DESIGN AND DEVELOPMENT OF A MODULAR AND REUSABLE CUBESAT BUS

TRAVIS KIMBLE IMKEN

DECEMBER 2, 2011

ENGINEERING HONORS UNDERGRADUATE THESIS

THE DEPARTMENT OF AEROSPACE ENGINEERING AND ENGINEERING MECHANICS
THE UNIVERSITY OF TEXAS AT AUSTIN
AUSTIN, TX 78705

_____

DR. GLENN LIGHTSEY, PH.D.
DEPARTMENT OF AEROSPACE ENGINEERING AND ENGINEERING MECHANICS
FACULTY ADVISOR

_____

DR. TODD HUMPHREYS, PH.D.
DEPARTMENT OF AEROSPACE ENGINEERING AND ENGINEERING MECHANICS
SECOND READER

TABLE OF CONTENTS

ABSTRACT

In the past decade, CubeSats have started a new chapter in space exploration and research. CubeSats present university students with a unique opportunity to gain hands-on experience in designing and flying satellites. Advancements in technology have allowed more powerful electronics to fit within the CubeSat constraints and launch providers are willing to launch these spacecraft as secondary payloads. With new advancements in technology and flight heritage, CubeSats are becoming a valuable research tool for NASA, the NSF, and many other institutions.

Beyond the standard size, mass, and testing requirements, there are few similarities between CubeSats developed at different institutions; the structures are often uniquely designed and the software is written to be functional for the current mission. Now, as small satellites become more accepted and useful to the scientific community, there is an increased need to create a reusable and modular satellite support bus for a third party payload. Science customers can develop their own instruments, integrate into a working satellite through simple software and hardware interfaces, and gain access to space.

This thesis is focused on the research conducted for the University of Texas at Austin Satellite Design Laboratory's current ARMADILLO and Bevo-2 CubeSat projects. Though the missions for these two satellites are very different, both satellites benefit from the same research around a common hardware and software bus design. Both the structure and software research is inspired by modularity. Hardware is grouped into modules and included within a 'wall shell'. All of the modules, including a designated payload module, are easily connected by 'section connectors' to create a functioning satellite. The flight software 'Hookem' also creates modules

of code for each subsystem. This ‘Interface Object Software’ gives the Command and Data Handling subsystem full access to the subsystem software functions the satellite can execute in its mission. Both the structure and software modules can be changed out as hardware or mission objectives change for future projects or to support a payload.

The Satellite Design Laboratory has a rich history in designing, building, and operating small satellites. Because the lifespan of small satellite projects are only a few years, the Lab has recognized the challenges of transferring knowledge and work between missions. The target of this research is to create a modular and reusable bus design that reduces the non-recurring engineering time and costs of starting new satellite projects. With careful future planning, the structure and software for a small satellite bus can be designed and developed to benefit future missions. The ultimate goal is to create a capable satellite to support nearly any customer’s needs.

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 UNDERGRADUATE MOTIVATION

At the end of my freshman year in aerospace engineering at The University of Texas at Austin I was looking for an extracurricular activity to supplement my classroom education. In March of 2008 I received an email from a student working on a project in the Satellite Design Laboratory (SDL) searching for undergraduate help on one of their projects. I had high school experience in computer aided drafting work so I began working on modeling and design issues for the PARADIGM satellite. This was the start of a still-continuing adventure in the SDL and deep fascination in our projects. I have had the chance to design spacecraft, assemble a satellite and watch it launch from the Space Shuttle, learn how to use Linux, and construct flight software. In all, this thesis focuses on a very specific subset of all of the research and experience I have gained in the SDL over the past four years. I truly hope that I can pass on my knowledge and expertise to the next generation of student satellite engineers that are the future of our lab.

1.2 RESEARCH INSPIRATION

The inspiration for this thesis comes from two topics in the realm of satellite design: structure and software. These topics are polar opposites in many ways, but the driving concepts of modularity and reusability can be applied to both. The structure subsystem of a spacecraft is an excellent way for new students, many with no design expertise, to get involved with satellite design. Due to the inherent nature of CAD work, team members are able to visually interpret how all of the subsystems integrate into the spacecraft system as a whole. However, even within the structures subsystem, new engineers can only master principles such as machinability, ease-

of-assembly, and space-tolerant design through experience. Flight software design and systems-level coding is not emphasized in the engineering degrees at UT Austin so students seem reluctant to get involved with programming. The learning curve is steep and intimidating.

As the SDL began work on the ARMADILLO and Bevo-2 missions, there were clear limitations in the transfer of structures and software knowledge from previous projects; much of the work would have to be redone. From this experience, the complexities of the software and structure dictated a need for standardized interfaces to direct projects as their missions evolved over time. The concepts and ideas presented in this thesis are not meant to confine creativity or stifle inspiration. Instead, they promote a solid foundation that future projects can use as a well-defined starting point for true engineering.

1.3 RESEARCH TARGET

The structure and software research ideas presented within this thesis are targeted at the bus design of a satellite. In the small satellite context, the ‘bus’ refers to the hardware, electronics, and software that provide support and resources to a payload. The payload can range from scientific instruments to technology demonstrations. Third party customers, who may not have experience in building satellites, should be able to integrate payloads into a bus design that meets their requirements. It is feasible to imagine someday that the UT SDL will provide a fully functioning satellite bus to scientific and research institutions. In the end, this research is inspired by the notion of creating a commercial product that benefits the small satellite community.

1.4 THESIS ORGANIZATION

This thesis is organized into four further chapters. Chapter 2 focuses on the past and present missions in the Satellite Design Laboratory. The scope and goals of these missions will

help establish the need and benefit of the implementation of a modular and reusable design into satellites. From here, Chapter 3 describes the limitations of small satellite structures, several iterative design concepts, the finalized structure development, and creation of standardized interfaces for other subsystems and partners. Chapter 4 looks into past satellite software design from heritage SDL projects, the implementation of the mode manager, creation and integration of the ‘interface object software’ philosophy, and satellite operations software planning. Chapters 3 and 4 also explore future research ideas. Finally, Chapter 5 provides a brief conclusion on the research and the impact it could have on future spaceflight.

2 PROJECTS IN THE SATELLITE DESIGN LABORATORY

The Aerospace Engineering and Engineering Mechanics Department at The University of Texas at Austin has established a unique lab for students interested in spacecraft. The Satellite Design Laboratory (SDL) allows both undergraduate and graduate students the chance to further their education outside of the classroom and gain hands-on experience building satellites.

2.1 PAST PROJECTS

The SDL has a rich experience in developing and operating satellites. Three student-built satellites have been launched into space: the FASTRAC satellites Sara Lily and Emma, and the Bevo-1 satellite from the LONESTAR program.

2.1.1 FASTRAC Project

The FASTRAC (Formation Autonomy Spacecraft with Thrust, Relnav, Attitude and Crosslink) project began in 2003 and was selected as the winner of the third iteration of the University Nanosat Program (UNP) competition in 2005. The UNP competition is sponsored by the Air Force Research Laboratories as a chance for universities to gain funded, hands-on experience in designing satellites. The UNP competition selects several universities to compete through a series of design reviews for a launch opportunity aboard an Air Force provided launch. The FASTRAC project was composed of two nearly-identical nanosat-sized (25 kg each) satellites testing real-time GPS relative navigation and communication between the two satellites. [1] FASTRAC was launched in November of 2010 and is still continuing its operationally successful mission.

2.1.2 LONESTAR: Bevo-1

Bevo-1 was the UT satellite developed for the first mission of four prescribed by the NASA LONESTAR (Low Earth Orbiting Navigation Experiment for Spacecraft Testing Autonomous Rendezvous and docking) program. The program is a collaborative effort between UT Austin's SDL and Texas A&M University's AggieSat Lab, with the ultimate goal of achieving autonomous rendezvous and docking in-orbit on the fourth flight. The first three missions of the program are targeted for technology demonstration and testing operations that will be necessary for the final mission. For the first mission, both Bevo-1 and AggieSat-2 were five-inch cubes placed into the SSPL (Space Shuttle Picosat Launcher) as seen in Figure 1.

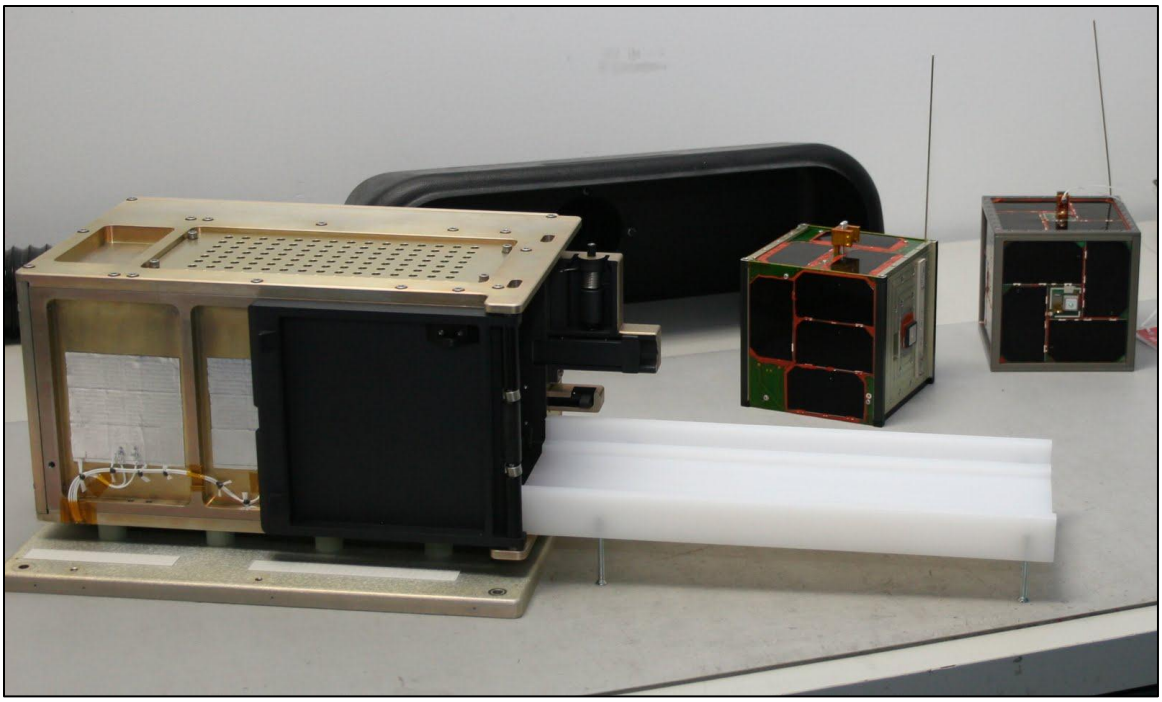


Image Credit: Henri Kjellberg

Figure 1. The SSPL with Bevo-1 (left) and AggieSat-2 (right)

The Bevo-1 satellite was developed independently of the Texas A&M satellite. The mission was relatively simple; the main payload was the NASA-designed DRAGON GPS

receiver and the main objective was to collect and downlink two orbits of GPS data. The satellites launched aboard the Space Shuttle Endeavour in July of 2009 but failed to separate completely from their launcher configuration upon deployment. Bevo-1 never powered on and no contact was ever established with the satellite.

2.2 CURRENT PROJECTS

The SDL is currently working on two satellite missions. The Bevo-2 satellite is the continuation of the LONESTAR program and the ARMADILLO satellite is the project entry for UT in the UNP-7 competition. Both spacecraft are designed around the 3-Unit CubeSat specifications, developed by California Polytechnic State University as a standard for small satellites. A 3-Unit (3U) CubeSat is 100 by 100 by 340.5 mm with a maximum volume of 4 kg and is launched in a deployer, such as the P-POD (Poly Picosat Orbital Deployer). Generally, CubeSats ride into space as secondary payloads and operate in the same orbit as the primary payload. [2]

2.2.1 LONESTAR: Bevo-2

Bevo-2 is the second mission in the LONESTAR program and has advanced mission objectives. The spacecraft will have a 6 degree-of-freedom active attitude determination and control (ADC) system, an optical system that can be used both as a camera and a star tracker, and a cold gas thruster to achieve small translational maneuvers. All of these systems will be necessary for the ultimate rendezvous and docking objective. Bevo-2 will be deployed from Texas A&M's AggieSat-4, a 50 kg nanosatellite that will be launched from the ISS. The goals of the Bevo-2 mission are to crosslink information with the AggieSat-4 after separation, conduct tests to understand and characterize the performance of the ADC system, and perform a state

rendezvous in space using the thruster. Figure 2 below is the concept of operations for the Bevo-2 mission.

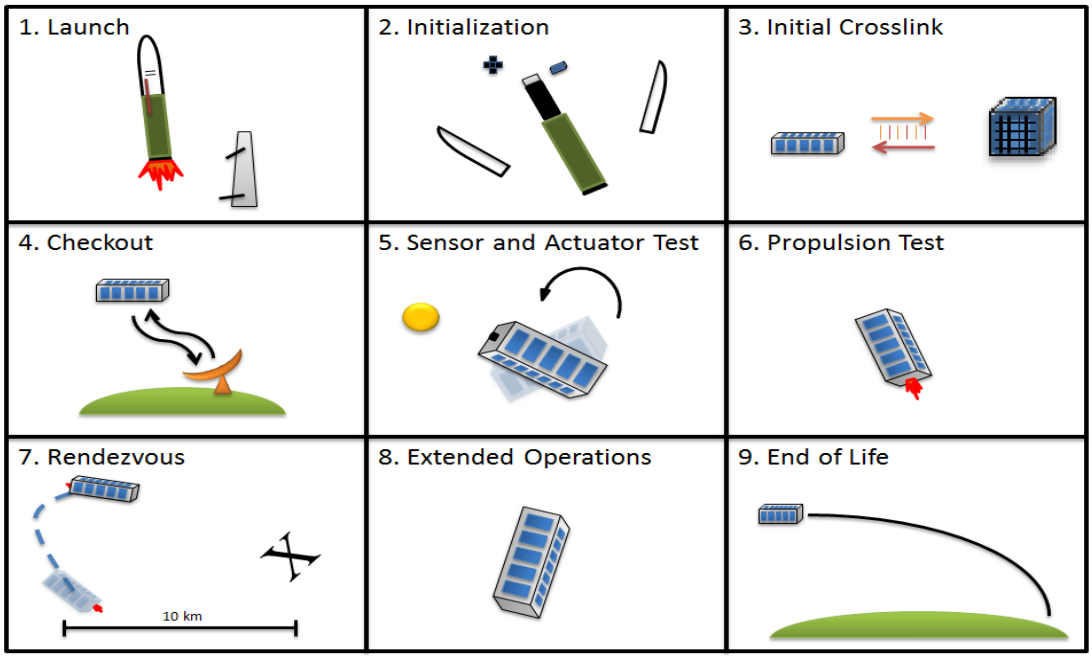


Image Credit: UT SDL

Figure 2. Concept of Operations for the Bevo-2 Mission

Bevo-2 is currently set to be delivered in August 2012 for integration with AggieSat-4. The pair of satellites will be launched on an ISS resupply mission in early 2013.

2.2.2 ARMADILLO

The ARMADILLO (Attitude Related Maneuvers and Debris Instrumentation in Low (L) Orbit) has a nearly identical bus as the Bevo-2 satellite but will complete a very different mission. While the spacecraft features the same attitude suite of sensors and actuators, camera, and thruster, it has a more sophisticated GPS receiver and a Piezo-electric Dust Detector (PDD). The Foton GPS is a dual-frequency receiver and will enable the spacecraft to conduct GPS radio occultation measurements to characterize the Earth's atmosphere. The PDD is a scientific

instrument delivered by Baylor University and will collect information on sub-millimeter sized pieces of orbital debris in the low Earth orbit environment [3]. The ARMADILLO mission is scheduled to conduct the UNP-7 CDR in March 2012. The ultimate flight selection will happen in January 2013.

2.3 NEW PROJECT CHALLENGES

The transition from the FASTRAC and Bevo-1 satellites to the ARMADILLO and Bevo-2 projects was technologically difficult. FASTRAC was an order of magnitude larger than the newly chosen 3U CubeSat size and there was poor documentation on design choices for the Bevo-1 mission. With the constraints of the UNP competition, the FASTRAC program helped establish good systems engineering and documentation practices that can be applied to future missions. However, many years of technical engineering were invested into the project and relatively little, apart from the lessons learned on a systems and operations level, was transferrable to future projects. With the beginning of two new projects, there was a critical need to not only address the objectives of the current missions but also prepare for the transition to new future projects.

While ARMADILLO and Bevo-2 are unique missions, they share many common subsystems and benefit from the same concurrent design work. This leads to one of the principle challenges behind developing a new program: how can concepts and products from one project be transferred successfully to future projects? For the work in the SDL, this question must be targeted specifically at small satellite design and students must begin thinking about the future LONESTAR and UNP satellites the lab will develop. Ultimately, every subsystem on a satellite benefits from good planning and research into the future. However, from previous experiences in

the SDL, flight software and structure are two subsystems that are constantly reengineered from scratch with little help from past work. Bevo-1 provided the design inspiration for much of the ARMADILLO and Bevo-2 missions. This inspired the research into the design and development of a modular and reusable CubeSat bus that is aimed at achieving both missions' objectives while establishing a foundation for the SDL's future.

3 STRUCTURE

The structure is one of the most important subsystems on a satellite. All devices and components within the satellite interact with the structure in some way, but there are few technological innovations that push the state-of-the-art forward. There are only a handful of commercial structures on the market targeted at universities and scientific institutions with less design expertise. As CubeSats become more prominent and capable, the need for a customizable yet standard structure is a largely uninvestigated and unsolved problem. This challenge is clear in the SDL with two satellite missions that are operationally unique but need to benefit from one-time engineering of the satellite structure. Bevo-2 and ARMADILLO are the first 3U CubeSats to ever be engineered by the SDL and demonstrate the first iteration of a reusable and modular structure design applicable to multiple missions.

3.1 TRANSITIONAL ISSUES

The Satellite Design Laboratory (SDL) is developing a strong expertise in the field of small satellites. The first small satellite (under 10 kg) developed by the SDL was the Bevo-1 satellite, a small five-inch cube that was designed to fit within the Space Shuttle Picosat Launcher (SSPL) envelope. However, with the retirement of the Space Shuttle in 2011, the future use of the SSPL on other launch vehicles was questionable. On the transition between Bevo-1 and -2 there was a lengthy discussion on the form factor for the new satellite. The ultimate choice was to continue work on small satellites and conform to the 3U CubeSat standard. The Bevo-2 satellite (and eventually the ARMADILLO satellite) has a smaller cross-sectional area (at 100 by 100 mm) but longer length (at 340.5 mm). The CubeSat standard offers nearly 75% more volume than the SSPL form factor. There is also a large community dedicated

to developing hardware and electronics that can fit within such a small volume. Additionally, the proliferation of the CubeSat standard has allowed these small satellites to have significantly easier access to space; launch providers are increasingly willing to launch CubeSats in enclosed launchers (such as the P-POD) as secondary payloads. [4] These attributes make the 3U CubeSat standard appealing for the SDL's future missions.

3.1.1 Commercially Available Products

There are many commercially available 3U structures. Pumpkin Inc. has developed a "Cube Sat Kit" that allows universities to essentially buy all of the components of a satellite, assemble and integrate them, and then work to manifest the satellites for launch. Pumpkin offers a very simple 3U design seen below in Figure 3. [5] The Pumpkin structure is made from a folded aluminum sheet that has stamped holes for Pumpkin's circuit boards and devices; it would be difficult to customize the structure for different electronics boards or mounting components as needed by the SDL's missions.

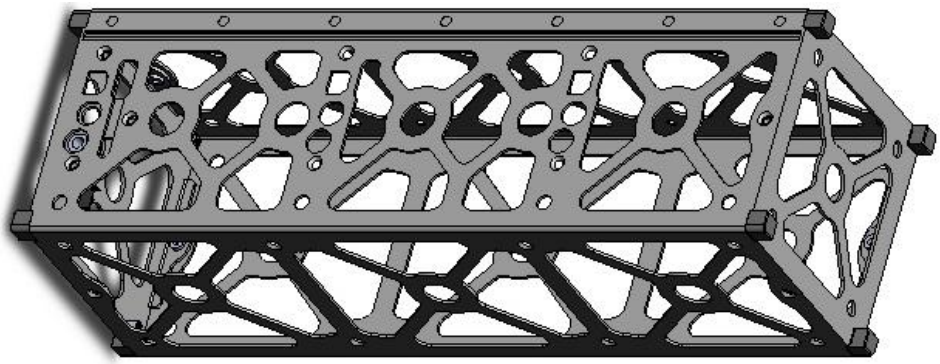


Image Credit: Pumpkin, Inc.

Figure 3. Pumpkin Inc. 3U CubeSat Chassis

ISIS (Innovative Solutions In Space), a Netherlands-based company, has introduced their own concept of a 3U structure seen in Figure 4 below. Their structure design is very minimalistic

and has no wall mounting capability except for mounting (ISIS's) solar panels. The ISIS structure is targeted at being modular. There are three distinct sections that combine to make a 3U satellite. However, similar to Pumpkin's design, the structure is targeted at the user buying their components from the same supplier to assemble the satellite as a 'kit'. [5]

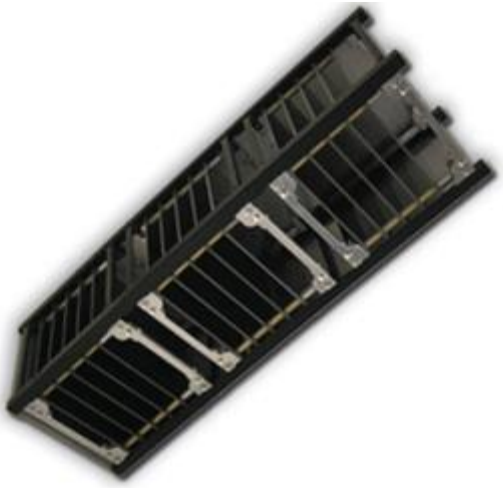


Image Credit: cubesatshop.com

Figure 4. ISIS 3U CubeSat Structure

3.1.2 A Unique Structure

The Pumpkin, ISIS, and other structures were not viable options for the Bevo-2 and ARMADILLO missions. Both of the SDL's missions would need the capability to arrange and mount hardware as needed without the additional requirement of conforming to a third party's structure requirements. The decision was made to create a structure that would conform to CalPoly's P-POD size requirements but could be customized to fit the need of the two missions. The decision for a custom structure was also supported by previous experience in designing, drafting, machining, and assembling structures from the FASTRAC and Bevo-1 missions.

The first significant challenge for the new 3U design was to fit the DRAGON GPS receiver. NASA requested that the SDL fly the DRAGON in future LONESTAR missions to continue to collect GPS orbit information. The DRAGON is a single printed circuit board (PCB) with an area of 4.25 by 4.25 inches (108 by 108 mm). The PCB fits easily into the 5-inch Bevo-1 sized satellite but theoretically could not fit into the smaller CubeSat standard. The CubeSat standard requires that the satellite have 100 by 100 mm dimensions on the 'rails'. Fortunately, the P-POD specifications allow for an additional 6.5 mm of space off each of the satellite's long faces between the rails for solar panels. This provided a total "width" of 113 mm for the DRAGON to fit in. With the DRAGON able to be mounted within the constraints of the 3U volume, the iteration of the structure for ARMADILLO and Bevo-2 could begin.

3.2 INITIAL CONCEPTS

Inspiration for the initial SDL structure concepts came from NASA. NASA developed two research 3U CubeSats that featured a completely customized structure. Genesat (2006) and then the follow-up Pharmasat (2009) featured a "payload" section as well as a "bus" section for support of the science experiments. Pictures of Genesat and Pharmasat can be seen in Figure 5.

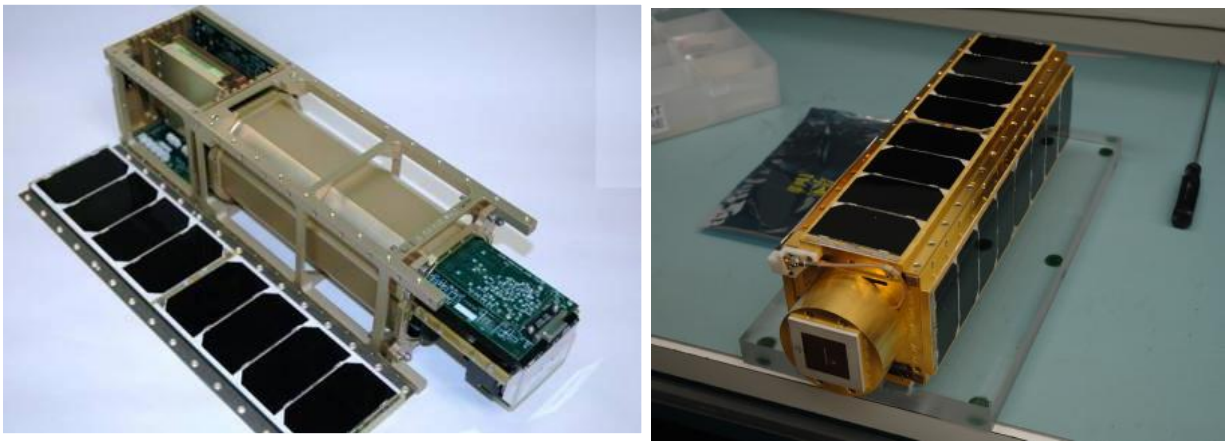


Image Credits: NASA

Figure 5. Genesat concept disassembled (left) and Pharmasat before flight (right)

The rigid frame allowed for mounting of the solar panels, the walls were hollowed out to save mass but could be left in place for mounting, and there were distinct modules. From the left picture in Figure 5, the electronics are all concentrated into the left module while the science payload consumes the right two-thirds of the satellite. These modules could be developed independently and then pieced together during integration. With the inspiration of Genesat and Pharmasat, the first iteration of the structure seen in Figure 6 was designed with the goals of being modular and adaptable for the Bevo-2 mission.

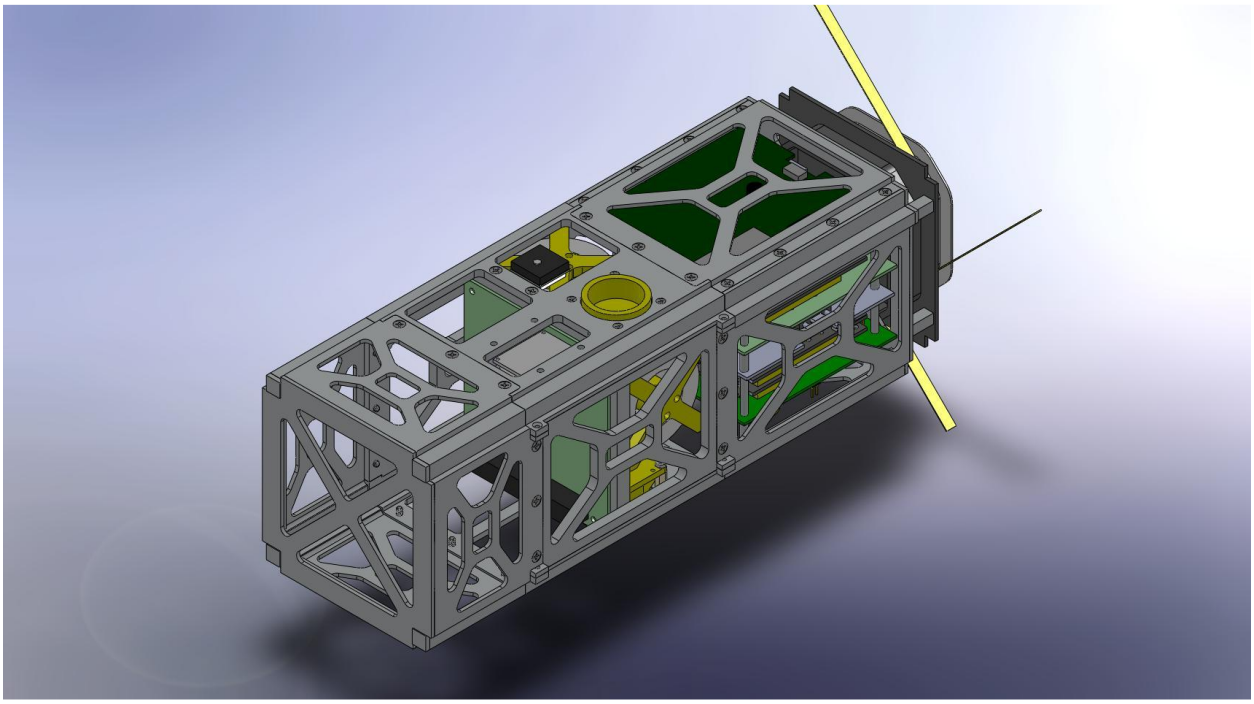


Figure 6. First iteration of Bevo-2 with a modular structure

The initial design of the structure was modular. The leftmost module was a dedicated payload section. The middle module was the Attitude Determination and Control (ADC) module that housed all of the actuators and sensors needed to give the satellite six dynamic degrees of

freedom (translation and rotation). Finally, the rightmost electronics module was for the computer, radios, power system, and the DRAGON PCB located on top (featuring the previously mentioned structure modifications to allow it to fit). The walls of the ADC and electronics module are machined separately and then fastened together to make a shell. The payload module is machined from a single piece of metal. All of the modules are held together with #4 standard fasteners. In general, the concept of modularity is addressed but the design suffers from several design limitations.

The structure in Figure 6 is a proof-of-concept rendering which demonstrates that modules are feasible in a 3U CubeSat. However, the above structure would be difficult to assemble. Because two of the modules require four walls to be fastened together, the structure is extremely massive to accommodate fastening hardware. The parts would be complex and costly to machine and would need to be customized for each satellite. Moreover, the interfaces between the modules are unique. While they work to assemble this concept satellite, there is no carryover between the current missions and future satellites the SDL would like to fly. The next iteration of the structure design would need to maintain the goals of modularity proven in the first iteration but focus on reusability. This lead to the focus to create an interface between modules that was simple and standardized.

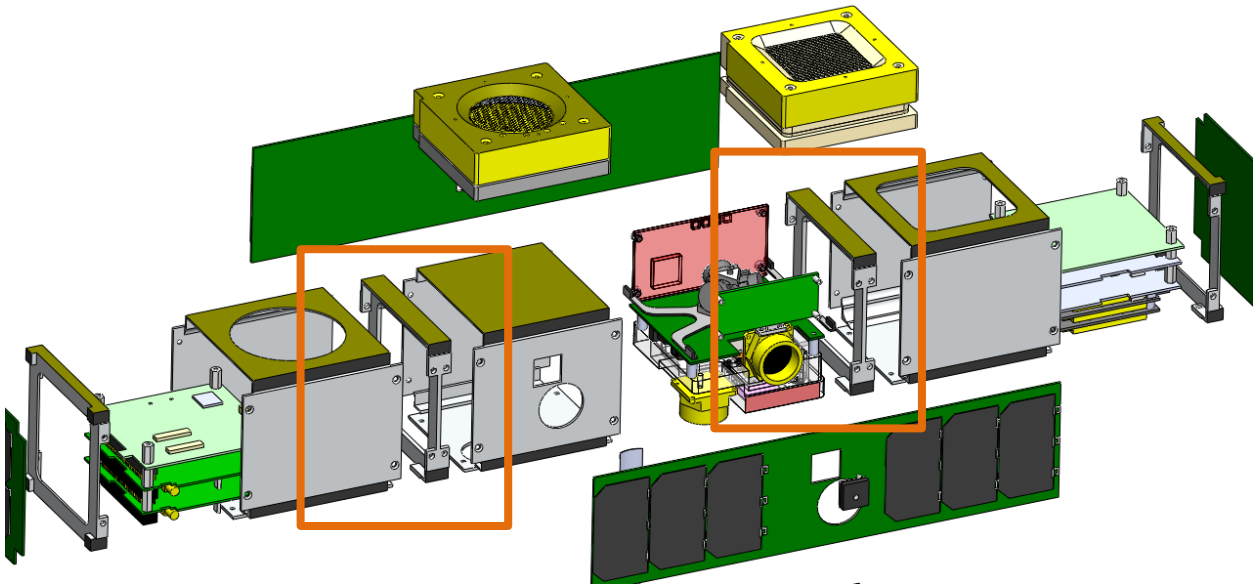


Figure 7. NSF EVE structure concept with the section connectors highlighted

Figure 7 above shows an exploded view of the NSF EVE (Equatorial Vertex Explorer) satellite. The NSF EVE mission was a proposal for the National Science Foundation developed by the SDL. EVE was composed of three modules that were customized for the mission; the left and right modules each held part of the electronics suite as well as one of the science sensors. The middle module held the SDL's current iteration of the ADC module. The breakthrough in the design was between the modules; the module section connectors, highlighted in the orange boxes, were identical and established a common interface between completely different modules. Each module simply had to be machined to meet the interface of the section connector. From there, each module was fastened to the section connectors and a complete satellite was made.

The EVE design opened up many design possibilities; modules could be developed and assembled independently and then integrated easily. If the SDL wanted to upgrade or change the hardware in the satellite, two of the modules could remain the same as the third one was

modified. The EVE satellite proved that modularity had great potential in the development of satellites and that designs could be reused between missions. However, the NSF mission was only a concept. The drawings and design were made to be included as a picture in a proposal rather than for submission at a design review. Certain pieces, such as the section connectors, were not able to be machined or would be extremely costly. A successful design would have to meet these challenges in order to be marketable to future missions, both within the SDL and as a commercial product.

3.3 FINAL DESIGN

The research journey from the initial decision to avoid a premade 3U structure, to the NSF EVE concepts, and finally to the resulting truly modular and reusable structure took almost two years. All of the structure concepts are iterative and have features that make them better than the previous ideas but present challenges for future improvement. The result of all of the structure research has been a section connector that is simple in design and easy to integrate and a standardized wall section that can be ultimately customized for any need. This is a major conceptual breakthrough that has not been promoted in the CubeSat community.

With the development of the standardized section connector and shell design, the commercialization of the structure becomes an interesting concept. Rather than UT Austin developing a structure that works for their own satellites, these modular interfaces allow a customer to easily work with the SDL. This ‘customer’ can be a different student team at UT or another school or even a scientific or government institution that has a payload to fly. In the remainder of the structure section, the research results are towards a generic customer to promote the broad sense in which these ideas can be applied.

3.3.1 Section Connector

The module section connector is seen below in Figure 8 and provides the essential common module interface between any two modules. Mechanical drawings and dimensions are presented in the Appendix. The section connector is machined from 6061-T6 aluminum as specified by the CubeSat standards. [2] Each section connector connects to a single module with eight #4 screws. The SDL's own initial structural analyses indicate that eight fasteners per module should be enough to withstand the vibrational and loading requirements specified by the 3U P-POD CubeSat standard. [2]

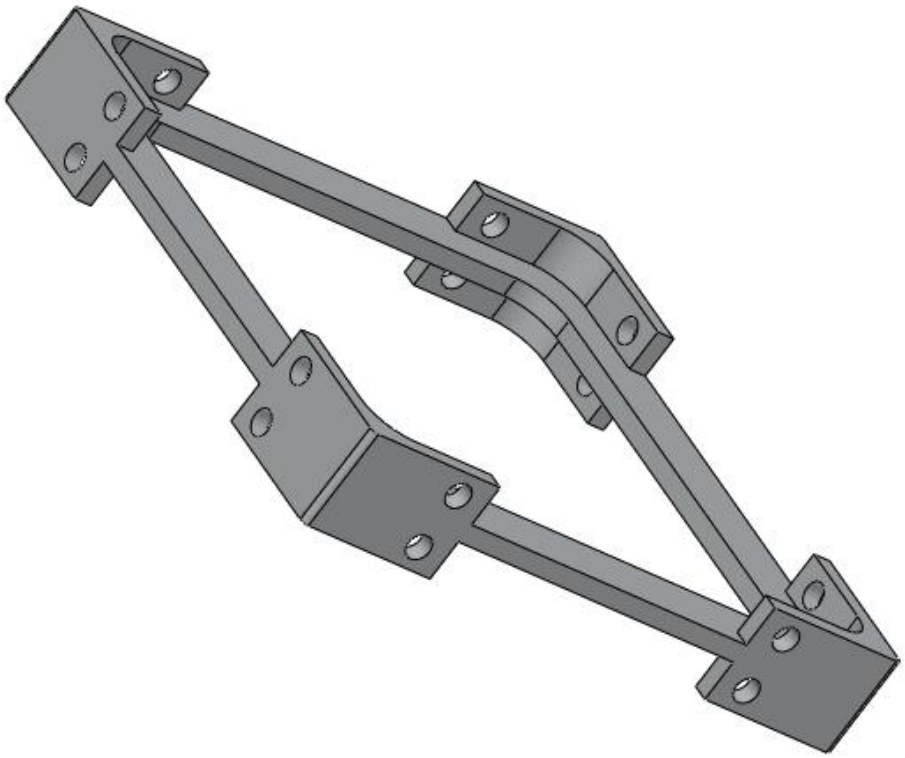


Image Credit: Steven Arestie

Figure 8. The section connector seen at an angled view

The fasteners used on the section connector are military standard #4 fasteners. The vertical walls on the connector are wide enough to accommodate #4 locking helical inserts; this allows the modules to be assembled together easily with the section connector because there are no nuts and therefore no tools need to be inserted on the inside. These thick walls also make a wide contact with the shell walls to transfer any forces from the launcher or launch environment over a relatively large surface area. The middle of the section connector is hollowed out to save mass and allow for cables to pass through. All of the interior fillets are standard machine bit sizes to save time and reduce the chance for error. The interior edges are broken (rounded to eliminate sharp corners) to prevent cables from being cut during assembly or launch. Though not shown in Figure 8, the entire section connector will be alodyned to increase electrical connectivity. The portion of the section connector that makes up the CubeSat launching rail will be anodized as per the CubeSat launcher specifications.

Finally, the section connector is two-face machineable on a mill. When parts are being manufactured, the majority of the cost comes from the time spent making each part. Strict tolerances on machined components increase the price for machining and verification. Each time the part needs to be rotated on the machine so a different face can be machined, the machinist must relocate the origin where the dimensions are based. This increases the manufacturing time and chance of error. The section connector can be machined from only the top and bottom faces. The helical insert holes, which are subject to a much less rigid tolerance, can be easily added later. The cost is also further reduced from the standard design of the section connector. Since all the connectors feature the same dimensions and are identical, the cost per part goes down as more are ordered. This leads to a cheaper and faster satellite fabrication for the customer.

The module section connector is the key in the reusability of the Bevo-2 and ARMADILLO structure design. It is a single design that has been adapted to these very different satellite missions successfully. For the ends of satellites, the section connector can literally be cut in half to create end caps for the satellites. A customer interested in the module design would be presented with interface information on the section connector and would have enough information to develop their own satellite modules if they wished.

3.3.2 Module Shells

While modules on a satellite using the section connector must only meet the interface requirements of the section connector, a standardized module 'shell' provides a preliminary level of support for module design. In a small satellite context, the term shell simply refers to the four walls of the satellite enclosing a hollowed volume. Figure 9 showcases a sample shell from the ARMADILLO and Bevo-2 missions that interfaces with the section connector in Figure 8. Appendix A has detailed mechanical drawings of the generic module shell.

Each shell component has two main features. On the top and bottom of the shell in Figure 9 are the extensions that allow for the shell to connect to the section connectors. Flat head screws are used to fasten into the section connectors to keep the outer face flat for solar panel mounting. Similar to how the section can be customized to the user's needs, more fasteners can be added to the extensions. The main feature of the shell is the area in between the extensions. Many different modules will require different lengths; a scientific payload may only require a 0.5U (about 57 mm in length) volume while the ADC module for Bevo-2 and ARMADILLO requires over 1U in volume (114.5 mm). The shell is designed to be linearly expandable to meet the needs of all of these custom modules. Figure 9 also highlights the conformance to the 100 mm rail limit

as specified by the CubeSat standard while expanding the walls between the rails to increase the interior volume for components.

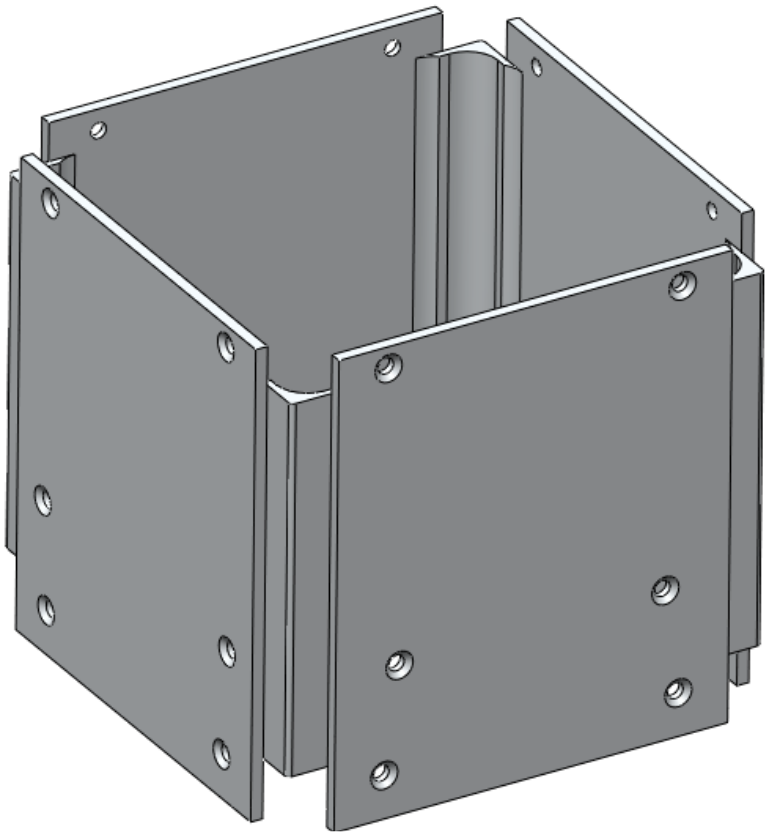


Figure 9. A sample shell from the Bevo-2 and ARMADILLO satellites

The shell's wall design is targeted at making integration easy for the customer. The interior space is compatible with the standard CubeSat electronics boards and provides more space around the sides of PCBs to route cables. On the shell, the walls are 2 mm thick to accommodate #2 helical inserts needed for solar panel mounting as well as provide necessary support for #4 flathead screws. These walls can also be thickened to accommodate certain mounting schemes for antennas and sensors. Interior components can be easily fastened to the walls of the shell either using helical inserts or nuts, as seen by the four screw holes featured in

the figure above. The customer can place mounting holes wherever they desire in the shell walls and the remaining unpopulated areas can be machined out to save mass or for ease of assembly.

Similar to the section connector, the module shell benefits from an ease of initial machining. Bevo-1 also employed a shell design similar to Bevo-2 and ARMADILLO, but the interior corners were too tight for traditional machining; instead, the shell was made using a Wire EDM which was costly and lengthy. To avoid this problem, the module shell design can be machined on a mill machine from the top and the bottom using standard bits. The sides can then be customized with fastener holes and wall cutouts (such as for solar panel connectors) as necessitated by the customer.

3.3.3 Assembly of a Satellite

The section connector and module shell provide the foundation for a satellite's structure. With these parts, the immense challenges of assembling and integrating a fully-functioning satellite have been simplified. Figure 10 below shows the Critical Design Review (CDR) level drawings of the integration of the Bevo-2 satellite. Bevo-2 is designed to be integrated using the section connectors and wall shell templates introduced above. The leftmost module is the ADC module, the middle section is the payload module, and the right module is the electronics module with the computer, power system, and radios. With 32 fasteners, these three modules can be pieced together into a satellite.

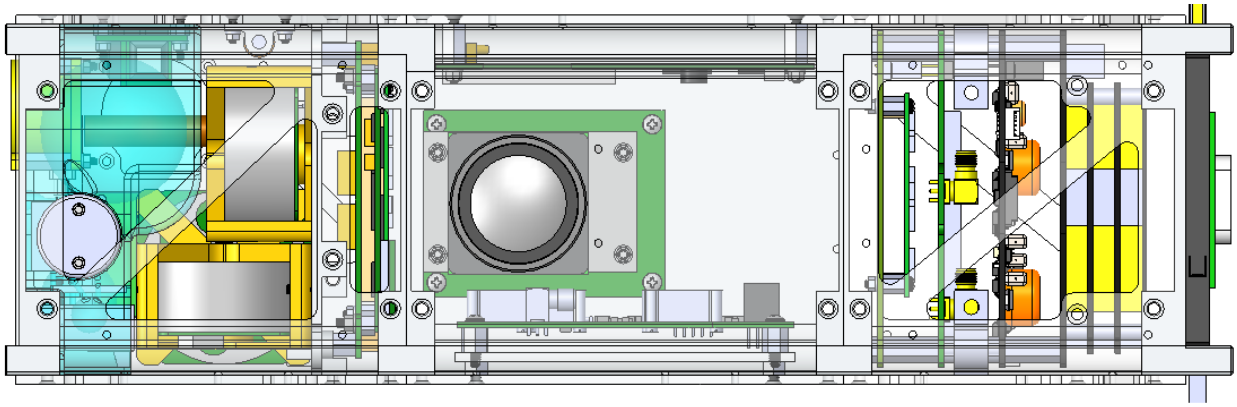


Image Credit: UT SDL

Figure 10. Bevo-2 is separated into three modules and connected by section connectors

The modularity of the sections and section connectors has been physically tested in an engineering design unit. In preparation for the ARMADILLO Preliminary Design Review in August 2011, the SDL "printed" the current satellite model in plastic as seen in Figure 11. All the circuit boards, interior components, module shells, and section connectors were constructed using a stereolithography 3D printing technique. The structures team was able to use the plastic models to conduct rough fit checks (to the tolerances allowed by the printing process) and develop and check assembly techniques. This also included the installation of fasteners, nuts, and standoffs. In the end, the cost of printing the model paid off in the experience gained; several assembly issues were discovered in the payload module as well as assembly tool issues within the ADC and electronics modules. However, the modular design greatly simplified the integration and proved that the section connector and wall shell concept would work during assembly of the flight satellite.

Figure 11. Plastic model of the ARMADILLO satellite

Beyond Bevo-2 and ARMADILLO, it is useful consider a more generic example of a satellite made up of three different modules. One of the modules is developed by a science customer and the other two are provided by a university lab. The university can assemble their modules while the science customer constructs their module separately by following their procedures and practices. The final integration of the satellite before testing would only involve connecting any wires between the modules through the section connector and then affixing the science payload to the module connector. This same concept is what allows the SDL to work on the ARMADILLO mission concurrently with Bevo-2. Looking back at Figure 10, the ADC and electronics modules are identical between these two missions; the only different module is the

organization and interior components of the payload module. Thus, a considerable amount of the structures engineering has only been done once and benefits two missions simultaneously.

3.3.4 Structure Interface Control Documents

The key to a successful standard is to establish firm yet flexible guidelines and maintain an interface control system to ensure all requirements are met. Interface control documents (ICDs) are pivotal in the assembly of a satellite, especially when different institutions may be developing different parts of the satellite. Currently mechanical ICDs have been developed for the interface between the section connectors and the wall shells. These documents specify tolerances on the machining of the parts, locations and types of fasteners to be used, as well as any cables or components that may pass through the section connector. However, these documents have been tailored to the specific missions within the SDL. To create a commercially viable modular design, the ICDs will need to be expanded on many fronts. These ideas are discussed in the future research section.

3.4 MODULARITY AND REUSABILITY

The common theme behind the development of the Bevo-2 and ARMADILLO structure is to establish modules that perform a specific purpose and are reusable on both satellites. As shown for these two unique missions, the ADC and electronics modules make up a complete, functioning satellite bus while only using two-thirds of the allotted 3U CubeSat volume. The design research into the section connectors and module shells has simplified the interfaces between modules to an easily accessible level; scientists who have no knowledge of spacecraft can develop an instrument to finite standards and fly in space. Likewise, universities with a standard satellite bus can fly numerous scientific missions from virtually any customer.

Every subsystem interfaces with the structure. With a modular and reusable design, a significant portion of the non-recurring costs associated with the structure has been eliminated. Pumpkin and ISIS have presented the market with a standard structure that addresses this concept generally, but the designs still present significant constraints. With the Bevo-2 and ARMADILLO structure research, a standard has been established that provides firm guidelines at the crucial interface points between modules but ultimate flexibility within the module. This simple concept has been proven and is clearly applicable to almost any future mission.

Someday, assembling a satellite could be as simple as pulling parts off a shelf. With the development of an industry standard interface, a customer could dictate a requirement for communications, power, or attitude determination and control. Previously designed, built, and tested modules with flight heritage could be combined to create a functioning satellite bus. The customer then has the remaining volume of the 3U satellite to design their own payload. Likewise, many different payloads can be developed for a common bus design. In a rapid response scenario, such as a hurricane that can be observed from space or loss of a communications satellite, modular satellites can be quickly integrated and flown. All of these ideas stem from a firm foundation of research into modular and reusable satellite bus designs.

3.5 FUTURE RESEARCH

There are many hidden subtleties in designing a structure, most of which will show up during the flight qualification testing or first flight of a standard. Most of the concepts and ideas presented above are the result of nearly four years of small satellite structures research. Though the ideas behind the shells and section connectors have been iterated many times, there are still significant avenues to pursue in the structures research.

The next logical step for the section connector is to investigate incorporating electrical connectors on the middle face of the connector. Some candidate connectors could be the Glenair Micro-Crimp connectors. [6] Ideally there would be two distinct sets of connectors: one would be for the satellite bus and the other would be dedicated to the payload. The bus connector would have enough pins for different power voltages, ground, all of the data lines, and any other needs. The other connector would allow the payload to communicate with various instruments over the satellite, such as temperature sensors spaced out on the solar panels or multiple GPS antennas not located on the payload module face. The inclusion of the electrical connectors would increase the difficulty of establishing and maintaining a standard, but provides a more sound solution than running cables through the satellite.

More ICDs need to be created for each module as well as the section connector. The interface control documents need to include infrastructure to communicate EMI and thermal properties of different modules that might affect the performance of adjacent modules. Additionally, with an active attitude determination and control system, mass properties such as the center of mass and moments and products of inertia should be included. This may lead to a more stringent routine of shaping the shells of the modules to achieve a precise location of the center of mass.

Beyond the 3U standard, 6U CubeSats (twice as wide) are becoming more accepted because of their increased volume but still compact and small form factor. Companies have begun to develop 6U launchers to deploy these satellites and will likely be used by many research institutions. The concepts behind the section connector could be expanded to adapt to the 6U standard or even as simple as joining two 3U CubeSats together. This research would be

exciting to pursue in design, but is probably not applicable until a specific mission requiring the 6U size is presented.

Finally, as mentioned in the modularity and reusability section, trade studies and systems engineering level analyses need to be conducted to examine the challenges and opportunities for developing off-the-shelf level components and the missions they could satisfy. The science and satellite community will benefit from simplicity behind launching missions quickly and reliably.

4 FLIGHT SOFTWARE

Structure and software exist in two separate realms; the structure is physical and can be concretely visualized during development while the software can be vague and hard to intrinsically understand. While structure and software are inherently different, many of the concepts from structure design such as creating modules and specifying a well-defined common interface can be directly applied in developing a reusable flight software architecture for CubeSats. Similar to the structure, Command and Data Handling (CDH) software programmers indicate their desire for their software to be applicable and transferrable between projects. However, from the SDL's experience with the software from FASTRAC and Bevo-1 to Bevo-2 and ARMADILLO, this is not always the case; the new programmers realize that it may take more time to decrypt (usually uncommented) code from a previous mission and adapt it to their own rather than to start from scratch. The UT SDL is in a unique position. The lab has had the experience of writing its code from scratch on two missions and then realized the issues in transitioning the code between missions. This has led to research into how the flight software code can be written in a way that is modular and reusable for the Bevo-2 and ARMADILLO missions as well as future projects that will be worked on in the SDL.

Finally, the ideas presented in this chapter are the result of five years of an aerospace engineering degree with little formal programming instruction. Many of the concepts are developed with an engineering mind and targeted at small satellite systems. Though the ideas here are not unique to the SDL, many other small satellite programs are somewhat closed-mouthed or reluctant to update their websites on their programming practices making it difficult

to compare the SDL's software architecture with other institutions. The flight software concepts here are the framework of a philosophy that has so far been successful.

4.1 COMMAND AND DATA HANDLING OVERVIEW

The Command and Data Handling (CDH) system is the master of the Bevo-2 and ARMADILLO satellites. The heart of the CDH system is the Phytec PhyCORE LPC3250 System on a Module (SOM) seen below in Figure 12.

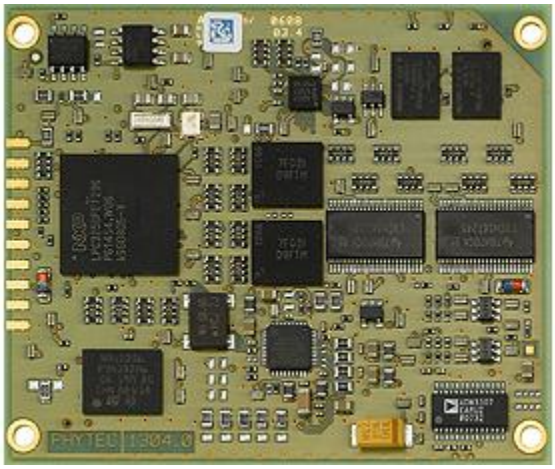


Image Credit: Phytec

Figure 12. The Phytec LPC3250 SOM

Several computers were considered for use in the two current 3U missions and a trade study was conducted comparing the LPC3250 to the BlueTechnix TCM-BF537, the Microsys MPX5200 and the Phytec i.MX31. [7] In the end, the LPC3250 was chosen for its size, power consumption, Linux compatibility, and interface selection detailed below in Table 1.

The LPC3250 is unique from the other processors studied because of its support from the Linux community. The Linux kernel for the SOM is developed through the LTIB (Linux Target Image Builder), which allows the user to select specific packages (such as drivers for the i2c

interfaces, etc.) and tweak the implementation of the system while compiling the kernel. Additionally, the online community of developers is extremely friendly and helpful with troubleshooting issues.

Table 1. LPC3250 characteristics [8]

Cost	$199
Processor Information	NXP LPC3250 ARM9 at 208 MHz
Size	58 by 70 mm
Mass	23 grams
Power	372 mW
Operating Temperature	-40 to 85 C
Interfaces	7 UART, 2 I2C, 2 I2S, 2 SPI, 1 USB
Memory	64 MB Flash, 64 MB RAM
Software	Full Linux Kernel, 2.6.34

Currently the LPC3250 is running the 2.6.34 Linux kernel and is connected to a desktop Linux PC for software development. When the computer turns on in flight, it will first boot Linux. Once Linux is loaded, the flight software will automatically launch. All of the flight code is programmed in C and C++ which gives the software incredible low-level processing power as well as the benefits of classes (and thus modularity) inherent in C++. C/C++ is also a well-known language that can be developed and transferred easily between projects and hardware.

The flight software for the Bevo-2 and ARMADILLO missions is codenamed ‘Hookem’ and is broken into two distinct approaches: (1) the software interfaces between the CDH and the other subsystems, and (2) the operational interface between the ground user or mission script instructing the satellite and the satellite completing the task. Within the flight software code, instructions and subsystem capabilities intertwine to produce a functional and capable satellite. The state of the software is controlled by a mode manager that responds to the satellite’s physical and virtual environment. All of the interactions between the CDH and the ground or other

subsystems are strictly governed by Interface Control Documents (ICDs) both for the safety and mission success of the satellite. This thesis introduces some of the operational level concepts to provide a simple background to support the subsystem level interfaces research but will not discuss the satellite operations-level software philosophy in great detail.

4.2 TRANSITIONAL ISSUES

The code used on the FASTRAC satellite was designed for a different system architecture and was therefore not directly applicable to future missions. 'Hookem' is more similar to the flight software for Bevo-1, codename 'Blackbird'. Blackbird was operationally successful in ground testing but was not tested in space since Bevo-1 did not power on. Similar to Bevo-2 and ARMADILLO, the software ran in a Linux environment, was programmed in C/C++, and was designed to be completely autonomous in operation. The concepts of modes and most of the functional specifications came from the design philosophies of Bevo-1. However, with the conclusion of the Bevo-1 mission, there was a lack of transferred understanding of programming and Linux to the next project; the code was poorly commented in critical places (perhaps due to the time crunch to produce functional code before delivery), and the documentation was never fully completed. Thankfully, one of the original programmers of Blackbird is still working on his upper level degree and is available for limited help on the software. Additionally, all of the code developed for Blackbird, including prototype code for system level interfaces, has been left behind and is easily accessible. However, there were still significant challenges that led to a complete fresh start on the flight software.

These transitional issues between projects highlight a significant problem in the reusability of flight software between projects; many of the hours of troubleshooting and lessons

learned developing Blackbird were lost with personnel changes. This led to a central research goal for the design of the 'Hookem' software: create software that is designed to be used on multiple missions and make it easily transferrable to future programmers. The result of this research turned into a modular software architecture governed by ICDs. The core CDH software will remain unchanged between the missions while the subsystem software modules can change.

4.3 SOFTWARE DESIGN

The 'Hookem' software is designed to operate both autonomously or under ground command. Operational modes and transitions are actively monitored depending on the satellite's state. The 'Hookem' software encompasses software specific to the CDH as well as most of the subsystem level code (excluding ADC and payloads). For this research, it is assumed that the flight software investigated will incorporate the same design architecture between the ARMADILLO and Bevo-2 missions. The actual version of 'Hookem' will vary between missions because of different hardware and mission operations requirements. Both missions will read commands from a mission script. The mission script contains a complete list of tasks for the satellite to execute and can be modified on-orbit.

4.3.1 Mode Diagram

The software operates in modes depending on the physical state of the satellite. 'Hookem' has four operational modes: Startup, Autonomous Command Execution (ACE), Fail Safe/Low Power (FSLP) and Ground Uplink and Downlink (GUD). These are shown in Figure 13.

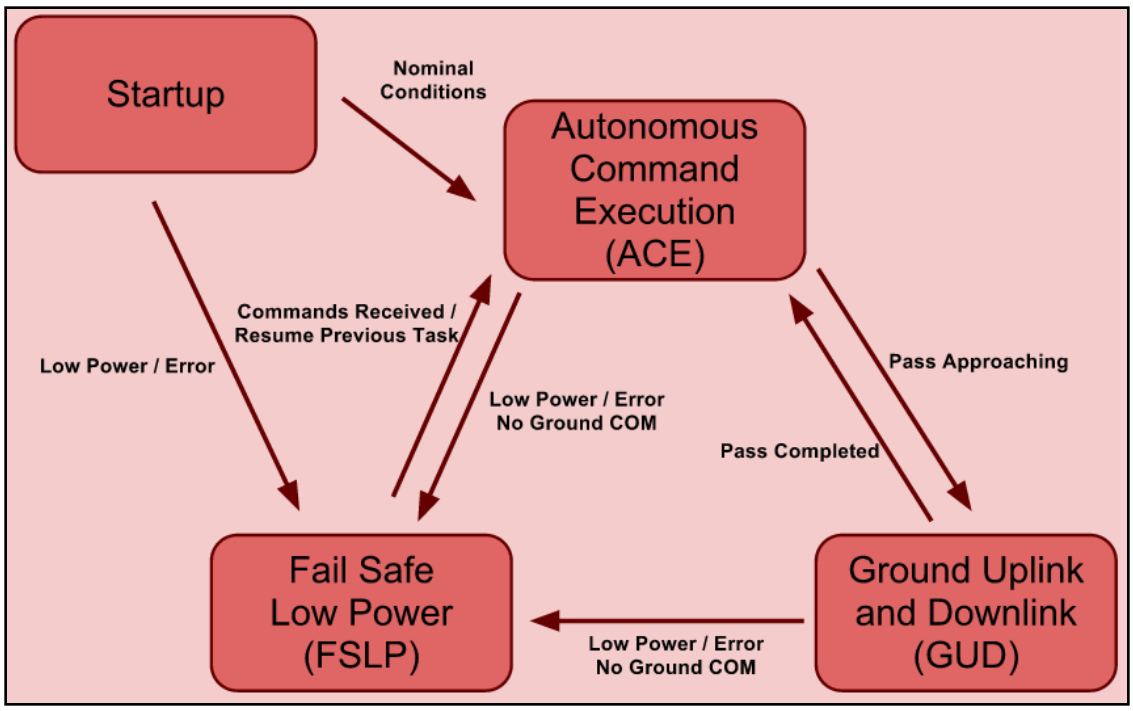


Figure 13. Hookem mode diagram with transitions

'Hookem' enters the startup mode either upon initial ejection and power-up from the CubeSat deployer or after a hard reset in flight. The software will have the capability to determine if it has already been operating in space by looking at the mission script and deciding where to continue its operation. From the startup mode, Hookem will nominally transfer to ACE. Most of the mission will take place in ACE, including all mission objectives dictated by the mission script. The GUD mode is defined separately from ACE. Though uplink and downlink will be designed to be autonomous in nature, the hardware and software resources are significantly different than any other activity in ACE so it makes sense to establish a separate mode. Finally, the FSLP mode is present in Hookem to provide a safe environment for the satellite to recover from a mission threatening situation. There will be a finite number of reasons for the satellite to enter FSLP and these are defined in the software. The most common transition

to FSLP is due to low power and the satellite will recharge until the batteries are at an acceptable level. Other major malfunctions, such as a failure of the ADC computer, will warrant a transition into this mode. From FSLP, the satellite will beacon an error code informing the ground of its status and await a reply. Additionally, per FCC regulations, the satellite must transfer to a silent mode if no ground contact is established in a reasonable amount of time.

While mission level activities will be confined to their order on the mission script, many other activities will be running in the background within all modes. For example, the LPC3250 features a ‘watchdog’, a chip that physically cycles power within a set amount of time unless the watchdog is ‘fed’; this prevents the software from locking up and becoming unresponsive. Other activities, such as a basic Morse code beacon and constant battery voltage level monitoring also continuously take place within the software. These activities necessitate the need for a simple software interface between the CDH and all of the subsystems.

4.3.2 Software Operations

Beyond the modes, the software follows a specific flow between a command being read from the mission script (placed there before launch or by on-orbit ground communications) and the command actually being executed. Figure 14 details the flow of the Hookem software.

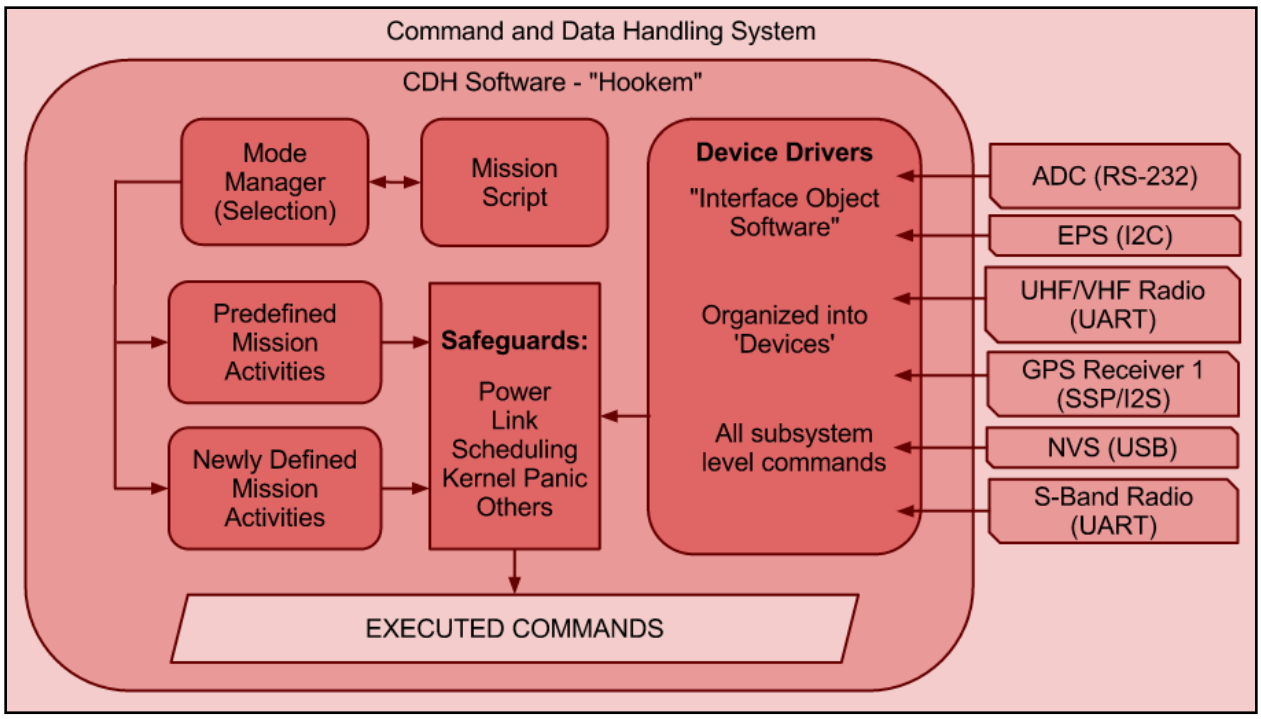


Figure 14. Hookem software flow diagram for commands

Starting with the mission script, the flight software interprets a command. The code then checks if the satellite is currently in the correct mode and decides if it needs to wait to complete the command or can transition modes at that instant. From there, the flight software looks to see if the mission activity has been predefined (programmed before launch) or is a new mission activity that has been programmed on-orbit. From the other side of the software flow, the flight software accesses the full suite of subsystems from outside the CDH. Each subsystem interfaces into Hookem through device drivers called Interface Object Software (IOS). The IOS creates a device for each subsystem with every subsystem level command that can be called. Finally, each of these activities called by the mission is composed of commands made available to the software through the IOS. Hookem imposes safeguards before executing commands, such as checking the power levels of the satellite, monitoring the saturation of the data lines, as well as

managing Linux level problems like memory usage, processor scheduling, or kernel panic. After this, the command is executed. These core CDH responsibilities remain constant between missions even as the hardware or payloads change. Only the list of IOS commands changes between missions.

Consider a command to beacon the power system voltage through the radio. The mission script may read '*BEACON VOLTAGE'* and the mode manager realizes that a simple beacon can be completed in any mode. Because the satellite was preprogrammed with this command, the software runs through a short list of subsystem level commands it must complete to finish the command: first it must query the power system for the voltage, turn on the radio, create a packet for the radio to send, and finally pass the packet to the radio for transmission. Finally, the software verifies there is enough power to transmit the information without error and transmits the data. Modifying the above example, a new mission activity may be for the satellite to beacon the last 72 hours (or random increment of time) of power system temperatures. The activity can be programmed on-orbit to parse log files for the desired amount of time and then transmit the data, similar to the first example.

'Hookem' can also execute time or condition dependent commands. For example, the software may choose to take an action if a GPS signal is lost on on-orbit or automatically perform a safe reset of the computer every few days. The flight software is responsible for parsing the mission script for any activities that have these requirements, checking to see if the proper conditions are met, and then executing the command.

4.4 INTERFACE OBJECT SOFTWARE

Key to the success of mission operations for the ARMADILLO and Bevo-2 missions is the organization and execution of commands. The goal of the commands, whether read from the predefined mission script for acted upon from ground instructions, is to be high level enough for any ground operator to understand and powerful enough to complete complex actions with only a few instructions from the ground. The flexibility in the flight software stems from the capability of Hookem to modify its 'encyclopedia' of activities on-orbit; literally, the software can append all of the necessary steps to complete new activities to a certain definition file. This allows the software to respond to unpredictable changes in the mission or expand the mission past the initial planned useful lifetime. The success of being able to define new activities stems from the full suite of functionality provided through the interface object software.

4.4.1 Interface Object Software Concepts

The interface object software is defined as an abstraction layer within the coding hierarchy of the flight software. In the SDL, each subsystem is responsible for writing the subsystem level code. This subsystem level software is the driver of the subsystem and is responsible for managing the interfaces (such as I2C or UARTs), writing and reading from memory buffers, and preparing data to be transmitted to the CDH. The IOS defines an upper level interface for this code; the subsystems are responsible for providing functions that can be called by the CDH mission activities. On the other side of this abstraction layer, the high-level subsystem functions are the lowest level pieces of code seen by the CDH operations. This is shown in Figure 15. IOS software abstraction layer

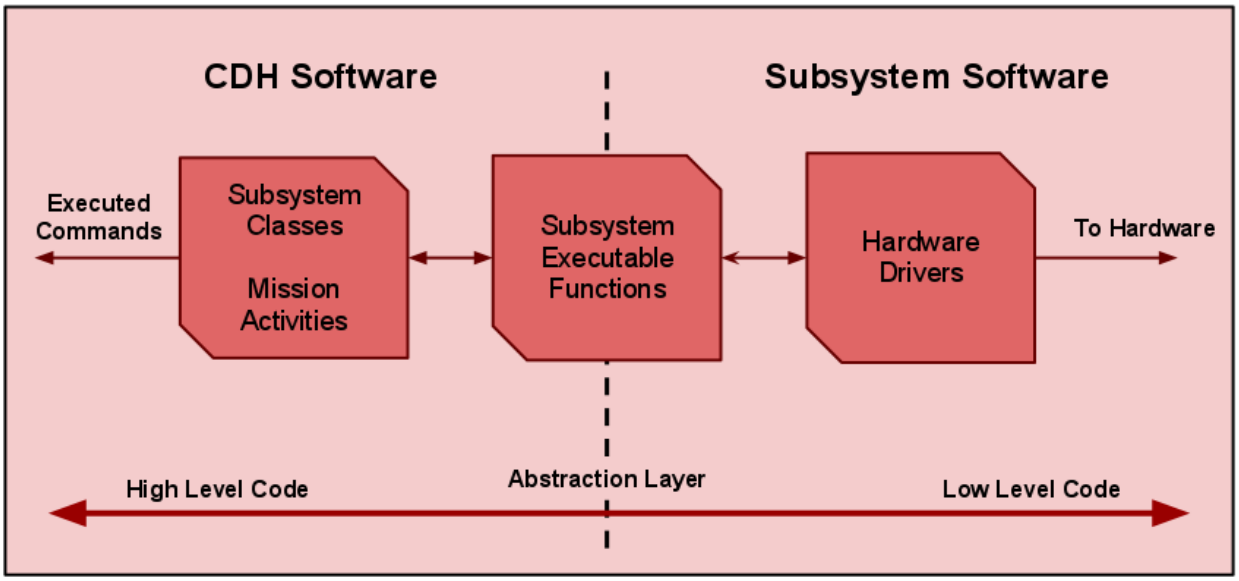


Figure 15. IOS software abstraction layer

The CDH software uses C++ classes to create devices of each subsystem, each with a full suite of the specific subsystem's commands. From there, these functions make up the activities that can be called by the mission script as discussed above. In general, the IOS defines the guideline that ensures that the Hookem flight software will work; it creates a simple interface between the CDH and subsystem code within Hookem. Beyond programming the interface, the IOS is separated into distinct modules and governed by ICDs.

The basic functions that the CDH can call within a subsystem are referred to as native functions. However, within satellite mission operations, there may be a need to call more than one native function to complete a task. For example, the CDH will have the capability to execute six different functions on the power system to query power being generated by each of the six solar panels. However, it may be more operationally useful for the CDH to execute one command asking for all of the solar panel power levels and receive an array of six values. This is

a hybrid command. Hybrid commands are programmed as needed into each subsystem IOS and are made up of a collection of native commands. Through the IOS, the CDH is presented with a single hybrid function command to execute. Ultimately, the CDH still could call all six functions to receive the same data and the functionality is left in place within the power system.

4.4.2 Software Modules

Essentially the IOS software is like a USB driver on a modern Windows computer. The user can carry a portable flash drive around with them to plug into several computers. There is a defined abstraction level of software that aligns the drivers within Windows to the capabilities of the flash drive. Otherwise, the user would have to install new drivers every time they bought a new flash drive or went to another computer. The IOS for Hookem aims to create this simplicity within the flight software by establishing modules within the software. This is illustrated by Figure 16 below. Each satellite will have a specific number of subsystems which defines a set number of modules and thus device drivers. Figure 16 highlights the basic subsystems of power (EPS), radios (COM), a camera (NVS), a GPS receiver, and an ADC system. The solid red arrows between the modules and the CDH symbolize the IOS interaction and transmission of data between the CDH and the subsystems within ‘Hookem’.

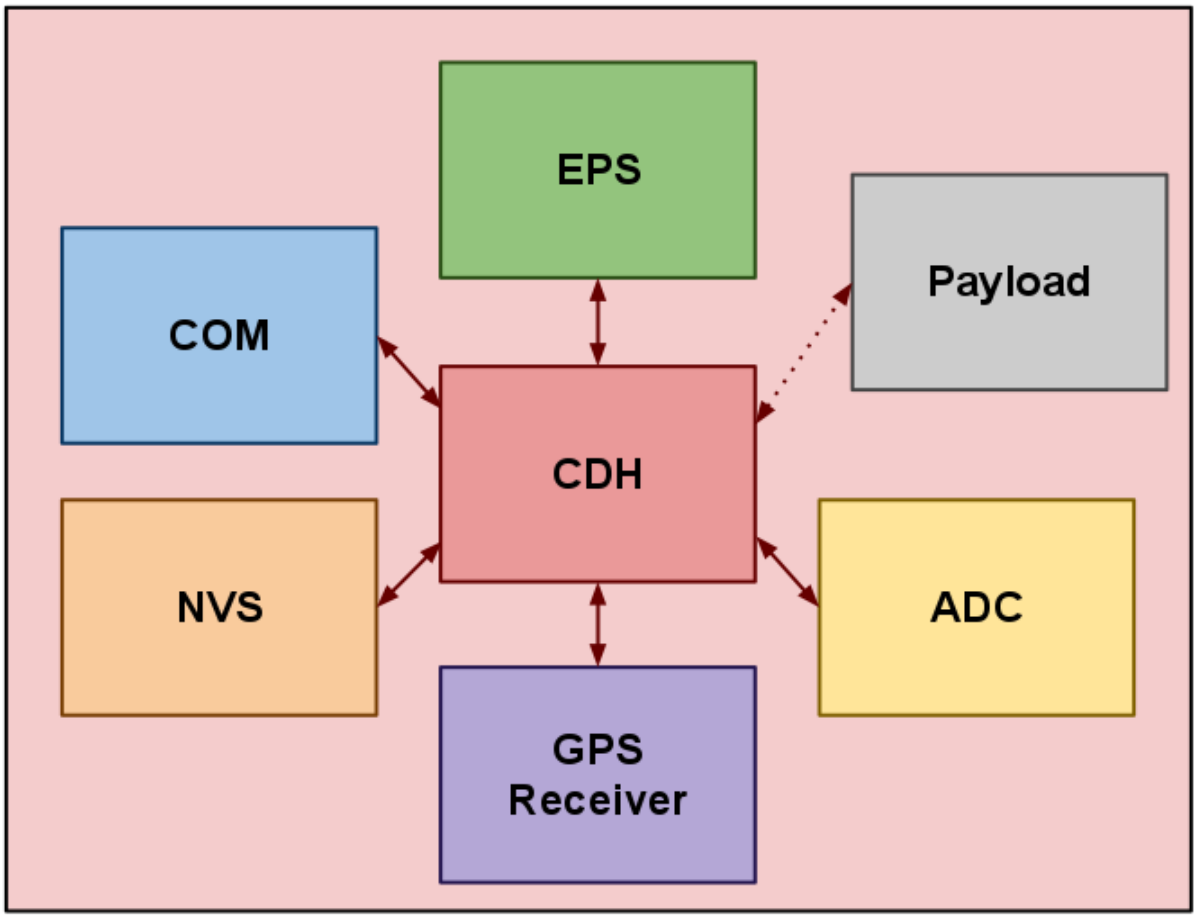


Figure 16. Hookem flight software modules

Figure 16 defines a fully functional satellite but there may be a need to carry another payload of any type. As highlighted in the structure section, the ideal satellite bus is one that can accommodate a suite of payload designs independent of the design of the rest of the satellite. The above figure also illustrates the relative simplicity in adding the software support for a payload. The driver software for the payload needs to be written by the customer (illustrated by the dashed line), and through the IOS, the CDH gains the capability to call all commands that deal with the payload.

This same concept applies to switching out subsystems for different ones. For example, UHF/VHF the radio currently being used on the ARMADILLO and Bevo-2 could be switched out for a new radio with more features. The new radio code module, governed by its own IOS and functions would be included in Hookem. The CDH would already have a device called ‘radio’ and the programmer would only need to simply modify the functions that can be called within that device. From there, the last modification would be to check the activity definitions file to check for any syntax changes (such as changing a command from sendBeacon to beaconSend, etc.). This ease of including, removing, or altering software pieces allows the software to be modular around a static CDH architecture.

Figure 16 purposely shows a lack of communication between the subsystem modules; instead, all communication between subsystems is controlled by the CDH. This is to maintain the modularity of the code and eliminate inter-subsystem dependence. From programming experience, it is relatively easy to allow software modules to communicate with each other and even easier to forget where the interfaces are once the code has been in development for a long time. Consider a radio beacon that needs to know the battery voltage and asks the power system directly for the information. If the power system changes and the syntax of querying the voltage is altered, the CDH software and the COM software would not be able to successfully get the information. By preventing the interaction between subsystems these issues are avoided as hardware and software changes over time or transitions to a new project.

4.4.3 IOS Interface Control Documents

Controlling the IOS code is a library of interface control documents that firmly establish the abstraction layer between the work of the CDH team and the subsystem programming teams. An IOS ICD has been created for all of the subsystems on the Bevo-2 and ARMADILLO

missions to clearly communicate all the relevant information. These documents specify the hardware interface for the subsystem, physical locations and file names of all of the subsystem code, verification plans, known problems, and general information about the programming status, CDH variable requirements, and interrupt needs.

The IOS ICDs also detail every native and hybrid function that is programmed into the software. For each function, there is a description and location of the function, a prototype of how to call the function, a list of the number and type of all input and output variables with descriptions, (for hybrid functions) a list of native commands called, and finally any programmer comments related to the command. The subsystem programmer can fill out the IOS ICD as a guide for the CDH team to completely interface with the subsystem software. This is especially useful for payload customers that may be developing a payload and software at different locations and where integration and testing may not occur until weeks or months before launch.

4.5 MODULARITY AND REUSABILITY

Similar to the structure, the software is designed into modules that come together through a common interface. As discussed before, for Hookem, the arrangement is more like a wheel hub with CDH at the center and all of the subsystems branching out of the center. The CDH core software and handling of satellite operations is designed to remain mostly unchanged between missions; it will be responsible for maintaining the computer hardware, parsing the mission script, and deciding how to perform actions based on constraints. From a specific CDH standpoint (rather than Hookem as a whole), only the ‘encyclopedia’ file that defines the mission script activities will need to be altered as the subsystems and payloads change. This promotes a strong reusability between missions, especially if the same computer is used on more missions.

All of the subsystems are carefully contained within software modules that are isolated from each other and interface with the CDH through the well-defined IOS abstraction layer. This modularity allows for subsystem hardware to be changed out with only slight changes required to the Hookem programming. The bulk of the work will be done by the subsystem personnel building up the drivers to the subsystem 'high-level' of functionality dictated by the IOS.

The 'Hookem' software can be used on both the ARMADILLO and Bevo-2 missions with only small changes in the radio, GPS, payload, and mission script files. These two projects benefit greatly from the common computer between both of the satellites as well as the other similar subsystems. Currently, the SDL has not had the experience or need to investigate expanding the flight software to other computers or tracking how the code will change. In principle, most of the concepts presented are applicable to all small satellite systems but the true actual implementation can be more difficult. The 'Hookem' software also benefits greatly from the full Linux environment that provides considerable functionality and support for the mission level activities. Many basic yet complex functions, such as process scheduling or memory management, would be hard to design to be reusable on other satellite systems without Linux. The 'Hookem' software will run simple Linux kernels and any necessary packages can be included.

4.6 FUTURE RESEARCH

The Hookem software is in various stages of programming completion. Many subsystems have been able to develop their code to IOS level readiness while other teams are still waiting on hardware so software development can begin. The future research goals will evolve through trial and error in working to implement the Hookem design philosophy and troubleshooting the

inevitable errors. Ultimately, within the time constraints of the ARMADILLO and Bevo-2 delivery dates, the first priority is to produce code that will meet the mission objectives. This includes learning how to routinely find, diagnose, and prevent errors during programming as well as clearly defining the operations of the satellite as it is in orbit. Considerable attention must also be paid to other technical issues, such as data storage limits, data redundancy, error checking, and the possibility of reprogramming the entire flight software suite on-orbit.

One clear avenue for the future research is the implementation of software programming metrics to track the changes and progression of code as it is being developed. For a small satellite engineering project, this would start as simply as monitoring how many lines of code were changed or added and how many hours the programmer spent working. As the Hookem software develops and is eventually used on future missions the software metrics can become more complex and employ different models to monitor the reusability of software between projects. It will be interesting to see how the software metrics research evolves for the small satellite systems in the SDL.

5 CONCLUSIONS

CubeSats will be integral in the future of spaceflight. In his analysis of student-built satellites, Jayaram asserts "The CubeSat revolution has not played out yet. In fact, it can be argued that CubeSats are still in their infancy." [4] The work in the UT SDL and numerous other institutions is helping to direct the union of technology, capability, and launch availability in the CubeSat community. As the designs begin to mature from simple student-built projects to major-funded science and technology missions, the inherent need for a modular and reusable CubeSat bus design is clear. Structure design and software architecture are only two facets of a whole satellite but represent a significant amount of engineering time, effort, and money. The key to success is well defined interfaces that provide a simple and firm foundation but allow for flexibility in design and implementation. The designs for structure and software modularity and reusability will develop over time and with flight heritage.

The concepts presented here are a first iteration of a process that works for the SDL. In order to benefit the small satellite community, these ideas need to expand past the walls of the lab to the criticisms and concepts of the entire community. The next major challenge is to iterate on the structure and software modularity and reusability to produce products that can appeal to paying customers. This may take months or years and will ultimately be left in the hands of future students. CubeSats are leading a revolution of easy-space-access with advanced scientific and technological capabilities. The research being performed now at The University of Texas Satellite Design Laboratory and other institutions is only the beginning of a new chapter in spaceflight.

6 ACKNOWLEDGEMENTS

First and foremost, I would like to thank Dr. Glenn Lightsey for nearly four years of guidance, support, and encouragement in the SDL. He has allowed my engineering mind to follow my dreams and given me an undergraduate experience like no other. Also for my entire undergraduate career, Henri Kjellberg has been mentoring, motivating, and teasing me. His passion for engineering and any intelligent conversation has been an inspiration. Don't worry Henri, I'll quit band when I go to graduate school. Many acknowledgements must also be given to the entire SDL family: Kit, Katharine, Sarah, Brian, Tyler, Steven, Eric, Chris, Shaina, Juan, Robert, Sebastian, Alexandra, Chinmay, Travis, and everyone else. They have been the best coworkers and engineering friends a guy could ask for. Thanks for asking the tough questions to make me think. I also owe a debt of gratitude to Dr. Humphreys for being my second reader and making me fall in love with space flight; your space dynamics class assured me I was in the right major. I have also had the two best advisors in the world. Sarah Kitten has been by my side for my five years here in the department and even gone as far to change the meeting time of a class for me. Pam Dahl has given me the most free-swag than anyone else in college and been an incredible friend and guide through the engineering honors world. Finally, thank you to my Longhorn Band family. You have given me the college memories, stories, and friendships that I will never forget.

7 BIBLIOGRAPHY

[1] "FASTRAC Mission Objectives," 2011. [Online]. Available: http://fastrac.ae.utexas.edu/our_project/mission.php. [Accessed 10 November 2011].

[2] R. Munakata, "CubeSat Information," 1 August 2009. [Online]. Available: http://www.cubesat.org/images/developers/cds_rev12.pdf. [Accessed 10 November 2011].

[3] "ARMADILLO Mission," 2011. [Online]. Available: http://armadillo.ae.utexas.edu/. [Accessed 10 November 2011].

[4] S. Jayaram, "A Review of the Role of Student-Built Spacecraft in Workforce Training and Innovation: Ten Years of Significant CHange," in *AIAA SPACE 2010 Conference & Exposition*, Anahein, CA, 2010.

[5] "CubeSat Kit," Pumpkin, Inc., 2011. [Online]. Available: http://www.cubesatkit.com. [Accessed 10 November 2011].

[6] "Series 79 Micro-Crimp Connectors and Accessories," Glenair, 2011. [Online]. Available: http://www.glenair.com/micro-d/series_79_micro_crimp/. [Accessed 10 November 2011].

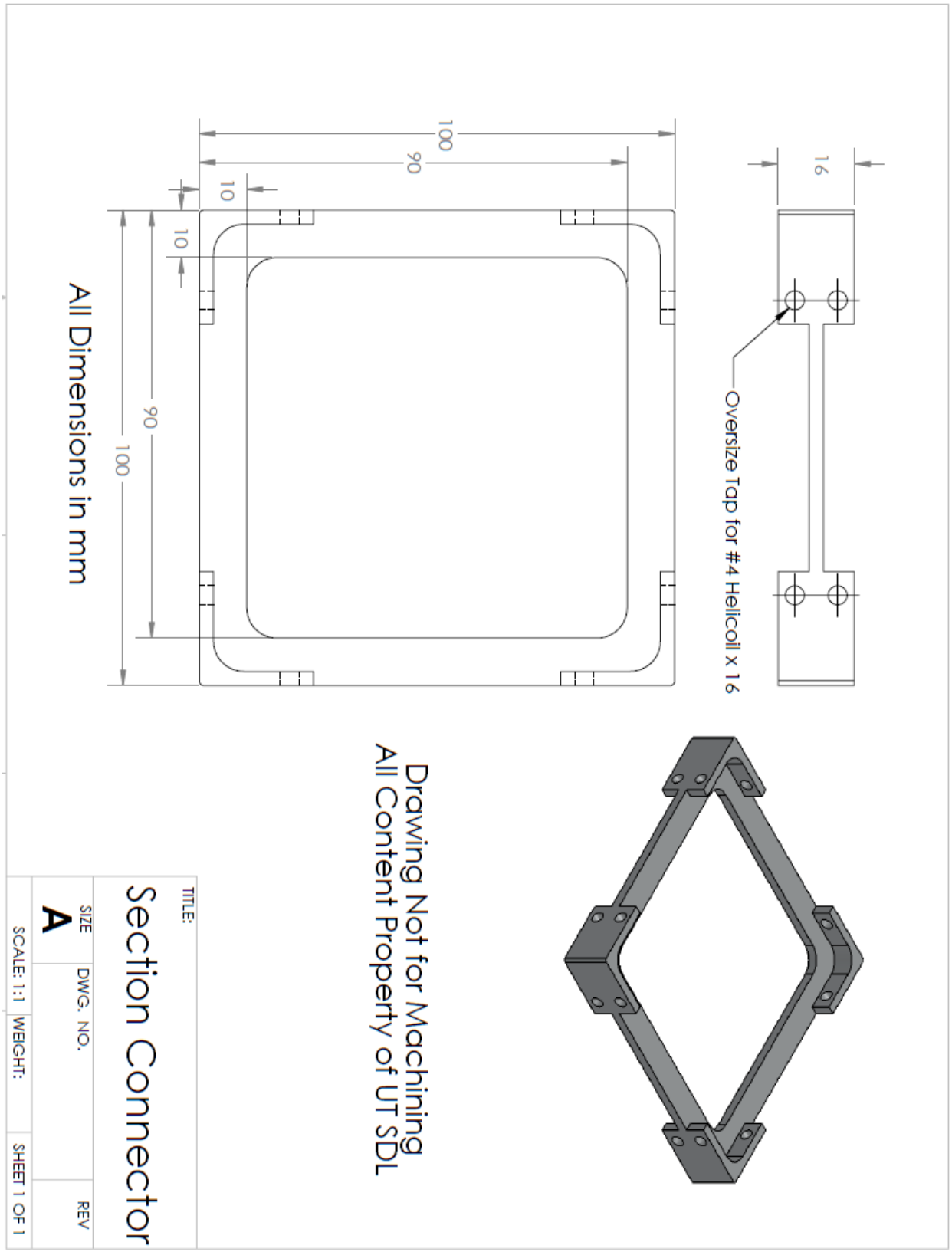
[7] T. Imken, "ARMADILLO Computer Trade Study," 2011.

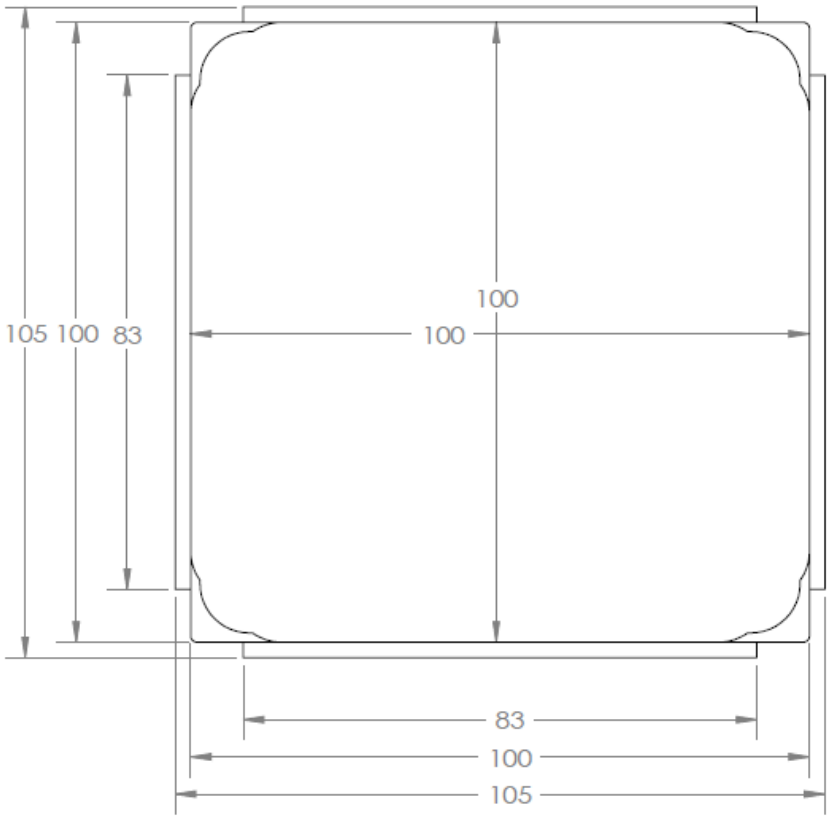
[8] Phytec, "phyCORE-LPC3250," 2011. [Online]. Available: http://www.phytec.com/products/som/ARM-XScale/phyCORE-ARM9-LPC3250.html. [Accessed 10 November 2011].

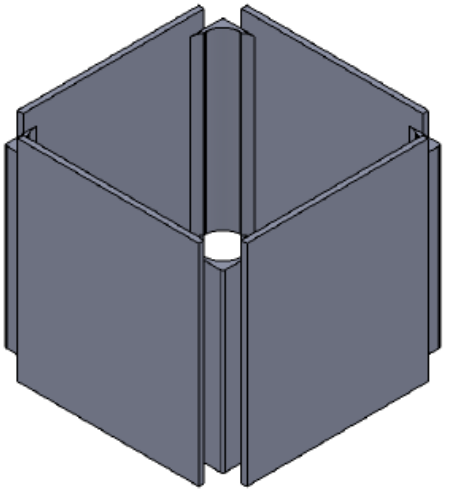
VITA

Travis Imken was born and raised in Austin, Texas. During his childhood and early adult years, all of his Lego models turned into some form of spacecraft or airplane; it was inevitable that he would become an aerospace engineer. Travis chose to attend The University of Texas at Austin because of the exceptional engineering school and for the unique chance to participate in the Longhorn Band. He managed to stick with both for his entire college career. Travis is also member of Tau Beta Pi, Kappa Kappa Psi, and has been declared 'Czar of the Undergraduates' in the Satellite Design Laboratory. He also worked one summer at the NASA Johnson Space Center. After graduation, Travis plans to pursue his dreams in graduate school by continuing to engineer small satellite systems. His favorite color is purple, he loves to eat BBQ, seafood, and texmex, and enjoys adding power tools and gadgets to his amateur handyman collection.

APPENDIX


16
Oversize Tap for #4 Helicoil x 16
100
90
10
10
90
100
All Dimensions in mm
Drawing Not for Machining
All Content Property of UT SDL
TITLE:
Section Connector
SIZE
A
DWG. NO.
REV
SCALE: 1:1
WEIGHT:
SHEET 1 OF 1





Drawing Not for Machining
All Content Property of UT SDL

All Dimensions in mm

TITLE: Wall Shell		
SIZE A	DWG. NO.	REV
SCALE: 1:1	WEIGHT:	SHEET 1 OF 1