial Uplink and Downlink system allows for payloads to communicate with the HASP gondola over an RS232 line; the HASP gondola then communicates with the ground. RAM will have an Arduino MKR-1000 whose sole job will be to manage all communications, not only between RAM and the HASP gondola, but also the local I^{2}C network between subsystems.

Serial Uplink and Downlink: Design

RAM will use an Arduino MKR-1000 to communicate with the HASP gondola over an RS232 shifter, as Arduino does not have a built in RS232 output. This RS232 shifter will take TTL output from the Arduino and shift it to RS232 levels, allowing communication with the HASP gondola. Arduino’s native TTL Serial library can be throttled to 4800 baud rate, which guarantees that RAM never exceeds its allotted bandwidth. Additionally, Arduino serial default communication parameters are aligned with HASP's (8 data bits, no parity, 1 stop bit, and no flow control.) The managing Arduino will act as sentinel and secretary for all data traveling between HASP gondola and payload.

All commands sent from the ground will go through the RS232 shifter and to the managing Arduino. The first byte of all commands will be reserved for identifying which device on the I^{2}C network the command is meant for. The second byte will be the command itself. The manager will take the received command and convert it to a format that is acceptable for the I^{2}C local network and send it over the network with the appropriate ID proceeding it.

RAM will plan to receive a GPS message from HASP every 60 seconds for time syncing since Arduinos do not (natively) carry any internal clocks.

All data sent from RAM subsystems will pass through the network master first. This means that all experimental related data will be coming to the master over the local I^{2}C network first. As such, it will be the master Arduino's job to manipulate the data into the planned formats and use bandwidth in the most efficient manner for downlinking with the 4800 baud restriction. The master will also be capable of storing data over a period of time for downlinking at a later instance, as pictures being sent from the camera subsystem may take more than one packet to get the entire picture down, and thus will utilize all available bandwidth for a time. Other information will fill up a queue for downlinking later.

Serial Uplink and Downlink: Planned Testing:

For testing the Serial Uplink and Downlink system, the team plans to develop a true HASP simulator script this year. In the previous year's flight, the team relied on using a second Arduino as a HASP simulator but it was frustrating to use; an Arduino cannot compare to a proper computer. The simulator will also mimic HASP software in generating files in 30kb chunks. We will write software to parse this information out quickly and efficiently. A stretch goal of ours is to have the servo position recordings pulled out and ported either into simulation or a mimic arm that will perform the exact actions our arm just performed at float. This will give the team live feedback (with a slight delay) of what actions the arm is doing during operation!

"Data Packet"; assume a standard packet size of 520 bytes	Byte	Title
Simple '\x1\x21' header indicating the start of a new packet of data.	1–2	Header
I2C ID specific to the system that generated the data.	3	System of Origin
8 byte time_t value: time data was sent from network manager to HASP gondola.	4–12	Time Sent Gondola
8 byte time_t value: time data was sent from its system of origin to the network manager.	13–21	Time Sent Master
Single byte: number of different data chunks in the Data Packet section.	22	Number of Data Chunks
Two bytes: total size of actual data represented in the Data Packet section.	23–24	Total Size of Data Chunks
Checksum / allows room for growth as the project develops.	25–30	Checksum / Filler
Actual meat of a packet. It could be any of the following: *- image data* *- ambient condition data* *- robotic arm data* *- Busy Box status* *- trial data*	51–518	Data Packet
Simple '\x3\xD' terminator indicating the end of a packet of data.	519–520	Terminator

Image Data; assume a maximum packet size of 467 bytes	Byte	Title
Constant for image parts indicating type: '\x30'	1	Type
Indicates position in an image for reconstruction.	2	Part
Which photo this part belongs to.	3–4	Photo ID
Actual amount of valid data sent in the image part portion.	5–6	Size
Data of an actual part of an image.	7–467	Image Part

Ambient Conditions; assume a static packet size of 28 bytes	Byte	Title
Constant for ambient conditions and busy box status: '\x31'	1	Type
8 byte time_t value: time these readings were recorded.	2–9	Time
Signed short value: current temperature from BME280 #1.	9–10	BME280 #1 Temp.

Signed short value: current pressure from BME280 #1.	11–12	BME280 #1 Pressure
Unsigned byte value: current humidity from BME280 #1.	13	BME280 #1 Humid.
Signed short value: current temperature from BME280 #2.	14–15	BME280 #2 Temp.
Signed short value: current pressure from BME280 #2.	16–17	BME280 #2 Pressure
Unsigned byte value: current humidity from BME280 #2.	18	BME280 #2 Humid.
Signed short value: current temperature from BME280 #3.	19–20	BME280 #3 Temp.
Signed short value: current pressure from BME280 #3.	21–22	BME280 #3 Pressure
Unsigned byte value: current humidity from BME280 #3.	23	BME280 #3 Humid.
Signed short value: current temperature from BME280 #4.	24–25	BME280 #4 Temp.
Signed short value: current pressure from BME280 #4.	26–27	BME280 #4 Pressure
Unsigned byte value: current humidity from BME280 #4.	28	BME280 #4 Humid.

Busy Box Status; assume a static packet size of 19 bytes	Byte	Title
Constant for the busy box status: '\x32'	1	Type
8 byte time_t value: time these readings were recorded.	2–9	Time
boolean value: position ON or OFF	10	Switch 1
boolean value: position ON or OFF	11	Switch 2
boolean value: position ON or OFF	12	Switch 3
boolean value: position ON or OFF	13	Switch 4
boolean value: position ON or OFF	14	Switch 5
boolean value: position ON or OFF	15	Switch 6
boolean value: position ON or OFF	16	Switch 7
boolean value: position ON or OFF	17	Switch 8
boolean value: position ON or OFF	18	Switch 9
Current reading from the potentiometer	19	Potentiometer

Robot Arm Conditions; assume a static packet size of 44 bytes	Byte	Title
Constant for robot arm conditions: '\x33'	1	Type
8 byte time_t value: time these readings were recorded.	2–9	Time
Signed short: temperature readout from this thermocouple.	10–11	Thermocouple 1
Signed short: temperature readout from this thermocouple.	12–13	Thermocouple 2
Signed short: temperature readout from this thermocouple.	14–15	Thermocouple 3
Signed short: temperature readout from this thermocouple.	16–17	Thermocouple 4

Signed short: temperature readout from this thermocouple.	18–19	Thermocouple 5
Signed short: temperature readout from this thermocouple.	20–21	Thermocouple 6
Signed short: temperature readout from this thermocouple.	22–23	Thermocouple 7
Signed short: temperature readout from this thermocouple.	24–25	Thermocouple 8
Signed short: temperature readout from this thermocouple.	26–27	Thermocouple 9
Signed short: temperature readout from this thermocouple.	28–29	Thermocouple 10
Signed short: temperature readout from this thermocouple.	30–31	Thermocouple 11
Signed short: temperature readout from this thermocouple.	32–33	Thermocouple 12
Signed short: temperature readout from this thermocouple.	34–35	Thermocouple 13
Signed short: temperature readout from this thermocouple.	36–37	Thermocouple 14
Current analog position reading from the B2122 servo.	38	Servo Position 1
Current analog position reading from the B2122 servo.	39	Servo Position 2
Current analog position reading from the B2122 servo.	40	Servo Position 3
Current analog position reading from the B2122 servo.	41	Servo Position 4
Current analog position reading from the B2122 servo.	42	Servo Position 5
Current analog position reading from the B2122 servo.	43	Servo Position 6
Current analog position reading from the B2122 servo.	44	Servo Position 7

Trial Data; here we assume a max possible size of 467 bytes	**Byte**	**Title**
Constant for trial data: '\x34'	1	Type
Value for trial type (normal, difficult, easy, integration test, etc.)	2	Trial Type
Unsigned short indicating the unique ID number for this specific trial.	3–4	Trial ID
Unsigned byte that indicates which mode the arm was in.	5	Mode
8 byte time_t value: start time of this trial.	6–14	Start Time
8 byte time_t value: end time of this trial.	15–23	End Time
IDs for specific actions (in case we need to debug/parse individual motions)	24	Action ID
8 byte time_t value: starting time of an action.	25–33	Action N Start Time*
8 byte time_t value: end time of an action.	34–42	Action N End Time*
**** TO N FOR THE NUMBER OF ACTIONS***		

Command List

Name	Byte 1	Byte 2	Description
--	*I2C ID*	*Command ID*	*General format for all commands. First byte indicates to the network manager which system this command should go to. The second byte is the actual command.*

Network System		
Name	**Command ID**	**Description**
Request Network Table	0x7F	Downlink a list of all systems on the network and their IDs.
Reset	0x31	Force the network manager arduino to reset.
System Reset	0x32	Instruct the network manager to send a hard reset command to all entities on the network.

Camera System		
Name	**Command ID**	**Description**
Request Picture 1	0x7F	Take picture with camera 1, send over I2C to be downlinked when bandwidth is available.
Request Picture 2	0x7E	Take picture with camera 2, send over I2C to be downlinked when bandwidth is available.
Request Picture 3	0x7D	Take picture with camera 3, send over I2C to be downlinked when bandwidth is available.
Request Picture 4	0x7C	Take picture with camera 4, send over I2C to be downlinked when bandwidth is available.
Request Ambient	0x7B	Pack update and send for downlinking.
Toggle AutoUpdate	0x7A	Camera system automatically reports ambient temp every minute; toggles that on or off.
Reset	0x31	Force the camera system arduino to reset.

Busy Box System		
Name	**Command ID**	**Description**
Request Status	0x7F	Request a system update from the Busy Box.
Toggle AutoUpdate	0x7E	Busy Box automatically reports any changes (ie, switches switched, potentiometers turned); this toggles that on or off.
Reset	0x31	force the Busy Box Arduino to reset.

Robotic Arm System		
Name	**Command ID**	**Description**
Warmup + Diagnostic	0x2E	Put the robot arm into warmup mode: move from resting position to operating position, perform a few simple actions, and report back its progress.
Full Stop	0x2F	Stop anything it is doing, clear all command queues, and return to its resting position.
Cancel	0x30	Cancel any trial in operation and force the arm back to its resting position.
Reset	0x31	Force the Raspberry Pi and RoboPi expansion board to reset.
Manual Operation	0x32	Force the robotic arm into manual operation. In this state, it will perform no actual without prompting from the ground via Serial.
Automatic Operation	0x33	Force the robotic arm into automatic operation. In this state, it will perform trial action when prompted by the ground via Serial.
Mode 1	0x34	Force the robotic arm into Mode 1.
Mode 2	0x35	Force the robotic arm into Mode 2.
Mode 3	0x36	Force the robotic arm into Mode 3.
Pause	0x37	Stop arm movement; hold in place until further action is sent, or until this command is sent again.
Reset Queue	0x38	Clear any queued trials.
Perform Trial 1	0x40	[Automatic Mode] signal that the arm should either start performing this trial or queue up to do this trial.
Perform Trial 2	0x41	[Automatic Mode] signal that the arm should either start performing this trial or queue up to do this trial.
Perform Trial 3	0x42	[Automatic Mode] signal that the arm should either start performing this trial or queue up to do this trial.
Perform Trial 4	0x43	[Automatic Mode] signal that the arm should either start performing this trial or queue up to do this trial.
Perform Trial 5	0x44	[Automatic Mode] signal that the arm should either start performing this trial or queue up to do this trial.
Perform Trial 6	0x45	[Automatic Mode] signal that the arm should either start performing this trial or queue up to do this trial.
Perform Trial 1-3	0x46	[Automatic Mode] signal that the arm should either start performing this trial or queue up to do this series of trials
Perform Trial 4-6	0x47	[Automatic Mode] signal that the arm should either start performing this trial or queue up to do this series of trials
Perform Trial 2-4	0x48	[Automatic Mode] signal that the arm should either start performing this trial or queue up to do this series of trials

Increment Servo 1 1	0x7F	[Manual Mode] force a specific servo to move 1 degree clockwise.
Increment Servo 2 1	0x7E	[Manual Mode] force a specific servo to move 1 degree clockwise.
Increment Servo 3 1	0x7D	[Manual Mode] force a specific servo to move 1 degree clockwise.
Increment Servo 4 1	0x7C	[Manual Mode] force a specific servo to move 1 degree clockwise.
Increment Servo 5 1	0x7B	[Manual Mode] force a specific servo to move 1 degree clockwise.
Increment Servo 6 1	0x7A	[Manual Mode] force a specific servo to move 1 degree clockwise.
Increment Servo 7 1	0x79	[Manual Mode] force a specific servo to move 1 degree clockwise.
Decrement Servo 1 1	0x78	[Manual Mode] force a specific servo to move 1 degree counterclockwise.
Decrement Servo 2 1	0x77	[Manual Mode] force a specific servo to move 1 degree counterclockwise.
Decrement Servo 3 1	0x76	[Manual Mode] force a specific servo to move 1 degree counterclockwise.
Decrement Servo 4 1	0x75	[Manual Mode] force a specific servo to move 1 degree counterclockwise.
Decrement Servo 5 1	0x74	[Manual Mode] force a specific servo to move 1 degree counterclockwise.
Decrement Servo 6 1	0x73	[Manual Mode] force a specific servo to move 1 degree counterclockwise.
Decrement Servo 7 1	0x72	[Manual Mode] force a specific servo to move 1 degree counterclockwise.
Increment Servo 1 5	0x71	[Manual Mode] force a specific servo to move 5 degrees clockwise.
Increment Servo 2 5	0x70	[Manual Mode] force a specific servo to move 5 degrees clockwise.
Increment Servo 3 5	0x6F	[Manual Mode] force a specific servo to move 5 degrees clockwise.
Increment Servo 4 5	0x6E	[Manual Mode] force a specific servo to move 5 degrees clockwise.
Increment Servo 5 5	0x6D	[Manual Mode] force a specific servo to move 5 degrees clockwise.
Increment Servo 6 5	0x6C	[Manual Mode] force a specific servo to move 5 degrees clockwise.
Increment Servo 7 5	0x6B	[Manual Mode] force a specific servo to move 5 degrees clockwise.
Decrement Servo 1 5	0x6A	[Manual Mode] force a specific servo to move 5 degrees counterclockwise.
Decrement Servo 2 5	0x69	[Manual Mode] force a specific servo to move 5 degrees counterclockwise.
Decrement Servo 3 5	0x68	[Manual Mode] force a specific servo to move 5 degrees counterclockwise.
Decrement Servo 4 5	0x67	[Manual Mode] force a specific servo to move 5 degrees counterclockwise.
Decrement Servo 5 5	0x66	[Manual Mode] force a specific servo to move 5 degrees counterclockwise.
Decrement Servo 6 5	0x65	[Manual Mode] force a specific servo to move 5 degrees counterclockwise.
Decrement Servo 7 5	0x64	[Manual Mode] force a specific servo to move 5 degrees counterclockwise.

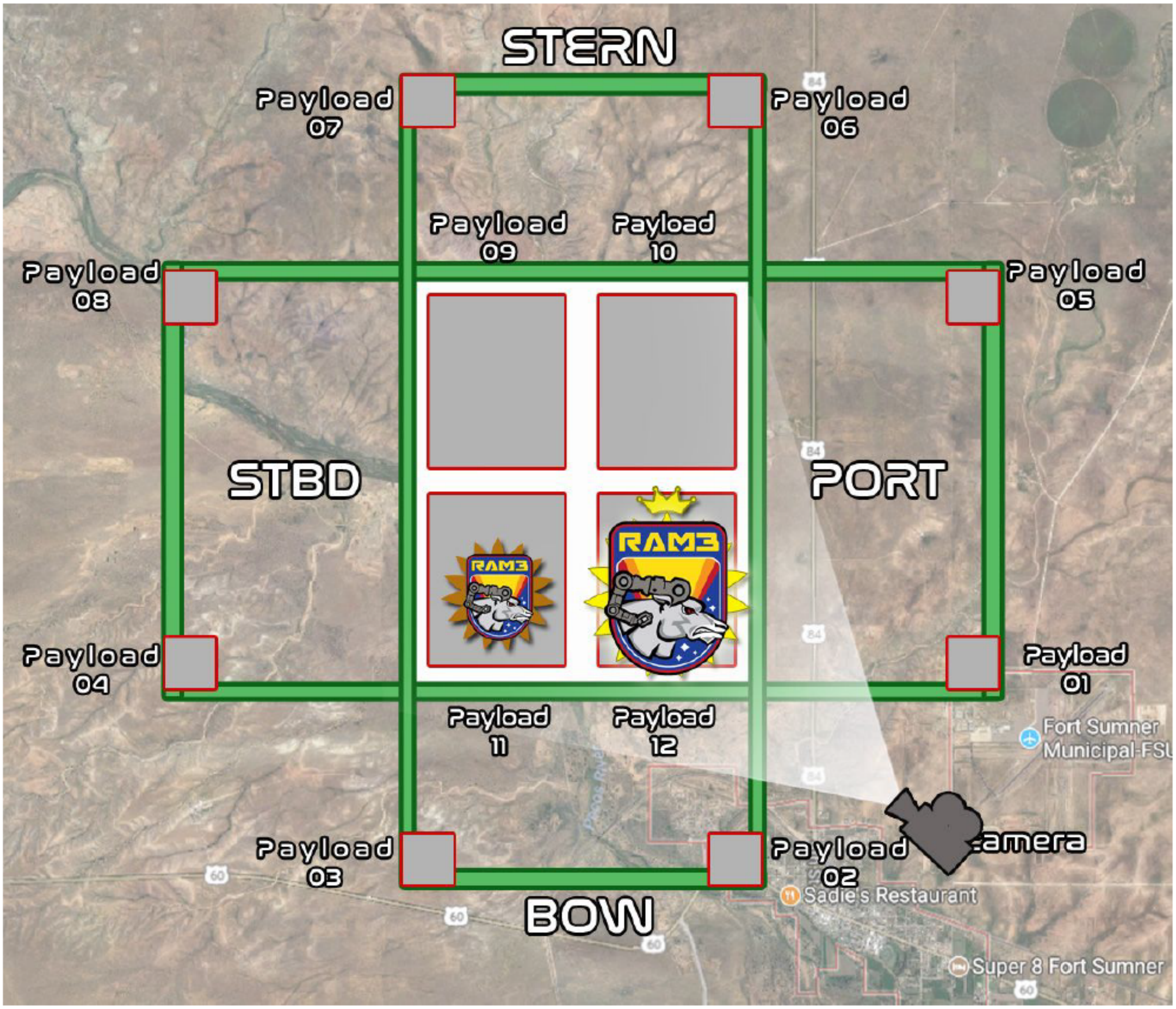
Perform Wave to Camera	0x30	[Manual Mode] perform a waving action at the HASP gondola camera.

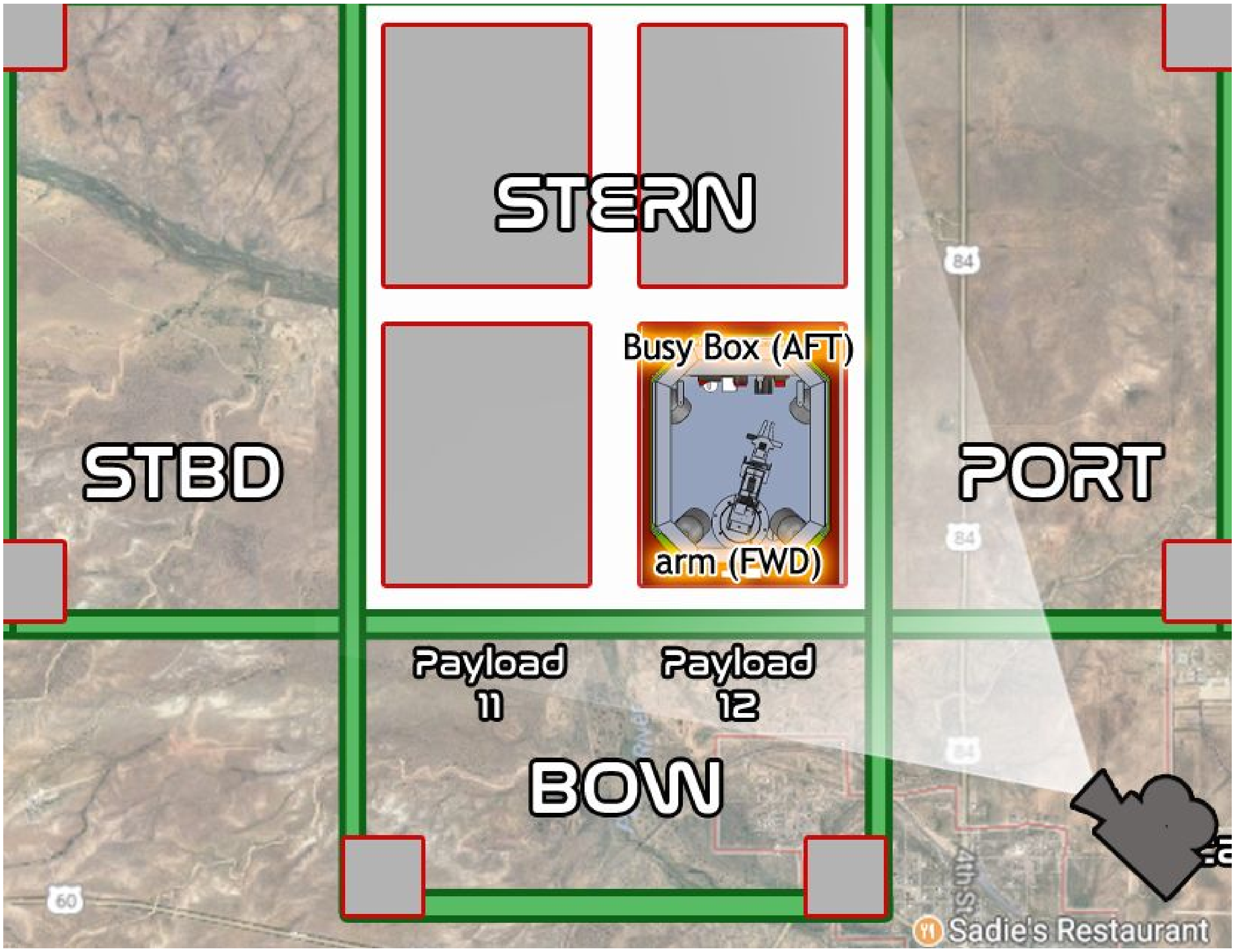
6. *Anticipated use of analog downlink channels or additional discrete commands*

- We will **not** be using analog downlink channels.
- We will **not** be using any additional discrete commands.

7. *Desired payload location and orientation*

- We have a mild preference for payload slot #12, as it is located closest to the camera (assuming that the HASP camera location does **not** change.) We'll fly our own cameras but would like to maximize the numbers of eyes we can get on our moving parts for best troubleshooting in the event of an error. #11 is a fine 2nd choice. We'll happily take whatever we can get, though.




STERN
Busy Box (AFT)
arm (FWD)
STBD
PORT
Payload
11
Payload
12
BOW
84
84
84
60
4th St
Sadie's Restaurant

8. *Potential hazards*

high	F. Personnel Burn Out		B. Personnel drop out	A. Wiring / software bug causes surge / blows HASP fuse
SEVERITY	J. Sapient robotic uprising (“Skynet event”) delays project and/or enslaves humanity	G. Higher complexity of networked computers	D. Glare and inconsistent lighting may interfere with our computer vision	C. Limitations of ground testing methods result in unexpected thermal build up
			H. Behind Schedule (we’re not, but we might!)	E. Durham Tech business office sabotages / introduces significant delays to procurement pipeline
low		L. Added electronic complexity makes assembly/maintenance awkward/tedious	K. Welding aluminum (for our frame) is finicky and unforgiving	I. Not accepted to HASP again
	low	POSSIBILITY		high

Mitigation strategy:

A. Wiring/software bug - Exhaustive, methodical testing is the best way to catch something like this before flight. THIS TIME WE’RE USING AN OSCILLOSCOPE. We have, in the past, only tested with current-limited power supplies; we will ensure that future testing reveals any potential overdraws.

B. Personnel Drop Out - This is a big risk. We’re going to front-load as much as possible with build days through December and January.

C. Thermal Build-Up - We’ll test our equipment at Goddard again and do some theoretical and experimental thermal modeling, but even the best T/V test isn’t a great simulator of solar load. At the end of the day, our worst-case plan is to add a large radiator/heatsink to the electronics. Our servos were selected based on U. Bridgeport’s 2015 HASP experiment, so we are reasonably confident in their spaceworthiness.

D. Glare - Fighting glare is a significant problem when our project is driven by computer vision. We can mitigate this with polarizing filters, non-reflective coatings, and adding our own lighting to make more consistent visual conditions. We have paired with two organizations familiar with this issue (Duke’s Marine Lab and University of New Mexico’s ‘Swarmathon’, which runs a yearly computer-vision competition.)

E. Business Office - We are continuing to meet with the Durham Tech business office to resolve previous issues with purchasing equipment & supplies and, in parallel, pursuing independent funding.

F. Personnel Burn Out - The majority of our team is in their senior year. As above, we are trying to front-load as much development as possible.

G. Added Software Complexity - With the added complexity of 5 networked systems, the odds that unforeseen bugs might surface increase. With our front-loaded development schedule we will have more time to test.

H. Behind Schedule - Ms. Hoover will yell at us more than normal if this contingency arises.

I. Not Accepted to HASP - If we don't get in this year, we'll just run this as a terrestrial experiment.

J. Robot Revolution/Uprising - Our robot has no legs.

K. Welding Aluminum - We're aware of the delicacy of welding aluminum, as we are planning to do for large sections of the frame due to the strength, rigidity, and long-term ease of assembly. We have a trained machinist with experience welding aluminum, however, and are comfortable with the risk.

L. Added Electronic Complexity - We are mounting our electronics on sliding racks for ease of maintenance and migrating over to Molex-style connectors rather than screw terminals.

9. *Integration & Flight Procedures*

Briefly describe of your anticipated procedures during integration with HASP and flight operations:

- When we arrive at CSBF, we will first power on RAM to ensure there was no damage sustained during travel.
- We will have written a ~5-minute "integration trial" that runs through the most rigorous possible combination of arm maneuvers and that draws the most electrical current. This will be helpful for us, internally, as a systems check, and for us, externally, to prove we operate within the power constraints of HASP. This will involve moving all our arm's servos at once while also powering the network, main computer, and Busy Box.
- Once we are integrated to HASP, we will run the integration trial. If we do not blow a fuse or otherwise exhibit electronic or mechanical failure during this trial (measured by the arm reaching target positions within a set time and corroborated by onboard system logs) we will regard it successful.

- The integration trial will output a special message that will confirm successful integration at-a-glance in the data.
- This should take 10-20 minutes.

10. *Additional Resources Requested*

You may also request resources that somewhat exceed those specified for your payload class or those that are not mentioned in this document. *However, each such request must be accompanied by a description of the impact if the requested resource is not granted.*

- We **do not require** any[4] additional resources or accommodations.

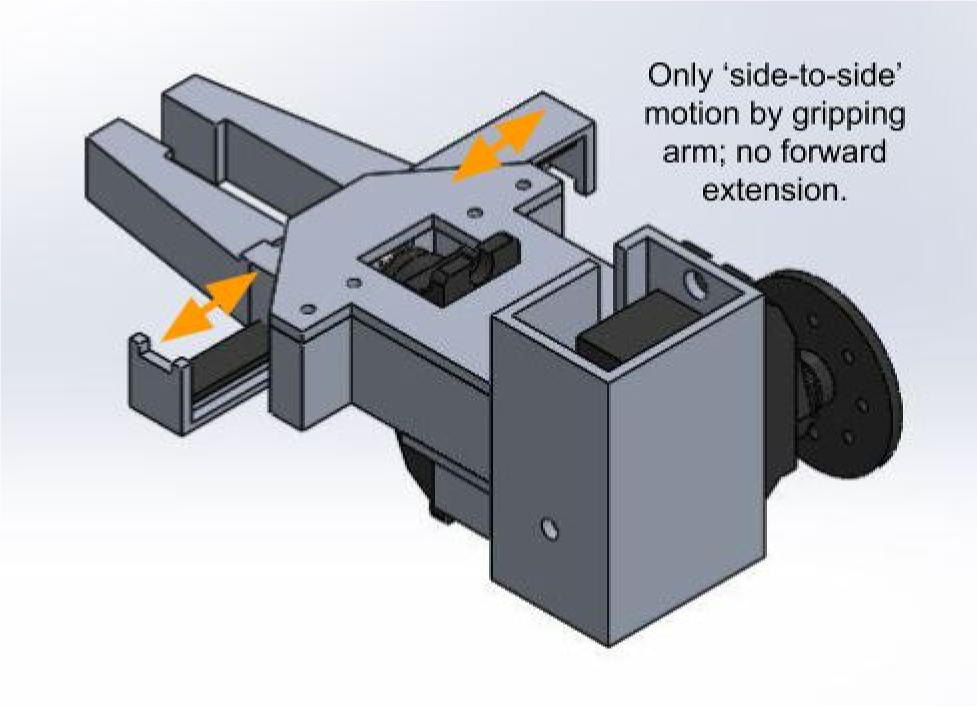
- Our payload design contains no pressure vessels, radioactive materials, biological materials, lasers, cryogenic materials, high voltage, strong magnets, pyrotechnics, intentionally-dropped components, or hazardous chemicals.

[4] Other than allowing our underaged, open-toed, foreign-national tarantula, wielding high-pressure propane tanks, on the flight line unsupervised.

Preliminary Drawings

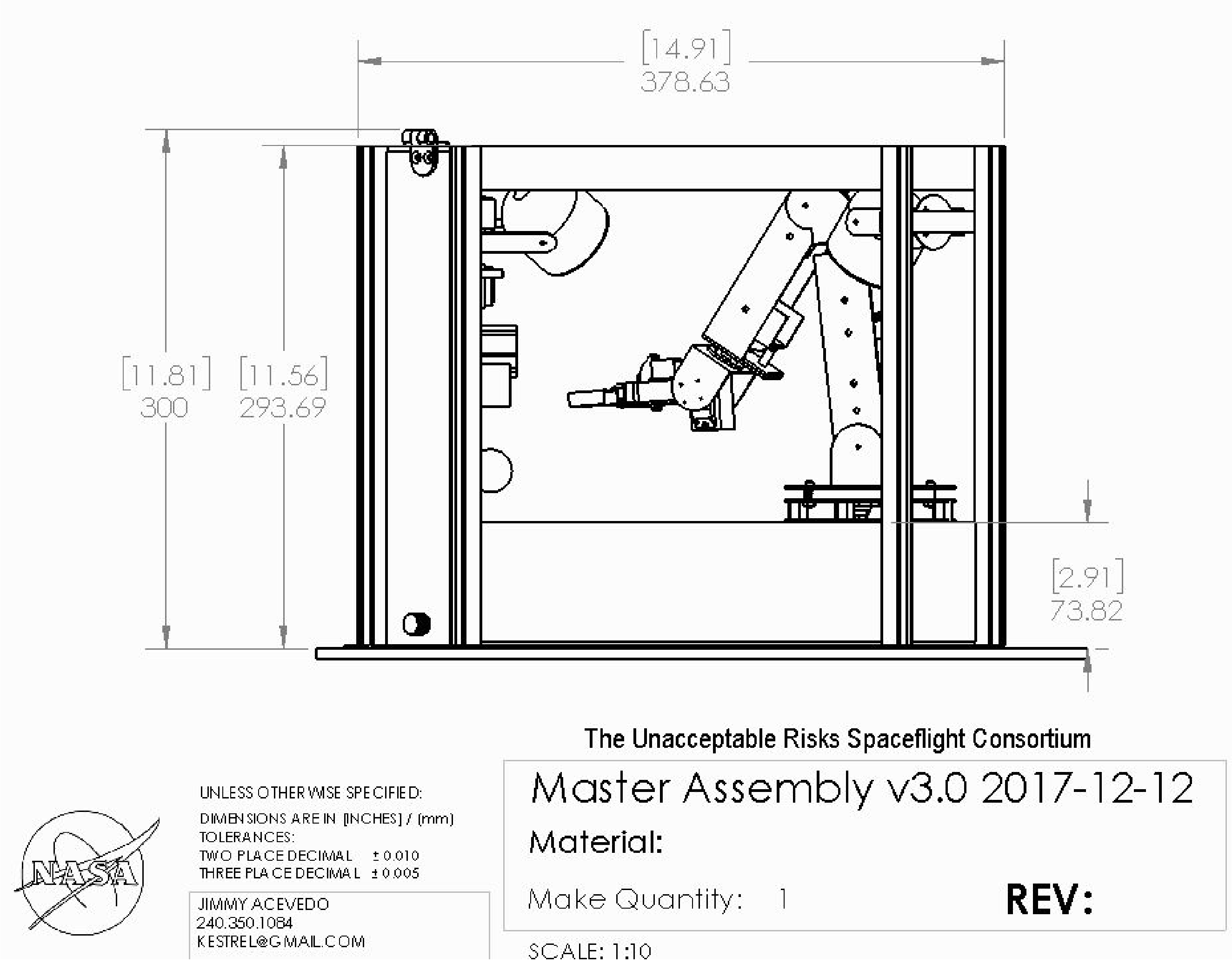
- **Construction narrative**
    - **Frame:** Our frame will be horizontal octagons of ⅛” 6061 aluminum angle, welded together, supported in the vertical by 8 pieces of ⅛” 6061 aluminum u-channel, also welded together. The frame’s primary purpose is to be an over engineered roll-cage to protect the arm and electronics in the event that the HASP rigging falls onto the payload during landing; its secondary purpose is to support our thermal insulation and our electronics.
        - The frame is bolted to the PVC plate and aluminum baseplate (from a sheet of 1/16” 6061) by 8 ¼” steel bolts.
        - The aluminum shelf in the middle of the payload which holds the robot arm will be made from 3/32” aluminum.
        - The electronics basement and Busy Box will be contained in 1/16” aluminum sheet, painted white, lined with multi-layer insulation (MLI.)
    - **Busy Box:** The Busy Box will consist of a single bulkhead of 1/16” aluminum into which are mounted all switches, knobs, etc. Electronics will be attached to the plate from behind, on insulating standoffs and/or 3D-printed boxes of ABS plastic. The entire Busy Box bulkhead will slide into two channels in the vertical mounting posts, which we will bolt to the frame. Two set-screws will hold it securely in place. We will be able to access Busy Box wiring and electronics by removing the lid plate and by opening the three hatches on the back of the Busy Box. This gives us a sturdy payload that requires relatively little bolting together, yet which is modular enough to replace or repair malfunctioning components.
    - **Electronics Basement:** The basement is defined by the bottom plate, the ‘middle shelf’ through which the arm is mounted, and the sides of the payload. We plan on routing and tagging most of our wiring while this shelf is removed, then adding the shelf and plugging the microcontrollers etc. in with mini-Molex connectors.
    - **Robot Arm:**
        - The body of the robot arm will be made from 1/16” 6061 aluminum. Most pieces will be machined individually, though some will require forming with a sheet metal brake.
        - We have designed our gripping ‘hand’ to simplify its kinematics; while most grippers extend forward as a part of the act of pinching shut, ours is simpler and only requires a single rack-and-pinion track/servo interface to drive it. (see image below)

- The bearing’s purpose is to distribute the stress on the servo output shaft. We copied our bearing dimensions and orientation from one of our ‘tutorial’ arms, a commercial-off-the-shelf toy robot arm, and it seemed to work well.



Only ‘side-to-side’ motion by gripping arm; no forward extension.

- **Dimensioned Drawings**



The Unacceptable Risks Spaceflight Consortium

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN [INCHES] / (mm)
TOLERANCES:
TWO PLACE DECIMAL ± 0.010
THREE PLACE DECIMAL ± 0.005

JIMMY ACEVEDO
240.350.1084
KESTREL@GMAIL.COM

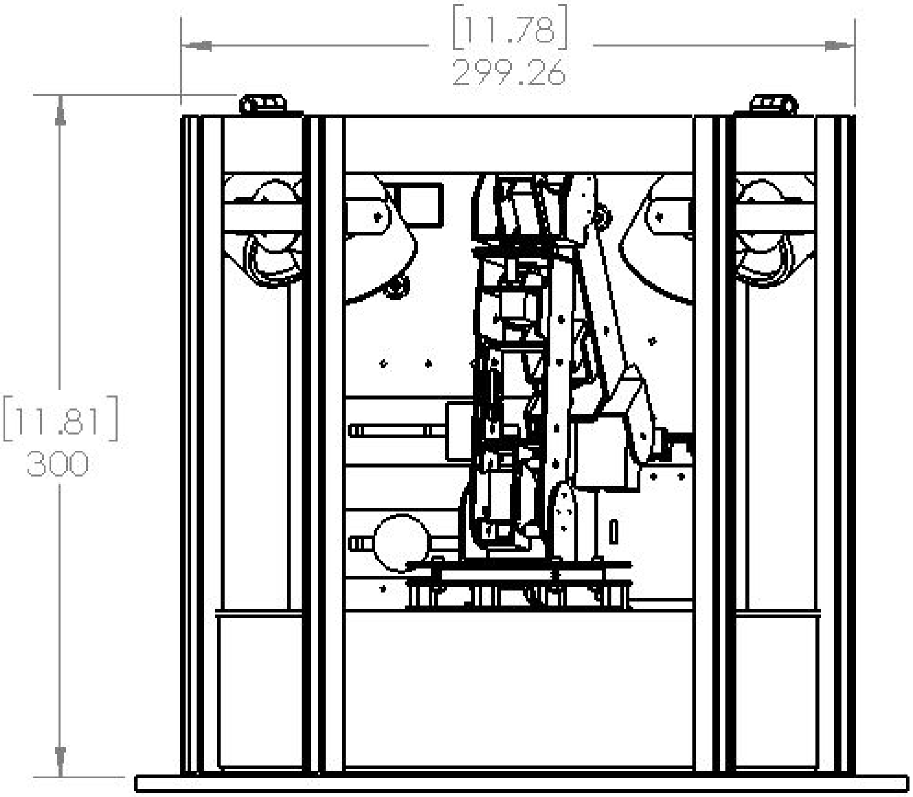
Master Assembly v3.0 2017-12-12

Material:

Make Quantity: 1

REV:

SCALE: 1:10



The Unacceptable Risks Spaceflight Consortium

UNLESS OTHERWISE SPECIFIED:

DIMENSIONS ARE IN [INCHES] / (mm)
TOLERANCES:
TWO PLACE DECIMAL ± 0.010
THREE PLACE DECIMAL ± 0.005

JIMMY ACEVEDO
240.350.1084
KESTREL@GMAIL.COM

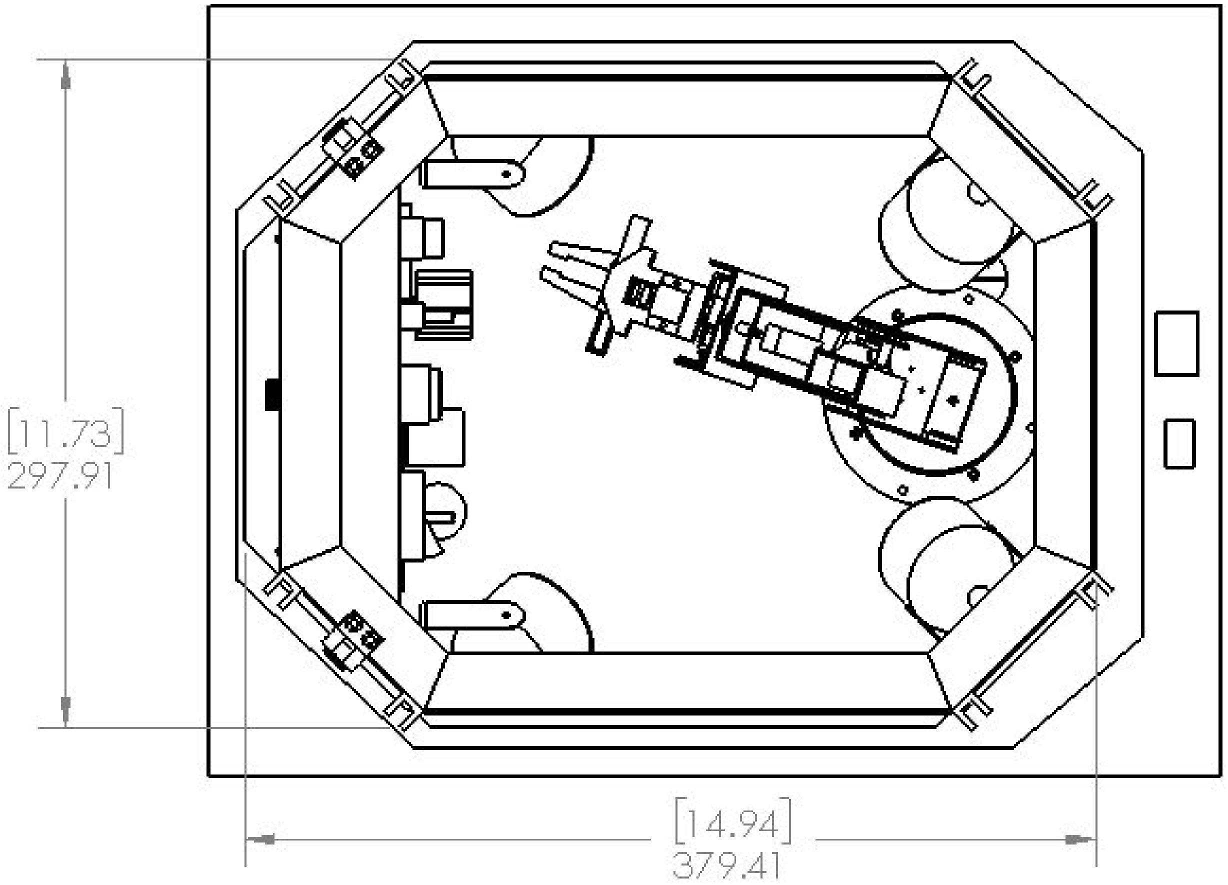
Master Assembly v3.0 2017-12-

Material:

Make Quantity: 1

REV:

SCALE: 1:10



The Unacceptable Risks Spaceflight Consortium

UNLESS OTHERWISE SPECIFIED:

DIMENSIONS ARE IN [INCHES] / (mm)
TOLERANCES:
TWO PLACE DECIMAL ±0.010
THREE PLACE DECIMAL ±0.005

JIMMY ACEVEDO
240.350.1084
KESTREL@GMAIL.COM

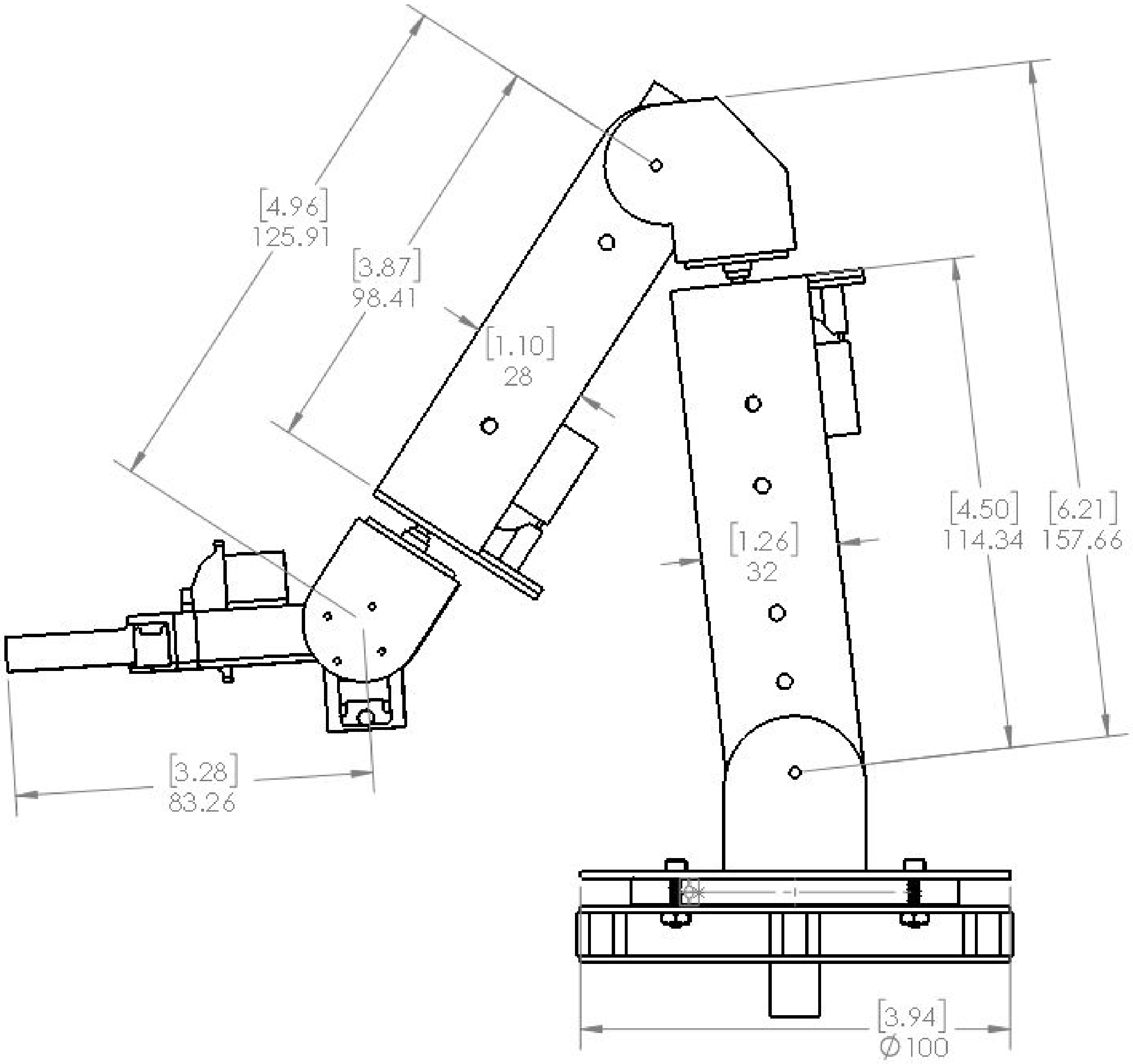
Master Assembly v3.0 2017-12-12

Material:

Make Quantity: 1

REV:

SCALE: 1:10



The Unacceptable Risks Spaceflight Consortium

UNLESS OTHERWISE SPECIFIED:

DIMENSIONS ARE IN [INCHES] / (mm)
TOLERANCES:
TWO PLACE DECIMAL ±0.010
THREE PLACE DECIMAL ±0.005

JIMMY ACEVEDO
240.350.1084
KESTREL@GMAIL.COM

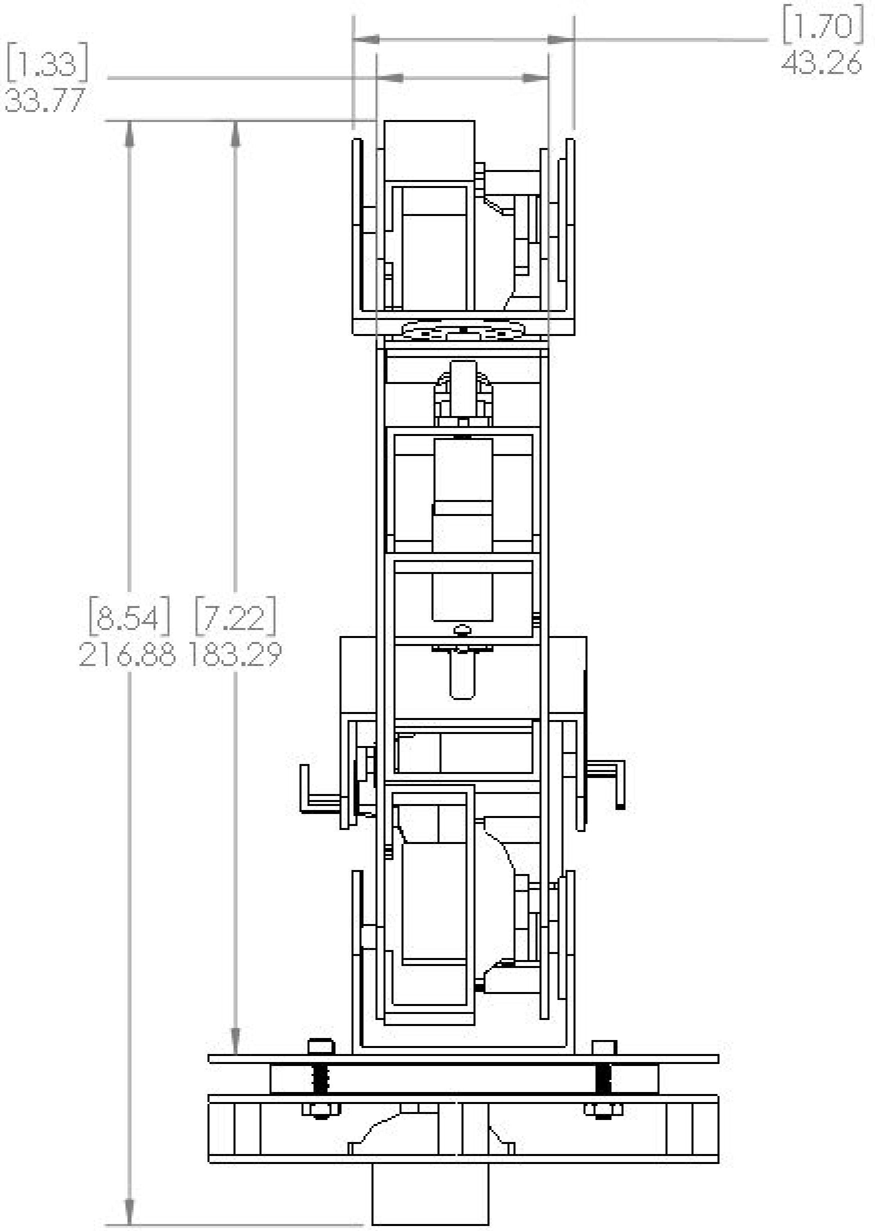
Robotic Arm Assembly 11-23-17

Material:

Make Quantity: 1

REV:

SCALE: 1:5

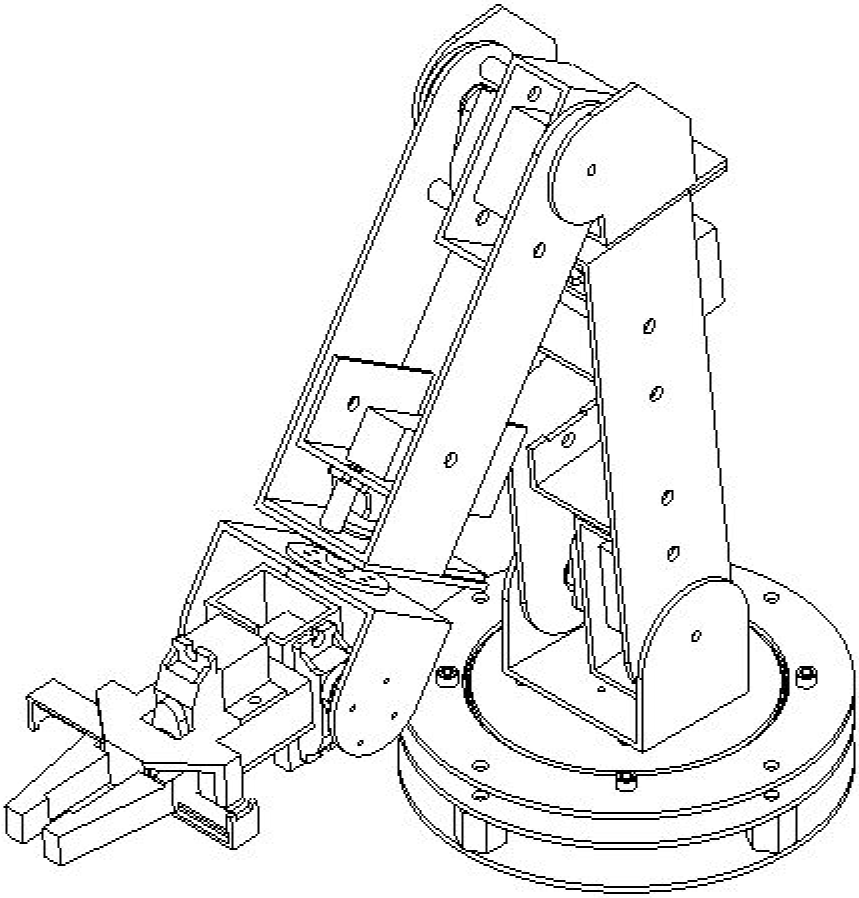


The Unacceptable Risks Spaceflight Consortium

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN [INCHES] / (mm)
TOLERANCES:
TWO PLACE DECIMAL ±0.010
THREE PLACE DECIMAL ±0.005

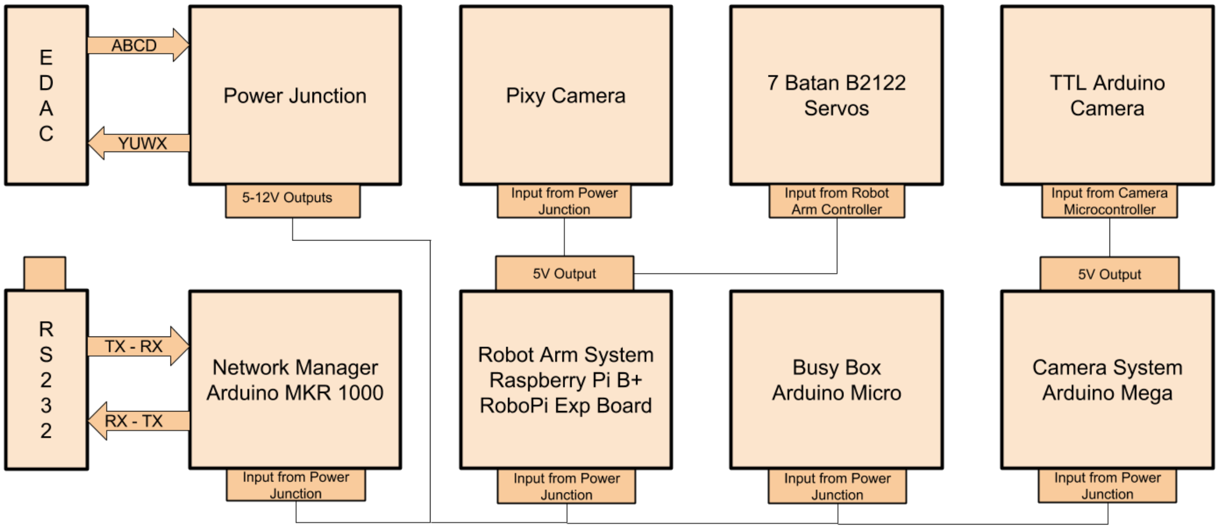
Robotic Arm Assembly 11-23-

Material:



- **Power Circuit Diagram**

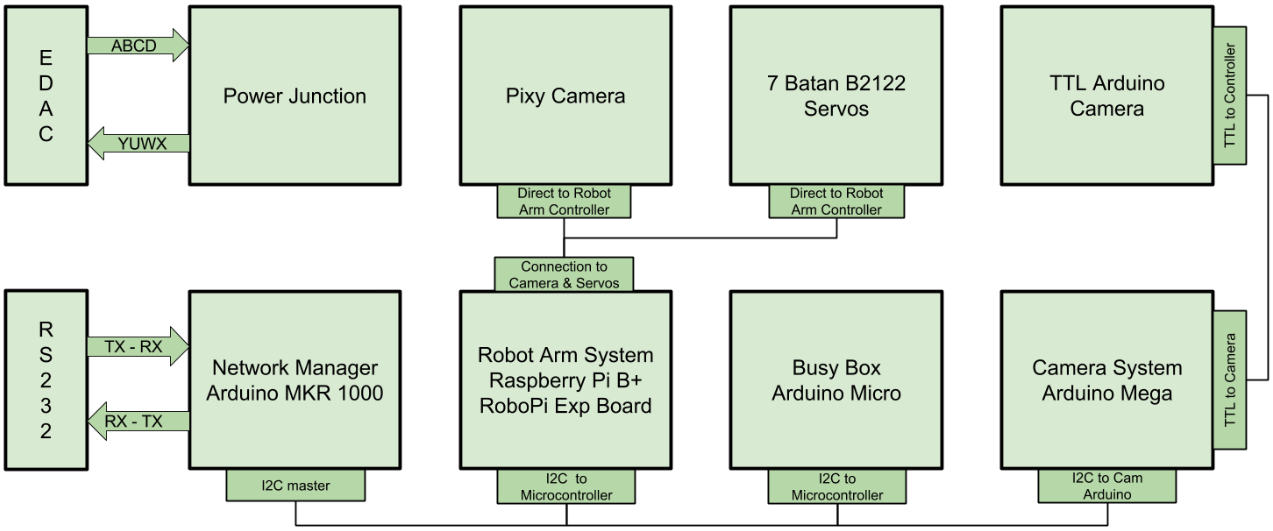
Power Block Diagram



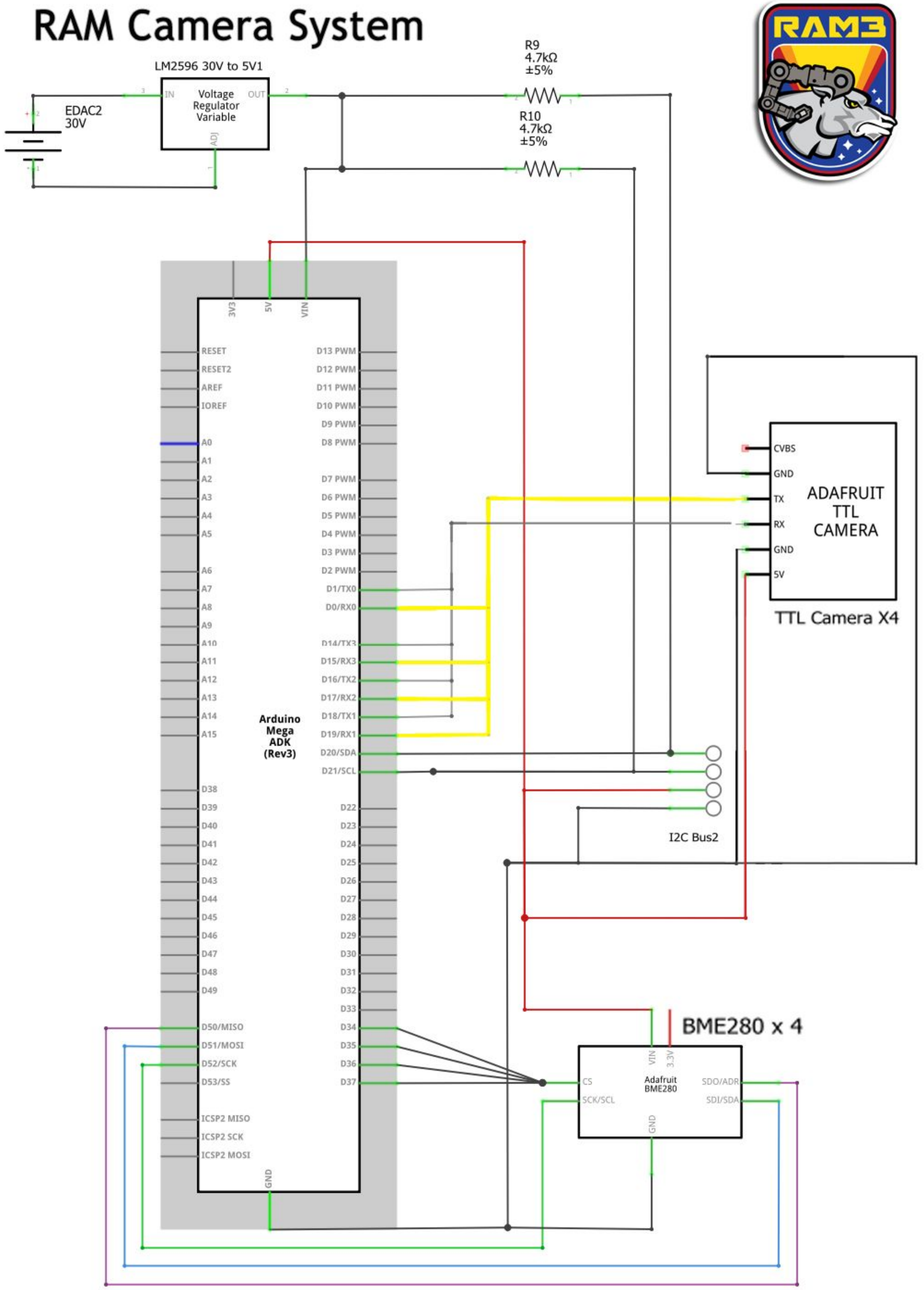
E D A C
ABCD
YUWX
Power Junction
5-12V Outputs
Pixy Camera
Input from Power Junction
7 Batan B2122 Servos
Input from Robot Arm Controller
TTL Arduino Camera
Input from Camera Microcontroller
R S 2 3 2
TX - RX
RX - TX
Network Manager Arduino MKR 1000
Input from Power Junction
5V Output
Robot Arm System Raspberry Pi B+ RoboPi Exp Board
Input from Power Junction
Busy Box Arduino Micro
Input from Power Junction
5V Output
Camera System Arduino Mega
Input from Power Junction

- **Network Block Diagram**

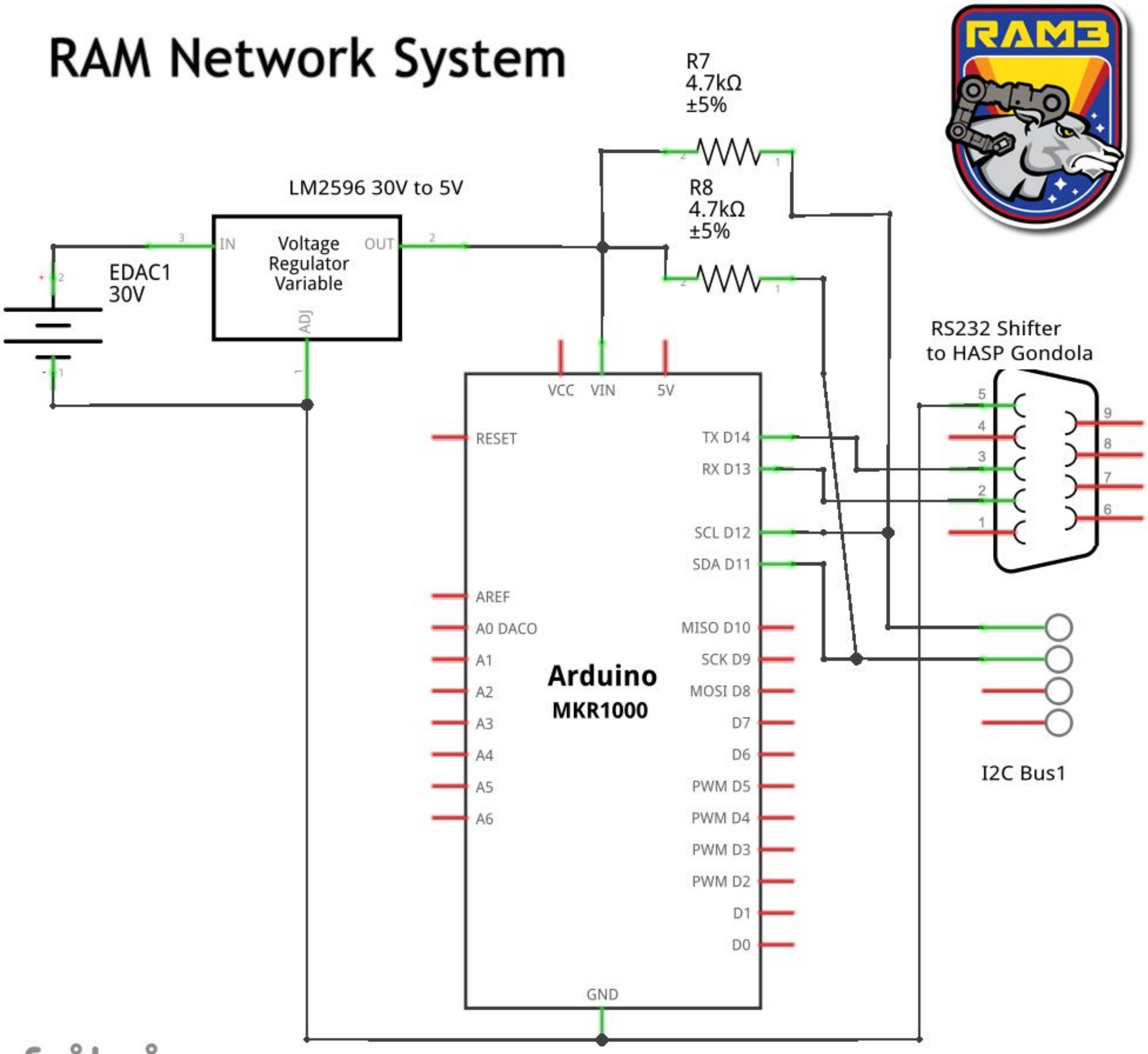
Network Block Diagram



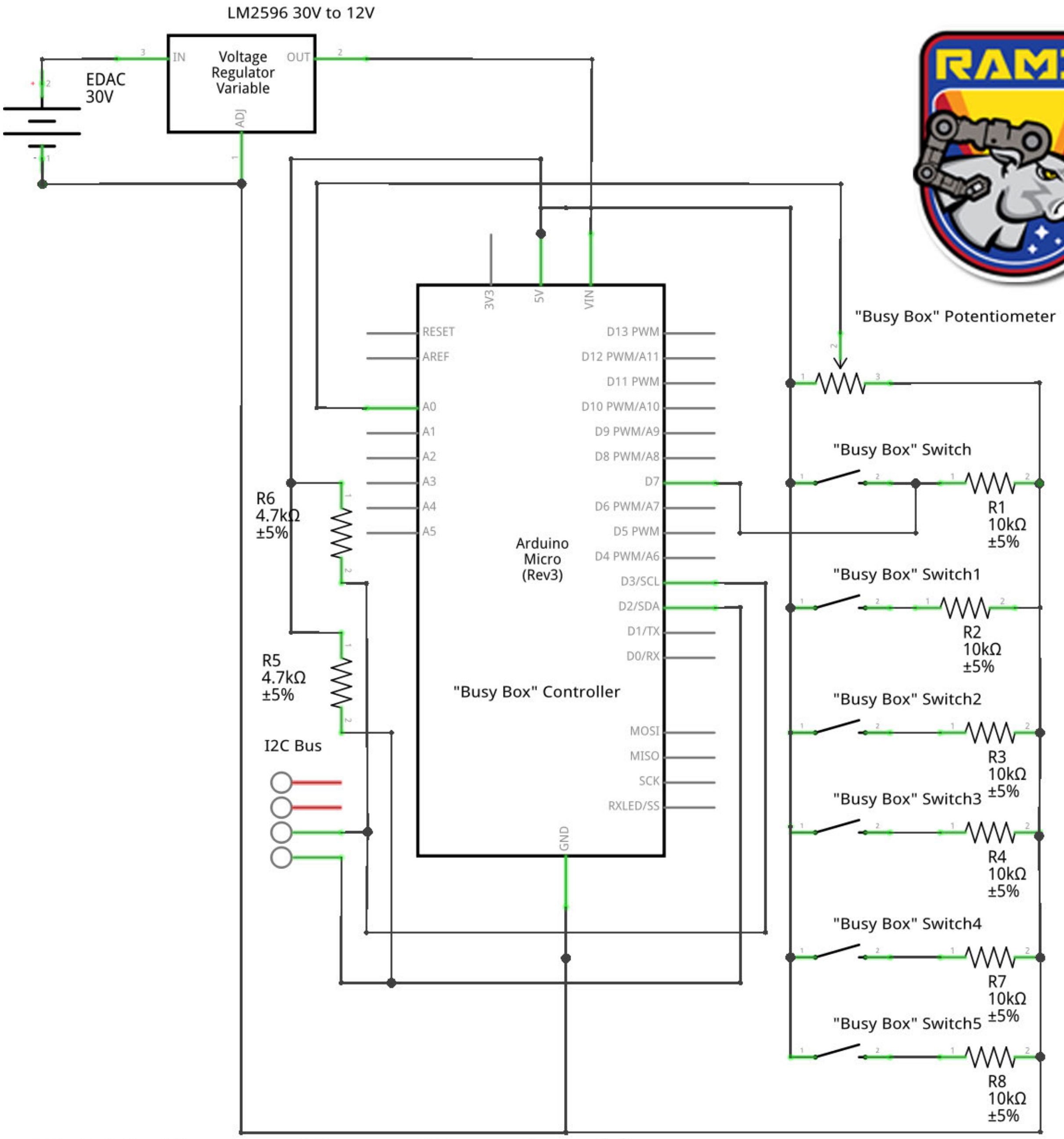
Wiring Diagram (*Please see [this folder]() for full-resolution images of all diagrams***)**



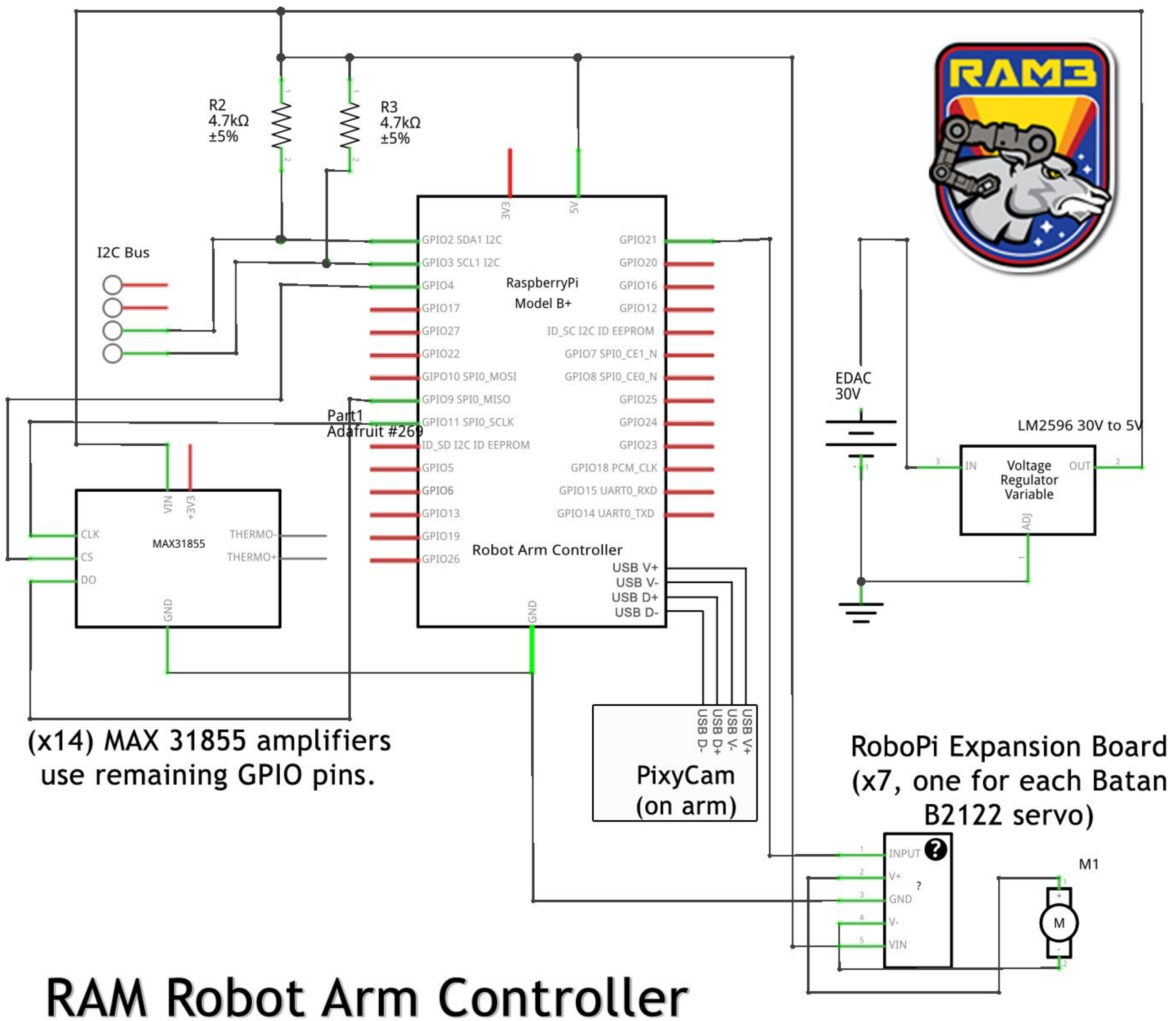
RAM Network System



fritzing



RAM Busy Box Controller

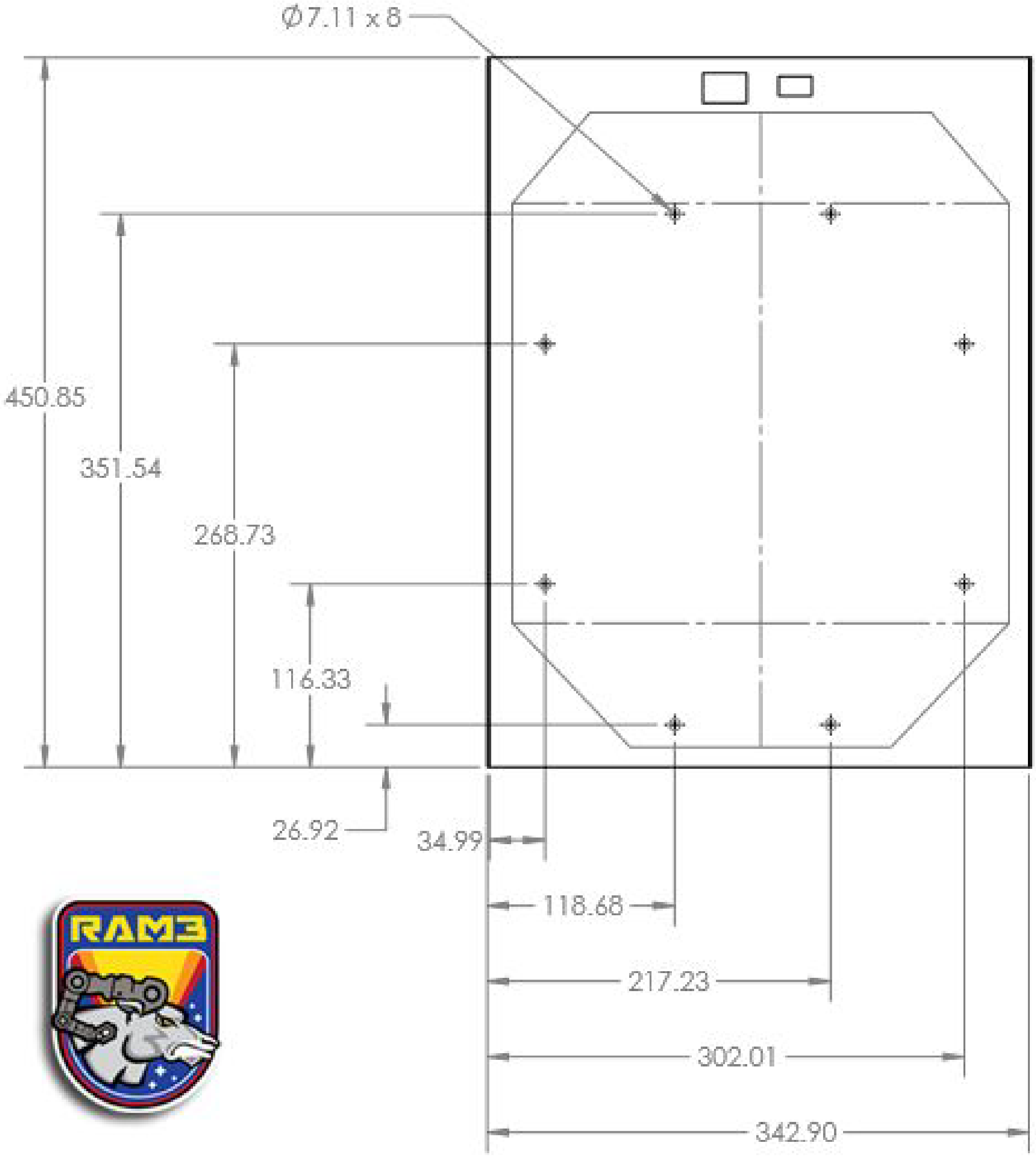


(x14) MAX 31855 amplifiers use remaining GPIO pins.

RoboPi Expansion Board (x7, one for each Batan B2122 servo)

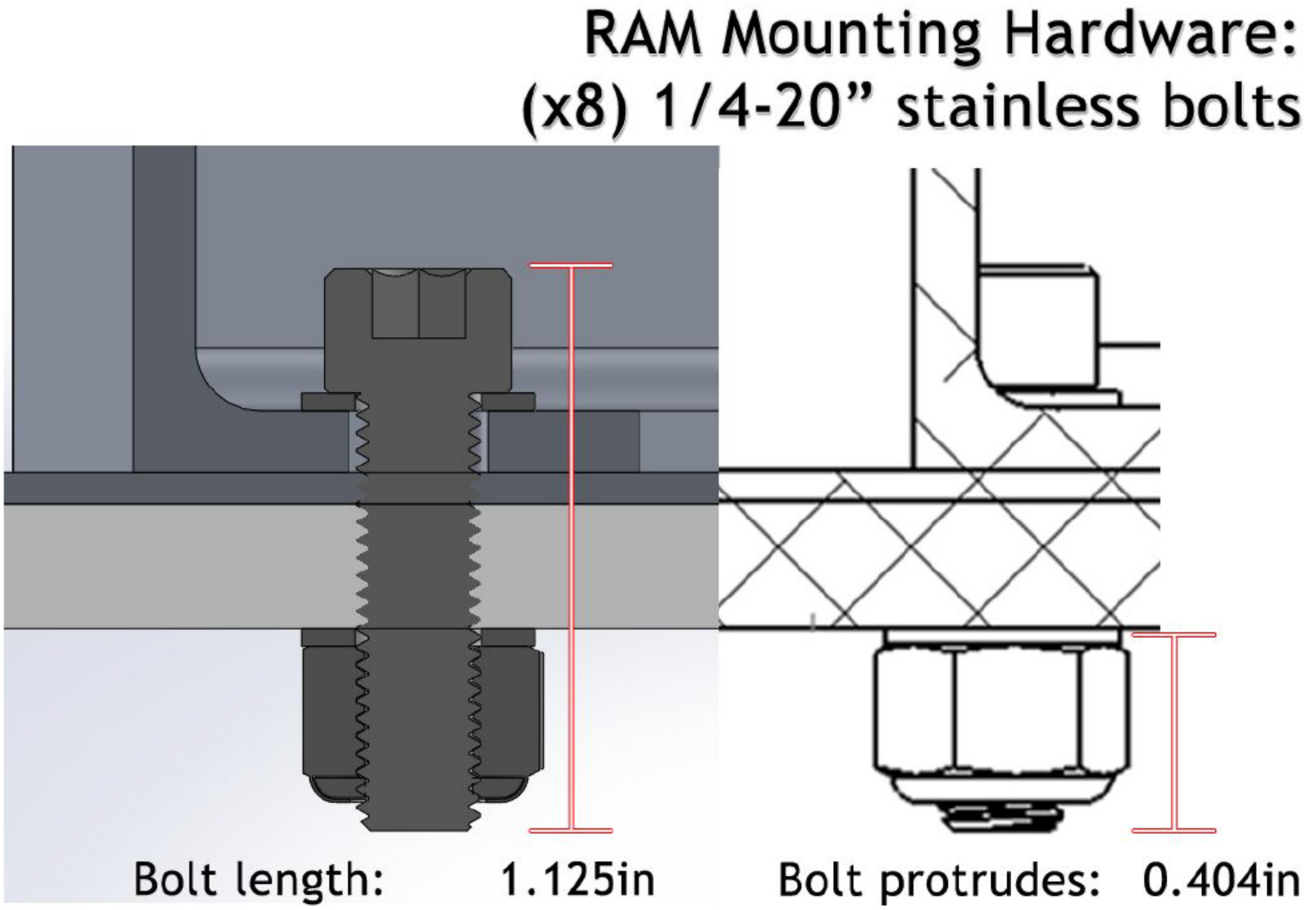
RAM Robot Arm Controller

- **Anticipated modifications to payload mounting plate**



RAM Mounting Plate Modifications
(all dimensions in mm)

- **Sketches of mounting structure**



- **Preferred payload orientation and location on HASP**
  - See diagram above; we prefer payload slot 12 or 11. Our payload's fore/aft will align with the HASP gondola's fore/aft (and will be clearly labeled as such.)

Works Cited

Dunbar, Brian. "In-space Robotic Manufacturing and Assembly (IRMA)." *NASA STMD: Tech Demo Missions.* 13 Sep 2017. Web. https://www.nasa.gov/mission_pages/ tdm/irma/index.html.

Dunbar, Brian. "Robotic Arm." *Mars Phoenix.* 22 May 2008. Web. https://www.nasa.gov/multimedia/imagegallery/image_feature_1089.html.

Dunbar, Brian. "The Robotic Servicing Arm." *NASA Satellite Servicing Projects Division.* Web. https://sspd.gsfc.nasa.gov/robotic_servicing_arm.html.

Greicus, Tony. "Testing for Instrument Deployment by InSight's Robotic Arm." *NASA InSight Mars Lander.* 4 Mar 2015. Web. https://www.nasa.gov/jpl/insight/pia19144.

"High Altitude Student Platform." *Louisiana State University Space Sciences Group.* Web.

http://laspace.lsu.edu/hasp/.

Pultarova, Tereza. "Company Aims to Launch Satellite-Servicing Spacecraft in 2020." *Space.org.* 15 Jul 2017. Web. https://www.space.com/37205-satellite -service-repair-spacecraft- 2020.html.

"UCS Satellite Database." *Union of Concerned Scientists.* 7 Nov 2017. Web. http://www.ucsusa.org/nuclear-weapons/space-weapons/satellite-database#/Wg5JMEqnEdU.

University of Bridgeport. "High Altitude Robot Servo Motor Test." *HASP Student Payload Application for 2016.* (2016): Web. http://laspace.lsu.edu/hasp/groups/2016/applications/ Payload_11/ UB_HASP2016_Flight_Application.pdf.

University of Bridgeport. "University of Bridgeport HASP Servo-Motor Testbed." Web.

http://laspace.lsu.edu/hasp/groups/2016/science_report/Payload_11/UB_HASP_2016_Science_Report.pdf.