

**16.851 Satellite Engineering  
Portfolio**

Anna Silbovitz  
December 10, 2003

## **Outline**

- I. Overall Learning Objectives
- II. Satellite Topics
  - a. STK
  - b. Orbits
  - c. Power
  - d. Environment
  - e. Attitude Control System
  - f. Communication
  - g. Software and Automation
- III. References

## **I. Overall Learning Objectives**

I was hoping from this class to gain a general understanding of how satellite engineering works, and how different satellite subsystems interact. I wanted to learn both technical and practical information about how to design the different subsystems.

I do realize that to thoroughly understand most of the subsystems, a separate class full of material could be taught on each. Considering this, I thought that what was taught in class did fulfill the objectives of the class. A good overview of each topic was presented; the basic technical material was shown as well as the high-level issues that are a concern in thinking how different satellite subsystems interact.

The homework provided an interesting opportunity to investigate the different topics taught in class. However, I often felt that there was not enough time to develop the problem in depth, so that the tools we produced made so many assumptions and used so many 'short-cut' type equations that, while they did demonstrate to us some of the trade-offs, they are probably not very useful in real design. I also felt, by around the fifth homework, that I was running out of fresh ideas; working with new people every week made it difficult to come up with problems unique to the whole group that involved interesting trades. Working with new people did add help to add a different perspective, since many of us have different backgrounds concerning what we already understood about satellites. However, I thought it would have been more beneficial to work with one group the entire semester, and for each homework to be built on the one done previously. That way, by the end of the class, each group would have a tool that analyzed many different subsystems with a large amount of depth and detail. Finally, I would like to add that throughout the semester I was continuously frustrated and confused as to what was expected on the homework. I felt the requests made of us were contradictory; it was impossible to keep each assignment brief but to adequately investigate several trade-offs. The result was that we had to make many assumptions, and restrict the problem, which made it less realistic as mentioned above. And often when my group did restrict the

problem in that way, in an attempt to keep the code, write-up, and amount of hours dedicated to the project reasonable, comments in the grading of the homework indicated that the problem was too restrictive.

In the remainder of this document, I will discuss many of the topics that were covered in class. Each discussion will include an overview of what was taught, additional information from outside sources, and also my personal comments on what I gained from the lesson. Each topic is intended to be a bare-bones overview; key equations and definitions are listed as well as particular points of interest. In the future, this document will be useful for me when I need to quickly reference the main points of a particular topic. Adding more detail would not be beneficial, since more in-depth information is so easily accessible through looking through class notes various textbooks, papers, and webpages (many of which are listed in the *References* section). For this reason as well, the problem sets are not included in this document, and interactions between subsystems are only mentioned occasionally. Each problem set provides a thorough explanation of how the different subsystems examined interact, and all problem sets and code are easy to reference when needed. Appending them to this would also have made the portfolio unnecessarily long, and would just put extra copies of them on my computer.

All source material is listed in the *References* section, but at the end of each topic the specific sources used in that topic are referenced. Class notes and handouts are not included in the *References*, but they did contribute to this report.

The topics covered were chosen because I have an interest in them, either because I investigated them more thoroughly in homework or because they are related to my fields of study. Each discussion will of course be influenced by my background and interests: my undergraduate studies were in Electrical Engineering, and my graduate work is being carried out in the Software Engineering Research Laboratory. I have previously taken classes that include Astrodynamics, Software Engineering (from a system design level) and System Safety. I am also currently an employee at MIT Lincoln Laboratory, in the Space Control group.

A final note: I do not have an equation editor on my personal computer, and do not easily have access to one that has it. I have rarely used it and do not find it difficult reading equations written without it; also I wanted to have the ability to edit this document and the equations at a later time if needed. For these reasons, I did not seek out a computer with an equation editor and instead have typed all equations directly into the Word document. Hopefully this does not inconvenience anyone else reading this.

## II. Satellite Topics

### a. *STK*

Although STK [2] is not a satellite subsystem, I believe that it deserves some attention and should have been given more in class. The first class consisted a very brief overview of an old version of the tool, with the expectation that we would then be able to use it in our homework. STK is a very powerful tool and can assist greatly in satellite system analysis. It is capable of doing a vast number of space-related calculations, such as determining distances between any objects in space over any length of time, or finding times that a satellite is in view of the sun. Also, it can propagate space vehicles and display in three dimensions the results; using these features, it can be used to verify results of calculations done for problem sets (such as, proving that a given change in velocity actually will transfer a satellite to a desired location) and to provide a visual demonstration of the results, such as the Hohmann Transfer shown in Figure 1.



**Figure 1. A Hohmann Transfer in STK**

Having used STK while working at Lincoln over the past two years, I also know that it is a fairly complicated tool as well. I was able to integrate it with some of my problem sets because of my previous knowledge; however, if I had not used the tool before, I would have found it nearly impossible to learn in the short amount of time provided to complete each homework. I strongly recommend giving students more opportunity to learn and use STK. If a full class was dedicated to teaching it, more of its functionality could be shown, including how it can be accessed through MATLAB, which is typically the most advantageous way to use it when designing a software tool. Additionally, it would be helpful if a professor or someone else (perhaps in a teaching assistant-type role) had a thorough understanding of STK and was available to help students use it on their homework. Even after using it regularly for two years, I am often on the phone with STK support (which is generally excellent) to get help on how to do something new, but having someone provide support at MIT would probably be easier for most students and make them more likely to try using it.

**b. Orbits**

The orbits section focused mainly on Geocentric orbits. Orbits are typically described by a set of six *orbital elements*:

- semi-major axis (a)
- eccentricity (e)
- inclination (i)
- true anomaly ( $\theta$ ,  $f$ , or  $v$ )
- right ascension of the ascending node ( $\Omega$ )
- argument of perigee ( $\omega$ )

Orbits are generally ellipses, and the value for eccentricity describes how elliptical they are. The point of closest approach to the planet being orbited is called the *perigee*, and the farthest point is called the *apogee*. The semi-major axis is the average of the apogee and perigee distances ( $r_a$  and  $r_p$ ):

$$a = (r_a + r_p)/2 \quad [1]$$

The perigee and apogee distances can also be defined by the semi-major axis and eccentricity:

$$r_a = a*(1 + e) \quad [2]$$

$$r_p = a*(1 - e) \quad [3]$$

The eccentricity vector is a vector which points from the center of the Earth to perigee, with magnitude equal to  $e$ . The true anomaly is the angle from the eccentricity vector to the spacecraft position. The right ascension of the ascending node is the angle from vernal equinox to the ascending node (the point where the spacecraft crosses the equatorial plane moving from south to north). The argument of perigee is the angle from the ascending node to the eccentricity vector.

Several special orbits were described in class. *Geosynchronous* orbits have a period equal to one sidereal day. A *Frozen* orbit, such as a *Molniya* orbit, is very eccentric and spends more time over a certain area of the globe. A *Sun-synchronous* orbit is designed using J2 effects to track the sun, so that its plane of orbit is always normal to the sun.

Transferring from one orbit to another is an important consideration in spacecraft design. There are several ways to make transfers, depending on the time allotted for the transfer, and the fuel available. A key equation for computing maneuvers is the equation which defines the velocity of a spacecraft in orbit:

$$v = \sqrt{2*\mu/r - \mu/a} \quad [4]$$

where  $\mu$  is the gravitational constant of the body being orbited,  $a$  is the semi-major axis of the orbit, and  $r$  is the location of the spacecraft in the orbit. For circular orbits, this reduces to:

$$v = \sqrt{\mu/a} \quad [5]$$

If only a plane change needs to be done, a *Simple Plane Change* can be used. In this case, the initial velocity and the final velocity of the spacecraft is the same. A change in velocity must be made in the direction normal to the current orbital plane to change the inclination. The change is defined as:

$$\Delta v = 2*v*\sin(\theta/2) \quad [6]$$

where  $v$  is the initial velocity, and  $\theta$  is the change in inclination.

If a change of altitude needs to be made, a *Hohmann Transfer*, *One-Tangent Burn*, or *Bi-Elliptic Transfer* can be made. When performing one of these transfers, a change in inclination or eccentricity can be made as well.

A Hohmann transfer (from a lower altitude orbit to a higher altitude orbit) is made by performing a tangential  $\Delta v$  at perigee of the initial orbit that will move it into a transfer orbit. The  $\Delta v$  should be calculated so that the apogee of the transfer orbit is also the apogee of the final orbit. Another tangential burn is done at this point to place the spacecraft in its final orbit. Eccentricity changes are accounted for if the correct above equation for velocity is used. If an inclination change needs to be made, it generally should be done at the highest altitude to reduce the amount of velocity needed for the change. The  $\Delta v$  for the inclination change is combined with the  $\Delta v$  for the orbit change, so that only one burn is made. However, a Hohmann transfer including a plane change can be more optimized by performing part of the inclination change at the lower altitude burn, and the rest at the higher. This method is known as the *Minimum Combined Plane Change*.

In a One-Tangent Burn, a tangential burn is also made at perigee of the initial orbit. This burn puts the spacecraft into a transfer orbit larger than the one that would be used for a Hohmann transfer. When the spacecraft reaches the altitude of the final orbit, it makes a second burn. This burn, however, is not at apogee, so it is not tangential. This transfer is quicker than the Hohmann transfer, but uses more  $\Delta v$ . Eccentricity and inclination changes could be accounted for as well, however, this topic was not investigated in class or in module development.

In a Bi-Elliptic Transfer, two Hohmann-type transfers are done in succession. The first transfer brings the spacecraft out to an orbit much higher than the final desired orbit. The spacecraft does a burn here to immediately enter another transfer orbit that brings it back down to the apogee of the final orbit. This transfer takes the longest; however, it uses less  $\Delta v$  than a regular Hohmann transfer in some cases. Inclination changes should be made at the highest altitude. Eccentricity changes are accounted for by using the above equation for velocity for eccentric orbits.

Another interesting orbit-related topic is perturbation effects. These were not covered in depth, but it would have been interesting to learn more about how perturbation effects

work, such as perturbations from the sun and moon on high altitude satellites, and how orbits are maintained in light of those disturbances.

Hill's equations are used for determining relative motion of satellites in orbit. They were briefly mentioned as part of lecture, and it also would have been interesting to discuss local reference frames in more detail, and how the Clohessy-Wiltshire equations can be used to determine how spacecraft can move relative to local frames (such as, how a spacecraft could move relative to another spacecraft). I have done some work with these equations and would have enjoyed hearing a thorough explanation of how and why they work, although perhaps that would have been too detailed considering the scope of the class.

References [1], [3], and [4] contributed to the above orbit discussion, and they additional information on topics such as Hill's equations.

*c. Power*

Power systems typically include solar cells and batteries. Solar arrays are sized based on average power needs of the satellite. Batteries are sized based on peak electrical power needed. The life of the mission, type of orbit, and satellite configuration all influence power system design.

The design of solar arrays was an interesting trade-off in homework problems. Solar arrays are sized depending on the power needs of the satellite, and the time the satellite is sunlit and in eclipse. The solar array must collect enough energy during sunlight to power the satellite during the entire orbit. The fraction of time a satellite is sunlit can be found by the following equations:

$$\text{Fraction sunlit} = (\pi + 2\alpha)/(2\pi) \quad [7]$$

$$\alpha = \sin^{-1}(R_E \tan \beta / (a \cos \theta)) \quad [8]$$

$$\beta = \cos^{-1}(R_E/a) \quad [9]$$

where  $R_E$  is the radius of the Earth, and  $a$  is the radius of the Earth plus the height of the satellite above the Earth. The angle  $\theta$  is the angle between the orbital plane and the sun. Alternatively, the amount of the time a satellite is sunlit can be found by modeling the orbit in STK.

The solar array power during daylight  $P_{sa}$  is found by the following equation:

$$P_{sa} = ((P_e T_e / X_e) + (P_d T_d / X_d)) / T_d \quad [10]$$

where  $P_e$  and  $P_d$  are the satellite's power needs during eclipse and daylight,  $T_e$  and  $T_d$  are the times per orbit the satellite is in daylight or eclipse, and  $X_e$  and  $X_d$  are the efficiencies of the paths from the solar arrays through the batteries to the loads, and the path directly from the arrays to the load. These values depend on the type of power regulation, either direct energy transfer or peak-power tracking.

The area of a solar array is found by dividing the solar array power  $P_{sa}$  by the power at the end of the satellite life,  $P_{EOL}$ :

$$A_{sa} = P_{sa}/P_{EOL} \quad [11]$$

The power at the end of life is equal to the power at the beginning of life,  $P_{BOL}$ , multiplied by the lifetime degradation  $L_d$  of the solar array. Degradation is caused by thermal cycling, micrometeoroids, plume impingement, material outgassing, and radiation. The power at the beginning of life is determined by the following equation:

$$P_{BOL} = P_o I_d \cos\theta \quad [12]$$

where  $P_o$  is ideal solar output performance per a unit area (based on the material used), and  $I_d$  is the inherent degradation, which is design inefficiencies in the array. The angle  $\theta$  is the sun incidence angle.

Different types of batteries are picked for different reasons, depending on their depth of discharge, lifetime, energy density, and other factors. Primary batteries are not rechargeable. They are typically used in short missions, and have a specific energy density of 90-400 Whr/kg. Secondary batteries are rechargeable, and are used for providing power during eclipse or during peak power loading. They typically have a specific energy density in the range of 25-200 Whr/kg. Primary battery types include silver zinc, lithium sulfur dioxide, lithium carbon monofluoride, and lithium thionyl chloride. Secondary battery types include silver zinc (again), nickel cadmium, and nickel hydrogen. A property used for choosing batteries includes a battery's depth of discharge (DOD). The DOD is the percent of the capacity used during a single discharge cycle. A higher DOD means a shorter cycle life.

The power subsystem focused on how to account for power needs of a spacecraft. How to determine what those power needs are was not really covered. Often for homework, the value of average power needed was chosen to be around 110 Watts, since that was used in the example for FireSat in the textbook, and the solar array design was based on

that value. Some other subsystems specifically mentioned what power was needed to drive them (ex. communications), but it would have been interesting to discuss in class the power needs of different satellite components, and how those varied needs specifically drive the design of the power system – such as what power sources (primary battery, secondary battery, solar array, other) best suit what specific needs.

References [1], [5], [6], and [7] contributed to the above discussion.

**d. Environment**

Environmental effects on a satellite include the solar cycle, the Earth's gravity and atmosphere, aerodynamic drag, the magnetic field, plasma effects, and radiation. How each of these factors affects a satellite, and methods to compensate for the effects, were discussed.

Different environmental factors affect space systems in different ways. For example, Sunlight and Earthshine effect heating and choice of surface materials of satellites. The materials are chosen based on their thermal properties, such as absorptance and emissivity. Solar energy may also degrade materials. The Earth's gravity affects the actual orbit of a satellite, as well as its alignment. Gravity gradient torques are applied to the satellite causing it to align in a particular direction that may or may not be desirable. The Earth's magnetic field also causes torques; both the affect of this and gravity are dependent on the satellite's distance from Earth. The magnetic field intensity  $B$  is found by the following equation:

$$B = (1 + \sin^2\lambda)^{1/2} B_0/R^3 \quad [13]$$

where  $\lambda$  is the magnetic latitude,  $R$  is the radial distance, and  $B_0$  is the magnetic field at the equator at the Earth's surface. The torque resulting from this field can be found by multiplying  $B$  by the satellite's residual dipole. The torque due to gravity  $T_g$  is found by the following equation:

$$T_g = (3\mu/2R^3)|I_z - I_y|\sin(2\theta) \quad [14]$$

where  $\mu$  is the Earth's gravity constant,  $R$  is the orbit's radius,  $\theta$  is the maximum deviation of the Z-axis from local vertical, and  $I_z$  and  $I_y$  are moments of inertia about z and y axes.

In terms of the textbook's discussion of the space environment, the environment subsystem (if it can even be called that) is coupled tightly with the attitude control system. Many of the torques produced on the satellite from environmental factors can be counteracted through various attitude control techniques (discussed more in the next section).

There is certainly a large variety of environmental issues that must be accounted for in satellite design, and which factors are important varies widely depending on the desired orbit of the satellite. From what I have seen at work, I am under the impression that models of many of the environmental effects are even much more complicated than what was presented in class and in the textbook. Many of the textbook equations, when used in homework, provided a simple way to find the worst-case behavior. However, it would be interesting to learn about a high-fidelity model for one or two of the environmental effects, and to examine that effect on a satellite over the course of an orbit or even a lifetime.

References [1], [8], and [9] contributed to the above discussion.

*e. Attitude Control System*

The Attitude Control System (ACS) consists of methods to keep the satellite in the correct position. The satellite may need to slew to face a particular direction, or just overcome environmental torques. Attitude coordinate systems aid in calculating a satellite's position with respect to a local frame, and rotations are used to convert between different coordinate frames. There are different types of ACS actuators, such as reaction wheels, control moment gyros, magnetic torquers, and thrusters. Additionally, sensors such as star trackers and sun sensors are used to help the satellite determine its attitude.

As mentioned in the previous section, the textbook seemed to focus on how ACS actuators are used to counteract environmental effects. For example, momentum wheels can be chosen to counter-act the effects of the gravity gradient torque  $T$ . The amount of momentum  $h$  the wheel must be able to store is calculated based on this value, and based on the assumption that the disturbance due to gravity gradient accumulates in  $\frac{1}{4}$  of each orbit:

$$h = TP(0.707)/4 \quad [15]$$

where  $P$  is the orbital period. Momentum wheels themselves are largely available commercially; once the size needed is known, they may be chosen based on other parameters, such as size or mass (see *References* for data sheets on commercial momentum wheels). Thrusters must be used to dump momentum; the amount of force  $F$  needed to dump momentum is found by the equation:

$$F = h/Lt \quad [16]$$

where  $L$  is the moment arm and  $t$  is the burn time. The actual amount of propellant needed for the thrusters is based on this as well as force needed for any large maneuvers the satellite must make.

ACS systems often have other jobs as well. These may include rotating solar arrays, controlling where antennas point, and recovering from failure conditions. Also, they may use sun and earth telemetry data found by the sensors to help determine attitude and position.

A satellite's attitude is often expressed as a quaternion, which describes the attitude of an object's body frame with respect to a reference frame. A quaternion has four elements, three that are part of a vector and one that is a scalar:

$$\mathbf{q} = q_1i + q_2j + q_3k + q_4 \quad [17]$$

A quaternion expresses the rotation about an axis  $n$  and an angle  $\theta$ . It can physically be interpreted as:

$$\mathbf{q} = [ \quad n\sin(\theta/2) \quad \cos(\theta/2) \quad ]^T \quad [18]$$

A quaternion can be propagated over time if the satellite has a rotation rate  $\omega$ . Additionally, a vector can be rotated from the reference frame to the body frame by using a quaternion matrix  $\Theta$ :

$$\Theta(\mathbf{q}) = (q_4^2 - \mathbf{q}^T\mathbf{q})\mathbf{I}_{3 \times 3} + 2\mathbf{q}\mathbf{q}^T - 2q_4[\mathbf{q}\mathbf{x}] \quad [19]$$

where  $\mathbf{I}_{3 \times 3}$  is the identity matrix. If quaternion rotations are known between different frames, such as arbitrary local references frames on satellites and Earth centered frames, then a satellite's attitude can be found with respect to any of those frames.

The ACS information was interesting, but I would have enjoyed it more if it had a slightly different focus and went a little more in-depth in some parts of the subject. As mentioned in the section on orbits, I am interested in how satellites move in local frames. Tied to this is how they orient themselves in these frames to keep a desired position. I have done some work with quaternions, but never actually been taught them, and was

hoping that they would be a part of the ACS presentation, instead of just a handout. Also, it would have been interesting to look a bit more at exactly how sensors can determine attitude movements. A good example of these issues is a satellite attempting to dock with another; if it is dependent on a sensor to track the satellite it is docking with, it must maintain an attitude so that its sensor can get detections, and may perform the docking maneuver in a local frame.

References used in this section include [1], [10], and [11] to [21]

*f. Communications*

The communications subsystem consists of how to design links between a satellite and the ground. A digital satellite link is governed by the link equation:

$$E_b/N_o = (PL_lG_tL_sL_aG_r)/(kT_sR) \quad [20]$$

where  $E_b/N_o$  is the ratio of received energy-per-bit to noise density,  $P$  is the transmitter power,  $L_l$  is the transmitter to antenna line loss,  $G_t$  is the transmit antenna gain,  $L_s$  is the space loss,  $L_a$  is the transmission path loss,  $G_r$  is the receive antenna gain,  $k$  is Boltzmann's constant,  $T_s$  is the system noise temperature, and  $R$  is the data rate.

The power received  $C$  at a ground antenna is:

$$C = (PL_lG_tL_aD_r^2\eta)/(16S^2) \quad [21]$$

where  $P$ ,  $L_l$ ,  $G_t$ , and  $L_a$  are as described above,  $D_r$  is the radius of the receive antenna,  $\eta$  is the antenna efficiency, and  $S$  is the distance from transmitter to receiver. The gain of the receive antenna  $G_r$  is then:

$$G_r = (\pi^2D_r^2\eta)/\lambda^2 \quad [22]$$

here  $\lambda$  is the wavelength of the transmitted signal.

The data rate  $R$  is a factor in determining how much data can be transmitted during communication. The total amount of data  $D$  a satellite can transmit to a ground station is:

$$D = R(FT_{max} - T_{initiate})/M \quad [23]$$

where  $T_{max}$  is the maximum time the satellite would be in view (that is, the satellite pass duration for a pass directly overhead),  $T_{initiate}$  is the time required to initiate a

communications pass,  $M$  is the margin used to account for missed passes, and  $F$  is the fractional reduction in viewing time due to a satellite passing at an angle away from the ground station (not directly overhead). Individual passes can be examined if a communications system is modeled in STK. STK can calculate the access time between a satellite and a ground station, which helps determine the actual amount of time a satellite is in range of a ground station during each pass.

Transmitting digital data has unique problems. Digital data is a series of bits of sampled analog data; not all of the information is included but enough to reconstruct the original waveform (if it is sampled at a rate of at least 2.2 times the original frequency). Often errors occur in bits; additional data is often sent, such as in Viterbi Decoding, to account for these errors. The data is also typically compressed, or encoded, so that not as much data needs to be sent. Encoding methods include Huffman encoding, Run Length encoding, and Differential Pulse Code Modulation. Some methods of compression are *lossless*: when decompressed, all original data will be recovered. Other methods are *lossy*: all original data may not be recovered.

The text included many of the link equations, as well as how to design telescopes. The chapter was very technical, and did not explain everything fully. It seemed that actual telescope design should have been separated from the discussion of how communication systems work.

I found the communications subsystem to be very confusing. It seemed to touch on many subjects, from how to calculate what data can be transmitted, to how to design a telescope, to general schemes for data encoding and compression. I think it would have made more sense to give a high-level overview of how communications systems are used in satellite systems – and this would have been more along the lines of how the other topics were covered. Although I am very interested in signal processing and methods for data encoding, it seems that information is very detailed considering the scope of the class. More basic communication needs could have been mentioned, such as when and why communication is done, and realistically how much data and how often

communication can be made. I attempted to find more basic information to supplement this section, but had trouble finding any information with the right balance between technical and general information, so I realize this topic is complicated and difficult to cover in a short period of time.

References used in this section include [1], [22] and [23].

***g. Software and Automation***

Software is a complicated and largely debated topic in safety- and mission-critical systems in general and in satellite systems in particular. Colonel Keese's lecture focused on the different parts of computer systems, and how safety features such as fault tolerance are handled. Kathryn Weiss discussed why software is hard to design, and methods to possibly make it easier. Seung Chung explained ways to safely add autonomy in space systems.

The textbook focused on how to estimate computer needs and choose computer components, and did not really discuss issues concerning actual software design.

Designing software is a much more difficult than most people realize. Different languages are applicable in different situations, but often choices are made based more on trends. Adding to the problem is the variety of different backgrounds and experience levels many software engineers on the same project may have, and the many COTS components available.

One approach to improving software is to improve how requirements and specifications are done. Studies have shown that many perceived software errors can actually be traced to errors in specifications. These are often errors of omission (such as forgetting to specify behavior for a certain error state) or not correctly specifying how COTS or older version software fits with a design. While many of these errors do indicate systemic issues as well (such as being complacent about safety or just not having any one group in charge of making sure all necessary issues are covered), using a formal, executable specification language can aid in creating a stable, 'safe' software system. One such method is called *Intent Specifications*. Intent Specifications is a way of specifying a system design (software or otherwise) that is executable (so that behavior can be checked at the specifications level). It was designed using cognitive psychology methods, so that the program is easy to use and understand.

An alternative approach is to reduce the scope of the software system, and to provide many usable methods of formally analyzing and testing software. The Gurkh project (part of which I'm working on for my thesis) aims to provide users with a complete development environment for real-time safety-critical systems. It supports the Ravenscar subset of Ada. Ada has been shown in studies and through use to provide cheaper, safer, and more robust software systems. Using subsets of Ada, such as Ravenscar, has been a key part in this; the subsets restrict the language in such a way that the software built becomes fully deterministic and predictable. A system with these properties can be more thoroughly analyzed. The Gurkh project will provide a custom run-time kernel in VHDL (also more deterministic than a traditional software run-time kernel), tools for converting VHDL and Ada into formal models that can be analyzed with model checkers, and a VHDL component that monitors the system while its running to check for any missed deadlines. Future extensions to the project include adding a way to model specifications, perhaps by using Intent Specifications or a different method, and then automatically generating Ada code from the specifications. When complete, the Gurkh project will enable users to create a deterministic software system that they can analyze with a variety of tools, and humans will be removed from many stages of the design process, thereby reducing errors.

References used in this section include [1], [24], [25], [26], [27], and [28].

### III. References

#### General:

[1] Wertz, Larson, Space Mission Analysis and Design, 3<sup>rd</sup> Edition, Space Technology Library, 1999.

#### STK:

[2] [www.stk.com](http://www.stk.com)

#### Orbits:

[3] Vallado, Fundamentals of Astrodynamics and Applications, 2<sup>nd</sup> Edition, Space Technology Library, 2001.

[4] Battin, An Introduction to the Mathematics and Methods of Astrodynamics, Revised Edition, AIAA Education Series, 1999.

#### Power:

[5] Jones, Spence, *Spacecraft Solar Array Technology Trends*, [www.aec-able.com/corpinfo/Resources/PAJ-IEEE-98.pdf](http://www.aec-able.com/corpinfo/Resources/PAJ-IEEE-98.pdf)

[6] [www.qrg.northwestern.edu/projects/vss/docs/Power/zoom-possible-powers.html](http://www.qrg.northwestern.edu/projects/vss/docs/Power/zoom-possible-powers.html)

[7] *Spacecraft Power Systems*, Virginia Tech, [mev.btg.cc/BTG-Library/files/space%20power.pdf](http://mev.btg.cc/BTG-Library/files/space%20power.pdf)

#### Environment:

[8] [www.sec.noaa.gov/primer/primer.html](http://www.sec.noaa.gov/primer/primer.html)

[9] [powerweb.grc.nasa.gov/pvsee/publications/seeov/seeov.html](http://powerweb.grc.nasa.gov/pvsee/publications/seeov/seeov.html)

#### ACS:

[10] <http://msp.gsfc.nasa.gov/tdrss/attitude.html>

[11] Ahronovich, Balling, “Reaction Wheel and Drive Electronics For LeoStar Class Space Vehicles”, 12<sup>th</sup> Annual USU Conference on Small Satellites, 1998, [www.sdl.usu.edu/conferences/smallsat/proceedings/12/ssc98/1/ssci5.pdf](http://www.sdl.usu.edu/conferences/smallsat/proceedings/12/ssc98/1/ssci5.pdf)

[12] Dynacon Enterprises Limited, “Dynacon MicroWheel 200”, [www.dynacon.ca/pdf/files/productpdf\\_6.pdf](http://www.dynacon.ca/pdf/files/productpdf_6.pdf)

[13] Honeywell Aerospace Electronic Systems, “Constellation Series Reaction Wheels”, [http://content.honeywell.com/dses/assets/datasheets/constellation\\_series\\_reaction\\_wheels.pdf](http://content.honeywell.com/dses/assets/datasheets/constellation_series_reaction_wheels.pdf)

[14] Honeywell Aerospace Electronic Systems, “Miniature Reaction Wheels”, [http://content.honeywell.com/dses/assets/datasheets/mini-wheel\\_reaction\\_wheel.pdf](http://content.honeywell.com/dses/assets/datasheets/mini-wheel_reaction_wheel.pdf)

[15] Honeywell Aerospace Electronic Systems, “Honeywell Model HR 0610 Reaction Wheel”, [http://content.honeywell.com/dses/assets/datasheets/hr0610\\_reaction\\_wheel.pdf](http://content.honeywell.com/dses/assets/datasheets/hr0610_reaction_wheel.pdf)

[16] Teldix Space Product Group, “Momentum and Reaction Wheels 14-68 Nms with external Wheel Drive Electronics”, <http://www.teldix.de/P22/RDR23-68.pdf>

[17] Teldix Space Product Group, “Momentum and Reaction Wheels 14-68 Nms with integrated Wheel Drive Electronics”, <http://www.teldix.de/P22/RSI25-68.pdf>

[18] Teldix Space Product Group, “Momentum and Reaction Wheels 2-12 Nms with integrated Wheel Drive Electronics”, <http://www.teldix.de/P22/RSI4-12.pdf>

[19] Teldix Space Product Group, “High motor torque Momentum and Reaction Wheels 14-68 Nms with integrated Wheel Drive Electronics”, <http://www.teldix.de/P22/RSI18-68.pdf>

[20] Teldix Space Product Group, “Momentum and Reaction Wheels 0.04-0.12 Nms with integrated Wheel Drive Electronics”, <http://www.teldix.de/P22/RSI01.pdf>

[21] Teldix Space Product Group, “Momentum and Reaction Wheels 0.2-1.6 Nms with integrated Wheel Drive Electronics”, <http://www.teldix.de/P22/RSI02.pdf>

### **Communications:**

[22] <http://ctd.grc.nasa.gov/rleonard/regcontents.html>

[23] <http://www.wtec.org/loyola/satcom/toc.htm>

### **Software:**

[24] Leveson, “Systemic Factors in Software-Related Spacecraft Accidents”, AIAA Space 2001 Conference.

[25] Leveson, “Intent Specifications: An Approach to Building Human-Centered Specifications”, Software Engineering, Volume 26, Number 1, 2000, pp 15-35.

[26] Weiss, “Component Based Systems Engineering for Autonomous Spacecraft”, MIT Masters Thesis, 2003.

[27] Asplund, Lundqvist, “The Gurkh Project: A Framework for Verification and Execution of Mission Critical Applications”, 22<sup>nd</sup> Digital Avionics Systems Conference, 2003.

[28] Silbovitz, Lundqvist, “A Hardware Implementation of a Ravenscar-Compliant Run-Time Kernel”, 22<sup>nd</sup> Digital Avionics Systems Conference, 2003.

**The following references were not mentioned in the previous chapters, but still provided valuable information in completing problem sets:**

[www.astronautix.com](http://www.astronautix.com)

Touloukian, Ho, *Thermophysical Properties of Selected Aerospace Materials Part I: Thermal Radiative Properties*, Purdue Research, West Lafayette IN, 1976

Boyer, Gall, *Metals Handbook*, American Society for Metals, Metals Park OH, 1985

Metals Reference Book, American Society for Metals 2<sup>nd</sup> Edition, 1983

Sala, *Radiative Properties of Materials*, Elsevier Science Publishers, New York, 1986

Metals Handbook: Properties and Selection of Nonferrous Alloys and Pure Metals, 9<sup>th</sup> Edition Vol. 2, American Society for Metals, 1979

Holman, Heat Transfer, McGraw-Hill, Inc, New York, 1997

Miller, *Lecture Notes: Cost Modeling*, Massachusetts Institute of Technology, 16.851, 2003

Wertz, Larson, Reducing Space Mission Cost, Space Technology Series, Space Technology Library, Microcosm Press, Torrance CA, 1996.

[www.me.poly.edu/hk/subject/me262/notes/chapt9.pdf](http://www.me.poly.edu/hk/subject/me262/notes/chapt9.pdf)