Design Principles for the Development of Space Technology Maturation Laboratories Aboard the International Space Station

by

Alvar Saenz-Otero

S.B., Aeronautics and Astronautics, MIT, 1998
S.B., Electrical Engineering and Computer Science, MIT, 2000
MEng, Electrical Engineering and Computer Science, MIT, 2000

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in Aeronautics and Astronautics at the
Massachusetts Institute of Technology
June 2005

Author . . . . . .
Department of Aeronautics and Astronautics

Certified by . . . . . .
David W. Miller, Associate Professor of Aeronautics and Astronautics
Director, MIT Space Systems Laboratory, Thesis Supervisor

Certified by . . . . . .
Eric Feron, Associate Professor of Aeronautics and Astronautics
Thesis Committee Member

Certified by . . . . . .
Jonathan P. How, Associate Professor of Aeronautics and Astronautics
Thesis Committee Member

Certified by . . . . . .
Brian C. Williams, Associate Professor of Aeronautics and Astronautics
Thesis Committee Member, Minor Advisor

Certified by . . . . . .
Javier de Luis, CEO Payload Systems Inc.
Thesis Committee Member

Accepted by . . . . . .
Jaime Peraire, Professor of Aeronautics and Astronautics
Chair, Committee on Graduate Students

Design Principles for the Development of Space Technology Maturation Laboratories Aboard the International Space Station

by

Alvar Saenz-Otero

Submitted to the Department of Aeronautics and
Astronautics on June 2005 in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in Aeronautics and Astronautics

ABSTRACT

This thesis formulates seven design principles for the development of laboratories which utilize the International Space Station (ISS) to demonstrate the maturation of space technologies. The principles are derived from the lessons learned from more than two decades of space technology research at the MIT Space Systems Laboratory and the existence of unique resources aboard the ISS. The thesis provides scientists with a design framework for new laboratories and an evaluation framework to responds to a call by the National Research Council to institutionalize science activities aboard the ISS.

Experience from previous missions and research on the resources available at the ISS led to the development of the SPHERES Laboratory for Distributed Satellite Systems (DSS), which constitutes the experimental part of the thesis. SPHERES allows tests in a representative, risk-tolerant environment aboard the ISS to demonstrate metrology, control, and autonomy algorithms for DSS. The implementation of ground-based and ISS-based facilities permits incremental technology maturation by enabling iterative research; algorithms can mature through multiple research cycles with increasing complexity. The SPHERES Guest Scientist Program supports research by multiple scientists: since the Spring of 2000 SPHERES has enabled research on formation flight, communications requirements, mass properties identification, autonomous rendezvous and docking, and tethered formation flight.

The design principles were formulated by first identifying the features of the SPHERES laboratory which allow it to fulfill the MIT SSL Laboratory Design Philosophy and utilize the ISS correctly, and then finding the applicability of these features to space technology maturation research. The seven principles are: *Principle of Iterative Research*, *Principle of Enabling a Field of Study, Principle of Optimized Utilization, Principle of Focused Modularity, Principle of Remote Operations and Usability, Principle of Incremental Technology Maturation*, and *Principle of Requirements Balance*. The design framework is used to assess SPHERES and suggest a new design iteration which better satisfies the design principles. The evaluation of SPHERES concludes that it is ready for operations aboard the ISS, since the modular design of SPHERES allows most of the proposed design changes to occur after the initial deployment.

Thesis Supervisor: David W. Miller
Associate Professor of Aeronautics and Astronautics
Director, MIT Space Systems Laboratory

Dedicated in loving memory of

Dedicado con amor a la memoria de

Fernando Saenz

1945-2005

ACKNOWLEDGEMENTS

This thesis represents not only my work at the keyboard, it is a milestone in more than one decade of work at MIT and specifically within the Space Systems Laboratory. My experience at MIT has been nothing short of amazing. Since my first day during RO on August 16th, 1994 I have felt at home at MIT. I have been given unique opportunities... and taken advantage of them. This includes working at the SSL for over ten years, starting as an undergraduate research assistant in the Spring of 1996. Throughout these years I have learned that there are those who build tools and those who use them; my passion is in creating the tools used in cutting edge research. This thesis presents the lessons learned in creating one of those special tools: SPHERES. SPHERES is the result of work by dozens of people, who I wish to thank. But this thesis is also the result of many experiences I have encountered at MIT from dozens of remarkable individuals who I also wish to acknowledge.

First and foremost I wish to thank my advisor, professor **Dave Miller**, director of the MIT Space Systems Laboratory. He has been supportive since the days I began working on the Origins testbed as an undergraduate; I remember he used to say something like "you're always the first one in and the last one out working on that project!" to encourage me to stay in the lab. Ever since, Dave has supported me not only by providing a research assistantship over almost seven years, but also academically and emotionally through the rough road to finish this thesis. Thanks to him I had the opportunity to build SPHERES... and test it in the KC-135! He helped me come up with the thesis topic and guided me over almost a year of development. And during the most difficult times when writing this thesis, he gave me the moral support and the freedom I needed to move on.

My thesis committee guided me through all these years. Thank you to **Jonathan How, Eric Feron, Brian Williams, and Javier deLuis** for being my major advisors. Jon and Brian supported SPHERES with several students; Eric guided me through my undergraduate capstone class; and Javier has supported SPHERES and my research since 1999. Also thank you to **Jeff Hoffman** for being part of my Generals committee, a thesis reader, and an inspiration in many ways. And thank you to **Dava Newman**, with whom I worked with as associate advisor for two years during my undergraduate career and whom became one of my thesis readers.

Dozens of people have helped and taught me immensely at the MIT SSL. *Mom* **Sharon** welcomed me to the lab my sophomore year in 1996. She has been a source of love and energy ever since. Dave's assistants, **Peggy** and **Marilyn** have been amazing help all this time. And then, there's the students before me, mentors in many ways. My UROP supervisors **Graeme Shaw** in 1996 and **Greg Mallory** in 1997-98, with whom my 'permanent' stay at the SSL began. **Cyrus Jilla** who has been an inspiration on how to make things 'perfect'. A special acknowledgement goes to my office mate of many years: **Laila Elias**.

She was a true friend ever since we began to share an office in 1999. Laila is an amazing person in too many ways. And I also thank **Bekcy** and **Scott** (also Aero/Astro class of '98!), my other great office mates who have been supportive in every way. Also thank you to **Rick** and **Luke** for three years of trips to the mountains.

A special group from the SSL is not mentioned yet, because they deserve their own part: the *SPHERES team*. I praise the enormous amount of help and teaching by **Paul Bauer** throughout these years. Thank you to those who helped the project as staff and graduate students: **Ray Sedwick, Edmund Kong, Mark Hilstad** (and for teaching me to ski!)**, John Enright (**and **Sarah**), **Alice Liu, Simon Nolet, Dustin Berkovitz, Arthur Richards, Al Chen, Soon-Jo Chung, and Ryan Lim.** Also thank you to everyone at Payload Systems who has helped with the project and supported me in many ways: **Javier de Luis, Steve Sell, Stephanie Chen, Bob Grimes, Edison Guerra, John Merk, Chris Rockhold** and the marvelous **Joanne Vining.** I also recognize the support and help from the ***Air Force STP*** and ***DARPA*** offices for their support of the SPHERES program. And, for providing the experience of a lifetime, thank you to **John Yaniec** and the ***NASA JSC RGO*** for their support of five SPHERES reduced gravity campaigns. Lastly, I wish to acknowledge the thirteen students who taught me how to teach a lab class and who built the prototype SPHERES: *George Berkowski, Chad Brodel, Sarah Carlson, Dave Carpenter, Stephanie Chen, Allen Chen, Shannon Cheng, Daniel Feller, Stuart Jackson, Fernando Perez, Bradley Pitts, Jason Szuminski*, and *Julie Wertz*.

I also thank an organization: the ***Tech Model Railroad Club***. Thanks to the hands-on work in building things at the club I got my second UROP at the SSL! I want to especially thank **Jeff Birkner** for teaching me about electrical things; about the professionalism needed when creating new things; and most importantly for becoming a lifelong friend.

I took part of many other activities, with countless people teaching and helping me every step. Thank you to everyone from ***4E***, especially **Atif, Rex, Chris,** and **Ann**. Your friendship was irreplaceable. Thank you to everyone at ***LSC*** for many years of movies and popcorn. Thank you to the ***Sailing Pavilion*** coaches and staff who made every summer in grad school remarkable. Thank you to everyone at the ***ASA*** and the ***DSL/SAO***, especially **Tracy Purinton** and **Laurie Ward**. Also thank you to everyone at the ***GSC***, of whom **Emmi Snyder** stands out notoriously as an amazing source of support. I also thank the members of the ***Career Fair*** and the ***GSC GradRat Committee*** who gave me years of fun while trying to write the thesis! And, behind all of these activities, are two people whom worked with me and many others throughout the way and I wish to thank for their support and inspiration: **Dean Larry Benedict** and **Dean Ike Colbert**.

I was also involved in activities outside MIT, where I met amazing people while showcasing Mexican Culture in the Boston Area. Thanks to everyone at the ***Mexican Consulate*** who has supported the community through good and bad times. And thank you to the many people from ***FOMNI*** and other organizations who have worked with me over the years in dozens of cultural events.

An talking about Mexico: I have special friends to thank, going back to my pre-MIT days. Thank you to the *cuatro diablitos:* **Javier Garcia-Terruel, Roberto Montañez, David Thomas**... and, I have to especially thank **Ashley Viberg,** whom I consider my best friend since high school. He taught me to sail... and between him and his parents they taught me better english! His unconditional support has been essential all these years. I thank his parents, **Gordon** and **Dale,** for it was them who suggested I should come to MIT while on a trip to Valle in 1993; I am not sure I would be at MIT without their encouragement. I also thank a special teacher in high-school, **Mrs. Jenka Guevara**. Thanks to her I could enjoy learning at the ASF, even after school! And thanks to **Ceani** for her continued friendship through high school and even at MIT. Lastly, I wish to acknowledge a friendship that began at the ASF and became ripe throughout my years at MIT: thank you to **Kevin Tangney** and his family. Since the first year I arrived at MIT (for eleven years now!) the ***Tangney's*** have welcomed me at their home for Thanksgiving, a tradition that I hope will continue for decades to come! Plus, I can't forget about the many sailing trips.

I finish with Mexico, where the most basic source of my life energy resides: my family. I have an amazing family, unique in many ways, and the stereotype of a *perfect* family in many others. Their support has been unconditional all these years; they have given up many things for me to be at MIT; they have cherished with me every great moment and supported me whenever I needed it. And I need to thank them in spanish now...

*¡GRACIAS a toda la familia! A la **Abuelita** lunática por prestarme su cerebro, a **Lucerito** por su amor inmenso**,** y a mi tía y madrina **Fer** que me apoya y ama inmensamente. ¡Su amor no cabe en ningún lado! Gracias a **Rocio** y **Patiño** por querer tanto a mis papas y a mi. Gracias a **Tío Luis** por ser mi tío de MIT, a tía **Margarita** por siempre apoyar a mis papas, y a todos mis **primos** por siempre recibirme en México con mucha alegría. Gracias a **Mane** por apoyar a mi familia tantos años; a **Sofi** y **Alex** por su amor hacia nosotros desde siempre; y a **Gabi** y familia por siempre estar con nosotros.*

*Y en muy especial, gracias a mi papá, mi mamá, y mi hermano. Esta tesis es de la familia **Saenz Otero**. Es una combinación de todos los conocimientos que me dio **papá**, desde los Legos, los viajes a la NASA, las computadoras, la carpintería, los trenes, y más. Todo esto unido con el amor y la personalidad maravillosa que tiene mi **mamá**. Y gracias a **Edgar**, que aguanto tantos años solito con mis papas para que yo pudiera seguir en MIT siempre tranquilo.*

*GRACIAS **FERNANDO SAENZ, ARACELI OTERO, Y EDGAR SAENZ OTERO.***

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYM LIST

#D	# dimensions
AFRL	Air Force Research Laboratories
ANSI	American National Standards Institute
API	Application Programming Interface
ARD	Autonomous Rendezvous and Docking
ASCII	American Standard Code for Information Interchange
CDIO	Conceive, Design, Implement, Operate
CDR	Critical Design Review
CLK	Hardware timer driven HWI
COTS	Commercial Off The Shelf
CRC	Cyclic Redundancy Codes (Checksum)
CSA	Canadian Space Agency
DARPA	Defense Advanced Research Projects Agency
DLS	Dynamic Load Sensors experiment
DoD	Department of Defense
DOE	Design of Experiments
DOF	Degree of Freedom
DR2000	RFM Monolithics transceiver module
DSP	Digital Signal Processor
DSP/BIOS	Texas Instruments Real-time Operating System Kernel
DSS	Distributed Satellite System
ESA	European Space Agency
FDIR	Fault Detection, Identification, and Recovery
FLASH	Type of electrically erasable programmable read only memory (EEPROM)
FPGA	Field Programmable Gate Array
GFLOPS	Giga FLOPS (Billion Floating Point Operations Per Second)
GIPS	Giga (Billion) Instructions Per Second
GNC	Guidance, Navigation, and Control
GPS	Global Positioning System
GSP	Guest Scientist Program
GUI	Graphical User Interface
HWI	Hardware Interrupt
IDE	Integrated Development Environment
IMU	Inertial Measurement Unit
IR	Infrared
ISS	International Space Station
JPL	Jet Propulsion Laboratory
JSC	Johnson Spaceflight Center
KC-135	NASA’s reduced gravity airplane
LEO	Low Earth Orbit

LIB	Larger is Best
MACE	Middeck Active Control Experiment
MBX	Mailbox data transfer construct
MCU	Micro-Controller Unit
MFLOPS	Million Floating Point Operations Per Second
MIT	Massachusetts Institute of Technology
MLE	Middeck Locker Equivalent
MODE	Middeck 0-gravity Dynamics Experiment
MOSR	Mars Orbit Sample Return
MSFC	Marshall Space Flight Center
NASA	National Aeronautics and Space Administration
NASDA	Japanese National Space Development Agency
NGO	Non-Governmental Organization
NIB	Nominal is Better
NMP	New Millennium Program
NRC	National Research Council
PADS	Position and Attitude Determination System (Metrology system)
PC	Personal Computer
PIP	Pipe data transfer construct
PIC	Microchip PIC line of micro controllers
PRD	Periodic SWI
PSI	Payload Systems Inc.
RAM	Random Access Memory
RF	Radio Frequency (wireless communications)
RGA	Reduced gravity airplane
RGO	Reduced gravity office at NASA JSC
RGP	Reduced gravity program
RSA	Russian Space Agency
RTOS	Real-Time Operating System
RX	Receive / Receiver
SCAR	Special Committee for Antarctic Research
SCS	SPHERES Core Services software layer
SEM	Semaphore synchronization construct
SIB	Smaller is Best
SMT375	Sundance Multiprocessor Technologies DSP board model 375
SOH	State of Health
SPECS	Sub-millimeter Probe of the Evolution of Cosmic Structures
SPHERES	Synchronized Position Hold Engage Re-orient Experimental Satellites
SSC	Station Support Computer (ISS)
SSL	Space Systems Laboratory
SSP	Space Shuttle Program
STG	Satellite-to-ground
STL	Satellite-to-laptop
STP	DoD Space Technology Programs Office
STS	Satellite-to-satellite

STS-#	Space Shuttle Mission #
SWI	Software Interrupt
TDMA	Time Division Multiple Access
TPF	Terrestrial Planet Finder
TRL	Technology Readiness Level
TSK	Background Task (periodic or aperiodic)
TX	Transmit / Transmitter
US	Ultrasound
UART	Universal Asynchronous Receiver Transmitter

NOTATION

°	degree
°C	degrees Celsius
°F	degrees Fahrenheit
A	Ampere
b	bit
B	byte (8 bits)
cm	centimeter
deg	degree
F	Farad
ft	feet
g	gravity constant ($9.8 m/s^2$)
h	hour
Hz	frequency in hertz (1/s)
in	inch
kbps	kilo bits per second
kg	kilogram (mass)
Mbps	mega bits per second
MBps	mega bytes per second
μ-g	microgravity
μg	micro-g: one millionth of the gravity constant ($9.8 m/s^2 \times 10^{-6}$)
μs	microsecond
Ω	Ohm
m	meter
min	minute
mo	month
ml	milliliters
mg	milli-g: one thousand of the gravity constant ($9.8 m/s^2 \times 10^{-3}$)
ms	millisecond
N	Newton
nm	nanometer
psi	pounds per square inch
rad	radian
s	second
V	Volt
y	year
W	Watt

Chapter 1

INTRODUCTION

This thesis utilizes the lessons learned from the development of the SPHERES experiment and other MIT Space Systems Laboratory (SSL) projects to define a set of design principles for developing facilities to conduct space technology research in the International Space Station (ISS). The thesis follows the standard scientific process to define the principles. The objective of the thesis is to create a design methodology for the development of microgravity laboratories which allows the maturation of space technologies. The objective is motivated from the lessons learned by the MIT SSL during the design and operation of multiple space-based experiments and by a call by the National Aeronautics and Space Administration (NASA) and the National Research Council (NRC) to define how to institutionalize research aboard the International Space Station (ISS). The thesis objectives address the use of the ISS in two ways: the need of multiple researchers to access microgravity conditions to cost-effectively mature technologies and to make the best possible use of ISS resources. The hypothesis rests on the use of the MIT SSL Laboratory Design Philosophy, which consists of a set of features desired from a laboratory identified through the review of past experiences at the MIT SSL, and the correct utilization of existing resources to mature space technology. The hypothesis states that by using this laboratory design philosophy to develop projects to operate aboard the ISS, the resulting laboratory environment facilitates the maturation of space technology in an ideal environment. The SPHERES facility constitutes the experimentation. Based on the lessons learned from building SPHERES, the laboratory design philosophy and the knowledge of the ISS envi-

ronment were condensed into a set of design principles that characterize successful laboratory environments. Frameworks to apply the principles both at the design and evaluation phases complete the results. The conclusions identify the ability of the principles to meet the objective by analyzing the success of SPHERES as well as other experiments already aboard the ISS. Figure 1.1 summarizes these steps of the scientific process (objective, hypothesis, experimentation, results, and conclusion) as they are addressed in the thesis.

Objective:
Create a design methodology for the development of microgravity laboratories for the research and maturation of space technologies.

Hypothesis:
The conjunction of the International Space Station as a host and the MIT SSL Laboratory Design Philosophy as the design guidelines enable the development of a low-cost environment for the development and operation of facilities to conduct space technology research.

Experimentation:
The SPHERES laboratory for distributed satellite systems has been developed following the MIT SSL Design Philosophy for microgravity operations specifically aboard the ISS.

Results:
The MIT SSL Design Philosophy and research on the characteristics and operations of the ISS are condensed into a set of Design Principles that define the proper design of a research laboratory for the ISS.

Conclusion:
While the availability of the ISS has not proved as efficient as originally desired, the Design Principles and corresponding frameworks do create a valid methodology for the development of microgravity research facilities which reduce both the cost and risk of maturating space technologies. Further, by following of these principles can allow facilities to benefit the research community even if not all operational environments are available.

Figure 1.1 Thesis research process

1.1 Motivation

Precision space systems are becoming increasingly difficult to fully test prior to launch. New mission architectures continuously increase the complexity of the system design, to the point where simulations or tests in the presence of gravity no longer provide the necessary results. Of particular concern are those that depend heavily upon accurate dynamic characterization as well as high bandwidth, multi-channel control to meet their requisite precision. Ground based testbed results and on-orbit behavior are different and therefore provide a reduced level of confidence that the system will perform to the required precision.

Similar issues have been faced in other fields. For example, wind tunnels fulfill an important role between aerodynamic modeling and aircraft manufacturing. By guiding the development of modeling capabilities, calibrating those models, providing high fidelity scale model tests, etc., they play an important role in evolving new technologies from theory to application. The question arises: is there an equivalent facility to wind-tunnels for microgravity research?

There is an opportunity to take advantage of a new development environment to aid in the technology maturation process that entails the use of dynamics and controls research laboratories which enable long duration, microgravity testing while facilitating the iterative research process and being tolerant of risk during the development of the technology. Throughout two decades, the MIT Space Systems Laboratory has deployed a series of microgravity experiments for the development of new technologies to help in the areas of dynamics and controls which have filled this step in different manners. These experiments were conducted in multiple microgravity facilities (space shuttle, MIR Space Station, and ISS) and under different operational scenarios (long-term, short-term, highly interactive, etc.). Important questions arise from the experience obtained in designing and operating these different experiments:

- What are the common design elements between these experiments?

- Which design elements helped these experiments fulfill the need for this new step in the technology maturation process?
- Can the lessons learned from these experiments apply to future experiments?

The answers to these three questions motivates the development of the design philosophy presented in this thesis.

Further motivation arises from the first question presented above: *is there an equivalent facility to wind-tunnels for microgravity research?*

The answer lies within the ability to make the best use of the ISS. In 1998 NASA asked the National Research Council (NRC) to study how to manage and conduct research in the International Space Station (ISS) over the long term. The NRC team, which included scientists, engineers, and educators, studied the options of maintaining all operations within NASA, outsourcing science management to industry or educators, or creating a new entity. The NRC concluded "that NASA should establish a Non-Governmental Organization (NGO) to manage all aspects of research on the ISS and the NGO should have sufficient authority to carry out its assignments and responsibilities." [NRC, 1999]. The NGO would carry out management of all research activities, while NASA and its international partners would continue to carry out maintenance and upgrades of the ISS. However, the NRC report did not specify the structure or operations of the NGO, rather NASA is accepting proposals from multiple groups, composed of industry and education leaders, on how to shape the NGO; NASA will then seek congressional approval once a proposal is selected.

The NRC report concludes that the principal use of the ISS must be for research. While other activities may take place (e.g., education, staging for human space exploration missions, commercial services, and possibly tourism), the only activity which is immediately ready to begin and which justifies the existence of the ISS is research. Therefore, the NRC recommends that the following principles should guide the operations of the ISS:

- High-quality basic and applied research should be paramount.

- Responsibility for managing and supporting research would not require that the organization manage other ISS activities.
- The research community should have early, substantive, and continuing involvement in all phases of planning, designing, implementing, and evaluating the research use of the ISS.
- The organization must be flexible and capable of adapting over time in response to a changing needs and lessons learned.
- Basic and applied scientific and engineering uses should be selected on the basis of their scientific and technical merit, as determined by peer review.

The report further states that the proposed non-governmental organization must fill four key roles:

- Provide the highest caliber scientific and technical support to enhance research activities
- Provide the research community with a single point of contact through which it can utilize the capabilities of the ISS
- Promote the infusion of new technology for ISS research
- Stimulate new directions in research, for both established and new user communities

This thesis presents methods to respond to the NRC guiding principles and help partially fulfill the key roles of the NGO. The thesis identifies the special resources of the ISS which enhance the ability to conduct science, presents a methodology for designing research experiments that best use these resources, and creates evaluation guidelines for research proposals for the ISS which are best performed by peer scientists. The goal of the design principles is to encourage the researcher to look at the ISS in new ways. Not only should the scientist see the ISS as a general tool in their research; they must realize the unique capabilities of the ISS and utilize them to their greatest extent in support of their research, making the best use possible of what the ISS offers.

Research on the ISS will cover a broad range of areas that range from human physiology to space technologies to education. NASA identified the following research directions for the ISS in 2000 [NASA, 2000]:

- Biological Research and Countermeasures / Advanced Human Support Technology
- Biotechnology
- Combustion Science
- Fluid Physics
- Fundamental Physics
- Gravitational Biology and Ecology
- Materials Science
- Space Science
- Engineering Research and Technology Development
- Space Product Development
- Earth Science

This thesis will concentrate on the aspect of engineering research and technology development. The advancement of space technologies has been closely tied to a set of levels called the "Technology Readiness Levels" (TRL). Therefore, when considering the use of the International Space Station for space technology, a goal is to permit an experiment to advance in TRLs. This thesis studies how to ensure that a technology destined to be tested in the ISS can move closer to space worthiness.

1.1.1 NASA Technology Readiness Levels

> "Technology advances do not occur and mature in an orderly or even predictable manner, and they certainly do not occur in regular, well-organized steps. Still, the progress of a technology advance from that first glimmer of inspiration to its implementation on an operational spacecraft can be conceptualized as progress on a road toward ever increasing understanding, modeling fidelity, and confidence. The technology readiness levels described below represent milestones that demark progress along that road." [NMP, 2003]

Space technology maturation is a challenging process. Substantial amounts of money, time, and human resources go into the development of new spacecraft. At every point in the design life of a new spacecraft there are substantial risks involved, especially as the complexity of new design increases. Over a decade ago NASA developed the Technology

Readiness levels to determine where in the design process a specific technology stands. Is the technology in its infancy? Is it ready for use in spacecraft? These levels are a guide to engineers and scientists in the development of new technologies, with the goal to reduce the ultimate risk of deploying a space technology. The levels attempt to divide the design process into nine steps, each one building upon the previous steps, driving a technology to mature in increments.

> "Technology Readiness Levels (TRLs) are a systematic metric/measurement system that supports assessments of the maturity of a particular technology and the consistent comparison of maturity between different types of technology. The TRL approach has been used on-and-off in NASA space technology planning for many years and was recently incorporated in the NASA Management Instruction (NMI 7100) addressing integrated technology planning at NASA." [Mankins, 1995]

Appendix A presents the definition of the nine TRLs as presented in the TPF Technology Plan, which presents a concise general description of the levels.

While the use of TRLs is not universal, they have been widely accepted as one important method to determine the state of development of a technology. TRLs are widely used within NASA in major programs such as the New Millennium Program (NMP) and the Origins Program. The use of TRLs, which began at NASA, has expanded to other major research institutes, including part of the DoD. In this case an independent study concluded that "it is feasible for TRLs (or an equivalent) to support or add value to the decision-making process. However, it is only one of several critical factors in the decision-making process..." [Graettinger, 2002] In most cases when TRLs are used, these are refined for the specific application. In the case of the DoD, for example, the TRLs have been modified to more directly follow specific technologies: "TRLs are described in the DoD 5000.2-R document from a systems perspective, and thus are intended to be appropriate for both hardware and software... The Army, for example, has developed a mapping of the TRLs to software... and the Army Medical Research and Materiel Command is working on defining corollaries for biomedical TRLs" [Graettinger, 2002]. The NASA NMP has made similar modifications: "Added to their description are criteria used by NASA’s New

Millennium Program to determine when a particular TRL has been reached." The wide use of the TRLs and the maintenance of their overall guidelines show that the concept behind them is valid across a wide range of disciplines.

But TRLs are not necessarily simple to follow. While initially defined as "systematic", the TRLs are not necessarily linear, and every step is not always followed: "The linear metaphor of a road is not a perfect one. On a road every milestone must be passed to go from one end to another. Sometimes one or more Technology Readiness Levels are skipped because they are not appropriate to the technology advance at hand." [NMP, 2003]. The amount of cost, complexity, and risk from one TRL level to the next are not always the same nor small; by the definitions of TRL 7 itself: "Because of cost, it is a step that is not always implemented." Achieving TRLs 1-4 usually present small risks, complexity, and cost. Developing the representative hardware called for in TRL 5 adds a substantial amount to the cost. Creating the operational environment of TRL 7 adds substantially to the cost, risk, and complexity. Once TRL 8 is achieved, the only substantial increase is on cost to develop the flight system. Figure 1.2 shows a pictorial representation of how complexity, risk, and cost may increase for a program if it were to follow each TRL one at a time. As mentioned, TRLs are not necessarily followed one at a time; but skipping one TRL which may not be appropriate for the technology does not cancel the fact that these factors increase substantially from the previous TRL.

The amount that cost and risk increase from one TRL to the next often depends on the ability to demonstrate the technology in a *relevant environment.* In some cases this means demonstrating the technology in space. These demonstrations were limited to free-flyer spacecraft or space-shuttle experiments after the MIR Space Station was retired. The ISS can fill the void in the availability of representative environments for technology maturation. A part of the motivation is to answer the question *how can the ISS help mature technologies through the TRL scale?*

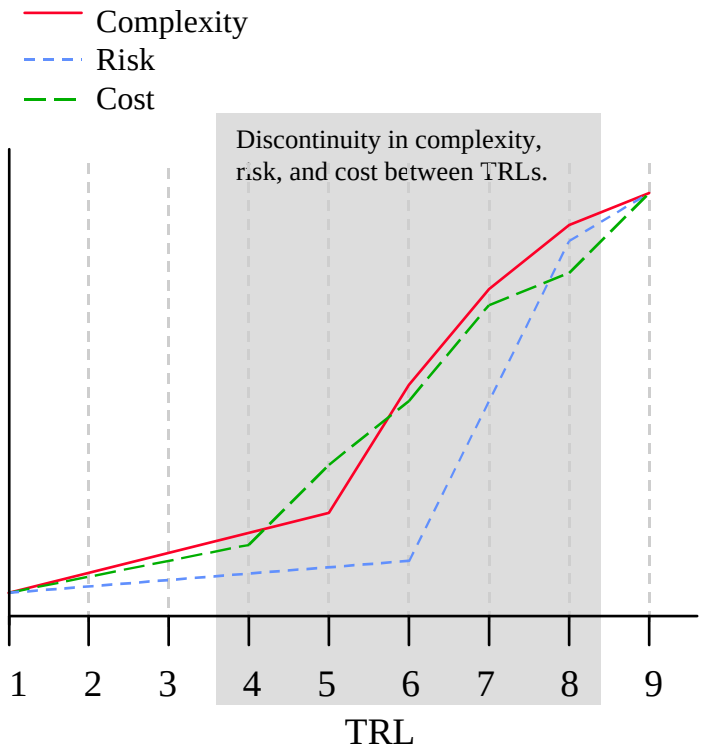


Figure 1.2 Discontinuity in complexity, risk, and cost at each TRL

1.2 Microgravity Research Facilities

Microgravity experimental research can occur in a wide range of facilities, depending on the fidelity, cost, and operational limitations necessary and/or available for the project. While not necessarily exhaustive, the list presented in Table 1.1 shows a wide range of possible facilities which can provide an environment to reproduce or simulate microgravity conditions for research purposes. The table lists 14 different environments to conduct microgravity research in different operational conditions. The first column shows facilities which can be housed by the individual researchers, but which don't necessarily simulate full 6DOF microgravity. The second column lists facilities which have full 6DOF capabilities, but which are usually managed by a third party. The third column lists the existing facilities which provide full microgravity conditions, but which present the largest development challenges.

TABLE 1.1 Sample of available facilities for μ-g research

In-house	3rd Party / Full μ-g	Space
Robot Helicopters	RGO (KC-135)	Free Flyer
6 DOF Robot Arms	Neutral Buoyancy Tank	ISS
Helium Balloons	Drop Towers	Shuttle Payload
Robot Cars		Shuttle Middeck
Flat Floor		
Air table		
Simulation		

Microgravity research has also taken place aboard several space stations that are no longer in operation. These past space stations provided NASA and its international partners with important concepts for the design of the ISS.

Appendix B presents an in depth review of the most distinguishable characteristics of the different microgravity environments and their general operational procedures, as well as an overview of the research conducted aboard prior space stations. Each of the facilities have shortcomings. Some shortcomings do not affect the scientific nature of the experiment (e.g., high costs), but they can affect the success of the mission. Other shortcomings affect the scientific results (e.g. limited dynamics or DOF). Each of these factors is important in selecting the most appropriate path for technology maturation.

Table 1.2 summarizes how the different facilities reviewed in Appendix B compare with each other. The table concentrates on the ability of the facilities to provide an environment representative of microgravity in terms of degrees of freedom and dynamics; they also described the operational nature of each facility, since a trade-off exists between achieving a realistic microgravity environment and the complexity and costs of the operations. The DOF column shows how many degrees of freedom are possible in the facilities; the number outside parenthesis shows the commonly achievable number of DOFs, the number in parenthesis shows the maximum achievable via special hardware. The last column indicates the relative cost of the projects; more expensive projects have a larger number of

TABLE 1.2 Sample of available μ-g research facilities

Facility	Representative Environment				Experiment Operations				Cost
	DOF	Dynamics	Exposure	micro-g Duration*	Campaign Duration*	Operations	Data Xfer	Accessibility	
Free Flyer	6	5	5	5 (mo-y)	5 (mo-y)	1	2	1	$$$$$
ISS	6	4	4	5 (h-y)	5 (mo-y)	2	5	3	$$$$
Shuttle Payload	6	4	4	4 (h-w)	4 (h-w)	2	3	2	$$$$
Shuttle Middeck	6	4	3	4 (h-w)	4 (h-w)	2	3	2	$$$$
RGO (KC-135)	6	3	1	2 (20s)	3 (1w)	3	5	4	$$$
Neutral Buoyancy Tank	6	1	1	3 (h)	3 (1w)	3	5	4	$$$
Drop Towers	6	4	1	1 (10s)	3 (1w)	3	4	4	$$$
Robot Helicopters	4(6)	2	1	2 (m-h)	5 (mo-y)	4	3	5	$$
6 DOF Robot Arms	6	2	1	3 (h)	5 (mo-y)	5	5	5	$$$
Helium Balloons	4(6)	1	1	3 (h)	5 (mo-y)	4	4	5	$$
Robot Cars	3(5)	1	1	3 (h)	5 (mo-y)	5	4	5	$
Flat Floor	3(5)	3	1	3 (h)	3 (1w)	4	4	5	$$
Air table	3(5)	3	1	3 (m-h)	5 (mo-y)	5	4	5	$
Simulation	6	2	1	5 (s-y)	5 (mo-y)	5	5	5	$

* Key to times: y = year, mo = month, w = week, h = hour, m = minute, s = second

dollar signs. The other columns use a scale of 1 (worst) to 5 (best) to illustrate the ability of each facility to better serve the project. The dynamics column indicates the ability of the facility to allow experiments to demonstrate their full dynamic effects, including

orbital dynamics. The exposure column indicates if a facility can provide an environment which exposes the project to the space environment. The operations column indicates how easy it is to operate the experiment; a lower number means more complex operations (it is not easy). The data transfer column shows the ability of a facility to support data transfer in real-time and at minimum cost to the scientist. The accessibility column indicates how easy it is for the researcher to access their experiment for upgrades, changes, and repairs. The microgravity duration column indicates how long the experiment is exposed to microgravity continuously; while the experiment duration column indicates how long a campaign of tests can last.

This summary shows the ability of the ISS to create a representative microgravity environment. The review of past space stations indicates that the ISS has a clear set of qualities that set it apart from the other experiments. Chapter 2 identifies the special qualities of the space station, especially as they differ from other facilities that can provide good microgravity conditions and with respect to free flyer experiments

1.3 Other Shared Remote Facilities

The development of both Antarctic and Ocean research facilities provides several insights into the design of microgravity research laboratories. Appendix C presents an in depth review of these two remote environments. As the reviews indicated, the Antarctic program stresses the need to ensure that science guides the design of the facilities. Both types of research address the need for life support and operations in stressful environments. Ocean research provides further insight into where to conduct analysis and the need for large areas to conduct the actual experiments. Both Antarctic and Ocean research facilities ensure that multiple projects are supported; neither of the programs would be viable if they did not continuously welcome scientists to conduct new research.

But these facilities account not only for humans to be present, but for the researcher themselves to conduct the research. This is not an option available, at least yet, for space research. Antarctic researchers reported that human presence was essential to maintain the

programs operational; emphasis was placed on the need to have staff to support researchers on location. Ocean research vessels are designed to host scientists on board; the capability of on-board laboratory equipment continuously grows, allowing scientists to analyze data during the mission. Space research is constrained by the need for experiments to be conducted by a limited set of humans, rather than the researchers themselves. The need for this type of remote operations where the scientist is not in direct contact with the experiment will be further addressed in this thesis in subsequent chapters.

1.4 Thesis Roadmap

Figure 1.3 presents the thesis overview graphically. It summarizes the content of all the chapters and relates them with the steps of the scientific method presented at the start of this chapter. This first chapter presents the objectives of the thesis and the motivation behind it, as well as background research on microgravity and remote research facilities.

Chapters 2 and 3, together, present the two parts of the hypothesis presented at the start of this chapter. Chapter 2 defines the major challenges of space research for successful technology maturation. The chapter also presents an in-depth review of the facilities available in the ISS and the challenges faced in conducting successful scientific research. Through this review the chapter identifies the special resources of the ISS which clearly distinguish it from the other microgravity facilities presented in Chapter 1. These special resources will be taken into account later on in the development of experiments.

The MIT SSL Laboratory Design Philosophy is presented in Chapter 3. The chapter first identifies the qualities that demonstrate successful research in the specific area of dynamics and control, an area of expertise for the MIT SSL. Next the chapter defines the 11 features identified as essential for a successful research facility; these are grouped into four main areas. The basic scientific guidelines that stand behind these groups are then presented. The chapter concludes by a review of the past MIT SSL microgravity experiments which inspired this philosophy.

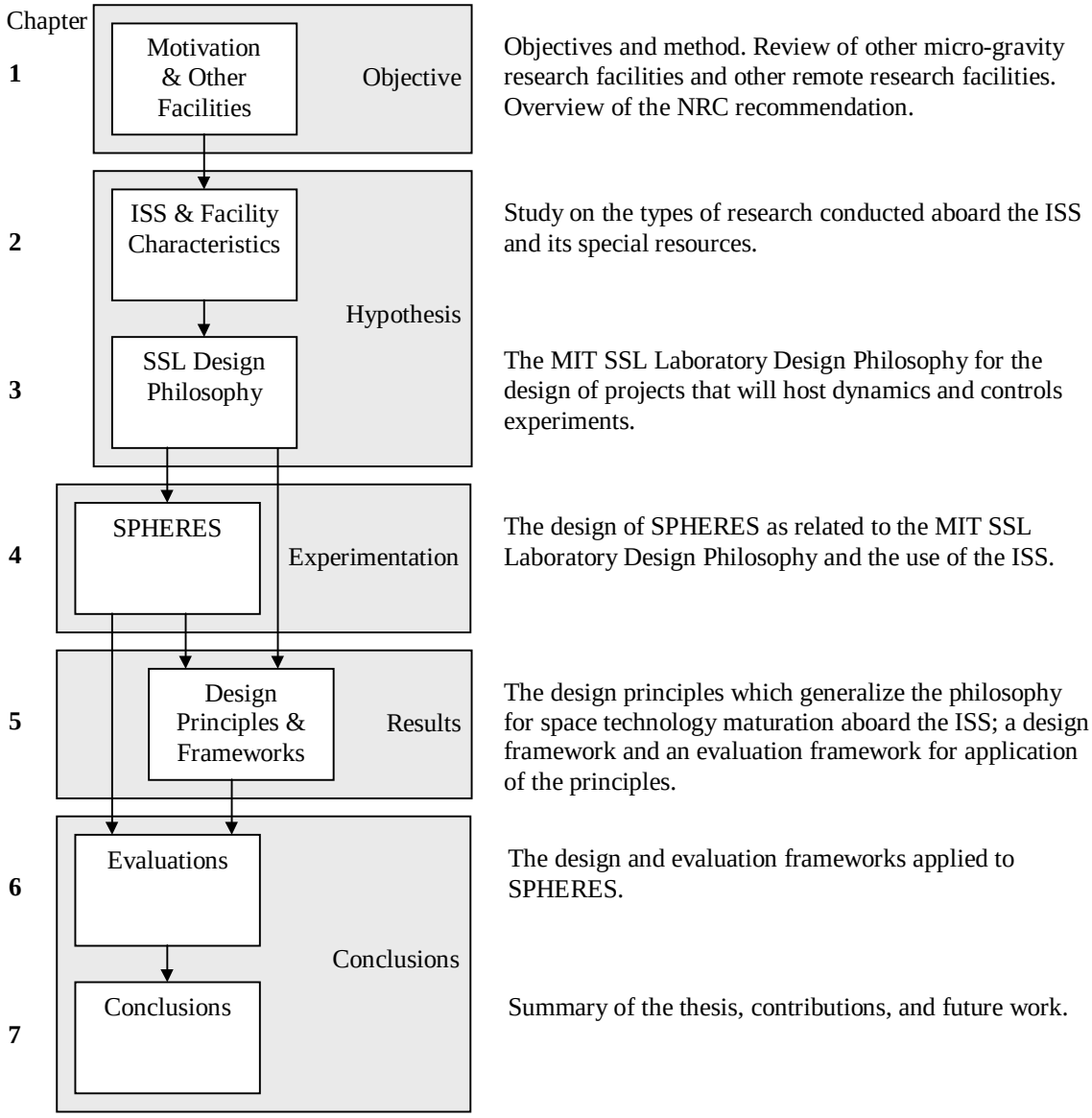


Figure 1.3 Thesis roadmap

Chapter 4 describes the design of the SPHERES laboratory for distributed satellite systems (DSS), which constitutes the experimental portion of the thesis. After introducing the overall design of the hardware and operational programs, the chapter describes in further detail how SPHERES implemented the features of the MIT SSL Laboratory Design Philosophy presented in Chapter 3. Each of the four groups is presented separately.

Chapter 5 presents the seven design principles that resulted from implementing SPHERES to a) follow the MIT SSL Laboratory Design Philosophy and b) to operate in the ISS. This chapter presents each of the principles in a separate section, explaining the derivation of the principles from the experimentation with SPHERES, and then describing the principle itself. Two application frameworks are presented in Chapter 5: a design framework to aide investigators in the creation of experiments that best utilize the resources of the ISS and an evaluation framework to determine if a project uses the ISS appropriately. These frameworks can be utilized as part of an "institutional arrangement" for conducting science on the ISS. Chapter 6 thoroughly analyses the SPHERES facility using both frameworks.

Chapter 7 concludes the thesis by summarizing how the design principles and frameworks fulfill the objectives of the thesis.

Chapter 2

MICROGRAVITY RESEARCH ABOARD THE ISS

This chapter expands on the first part of the hypothesis presented in Figure 1.1: the use of the International Space Station as a host creates the perfect low-cost environment for technology maturation. The chapter discusses the challenges of μ-g research identified from the literature search and through past experiences of the MIT SSL. Literature about the ISS, including a review of research up to date, helps identify the type of experiments conducted in the ISS; this chapter specifies what the thesis regards as a technology maturation experiment, as related to current research conducted aboard the ISS. Lastly, the chapter presents the special resources offered by the ISS.

2.1 Issues and Challenges of Microgravity Research

The literature review of Chapter 1 provides insight into the issues and challenges faced by microgravity research. Achieving maturation of space technologies was tied by the Technology Readiness Levels to the ability to operate in representative environments. The TRLs and availability of these environments define the challenges of micro gravity research. TRLs were introduced in Chapter 1 as a proposed method to mature technology in a step-wise manner. As shown in Figure 1.2 on page 39, three primary drivers have impact on the ability of a technology to follow all TRLs: risk, complexity, and cost. The review of other facilities indicated that remote operations also pose a challenge to space technology maturation. Lastly, it is shown by the fact that previous space stations pro-

grams were driven in many ways by political and social needs, and that the high visibility of these programs is an issue which cannot be ignored.

Risk. Risk exists in every stage of space technology maturation, from the feasibility of the program itself to the actual operation of equipment. Risks are created by the environment, costs, and politics which surround microgravity research. The space environment creates risks not experienced inside the earth atmosphere, such as space radiation and collision with natural objects. The inability of humans (in most cases) to work directly with deployed spacecraft of the projects can result in the permanent reduction of capabilities unless full redundancy is implemented. When humans can access the spacecraft, the availability of resources (including time, equipment, and parts) to repair spacecraft is limited. Costs, while an important factor on their own, also contribute to the risk of a space mission; the costs drive the development time down and limit the ability to create fully redundant systems. Politics also adds to the risks of a mission, although in a different manner. Due to politics, space engineering tends to work in a conservative fashion, many times utilizing old-but-trusted technologies, rather than the latest technologies, for common parts of a space craft; these older technologies usually work behind highly advanced science items. Creating interfaces between the technologies puts a risk the feasibility of the mission and can potentially limit the usefulness of the new advanced technologies to be tested. When only advanced technologies exist, the risk of using them is too high for the political drivers behind the project. Politics can also reduce the time for development, creating new risks due to unforeseen problems. Reducing the risk of a mission by allowing humans to operate new technologies in a controlled environment is a goal for the use of the ISS.

Complexity. Space systems are some of the most complex systems created by human kind. Spacecraft interface dozens of sub-systems, contain up to miles of cable, which carry thousands of electronic signals, utilize advanced science items, and operate using a number of different robust real-time software implementations. While a specific tool for a spacecraft can be tested on its own in simple manners during preliminary tests, as that tool

is integrated into the rest of the spacecraft, the complexity of its operations grow. That is, as a technology matures towards a high TRL, the complexity of using the tool grows. Increased complexity usually results in higher costs and the need for more personnel to work on the development of the technology. The increased complexity also adds to the risk, as the addition of interfaces creates new possible failure points. Therefore, it is desirable to lower the complexity of a mission and/or to mature individual sub-systems as far as possible prior to integration into the more complex spacecraft. Further, it is desirable to test the integration of sub-systems in an environment which does not necessarily add as much complexity as developing the space-qualified product.

Cost. For many space programs, cost becomes the deciding factor in the future of the mission. Space missions have costs higher than most other research on the ground due to the need for expensive specialized equipment, launch vehicles, and operational costs. The other issues presented also create an increase in cost, for example: reducing risk by redundancy increases cost; increased complexity increases cost; the drivers behind politics are mostly economic. The high cost of these missions creates inbalance in the funding of the science programs for ground-based research and space-based research; this forces space-based research to be highly beneficial to the funding sources, something adding extra burden to the researchers beyond the direct science goals of a mission. Therefore, to overcome the issue of cost for space research one must first, allow multiple researchers to benefit from the research, ensuring that the research benefits a large portion of the population; and second, that the other factors which affect the cost of a mission are reduced in such a way that the ultimate cost of the mission is also reduced.

Remote Operations. The need for remote operation means that the scientists will not be present in the actual tests; rather an astronaut is trained to operate the facility. While astronauts are highly-educated members of the space community, they are rarely experts on all the experiment fields to which they are assigned. Yet, in some cases astronauts will have to make decisions based on real time results; these decision potentially affect the success of the research. In these cases astronauts will require substantial training to be able to

make the best decisions; at the same time the experimental facility will need to provide astronauts real-time feedback information for them to make the necessary decisions. In other cases astronauts may not need to do any decision making, but in that case a researcher must create an automated experiment and/or create the necessary data links to make the decisions on the ground and command the space-based experiment remotely. A researcher needs to balance the need of astronauts to make real-time decisions as compared to the complexity needed to automate the equipment.

Visibility. The visibility of space missions is usually on the extremes: the major missions are highly visible and subject to substantial public review while smaller missions go unnoticed, very few are in the middle ground. This presents a challenge to the researcher. Highly visible missions will face extreme safety and public relations pressure. This tends to increase the cost of the mission as the safety requirements increase. Public relations pressure tends to affect the timeline of the mission, sometimes forcing steps to be skipped; at the same time, public relations tend to criticize high costs, forcing the mission to balance the cost to achieve the necessary safety with the cost to achieve the scientific goals (sometimes causing cuts in the goals of the mission). In a similar fashion, a high-visibility mission calls for the use of advanced technologies to attract the attention of the public; but the safety concerns drive towards the use of conservative technologies in other parts of the project. On the other hand, a low-visibility mission will face hard times to obtain the necessary funding and attention to be successful. Even if the necessary funding is obtained, low visibility of a mission may cause its facilities and results to not be used effectively, making the mission short-lived.

The use of the International Space Station should address these issues and challenges. Ultimately we wish to answer:

- Can the use of the ISS reduce the risk of space technology maturation?
- Is the complexity of a project that goes through the ISS reduced?
- Can the cost of a project be reduced by using the ISS?
- Are the remote operations of the ISS effective?

- Can the use of the ISS remove the visibility factor from the feasibility of a mission?

2.2 Research Areas of the International Space Station

To answer whether the ISS can address the issues and challenges of space research one must first understand what the ISS is. Appendix D presents a detail review of the resources available aboard the ISS and the current challenges and future upgrades of the program. This section concentrates on the objectives of the ISS program, creating a direct relationship with the success of past space stations, and helping identify the research conducted aboard the ISS which directly relates to the results of this thesis.

The objectives of the ISS as stated in the ISS Familiarization Manual developed by NASA are:

> "The purpose of the ISS is to provide an “Earth orbiting facility that houses experiment payloads, distributes resource utilities, and supports permanent human habitation for conducting research and science experiments in a microgravity environment.” (ISSA IDR no. 1, Reference Guide, March 29, 1995)
>
> "This overall purpose leads directly into the following specific objectives of the ISS program:
> - Develop a world-class orbiting laboratory for conducting high-value scientific research
> - Provide access to microgravity resources as early as possible in the assembly sequence
> - Develop ability to live and work in space for extended periods
> - Develop effective international cooperation
> - Provide a testbed for developing 21st Century technology."
>
> [NASA, 1998]

After creating these objectives, NASA worked to further detail the research objectives of the ISS. To this purpose, NASA has created an ongoing program to determine the "research directions" of the ISS. During the development of these directions, NASA first defined the ISS as a special type of laboratory, one which has three special purposes:

- "an *advanced testbed* for technology and human exploration;

- a *world-class research facility*; and
- a *commercial platform* for space research and development." [NASA, 2000]

As of January 2000 the NASA Office of Life and Microgravity Science Applications had identified a number of research fields which can directly use the resources provided by the ISS to advanced human knowledge and provide benefits to the people in the ground; these are presented in Appendix D.

The objectives and research directions of the ISS address some of the challenges identified in the first section of this chapter by creating a facility which will benefit a large number of scientists; ultimately the science obtained will benefit a large portion of Earths population once NASA's science objectives are met.

2.2.1 Thesis Research Area Identification

The ISS creates a special environment in space for conducting a wide range of microgravity experiments. This section studies the types of experiments conducted aboard the ISS and defines the type of experiments that this thesis concentrates on.

NASA conducts multiple research experiments in the ISS simultaneously. Each "expedition" of the ISS – each crew rotation – is given a delimited set of tasks, which are published by NASA. Table 2.1 shows the experiments that Expedition 6 conducted through their six month rotation. This expedition was chosen as a sample since it constituted a six month period when the ISS operated normally with three crew members and standard supply missions.

Research of the goals behind each of the twenty experiments that took place on Expedition Six allows division of the experiments into the following main areas:

- Experiment Operation Types
  - Observation
  - Exposure
  - Iterative Experiments

TABLE 2.1 Research experiments of Expedition 6

Id	NASA Field	Experiment	Area	Type
1	Bioastronautics Research	The Effects of EVA on Long-term Exposure to Microgravity on Pulmonary Function (PuFF)	Science	Iterative
2		Renal Stone Risk During Space Flight: Assessment and Countermeasure Validation (Renal Stone)	Science	Exposure
3		Study of Radiation Doses Experienced by Astronauts in EVA (EVARM)	Science	Exposure
4		Subregional Assessment of Bone Loss in the Axial Skeleton in Long-term Space Flight (Subregional Bone)	Science	Exposure
5		Effect of Prolonged Spaceflight on Human Skeletal Muscle (Biopsy)	Science	Exposure
6		Promoting Sensorimotor Response Generalizability: A Countermeasure to Mitigate Locomotor Dysfunction After Long-duration Space Flight (Mobility)	Science	Exposure
7		Spaceflight-induced Reactivation of Latent Epstein-Barr Virus (Epstein-Barr)	Science	Exposure
8		"Monitoring of Heart Rate and Blood Pressure During Entry, Landing, and Egress: An Index of Countermeasure Efficacy (Entry Monitoring)"	Science	Exposure
9		Chromosomal Aberrations in Blood Lymphocytes of Astronauts (Chromosome)	Science	Exposure
10		Foot/Ground Reaction Forces During Space Flight (Foot)	Science	Iterative?
11	Physical Sciences	Protein Crystal Growth—Single-locker Thermal Enclosure System (PCG-STES)	Science	Iterative
12		Microgravity Acceleration Measurement System (MAMS)	Technology	Exposure
13		Space Acceleration Measurement System II (SAMS-II)	Technology	Exposure
14		Investigating the Structure of Paramagnetic Aggregates from Colloidal Emulsions for the Microgravity Sciences Glovebox (MSG-InSPACE)	Science	Iterative
15		Vibration Isolation System for the Microgravity Sciences Glovebox (MSG-g-LIMIT)	n/a	n/a
16		Coarsening in Solid-Liquid Mixtures for the Microgravity Science Glovebox (MSG-CSLM)	Science/Tech	Iterative
17	Space Product Development	Zeolite Crystal Growth Furnace (ZCG)	Science/Tech	Iterative
18		Microencapsulation Electrostatic Processing System (MEPS)	Science	Iterative
19	Space Flight	Crew Earth Observations (CEO)	Education	Observation
20		Earth Knowledge Acquired by Middle-School Students (Earth-KAM)	Education	Observation
21		Materials International Space Station Experiment (MISSE)	Science	Exposure

- Major areas of study
  - Educational
  - Pure Science
  - Technology

Experiment Operation Types

Observation. Experiments that consist solely of the observation of celestial bodies (either the Earth or others), are considered observation experiments. For example, when astronauts are asked to take pictures of Earth, without conducting any further research on the results.

Exposure Experiments. Exposure experiments are those that utilize the μ-gravity environment of the ISS solely to expose material to the reduced gravity and/or space environment, without actively conducting experiments in the ISS with the materials or subject being tested. These experiments include, for example, medical experiments where astronaut biological data are measured before and after the flight, but no science is performed during the expedition – possibly the astronauts may conduct special exercises during the expedition, but since no measurements or other science is conducted during the expedition itself, these are considered exposure times, not research times.

Iterative Experiments. The other main type of operations for ISS experiments are those that require multiple iterations of test runs while the experiments are aboard the space station. This definition does not preclude the type or location of the experiments, but rather identifies their operational nature. An experiment may be performed either inside our outside the station, and it may be for pure science or tests of new technologies. The most important concept for this type of operation is that the facilities must be able to present results and perform new experiments during their time in the ISS.

Experiment Areas

Educational. The ISS is often used to conduct activities with an educational goal. The ISS crew continuously communicates with students on Earth, via both audio and video; they take pictures to be used in educational exercises, and even sometimes conduct simple experiments developed by children. This research time is outside the scope of this thesis, since the goal is not directed towards the development or understanding of new technologies.

Pure Science Experiments. A large portion of experiments aboard the ISS are conducted to learn more about the pure sciences. These experiments use μ-g to understand how things behave differently between gravity and micro- gravity conditions. They also help create materials in new ways that are not possible on Earth. Ultimately these experiments provide results for use in ground products. In some cases, the experiments utilize many of the ISS resources to conduct iterations of the full research cycle, where results are obtained aboard the ISS and new experiments started with knowledge obtained from those initial results. In other cases pure science experiments consist solely of observation or exposure.

Space Technology Experiments. These experiments are those that test new technologies for use in future space missions. These technologies allow better understanding of the μ-g environment to facilitate the access and use of space. While pure science experiments study the effects of the space environment on biological or physical items, space technology experiments demonstrate the ability of human created items to operate correctly in a microgravity environment. The experiments aboard the ISS allow the necessary technology demonstration in a relevant space environment to advance the technology through several TRLs (the definition of a relevant environment is presented in Appendix A).

Thesis Concentration

This thesis concentrates on iterative experiments that serve science and technology goals. Emphasis will be on those experiments related to space technology, but some science experiments can serve as an important example of how the ISS enables research in space to advance an area by allowing iterations. The thesis does not dive into experiments that are solely for observation or exposure, other than to identify the division of time spent in the ISS between these types of experiments and to evaluate the subsequent effectiveness of the use of the ISS.

2.3 Special Resources of the ISS

This chapter begins with the introduction of the major challenges and issues of microgravity research identified through literature research: risk, complexity, cost, remote operations, and visibility. The goal of the chapter is to identify whether the ISS can help reduce the negative effect of these issues on space technology maturation. The chapter presents an overview of the ISS objectives and identifies the challenges of the ISS itself. This study of the ISS leads to the identification of several special resources of the station which do in fact help it reduce the effects of the identified challenges, and which contribute to the correct utilization of the ISS as a laboratory for space technology maturation. The following resources have been identified as most important:

Crew. The fact that humans are present in the space station to interact with and control different facilities is the most obvious and yet many times overlooked resource available in the ISS. While all reviewed reports identified crew availability as a major challenge for the ISS, clearly indicating the need to maximize their time dedicated to research, many times scientists put heavy emphasis on automation and independence from the crew. Yet, the crew can help reduce the effects of many challenges: risk is reduced since humans can stop an experiment which is operating incorrectly; complexity and cost can be reduced by the need to remove automation tools. Therefore, any project that uses the ISS should actively use the humans to help the science and reduce risk, complexity, and cost. The ultimate goal is to determine the correct balance between astronaut availability and need.

Communications. The issue of communications and data download resonated through all the reviews of the ISS. Correct use of the ISS communications system, and its constant expansion, is clearly a priority for NASA and a special resource which benefits all users of the ISS. The availability of continuous high-bandwidth communication to ground reduces the cost and complexity of missions which would otherwise need their own communications equipment. The availability of ever-increasing communications features will help with the issue of remote operation as real-time video and other teleconferencing options

become increasingly available. Therefore, scientists should utilize the ISS as a direct communications link between them and their experiments.

Long-term experimentation. A unique features of the ISS is that it allows long-term microgravity experimentation in a laboratory environment. The long-term nature of the ISS helps to reduce the effects of high visibility as space research becomes part of daily life at NASA and the scientific community. The ISS allows space technology advances to come over longer periods of time, where specific one-time events (such as a landing or a docking) no longer need to mark the success or failure of a mission. Instead, the long-term nature of the ISS allows technology to mature over small steps in a low-visibility environment, allowing scientists to better concentrate on their research rather than outside factors. At the same time experiments which reach the space station will always have high visibility among the scientific community. Further, once they demonstrate revolutionary advances, new technologies will gain high-visibility among the public in general.

Power sources. The ISS can provide several kilowatts of power to each experiment. Because power is usually a trade-off between mass (i.e., larger batteries provide more power but have larger mass), utilizing the existing power sources of the ISS can help to substantially reduce the mass of an experiment, and in turn its cost. Because power sources are a constant safety concern, removal of power sources from an experiment also reduces the risk of the mission. Therefore, ISS supplied power should be utilized by the experiments, otherwise experiments that send their own power sources are duplicating an existing resource and wasting up-mass to the ISS.

Atmosphere. While some times an experiment intends to demonstrate the ability of its hardware to operate in a space environment, the development of ‘rad-hard’ techniques has been understood for several decades. Instead, many experiments wish to demonstrate the ability of their hardware and software to perform correctly in a microgravity environment without the need to worry about hardware failures. In these cases the pressurized environment of the ISS not only provides safety for humans, but also for electronics and struc-

tures. Experiments that can be performed inside the station can have a substantial reduction in cost, complexity, and risk, as compared to free-flyers in space, since they no longer need to worry about being exposed to the space environment radiation and vacuum. Cost is reduced directly by the use of standard components; complexity is reduced since protection equipment is no longer necessary; risk is reduced since the experiment is no longer exposed to the harsh conditions of space and therefore the probability of failure is lowered.

Table 2.2 summarizes the special resources of the ISS and their effects on the challenges of microgravity research. The next chapter will present the MIT SSL Laboratory Design Philosophy, which also addresses those challenges, but from the perspective of creating a new experiment which not only uses existing resources but also creates new features to build upon the existing resources.

TABLE 2.2 Special resources of the ISS that facilitate microgravity research

Resource	Risk	Complexity	Cost	Remote Operations	Visibility
Crew	↓	↓	↓		
Communications			↓	↓	
Long-term experimentation					↓
Power Sources	↓	↓	↓		
Atmosphere	↓	↓	↓		

↓ = reduces challenge

Chapter 3

THE MIT SSL LABORATORY DESIGN PHILOSOPHY

This section presents the second part of the hypothesis presented in Figure 1.1: the MIT SSL Laboratory Design Philosophy can serve as a set of guidelines for the successful development of microgravity research experiments. These guidelines were created from the experience of designing, building, and operating multiple μ-g testbeds, and were the guidelines that drove the design of the SPHERES testbed.

This chapter contains three main parts. The first part discusses the characteristics of a space technology which must be demonstrated to prove technology maturation, as related to the fields of dynamics and controls. Next, the chapter presents the experiments conducted by the MIT SSL which allowed the demonstration of those characteristics. The development of these experiments led to the identification of features required of a testing environment to allow the demonstration of the technologies. That is, the demonstration must show the technology posses the characteristics to prove its maturation without being limited by the testing environment. These features are grouped into four common areas; each of these four areas is explained based on scientific research practices. These features form the MIT SSL Laboratory Design Philosophy.

3.1 Definitions

Before presenting the MIT SSL micro gravity projects and the Laboratory Design Philosophy which resulted from them, it is important to understand two concepts that will appear

continuously throughout the remainder of this thesis. The Laboratory Design Philosophy contains one key word: laboratory. The term laboratory is used not only to represent the physical research where research is conducted. From the dictionary (Merriam-Webster) definition of a laboratory we can obtain further insight:

> Main Entry: lab·o·ra·to·ry
>
> 1 a : a place equipped for experimental study in a science or for testing and analysis; broadly : a place providing opportunity for experimentation, observation, or practice in a field of study b : a place like a laboratory for testing, experimentation, or practice <the laboratory of the mind>

The meaning of the word laboratory in the design philosophy, and for the remainder of the thesis, specifically addresses the need to support experimentation in a field of study. The support is provided not only by physical equipment, but also by the correct organizational structure to ensure that a field of study can be researched.

The physical equipment which forms part of a laboratory is the facility. A facility is defined (Merriam-Webster) as:

> Main Entry: fa·cil·i·ty
>
> 1 : the quality of being easily performed
>
> 2 : ease in performance : APTITUDE
>
> 3 : readiness of compliance
>
> 4 a : something that makes an action, operation, or course of conduct easier -- usually used in plural <facilities for study> b : something (as a hospital) that is built, installed, or established to serve a particular purpose

This thesis follows the definition that a facility makes […] a course of conduct easier and is established to serve a particular purpose. In the case of this thesis, a facility serves to facilitate research in the field of study for which a laboratory is established.

Therefore, a reference to a facility indicates the presence of hardware equipment to make conducting research easier. The use of the word laboratory means that a full research program has been created to enable research on a field of study.

3.2 Characteristics of a Mature Technology Demonstration

The MIT SSL concentrates its research on dynamics and controls technologies. Therefore, the experiments developed at the MIT SSL test a wide range of metrology and control algorithms, as well as sensor and actuator technologies which enable the algorithms to succeed. This section presents the characteristics which must be exhibited by a dynamics and control technology to demonstrate it has matured. These characteristics form the basis behind the objectives of the different dynamics and controls experiments; while the goals of each specific mission are unique, the goal is that the mission-specific algorithms all exhibit the characteristics presented in this section. While these are related directly to the topics of dynamics and control studied at the MIT SSL, their application can be expanded to more general demonstrations in most cases.

These characteristics can also be related to the Technology Readiness Levels. The NASA TRLs provide high-level guidelines of when a technology matures for operation at different steps in its reach for space operation. These characteristics go one level down, they are those properties of a dynamics and control test that must be met every time to demonstrate that a specific TRL level has been met. To demonstrate fulfillment at a specific level, the technology must exhibit the following characteristics:

Demonstration and Validation. For a technology to mature, it must be demonstrated in the correct environment, with results clearly showing the accomplishments of the technology. Results observed in a physical system must be validated with data obtained during the successful completion of the demonstration.

Repeatability and Reliability. The results of a mature technology must be repeatable, that is, they must happen more than once under similar operating conditions. Further, positive results must be obtained in the presence of the different disturbances and commands that may be present during a mission to demonstrate the reliability of the algorithms.

Determination of Simulation Accuracy. A successful technology demonstration must help validate simulations and other tests of lower fidelity. The results of control experiments in a space research laboratory can be compared with simulations to provide confidence in simulation techniques and to gauge the simulation accuracy.

Identification of Performance Limitations. In order to determine the success of new technologies or algorithms one must push these to their limits. Mature technologies must provide insight into most of the physical constraints of a system that may not be observable in a simulation or ground test.

Operational Drivers. Systems issues such as sensor-actuator resolution, saturation, non-linearity, power consumption, roll-off dynamics, degradation, drift, and mounting techniques are most often constraints rather than design variables; that is, these quantities cannot be easily changed by the scientist, but rather scientists must design their experiment around them. Hardware experiments allow scientists to learn the quantitative values of these constraints, which are important during the creation of system models used in the design of control and autonomy algorithms. A mature technology operates successfully in the presence of these drivers.

Identification of New Physical Phenomena. New physical phenomena are usually discovered through observation of physical systems. A mature technology demonstration allows for the identification of these phenomena, creation of models for them, and the exploitation of this new knowledge in future investigations.

The MIT SSL has conducted microgravity experiments over the past two decades to demonstrate and validate dynamics and control technologies. While these experiments covered different areas of research (non-linear dynamics, fluid slosh, load sensors, robust control), each of them attempted to demonstrate each of these characteristics in the technology they tested. The following section summarizes the experiments.

3.3 MIT SSL Previous Space Experiments

The MIT SSL has designed, built, and operated a multitude of flight experiments in the past. The lessons learned from these experiments led to the development of the sets of demonstration characteristics and test environment features. The experiments include:

- Mid-deck 0-g Dynamics Experiment (MODE), which flew on STS-48 in September 1991 and its re-flight on STS-62 in March 1994.
- Dynamic Load Sensors (DLS), which flew on MIR for about three years.
- Middeck Active Control Experiment (MACE), which flew on STS-67 in March 1995.
- MACE Re-flight, which was the first crew-interactive space technology experiment conducted aboard the ISS by Expedition 1 in December 2000.

Appendix E reviews the research conducted through the three programs (MODE, DLS, and MACE) and further discusses the identification of the common features that enabled these experiments to advance dynamics and control algorithms for space technologies. Table 3.1 presents a summary of past MIT SSL microgravity experiments. The table summarizes the mission and its areas of study.The table also shows the total cost of the mission and the time to flight. Re-flight opportunities clearly lowered both metrics. The MODE experiment characterized itself by the creation of the generic equipment (the ESM), which allowed future missions, including DLS, to be developed with low cost and in a small time-frame. DLS further enhanced the success of MODE by operating over an extended period of time. The MACE program developed its own set of generic equipment, which was used over two flights. The MACE re-flight made substantial use of the original MACE hardware to lower its cost and time to flight. Further, MACE allowed algorithms to be selected and modified during the mission, allowing a larger number of areas of study to be investigated.

MODE, DLS, and MACE tested a number of different space technologies to aid in the development of new algorithms and sensors for dynamics and control. Each of these experiments exhibited special features which helped to mature the technologies in a cost effective manner by utilizing the available environments to their full extent. The identifi-

TABLE 3.1 Summary of MIT SSL microgravity experiments

Experiment	Host	Date (year)	Areas of Study	Cost	Time-to-flight* (years)	On-orbit time (weeks)
MODE	STS-48	91	Microgravity fluid and structural dynamics	$2M	3	1
MODE Reflight	STS-62	94	Non-linear structural dynamics on truss structures	$1M	2	1.7
DLS	MIR	96-97	Crew induced dynamic disturbances	$0.75M	1	40
MACE	STS-67	95	Advanced control design on non-linear structures	$4M	3	2
MACE Reflight	ISS (Exp 1)	00-01	Neural networks, non-linear characterization, reaction wheel isolation	$1M	1.5	36

* Time to flight = contract start to actual flight

cation of these features led to the development of the MIT SSL Laboratory Design Philosophy which helps guide the design of new experiments. These features are presented below.

3.4 Features of a Laboratory for Space Technology Maturation

In the area of dynamics and control different technology validation tools play different roles in the maturation process. Simulations, while versatile, low cost, and low risk, only address issues that the control engineer remembers to consider. Implementation on hardware forces the engineer to pay attention to not only the technology but also the details of its implementation. It is these details which a hardware testing facility must allow to be identified correctly.

A testing facility designed to mature technologies must ensure that the tests meet the characteristics presented above in such a way that the technology, rather than the facility itself, limits the ability to demonstrate the maturation of the technology. The facility must create the necessary environment for successful demonstration, so that the results are relevant to

the operational environment of the technology. The performance limitations of the facility must not limit the technology; it is the only way to ensure that the results of the tests are bound by the technology being tested. Further, to create a laboratory environment, the facility must allow the demonstration of all the areas of study which comprise a technology, allowing multiple scientists to conduct experiments over long periods of time. Overall, the laboratory must facilitate reaching technology maturation. Therefore, a laboratory must meet a minimum set of requirements that will surpass the capabilities of the technology.

Each of the MIT SSL microgravity experiments presented above satisfied one or more of these requirements. The features of these facilities which enable them to meet these have been identified and brought together into the MIT SSL Laboratory Design Philosophy. These features will drive the lower-level design of new testbeds, after a high-level design has been decided upon based on the project goals and TRLs to be met. The identified features of microgravity laboratories are:

Data Collection and Validation. A successful research environment must provide data collection of accuracy and precision scalable to the final system to demonstrate operation of the new algorithms or technologies. The collected data needs to ensure the technology is fully observable. The feedback from the research environment must be precise enough to validate the operation of the new technology. The facility must also ensure that the data is presented in a manner useful to demonstrate the validity of the results. Further, the data must be independently validated by a truth sensor.

Repeatability and Reliability. To demonstrate the repeatability of a new technologies, the research laboratory environment must have a better repeatability and reliability rate than the technologies to be tested. The environment must be able to provide similar test conditions through an extended period of time. Similarly, to demonstrate the reliability of a new controller, the environment should be easily changed so as to create different disturbances and commands. Therefore, the research environment must be a controlled setup,

where interaction with the facility can easily recreate an environment or change it in a controlled manner.

Physical End-to-End Simulation. The test environment must provide a sufficiently realistic simulation of the expected operational environment and performance metrics. The environment must correctly simulate the hardware required for the actual mission, including the use of representative sensors, actuators, electronics, and other active hardware. Further, to fully capture the dynamics of the mission, the physical end-to-end simulation must allow its dynamics to be fully understood such that the results can be applied to the actual mission which may have different dynamics. Otherwise, important couplings and perturbations may be masked and the ability to achieve requisite performance levels is difficult to ascertain. The hardware simulation must also allow all essential operational steps of the mission to occur, either continuously or step-wise, so that all parts of the technology can be demonstrated.

Generic versus Specific Equipment. All laboratories distinguish between that which is being tested and the facilities needed to conduct those tests. Since it is difficult to modify hardware in the space environment, it is desired that laboratories based in the ISS have a set of generic equipment able to provide basic operation of the laboratory. Test-specific equipment can be attached to the generic equipment to better model a specific mission. In this way the laboratory can accommodate a multitude of research projects. This re-usability improves the cost-effectiveness of the research.

Hardware Reconfiguration. To demonstrate the reliability of a new algorithm or technology, it is desirable to manipulate the hardware configuration during a specific test to demonstrate increasingly complex geometry or components. Therefore, both the generic and specific hardware should allow easy reconfiguration.

Supporting Extended Investigations. The effectiveness of experimental research is generally correlated with the number of iterative research cycles completed. Sometimes, a test can reveal totally unexpected behavior. Under these circumstances, a cycle cannot

closely follow the previous cycle since time needs to be spent re-exploring the theory before the original hypothesis can be appropriately refined. Therefore, there is a need to maintain access to the laboratory under repeatable test conditions following an extended period of no tests.

Risk Tolerant Environment. Laboratory tests are often conducted on immature and unproven technology. The environment must be designed to accommodate failure or unexpected behavior (e.g., control system instability). Such occurrences must not pose harm to the researcher, the test article or the test equipment. Furthermore, the researcher must expect such occurrences, otherwise they are not pushing the edge of knowledge and capability.

Software Reconfiguration. The ability to alter software provides much more versatility in manipulating test conditions. In particular, when the technology being tested is manifested as software (e.g., control, metrology, system identification, and autonomy algorithms), the ease with which that software can be altered directly impacts the productivity of the tests.

Human Observability and Manipulation. Research is a very human-in-the-loop process. The researcher's ability to observe behavior, refine a hypothesis, manipulate the test conditions, and observe new behavior is at the core of the iterative experimental research process.

Facilitating Iterative Research Process. While human interaction and laboratory reconfigurability are prerequisites for a laboratory, they alone are not sufficient to facilitate the iterative research process. The cycle time from posing an hypothesis to refining that hypothesis based upon correlation between experiment and theory must be sufficiently brief. This helps the researcher track the evolution of inquiry and offers the opportunity to explore alternatives more fully. Laboratory interfaces must be defined to minimize the resources consumed by each research cycle.

Supporting Multiple Investigators. Shared access to a research laboratory dramatically improves cost-effectiveness. Therefore, reconfigurability of the testbed should allow multiple investigators to participate in the program. Guest investigators must not only have access, but that access must be supported by the principal investigator, since the guest investigators may not be familiar with, or able to be present in, the laboratory environment.

3.4.1 Interactions Between the Features

The presented features of a laboratory are not independent of each other. A single characteristic of a system can help achieve multiple features, and success in some features helps to achieve success in others. Table 3.2 presents the interactions between the different features. The table shows when a feature in the rows helps a feature in columns; when an interaction between a feature in the rows exists with one in the columns, a check mark indicates this relationship. This section explains the interactions that occur as each feature in a row helps one or more of the other features.

Data Collection and Validation. The *iterative research process* depends on data analysis between iterations, therefore good data collection and validation is necessary. If the data quality is not precise enough, then the simulation will not achieve full *physical end-to-end simulation* of the experiment.

Repeatability and Reliability. The *iterative research process* depends on multiple iterations being carried out at different periods in time, with the guarantee that the test conditions will be similar. Otherwise, subsequent tests may waste resources trying to identify what has changed about the test environment. To support *extended investigations* the hardware must perform the same way over long periods of time, which cannot be done without a repeatable system. When a facility is known to be reliable scientists have greater confidence to push the limits of their technology towards the limits of the facility, aiding in creating a *risk-tolerant environment*.

TABLE 3.2 Interaction between the SSL Design Philosophy elements

These features ↓ support these features→	Data Collection	Repeat. / Reliab.	Iterative Process	Human Obs./Man.	End-to-End	Extended Invest.	Risk Tolerant	Generic/Specific	HW reconfig.	SW reconfig	Multiple Invest.
Data Collection and Validation			✓		✓						
Repeatability and Reliability			✓			✓	✓				
Facilitating Iterative Research Process											
Human Observability and Manipulation			✓			✓	✓		✓		✓
Physical End-to-End Simulation											✓
Supporting Extended Investigations			✓								✓
Risk Tolerant Environment			✓								
Generic versus Specific Equipment					✓				✓	✓	✓
Hardware Reconfiguration					✓						✓
Software Reconfiguration			✓								✓
Supporting Multiple Investigators											

Human Observability and Manipulation. Humans help the *iterative research process* by reducing the time to get feedback (data download and comments on observed behavior) on experiment runs. Humans aid *extended investigations* by enabling simple ways to re-supply consumables and store hardware in between experiments. Humans provide for a *risk tolerant environment* in two ways: first, strict safety requirements reduce the risk of catastrophic failures; second, humans can intervene in the case of failures that would otherwise damage the facilities. Humans help support both *hardware reconfiguration* capabilities and *multiple investigators* by simplifying the methods to change the hardware and, in doing so, allowing the hardware to operate in different modes for different scientists.

Physical End-to-end Simulation. The participation of *multiple scientists* in a project usually implies that they are working on several areas of a technology. A system which achieves physical end-to-end simulation will provide better results for multiple scientists

working on different areas of a technology, since the system behavior will be valid for all of them.

Supporting Extended Investigations. The *iterative research process* depends on the availability of sufficient time and data for scientists to analyze results and make modifications to their initial hypothesis. Operating for extended periods of time provides *multiple investigators* with sufficient time to run their tests.

Risk Tolerant Environment. The *iterative research process* is helped by a risk-tolerant environment since tests can be pushed to their limits, rather than taking conservative steps every time.

Generic vs. Specific Equipment. By creating a complete set of generic equipment, the facility can be later reconfigured with specific equipment to simulate all the different aspects of a technology being investigated to create a *physical end-to-end simulation*. Finding the correct interfaces between the generic and specific equipment facilitates the *reconfiguration* of both hardware and software. The creation of generic equipment helps *support multiple investigators,* who can create their own specific components rather than have to work with equipment that does not necessarily meet their needs.

Hardware Reconfiguration. A *physical end-to-end simulation* needs to cover all aspects of a technology to be demonstrated; hardware reconfiguration enables physical changes to demonstrate the technology under different environments. The ability to change the hardware enables *multiple scientists* to configure the facility as necessary for their specific objectives.

Software Reconfiguration. The ability of software to change *facilitates the iterative research process* since modified hypothesis can be tested via data transfers, rather than hardware deliveries. Further, *multiple scientists* will investigate different parts of a technology, which will require different software.

At this point we note that *facilitating the iterative research process* and *supporting multiple investigators* do not necessarily benefit the other features, but rather are beneficiaries of them. Therefore, these two features are considered of a higher level than the other ones; they are major features that require other lower-level features to be present, but which ultimately provide the capabilities that are most desired of a facility.

Table 3.3 shows features grouped into sets that take into account their interactions and their support for other features. Facilitating the iterative research process and supporting multiple investigators are left as independent features that require individual attention. The *experiment support* group includes those features that support the ability to conduct experiments; all of these support the iterative research process as lower-level features of the facilities. The *reconfiguration and modularity* group contains the low-level features that help support multiple investigators. The following sections present the theoretical background behind these features.

TABLE 3.3 Grouping of the SSL Design Philosophy features

Group	Feature
Facilitating Iterative Research Process	Facilitating Iterative Research Process
Experiment Support	Data Collection and Validation
	Repeatability and Reliability
	Human Observability and Manipulation
	Supporting Extended Investigations
	Risk Tolerant Environment
Supporting Multiple Investigators	Supporting Multiple Investigators
Reconfiguration and modularity	Generic versus Specific Equipment
	Hardware Reconfiguration
	Software Reconfiguration
	Physical End-to-End Simulation

3.4.2 Facilitating the Iterative Research Process

> "Research is the methodical procedure for satisfying human curiosity. It is more than merely reading the results of others' work; it is more than just observing one's surroundings. The element of research that imparts its

> descriptive power is the analysis and recombination, the "taking apart" and "putting together in a new way," of the information gained from one's observations." [Beach, 1992].

The MIT SSL Laboratory Design Philosophy guides the development of laboratories for space technology research. To ensure success of the laboratory, the research conducted within must be supported by formal research methods that guarantee valid results as expected by the scientific community at large. The methodical procedure most widely accepted, although by no means defined in one single manner, is the scientific method. The most basic interpretation of the scientific method can be found in its dictionary definition [Merriam-Webster, URL]:

> Main Entry: scientific method
>
> : principles and procedures for the systematic pursuit of knowledge involving the recognition and formulation of a problem, the collection of data through observation and experiment, and the formulation and testing of hypotheses

A wide range of research exists on the philosophy of the scientific method as demonstrated by the large number of publications that reference the Scientific Method. A quick review of these publications demonstrates that a large portion of literature on the design of experiments ([Fisher, 1935],[Mead, 1988],[Antony, 2003]) concentrates on the use of statistics to provide useful results. Yet, a single definition of the principles and procedures that constitute the method applicable to all sciences and research does not exist. Every reference presents a slightly different procedure for the scientific method, based on their expected application. From the start of the scientific revolution, the scientific method was applied on a case by case basis. The method began in the fields of anatomy and physiology in the 17th Century; two versions of the method each called for starting research based on facts/observations, or on the development of theory (models). In the 18th century Newton joined the two concepts together, showing how a well developed hypothesis (theory) leads to relevant experimentation that helps develop a coherent theory.

We shall build upon the concept introduced by Newton. The goal of the research process shall be to validate a hypothesis by experimentation and modification of the hypothesis until the theory matches the physical world. The basic steps of this process are encompassed in an elementary definition of the scientific method as presented by Gauch in his introduction:

> "Elementary Scientific Method" [Gauch, 2003]
> - Hypothesis formulation
> - Testing
> - Deductive and inductive logic
> - Controlled experiments, replication, and repeatability
> - Interaction between data and theory
> - Limits to science's domain.

The basic method as presented above already supports our call to support the iterative research process, as it calls for the development of controlled experiments which can be replicated and repeated and the study between data and theory. But the basic method only implies the need for repetitions or iterations during the controlled experiments. As Gauch argues, the scientific method calls for a much deeper understanding of each step. He presents the steps shown in Figure 3.1 [Gauch, 2003] as the full scientific method. Of special importance to us is the fact that this advanced scientific method is iterative in its entirety. The development of the hypothesis leads to two paths: development of a model used in deduction of the science, and design of an experiment to observe and collect data from the physical world. His process introduces noise in data collection to remind the scientist that no observation is perfect, this step will be addressed in a later section. Next, Gauch calls for induction: the combination of the deductive theory and the observed data to determine the validity of the hypothesis. The last step closes the iterative loop: creating a hypothesis to test via deduction and observation.

The philosophy of science supports the need for iterations on the hypothesis and design of the experiment. The theory behind the design of experiments [Mead, 1988] calls for con-

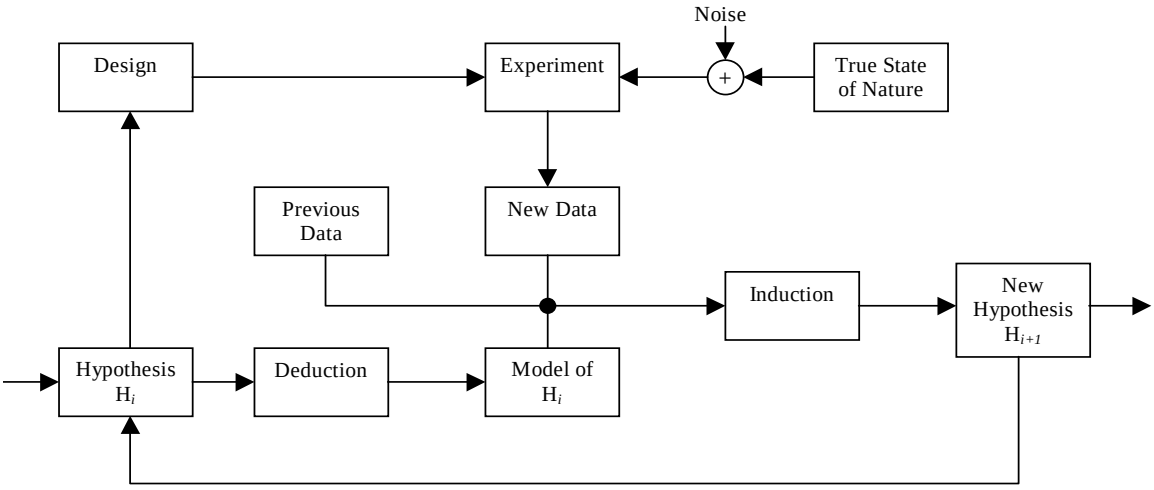


Figure 3.1 Overview of the scientific method by Gauch

ducting a number of experiments changing design variables in a controlled manner. The major references on the design of experiments concentrate on the topics of probability and the design of block methods [Fisher, 1935] [Montgomery, 1991] [Antony, 2003] to ensure full coverage of the design space. The concept of probability by itself implies the need to conduct multiple trials of experiments in order to obtain a meaningful set of data for analysis. Therefore, the design of a facility must allow researchers to conduct multiple tests in a repeatable environment with the ability to change the design variables of importance to the research:

> "The designer of an actual experiment is required to produce a design appropriate to a very particular set of circumstances. But, except where the designer is very experienced, he or she will not be able to assess the entire spectrum of design ideas, and make decisions about the design, from the basis of a comprehensive knowledge of how all the principles of design might relate to this particular problem." [Mead, 1988]

This concept further emphasizes the need to allow both a hypothesis and an experimental design to go through the iterative research process. Only through iterations will the hypothesis be understood and refined: "There are no hard and fast rules that lead to the selection of the best possible design for a given set of circumstances. The more one creates and evaluates designs, the better the chances of finding the best possible design. … We

have observed that beginners require something in the vicinity of eight to ten redesigns before their comfort level is reached. Experts require four to five designs." [Lorenzen, 1993]. The iterative research process is essential to the true understanding of a research topic and the formulation of its hypotheses and models.

The philosophy of the scientific method and the design of experiments defines what it means to iterate a research experiment: to be able to repeat an experiment multiple times changing variables so that statistically relevant data is obtained and to have the ability to change the hypothesis behind the experiment and re-design the experiment to account of these changes. Therefore, to facilitate the iterative design process, a facility must ensure that both of these activities are as easy to perform as possible. An environment that truly facilitates the iterative research process allows experiments to be repeated with minimal overhead. This includes the full process of conducting each experiment run: resetting the facility in the same state, controlling the initial conditions; ensuring that the experiment behaves the same way given the same disturbances and actuation commands; collecting valid data continuously; and allowing the replacement of any consumables with ease. The design of the facility must account for the correct number of times an experiment must be repeated to obtain meaningful data and ensure that number of repetitions is possible.

The facility also needs to account for the different variables that are relevant to the research being conducted. But, as expressed by Mead, not all design variables will be known. Therefore, the design of the facility must contemplate the need to change the known variables and to expect the appearance of new variables. Conducting the iterative research process will result in the identification of those new variables, and will likely require the design of the experimental setup to change. A well designed facility must allow those changes to take place with ease.

The need for iterative design is well summarized by Ernst as it applies to our field of space technology:

> "Doing exhaustive design ahead of time may not be desirable, even if it were feasible, because of uncertainty - and because of the certainty of change. (The more successful the design and the more long-lived the resulting system, the more change there will be; thus, a successful design will eventually become less appropriate, less clear, and less true to its original conception.) … Few designs perfectly model the constructed system; artifact understanding and design recovery are crucial in such circumstances. … the implementation might have begun before design was complete, or might use pre-existing or separately constructed components that may not have their own designs or may not mesh perfectly with the overall system's design.
>
> "Iterative design encompasses design after part of an artifact has already been completed; re-design; design in the presence of changing requirements; and adjusting a design in response to changes to an artifact. … iterative design must take account of, and respect, existing components and their interactions. Iterative design goes further in comparing versions of requirements, designs, systems, and in being part of a continuing process…" [Ernst, 2003]

3.4.3 Experiment Support Features

The iterative research process depends on the ability to successfully perform experiments, collect data, interpret it, and then iterate on the hypothesis. Returning to the idea that the design of experiments is highly dependent on the statistical relevance of the collected data, it is further necessary that scientists be able to perform a relevant number of experiments in between each iteration. This group of features addresses the need to ensure individual experiment runs are effective and provide the right data.

Data Collection and Validation

The initial definition of this feature called specifically for the following requirements on data collection:

- Ensure data accuracy and precision scalable to the final system
- Ensure observability of the technology
- Provide a useful presentation of data
- Allow for a truth sensor

These requirements address a range of important practices in data collection. To ensure data accuracy and precision, the experimental setup must address the issues of frequency response/aliasing, bit depth/digital precision, and input/output ranges. The collection of data must be made at the necessary bandwidth to satisfy two goals: first, that the data is relevant to the technology being demonstrated. For example, saving data at 1Hz is not useful to demonstrate a controller for a 10Hz system, unless the frequency can be scaled, in which case scalability must be demonstrated. Second, the data sampling rate and/or hardware must ensure that aliasing does not occur. Scientists must also ensure to use the correct precision of the data. The bit depth affects two parts of the data collection process: conversion of analog signals and saving the data itself. The conversion to and from analog signals must be of sufficient bit depth to ensure that the single-bit precision of the data shows the necessary fluctuations in analog signals. When saving data, whether they originated in analog or digital form, the scientist must balance the number of bits used for each piece of data with the storage volume and communications bandwidth available to the experiment. It may not always be possible to save an analog measurement in floating-point format, and therefore the scientist must decide to what fixed-point precision the data should be saved. Lastly, the experiment must be such that the range of the inputs and outputs is able to both measure and actuate the system to ranges scalable to the final system. For example, if an experiment can only provide a limited amount of actuation from a reaction wheel, the scientist must ensure that the actuation is scalable to a larger satellite.

One of the characteristics of a mature technology is that it allows the identification of new physical phenomena that affects a system. To allow this identification the data must ensure that the system is fully observable, so that the scientists can demonstrate where the new phenomena originated. For example, in a dynamics and controls experiment, the scientist may be able to exactly reproduce the output (actuator commands) of a controller by knowing the inputs (sensor readings), since the controller is a deterministic mathematical algorithm. In developing that controller the scientist may have assumed some noise in the input. In models the scientists can use random noise generators, but the modeled noise may not correspond to the physical system. It is necessary that the saved data include the

measured noise, otherwise the scientist would not be able to confirm their model. Further, the scientist may discover an unknown coupling with the noise observed during the experiment.

To truly support experiments and ultimately facilitate the iterative research process it is necessary to easily interpret the data. This means not only that the right data must be downloaded, but also that tools must be created a-priori to evaluate the results. While a scientist may not know exactly in what format the data will need to be presented to learn all the information contained within it, the scientists should be able to identify the basic requirements. During the data analysis time the scientist should only create new data analysis tools when new phenomena are identified that need further examination, and not to interpret the basic data.

Lastly, there is a need for a truth measure which can verify the validity of the data acquired by the system when the sensors in an experiment are part of the research itself. A truth measure helps to identify any couplings of the sensors with the system being tested and to ensure that these sensors are operating correctly. For this it is critical that the truth measurement systems operate independently of the experiment itself. For example, in a closed loop control experiment one should not use the feedback sensors to measure control performance; sensors outside of the control loop should be utilized as a truth measure.

The capabilities of the truth sensor in terms of accuracy and precision depend on the specifications and requirements of the sensors which need validation. Especially in those cases where the whole experiment is the development of new sensors with precision beyond existing systems, the truth sensor will not be able to verify the operation of the new sensor to its utmost precision. When a higher precision truth sensor is not available, the use of redundant sensors is desired. This would allow multiple ways to calculate performance and asses the variability in the performance estimates. Data from different sensors should correlate, helping to validate the data.

Repeatability & Reliability

The need for repeatability is directly supported by the theory behind the design of experiments (DOE). The goal of a DOE process is to select one of two methods: either have a large number of samples to show statistically useful results, or ensure that the small number of samples demonstrate the success of the technology. Repeatability is defined by the International Organization of Standards (ISO) as:

> The closeness of agreement between independent results obtained in the normal and correct operation of the same method on identical test material, in a short space of time, and under the same test conditions (such as the same operator, same apparatus, same laboratory).

Repeatability means more than the ability to run multiple tests. For the results to be statistically useful, each time a test is run the operating conditions must be the same. To further benefit an experiment, a facility should allow complete system identification and/or control of the operating conditions of the test. When a facility allows measurement of the operating conditions, the scientist can trade-off between obtaining samples for identical controlled initial conditions, which could utilize a large amount of consumables, and running multiple tests with a large number of known but different starting conditions.

The reliability of a system is defined by ISO as:

> The ability of an item to perform a required function under stated conditions for a stated period of time.

The number of samples needed for a successful demonstration is inversely proportional to the reliability of the test. If a test is expected to succeed with high reliability, during DOE a small number of samples are planned. Low reliability experiments will require more samples. A research facility must ensure that it is not the driver in the selection of the number of samples. The reliability accounted for in the DOE process must be that of the technology being tested, with the security that the reliability of the testing facility is high.

Because the goals of these facilities is to test the limits of the new technologies, the development of the facilities must assume that the technologies will fail and new tests will be

needed. Therefore, the facility must be able to withstand failures of the technology without any critical failures of its equipment. The facility must be more repeatable and reliable than the technology being tested.

Human Observability and Manipulation

The review of antarctic and ocean research in Chapter 1 emphasized the need for humans to be present in the research environment. Humans in the Antarctic and below the sea are scientists, engineers, and mechanics; they ensure science occur and equipment works.

> "The most complex system cannot effect the simplest repair unless the particular failure mode has been foreseen and preprogrammed. An unmanned camera will happily shoot film when a dragging boom puts only bottom silt before the lens, and many manipulative functions are just best left for the human hand. There will always be the unexpected on a new frontier, and instruments are best regarded as extensions of man, reserved for areas where man cannot reach or function." [Penzias, 1973]

With the availability of humans, one must define what the human tasks should be. Human observability and manipulation of an experiment requires that humans control the experiment in several ways. The observation of an experiment means that there is a clear ability of the human to determine the progress of the test. In many cases it can be to visually observe the physical behavior of the system. Observation can also be the interpretation of results shown in real-time, such that the human can observe the progress of the experiment as it progresses. The critical element of human observation of an experiment is that the human obtains real-time feedback on the progress of the test, whether directly or indirectly. Manipulation of an experimental facility is composed of two parts. First, humans must control the operations of the experiment. While the facility's normal operations can be automated, the facility must allow override of such systems, so that a human can ultimately make the decisions on the progress and safety of a test. Because we are working with immature technology, which we expect to fail in many cases, a human should ultimately control when a test starts and ends, ensuring that the conditions to run the test are appropriate. Second, allowing humans to modify the system, either by reprogramming or changing hardware, can present considerable functionality and cost savings to the project.

Past experiences of the MIT SSL with microgravity experiments have demonstrated the success of human manipulation to help a mission. The SSL has benefited from the ability of humans to repair components of experiments which would otherwise have terminated the mission prematurely. The failure, an incorrectly wired connector which was not part of the original checkout procedure, was fixed on site by astronauts. The ability of humans to reconfigure the hardware of several experiments has allowed the SSL to proceed with those missions. Had the hardware needed automatic reconfiguration two problems would have occurred. First, the cost would be prohibitive for the program, forcing a substantial reduction in mission objectives. Second, the addition of motors and other physical elements would have complicated the structural components of the facility, changing the dynamics in ways incompatible with the mission goals.

Support Extended Investigations

The support of extended investigations does not refer to the ability to run individual tests for a long period of time. Referring back to the scientific process shown in Figure 3.1, the scientist needs time for induction - analysis of the data - and review of the hypothesis, with the ability to perform a new iteration shortly after the new hypothesis is created. Therefore, the support of extended investigations refers to the ability of a facility to allow storage of an experiment in safe conditions after a number of tests have provided enough data to iterate on the hypothesis. After the hypothesis has been modified, the experimental apparatus must be able to perform new tests in minimal time. Repeatability and reliability play a role in this feature, since it is expected that the new tests perform under conditions similar to those conducted originally.

Risk Tolerant Environment

The MIT SSL Laboratory Design Philosophy has been created for the maturation of new technologies. This implies that the technologies are not yet mature, and that before they are mature tests are likely to fail. The Innovation Network, when presenting the challenges

of organizational innovation, provides a good summary of the need for a risk-tolerant environment to allow for maturation of untested technologies:

> "an environment that welcomes and continuously searches for opportunities -- one with a rich flow of ideas, information and interaction within and without the organization... among customers, the environment, competitors, suppliers and employees at all levels and functions. This is a risk-tolerant environment that celebrates successes as well as great tries that didn't work." [Wycoff, URL]

To truly allow for new technologies to be developed, the environment must be designed to accommodate failure or unexpected behavior; it should welcome failure as much as success. To achieve this, the environment must ensure that its operation never poses harm to the researcher, and that failures of the technology do not cause critical failure of the apparatus, while at the same time ensure that the controls put in place for this safety do not inhibit the research process.

3.4.4 Supporting Multiple Investigators

> "The most compelling rationale for engaging in collaborative relationships... is the advantage an organization accrues by gaining access to complementary areas of expertise, knowledge, skills, technology, or resources that it cannot produce on its own. Most researchers on strategic alliances concur that the value added from collaboration comes primarily when partners have complementary needs and assets... Consortia are advantageous when the knowledge base of an industry is both complex and expanding, the sources of expertise are widely dispersed, and the pathways for developing technology are largely uncharted." [Merrill-Sands, 1996]

Aerospace technology clearly lies within the industries that address complex problems. The advancement of microgravity technologies to full operational level, if we are to follow NASA TRLs, depends on the ability to demonstrate these technologies with a full system test in a relevant space environment. Therefore, the maturation of a space technology depends on the demonstration of its ability to integrate and operate with all the sub-systems of a spacecraft. For example, we can easily identify the needs for propulsion, avionics (navigation, control, and data processing), communications, thermal, and structures sub-systems. Advancing a technology in the area of dynamics and control may depend on

advanced propulsion and structures technologies. Even a specific area may cover a wide range of studies; for example the area of controls, within avionics, requires sensors (metrology), data processing (and control theory), and actuators. The inter-dependence of all these areas are vast and deep. As such, collaboration has a high potential to benefit the advancement of space technologies and is essential to fully advance technologies for integration into new spacecraft.

Explaining how to best conduct collaborative research requires that the term be first defined in a concise and clear manner. One such definition is presented by the FENIX team: "*Collaborative research is defined as an emergent and systematic inquiry process embedded in a true partnership between researcher and members of a living system for the purpose of generating actionable scientific knowledge*" [Adler, 2004]. This definition consists of several parts, each of which presents its own challenges to collaborative research. The need for the process to be "emergent and systematic" requires that the collaborative process be in constant review and update. For the process to be embedded in a true partnership it must have been designed as an integral part of the research process at all levels, rather than only being a high-level process. Lastly, the definition calls for the results to produce actionable scientific knowledge; the results must provide the partners with new knowledge that have a practical use for each of the partners that entered into collaboration. Huxham illustrates this last point best:

> “Collaborative advantage will be achieved when something unusually creative is produced that no organization could have achieved on its own and when each organization, through the collaboration, is able to achieve its own objectives better than it could alone.” [Huxham, 1996]

With the concept defined, it is now possible to identify the challenges and different methods to enable collaborative research. Several past collaborative experiences [Merrill-Sands, 1996] [LeGris, 2000] demonstrated that an important challenge is that collaborative projects have higher management costs. Control of the project is shared among multiple entities, and division of responsibilities usually have to be negotiated. These

challenges are best addressed at the start of the collaborative program. [LeGris, 2000] proposes the following steps to successfully initiate a collaborative effort:

1. Determining the relationship: define the goals, ensuring that they do not interfere with each organization's primary purposes. The definition also includes expected participation by staff of the different institutions.
2. Determine the structure: set meeting schedules and ensure visibility of the collaborative effort as appropriate through the organizations.
3. Assessing the organizational climate: before implementing the collaboration, ensure that the different parts scheduled to work together through 1 and 2 are ready to participate.
4. Recognize similarities and differences: take into account the different goals of the parties involved; researchers value the process, while industry values the product. Identify these differences and address them in the definition of the collaboration. Find common elements, such as quality management, to help bridge the differences.
5. Enhancing commitment through communication: ensure that the full staff of each organization which will be involved in the collaboration is aware of the project.

This process puts heavy emphasis on the need for all involved parties to be aware of the collaboration that will take place and be comfortable with it. [Davenport, 1999] condenses these ideas into the building of trust between the different organizations. Three types of trust are defined:

1. Contractual trust: adherence to agreements
2. Competence trust: adherence to expectations and performance
3. Goodwill trust: mutual commitment to the partnership

Only through goodwill trust can a relationship continue over the long term. "Cooperation between academic institutions and industry will be more likely to survive over time, the more there are initial assets of good will, trust, favourable prior beliefs, mutual psychological commitment and prior relations between the parties."

The need for substantial communications between the partners presents another challenge:

> "Collaboration requires frequent communication among all involved parties. The likelihood of success is greatly enhanced by the presence of a product or collaboration champion." [Littler, 1995]

[Kraut, 1988] presents a study on the effects of physical proximity on scientific collaboration. The study summarizes data on the amount of cooperation between researchers based on their physical location with respect to each other; next, it presents how the use of technology can help achieve that virtual presence. The study emphasizes the need for informal communications. The communication frequency and quality were proportional to the success of collaborations. On the other hand, high costs of communications greatly hindered the collaborative process. The study concludes that:

> "Omnipresent video might provide the low-cost and therefore frequent and spontaneous interactions that are crucial to initiating collaborations, monitoring and coordinating the project, and maintaining a smooth personal relationship. Multimedia meeting tools might provide the high quality communication to support planning and review. While many other specific tools have been proposed and could be built to support particular tasks that occur frequently in a collaborative project, most are likely to build from these two foundations."

Communications in a collaborative environment involves more than personal relationships, they also require successful data exchange. Therefore, further tools are required beyond video conference and multimedia. Projects such as the Electronic Laboratory Notebook (ELN) [Myers, 1996] [Myers, 2001], and Collaborative Experimental Research Environment (CORE) [Schur, 1998], and Jazz [Hupfer, 2004] are geared to support the data handling of collaborative research. The electronic lab notebook project is based on an important premise:

> "The laboratory notebook is a vital tool in scientific research. It is the central repository of information about the reasoning and preparation behind experiments, about the analyses done to obtain results, and about plans for future research. The notebook captures the scientific process that gives meaning to a scientist’s observations. Sharing a notebook can help collaborating researchers build a common understanding of their work." [Myers, 1996]

The ELN consists of an internet based website which collects and presents data information as well as annotations. It can interface to data collection programs such that the data is placed directly in the website; the website handles threading of annotations, such that comments of a similar topic remain in the same thread, rather than being forced into a chronological pattern. The notebook also maintains a database of arbitrary electronic files, such that researchers can share data even with programs that did not exist during the development of the ELN. The ELN also provides query functions to provide concise reports by researcher, topic, etc.

CORE "provides a loosely integrated suit of Internet collaboration tools that appear as web-browser extensions... The goal... was to develop a system that would support the identified workflows, activities, and different collaboration types." The tools that form CORE are:

- Chat
- Audio/video conferencing
- Whiteboard
- File transfer
- Shared computer display (tele viewer)
- Electronic notebook
- Web browser synchronization
- Shared instrument control

CORE grows upon the ELN by not only providing space for results and procedures, but also the highly-interactive tools that enable inter-personal communications. CORE goes as far as to provide shared control of instruments.

The Jazz team bases their design on *contextual collaborations*, an approach where collaboration is enabled by expanding standard applications, rather than having to use special tools. The benefits of contextual collaborations include: reduced friction in the use of applications; enhanced collaborative work by easing the collection of collaborative artifacts: better informed collaborative work since researchers are more aware of the process

while they conduct research, not only when they use the special tools; and reuse of collaborative components used on a wide range of standard applications. The Jazz project is based on the metaphor of an "open office", where developers communicate easily and use shared resources such as a whiteboard. The interface for the development environment integrates teams and team-members into standard one-user projects. Each team-member is immediately aware of the other team members with only minor changes to the interface, and without hindering their own work. The interface also integrates live-chat sessions into the IDE, but presents them in separate windows to prevent taking space from the main window. Through these simple but highly-integrated tools, Jazz allows a standard programming IDE to inherently support collaboration.

So far the challenges for collaborative research have concentrated on programmatic and inter-personal communications, but for space technology maturation there exists one other important aspect: sharing of the available facilities. The CORE project addressed the issue of sharing expensive/limited instruments by creating the toolkit for "secure collaborative instrument control" as part of the main application. Part of their research concluded that "some researchers were concerned about how their roles on research projects would change. For example researchers local to instruments voiced concerns about becoming technicians for remote users and no longer sharing physical maintenance tasks." The results of using CORE proved positive, and actually helped scientists have more time for research as the remote scientists used the tools without needing the constant help of local scientists. But the project concentrated on the creation of tools to control existing instruments rather than on the development of new instruments inherently designed to be shared by collaborating scientists. The development of new research facilities for the space station allows for new hardware designs that inherently support collaboration.

> "Would it not be better to build an entire family of... common product technologies, and a common set of highly automated production processes? Rather than have separate development teams each working on single products, wouldn’t it be better to have them join forces in building a common *platform* or a design from which a host of *derivative products* could be effectively and efficiently created?" [Meyer, 1997]

The development of hardware to support multiple scientists must consider the common parts that will help all of the researchers, and allow for the development of specific equipment to support them. Following the idea of product platforms used in industry, this means that the main developer of the project must identify those common features and build them. Individual researchers should then be able to use this common equipment to implement their specific science. The basic common elements should define the overall architecture of the experiment, while individual experiments are derivative products. Ultimately, the goal is for developers of new ISS experiments to become the *supplier* of a *product platform*, while the collaborator scientists who use the experiment become the *customer* and create the *derivatives*. Note that the idea of product platforms is not limited to hardware; software can also become a platform. This idea is further explored in defining the concepts of modularity and reconfiguration.

Several key points arise from this review of collaborative science:

- For collaborative science to be effective it must allow each individual organization to achieve goals they would otherwise not be able to do on their own.
- A systematic approach to enabling collaborations is essential. This process must at the very least address:
  - Definition of the goals & structures of the collaboration
  - Trust between the parties
- Both inter-personal and data communications play an essential role in the success of collaborative endeavours
- New experiments developed for collaborative research must support multiple investigators by design; it is essential to identify the common elements of the project and allow individual scientists to add their own components

Successful collaboration provides benefits for all parties involved. If collaborative research is included as an integral part of a program, then it will have a high probability of success.

"**Everyone Wins**

> "Our open collaborative model provides benefits for all participants. It allows university researchers to amplify their thinking and their work - and potentially see it translated into commercial products - without having to leave academia. It enables Intel to accelerate research in areas we find interesting and worthy of exploration, by conducting research concurrently in the labs and deeper without our company. By facilitating synergy and open exchange of ideas, the model enables Intel and participating universities to jointly lead the industry toward breakthroughs that will continue to advance the state of the art. Under this new model of industry-university research, we believe everyone wins." [Intel, 2003]

3.4.5 Reconfiguration and Modularity

Reconfiguration and modularity affects both higher level tasks to support the iterative research process and multiple investigators. The need for reconfiguration and modularity is exhibited strongly by the philosophy of the scientific method, presented above, which includes as a critical element the need to revise the hypothesis and implement the changes for further experimentation. Supporting multiple investigators depends on the ability of the facilities to provide common parts and the individual researchers to create their specific equipment.

The idea of reconfiguration is closely linked with several studies on the need for flexibility of a system. [Saleh, 2002] proposes a definition of flexibility which applies to our case:

> "The property of a system that allows it to respond to changes in its initial objectives and requirements - both in terms of capabilities and attributes - occurring after the system has been fielded, i.e., is in operation, in a timely and cost-effective way."

That definition is further detailed by comparisons with other terms which are usually confused with flexibility, but which do not guide a products towards reconfiguration or modularity:

- Flexibility vs. Robustness - robustness is the ability of a system to satisfy a fixed set of requirements despite changes in the system’s environment. Flexibility satisfies changes in requirements after the system has been fielded.

- Flexibility vs. Universality - universality applies to a system that can be used in a wide range of situations without any changes. Flexibility implies the ability to change and adapt with ease.

These definitions clearly point towards a reconfigurable system, one which can adapt not only to changing conditions, but also to changing requirements. At the same time, the distinctions indicate that the goal is not to create a single facility that satisfies every foreseeable need, but rather one that can adapt to unforeseen needs.

The MIT SSL Laboratory Design Philosophy separates the concept of reconfiguration and modularity into four main areas: identification of generic and specific equipment, hardware reconfiguration, software reconfiguration, and physical end-to-end simulation.

Generic versus Specific Equipment

A ground-based laboratory usually includes a set of generic equipment used for the design and construction of multiple projects. For example, a laboratory for electronics includes oscilloscopes, multi meters, computers, soldering irons, wire, pliers, and even a set of generic electronic components such as resistors, amplifiers, and standard logic chips. The MIT SSL Laboratory Design Philosophy defines a laboratory as a place to enable a field of study, therefore the equipment referred to in the philosophy is that which enables the research. The ground-based generic equipment is utilized to create higher-order generic equipment specifically designed to aid in the research of the specified field in a microgravity environment.

The support of multiple investigators introduced the concept of a *product platform* as a model for the development of research facilities where the different researchers are the customers who decide the *derivative* products needed. This concept directly fits the idea of generic versus specific equipment. The definition of product platform as presented by *[Meyer, 1997]* is:

> A product platform is a set of subsystems and interfaces that form a common structure from which a stream of derivative products can be efficiently developed and produced.

The definition first considers the subsystems and interfaces of a larger product. These subsystems are brought together to create a platform for the development of derivative products. The generic equipment of a microgravity laboratory should be composed of those sub-systems which can create such a platform, and provide a set of well-defined interfaces so that the specific equipment can be efficiently added to the generic equipment to form a complete product (experimental apparatus).

Meyer cautions on the distinction between platforming and standardization, a warning relevant to this philosophy. Standardization of all parts fixes too many aspects, ultimately resulting in an inflexible design of the platform. The use of standards should always be balanced with the ability to identify the special elements of a platform that add value to the overall project, and allow those parts to continuously be improved. In relation to the MIT SSL Laboratory Design Philosophy: selection and standardization of the generic equipment should not limit the capabilities of the specific equipment.

It is important to point out that generic/specific equipment can exist for both hardware and software; generic equipment is not limited to hardware implementations.

Hardware Reconfiguration

Hardware reconfiguration works in parallel with the concept of generic vs. specific equipment; still, they are different concepts. While the addition of specific hardware to the generic setup does in fact reconfigure the overall facility, hardware reconfiguration refers to the ability to change the hardware for a specific test. In the area of dynamics and control, for example, the hardware configuration of a test apparatus directly affects the results. Changing the hardware configuration means that the dynamics of the system being tested will change. This is sometimes desirable, for example, in order to demonstrate robustness of an algorithm. In these cases both the generic and specific equipment may change configuration, meaning that the ability to reconfigure hardware should be considered in the design of all parts of the facility.

Software Reconfiguration

The concepts of flexibility and platforming apply equally to software as they do to hardware. Software has become an important part in the implementation of algorithms. The software controls the behavior of the hardware, sometimes commanding the hardware itself to change. Therefore, in order to complete full cycles of the iterative research process and to support multiple investigators, the software of a system must be able to change.

Modular software or that used in collaborations is usually shared using the concept of data abstraction; each module is a *black box* with certain inputs and outputs, and the user does not know what happens inside the box. The interfaces to these black boxes are commonly called Application Programming Interfaces (APIs). An API explains the functionality of the module and defines the inputs and outputs for each module and allows a programmer to use those functions without knowing the actual implementation. There are both positive and negative aspects to this modularity.

The concept of data abstraction flows directly into the idea of product platforms, as long as one accounts for the programatic aspects. From the product platform point of view software should be based on a core set of modules which interface through API to plug-ins that provide the specific functionality. The APIs should be standardized, such that changes to the core system can be continuously performed to maintain the platform up to date, while not breaking the functionality of the more specific modules:

> "One set of... engineers has been constantly building new add-in modules for the current version of the product platform or engine. Concurrently, other teams have worked to renew the core platform and to embrace technological change occurring in the broader industry... The only way that such a smooth migration can be accomplished is to develop and sustain clear, robust interfaces between the underlying engine and the add-in modules" [Meyer, 1997]

[deSouza, 2004], on the other hand, points out how APIs can hinder collaboration of multiple scientists. Sustaining clear and robust APIs is a major challenge; real world experi-

ences show that APIs are usually unstable. Current programming tools do not help account for changes in APIs, leaving programmers dependent on personal communications to account for the changes. APIs are also challenged by their inability to ever be truly complete. As one programmer works on their base function, the users work with previous versions. Lastly, the black box concept creates a lack of awareness among the different people working in the same project. Data abstraction causes people to make unsupported assumptions about the functionality implemented in other modules.

The need to support iterations and multiple scientists requires that facilities provide software reconfiguration. The correct use of software reconfiguration can lead to the development of a good platform where multiple scientists can implement their own plug-ins for specific research. But this development must ensure that APIs do not hinder the collaboration efforts by abstracting too much information into an interface.

Physical End-to-End Simulation

The previous features allow multiple scientists to perform specific experiments in an iterative environment, but with what goal? The ultimate goal of the MIT SSL Laboratory Design Philosophy is to allow technology to maturate. A critical part of the technology maturation is to operate the experiments in a relevant environment:

> "Relevant environment" is a subset of all the "environments" to which the technology advance will be exposed. "Relevant environment" is defined to be that environment, operating condition, or combination of environments and operating conditions that most stresses the technology advance and is consistent with that expected in the spectrum of likely initial applications. It is to be delineated in detail with the appropriate NMP Project Manager and concurred by the NMP Program Manager. [NMP, 2003]

A review of TRLs [Graettinger, 2002] for software projects further defines what a relevant environment means. For example, in the case of TRL5 the following definition is used:

> SW: Reliability of software ensemble increases significantly. The basic software components are integrated with reasonably realistic supporting elements so that it can be tested in a simulated environment. Examples include "high fidelity" laboratory integration of software components.

> System software architecture established. Algorithms run on a processor(s) with characteristics expected in the operational environment. Software releases are “Alpha” versions and configuration control is initiated. Verification, Validation, and Accreditation (VV&A) initiated.

Requiring an experiment to fulfill end-to-end simulation means that the experiment includes the necessary sub-systems and operates in the correct environment to provide realistic operations. No critical elements of a program can be missing in the tests, otherwise the experiment does not satisfy being an end-to-end simulation and the technology cannot advance.

To satisfy end-to-end simulation the other reconfiguration and modularity features can be used. As individual scientists create their specific equipment they are ensuring that the test apparatus simulates their experiment in a valid way. Allowing hardware and software reconfiguration allows scientists to test a wide range of operational environments and conditions. Being able to demonstrate that a facility allows end-to-end simulation ensures that successful tests lead to technology maturation.

3.5 SSL Experiments and the Laboratory Design Philosophy

Table 3.4 serves two purposes. First, it cross-indexes the past laboratories of the MIT SSL with the attributes that they contained. As shown in the table, the more basic attributes such as data collection, repeatability, separation of test-specific from generic hardware, and hardware reconfiguration were introduced in the earliest laboratories (MODE) and adopted in subsequent designs. The more advanced attributes such as software reconfigurability, facilitating the iterative research process through human observation and data downlink with uplink of refined algorithms, and multiple guest investigators were not introduced until later.

Second, the table shows the goal of the SPHERES project with respect to the principles. As shown, the requirements for SPHERES are to meet all the features: support experi-

TABLE 3.4 Past Experiments and the philosophy features

	Data Collection	Repeat. / Reliab.	Iterative Process	Human Obs./Man.	End-to-End	Extended Invest.	Risk Tolerant	Generic/Specific	HW reconfig.	SW reconfig	Multiple Invest.
MODE	✓	✓						✓	✓		
MODE Reflight	✓	✓						✓	✓		
DLS	✓	✓				✓		✓			
MACE	✓	✓	✓	✓			✓	✓	✓	✓	
MACE Reflight	✓	✓				✓	✓	✓	✓	✓	✓
SPHERES	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

ments and provide enough flexibility to ensure that the iterative research process is facilitated and multiple guest investigators are supported.

Chapter 4

THE SPHERES LABORATORY FOR DSS RESEARCH

New scientific and economic objectives are creating a demand for small satellites capable of autonomous formation flight. Current missions, such as separated spacecraft interferometry, require large diameter synthesized apertures than today's monolithic satellites can provide since they are limited in size by launch and deployment capabilities. A group of smaller satellites flying in an array would provide the desired improvement; furthermore, the smaller size of individual satellites and the increased modularity of a constellation system would result in a reduction of launch and maintenance costs.

SPHERES is designed to create a testbed to demonstrate the viability of autonomous formation flight control algorithms. SPHERES provides a facility with six degrees of freedom to evaluate the dynamics of a multiple satellite system. It also tests the ability of a constellation of independent objects in a microgravity environment to interactively communicate, maintain position, run diagnostics, regroup after disturbances, and move to commanded locations.

SPHERES was designed specifically for operations in the ISS, following the guidelines of the MIT SSL Laboratory Design Philosophy presented in Chapter 3. Because of direct involvement and deep knowledge of SPHERES, it is a primary candidate to demonstrate how to best implement the features of the philosophy and ensure the best use of the ISS.

This chapter first presents the science requirements of SPHERES as well as the different constraints imposed by operations aboard the ISS. A introductory description of the design of the facility and its sub-systems follows. Next, the chapter discusses how the different sub-systems of SPHERES implement the features called for in the design philosophy.

4.1 SPHERES Problem Statement

The primary goal for SPHERES is to create a testbed for the development of formation flight and docking algorithms for separated spacecraft systems. The requirements were based on both the primary goal and the MIT SSL Laboratory Design Philosophy. The first step in the design process was to develop a clear set of requirements for the facility, and understand the constraints of the operational environments.

4.1.1 SPHERES Requirements

The initial design of SPHERES had its first milestone in the Spring of 1999 when the undergraduate senior class presented a preliminary design. The science requirements realized from the initial problem conception are [SPHERES, 1999]:

1. Develop a set of multiple distinct spacecraft that interact to maintain commanded position, orientation, and direction.
2. Allow reconfigurable control algorithms, data acquisition and analysis, acquisition of a truth measure.
3. Enable the testbed to perform array capture, static array maintenance under disturbances (attitude control and station keeping), and retargeting maneuvers.
4. Enable testing of autonomy tasks, including fault-detection and recovery, health and status reporting, and on-board replanning.
5. Ensure traceability to flight systems via communication, propulsion, structural, avionics, guidance, control, and power capabilities.
6. Design for operation in the KC-135, shuttle mid-deck, and ISS.

These requirements reflect both the mission objective and the guidelines presented in the MIT SSL Laboratory Design Philosophy. The first, third, and fourth requirement directly

relate to the mission objective of creating a formation flight development facility. The second, fifth, and sixth requirements evolved from the philosophy so that the resulting facility enables technology maturation. Requirements three and four define individual areas of formation flight technology, providing initial insight into the different tasks which are required to demonstrate maturation.

4.1.2 ISS Constraints

SPHERES had to meet constraints for operation aboard NASA facilities. The initial design was considered for flight in the Shuttle Middeck, while the final flight configuration was designed for operations about the ISS "Unity" node and/or US "Destiny" laboratory space. This section presents the main constraints imposed by operation of SPHERES within the ISS, where it was certified for operations.

1. **Crew availability.** While the ISS presents the only space where humans can interact with μg experiments over an extended period of time, crew availability is limited. Even when three humans manned the ISS, SPHERES has been allocated only one US astronaut for operations - therefore SPHERES has to be operational with the supervision of one human.
2. **Safety requirements.** The ISS safety panel imposed strict margins on the safety of the SPHERES satellites and support hardware. These included:
   - Structural - the structure had to withstand a specified impact and prevent shatter; all edges had a minimum radius requirement
   - Compressed gases - any compressed gases needed safety factors from 1.5x up to 3x the operational pressures; triple hardware redundancy was required of all pressurized elements (i.e., two serial hardware failures could not pose a danger to the astronauts or the ISS)
   - Power systems - all electrical power systems require double redundancy from causing harm to the astronauts or the ISS; the major concern in power systems is the start of fires due to uncontrolled currents
   - EMI - electromagnetic emissions were closely monitored; this included a limited range of RF frequencies available for free use
   - Software - any safety-critical software had to be NASA certified
3. **Volume**. The volume of all SPHERES hardware was originally limited to fit within one Middeck Locker Equivalent (MLE). The requirement evolved over time. The need for each individual unit to fit within an MLE remained a

hard constraint; the need for all the hardware to fit within one MLE became a soft constraint.

4. **Mass**. The mass of the SPHERES hardware was also constrained to that allowed in one MLE. As with any space project, the total system mass should be minimized to increase the launch possibilities and reduce cost.
5. **Operations.** The ISS requires remote operations of the satellites, with no direct communications or control from ground. Further, the majority of the test sessions will be conducted independently by the astronauts; real-time communications with the astronauts is only expected in the first two test sessions.

4.2 SPHERES Design Introduction

The SPHERES laboratory for distributed satellite systems consists of five nano-satellites (Figure 4.1), metrology and communications hardware, a researcher interface, an astronaut interface, and a guest scientist program to allow multiple researchers to use the facility. In its final configuration, three of the satellites will be aboard the ISS, where the astronauts will conduct tests in 6DOF. Two units will remain in the ground facilities of the SSL where MIT researchers will tests algorithms prior to up-link to the ISS. The guest scientist program provides a simulation which allows researchers outside of the MIT SSL to develop their initial algorithms in house.

Figure 4.1 The five flight-qualified SPHERES nano-satellites

SPHERES was designed specifically for operation in the shirt-sleeve environment of the MIT SSL laboratory (3 DOF), NASA's KC-135 reduced gravity airplane (short duration 6 DOF), and the International Space Station (long duration 6 DOF). The KC-135 and ISS environments provide 3-D environments to test algorithms that may be directly applied to real satellites. The additional laboratory environment at the MIT SSL enables 2-D experiments to be performed before testing on the KC-135 or ISS, thereby reducing the cost and risk to develop and verify algorithms in the ISS. Figure 4.2 shows an operational concept for SPHERES: the scientist first develops their algorithm using the simulation; the algorithms are then sent to the MIT SSL where tests are conducted with flight hardware in 2D; once tested, the algorithms are sent to the ISS for 6DOF tests.

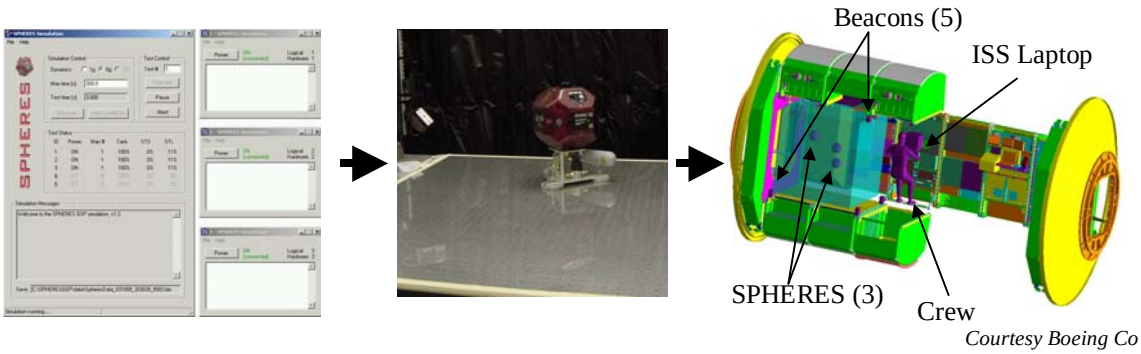


Figure 4.2 SPHERES operational concept

Operating SPHERES in one of the mentioned operational environments requires the following hardware components:

- one to five satellites
- five metrology transmitters
- a communications transceiver
- multiple battery packs
- multiple gas tanks
- a computer with a SPHERES graphical user interface

These hardware elements comprise the package of components which will be delivered to the ISS (a standard NASA supplied laptop is used aboard the ISS, only the software is delivered). The ISS rendering of Figure 4.2 shows this setup graphically. The three satellites, which use the battery packs and gas tanks, will operate inside the blue cube region. The five metrology transmitters, placed on the corners of this region, define the 3D frame of reference. The communications transceiver attaches to the computer via a serial port to store telemetry data, uplink programs to the satellites, and send commands to control tests.

During the design phases of SPHERES the team had to determine a set of requirements that ensure future algorithms will run in the testbed and provide significant results. These requirements include the precision, accuracy, and operational ranges of sensors and actuators, processing power, operational lifetime, and communications bandwidth. Table 4.1 summarizes the resulting quantitative "straw-man" requirements.

TABLE 4.1 SPHERES quantitative operational requirements

Item	Requirement
Translation (1m start to stop)	5s
Rotation (360° start to stop)	5s
Translation accuracy	0.5cm
Rotation accuracy	2.5°
Propulsion lifetime	20s
Power lifetime	90min
Mass (all units + consumables)	24.5 kg
Processing Power	23 MFLOPS
Communications Data Rate	40kbps

To produce results traceable to proposed formation flight missions the individual self-contained satellites have the ability to maneuver in six degrees of freedom, to communicate with each other (satellite to satellite: STS) and with the laptop control station (satellite to laptop: STL), and to identify their position with respect to each other and to the experiment reference frame via a custom metrology system. The laptop control station is used to

collect and store data as well as to upload control algorithms to the satellites. Figure 4.3 shows a picture of an assembled SPHERES unit. Physical properties of the satellites are listed in Table 4.2.

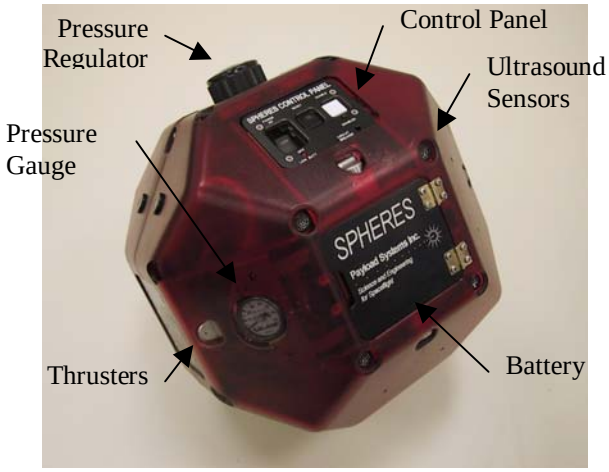


Figure 4.3 SPHERES satellite

TABLE 4.2 SPHERES satellite properties

Diameter	0.25 m
Mass (w/tank & batteries)	4.0 kg
Max linear acceleration	0.17 m/s^2
Max angular acceleration	3.5 rad/s^2
Power consumption	15 W
Battery lifetime	2 h

4.2.1 SPHERES Sub-systems

The SPHERES project was subdivided into six major sub-systems: avionics, software, communications, propulsion, structures, and operations. The avionics sub-system is further divided into processing and support avionics, power, and metrology. The sub-system teams concentrated mostly on the design of the individual satellites, although some sub-systems directly affected the full facility. The avionics sub-system mostly affects the satellites, although the metrology team had to create other hardware. The software team con-

centrated on the operating system that controls the satellites. The communications team had to work both on the hardware within the satellites as well as communication protocols and hardware for the laptop. The Interface/Operations sub-system dealt directly with the whole system, rather than with the satellites. The propulsion and structures sub-systems were limited to the design of the satellites. The sub-system division is summarized in Table 4.3.

TABLE 4.3 SPHERES sub-systems

Sub-System	Scope
Avionics	
Data Processing	Satellites
Power	Satellites
Metrology	System
Communications	System
Software	Satellites
Interface/Operations	System
Propulsion	Satellites
Structures	Satellites

The design of the sub-systems considered the need to satisfy the science goal for formation flight as well as the creation of a broader laboratory for separated spacecraft algorithms. Table 4.3 shows the cases where each of these two goals heavily affected the design of the sub-system. The sections below present short descriptions of the major subsystems and illustrate how each of them fulfills one or both of the goals.

4.2.1.1 Avionics

Figure 4.4 shows an overview of the SPHERES avionics. The figure shows the further sub-divisions of this sub-system. The data processing is the central element. It is surrounded by metrology, communications, propulsion, and power sections which support the other sub-systems of the SPHERES satellite. The metrology system FPGA is an essential element of the avionics system, as it also provides the interfaces for the control panel

and the propulsion system. An external watchdog was implemented to ensure that, if the avionics stop responding, the system is reset. The major elements of the avionics system are described next.

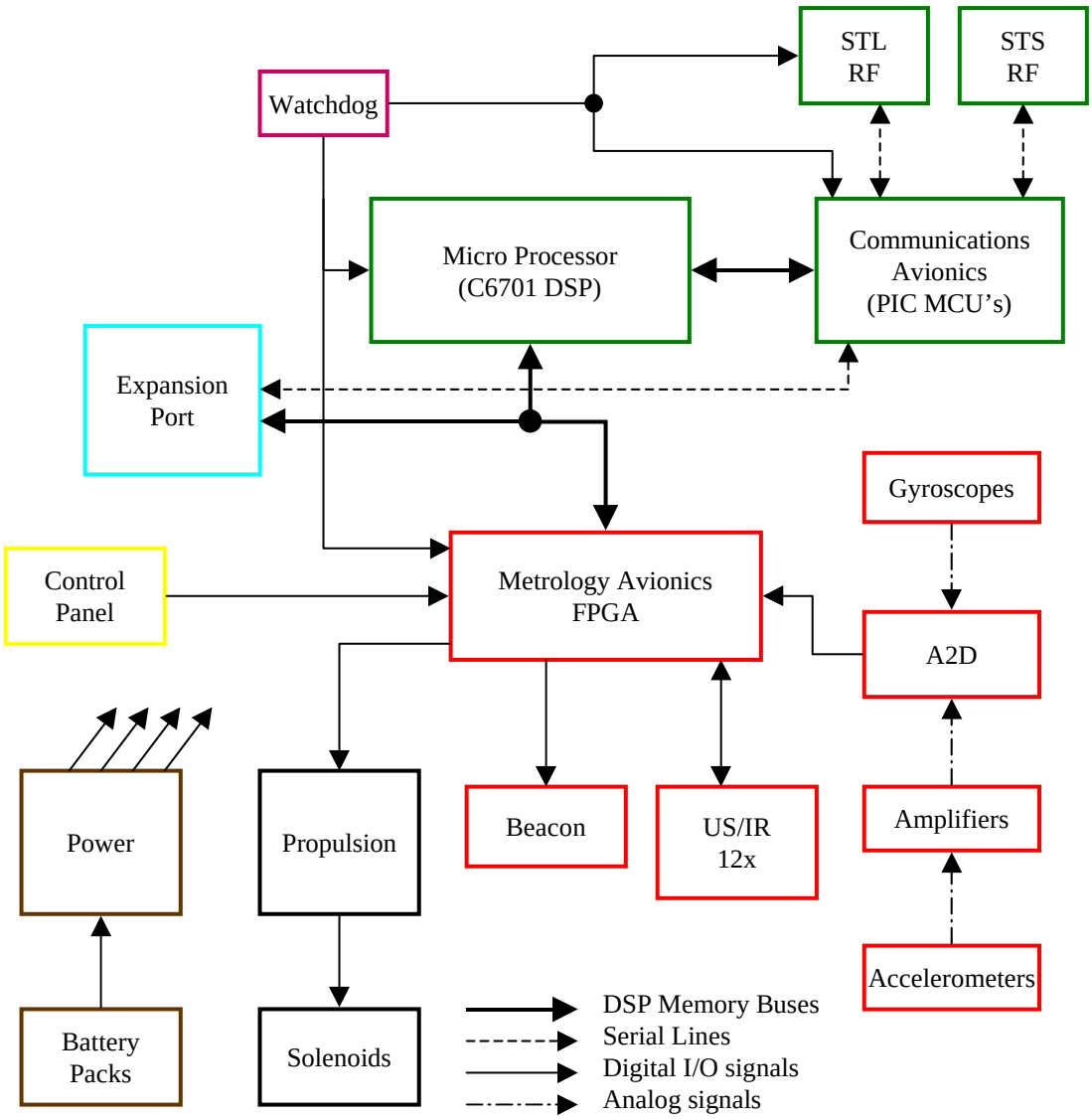


Figure 4.4 SPHERES avionics overview

Data Processing. A Texas Instruments C6701 Digital Signal Processor (DSP) ([TI, SPRS067E], [TI, SPRU189F]) provides the computational power. DSP processors provide multiple features that ensure real-time operation. Further, the DSP processors include

all support functions of a standard processor, allowing it to control the whole unit. The ability of the C6701 to provide between 167MFLOPS up to 1.0GFLOPS, provides significant processing power to prevent being the limiting factor in the performance of the system. The processor is supported with 16MB of RAM and 512KB of FLASH memory. The FLASH memory stores the programs in each satellite. A customized bootloader program allows the FLASH to be reprogrammed during normal operations, allowing unlimited changes to the software.

The original requirement for 23 MFLOPS, based on the estimated needs for formation flight control algorithms, is easily met by the C6701. The selection was ultimately based on the need to provide enough processing power for a wide range of scientists while being compact, low power, and passively cooled. The level of processing power takes into account that some of the algorithms that SPHERES could test have not even been conceived yet.

The metrology sub-system, described below, utilizes inertial sensors (accelerometer and gyroscopes) and a global system (which uses infrared and ultrasound pulses to measure time of flight) to determine the state of the satellite both inertially and with respect to a fixed frame of reference. To perform these calculates the metrology system must support several multiple level interrupts asynchronously with the rest of the software system, which would otherwise consume too much DSP processing time. Further, the system needs analog input lines not available in the DSP. A VIRTEX FPGA [Xilinx, DS001-1] supports the metrology functions. The FPGA handles the asynchronous interrupts of the metrology sub-system and the data capture. The DSP takes the raw information from the FPGA and runs the estimation algorithms.

The rest of the avionics subsystem consists of a propulsion solenoid driver board, a power distribution board, a digital communications board, two RF communications circuits, and the metrology infrared/ultrasonic receiver boards.

Power. The power system for ISS operations consists of two packs of eight AA alkaline batteries per satellite. The packs provide each unit with approximately two hours of operation; once a pack is consumed, it can be easily replaced. The power sub-system provides electrical power to the other subsystems via electronics compatible with the KC-135 and ISS. The power is regulated to provide the necessary voltages for all the subsystems. The total power requirement for a SPHERES free flyer is approximately 13 W. The demonstrated lifetime of the batteries, during operation in both a one-g laboratory environment and the KC-135, is approximately 120 minutes.

For ground-based operations, such as the MIT SSL and the KC-135, rechargeable NiMH battery packs were built. These packs also provide approximately two hours of operations. Due to safety concerns during recharging the rechargeable packs cannot be used in the ISS.

The design of the power sub-system was guided by the requirements for operations aboard the ISS. The goal was to ensure that the primary mission could be accomplished in the ISS. At the same time, the need for replaceable consumables (and rechargeable batteries in ground-based environments) was driven by the MIT SSL Laboratory Design Philosophy.

Metrology. The metrology systems generates real-time estimates of the satellite's state. The metrology measurement system include a global metrology system used to estimate the satellite state with respect to the external reference frame, and an inertial measurement system with accelerometers and rate gyroscopes that is used to measure high-frequency body frame accelerations and angular rates. The global metrology system measures time-of-flight using a combination of infrared and ultrasound signals. The time-of-flight is used to determine the distances between 24 sensors located on the surface of each satellite and five ultrasonic beacons placed at known locations in the work volume. The SPHERES team provides an Extended Kalman filter that uses the inertial and global systems to determine the states of each satellite with respect to the reference frame, although scientists may develop their own estimation routines. Relative state information (e.g. range and

bearing), can be obtained by exchanging the global state information, or by using beacons located on the satellites themselves.

While the metrology system provides essential information for a wide range of distributed satellite systems (DSS) algorithms, the design and implementation were a result of the need to support formation fight.

4.2.1.2 Communications

Each SPHERES unit uses two separate frequency communications channels with an effective data rate of approximately 45kbps per channel. One channel is used for satellite-to-satellite (STS) communications; the other channel enables satellite-to-laptop (STL) communications. Both channels are bidirectional; however, the communication hardware is half-duplex, meaning that only one unit can transmit at a time. The choice of two communications channels closely models the expected operations of future formation flying missions where a high-bandwidth, low-power (short distance) communications link sends data between the units while in space and a separate high-power ground communications link is provided. The two channels of SPHERES are identical in functionality, other than their different frequencies, and therefore the scientists can decide how to best use the two separate channels.

Access to the STS and STL communications channels is controlled by a Time Division Multiple Access (TDMA) scheme. A fixed period of 200ms is shared between the satellites and the laptop (for STL, STS does not include the laptop); the allocated transmission time for each satellite can be configured manually or automatically. The communications module manages transmission and reception of the messages generated by both the SPHERES Core software and the experiment code, such as custom telemetry or command data. If a data transfer is too long for a single packet (32 data bytes), the communications module segments the transmission and sends one packet at a time. The communications module on the receiving SPHERE automatically reassembles the original message from the constituent packets.

4.2.1.3 Software

The software sub-system for the SPHERES satellites is built on a four layer structure, illustrated in Figure 4.5. The software creates an interface for the scientists such that they are never required to program the hardware directly (although it is possible), but rather use higher level functions, simplifying the implementation of their code. The lowest level is comprised of the actual hardware being controlled (e.g., thrusters, RF boards, etc.). The Texas Instruments DSP/BIOS real-time operating system ([TI, SPRU403E], [TI, SPRU423B]), designed for DSPs such as the C6701, is used as the base operating system on the SPHERES satellites. DSP/BIOS provides multi-processing capability, inter-process communication, and a number of input/output management tools. It is the layer which interacts directly with the hardware and manages many of the details for thread and interrupt handling.

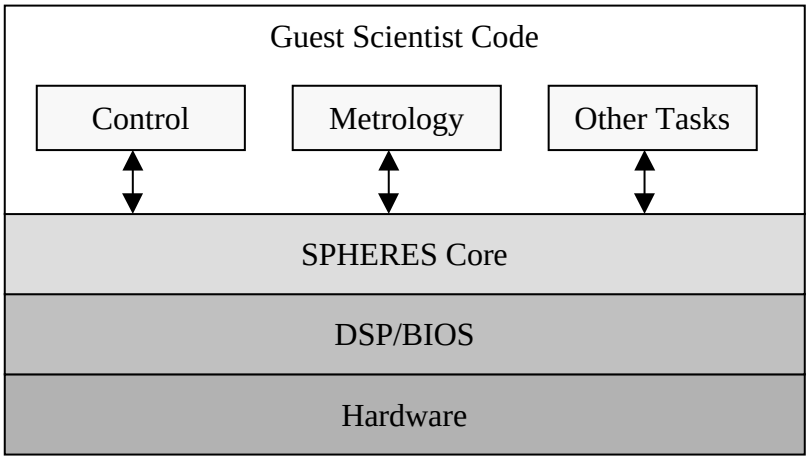


Figure 4.5 SPHERES software layers

The SPHERES Core interface implements multiple distinct execution threads which define the SPHERES Operational environment. This layer implements the basic housekeeping functions required to operate the satellite. These functions run separately from another set of threads designed to execute the test specific algorithms. This separation ensures that activities such as communications and telemetry processing are not affected by any computation-intensive algorithms supplied by the guest scientist. The primary ele-

ments of the SPHERES Core layer are: interfaces to the metrology and propulsion hardware, management of tests and the control interrupt, and management of the communications. The metrology functions capture the data and make it available to the scientist and other SPHERES Core functions, but do not process the data. The propulsion interrupt serves to ensure the thrusters operate as required by the other avionics elements and provides a third level of safety beyond the required NASA safety requirements. The test management functions allow scientists to program multiple individual tests in one program; the functions run the initialization routines and ensure that the tests start synchronously among multiple satellites. The control interrupt management allows the scientist to specify multiple rates without compromising the performance of other threads. The communications core functions implement the communications protocol and ensure compliance with NASA requirements.

The highest level is the actual program implemented by the scientist. To implement their program the software provides six different insertion points for code: two separate periodic, high priority interrupts to collect the metrology IMU and global data; a function to initialize a test, automatically run by the test management part of SPHERES core; a periodic interrupt to run control algorithms; a general purpose background task to be used for functions that require long processing time, but which need not be periodic; and a background task directly linked with the high-priority metrology interrupts to run metrology estimators which take long processing time. The period of both the metrology and control interrupts can be changed by the scientist in the initialization function of each test.

The SPHERES Core software environment was designed specifically to support multiple investigators. Satisfying the need to demonstrate formation flight control algorithms could be accomplished with a design that contemplates less threads and interfaces. Instead, the software sub-system was designed to ensure that the different scientists have access to high-level functions so that their algorithms are easy to implement.

4.2.1.4 Interface/Operations

SPHERES operations were planned specifically to create an environment which meets the MIT SSL Laboratory Design Philosophy. Figure 4.6 presents an overview of the operations plan for SPHERES. The plan consists of providing scientists with a simulation to use in-house, for quick turnaround of tests. Once the simulation demonstrates the algorithms are ready for hardware test, these are sent to the SPHERES team for testing in the MIT SSL 2D environment. The tests are sent to the ISS only after the SPHERES team has demonstrated that the tests can be run in the hardware.

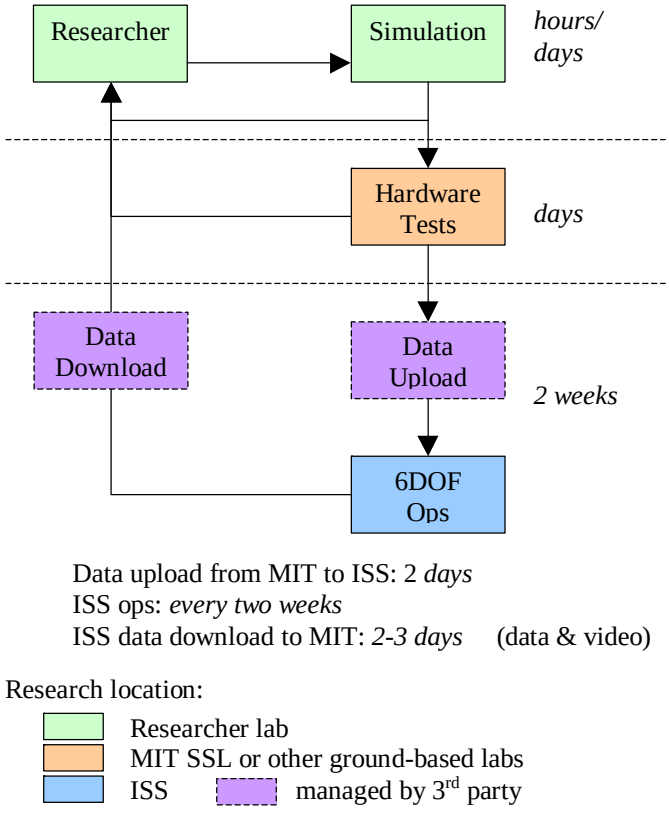


Figure 4.6 SPHERES operations overview

To implement this operational plan three interfaces were created:

- **GSP Simulation.** A simple interface to start/stop tests which provides scientists with data files to determine if an implementation is successful. The

algorithms developed in the simulation are easily portable to the SPHERES hardware

- **SSL Laboratory -** The interface for use in the MIT SSL provides detailed information on the tests being conducted as well as real-time data. The interface was designed to maximize the information to the researchers to help in the debugging and development processes. It provides the same data files that would be available from ISS tests.
- **ISS Interface -** The interface for the astronauts to conduct tests aboard the ISS was designed to meet NASA's usability requirements and to present high-level information about the tests in such a way that astronauts can make decisions on the success or failure of a test in real-time. Yet the amount of data is simplified from that of the SSL laboratory interface so that astronauts can concentrate on running the tests rather than analyzing them.

4.2.1.5 Propulsion

The satellites are propelled by a cold-gas thruster system which uses carbon dioxide as propellant. The CO2 propellant is stored at room temperature in liquid form at 860 psig, without the need for a cryogenic system. A regulator reduces the pressure to between 20-70 psig; the operating pressure may be adjusted manually prior to each test. A Teflon tubing system distributes the gas to twelve thruster assemblies, grouped in six opposing pairs. The thrusters are positioned so as to provide controllability in six degrees of freedom, enabling both attitude and station keeping control. Each thruster assembly consists of a solenoid-actuated micro-valve with machined nozzles optimized for the desired thrust of 0.125 N. The propulsion system may be easily replenished by replacing a spent propellant tank with a fresh, unused tank. The propulsion system is directly traceable to the propulsion systems of most existing spacecraft. The dynamics created by the SPHERES propulsion system directly simulate those of other thruster systems: non-linear dynamics, on/off operation, pulse width modulation or frequency modulation, and full controllability in 6-DOF. The system's bit pulse of 5ms (with an equivalent impulse bit of 0.625×10^{-3}Ns) ensures the precision necessary to operate the system at frequencies of up to 50Hz.

4.2.1.6 Structures

The primary structure consists of all of the internal and external components necessary to provide rigidity and support for the SPHERES satellite units. The primary structure functions to provide a physical base to which everything else attaches. It consists of the internal and external subassemblies. The internal subassembly consists of an aluminum frame which provides for the physical mounting of internal devices. Six internal rings comprise the main elements of the internal structure. The rings are grouped in pairs; each pair is aligned with each axis of the SPHERE. The rings fit together without a rigid connection between them. Instead, each pair is held together by four brackets; once each pair is held together, the assembly holds without connecting the rings. The external structure consists of two molded Lexan shells. The shells attach to the brackets which hold the rings together. Figure 4.7 presents CAD drawings of the internal and external subassemblies of the SPHERES nano-satellites.

The structure was designed to ensure the safety of the satellites, therefore it provides quick access only to the tank and batteries. Replacing tanks and batteries does not require any special tools nor to remove any structural elements. To safeguard all other subsystems, the structure does not allow direct access to any other internal elements of the satellites. The expansion port (described in Section 4.3.3.2) can be accessed by removing a panel attached with four screws, but it not designed for immediate access without tools.

4.2.2 Further Information on SPHERES

The previous section presents a summary of the design of the six primary SPHERES subsystems. The SPHERES hardware, software, and operational plans have undergone substantial review processes over more than four years of design and operations. The design history of SPHERES, as well as the current design, have been documented in several documents and multiple presentations. [SPHERES, 1999] and [SPHERES, 1999a] describe the design of the prototype units. [Saenz-Otero, 2000], [Chen, 2001], and [Saenz-Otero, 2002] present results obtained with the prototype units.

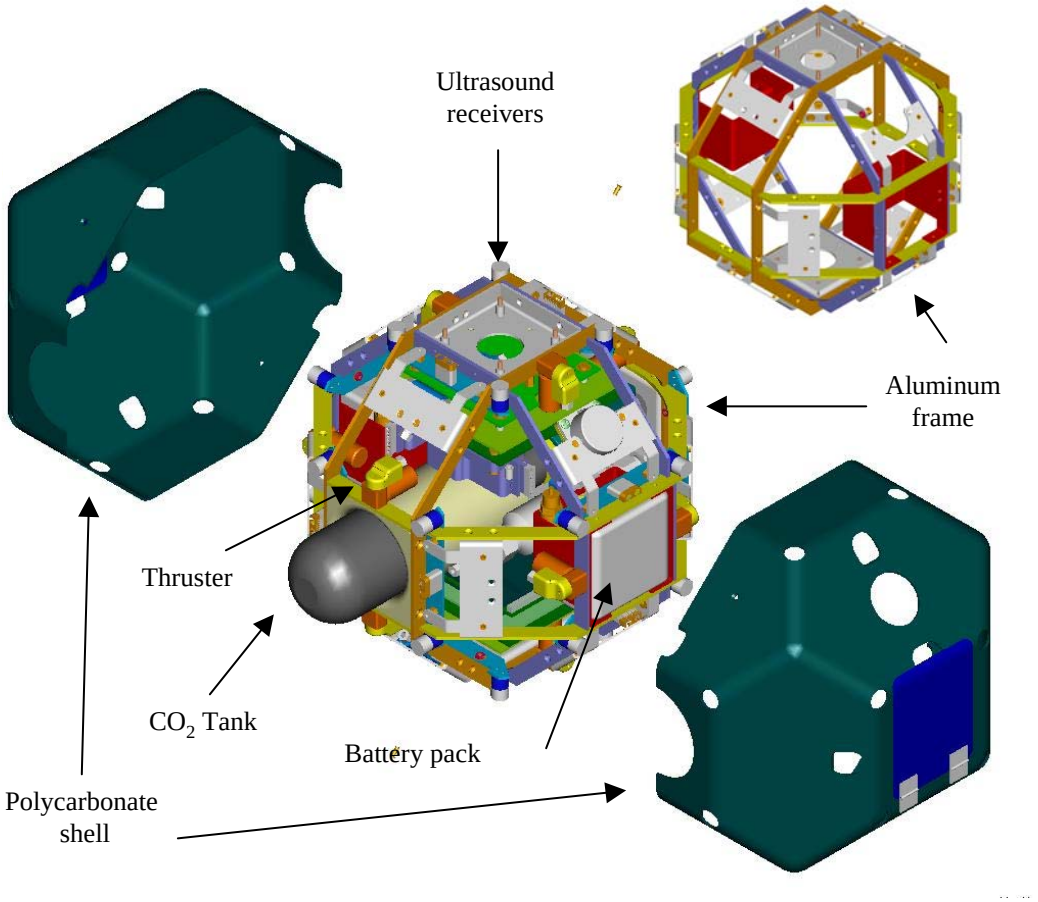


Figure 4.7 SPHERES nano-satellite structural design

[SPHERES, 1999] and [SPHERES, 1999a] are the critical design reviews for SPHERES. These presentations provide further detail on the design and operations of the flight units. [SPHERES, 2001] details the NASA safety requirements and the specific design elements which satisfy them. [Hilstad, 2003a] presents the interfaces of the SPHERES flight software available to scientists. [Nolet, 2004] and [Kong, 2004] present results from the flight qualified units in ground operations.

Due to the direct impact of the avionics, software, and communications sub-systems on the ability of SPHERES to satisfy the MIT SSL Laboratory Design Philosophy (as explained below), further detail on these sub-systems is presented in several appendices of this thesis. Appendix F presents detailed information on the avionics design of SPHERES.

The appendix present the functional block diagrams and schematics for all the electronics in the SPHERES nano-satellites, the external communications antennae, and the global metrology beacons. Appendix G presents in detail the design of the SPHERES bootloader and the flight software. Appendix H is the SPHERES communications interface document, which details the implementation of the SPHERES packets and the TDMA protocol.

The design of SPHERES contemplates the need to satisfy the goal to develop a testbed for formation flight while at the same time creating a laboratory for DSS. The following section describes how the sub-systems introduced in this section implement a wide range of features which help meet all the aspects of the MIT SSL Laboratory Design Philosophy.

4.3 Meeting the MIT SSL Laboratory Design Philosophy

The design of the SPHERES project considered each one of the design features for a laboratory, while ensuring the formation flight goal was accomplished. Table 4.4 shows the cases where a sub-system was designed specifically to meet the philosophy, to meet formation flight requirements, or both. The avionics, software, communications, and operations sub-systems most directly relate to designing a laboratory. The metrology, propulsion, and structures sub-systems were designed to meet the formation flight mission-specific goal.

The SPHERES system as a whole helps to fulfill some of the features which could not be done by an individual sub-system. Still, the avionics, software, communications, and interface/operations sub-systems implement capabilities which directly fulfill features of the philosophy. Table 4.5 cross-references the philosophy's features with those sub-systems which most influence the ability of SPHERES to satisfy the MIT SSL Laboratory Design Philosophy. The avionics system design helps to meet the lower-level features in the support of experiments and modularity & reconfiguration groups. The communications sub-system mostly supports running experiments, which in turn facilitates the iterative research process. The software and interface/operations sub-systems work at a higher

TABLE 4.4 Design for formation flight (FF) vs. design philosophy (Lab)

Sub-System	FF	Lab
Avionics		
Data Processing	✓	✓
Power		✓
Metrology	✓	
Communications	✓	✓
Software		✓
Interface/Operations		✓
Propulsion	✓	
Structures	✓	

level, using the avionics and communications systems to support the iterative research process and multiple investigators.

TABLE 4.5 SPHERES sub-systems and the design philosophy

Sub-System	Facilitating the Iterative Process	Experiment Support	Multiple Investigators	Reconfiguration & Modularity
Avionics		✓		✓
Communications	✓	✓		
Software	✓		✓	✓
Interface/Operations	✓		✓	

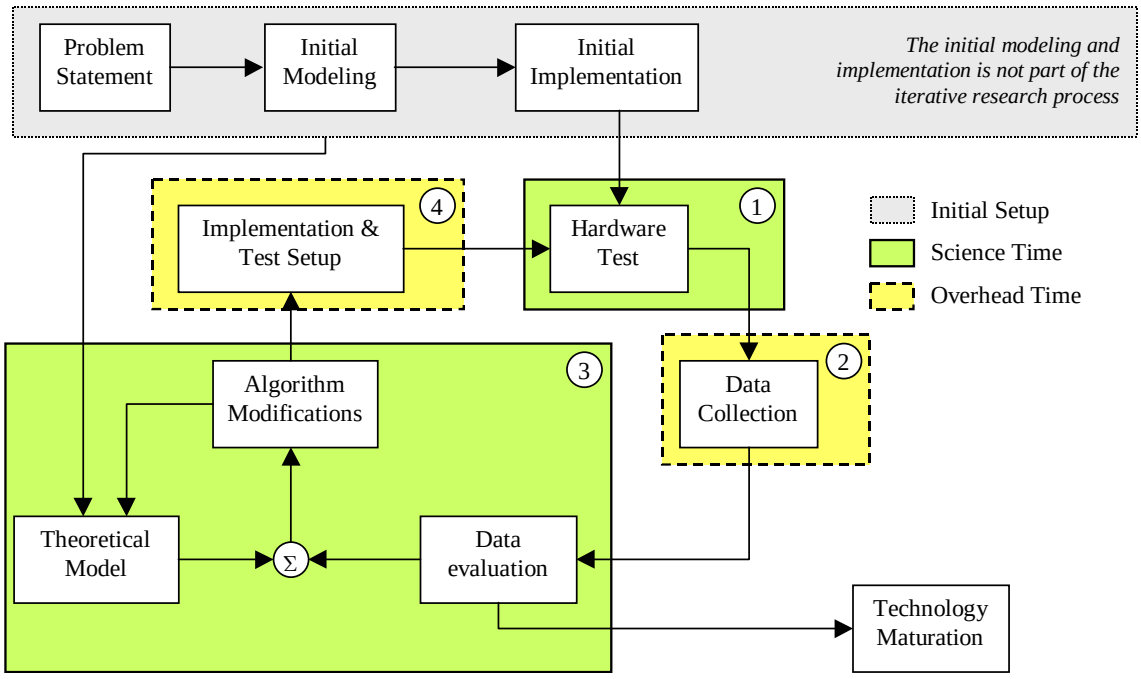
This section progressively details how SPHERES implements different capabilities which help fulfill the features called upon in the MIT SSL Laboratory Design Philosophy. Each of the next four sections concentrates on each of the four main groups of the philosophy: facilitating the iterative research process, support of experiments, support multiple investigators, and reconfiguration and modularity. Within each section, details are presented on how the implementation of a specific capability fulfills one or more of the features of the

philosophy; the sections also describe how enabling a feature sometimes required the proper integration of multiple sub-systems. At the end of the description, a summary provides a direct relationship between the characteristic and one of the MIT SSL Laboratory Design Philosophy features.

The *iterative research process* section describes the high level implementation of the SPHERES operations plan and software system which facilitate conducting science. Next the section on *support of experiments* describes several low-level hardware capabilities that were used to ensure the high-level features of the philosophy were successful. Third, the SPHERES Guest Scientist Program (GSP) and other high level features are presented to demonstrate how SPHERES allows research by *multiple investigators*. Lastly, the section on *reconfiguration and modularity* explains the low-level capabilities of the SPHERES hardware to change their configuration and create a modular system.

4.3.1 Facilitating the Iterative Research Process

SPHERES was conceived to allow for the development and maturation of control and metrology algorithms for use in formation flight spacecraft. Therefore, the iterative research process for tests performed in the SPHERES facility consists of the steps necessary to create models, develop algorithms, execute the experiments, and analyze the data to evaluate the algorithms and update them. This process must be repeatable so that the researcher can iterate during the development of the theory with confidence that environmental conditions are not changing. The specific steps identified are: initial model and algorithm development and implementation; execution in the SPHERES hardware; data collection and delivery to the researcher; analysis of the data to determine the need for further development or the achievement of maturity; and, if necessary, modification to the algorithm at different levels (either the major concepts or detailed structures such as control gain). Figure 4.8 illustrates each step of the iterative design process, as adapted from the scientific method presented by [Gauch, 2003] shown in Figure 3.1 on page 74.



Four major steps which support the iterative process:

1. Test execution (science time: allow enough time)
2. Data collection and delivery to researcher (overhead time: minimize)
3. Data evaluation and algorithm modification by researcher (science time: allow enough time)
4. Modification to tests and new program upload (overhead time: minimize)

Figure 4.8 Iterative research process for SPHERES

Figure 4.8 also identifies three different ways in which time is spent during the iterative research process:

1. **Initial development:** developing the problem statement, initial modeling, and initial implementation to prepare for the first experiments.
2. **Science time:** investigating the scientific aspects of the problem, which include the actual test time to run a significant experiment, data analysis, and development of new theoretical models and hypotheses.
3. **Overhead time:** the time necessary to collect the data and make it available to the scientist, and the time needed to implement changes in the hypothesis and start a new test with the updated algorithms.

The design of SPHERES concentrates on the four main steps that support the iterative research process, identified in Figure 4.8 by numbers within circles: first, on providing scientists with the correct amount of time to run tests (science time); second, minimizing

data collection and delivery time (overhead time); third, providing enough data evaluation and model refinement time (science time); and fourth, enabling easy modifications of the algorithms (overhead time). The initial theoretical analysis and implementation is considered a constant outside of the scope of the iterative research process; these issues are addressed by other features of the design philosophy, such as the ability to support multiple investigators.

The goal of the SPHERES facility is to provide sufficient science time, while minimizing the overhead time. A successful facility allows researchers to run tests for long-enough periods of time to return valuable data. If the time to perform experiments is too short, then the amount of useful data will be reduced; short experiment time may even prevent a test from completing, in which case the overhead of restarting a test becomes substantial. The time for step two should be minimized, meaning that the time to collect the data and make it available to the scientist in a useful format should be as short as possible. The third stage, where the researcher analyzes the data must be more flexible. This time should not be so short that the researcher is unable to perform careful data analysis; nor should it be so long that the scientist is unable to effectively track the evolution of the process or meet mission deadlines. It would be preferable for this time to not be fixed, but rather allow the scientists some leeway to ask for more time if necessary, or potentially to speed up the process if new algorithms are created quickly. The time of the fourth stage, the implementation of changes and setup of new tests, should be minimized. The time must be such that the researcher will not loose interest on the next test, and will remember all the changes performed in the last iteration along with their rationale. In the case of SPHERES this process involves the creation of a new program and its delivery to the appropriate location for upload to the satellites.

SPHERES must allow researchers access to each step of the iterative research process with efficiency, allowing the algorithms to be developed not only correctly but within a reasonable amount of time. For this purpose, the team considered not only the design of

the testbed itself, but also the resources available within the ISS that should interface with the facility to achieve this goal.

To facilitate the iterative research process, i.e. to minimize overhead time and maximize science time, SPHERES implements the following capabilities:

- Multi-layered operations plan
- Continuous visual feedback
- Families of tests
- Easy repetition of tests
- Direct link to ISS data transfer system
- De-coupling of software from NASA safety controls

The implementation of these capabilities are detailed next.

4.3.1.1 Multi-layered operations plan

The SPHERES operations elements that benefit the iterative process consist of three main elements: the Guest Scientist Program (GSP) simulation, ground based facilities, and ISS operations. Figure 4.6 on page 111 shows an overview of the SPHERES operational modes that enable iterations. As seen, the longest cycle, when the experiments are conducted aboard the ISS, completes in a matter of a few weeks. The benefits of each of these elements is described below.

Iterations with the GSP Simulation

The GSP Simulation allows remote researchers to develop their algorithms in house at their own pace. Figure 4.9 illustrates the iterative research process of the GSP simulation environment. The only overhead related to the simulation is the need to convert all of the researcher's algorithms into C code and make it fit within the SPHERES software Application Programming Interface (API). While the initial time spent on converting the code to C may not be negligible, it is a necessary step to operate on the SPHERES hardware. Therefore, the initial time spent to convert the code will be useful in the long-term, as that

code will serve as a base for tests that may ultimately be performed aboard the ISS. The simulation allows multiple iterations of a technology to be accomplished in a few hours, after the initial overhead to translate the algorithms (expected to be a period of a few days).

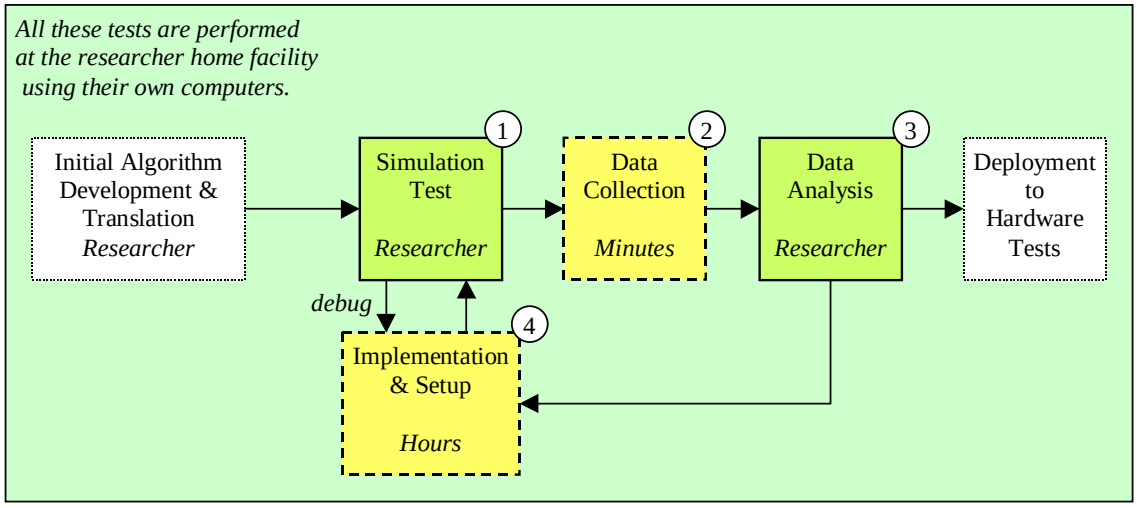


Figure 4.9 GSP iterative research loop

Once the code has been adapted to the SPHERES API, the researcher may run tests using the simulation, which provides the same data that a hardware test would provide. The data is augmented with the state of each satellite as calculated by the simulation independent of the metrology algorithms in use by the researcher. The flight-style data ensures that iterations in the hardware will contain the necessary telemetry; the simulation calculated state serves as a truth measure to determine the success of the researcher's algorithm within the simulation.

Iterations at the MIT SSL

The MIT SSL provides researchers with a low-stress environment where 2D (3DOF) tests can be performed. In this environment the researcher can easily modify tests and programs in multiple development stations, and the overhead to reload a program (approximately 2-10 minutes), does not present a considerable delay. While at the MIT SSL, the researcher

will spend most of their time evaluating the data, running tests, and modifying tests to improve their performance. Further, the researcher need not be present physically at the MIT SSL. The MIT SPHERES team will match on-site students with partner researchers in such a way that the researcher remains in their main location while the member of the SPHERES team will conduct the experiments and relay data to the researcher as needed. Because the team members are fully proficient on the operations of SPHERES, this process speeds up the iterations by allowing the researcher to concentrate on their science, instead of the SPHERES operations.

In the ground-based SSL facility, the first step of the iterative process, running the actual test, is limited by the capabilities of the equipment that allows 3DOF operation and the consumables of the testbed, regardless of the operating scenario. The consumables of the air-carriages that support the satellites last up to 20 minutes. The gas on the satellites, in ground operations, lasts approximately 20-30 minutes. Therefore any continuous test is limited to 20 minutes. While all the ground operations so far have required test times of less than 10 minutes, this constraint remains a valid hard constraint on the iterative research process steps.

Since researchers may or may not be present at the MIT SSL to run their experiments, there are two possible iteration time lines for this environment. When the researcher is present at the MIT SSL, the overhead time is minimized greatly, requiring only minutes to obtain the data and to setup experiments. On the other hand, unless the researcher is based out of the MIT area, the time in between iterations will be restricted by the costs of remaining on site. Conducting tests on-site has proven most useful when the researchers are on the last steps of their design and wish to only optimize the last details of their algorithm with a quick turnaround between tests. Figure 4.10 illustrates the on-site iterative research process.

The other operating scenario is when the researcher operates remotely and is supported by a member of the SPHERES team. Figure 4.11 illustrates the off-site iterative research pro-

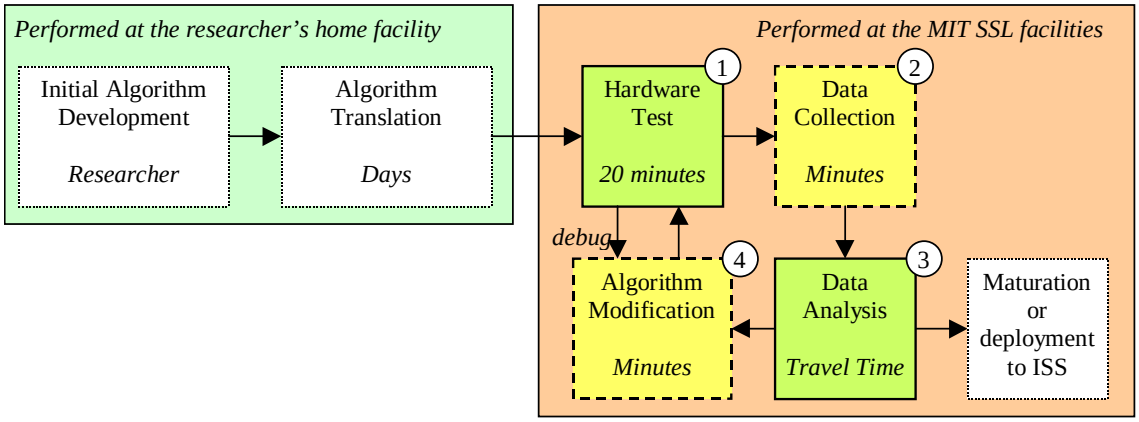


Figure 4.10 MIT SSL on-site iterative research loop

cess. In this case the overhead time to collect data and setup experiments increases to days. The data transfer can usually take place within hours, but the uploading of new programs requires the SPHERES team member to compile the program for use in the hardware; this process can take up to a few days. On the other hand, the researcher has practically unlimited time (up to months, if so desired), to analyze the data and produce new or modified algorithms. The GSP Simulation enters the loop once more, as scientists will test their algorithm modifications with the simulation prior to sending new programs to the SPHERES team.

Iterations aboard the ISS

Operations aboard the ISS tie in all the steps of the SPHERES operations procedures: GSP simulation, ground facilities, and ISS operations. Any tests to be performed in the ISS must prove capable to operate both on the GSP simulation and the SSL before the SPHERES team will allow delivery to the ISS. While tests at the SSL are not expected to perform all maneuvers expected from the 6DOF environment of the ISS, all tests will be checked for errors that could affect the operations of SPHERES; in those cases where the success of the testbed cannot be shown in the 3DOF SSL environment, they will be expected to perform correctly in the GSP simulation and, at a minimum, successfully load

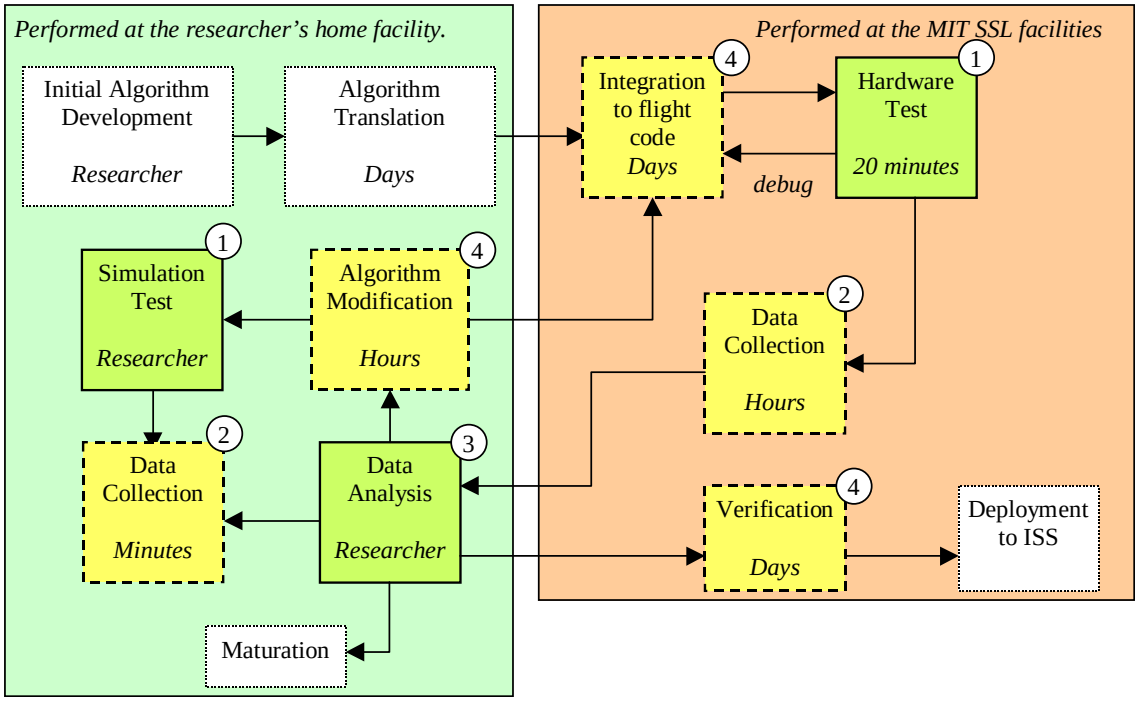


Figure 4.11 MIT SSL off-site iterative research loop

and run on the C6701. Therefore, the iterations aboard the ISS will include the overhead of both the GSP process and the SSL operations.

The ISS also adds other overhead times of importance to the iterative research process. To operate aboard the ISS the SPHERES team must interface with NASA via our payload sub-contractor, Payload Systems Inc. (PSI), and the payload sponsor, the Department of Defense (DoD) Space Technology Program (STP). While the software, as described above, presents no safety-critical items that will require NASA verification, it must be uploaded to the ISS via both PSI and then STP, followed by the NASA ISS office. This process will require the SPHERES team to have software ready for upload several days in advance of the up-link. Once NASA has received the programs for upload, the up-link to the main ISS server will occur within one day. The astronaut can then copy the program from the server to any one of the general purpose laptop computers aboard the ISS prior to the operating session.

Data download will also have added overhead, since both PSI and STP must be involved. Further, the data down link from the ISS is not necessarily real time. For the majority of the SPHERES operations NASA has indicated a one day download time of raw data (including astronaut questionnaire and feedback), and a two to three day video download. The data must then pass through STP and PSI before reaching the SPHERES team and/or researcher (in some cases PSI may send data directly to the researcher).

As explained in Chapter 2, astronaut time is a precious resource aboard the ISS. SPHERES has been allocated a fixed time for operations over a period of six months. The SPHERES team decided to allocate the time in intervals of two hours of operations every two weeks. This time will be fixed by NASA once operations start (although SPHERES operations may be preempted by other NASA activities). Therefore, the minimum time for data analysis and algorithm modification for researchers will come in intervals of two weeks. A scientist may decide to have the time between iterations be every two, four, even six weeks or more depending upon their needs, but not less than two weeks.

While the total time per session is two hours, one must recall that the limiting factor for each test are the consumables. Specifically, the available propellant in each tank is estimated to last for up to 30 minutes (depending heavily on controller usage). Therefore, each test can be at most 30 minutes long, assuming conservative gas consumption and that a new tank was used at the start of the test. Longer tests may be possible for minimal gas consumption. At that point, the limiting factor becomes the batteries, which last up to two hours, regardless of the maneuvers of the satellites.

At this point it is important to note that, while in ground operations the time to upload a program to the SPHERES satellites was considered negligible, this can no longer be ignored in the ISS. Uploading a program takes up to five minutes per satellite that must be programmed. Therefore, for a three satellite test, the overhead will be up to 15 minutes; this is a considerable amount of time out of the two hours allocated per session. Therefore,

the SPHERES operational plans call for no more than two programs (each with multiple tests) to be uploaded per session.

Figure 4.12 on page 127 presents the ISS iterative research process graphically. The process starts at the researcher facilities utilizing the GSP simulation. The total overhead at this point is in terms of hours to run enough simulations and debug software. The process continues with the delivery of the software to the MIT SSL for integration into a flight package. This process will add several days of overhead to ensure the software is ready for delivery. Once validated, the program is sent to the ISS via PSI/STP/JSC, for a total overhead of approximately two days. The astronaut, once scheduled to operate SPHERES, copies the program to a local laptop and loads the program, for a total overhead of approximately 15 minutes. A cycle of tests follows; multiple tests are run, guided by the astronaut's decisions and the operations plans provided by the scientist. The data collected on the ISS laptop is copied to the main ISS server immediately after the session in a matter of minutes; the data is made available to STP/PSI by JSC within one day. Within another day the data reaches the researcher, for a total data download overhead of approximately two days. The video is downloaded from the ISS in approximately two or three days, and made available in digital format to STP/PSI in another few days. In total, the video of a test session is expected to be available within one week of the test session. The researchers are expected to operate on a four or more week cycle to allow sufficient time for data analysis and algorithm modifications.

> *A multi-layered operations plan allows scientists to perform research iterations in-house, remotely and locally at the MIT SSL, and remotely at the ISS. The iteration period ranges from hours to a flexible 2-week schedule for operations aboard the ISS.*

4.3.1.2 Continuous visual feedback

A major obstacle to maturate technologies via simulation is the inability to fully understand the dynamics of a system and visualize them properly. SPHERES provides research-

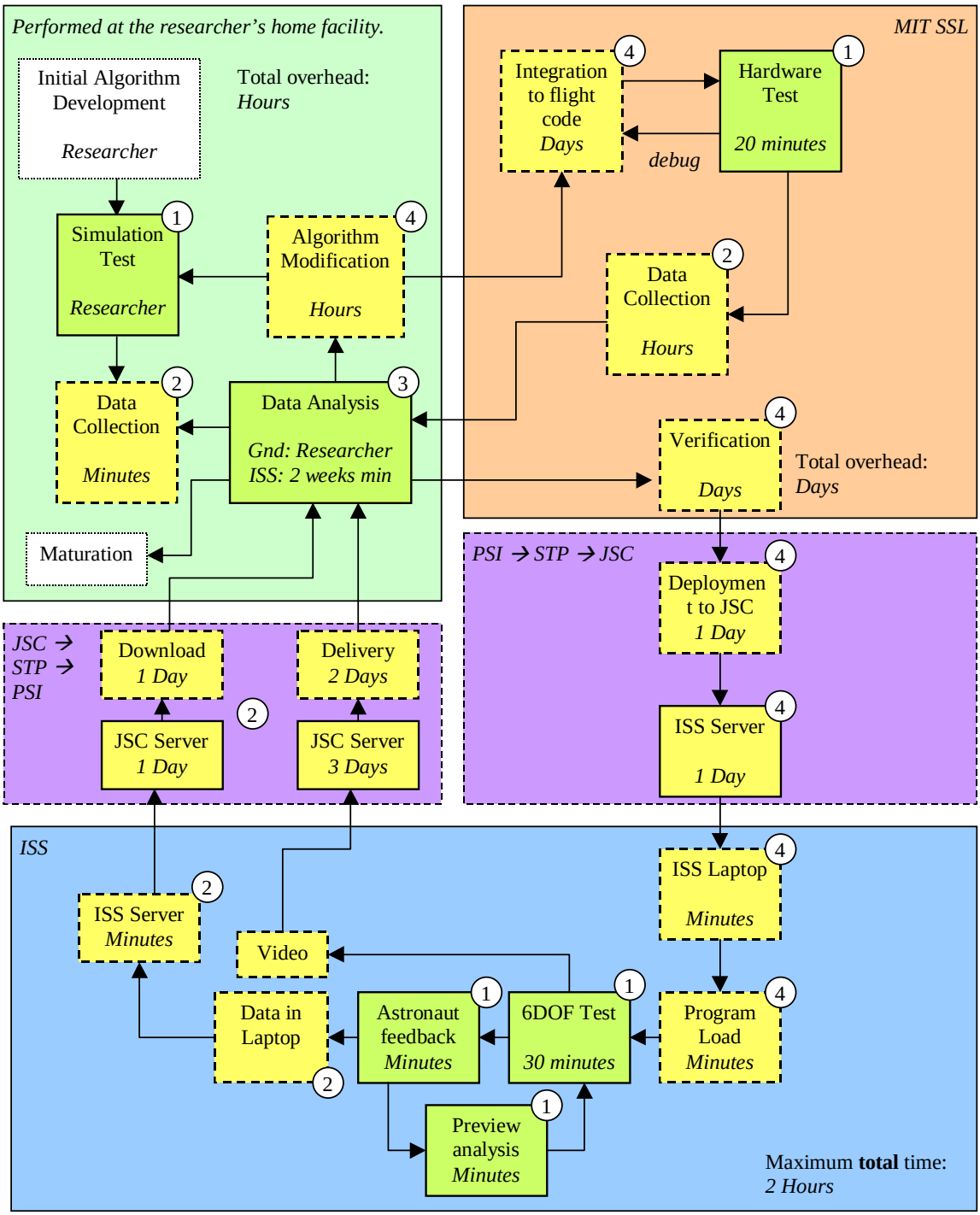


Figure 4.12 ISS iterative research process

ers with a physical system where they can carefully study the behavior of the satellites. When the researcher is present, they can directly identify the dynamic behavior of the satellites, which allows them to quickly determine the success of a test. Further, they obtain much more insight into the actual three dimensional behavior of the units, even if conducting tests in the 2D ground facilities. When SPHERES is operated remotely of the researcher, the facility always provides researchers with two visual feedback elements: a human is always present to evaluate tests in real-time, and video will always be available. In many cases video is available from multiple angles (including ISS video), which can potentially be used for data analysis.

The ability of SPHERES to minimize the data collection overhead time is directly related to the availability of visual feedback and video in all locations. The physical operations provide the researchers with immediate feedback to make rough determinations of success or failure of the experiments. Since the SPHERES facility was designed for the development of dynamics algorithms, in many cases it will be clear from the video when an experiment succeeds or not, since specific motions will be expected. Using video the scientist can then determine which data sets to investigate further, saving time by not having to analyze every single data set.

> *Continuous visual feedback by humans allows scientists to reduce the data collection overhead time by filtering useful experiment runs.*

4.3.1.3 Families of tests

The SPHERES software enables researchers to run a full family of tests with ease. The SPHERES software consists of programs that contain multiple tests. At any point in time only one program can be loaded into a SPHERES satellites. The program can be changed easily with the bootloader, described in Section 4.3.4.7 below. But changing a program does have a two to five minute overhead, which, as described above, is not negligible in the ISS environment. Therefore, to minimize the overhead in starting experiments, SPHERES allows each program to perform multiple tests.

Theoretically the different tests in each program can be completely independent of each other; in practice the SPHERES team attempts to maintain similarity between the different tests in each program to minimize the size of the program and to ensure that subsequent tests make sense operationally. In the case of a controls problem, for example, a program may contain multiple tests of the same algorithm with different gains for the controller; but the tests could also sequentially build on the controls problem, adding steps with each test. The ability to run these families of tests sequentially, without any substantial overhead, allows for different parts of an algorithm to be tested individually and then collectively, until the full algorithm is demonstrated.

Individual tests can be further divided into maneuvers. Each test can contain one or more maneuvers, allowing the researcher to further divide the test and identify up to what maneuver a test performed as expected. As opposed to tests, which can be started in any order, the user does not have control to start a test at a specific maneuver; tests must always start with the first maneuver. But the software architecture allows maneuvers to be shared among tests, and a researcher can create tests with overlapping maneuvers to test different parts of an algorithm.

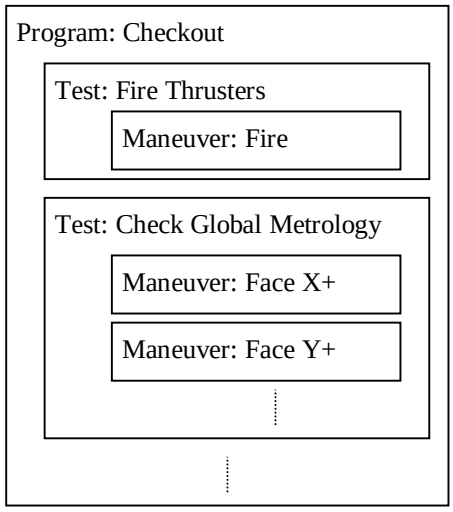
Figure 4.13 illustrates the structures of programs as implemented in the SPHERES software environment, and describes an example for the "checkout" program. This program tests the different sub-systems of the SPHERES satellites independently of each other, therefore demonstrating the ability of one program to perform substantially different tests. Operationally, though, the checkout program is congruent, as the operator will clearly identify each sub-system with a test. The example also shows how maneuvers can be used to simplify test design. In the checkout program the "global metrology" test, which checks the functionality of the ultrasound receivers in the satellites, uses maneuvers to keep track of which satellite face is being tested, rather than having to create special telemetry.

The implementation via families of tests and ability to change programs further facilitates the iterative research process by effectively allowing the software to be modified at any

One *program* is loaded on the satellites at a time

Programs consists of multiple *tests*, selected in any order to accommodate real-time results

A tests consists of one or more *maneuvers* to identify test progress.



Example:
- The "Checkout" program tests all sub-systems, including propulsion and metrology; to test all sub-systems only this one program needs to be loaded to a satellite
- The propulsion test has only one maneuver, to fire all thrusters; the astronaut can run the test multiple times at any point
- The metrology test has one maneuver for each face of the satellite; the standard telemetry will indicate which face was being tested by saving the maneuver number, without the need for special telemetry

Figure 4.13 SPHERES programs composition

level. When considering steps three and four, algorithm modification and test implementation and setup, this design gives the scientist several advantages. During the algorithm modification time the scientist can decide to review only specific maneuvers, combine tests, or even redefine the program completely. There is no overhead in terms of reloading the program to choose any level of modification (albeit, small changes result in the need to load the full program again). By allowing the scientists to modify their algorithms at any level, SPHERES maximizes the time scientists can spend in re-defining their algorithms, rather than implementing them. The software sub-system ensures that the iterative process is not slowed down whether small or large changes are needed.

> *Running families of tests minimizes the overhead to start a test and allows multiple parts or types of algorithms to be tested in one session.*
> *The layered program structure enables both small and large changes to the algorithms.*

4.3.1.4 Easy repetition of tests

Chapter 3 reviewed two essential concepts in conducting research: the scientific method and the design of experiments. The operations plan presented above works to fulfill the

need of the scientific method to iteratively improve a hypothesis. The design of experiments, on the other hand, closely relates to the ability to conduct the correct number of tests, with a number of variables, to demonstrate the statistical viability of a hypothesis. This concept lies within step one and four of the iterative process presented in Figure 4.8 on page 118: minimize the time to setup a test to maximize the science time. To accomplish this goal, the design of SPHERES ensures that it is possible to repeat tests with ease.

This repetition of tests takes place within a specific program; the times to minimize are the ones involved with setting up the units for a test and commanding the start of the test. Starting a test with SPHERES consists of four simple steps:

1. Check battery and gas pressure: visual feedback of both available battery (low battery indicator) and gas tank contents (estimated use) allow this check to take place in seconds.
2. Enable the satellite (mandatory only in the ISS): for safety reasons the satellite must be enabled prior to any actuation. This process requires the astronaut to depress the enable button for more than one second.
3. Position the satellite correctly: this is potentially the most time-consuming part, since it is desired that all tests of the same algorithm start with similar conditions. Still, the small physical size of the satellites (both mass and volume) allows easy manipulation so that similar starting conditions can be achieved. Tests in the ISS by astronauts, with objects of similar size to the SPHERES satellites, have demonstrated that preventing drift of one unit while other units are re-positioned does not pose a challenge.

   In the case of ground tests, positioning the satellites also requires positioning their air carriages correctly and turning on the gas for the carriages. This process takes only a few seconds for one or two SPHERES, although it can pose an operational challenge for three or more units. Ground tests have shown that the air-carriages do drift once floating, and therefore require more human attention to ensure a successful start.
4. Command test start: the ground-based interfaces command a test start with the simple press of one key. The ISS interface requires two steps to satisfy safety requirements. (Both interfaces are described in further detail below).

The total time to start a test is less than one minutes in ground stations when using up to two satellites, and approximately one to two minutes for the ISS. When using three or more satellites in ground facilities it may take up to five minutes to start a test, depending

on the conditions of the surface where the carriages are floating and the amount of drift experienced.

Another important aspect of repeating multiple tests easily is the ability to stop a test that has gone wrong and reset the facility for the next test. SPHERES allows multiple ways to stop a test and setup the facility ready for a new test. The software can restart a test in multiple scenarios. The researcher can program special conditions which cause a test to automatically end, leaving the satellite ready for the first step of setup. The software implementation ensures that a test stop will not affect the behavior of the rest of the system. This feature allows tests to be run only as long as necessary, allowing tests to be stopped if they fail in an obvious manner, while ensuring the data is safely archived for post-examination. A remote restart command, which causes the units to re-start in a manner equivalent to cycling the power, was implemented in case the satellites are out of reach. The satellites' control panel allow the scientist to disable the current test, which does not perform an actual reset of the avionics, or to reset the unit completely, also equivalent to cycling power. Through these four different reset scenarios SPHERES allows a scientist to both stop tests without affecting the state of the avionics or to fully cycle the satellites for a clean start.

The ability to physically restart a test is not useful unless it is possible to identify which set of data corresponds to each test. SPHERES transmits multiple telemetry packets which clearly identify each run of a test, regardless of whether a previous test initiated, ran, or concluded as expected. The system is further enhanced by the different operator interfaces, all of which save all raw data, allowing for error-correction in post-test analysis.

SPHERES implement these features by creating a high-level state machine as part of the basic SPHERES software and utilizing the external watchdog of the avionics system to force resets. Figure 4.14 illustrates the state machine implemented by the software. Four states are implemented: boot time, which initializes the satellites and load the current program; idle, which does not perform any actuation and which is not running any user

threads (see figure Figure 4.5 on page 109), but which sends the "state of health" (SOH) information once a second, which saves the current state, the satellite on-time, and tank usage, among others; the "ready" state is an intermediate step which allows a test start command to be acknowledged, but which otherwise is not functionally different from idle; the running state, which starts after a command has been acknowledged, first runs the test-specific initialization code, and then start the user threads. If the researcher programmed their test-specific initialization code correctly, then whenever the satellite starts a new test the conditions of the satellite will be the same. Note that the SOH packet downloads full test information when a test is running, so that each test can be clearly identified.

> *The SPHERES state machine implementation, coupled with hardware reset capabilities, ensures that tests can be stopped and started with minimal overhead.*

4.3.1.5 Direct link to ISS data transfer systems

While both video and experiment data are easily available in ground testbeds through local computers and camcoders, the iterative research process can be greatly disturbed by lag of data transfer while operating aboard the ISS. Previous experience taught the SPHERES team to minimize the need for communications hardware outside of the available ISS resources, such as using our own hardware for storage of data and/or requiring transfer of physical items such as CD’s or video tapes. Therefore, the SPHERES maximizes the use of ISS resources to minimize the amount of time spent transferring data between systems, while at the same time balancing the fact that this makes the collection of SPHERES data and video dependent on a third party (NASA).

The SPHERES facility utilizes an ISS laptop as the control station. The SPHERES interface runs directly on that laptop; all the data received by the laptop are saved directly to the laptop (at the end of an ISS test session a single compressed file collects all the data). NASA has provided the SPHERES team with a shared drive for use, which means the SPHERES data files will have a direct link to the main ISS server. Under normal opera-

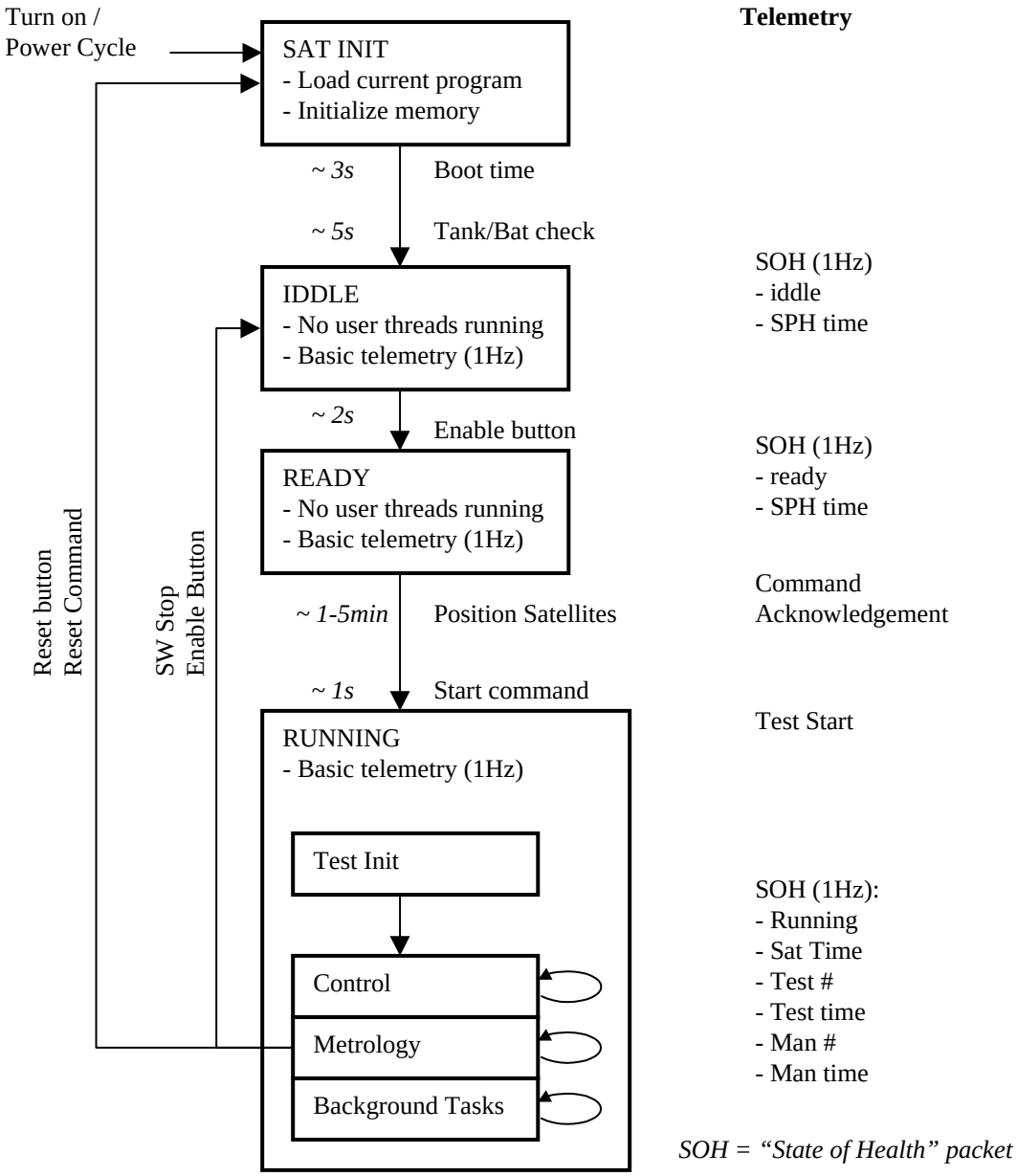


Figure 4.14 SPHERES satellites initialization

tions the astronaut will not need to transfer files manually, the operation occurs automatically. The data from the main ISS server is downloaded every day by JSC. At that point the team depends on the JSC staff to forward the data to the SPHERES team; this usually occurs within 24 hours.

SPHERES also utilizes the ISS video capabilities for both capture and download. While this process can take longer than data download, it is still faster than any custom solution. Utilizing the ISS existing hardware prevents the need for further mass/volume if SPHERES custom video equipment were required. To obtain reasonable download times the ISS data system would still have to be used for video download, possibly slowing down the transfer of data overall. Because downloading server data and video are separate processes at NASA, the use of the ISS video system decouples the download of the two sets of feedback.

As a result SPHERES does not require of any data transfer between custom media and does not need to wait for the physical return of any data storage facility. The only SPHERES specific communications hardware is the STL (Satellite-to-laptop) transmitter, which connects directly to the ISS laptop.

> *A direct link to the ISS data system minimizes the time to collect microgravity data and requires no transfer of physical media.*

4.3.1.6 De-coupling of software with NASA safety controls

Throughout the development of the SPHERES facility great care was taken to ensure that the software was not part of NASA's safety controls. As explained above, the software facilitates the iterative research process by allowing scientists to change their implementation at many levels. If the scientist discover that only small changes are needed, but the SPHERES software had to go through NASA safety reviews to be implemented, then the overhead for small changes would be disproportional to the changes. To ensure that the implementation step during ISS operation remains reasonable, it was essential for the SPHERES software to remain independent of NASA safety controls.

To achieve this goal the hardware sub-systems provide the double and triple redundancy required by NASA. The NASA safety reviews concentrated on the propulsion, power, and communications sub-systems, on the fact that the SPHERES satellites are free-floating in

the ISS, and on the noise levels produced by the satellites and metrology hardware [SPHERES, 2001].

The propulsion system provides hardware relief valves to ensure that the unit is not dependent on the software to vent the unit in case of pressure build-up. Further, even though not required by the NASA safety review panel, the propulsion hardware driver circuits defaults to an off state, such that resetting the SPHERE will always close the solenoids and stop any firings, independent of test-specific software. Lastly, the amount of thrust exhibited by the satellites is such that an astronaut can hold a satellite until the gas tank depletes without any hazards. The power system provides hardware current limiting devices directly in the battery packs (redundant), a magnetic circuit breaker, and always operates below 32V. These specification mean that the software can never cause a safety hazard situation due to electrical power. The communications system defaults to idle mode, and requires initialization from the software. Further, the reset signal is sent independently to the communications hardware, ensuring that a reset forces communications to stop regardless of the state of the software. Lastly, the dynamics of each satellite were designed so that even in the case of a software failure any collision is under the limits specified by NASA and the noise produced by the satellites was designed to be below NASA requirements during all types of operations, including excessive thrusting. These measures ensure that once a researcher decides to change a program, this can be uploaded to the SPHERES project without any delays due to software reviews.

A further benefit of the lack of safety checks for the ISS is their applicability to operations in multiple NASA locations, including the KC-135 Reduced Gravity Airplane. The software cannot cause any hazards by itself, and therefore does not require verification. The lack of safety controls in the software ensures that such lag does not exist, and that the iterative research process is only affected by the availability of communications links to and from the facility.

> *De-coupling all safety controls from the software minimizes the overhead to deploy new algorithms and allows scientists to program algorithms that push the limits of the theory without delaying the iterative research process.*

4.3.2 Support of Experiments

The need to repeat experiments in an efficient matter is an essential part of the iterative research process. As presented in Chapter 3, the "support of experiments" features directly affect the ability to enable the iterative research process, but they go into further detail on how to achieve efficient tests than the high-level feature of "facilitating the iterative research process". The last section presented the high level operations plan and other special characteristics of SPHERES which facilitate the iterative process over all. This section will present in more detail the several specific functions of SPHERES that directly affect its ability to run an experiment correctly and efficiently.

The following functionalities of SPHERES enable it to support conducting experiments:

- Data Collection and Validation Features
  - Layered metrology system
  - Flexible communications: real-time & post-test download
  - Full data storage
  - 32 bit floating point DSP
- Redundant communications channels (reliability)
- Test management & synchronization (repeatability)
- Location specific GUI’s (human observability & manipulation)
- Re-supply of consumables (repeatability, supporting extended investigations, risk-tolerant environment)
- Operations with three satellites (reliability, risk tolerant environment)
- Software cannot cause a critical failure (risk tolerant environment)

4.3.2.1 Layered metrology system

Chapter 3 calls for the following requirements on data collection (among others):

- Ensure data accuracy and precision scalable to the final system

- Ensure observability of the technology

Developing the SPHERES environment presented a major challenge in the development of a valid metrology system. Not only was a 6DOF measurement system for use inside the ISS was not available "off-the-shelf", but the design of SPHERES called for it to be a development testbed for metrology algorithms itself. Therefore the metrology system had to provide scientists with both the necessary data and computation power to both develop new metrology algorithms and to trust the data if they use the SPHERES provided procedures.

To obtain the accuracy and precision scalable to the final system the SPHERES metrology system utilizes a two-layer implementation. High-frequency accelerometers and gyroscopes capture the inertial motion of the satellites. The selection of the COTS accelerometers and gyroscopes was driven by the need to provide accurate data for the expected motion of the satellites using the cold gas thrusters. With a thrust of approximately 0.125N and a mass of 4kg per satellite, each thruster would cause an on-axis acceleration of approximately 0.03125 m/s^2; the thrusters are located approximately 0.125m off axis, providing a torque of approximately 0.004Nm, which equates to a rotational acceleration of 0.25rad/s^2. Therefore, the accelerometer must measure 3.2mg's, while the gyroscope range was selected as ±50°/s (~0.9rad/s) since the expected thrust periods are no longer than one second (which results in approximately 30°/s with two thrusters firing on axis). Table 4.6 summarizes the satellite dynamics under thruster actuation.

TABLE 4.6 Satellite dynamics under actuation

	Value
Single thruster force	0.125N
Minimum opening time	10ms
Acceleration	0.03125m/s^2 (~3.2mg's)
Rotational speed (minimum impulse)	0.25rad/s^2

The gyroscope selection was relatively straight forward, as the range of rotation rate was easily available from COTS equipment and the thrusters have a large enough level arm to easily excite the single-bit resolution of a wide range of gyroscopes. The final selection took place for their size and small drift which allows rotational control of the units for extended periods without the need for the global metrology system. The gyroscope specifications are presented in Table 4.7.

TABLE 4.7 Gyroscope specifications (BEI Gyrochip II)

Specification	Value
Input range	±50°/s
Scale Factor	30mV/(°/s)
Bias stability (<100s)	0.05°/s
Bandwidth	50Hz
Sensitivity (12bit A2D)	0.0407(°/s)/bit

The selection of the accelerometer presented a bigger challenge, since the thrusters linear acceleration is in the milli-g's range. Therefore the selection had to trade-off between selection accelerometers which would provide feedback on external disturbances (such as collision of two units at dock time or forces by humans) or from the thruster firings. The final selection gave priority to measuring thruster actuation. The accelerometer will saturate whenever external forces, which will be much larger than the thruster forces, are applied; the saturation of the accelerometers can be used as a binary method to detect external forces, even if not to model them. The characteristics of the accelerometer selection are presented in Table 4.8.

Testing of the gyroscopes and accelerometers in the KC-135 reduced gravity airplane showed the ability of the sensors to detect the thrusters. Figure 4.15 plots sample data from the micro gravity tests. The data are from testing firing thrusters number zero and one, which produce acceleration on the X axis and rotation about the Z axis. The top plot presents the accelerometer readings; the bottom plot the gyroscope data. While in an ideal

TABLE 4.8 Accelerometer specifications (Honneywell Q-Flex QA-750)

Specification	Value
Input Range	±30g
Scale Factor	1.33mA/g
Resolution	1μg
Bandwidth	300Hz
Amplifier Gain	2000Ω * 40
Sensitivity (12bit A2D)	.011473 mg/bit
Effective Range	±24mg

situation the accelerometer data would show a square wave as a thruster turns on and off, the plot shows the effects of placing the accelerometers off axis. The sensitivity of the accelerometers is so low that they measure not only the thrusters, but also the centripetal acceleration due to the offset. Following the plot piece wise we see the following parts:

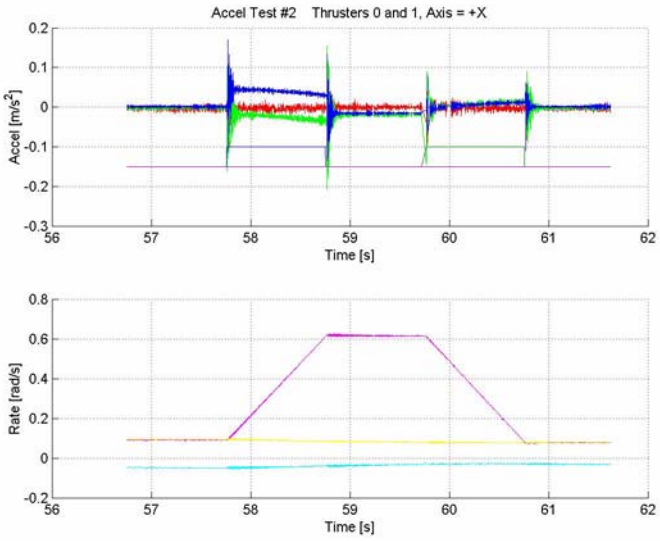


Figure 4.15 Accelerometer and gyroscope measurements in micro gravity

56.7-58.7s: initially at rest; no acceleration or rotation

57.7-58.7s: Thruster 0 turns on, causing +X acceleration (top lot blue line) and +Z (bottom plot purple line) rotation. Note that as the rotational speed goes up,

the off-axis effects cause the +X acceleration to go down. Also note that thruster zero causes a small effect on the Z axis accelerometer.

58.7-59.7s: All thrusters are off again; note that the accelerometers do not return to zero because of the rotation of the unit.

59.7-60.7s: Thruster one turns on, stopping the rotation of the unit. The effect of the thruster is not seen much in the accelerometer in absolute numbers because the acceleration due to rotation was similar to that caused by the thruster; the effect is seen my the difference in accelerometer readings between the past period and this period.

60.7-61.7s: The units return to rest; with no more rotation the accelerometers return to their zero state values.

A simplistic model of the SPHERES satellites which does not account for the coupling between rotational speed and the accelerometer readings would not be able to use the accelerometer data. Therefore, while the accelerometers can provide full observability of the system, this is only true if the correct model of the satellites is used.

A low-frequency ranging system uses a combination of infrared and ultrasound signals to measure distances using the "time-of-flight" of the ultrasound signals and provide position and angular direction in 6DOF within a pre-defined operating area. Five external transmitters are used. An infrared pulse (treated as instantaneous) commands the start of ultrasound pulses and marks time "zero". The transmitters pulse one at a time in 20ms windows (allowing up to 6m of travel by the ultrasound). Each satellite uses 24 ultrasound receivers, four per face, allowing scientists to use the information for both triangulation of position and differential measurement for angular direction. Scientists can use the direct measurements or filtered data as best fits their application. Figure 4.16 illustrates the raw distance measurements of the global metrology system. The system provides resolution of ±5mm and ±2° in a 3m by 3m space.

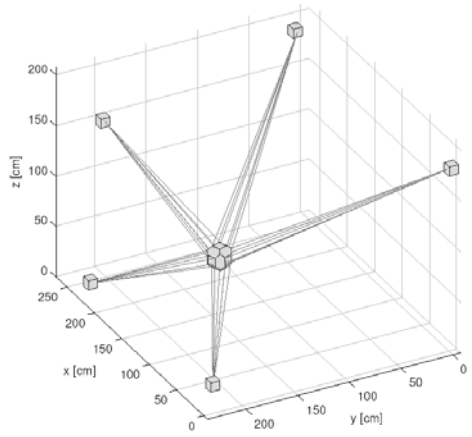


Figure 4.16 Global metrology system time-of-flight distance measurements

Although the actual observability of the system depends on the implementation of the model by each scientist, the SPHERES metrology system allows measurement of all the elements of a standard state vector for dynamics and control for a second order rigid body system such as a SPHERES satellite: position and velocity (linear and angular). Figure 4.17 illustrates how the state vector for each satellite can be filled by the use of the global metrology system and the inertial measurement sensors.

Each satellite also includes its own beacon to accommodate a different type of state vector: differential states between two satellites. This state vector can be used in the development of docking algorithms as well as formation flight maneuvers which require direct measurements between satellites, rather than by finding the differences using the global system. Figure 4.18 illustrates how the metrology system can be used to fill the state vector between two different satellites. The satellite beacon systems have a limitation due to the beacon location (directly on the X axis): it is not possible to use the pathlength differences to determine the attitude between the two satellites along the X axis.

4.3.2.2 Flexible communications: real-time & post-test download

Selection of metrology is not good enough. We need to make sure we can save all the data we need. The team had to recognize the limitation of the wireless communications bandwidth and account for that. Therefore the communication system is flexible in several

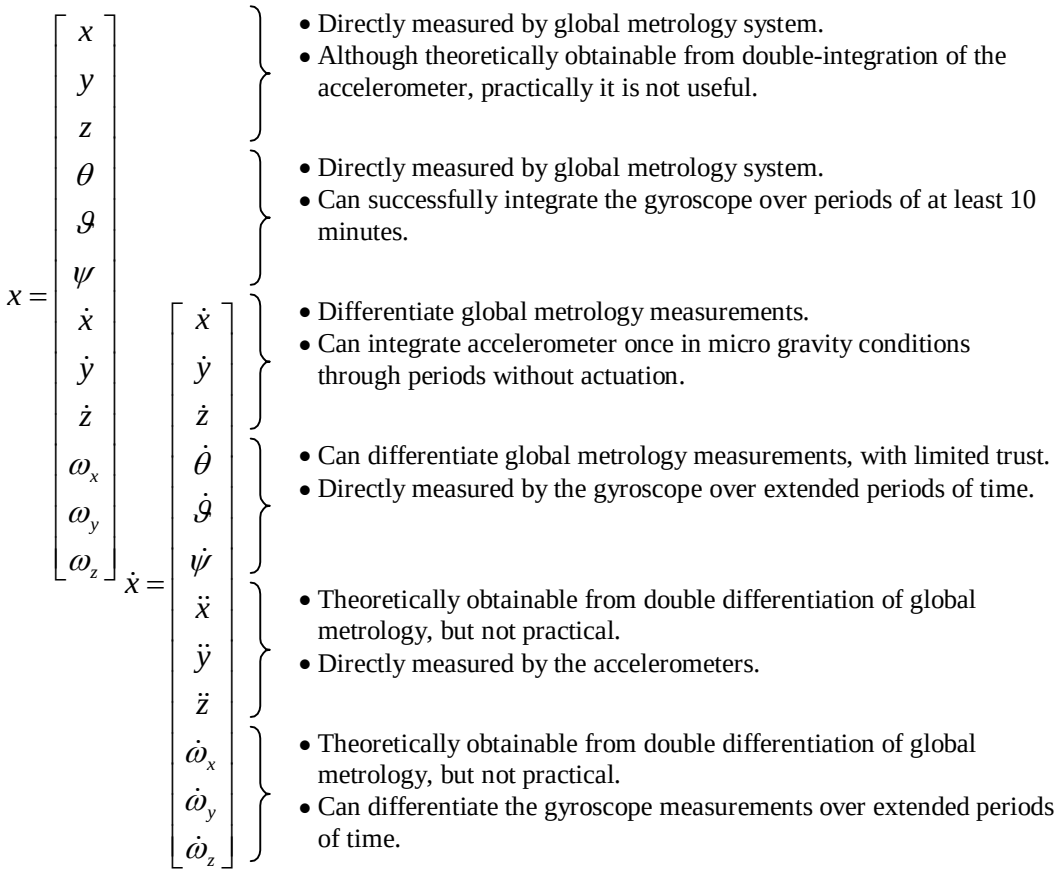


Figure 4.17 Measuring the state vector with the layered metrology system

ways: it has two types of data download (real-time and post-test) and can use custom protocols for inter-satellite communications.

The real-time data download must be transmitted within the TDMA windows of the satellites, meaning that data can be sent between units at up to 5Hz (200ms frames). Each communications packet consists of up to 32 bytes of data and takes approximately 15ms to transmit. The total amount of data that can be transferred in real time ranges from 320 bytes at 5Hz (12.8kbps) for one unit to 64 bytes at 5Hz (2.56kbps) for five units. Since synchronization of the STL channel with the laptop is essential for the test management elements of the SPHERES software, the STL channel must always remain under the TDMA protocol and strictly limited to these data rates.

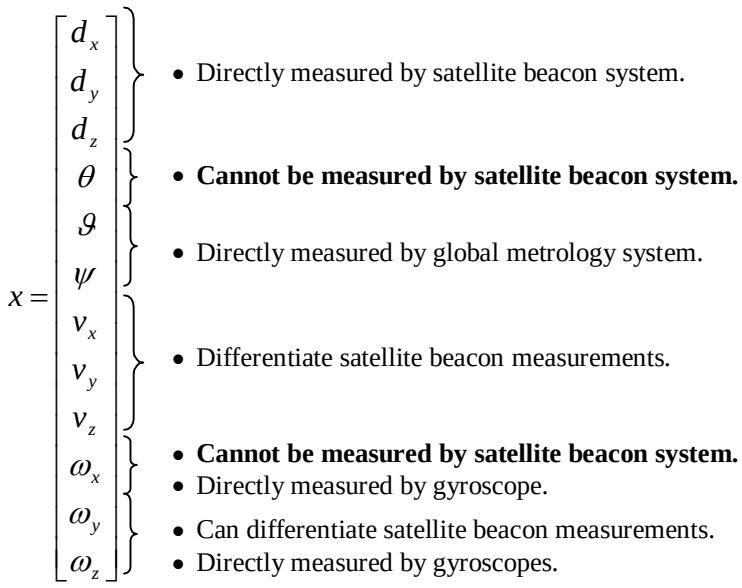


Figure 4.18 Differential measurements between two satellites.

The SPHERES layered metrology system allows full observability of a second order rigid body system in an inertial and a reference frame.
The satellite beacon system further allows limited direct measurement of distances and angles between two satellites.

The communications bandwidth creates a necessity to trade-off data download and operations time. The communications bandwidth is large enough to help scientists see real-time results of debug and processed data in the laboratory environments and to capture processed data in the space environment (since many of the algorithms will have been tested to trust the data). But it can be too limiting for tests which require high frequency data capture or when several units are used. When low-frequency data capture is enough or once processed data provides the necessary information, all data download can occur in real-time. But if large amounts of data must be saved to provide the necessary information, SPHERES allows scientists to download the data after a test concludes.

Of the 16MB of RAM available in each satellite, up to 10MB are available for data manipulation and storage (depending on the specific program). This memory can be allocated both statically or dynamically to create one or more data buffers. A test can be pro-

grammed to continue until all the buffer is emptied, even if the actuation maneuvers have ended. Further, the test management portion of the SPHERES core software can be programmed so that after a test is run the data buffer is left intact and a second test can then be run to download the data buffer; this buffer can be shared among multiple tests and as long as the *download* test is run in between all other tests the data will be safely downloaded. This operating scenario presents a trade-off: utilizing operations time solely to download data rather than to test more algorithms. Scientists will have to balance their need for high-frequency unprocessed data and the available time to operate the satellites. Figure 4.19 presents a sample algorithms where tests 1...N-1 run algorithms and collect large amounts of raw data in two satellites. Tests N and N+1 are run after each of the other tests to download the data. The time spent on tests N and N+1 must be considered as an overhead to the iterative process, but could be essential for the science.

To accomplish this the communications functions interface with the data solely via pointers, rather than by copying data from the original location to a communications buffer. Doing this minimizes the memory space used by the communications interfaces, but it also places the responsibility on the scientists to ensure the data remains safe prior to its transfer. The core software implements four mailboxes of 20 packets; these mailboxes are managed internally by the software to accommodate data of two different priorities in the two communications channels, but are not available as storage space for the user. The SPHERES software routines copy the data from the memory separated by the user to these mailboxes automatically, and transmit the data out of the mailboxes following the TDMA protocol.

The flexibility of the communications system is further enhanced by the ability to use the STS channel using custom protocols and software. While the STL channel is constrained to use the TDMA protocol, the STS channel can be programmed differently and independently. The core software implements a TDMA protocol by default, also operating at 5Hz. The lack of a *ground* station (the laptop) allows inter satellite communications to use the full window to transfer data, with data rates ranging from 2.56kbps per satellite for trans-

```
Test 1
  Init
    Setup comm for multiple units
    Initialize buffer
  Run
    Actuation
    Raw data collection -> Buffer
    Data processing -> RT Comm

Test N
  Init
    Setup comm for single unit (Sat 1)
  Run
    Download comm buffer

Test N+1
  Init
    Setup comm for single unit (Sat 2)
  Run
    Download comm buffer

Test 2
  Init
    Setup comm for multiple units
    Initialize buffer
  Run
    Actuation
    Raw data collection -> Buffer
    Data processing -> RT Comm

Test N
Test N+1

Test 3
Test N
Test N+1
…
```

Figure 4.19 Sample real-time and post-test data telemetry algorithms

mission between five satellites up to 17kbps for one satellite transmitting full time to the other SPHERES.

4.3.2.3 Full data storage

During the initial development of SPHERES strong emphasis was put on the ability to save processed data, ready for real-time display and immediate analysis. The software

> *Flexible communications allow telemetry download in real-time for limited data and post-test for large amounts of data; the use of post-test data download requires the scientists to trade-off the amount of data with operations time.*
> *The inter-satellite communications channel can be programmed independently of the satellite-to-laptop channel to use a wide range of protocols or default to the core TDMA algorithm.*

which operated in the laptop would not only receive the data and check the packet integrity, it would also translate the raw binary data into special structures such as the state of the satellites; the software would only save the processed data. This practice proved to be too restrictive as any change in the communications structures would require not only to modify the software of the satellites, but also the software of the laptop (during the prototyping stages of SPHERES there were at least six version of the laptop software, depending on the data to be saved).

To facilitate data collection of any type the SPHERES communications design defines only a limited number of special packets; further, all data is saved in its raw form as received by the communications hardware. The special SPHERES packets are:

- General Purpose Commands - outgoing: from the laptop to the satellites. These packets send the satellites information on starting and stopping a test, resetting the units and/or control variables, and starts a frame. The packet is transmitted at 5Hz from the laptop.
- Initialization Packet - outgoing: from the laptop to the satellites. These packets transmit the measured setup of the metrology transmitters such that the satellites can use the global metrology system.
- State of Health - incoming: from the satellites to the laptop. The SPHERES core software automatically transmits its state of health at 1Hz. The packet transmits information about the satellite on-time (since the last reset), the current loaded program, tank usage, test management, current operating mode, and the individual satellite's role in a multiple satellite configuration.
- Background Telemetry - incoming: from the satellites to the laptop. By default the core software queues these packets at 10Hz; they are downloaded as the test progresses. These packets transmit the estimated estate of the satellite as determined by either the SPHERES core estimator or a scientist specified function. The frequency of these packets can be modified by the scientist as needed, and can be stopped if desired.

- Debug Packets - incoming: from the satellites to the laptop. A special packet with up to 16 shorts (16 bit integers) which can be used as needed by the scientists.
- "Datacomm" packets - incoming from the satellites to the laptop. These are special packets used to split data longer than 32 bytes automatically. When used between satellites the data is re-assembled automatically on the receiving units; the laptop simply saves the raw packets without processing. The protocol which manges these packets enables scientists to transmit large amounts of data without having to worry about splitting it themselves.

No other special packets were deemed necessary by the SPHERES team after substantial use of the satellites in several environments and development of the interfaces (described below) together with NASA.

Only these special packets are processed in real time in either the satellites or the laptop by the core software. General purpose commands and initialization packets are processed only by the satellites. The state of health packet is processed in real time by both the satellites and the laptop; the satellites use the information to configure multiple-satellite tests. Background telemetry is processed in real-time in the laboratory environment to expedite tests; the satellites also process these packets in case the scientists need to share state information between units. Debug packets are only processed in real-time by the laptop station in the laboratory environment to help test algorithms. Datacomm packets are only processed in real-time by the satellites to allow scientists to share large amounts of data between units in real-time; the datacomm packets are re-assembled post-test from the laptop telemetry.

In all cases, including outgoing laptop packets, all received data is saved by the laptop programs intact. The full received data allows scientists to create their own data types for examination after the tests without restrictions. Further, it allows scientists to perform error detection and even error correction post-test. Scientists can save data manually and even design algorithms which would not be possible to run in real-time to detect lost bytes and save data in case of communications errors.

In all cases the full incoming and outgoing data from the laptop are saved intact. Only a limited number of clearly defined packets are necessary for real-time processing of data, and even these are saved as received.

4.3.2.4 32 bit floating point DSP

The selection of a computer which would allow scientists to perform their calculations with the needed precision had to be balanced with the need to minimize its size, mass, and power consumption (and in turn heat dissipation). The scientific goal of SPHERES (mature formation flight algorithms) strongly called for the use of a processor which would allow a large number of precise mathematical calculations. The majority of COTS micro-computers utilize fixed point MCU's. Several fixed point processors are capable of more than 1 GIPS, but their performance with floating point calculations drops considerably (up to 16 fold); their benefit is a small form factor and small power consumption. Standard 32bit microprocessors used in PC's, although powerful and capable of over 1 GFLOPS, are prohibitive in their power consumption. Still, the SPHERES team decided that a floating point processor was essential for the success of the mission.

The selected 32 bit floating point Digital Signal Processor allows scientists to not only collect high-precision data (at this point the limits on raw data collection lie within the sensors, and not the processor), but to also process this data without loosing any precision from the original measurements. The scientists does not need to worry that utilizing floating point numbers will hinder the ability of the processor to maintain accuracy or have enough processing power. Further, the inherent support of floating point values by the processor makes it trivial for these numbers to be transmitted between units and saved in telemetry without any overhead to the processes.

The use of a 32 bit floating point processor allows scientists to perform precision mathematical calculations to maintain data integrity through any data processing.

4.3.2.5 No precision truth measure

The conclusion on the data collection and validation aspects of the SPHERES facility must conclude with the remark that the facility lacks a precision truth measurement system. Throughout all operations video will be available, sometimes in real-time, to allow scientists to observe the behavior of the units. This video can be post-processed to validate maneuvers in a coarse manner. It could be possible to add markers to the satellites such that image processing software could help calculate the behavior of the units from the video. Still, even with these options, the team has determined that there is no truth measure which can provide data of the same precision as the metrology system of the facility. In the best cases the telemetry from the satellites will be corroborated by coarse motions in the videos.

4.3.2.6 Redundant communications channels

While the selection of 2 communications channels was due to simulating both satellite to ground and satellite to satellite communications, the SPHERES facility allows these two channels could be used interchangeably to enhance the reliability of the testbed. In the case that one of the channels fails, the second channel can substituted with minor changes. The default configuration defines STL as the 868MHz equipment and STS uses 916MHz. Two laptop interfaces, one of each frequency, are always available in the laboratory environment, and two will be sent to the ISS. If an 868MHz channel fails in a satellite or the laptop, it can be replaced with the 916MHz channel. The core software, which interfaces between the hardware and the user program, was designed to allow changing channels with a single command in the initialization routine. Further, the bootloading software, which allows new programs to be loaded into a satellite, interfaces with both communications channels in identical manners. Therefore, in the case of a failure, new software can be programmed which swaps the use of the channels, ensuring the availability of telemetry and continued (even if limited) operations.

> *The ability to use the two communications channels interchangeably increases the reliability of SPHERES.*

4.3.2.7 Test management & synchronization

As described above, the SPHERES core software enables scientists to program *families* of tests at a time, such that running these tests has minimal overhead. The SPHERES test-management software provides several other features to enhance the repeatability and reliability of tests:

Automatically run the initialization code whenever a test starts. The SPHERES core software provides scientists with two initialization areas: program and test. The program initialization code sets up the satellite properties which will be in effect for the full family of tests. These initialization steps only take place once upon boot. To simplify the repeatability of individual tests the code also provide a routine which is run before a test is started, without the need to reboot the satellites. This initialization routine can be used by the scientist to ensure that all initial conditions are set correctly every time a test is started. Further, the independent initialization routine allows scientists to use the same control code to test for different initial conditions without the need to program multiple controllers. These initialization routines are expected to be simple and quick, so that the controller can start immediately after the routine ends. If the scientist requires further initialization, they can programming the first maneuvers of a test to satisfy the initial conditions before the actual test maneuvers are conducted.

While the core software provides the functions for the scientist to initialize their code, it cannot guarantee that an individual scientists will initialize their program correctly. The responsibility to initialize their code correctly still lies within the researchers.

Manage the controller timing independently of the controller itself. A critical part of control algorithms is the correct timing of the controller. The majority of control algorithms use periodic processed to determine the actuation of units. The SPHERES core software accounts for this need by separating the timing of the unit and the controller

function from the control function itself by utilizing software interrupts of different priorities. Two high priority interrupts, driven directly from hardware timers, maintain the timing of the controller and the units time management. A middle priority software interrupt, triggered by the high priority timing software interrupt, performs the control task itself. While this does not guarantee that the controller software will terminate within its allotted period, it ensures the reliability of the timing information provided in the telemetry. This reliable timing information can then be used to identify controllers which overrun their allotted periods.

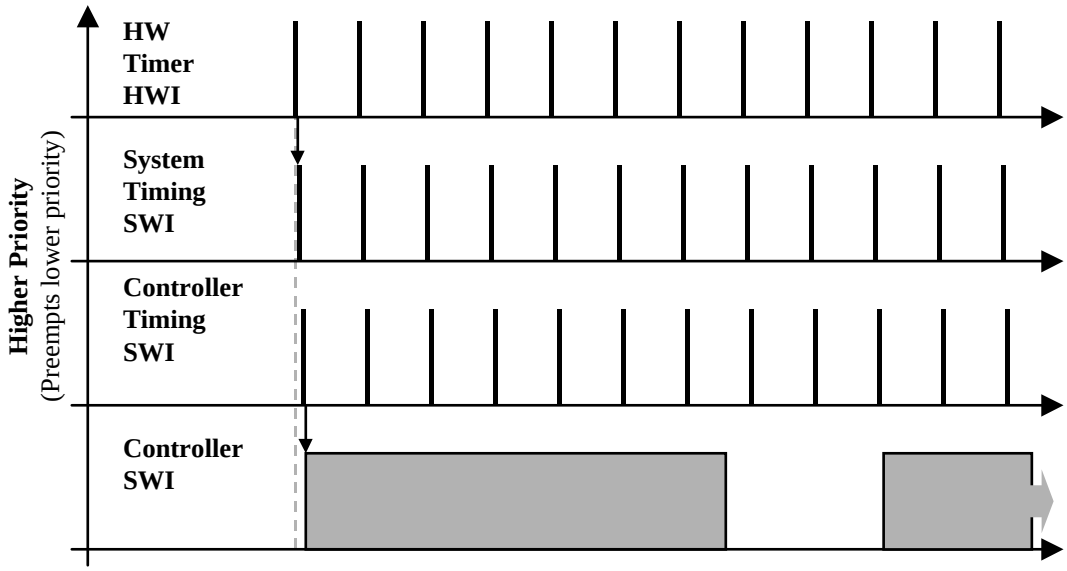


Figure 4.20 High priority scheduling of system timing and controller interrupts

Automatically synchronize test starts among multiple satellites. The ability to repeat tests of multiple separated spacecraft with ease depends on the ability to synchronize all the satellites every time, such that the same initial conditions can be repeated and controlled. Because SPHERES was designed to test algorithms of multiple satellites, the SPHERES core software integrates functions of the communications, avionics, and software sub-systems to synchronize the start of tests to within 1ms.

While the timing of the laptop synchronization packet is not precise, since it fluctuates up to 20ms, its reception by all satellites occurs at exactly the same time. This feature is used by the test-management software to define the time "-1000ms". A general purpose command which includes a "test start" also requires the packet to be acknowledged by all units. The laptop awaits the acknowledgement in a state of health packet, which is created immediately when the start command is received and transmitted in the next available frame after it is created. If the laptop does not receive the acknowledgements of all units within three frames (600ms) it will send a new start command, which once again resets the start time to "-1000ms". If all acknowledgements are received the laptop simply awaits for data, allowing the units to reach test time "0ms", which is the start of the test. Figure 4.20 illustrates this timing sequence for the operation of two SPHERES. In this example satellite number two looses the original start command, and does not send its acknowledgement. Therefore in the third frame after the original start command the laptop sends a new start command packet. Both units receive this new command and after 1000ms start the test synchronized to each other.

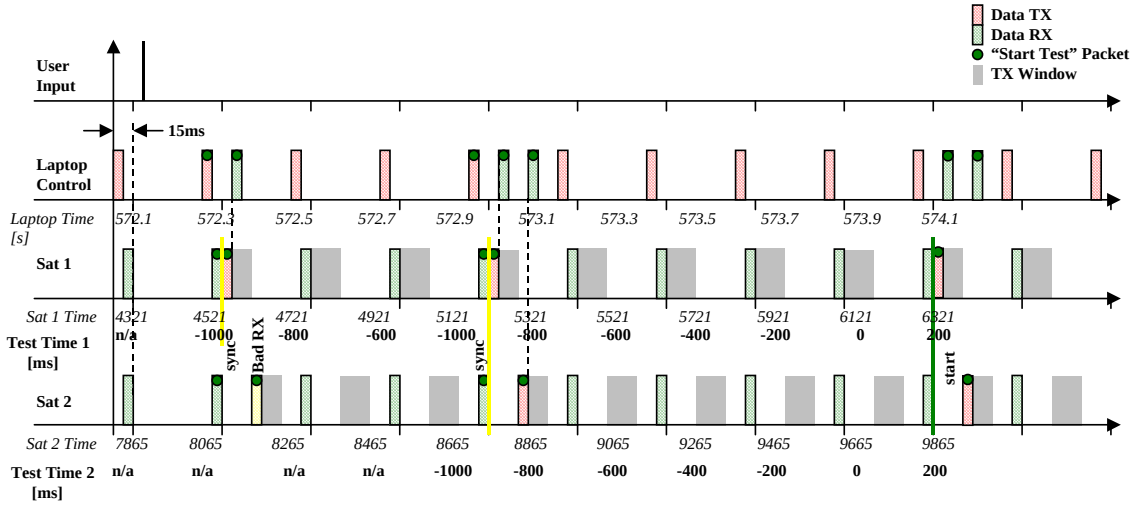


Figure 4.21 Test synchronization via communications.

The synchronization within one ms is achieved by utilizing high priority, non-preemptable hardware interrupts to process the general purpose command packets, while all other

packets are processed with variable latency in a separate lower priority software interrupt. The general command packet is always received by all units at the same time; the transmit time variability is within 60µs. The hardware interrupt has a fixed latency of 80ns and is processed within 60µs. The only variability is due to clock drift between the units. Since the internal time of the SPHERES is maintained to a 1ms precision, the variability of the start time is within 1ms. Figure 4.22 illustrates the fixed time to process a test start command as compared to the general processing of other commands.

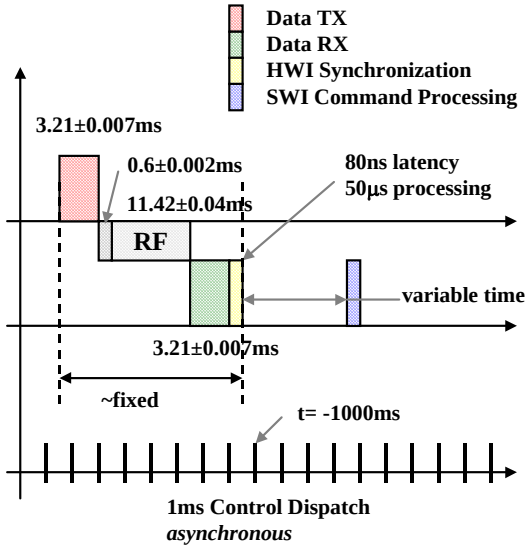


Figure 4.22 Test synchronization to within 1ms

Provide test timing information for post-analysis. Figure 4.21 illustrates a function of the test management software which contributes not only to repeatability and reliability of the tests, but also to good data collection: redundant test start packets. The state of health packet which is created immediately after a start command is received can be used to synchronize all the data between the units, since it provides the exact (to within on 1ms) SPHERES satellite on-time, used in the standard telemetry, at which point the start command was received - all tests start 1000ms after that time. The data management software sends a second start test packet immediately after a test starts, indicating both "test time 0"

and the satellite on-time. The redundant packets ensure that the collected data of multiple satellites can be synchronized.

> *The SPHERES test management procedures initialize tests, control periodic functions, and synchronize the start of multiple units. Redundant data is downloaded to ensure the data can be synchronized after the tests are performed.*

4.3.2.8 Location specific GUI's

While SPHERES clearly depends on humans to manipulate the satellites, the observability by humans is not necessarily guaranteed by the physical nature of the tests. For example, once precision alignment algorithms start to be tested, it may not be possible for humans to determine through without any other help if a test performed better than previous ones. Therefore, the correct instrumentation must be provided in the different environments to allow observability of the experiments by humans, so that they can make the correct evaluations on when to proceed with new tests and when to repeat them. At the same time the user interface play a major role in the ability to quickly repeat tests; the presentation of the correct data must not hinder the ability of the human to observe the test by distracting them with data overload, nor should it prevent the operator from quickly starting new tests.

The ground based GUI is a streamlined interface consisting of one main dialog window. This window provides real-time information on the status of all powered satellites with active communication links. To provide the user with enough feedback, the GUI displays the state of the satellite, its up-time, and current test information (test number, test time, and maneuver number). This information is useful in ground-based facilities since the operators are either the researchers themselves, or MIT SSL personnel with deep knowledge of the tests being conducted. The ground-based GUI also provides direct access to control the satellites with minimal need for input. This includes the ability to start and stop tests with a single keystroke, to force a hardware reset of the units remotely, and to reset

any communications channels. The ground based GUI also provides a streamlined method to upload new programs to the satellites, although this method requires direct knowledge of the individual files that must be loaded (as will be explained, the flight GUI adds elements to require less knowledge of the files, but the process adds more steps). The ground-based GUI also includes optional, separate windows, to show the state and debug vector of each operating satellite. This real-time presentation of data allows the researcher to immediately see if the satellites are calculating their state correctly, and reduces the time spent when a test is not operating correctly. By making these windows optional, this GUI ensures that only the information the researcher desires is present, simplifying the operations if so desired. Figure 4.23 shows a screen-shot of the ground-based GUI together with its optional display of real-time data (state - position, velocity, angle, and angular rate - and debug packet).

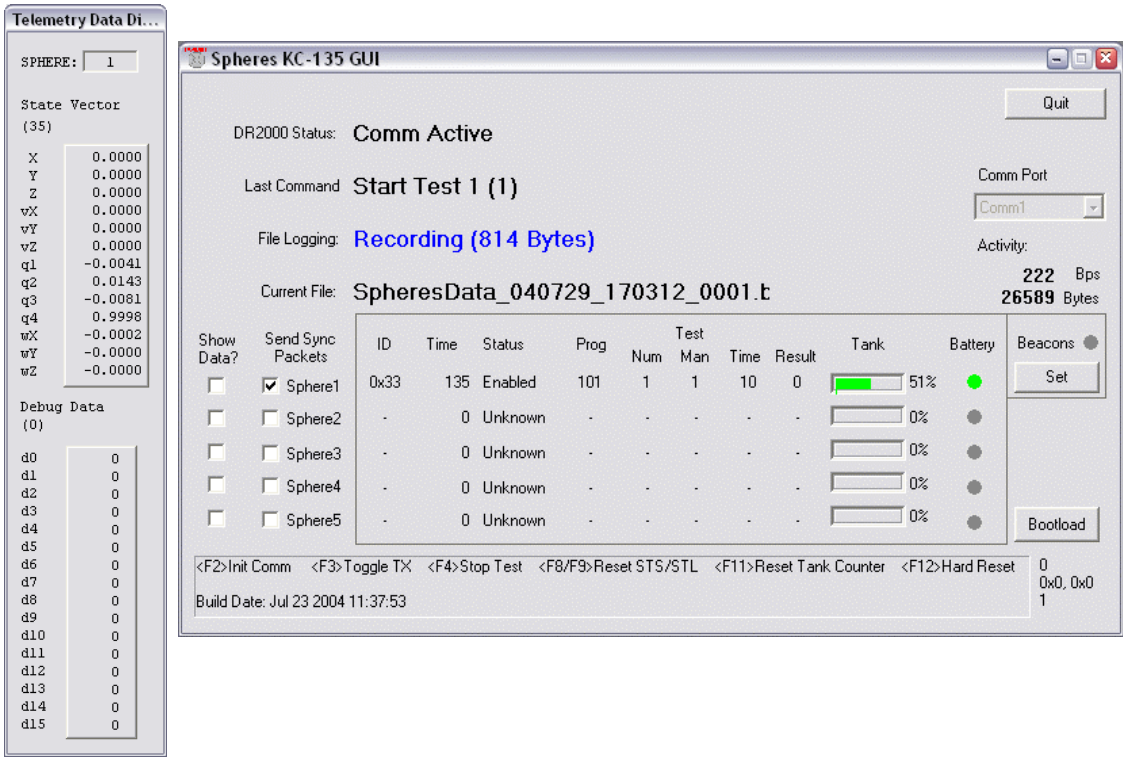


Figure 4.23 SPHERES GUI for ground-based operations

Operations aboard the International Space Station must consider the fact that the operations take place in a remote environment; the researcher is located at their ground facilities, while astronauts operate the tests. While the astronauts will have some knowledge of the SPHERES facility and its operation, they will not have the deep knowledge of neither SPHERES or the science being conducted as the SPHERES team members or the research scientists. Therefore, the SPHERES operational elements within the ISS provide a special graphical user interface which provides the astronauts with enough information to conduct the experiments, without them having to know the system or science deeply. On the other hand, the GUI also provides multiple points of input for the astronauts, to help determine the success of each test run.

Operations within the ISS must meet several NASA requirements that involve both safety factors and interface requirements. The safety factors that affect the operations include the need for the astronaut to require a positive action to enable the satellites prior running tests; the satellites must not perform any thrusting activity prior to being enabled. Further, the satellites may not transmit any communications unless they are in range of the control laptop and the laptop has enabled communications. The interface requirements include the need to maintain a window in the foreground if it provides an action to start or stop a test; the graphical requirements of the interface are not addressed in this section, since they did not cause any changes on the research aspects.

Figure 4.24 shows a screen capture of the flight GUI with the test introduction window displayed, as well as the test start window. The GUI provides the following basic information to the astronaut: status of communications, satellite enabling, battery charge, and tank fill level. The GUI also informs the astronaut of the program currently loaded, and the test (if it is running). Note that the ISS GUI does not inform maneuver numbers, since these are only used by the scientists or SPHERES team members for debugging or data analysis purposes.

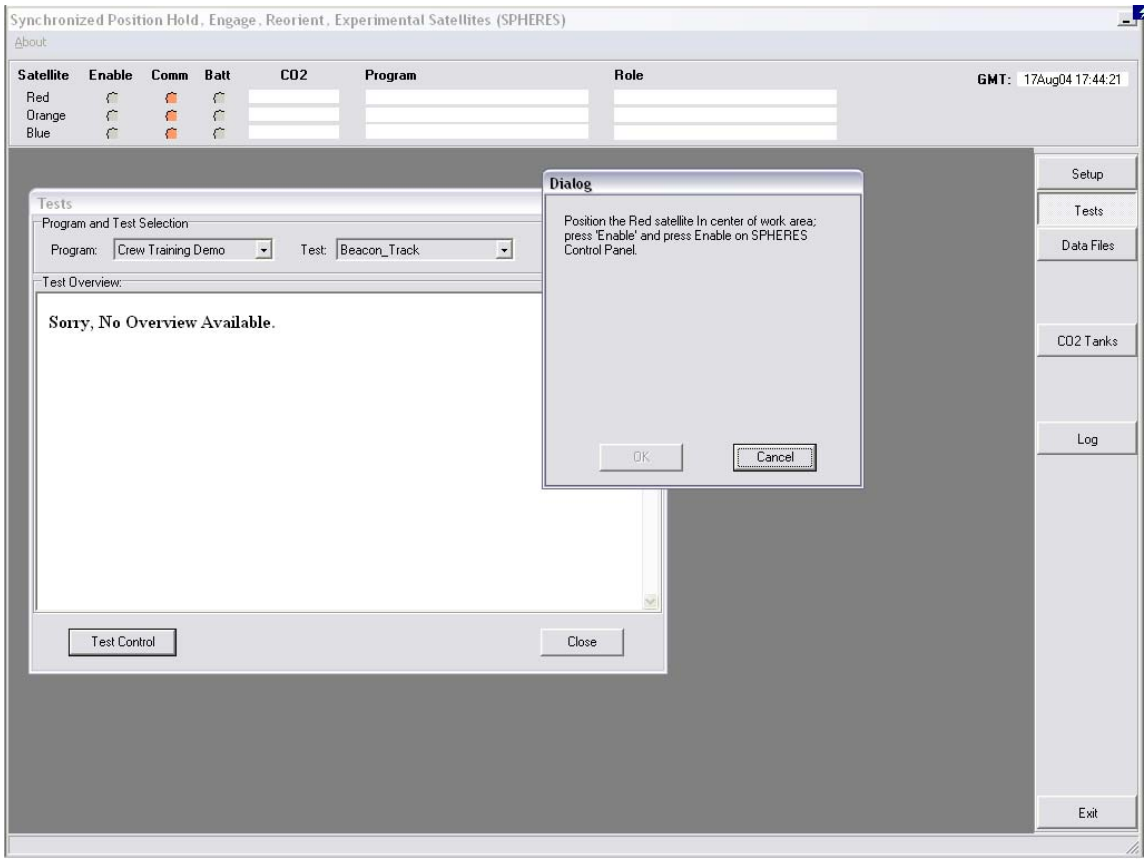


Figure 4.24 ISS astronaut interface

Within the test window the ISS GUI provides a description of the selected test, including the expected behavior, positioning, and in some cases either a picture or a movie that give the astronaut a preview of the test to be performed. This description and preview are especially useful for the iterative research process, since they provide the astronaut enough knowledge to determine on their own the success of a test. Using this knowledge, the astronaut is given the liberty to decide when to repeat tests and when to move on into the next test. While the astronaut will not perform any data analysis, the ability to determine success of a test maximizes the amount of useful data from each time-limited test session aboard the ISS.

The SPHERES software will automatically determine when a test finishes (the astronaut may also cancel the test if they determine it is not proceeding correctly). At that point the

GUI will present the astronaut with a pre-determined termination code (the astronaut will have a look-up table for each program of an ISS session). This code provides the astronaut with further feedback on whether the test was successful or not.

Afterwards, the astronaut is presented with a questionnaire written specifically for each test. The questions are drafted by the scientists to acquire fast knowledge of the success of the test in order to minimize the time spend in data evaluation; if an astronaut provides feedback that a test failed substantially once, but then performed correctly multiple times, the researcher may decide to only look at one good run and at the one bad run, to determine what was different.

Further, the astronaut is given the opportunity to enter free text into the questionnaire form. This open area effectively becomes a lab notebook for the astronauts, where they can inform the SPHERES team and researchers of any problems in executing the tests and any behavior not covered by the pre-defined questions.

Apart from the specialized interface, operations within the ISS will also provide video feedback of all operations. While the astronaut will have substantial opportunities to provide feedback, the researcher on the ground has the most knowledge of the expected behavior; therefore, it is essential for the researcher to corroborate the feedback of the astronaut by looking at both the data and the video of the operations. Past experience shows that astronauts may be more interested in the cases where algorithms do not perform correctly (MACE), rather than successful runs. Therefore, the researcher must be able to determine if a test performed correctly themselves.

> *Two GUIs were designed: one for researchers operating the satellites directly (maximizes real-time data availability and details) and one for astronaut operations (maximizes information on tests, provides summary results, and allows for astronaut feedback).*

4.3.2.9 Re-supply of consumables

SPHERES was designed so that its only physical limitation in mission life are easily replaceable consumables (gas tanks and batteries). Otherwise the design of SPHERES does not limit the mission life, which could be extended for several years. Of course, the operations do require that consumables can be launched to the ISS if they run out, which is not necessarily trivial. Yet, the challenges with sending new supplies are substantially less than with deploying new missions.

SPHERES will operate for at least six months in the ISS, and could be there for multiple years if the research calls for it and the resources permit it. Through this period researchers will be able to conduct extended, iterative investigations. The ability to re-supply consumables supports experiments in three ways:

Simplifies repeatability of tests. Not only does the resupply of consumables allow a large number of tests to occur, it also improves on the repeatability of initial conditions. If necessary, the operations can call for tests that are highly dependent on mass to be performed immediately after the gas tank is replaced, while other tests that do not depend on mass can be performed later. This allows initial conditions to be controlled as necessary.

Because SPHERES was designed to specifically allow the re-supply of consumables, this task was designed to be performed with ease. Replenishing the gas tanks or batteries takes less than one minute, adding only minimum overhead to repeat new tests in case an algorithm failure empties the tanks or batteries run out. Operators, be it in ground laboratories or the ISS, can repeat tests without major worry of consumables as long as replacements are available.

Enables extended investigations. The selected consumables are easily removable, even without being fully depleted, and have extended shell life whether used or new. Therefore, the researchers have wide flexibility in conducting their experiments. The consumables only deplete during the actual operations, and can be stored safely in between tests. In this

manner scientists have the ability to analyze data and modify algorithms over extended periods of time.

Creates a risk-tolerant environment. The ability to re-supply consumables allows researchers to continuously push the limits of their algorithms. Since depleting consumables does not result in the end of the mission, scientists can perform tests which could potentially deplete the propellant, but which could otherwise provide substantial insight into the science behind the algorithms. The ability to replenish the propellant allows scientists to test the high-risk but high-payoff algorithms which cannot be performed in other environments. By allowing researchers to find the true limits of the algorithms, each of their research iterations will be more productive.

> *The ability to resupply consumables provides three major benefits for experiments: provides repeatability, enables extended investigations, and allow scientists to push the limits of their algorithms.*

4.3.2.10 Operations with three satellites

The main driver in the final configuration of three satellites for the SPHERES facility was the need to perform substantial formation flight maneuvers with multiple satellites. This selection has a secondary effect: it improves on the reliability of the testbed. While three satellites will be needed to demonstrate several formation flight algorithms, the use of two satellites can still enable a substantial amount of science for distributed satellite systems.

> *Full understanding of the science needs allows layered reliability of the facility: with SPHERES the deployment of three satellites provides redundancy for tests that require one or two satellites.*

4.3.2.11 Software cannot cause a critical failure

The importance of separating the software from any safety controls was presented above. From the perspective of the NASA safety panel that would be enough; the software could

potentially cause a mission failure, as long as safety is not at risk. From the perspective of the SPHERES design plan the goal goes one step further: the software cannot cause a mission failure. This ensures that scientists can develop their algorithms to their limits; regardless of the program created by the scientists they are assured that if their program fails, they will be able to load new programs to try again.

The design of the SPHERES core software operating environment does not directly control the ability of any other sub-system to perform its functions, only how the data managed by the other sub-systems is processed. The core software could be fully redesigned without causing any failures of the equipment. In other words, the operation of any individual sub-system does not depend in any way on the operating software. The following points describe the de coupling of the software from the other SPHERES hardware subsystems.

- Communications. The communications sub-system interfaces with the core software via both inputs and outputs. The inputs consist of data to be transmitted and configuration commands. The outputs are data received and configuration command confirmations. The communications sub-system operates via two levels of firmware which isolate it from the core software. The processor, which runs the core software, cannot modify that firmware.

  The failure modes which can be caused by the software are purely operational. For example, the software could configure the DR2000 hardware incorrectly, preventing a satellite from communicating. Upon rebooting the satellite, the DR2000 returns to its default configuration. The software can also saturate the micro-controllers which transfer data to or from the DSP, or configure the data transfer rates incorrectly. The firmware automatically discards excess data, ensuring continuos operations; a problem of excess data is corrected automatically once the software reads or writes data at the correct rate. Like the DR2000 hardware, the microcontroller firmware returns to a valid configuration upon reset. The core software cannot cause the communications sub-system to fail permanently.
- Propulsion. The propulsion sub-system interfaces with the software via twelve digital output lines; there are no other interfaces. This sub-system requires special timing on the signals which actuate the solenoids. This timing is performed by external circuitry, which takes as its only input the digital signal, which indicates whether the thruster should be open or closed. The external circuitry determines which signal to create.

The solenoids do have a limit on their actuation frequency (50Hz); the external circuitry does not limit the frequency of operations to within this limit. Therefore it is potentially possible for the core software to drive the solenoids beyond their operational limits to the point of failure. Still, this problem would have to occur for a prolonged period of time without notice for a mission-critical failure to occur. The presence of humans in all tests minimizes the probability that the software can cause permanent damage to the propulsion system, since tests which overdrive the solenoids can be stopped and the satellite can be put into a debug mode which does not perform any actuation. At that point the software can be reprogrammed to prevent mission-critical damage.

- Metrology. The metrology hardware is driven by firmware which operates in an FPGA. It interfaces with the software via the general data bus of the microprocessor; its interface is the most complex of all sub-systems. The core software can configure the metrology system widely; it commands the transmission of infrared signals for global metrology, enables the global metrology sensors individually or collectively, configures the A2D conversion rate, and enables the on-board beacon. But the core software cannot change the actual firmware of the metrology system.

  The core software can cause temporary failures in the configuration of the metrology system, which could potentially saturate the processor and prevent operations. Like with the communications sub-system, the metrology system returns to an operational state upon resetting the satellite, and a debug mode can be entered to prevent further operational problems.

  The metrology firmware protects its hardware directly. The firmware prevents the infrared transmitters from being active for prolonged periods of time, which could cause the infrared LEDs to fail. The firmware also limits the A2D conversion rate to ensure that valid data is always available. The on-board beacon protects itself by only actuating within its established limits.

- Power. The power sub-system interfaces to the core software via digital inputs and outputs. The power sub-system provides the core software with a low battery indicator. The core software must continuously toggle the watchdog data line to prevent a hardware reset. The only failure which could be caused by the core software would be to not toggle the watchdog, which would cause continuous reset of the unit until it is put in debug mode. This continuos reset does not cause critical failure of any sub-systems.

- Software. The only mission critical failure which can be caused by the core software lies within the software sub-system itself. The ability to load a new program is the only mission critical software present in the SPHERES satellites. This software, referred to as the "bootloader", configures the satellites into valid configurations upon boot and allows the satellites to enter the

debug mode necessary to load a new program. This special part of the SPHERES software is treated as firmware, and is not changed when a new program is loaded; new programs are loaded into separate spaces of FLASH memory, and the bootloader ensures that it does not overwrite itself. But it is possible that once a valid program is loaded it could overwrite the bootloader, which would cause a mission-critical failure, since the unit would no longer be able to boot after a reset. Therefore, it is essential for the SPHERES team to ensure that the bootloader is not overwritten. The SPHERES core software provides a special interface to access the FLASH memory which restricts writing only outside the bootloader space. As long as scientists only utilize this interface to the FLASH the bootloader is safe. But since a scientist can modify the FLASH directly, the SPHERES team members must validate any software to ensure the bootloader is not overwritten.

All of the failures which can be caused by the core software to other sub-systems are not mission critical; they are temporary failures which can be corrected by resetting the unit and loading a new program which corrects the problem. The correct use of the core software provided by SPHERES ensures that the software created by the scientists cannot cause a mission failure.

> *The ability to prevent the software from causing a mission-critical failure allows scientists the freedom to push their algorithms to the limits of either the science or the hardware. In this manner SPHERES truly provides a risk-tolerant environment for development of new algorithms.*

4.3.3 Supporting Multiple Investigators

The original goal of SPHERES was to develop a testbed for formation flight and docking. These two subject areas constitute a part of the larger field of Distributed Satellite Systems. The MIT SSL identified the following major topic areas for study within DSS:

- Metrology – Each satellite in a DSS requires knowledge of both its attitude and position as well as that of the other satellites. One must investigate the need for absolute measurements (e.g. a radar pointing towards Earth) versus differential measurements (e.g. docking) and between coarse (e.g. radar) and precise measurements (e.g. interferometry).

- Control – The control fields vary over a large range. High-level architecture determines the type of hierarchy in the system (e.g. leader/follower); an example of an intermediate level is fuel-balancing algorithms; low level control includes rigid body control of each unit.
- Autonomy – One goal of DSS is to minimize human intervention. At a minimum, the main maneuvers of the system should complete autonomously; human intervention should only occur at high levels, such as specifying the current task.
- Artificial Intelligence – AI goes a step beyond autonomy by providing the extra advantages of automatic system reconfiguration and error detection and correction, among others. AI technologies in DSS help further minimize human intervention in the case of a problem or a new mission goal.
- Communications – DSS satellites require communications both to ground (high power) and between the units (low power). Each program must study its optimal communications configuration.
- Human/Machine Interfaces – Given the limited interaction between humans and free-fliers in space, the possible uses and interfaces between satellites and humans must be studied.

The final design of SPHERES contemplates the need for research on these areas. The design takes into account that maturation of these technologies will require the cooperation of multiple scientists. Providing a system that allows multiple scientists to participate in a research program creates a set of requirements that cannot easily be defined as a simple list of qualitative specifications. The requirements are qualitative in nature and of a broad scope. The most important, yet broadest, requirement is to provide as much operational flexibility as possible so as to meet the project goals.

SPHERES implements its operational flexibility through the following features:

- Guest Scientist Program
  - Information Exchange
  - SPHERES Core Software
  - GSP Simulation
  - Standard Science Libraries
- Expansion port
- Portability

- Schedule flexibility

4.3.3.1 Guest Scientist Program

Immediately after the design of the prototype units was complete, the SPHERES Guest Scientist Program came under development to create a true relationship between the MIT SSL and the guest investigators elsewhere. Based on past experiences, the MIT SSL knew that the creation of relationships with multiple scientists to use the same facility required the development of both logistical and operational tools which facilitate the interactions and minimize the physical presence of the scientists with the hardware. The GSP became an integral part of the SPHERES program, making use of both the human and computing resources available. The GSP was a major element in the definition of the scheduling of mission operations (requests to NASA) and the main driver in the design of the software interfaces.

The SPHERES Guest Scientist Program consists of information exchange, special tools (software and simulation), and operations plans. The operational characteristics are introduced in Section 4.3.1.1, which describes how scientists make the best use of the iterative design process through multiple iterative loops. One of the layers includes the development of algorithms in-house by use of a simulation, which can be performed independently by a number of guest scientists. Further, the operations of 2D laboratory tests at the MIT SSL have been designed to support guest scientists in multiple levels. This section describes the information exchange and tools developed to support multiple scientists in further detail.

Information Exchange

The initial communications with a guest investigator include delivering the description of the SPHERES testbed, including extensive numerical data (empirical and theoretical) on the characteristics of the satellites. Scientists receive information on the mass properties of the satellites, sensor characteristics and locations, and the thruster profiles. Through the first years of development, and even in ongoing programs, developing a full system-iden-

tification of the satellites has been an integral part of the Guest Scientist Program. This system ID will allow scientists to fully model the satellites to understand the differences between their intended applications and the SPHERES testing facility.

SPHERES Core Software

The goal in the software design was to create an architecture that was relatively easy to learn and flexible enough to accommodate a wide variety of the sophisticated applications in advanced control, estimation and autonomy. The main challenge during this process was to balance the often contradictory goals of usability and capability. The goal of ease-of-use called for a clear and logical model of software operation, and the automation of tedious or non-productive tasks. In contrast, the goal of versatile functionality suggested an emphasis on real-time performance and a flexible execution model. Clearly, the design must reflect these high level goals within the constraints imposed by the testbed hardware.

The model of structured, user-supplied routines was an attractive framework, and with the processing power available with the flight hardware, a simple operating system could be developed to meet these needs. An operating system was needed to improve interface and execution flexibility, and to allow multiple threads to execute concurrently

The Texas Instruments DSP/BIOS [TI, SPRU423B] real-time operating system, designed for DSPs such as the C6701, is used as the operating system on the SPHERES satellites. This product provides multi-processing capability, inter-process communication, and a number of input/output management tools. This simple OS (or kernel), interacts directly with the hardware and manages many of the details thread and interrupt handling. Through the addition of multiple distinct execution threads, the core housekeeping functions are separated from the test software. This separation ensures that activities such as communications and telemetry processing are not affected by any computationally-intensive algorithms supplied by the guest scientist. In addition, increased flexibility and other benefits of multi-threading are extended to the end user.

Although DSP/BIOS solved of problem of flexibility, it was necessary to take steps to simplify the user's interface to the core software and underlying hardware. Giving the guest scientist general access to the entire OS would give them maximum flexibility, but this approach is undesirable for several reasons. First, to use the DSP/BIOS operating system directly, the user would have to purchase and then learn how to use DSP/BIOS. Second, without knowledge about the structure of the user-supplied code, it would be very difficult for us to guarantee the performance of the housekeeping functions and to meet NASA safety constraints.

As a compromise, the user is provided with a strict framework into which specialized source code may be inserted. Each module is executed when certain conditions are met. This allows the core software to manage the experiment's execution. The user's code does not interact directly with the hardware or with the DSP/BIOS interfaces. This simplifies the guest scientist's learning process, ensures proper operation of critical housekeeping functions, and facilitates the implementation of the SPHERES simulator. The core services also manage communications between the different processes. This helps to prevent race conditions between the periodic and aperiodic processes by ensuring atomic functions are used when required. Critical variables are accessible only via functions that have been designed to guarantee the preservation of data integrity. Although this model is not flexible enough for general-purpose computing, it is well-suited to the specific applications of estimation, control and autonomy for which the SPHERES testbed has been designed.

The SPHERES Core Software (SCS) layer performs two functions. First, it acts as a buffer between the user-provided experiment code and the operating system and hardware. Mediating between these layers, the core services control the execution of the user-configureable processes and encapsulate the operating system and hardware-specific interfaces. Second, this layer performs a number of background activities that are critical to successful operations. These functions are summarized below.

- **Communications.** SCS is responsible for receiving and processing incoming communications packets, and for transmitting out-going messages when allowed to do so by the TDMA protocol. The communications module also manages transmission and reception of the messages generated by the experiment code, such as custom telemetry or command data. If a data transfer is too long for a single packet (32 data bytes), the communications module segments the transmission and sends one packet at a time. The communications module on the receiving sphere automatically reassembles the original message from the constituent packets.
- **Housekeeping and Telemetry.** The SCS performs a number of routine tasks automatically, without direct command by the user. During normal operations, the spacecraft monitors the tank fill status (by tracking thruster firing), battery charge level, and operational mode. In addition, automatic processes perform a rough estimation of the satellite state. These data are broadcast over the "State of Health" packets previously described.
- **Propulsion.** SCS interfaces between the user code and the digital outputs to the propulsion hardware. The simplest operating mode allows the user to command a fixed-duration firing. This approach mimics the standard practice on-board most real spacecraft. SCS also implements pulse-modulation and provides an approximation of continuously-variable control over force and torque.
- **Test Management.** The SCS implements the test management functions described in Section 4.3.2 above. It monitors the crew commands, and then initializes and begins the user's test. Once the test completes, the software disables the user code, the thrusters, and the active sensors. During the test operation this module ensures that the user code is run at the correct time and communication bound for the GSP layer is received correctly.
- **Metrology.** The SCS implements a special thread to run the MIT designed kalman filter routines in the background. Guest Scientists are given access to the data created by this module and the option to run their own metrology algorithms in parallel or in place of this module.

The usefulness of the GSP hinges on the interface to the user's code. The relationship between the SCS modules and the guest scientists interfaces is depicted graphically in Figure 4.25. The next sections describe the SCS execution model, which controls the threads, and the supplemental libraries which provide support for a wide range of sicentists.

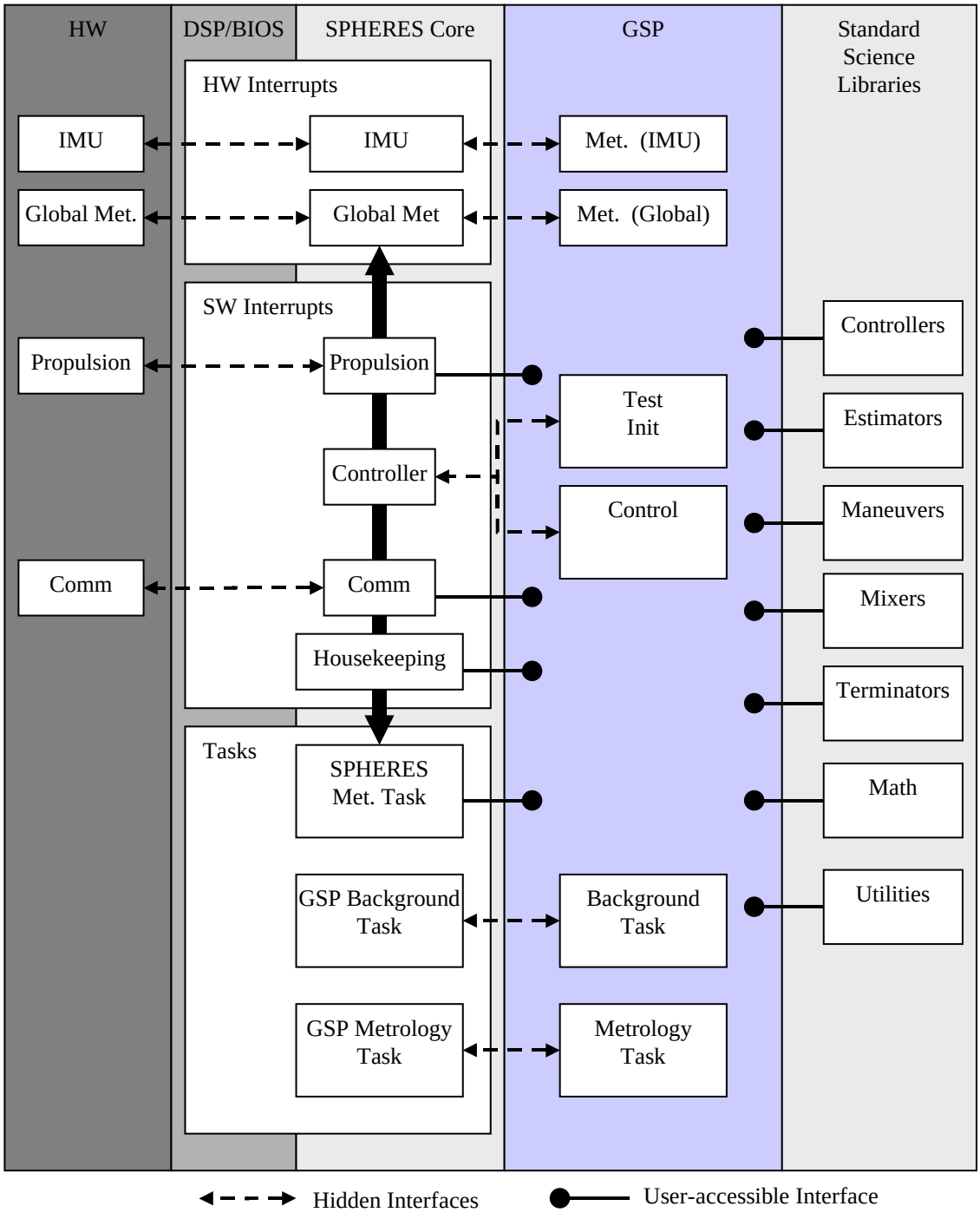


Figure 4.25 SCS interfaces to user code, DSP/BIOS, and hardware

The Execution Model

When writing experimental algorithms for SPHERES it is important to understand the manner in which the code will run. As mentioned earlier, the software framework describes certain modules that the user must provide. These modules are executed by the SCS layer when particular conditions are met. Some modules execute periodically, others in response to events such as incoming communications or sensors.

An important feature of the SCS architecture is that the code is multi-threaded. The highest priority thread waiting to execute is given control of the processor. This helps to guarantee that real-time deadlines are met. Although users cannot create arbitrary threads, they can mix periodic and aperiodic processing.

Guest scientists are provided with the module interfaces presented in Table 4.9 to develop their algorithms. These modules fall within four main threads of the SCS: initialization, control, metrology, and background tasks. The functions of each module are explained below.

TABLE 4.9 SCS guest scientist interface modules

Module	Thread	Repetition	Priority	Time Avail.	Typical Purpose
Program Initialization	Initialization	Once after unit reset	N/A	Long	Initialize the satellite for the full program
Metrology - Inertial	Metrology	Periodic; high frequency	High	Short	Capture inertial; sensor data; integrate data
Metrology - Global	Metrology	Periodic; low frequency	High	Short	Capture global sensor data
Test Initialization	Control	Once at test start	Medium	Short	Initialize individual test
Control	Control	Periodic; mid frequency	Medium	Medium	Periodic controller
Background Task	Background	Aperiodic or long term	Low	Long	Long term processing of data
Metrology Task	Background	Aperiodic or long term	Low	Long	Long term kalman filters

- **Program Initialization.** This module is run once when the SPHERE is turned on or reset. User code in this module can be used to allocate memory or initialize global data-structures.
- **Metrology.** The two metrology modules are used to capture sensor data and place it in an appropriate space for further processing in lower priority modules. Both modules are high priority to minimize the response time, hence maximizing temporal accuracy of the incoming data. As a consequence, there is only a short time available to perform calculations – typically just enough to store the data and perform some basic processing.

  The inertial sensors (the rate gyros and the accelerometers) can be sampled at up to 1000Hz. Simple integrations or filtering can be performed in this module.

  The global module is triggered when data are received by the ultrasonic sensors; it is triggered once each time a metrology beacon signal is received (up to nine times per global metrology request). Every time the module is triggered its data must be saved, as the current data gets overwritten.
- **Test Initialization.** The initialization code described in the Test Management section above (Section 4.3.2.7) runs in the control thread once each time a test starts. Because the module runs within the control thread it must complete within a short time so as not to overrun the configured control period.
- **Control.** The control thread is a fairly common construct. It executes periodically at a user selectable rate. Standard, discrete control laws can be implemented in this module. Although execution rates of up to 1kHz are possible, most experiments to date operate at 1-20Hz. The controller has a medium level of priority. This gives good real-time performance. Significant calculation can be performed inside the controller, but execution must finish before a control-period elapses.
- **Background task.** The background tasks perform general purpose computation in response to specified system events. During initialization, the user's code selects the particular conditions they want to activate the task. Some of these events are unique to the task. For example, the user may make the task responsive to incoming communication. There is also the option to trigger the task from standard actions such as sensor sampling. Once active, the low-priority nature of the task allows long-term background calculations, without the risk of disturbing time-critical periodic activities.
- **Metrology task.** The metrology task allows scientists to perform long term estimations with a direct link to the metrology data, and without the need to program other types of long term estimation in the same thread. Like with the background task, the metrology task will not disturb time-critical periodic threads.

GSP Simulation

An integral part of the GSP is the non-real-time simulation of the SPHERES testbed. The simulation was introduced as one step in the iterative research process using SPHERES, it is further detailed here. The guest scientist begins the custom software development process by writing source code that adheres to the rules described in the GSP interface document [Hilstad, 2003a]. The guest scientist compiles this source code and links it to precompiled SPHERES simulation objects; the resulting program is a simulation client, which represents a single satellite in the simulation environment. The build process is simplified through the use of a compiler configuration file and a standardized directory structure, enabling a client to be built in a single step.

The complete simulation environment consists of one server program and up to five concurrently operating clients. The server contains a graphical user interface for specifying values for simulation and test parameters, as well as for displaying run-time feedback to the user. Simulation parameters include the dynamics environment, the maximum simulation duration, and the test number. Displayed on the GUI are the power status, maneuver number, propellant usage, and communications usage for each satellite. Errors, warnings and informational messages are printed to the Simulation Messages window. The GUI has buttons that open dialog boxes for specifying additional parameters such as the satellite initial state and the locations of the ultrasound beacons. Each client has a message window and a single button that functions equivalently to the power button on the satellite. The SPHERES simulation server and three client programs are shown in Figure 4.26.

The simulation supports all aspects of single and multi-satellite SPHERES operations, including start-up and initialization, STL and STS communications, and vehicle maneuvering. The simulation code base consists of almost all of the SPHERES core code, supplemented by additional code that simulates dynamics, communications, and hardware-level interaction. The simulation records the true state of each vehicle at 10 Hz, and saves all STL telemetry as it would be recorded by the laptop control station in the laboratory. A

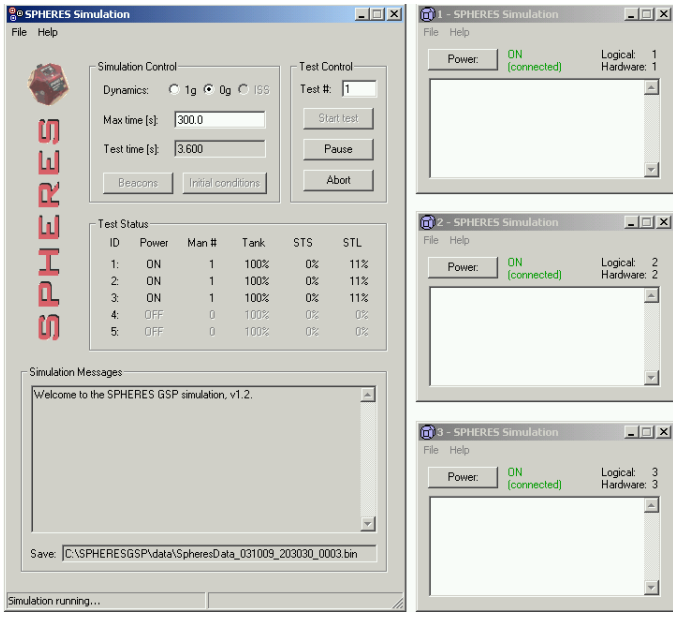


Figure 4.26 GSP simulation window

MATLAB function is provided to read, sort, and plot the data. The simulation is used both to verify syntactic correctness of custom code and to predict the behavior of the hardware in the laboratory and on-board the ISS. Once the simulation has shown that the custom code produces the desired behavior, the code is sent to MIT for verification on the SPHERES hardware in the laboratory.

The simulation guarantees synchronization between the client programs for all timed processes to within one simulated millisecond, the period of the fastest periodic interrupt on the SPHERES hardware. The server enforces synchronization by waiting for all clients to complete each one-millisecond time step before allowing any client to continue to the next time step. This step-by-step process is managed by the server, which sends out step commands and waits for a step completion report from each client. Included in the step command and completion messages are additional data such as state information and communication packets. The clients are multi-threaded, with the main thread handling the user interface and all timed processes, and one child thread running each of five task processes. This multi-threaded implementation allows the use of unmodified SPHERES

source code in the task processes, including functions containing infinite loops, and preserves the free-running nature of the tasks with respect to the timed processes.

Standard Science Libraries

One of the objectives in the design of the GSP interfaces is to minimize the effort that the Guest Scientists must expend on non-productive tasks. For example, if they are interested in developing new estimators, we want to minimize the effort spent on getting the control-system to operate satisfactorily. To this end, we have developed a number of specific function libraries to help accelerate the development process.

The SPHERES core software creates the essential framework to support multiple scientists in the development and maturation of new algorithms. Figure 4.25 on page 170 illustrates the framework created by the SCS on the three left panels; the right-most panel, supplemental libraries, presents an enhancement to this framework which further simplifies the use of SPHERES by multiple scientists. These supplemental libraries are not required by the general SCS framework; they are not operationally required elements. Yet, they transform the SCS API into more than a framework, they create a software platform for the development of DSS algorithms. Through the standard science libraries the SPHERES core software becomes a fully functional facility with basic estimation and control. Individual scientists then take the base SCS environment and create derivative algorithms based on their individual needs.

The standard libraries are optional complementary functions to the SCS. Scientists can select to use the provided functions, provide their own developed independently, or use the standard libraries as a starting point for custom functions. These libraries help provide scientists with guidelines on the development of their own algorithm, but by being optional and independent of the SCS, do not constraint the scientists in any manner.

Figure 4.25 groups the standard science libraries into their major elements:

- **Math**. The library of math functions was developed to ensure compatibility of complex mathematical functions with the C6701 DSP. These functions include standard matrix manipulation routines, inversion methods, and LTI filters commonly used in control and estimation algorithms.
- **Control.** This library includes a number of 1DOF, 3DOF, and 6DOF proportional (integral and derivative) closed-loop controllers. A non-linear switchline controller is also available. These controllers have not been optimized for any specific condition; rather, they have been designed to guarantee stable operations. In this manner scientists who concentrate on other topics, such as estimation or autonomy, need not worry about the development of controllers.
- **Estimation.** The estimation libraries include several different Kalman filter routines. These estimators use both the inertial information and the global metrology sub-system to determine the full state of a satellite; they also include estimators to calculate differential states using the on-board beacons. The standard SCS estimator, which operate in the SPHERES Metrology Task (Figure 4.25), is part of this library. The library also includes other estimators under development at the MIT SSL.
- **Maneuvers.** A range of individual maneuvers, such as single-axis translation or rotation are available in this library. These maneuvers can be combined with standard or custom control and estimation functions to complete a test. The library also includes a set of *terminators*, functions which test when a maneuver and/or test has completed and indicates the fact to the higher level SCS components.
- **Mixers.** The SPHERES GSP uses a broad range of knowledge on the satellites' physical characteristics to provide scientists with accurate mixers which translate a force/torque command into thruster on-off times. These mixers take into account the mass properties, the thruster locations, and the thruster IDs.
- **Utilities.** The standard science libraries provide several utilities not directly related with algorithm development, but which support their development. These include data compression functions for post-test analysis and communications debugging routines for ground-based tests.

Although the libraries are designed specifically to operate in the SPHERES environment, these routines do not issue commands directly to the hardware interfaces. Instead, they perform the requested calculations and prepare a command. Since the user must issue the thruster command there is never confusion or contention about where the command originated, and the scientist always has access to that information.

> *The Guest Scientist Program is an integral part of SPHERES which combines operational and software features to support multiple scientists. It provides a simulation for inhouse software development. A flexible yet robust software environment creates the execution framework for the satellites. A set of optional standard science libraries creates a software platform upon which scientists can develop their own derivative algorithms.*

4.3.3.2 Expansion port

The SPHERES team realized that custom software had realistic limitations in the ability to mature science completely. Maturation with respect to TRL's requires the demonstration of algorithms in representative environments, and the SPHERES hardware could only represent general spacecraft. To ensure that SPHERES provides the opportunity to mature algorithms through higher TRL levels, SPHERES provides for the expandability of hardware components so that the generic SPHERES satellites can be customized with mission-specific science-type payloads.

Each SPHERES satellite has two flat panels on opposite sides that can be used to expand the hardware payload. One side provides a passive mechanical attachment point, where expansion items that do not need any connections to the satellite electronics can be attached. For example, this panel can be replaced by a passive "docking pin."

The other side provides both mechanical and electronic connection points. This side, called the SPHERES Expansion Port (Figure 4.27), interfaces to the main electronics stack via both serial and parallel lines, and provides power for external components. Expansion items can interface to the main processor, allowing all algorithms to reside within the main SPHERES software. The Expansion Port can be used for items such as an active docking mechanism with sensors and actuators.

The design of the expansion port contemplates two needs: easy of integration of simple payloads and the capability to support complex payloads. The port provides three output voltages (+5V, +15V, and -15V) to support standard electronics as well as analog components. Simple payloads are supported via a standard UART serial line (up to 1.25Mbps).

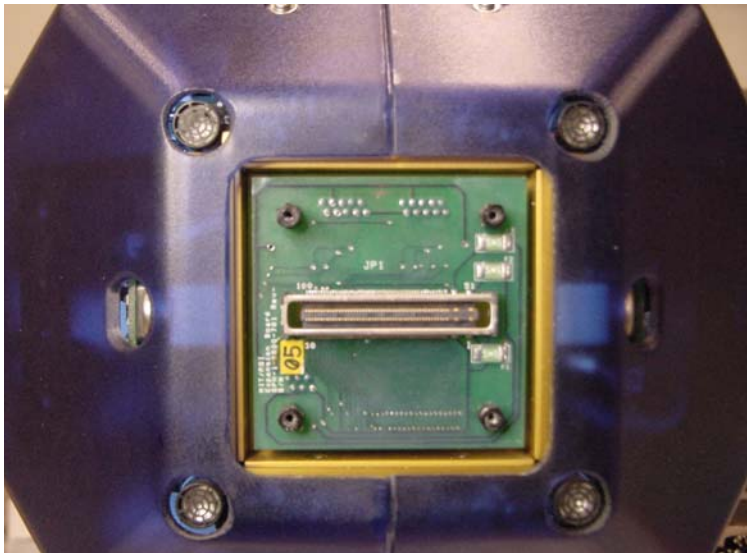


Figure 4.27 SPHERES satellite expansion port face (without cover)

Complex payload communicate with the DSP directly over the processors global data bus, a 2GB 32-bit memory space. Three analog input lines are available directly on the expansion port connector. The expansion board also includes hardware to allow the substitution of the global metrology sensors in that satellite face with new sensors in the expansion item, to account for the case when the expansion item covers the sensors and the global metrology system must be used. A schematic overview of the expansion port is presented in Figure 4.28.

> *The SPHERES expansion port allows hardware expandability for new science payloads. The port provides simple interfaces for quick integration and high capacity memory interfaces for complex payloads.*

4.3.3.3 Portability

A side benefit of the requirement to design the satellites such that they fit within one MLE was the easy of portability of the hardware. Flight-identical hardware can be transported without special considerations. All the necessary hardware for full operations in ground-based facilities can be transported using two to six hard-shell transport cases (depending on the number of satellites to be used), with mass ranging from 30kg to 150kg. Demon-

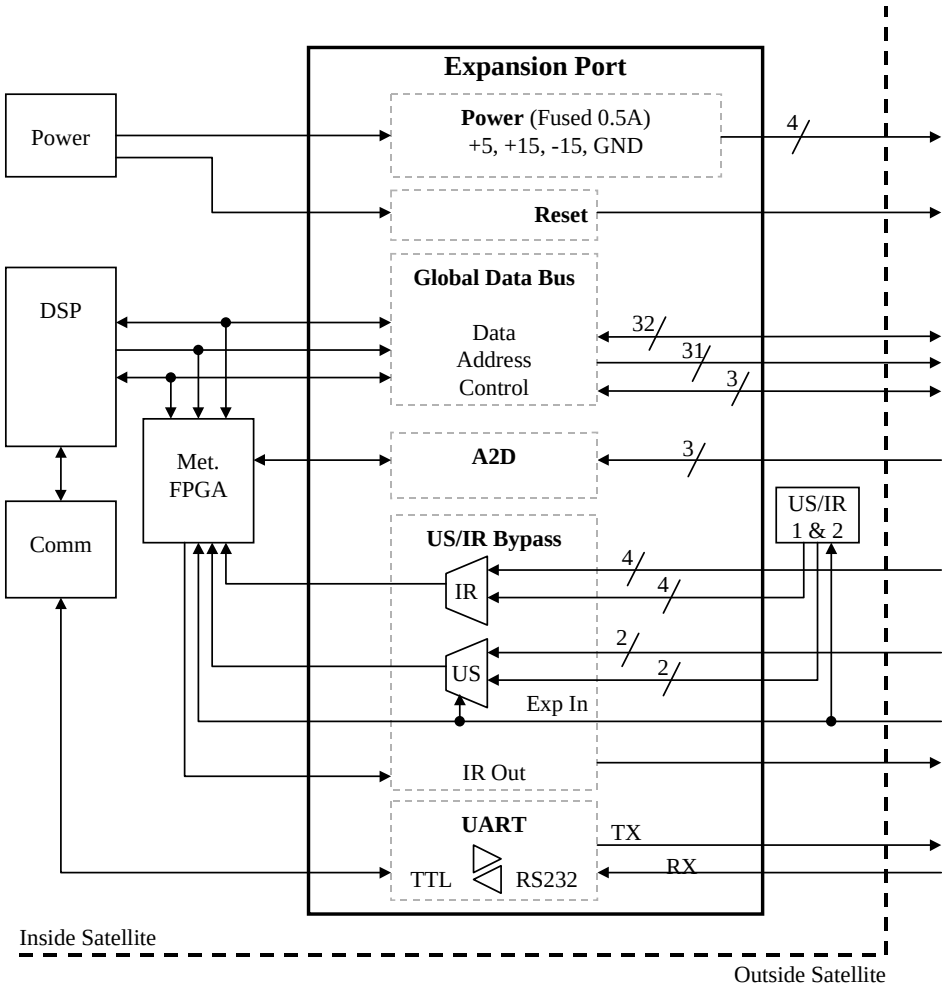


Figure 4.28 SPHERES expansion port design overview

stration of algorithms which do not require science iterations can be performed with only the satellite(s), batteries, tanks, laptop, and a communications box; these can fit in a single hard-shell box at around 20kg.

While not necessarily viable for all types of space maturation experiments, this portability helps SPHERES support multiple scientists by allowing operations in the necessary environments to advance their science. Portability does not necessarily simplify the involvement of multiple scientists directly, rather, it opens the operational environments of SPHERES to support a wider range of environment that become representative of those needed to mature DSS algorithms. The portability opens the operational environments to

locations beyond the MIT SSL and the ISS to other facilities presented in Chapter 1. For example, the hardware can be easily transported to the NASA Reduced Gravity Office for test in the reduced gravity airplane. These tests allow microgravity experiments in a ground-based facility, providing scientists with data beyond that capable at the MIT SSL. The hardware can also be transported to flat floor facilities, when scientists require larger operational areas than those allowed at the MIT SSL or even the ISS. Lastly, the hardware could be sent in a temporary basis to the locations of the scientists themselves.

> *The portability of SPHERES allows the facility to operate in a wide range of locations to better resemble the representative environments required for technology maturation of the different DSS science fields.*

4.3.3.4 Schedule flexibility

From its conception the SPHERES operational plans called for flexibility in the scheduling of operating sessions in the ISS. The baseline plan of one operating session every two weeks drives the frequency of total operations, but does not necessarily constraint scientists to follow that timeline strictly. Instead, the program calls for the MIT SSL to manage the schedule among participating scientists to make full use of each operating session but also to allow scientists to set their own schedule as necessary. The schedule allows the intercalation of scientists so that each session can concentrate on a limited number of science goals (simplifying the work of operators) and allow each group of scientists enough time to review their data between their sessions. At the same time, if scientists only require a small amount of operational time and prefer quick turn-around of tests, the schedule (and core software) allows for multiple types of science to be conducted in the same session every two weeks.

> *Both SPHERES and guests scientists make use of schedule flexibility by ensuring that ISS operating sessions are used in full and that scientists conduct operations as frequently as they need without strict limitations beyond the minimum two week cycle.*

4.3.4 Reconfiguration and Modularity

Modularity formed an integral part of the SPHERES design from its initial stages. The prototype development teams were divided into teams which designed individual sub-systems in a modular fashion: each sub-system minimized its dependence on the others for operations. The SPHERES satellite design is modular. The design of the individual sub-systems can be (and has been) easily integrated into other project which use different configurations due to their simple interfaces and operational independence. Still, once the flight hardware design was finalized and the satellites were assembled, this modularity is no longer visible to the scientists which operate the facility.

The modularity of SPHERES which matters to the scientific community is that which enables wide flexibility in the use of the facility. Through system-wide features and specific sub-system design choices, the SPHERES facilities can be changed to better reflect the science needs of individual scientists. The facility allows for reconfiguration of both software and hardware, as well as flexibility in the use of one or more satellites to create representative environments.

The primary characteristics of the SPHERES facility which enable reconfiguration and modularity are:

- Generic satellite bus
- Science specific equipment: on-board beacon and docking face
- Generic Operating System
- Physical Simulation of Space Environment
  - Operation with three units
  - Operation in 6DOF
  - Two communications channels
- Software interface to sensors and actuators
- Hardware expansion capabilities
- FLASH memory and bootloader

4.3.4.1 Satellite bus

The SPHERES satellites provide generic equipment for space technology maturation experiments by implementing a general spacecraft bus for use by scientists. The primary functions of a spacecraft bus are to support the payload, provide maintenance of orbit and pointing of the payload correctly, and provide power, communications, and data storage. To accomplish these goals, spacecraft payloads utilize the following main sub-systems: propulsion, attitude determination and control, communications, command and data handling, thermal, power, and structures sub-systems [Larson, 1992]. The SPHERES satellites provide each of these sub-systems (except thermal, which is not required in the ISS) and allow the scientist to utilize them in their science as needed. By including all parts of a generic satellite, the SPHERES satellites provide scientists with a true physical representation of an operational spacecraft. Developing a full satellite bus fulfills the need for a physical end-to-end simulation of a spacecraft with realistic physical responses and interactions between sub-systems.

The basic SPHERES satellites enable scientists to mature DSS algorithms for coarse control of systems; i.e., the default configuration, without science-specific expansion items, allows scientists to test algorithms that would perform general maneuvers to initiate and maintain formations, docking tasks, or similar. High precision control can be tested in the future by the addition of science-specific payloads. The SPHERES basic satellite bus configuration provides generic space sub-systems (Table 4.10).

The position and attitude determination and control sub-systems (propulsion and metrology) provide basic actuation and sensors similar to those found on current spacecraft. Actuation is provided by on-off thrusters, providing similar response curves to standard space thrusters. Precision actuators are not provided in the basic satellites: reaction wheels, active optical elements, and other actuators can be added via the expansion port. The metrology system resembles a GPS system, in a local fashion. It provides state information to sub-centimeter precision. This precision is valid for coarse control of spacecraft,

TABLE 4.10 SPHERES implementation of a spacecraft bus

[Larson, 1992]	SPHERES	Generic	Specific
Propulsion	Propulsion	✓	
Attitude Determination and Control	Propulsion and Metrology	✓	
Communications	Communications	✓	
Command and Data Handling	SPHERES Core Software		✓
Thermal	*n/a*		
Power	Power	✓	
Structures	Structures	✓	

but higher precision sensors will need to be added to demonstrate technologies for optical imaging via separated spacecraft.

The communications sub-system selection was based on the need to provide wireless communications, but not driven by the requirements of specific mission. The selection simplified the integration into the ISS. The implemented protocol answers to the behavior of the selected hardware, rather to a specified protocol for DSS. The system allows the protocol to change between satellites, such that the only true constraints are the half-duplex nature of the wireless system (which affects all wireless systems, including existing space communications) and its determined maximum data rate.

The power and structures sub-systems simply ensure the functionality of the satellites. Their design does not answer to any mission specific requirements, but rather to the general need to ensure operations in the ISS and other facilities over extended periods of time.

The command and data handling sub-system (the SPHERES core software) is potentially the only non-generic system; its does not necessarily mimic space systems entirely. Its design was driven directly by the objective to mature test control, estimation, and autonomy algorithms; therefore, rather than simply being a command-handling engine, it also provides routines to specifically meet that objective. At some level it was required that

part of the SPHERES sub-systems specialize in meeting the mission requirements; the software sub-system deviates from the generic nature of the other sub-systems to fulfill these requirements.

> *Each SPHERES satellites is a physical end-to-end simulation of a spacecraft bus. The individual sub-systems are generic in nature, except for the software sub-system which is specialized to meet the mission objectives.*

4.3.4.2 Science specific equipment: on-board beacon and docking face

The initial deployment of SPHERES was driven directly by two specific DSS fields: formation flight and rendezvous/docking. The generic satellite bus provides the necessary tools for formation flight tests; rendezvous/docking algorithms required the addition of science-specific equipment to truly meet the requirements for a physical simulation of the intended systems. To better model docking applications, the SPHERES satellites include two elements specifically designed to enable testing docking algorithms:

- **On-board beacon**. The on-board beacon is a replica of a global metrology transmitter box placed internally on the -X face of the satellites. The design is almost identical to the external beacons, except that it uses the satellite's exiting power sources and infrared receivers, to avoid redundancy in electronics. Otherwise, the on-board beacon includes its own microcontroller and ultrasonic driving circuity, and behaves identically to the external beacons. This beacon interfaces with the satellite avionics so that it can be enabled during tests that require its use and otherwise disabled to minimize power consumption when it is not needed. The internal avionics can configure the beacon number so that it can be used as a stand-alone global system (to determine differential states between satellites, see Section 4.3.2.1), or as part of the larger global reference.
- **Docking face.** Also place on the -X face of the satellites, the docking face is a simple docking mechanism so that satellites remain joined after a docking maneuvers, rather than produce an elastic collision and separate after impact. The docking face consists of a special pattern of velcro strips; the pattern maximizes the amount of angle (roll between units) error allowable so that capture still occurs. The velcro is located around the on-board beacon so that two units can approach each other with direct measurements between them during the docking maneuver.

The -X face "docking face" of the satellites is pictured in Figure 4.29. The on-board beacon ultrasound transmitter is visible in the center of the face. The velcro pattern is shown around the ultrasound transmitter.

> *The SPHERES "Docking Face" provides an example of specific equipment developed to satisfy a specific mission objective: demonstration of docking and rendezvous algorithms.*

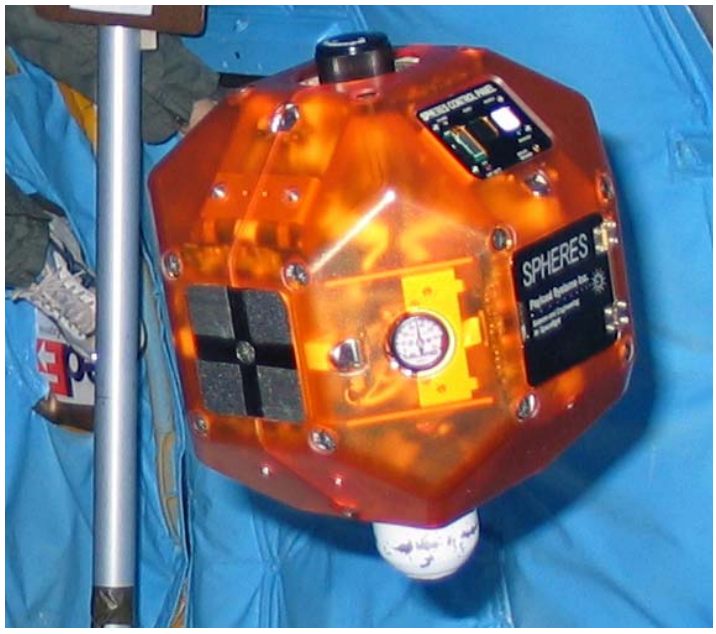


Figure 4.29 SPHERES -X "docking face"

4.3.4.3 Generic Operating System

The software of the satellites must allow multiple researchers to use the general bus provided by the hardware and to interface with any specific equipment added by the scientists. To this purpose, the testbed's software design was almost entirely driven by the need to accommodate multiple researchers. The goal in the software design was to create an architecture flexible enough to accommodate a wide variety of the sophisticated applications within the main areas of DSS. The resulting SPHERES Core Software (described in Section 4.3.3.1) creates a generic operating system for the SPHERES program.

SCS grows upon a real-time operating system (DSP/BIOS) to create a structured framework to develop a wide range of programs using a standard programming language. The development of the SCS in standard ANSI C, with support for C++, generalizes the nature of the operating system. Its use of a generic programming language ensures that a wide range of scientists can develop their algorithms for use on SPHERES. While a custom API was created to the SCS, all of the interfaces are fully complaint with the language standards. DSP/BIOS features generic tools of any RTOS, such as hardware and software interrupt management, pipes, mailboxes, and semaphores. Although scientists do not need to interface with those tools directly, the resulting SCS is based directly on these generic tools; further, scientists can access these tools if necessary.

> *Rather than implementing a generic version of a spacecraft command and data handling program, the SCS implements a generic real-time operating system framework for algorithm development.*

4.3.4.4 Physical Simulation of Space Environment

SPHERES simulates the expected operational environments of formation flight, docking, and other DSS missions closely. To meet the feature of physical-end-to-end simulation SPHERES operates with three satellites in a 6DOF environment using two separate communications channels.

Operation with three units

An important part in the original design process of SPHERES was the determination of the number of units to operate with. Because the primary science goals of SPHERES at the time considered formation flight and docking algorithms, it was clear that an absolute minimum was two units. Two units allows full demonstration of docking algorithms. Two units also allows demonstration of multiple formation flight algorithms, including initial development of any type of algorithms. But the use of two units did not truly meet the feature of a physical end-to-end simulation with realistic simulation of the expected opera-

tional environment for formation flight missions. Intended missions at the time (e.g., TPF and Orbital Express) utilized more than two units in all of their expected operational environments. For example, the use of two units does not simulate the results of two followers maintaining formation with a leader spacecraft, but independently of each other. This simple example results in the requirement to operate three units to fulfill the need for a physical end-to-end simulation.

The use of three units increases the trust on the formation flight demonstrations performed with SPHERES. First, formations can be defined in terms of planes rather than lines; maintaining the plane is essential for imaging applications, and two units could not demonstrate that capability with confidence. Further, the use of three units allows the demonstration of how different architectures [Saenz-Otero, 2000] compare with each other under realistic operations. Leader/follower architectures can operate with multiple followers and show their advantages over master/slave architectures where the slaves are completely blind from each other; peer-to-peer architectures can demonstrate failures in one unit and recovery by the other units. Three units can potentially demonstrate the capabilities of hierarchical structures by defining each of the units as one level under the other. While three units do not model all formation flight missions identically, the use of three units captures the most important physical characteristics which must be demonstrated to mature the algorithms.

Operations in 6DOF

Even in the theoretically ideal case where a physical system has a diagonal inertia matrix and all sensors and actuators are de-coupled along each major axis, expanding the dynamic equations from 1DOF to 3DOF and then to 6DOF is not a trivial process. Adding rotational degrees of freedom adds substantial complexity to all dynamics equations; moving from a 3DOF to a 6DOF system adds two rotational degrees of freedom. The physically realistic scenarios of a non-diagonal inertia matrix further complicates the expansion of problems to 6DOF.

Therefore, to demonstrate algorithms for spacecraft the environment should allow natural asymptotic dynamics to emerge and system dynamics to develop in 6DOF. In order to properly model the system, the full complement of six degrees of freedom are required. In this manner the environment allows traceability and modeling of formation flying maneuvers, especially large out-of-plane coordinated movements.

Two communications channels

The primary driver in the selection of two independent communications channels was to simulate the communications methods of separated spacecraft systems as close as possible. Each of the channels simulates the two types of expected communications present in DSS operations: satellite-to-ground (STG) and satellite-to-satellite (STS). Actual systems will use different systems for each type of communications. STG channels are expected to be high-power, high latency (long distances) systems which download science data to ground after the satellites capture and process information (the STG channels are not necessarily low-bandwidth, since the throughput can be high, but the latency is prohibitive for controls). STS channels are expected to be low-power, low latency, high bandwidth (short distances) systems which transfer data between the satellites necessary to maintain precision formations or perform autonomous docking. SPHERES implements two channels which are operationally identical; their only difference is in the actual RF frequency (868.5MHz vs. 916.5MHz).

The implementation with identical channels helps SPHERES fulfill other features of the MIT SSL Laboratory Design Philosophy, but does not hinder its ability to provide a physical end-to-end simulation. Because the hardware of the two channels is operationally identical and the frequency choice can be easily swapped in software, the only physical limitation of the implemented system is in the available data transfer rate of up to 16kbps for a single unit using the implemented TDMA protocol. Otherwise, the communications channels can be used by scientists with software filters to simulate the different types of communications. For example, if a scientist wishes to simulate a system where only the

master satellite has an STG channel but slave units do not, the software in the simulated slave units can be programmed to ignore all STG communications. Similarly, software filter can implement delays in the STG channel, or limit the throughput of either.

> *SPHERES creates a realistic physical end-to-end simulation of expected formation flight missions by operating with three satellites in a 6DOF environment. The two independent communications channels add further realism to the simulated operations.*

4.3.4.5 Software interface to sensors and actuators

Section 4.3.3.1 described how the SPHERES Core Software mediates interactions between the scientist user code and the DSP/BIOS and hardware. This layer not only simplifies the interfaces to the hardware, it also allows the creation of custom interfaces to the sensors and actuators. Scientists can create a third layer of interfaces to the sensors and actuators which better model their intended operational environments. Specifically, scientists can create filters or special models to interact with the propulsion and metrology subsystems.

The default core software implements standard pulse width modulation actuation via the thrusters. The thrusters are commanded on-off periods of actuation; the basic software immediately implements the commands. Scientists can create functions which first center the pulses on specific frequencies, or they could implement frequency modulation actuation, rather than pulse width. These filters could also add delays in the actuation, model saturation levels, and help simulate analog actuators with slow frequency responses by using the minimum impulse bit available with the SPHERES hardware.

The inertial and global metrology hardware provide the data accuracy, precision, and observatbility called for in the SPHERES requirements. But this data does not necessarily match the expected metrology information of specific missions. The system requires flexibility to allow scientists to use the data as appropriate for their research. This flexibility comes from the software implementation. First, the metrology system allows scientists to

directly specify the data capture rates of the inertial and global systems independently; the software allows frequency ranges from under 1Hz to up to 1kHz (for the inertial system). Without any special code the SCS allows scientists to model the frequency responses of their sensors. Second, a layer can be created between the standard SPHERES estimator and the scientists use of the states. These modules can simulate sensors not directly available in the SPHERES hardware, such as a star tracker, by modeling the sensor and providing a second state which is used by the scientist's algorithms. In this way scientists can present their algorithms only with the expected available state information, and use the full state calculated by the default estimator as a truth measure to their sensor models.

The limitations of these models lie within the specifications of the SPHERES hardware; the software does not limit the models under the capabilities of the hardware. The propulsion hardware is limited to a frequency of 50Hz; therefore all models will have that maximum frequency. The minimum thruster on-time of 10ms limits the minimum impulse time. Similarly, the maximum sampling rate for the inertial sensors is 1kHz; the maximum rate for the global metrology system is 5Hz. The SCS interfaces with the propulsion system at 1kHz, easily allowing 50Hz operations. The interface with the metrology sensors, both inertial and global, also operates at 1kHz, ensuring that the maximum sampling of the inertial sensors can take place and creating no barriers to access the global system.

> *To better create a physical end-to-end simulation of their system, scientists can create software models of their sensors and actuators which are only limited by the hardware capabilities of the SPHERES satellites but not by the software.*

4.3.4.6 Hardware expansion capabilities

The SPHERES expansion port, presented in Section 4.3.3.2, and the "docking face" directly enable hardware reconfiguration. The primary objective of the expansion port is to support multiple scientists by allowing the addition of specific scientific hardware. The primary objective of the docking face is to enable the demonstration of docking algo-

rithms. But both of these fulfill a second objective: they allow easy manipulation of the hardware to demonstrate increasing complexity of the geometry and/or components.

The expansion port and docking face allow the addition of both passive and active elements with easy. Passive elements can be attached to the docking port by using Velcro in the correct configuration on the additional hardware. This allows the dynamics of the system to change immediately by the addition of different masses. The expansion port allows active elements, be it sensors or actuators, to modify the dynamic behavior of the satellites.

The ability to modify the hardware with active elements depends on the ability of the software to identify those new active components and make use of them. The SCS provides the necessary interfaces so that scientists can access all of the signals available in the expansion port with ease. The SCS always remains as a necessary layer between the hardware and the software. Access to the global bus requires initialization by the SCS; the expansion port global bus data must be accessed through special SCS routines. The SCS also initializes and provides the interfaces for the serial data line of the expansion port. The analog inputs are read automatically by the SCS and made available to scientists via the metrology routines.

> *Passive and active elements can be added via two different locations to implement hardware reconfiguration which increasingly adds complexity to the geometry and dynamics of the satellites.*

4.3.4.7 FLASH memory and bootloader

Previous MIT SSL experiments implemented software reconfiguration [Miller, 1996]; that reconfiguration included the ability to change the state-space matrices of controllers and in some cases the controllers themselves. SPHERES was challenged with allowing high levels of software reconfiguration. The wide range of fields that comprise DSS required that the software reconfiguration not be limited to a specific section of the software, but

rather to a number of major sections. Therefore the SPHERES design implemented a custom bootloader which allows to fully reconfigure the software. As introduced in Section 4.3.2.11, the bootloader allows the operations software to be de-coupled from the hardware implementation. The current implementation of the SCS is not permanently fixed; the SCS can evolve over time, and even different frameworks could be created in parallel to the SCS.

The decision to allow to fully reconfigure the software trades between operational overhead time and flexibility. The decision presents some drawbacks during the development stages of the algorithms. The need to program the software in its entirety adds overhead time to the development process, since even small errors in the code will require to load the full program every time. Yet, the operational plan of SPHERES indicates that during initial development, when operations occur via the simulation or at the MIT SSL, the time to reload a program is not significant. On the other hand, the ability to fully reprogram the satellites is the only way to ensure that the many areas of DSS can be studied over the long term. This ability will enable SPHERES to be used in areas of DSS not currently accounted for by allowing the creation of new threads and interfaces.

To enable full software reconfiguration, the avionics required non-volatile memory which can be overwritten electronically. The selected DSP hardware contains 512kB of FLASH memory onboard. Of that space 256kB are reserved for the board configuration and 34kB for the SPHERES bootloader. Therefore, each satellite provides up to 54k words (216kB) of FLASH memory space for programs; the SCS takes approximately 22k words (88kB), leaving 32k words (128kB) to the scientists. The FLASH memory map is presented in Figure 4.30.

Booting a DSP is a multiple step process. All DSP’s have their own boot program for internal configuration; this boot program is created by Texas Instruments and resides permanently in the DSP chip itself. This process completes in micro seconds. A second boot program configures the SMT375 peripherals so that it can communicate via its TIM 40

01400000 – 0140FFFF	16 kB	Sundance boot loader
01410000 – 0150FFFF	256 kB	FPGA configuration data
01510000 – 0151FFFF	16 kB	FLASH Loader
01520000 – 015FFFFF	224 kB	Application Space
	- 88 kB	- SPHERES Core Services
	- 128 kB	- Scientist code and optional data storage

Figure 4.30 FLASH memory map

standard communication ports, enables the global bus interface, and initializes the interfaces to the internal features of the SMT375. This boot program was custom made for the SPHERES program to minimize the complexity to interact between the SMT375 and the SPHERES peripherals. The SMT375 boot process completes within a few milli seconds.

The SPHERES bootloader is the third boot process. A explained in Section 4.3.2.11, the bootloader is the only mission-critical software in the SPHERES program. Its operation is essential to the success of the mission. This program is loaded by the SMT375 after it is configured. The bootloader first configures the metrology FPGA so that it can communicate with the control panel and all other digital I/O lines. Second, the bootloader configures the three communications micro-controllers to operate at the default data rate of 115.2kbps. Third, the wireless communication channels are set to their default configuration, to ensure that they are functional with the bootloader regardless of any configuration changes by the SCS or other programs (this step takes approximately 2 seconds). These steps leave the satellite in a valid configuration ready for operations.

Next, the bootloader checks the three used communications ports (wireless 868.5MHz and 916.5MHz, and the expansion port serial line) for data commands to initialize the bootloading process as well as the state of the enable button in the SPHERES control panel to determine user override. If data is available or the user forces entry into bootloader mode, it begins to load a new program. The bootloader uses a custom communications protocol with large data packets to minimize overhead; all packet have a two byte checksum. The packets are confirmed at fixed intervals; if a packet is not confirmed the packets are sent again. After loading all the packets, the bootloader calculates a 32bit program checksum

to confirm program integrity. Once a valid program has been loaded a special register in the FLASH memory is enabled, and the boot loader proceeds to load the program.

When no data is available in the communications port the bootloader checks a special register; if the register indicates that no valid program is present it automatically enters into bootloader mode and indicates a "no program" condition in the control panel.

When a valid program is present and the bootloader has no other pending actions, it loads the program into memory. Loading the program takes a few milli seconds. If the program is a standard SCS application, the program first configures the SPHERES peripherals for use with the SCS standard interfaces, and then runs the SAT INIT process and enters the idle mode described in Section 4.3.1.4. Table 4.11 summarizes the full SPHERES boot process.

A custom bootloader allows the full software of a SPHERES satellite to be reprogrammed and stored in FLASH. The bootloader automatically starts an existing program if no command is received to load a new program.

TABLE 4.11 SPHERES bootloading process

Step	Process	Time	Enables
1	C6701 Boot Process	μs	C6701 core, memory interfaces, and embedded peripherals
2	SMT375 FPGA/ DSP configuration	ms	SMT375 communications ports, global bus interface, LED's, DSP/FLASH interface
3	SPHERES Bootloader	2s	SPHERES FPGA (metrology, propulsion, internal beacon, housekeeping, and control panel I/ O's), DR200x wireless communications
4	SCS Sat Init	ms	API to SPHERES peripherals, TDMA wireless communications, metrology configuration, background telemetry, DSP/BIOS real-time environment, satellite logical identity

4.4 Summary

Table 4.12 summarizes the characteristics of SPHERES which enable it to fulfill the MIT SSL Laboratory Design Philosophy. A thorough operations plan and carefully designed software and avionics (enabling families of tests, easy repetitions, separation from safety controls, and quick data feedback) *facilitates the iterative research process.* The visual nature of SPHERES further helps to speed up iterations.

The design of the nano-satellite hardware *supports experiments*, satisfying all but one features called upon by the philosophy. The metrology and communications systems enhance *data collection*. The 32-bit DSP ensures *data precision* throughout all data processing. The test management plan and location specific GUI's facilitate *repetitions*. The re-supply of consumables provides system *reliability,* enables *extended investigations,* and creates a *risk-tolerant environment.* The use of more units than essentially necessary and the fact that software cannot cause a critical failure also create a *risk-tolerant environment*.

The Guest Scientist Program, through its logistics, the SPHERES Core Software, simulation, and standard science libraries, together with the flexible schedule of SPHERES, directly *supports multiple investigators.* The Expansion port further enhances the ability to *support multiple investigators* by allows investigator-specific hardware to be used in experiments. The portability of SPHERES increases the number of operational locations for the facility, such that *multiple investigators* can use the hardware in the preferred locations for their specific science.

The implementation of the SPHERES nano-satellites as a standard satellite bus provides a perfect example of the development of *generic equipment*, while at the same time creating a *physical end-to-end simulation* of a spacecraft. At the same time the implementation of docking-specific equipment, the SPHERES hardware also demonstrates the use of *specific equipment*. The physical nature of the SPHERES satellites, with their ability to fully simulate complex DSS missions, creates a realistic *physical end-to-end simulation* of expected missions. The generic operating system and software interface to the SPHERES sensors

TABLE 4.12 Summary

SPHERES Characteristic	Sub-System					Feature Group			
	Avionics	Communications	Software	Operations	System	Iterative Research	Support of Experiments	Multiple Investigators	Reconfig. & Modularity
Multi-layered operations plan				✓	✓	✓		✓	
Continuous visual feedback					✓	✓			
Families of tests			✓			✓			
Easy repetition of tests			✓			✓	✓		
Direct link to ISS data transfer system		✓				✓	✓		
De-coupling of SW from NASA safety controls	✓		✓			✓		✓	
Layered Metrology System	✓		✓				✓	✓	
Flexible communications		✓	✓				✓	✓	
Full data storage		✓					✓		
32-bit floating point DSP	✓						✓		
Redundant communications channels		✓					✓		
Test management and synchronization			✓	✓			✓		
Location specific GUI's				✓			✓	✓	
Re-supply of consumables	✓				✓	✓	✓		
Operations with three satellites					✓		✓		
Software cannot cause a critical failure	✓		✓		✓		✓		
Guest Scientist Program				✓				✓	
Expansion Port	✓							✓	
Portability					✓			✓	
Schedule flexibility				✓				✓	
Implementation of a satellite bus	✓	✓	✓		✓				✓
Science specific equipment	✓				✓			✓	✓
Generic operating system			✓						✓
Physical simulation of space environment	✓	✓			✓				✓
Software interface to sensors and actuators	✓		✓			✓		✓	✓
Hardware expansion capabilities	✓								✓
FLASH memory and bootloader	✓				✓	✓			✓

and actuators provide a *modular* software platform which provides *generic* command and data handling functions while allowing *software reconfiguration* to meet the *specific* needs of scientists. Lastly, SPHERES enables both *hardware* and *software reconfiguration* through its expansion port, use of FLASH memory, and the development of a boot-loader which works independently of the SPHERES Core Software applications.

By meeting practically all the features of the MIT SSL Laboratory Design Philosophy and operating making the correct use of the resources of multiple facilities (SSL Lab, KC-135, and ISS), SPHERES is more than a testbed for formation flight, it is a laboratory for DSS. Recall the definition of a laboratory (page 60): *a place providing opportunity for experimentation, observation, or practice in a field of study.* SPHERES does provide the opportunity for experimentation, as it facilitates the iterative research process. Further, SPHERES supports the research of multiple scientists whom can work on different areas of DSS, enabling the practice in a field of study.

Lessons were learned from following the MIT SSL Laboratory Design Philosophy in the development of the SPHERES laboratory. These lessons are presented in the following chapter as the Design Principles for the Development of Space Technology Maturation Laboratories.

Chapter 5

MICROGRAVITY LABORATORY DESIGN PRINCIPLES

Through more than two decades the MIT Space Systems Laboratory has developed a number of successful microgravity experiments for the maturation of space technologies. Throughout the design and operation of these experiments researchers at the MIT SSL have learned a number of important lessons; initially those lessons were expressed as the MIT SSL Laboratory Design Philosophy, presented in Chapter 3. The development of the SPHERES laboratory for distributed satellite systems, presented in Chapter 4, implemented all the lessons learned from the past experiments, and led to the creation of a new philosophy which combines the original MIT SSL Laboratory Design Philosophy and the use of the International Space Station (Chapter 2). This new design philosophy condenses the lessons learned from all the previous chapters.

The intent of the principles presented in this chapter is to give both designers and evaluators of microgravity experiments for technology maturation a clear idea of what qualities a specific project must meet, rather than a long list of individual specific items. By generalizing the concepts, the principles encompass a wider range of technology maturation experiments, beyond the dynamics and control scope of the MIT SSL. The principles capture the most important concepts of the MIT SSL Laboratory Design Philosophy. The features of the philosophy lie within the principles as lower level methods to implement the principles. The principles also capture the lessons learned from the literature review about the ISS and the operations of MACE-II aboard the ISS. As presented in Chapter 2, the

principles deal directly with iterative experiments for space technology maturation; while other types of iterative research (such as pure science) could benefit from the principles, the principles do not account for all aspects involved in the other types of research.

In order to define a set of principles, the concept of a principle must be clearly understood and defined first. The following definitions of *principle* guided the development of the ones presented in this thesis:

> *[Merriam-Webster, URL]*
>
> Main Entry: prin·ci·ple
>
> 1 a: a comprehensive and fundamental law, doctrine, or assumption b (1): a rule or code of conduct (2): habitual devotion to right principles <a man of principle> c: the laws or facts of nature underlying the working of an artificial device
>
> *[Crawley, 2003]*
>
> Principles are the underlying and long enduring fundamentals that are always (or almost always) valid.

Therefore, the objective of the principles is to address those *fundamental design issues that should hold true for all well-designed microgravity laboratories for space technology maturation operated aboard the ISS*.

The first three chapters provide the basis to understand the concepts that comprise the objective of the principles. These concepts are: *microgravity research, laboratory, space technology maturation,* and *ISS*. The concept of space technology maturation is explained in Chapter 1, which introduces the Technology Readiness Levels as an example of current evaluation methods to demonstrate space technology maturation. The chapter also discusses several microgravity and remote research facilities; Chapter 2 uses the literature research of the introduction and further research on the International Space Station to better identify the special resources of the ISS and the research conducted within. Chapter 3 introduces the dictionary (Merriam-Webster) definition of a laboratory, and specifies that this thesis concentrates on the need for a laboratory to support experimentation in a field of study. Chapter 3 also introduces the definition of a facility, stating that a facility must

make a course of conduct easier and is established for a specific purpose. Therefore, it is possible to expand further on the objective of these principles: they guide towards the *development of a laboratory environment, supported by facilities,* to allow *multiple scientists* the conduct of research under *microgravity conditions, correctly utilizing the resources provided by the ISS,* such that they *cover a field of study* to *accomplish technology maturation.*

The following are the Microgravity Laboratory Design Principles presented in this chapter:

- Principle of Iterative Research
- Principle of Enabling a Field of Study
- Principle of Optimized Utilization
- Principle of Focused Modularity
- Principle of Remote Operation & Usability
- Principle of Incremental Technology Maturation
- Principle of Requirements Balance

The principles were derived by David Miller, Javier deLuis, and Alvar Saenz-Otero following guidelines presented in formal systems courses at MIT [Crawley, 2003]. Using these professional guidelines, the principles are presented using the following structure:

1. Principle name
2. Descriptive version of the principle - presents the principle in a way that its characteristics are understood for observation of a design to determine if said design includes the principle
3. Prescriptive version of the principle - presents the principle so that it can be used as a guideline in the creation of design goal or requirements
4. Basis of the principle - relates the principle to previous chapters to explain the basis upon which the principle was derived
5. Explanation - describes the principle in full

5.1 Principle of Iterative Research

Descriptive:

A laboratory allows investigators to conduct multiple cycles of the iterative research process in a timely fashion.

Prescriptive:

Design a laboratory so that complete research iterations can be performed at a pace appropriate for technology maturation.

Basis:

Facilitating the iterative research process was found to be a primary high-level feature of the MIT SSL Laboratory Design Philosophy (Chapter 3). The scientific process, the most common procedure used for scientific research, is iterative in nature. Therefore, conducting microgravity research must be an iterative process and a laboratory to conduct research must facilitate iterations.

Explanation:

It is essential for the scientific process that a hypothesis can be tested and modified as experiments are performed. As compared to the iterative research process originally explained in the development of the SPHERES laboratory (Figure 4.8 on page 118), the principle of iterative research dives further into the full process of technology maturation. This principle covers all the areas of the process: the conception of the problem, development of high-level hypothesis and designs, and test and evaluation of specific implementations.

For completeness, we define the different steps of the iterative process as utilized by this principle and the different feedback loops in the process:

- Conceive the need for a new technology and define its required capabilities.

  - Specify the intended benefits of the technology for the intended audience.
  - Develop the science requirements of the technology.
- Hypothesize about the goals and performance that can be achieved using a particular instantiation of a technology.
  - Develop the initial functional requirements needed in a facility to test the hypothesis.
  - Define the operational environment necessary to mature the technology.
- Design the facilities that allows this performance to be tested and confirms or refutes this hypothesis.
- Develop specific experiments to test the technology.
- Conduct the experiments to obtain data that is sufficient to support (or refute) the hypothesis.
- Analyze the data obtained, compare it with the goals and performance requirements developed during the hypothesis formulation, and determine whether to run further tests, change the experiment, update the hypothesis, or finish the tests reaching successful technology maturation.

Figure 5.1 illustrates the iterative research process used under this principle. The figure illustrates three possible decisions after data analysis:

1. Repeat the test to obtain further data. This feedback loop requires the experiment to run multiple times with repeatable and reliable results while maintaining a low risk of failure in case an unreliable experiment is run.
2. Modify the experiment design to allow for comparison of different designs conceived after the hypothesis to find the best design possible. To enable different designs the experiment facility must allow reconfiguration of its hardware and/or software.
3. Modify the hypothesis about the goals and performance requirements for the technology. This option results in changes to the science requirements for the facility, and therefore the ability to respond to these changes requires a facility to support substantial reconfiguration. Therefore, it is possible that a single facility cannot support this feedback loop, but rather that in these cases a new facility will have to be designed. The scientist must be aware of the existence of this loop not necessarily to design a facility which allows these types of modifications, but rather to be aware that a single facility may not be sufficient to mature a technology.

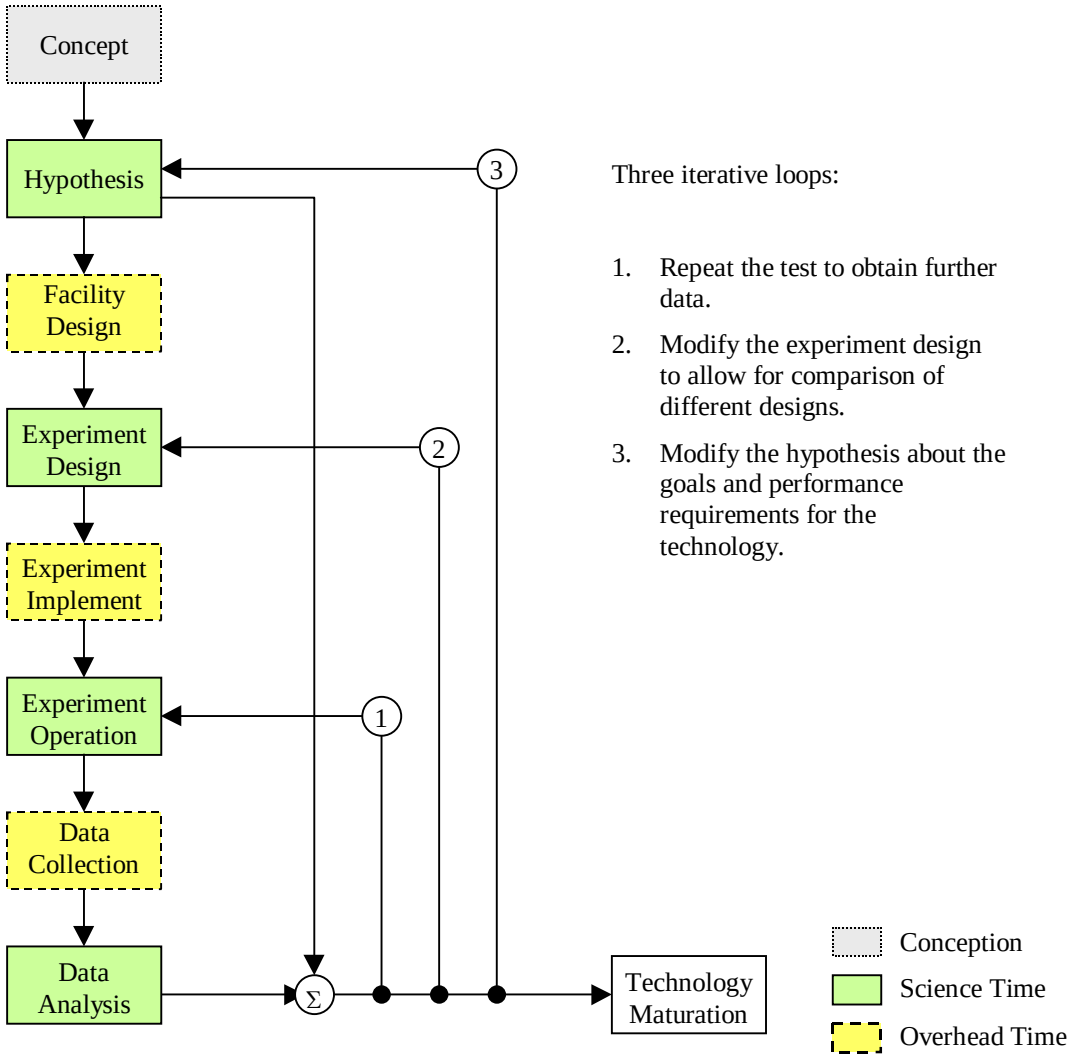


Figure 5.1 The iterative research process

Figure 5.1 shows the steps of the process (problem conception, hypothesis formulation, facility design, experiment design, experiment operations, data analysis, and technology maturation) and the main three feedback loops (repeat experiments, modify experiments, or modify the hypothesis). The figure categorizes the steps into three groups: the conception stage, science time, and overhead time. The definitions these times follow those presented in Section 4.3.1 on page 117: conception time is spent in the initial development of the problem; science time is spent by researchers developing new hypothesis or experi-

ments and analyzing the data; overhead time is spent in enabling science time to occur. To actually *facilitate* the iterative research process, a laboratory must ensure that science time is maximized and flexible, while overhead time is minimized.

The principle of iterative research defines as science time the time spent formulating and modifying a hypothesis, developing specific experiments to test the hypothesis, operating the facility to obtain sufficient data, and analyzing the data (similar to what is presented in Chapter 4). Science time should be maximized and it should be flexible. That is, a researcher needs to have ample time to analyze data and determine new experiments and hypothesis without the pressure that the ability to conduct new experiments may expire. But the time must also be flexible, so that if a scientist is ready to conduct a new experiment, they can do so quickly, without a wait that would cause the loss of interest and/or relevance in the investigation or depletion of the resources available. Therefore, the operational plans of a laboratory should not prescribe strictly fixed research intervals, but rather provide scientists with a flexible schedule to conduct experiments. By minimizing the overhead time, a laboratory allows scientists to conduct experiments within short periods of time if they so desire. By ensuring the laboratory operates over an extended period of time, a laboratory provides researchers with enough science time.

The overhead periods are the time spent in designing the facility, implementing a specific experiment, and collecting data. The implementation of an experiment and data collection are described in Section 4.3.1 on page 117. Of special importance is the fact that the design of a facility is considered overhead time. A facility is built to support technology maturation, but it is not the technology itself. Therefore, if a scientist changes a hypothesis and must modify a facility, the time spent in implementing those modifications represent an overhead. A successful laboratory utilizes facilities which minimize the time needed to modify them, so that scientists can modify their hypothesis freely, without the worry that a change in a hypothesis will result in changes that would drive the project beyond its constraints. The principles presented in this thesis guide directly towards this goal: minimiz-

ing the time to design a facility by providing design guidelines and minimizing the time to modify a facility by considering the use of resources available in the ISS and modularity.

This principle considers the "depth" of the research: how deep an understanding of a specific area of research the laboratory allows a scientist to obtain. The more iterations, the better results for that specific experiment can be, and the deeper the understanding of the technology. This allows that specific area of the technology to mature utilizing the laboratory facilities designed under this principle.

5.2 Principle of Enabling a Field of Study

Descriptive:

A laboratory provides the facilities to study a substantial number of research areas that comprise a field of study.

Prescriptive:

The development of a facility that is to be part of a laboratory must allow investigation of multiple research areas within the field of study, supporting the necessary number of scientists to cover the field.

Basis:

The definition of a laboratory calls for it to allow research of a *field of study.* The MIT SSL Laboratory Design Philosophy (Chapter 3) calls to *support multiple investigators*. This principle originates from the two concepts. Past experience has demonstrated that to achieve technology maturation a filed of study must be researched by several scientists. The combination of their knowledge achieves technology maturation. While a successful experiment could conceivably allow research on a field of study without supporting multiple scientists, it is *almost always valid* to claim that multiple scientists will need to research the technology to achieve its maturation. In the rare case that a laboratory may

allow a field of study to be researched by a single scientist, that is sufficient to satisfy this principle, as it would meet the definition of a laboratory.

Explanation:

In order to provide experimentation in a field-of-study, a laboratory must allow for experiments within the different research areas of the field. In order to conduct research on a field of study, all aspects of that field of study must be researched. Because researching a field of study is a large endeavor, it usually involves multiple scientists to work together to understand the field. Individual scientists concentrate on specific areas of the field, so that together the field is understood.

Therefore, this principle prescribes that:

- The study of multiple topics requires multiple experiments to be performed.
- Multiple investigators must work on individual topics to cover the whole field of study.
  - *Therefore* multiple investigators, whom perform experiments in their specific area of expertise within the field, must be supported.
- The laboratory must facilitate bringing together the knowledge from the specific areas to mature understanding of the field of study.

This principle considers the "breath" of the research, how much of a research area can be learned from the experiment. The larger the number of specific areas that a laboratory enables, the more technology matures.

5.3 Principle of Optimized Utilization

Descriptive:

A well-designed laboratory considers all the resources available and optimizes their use with respect to the research needs.

Prescriptive:

Consider all resources available to support the facility and optimize their use to benefit the research goals.

Basis:

Chapter 2 identifies the many special resources provided by the ISS, presenting the different facilities and tools available for research. Past MIT SSL experiments, presented in Chapter 3, demonstrate the need to use those resources correctly. The development of the SPHERES testbed (Chapter 4) concentrated heavily on the use of the ISS resources to reduce the challenges of microgravity research and fulfill the MIT SSL Laboratory Design Philosophy. But SPHERES does not utilize every one of the facilities and tools available aboard the ISS; rather, it makes the *optimal* use of those resources available to help it achieve its mission. Therefore, this principle originates not only from the fact that special resources exist on the ISS, but also from the need to customize the use of those resources to best fit the research objectives.

Explanation:

As presented in Chapter 2, the International Space Station offers a wide range of unique resources that make it ideal for the maturation of space technologies. While available to scientists, these resources are highly valuable, and they should be used in the best possible ways. Rather than thinking about using the least resources possible, this principle guides the researchers to use the resources in the best manner possible; i.e., the goal is not to minimize the use of resources, but to optimize its use with respect to the research goals.

The special resources of the ISS were identified in Chapter 2; these are the resources that we wish to utilize to fulfill the science goals:

- **Crew** - Human presence is one of the most important characteristics that separate ISS operations from standalone spacecraft. The crew can help reduce the risk of an experiment, intervening in the case of unsuccessful tests

to allow continuos operation of the facility even after failures in the theory. The correct use of the crew also reduces the complexity of facilities as less automation is needed. Most importantly, the crew can provide feedback to the researcher based on observations during the conduction of the experiment. The presence of the crew allows a human to interpret the operations of the facility and success of experiments, rather than depending solely on machine-captured data.

- **Power sources** - The ISS was designed to provide substantial amounts of electrical power research experiments, as well as several pressurized gas and liquid resources. Each experiment location is provided with kilowatts of electrical power. Many locations also provide cooling elements, nitrogen, and carbon monoxide. The use of these resources can greatly reduce the cost of a mission by directly reducing the required mass; alternatively, it can increase the value of the mission by allowing more mass and volume to be used for research activities.
- **Data telemetry** - The ISS communications system, in constant expansion, is clearly a special resource which benefits all users of the ISS. The availability of continuous high-bandwidth communications to ground reduces the cost and complexity of missions which would otherwise need their own communications equipment. Existent resources allow scientists to obtain their data, if saved within the ISS data handling systems, within hours of the experiments; scientist can use the system to upload new software. The bi-directional nature of the existing communications enables an ISS laboratory to close iterative research loops, allow software reconfiguration, and support multiple scientists in the use of one facility. Further, the availability of ever-increasing communications features will enable real-time video and other teleconferencing options as part of daily research operations to better create a virtual presence of scientists aboard the ISS.
- **Long-term experimentation** - A unique features of the ISS is that it allows long-term microgravity experimentation in a laboratory environment. The long-term nature of the ISS allows a laboratory to enable the iterative research process by creating flexible operations schedules. Further, the long-term nature of the ISS allows technology to mature over incremental, controlled steps, without the need to constantly test high-risk equipment.
- **Benign Environment / Atmosphere** - All projects, whether they reside inside or outside of the ISS pressurized environment can benefit from the benign environment. A facility operated aboard the ISS can concentrate on the science rather than on survival of the project, since the ISS provides substantial infrastructure to protect the projects and their operations. The presence of humans, even if they don’t interact with the experiment, protects the facility. Continuos monitoring of all ISS operations further safeguards the experiments. The controlled and measured environment protects the facili-

ties through the availability of structural elements designed specifically to support research.

The pressurized environment of the ISS not only provides safety for humans, but also for electronics and structures. Experiments that can be performed inside the station can have a substantial reduction in cost, complexity, and risk, as compared to free-flyers in space, since they no longer need to worry about being exposed to the space environment radiation and vacuum.

This principle considers the resources of the ISS as elements which provide *value* to a laboratory. Rather than thinking about the use of the resources as a cost to the project or the ISS, the principle states that the correct use of each resource can provide positive value to a laboratory, and that the correct use of its resources has a positive effect on the ISS itself.

5.4 Principle of Focused Modularity

Descriptive:

A modular facility identifies those aspects of specific experiments that are generic in nature and allows the use of these generic components to facilitate as yet unforeseen experiments. Such a facility is not designed to support an unlimited range of research, but is designed to meet the needs of a specific research area.

Prescriptive:

During development of a facility identify the generic components while ensuring the initial research goals are met.

Basis:

The MIT SSL Laboratory Design Philosophy (Chapter 3) calls for the creation of *generic vs. specific equipment* while allowing both *hardware* and *software reconfiguration*. Further, it calls for the creation of a *physical end-to-end simulation* of the technology. The SPHERES laboratory, even without having reached the ISS, has allowed multiple scientists to perform experiments over several years due to its generalized hardware and sup-

port of reconfiguration. Therefore, it is concluded that any successful laboratory that is to operate aboard the ISS can benefit from a clear distinction between general purpose equipment and science-specific features while remaining focused on its initial science goals.

Explanation:

Since experiments *almost always* contain basic elements that can support other similar experiments, the design phase of a facility should identify these common elements. These generic parts should be made available for future experiments as long as it does not compromise the mission of the original experiment. In this fashion, a laboratory is created by accepting facilities that provide some form of generic equipment which can be later used by new experiments.

The call for *focused* modularity is to prevent a "do-everything" system which may deviate the facility from meeting its original goals. The generic equipment should be identified after the design of the original experiment; the original design should not be to create generic equipment.

If a system does not have any components that meet any of this criteria, then there is a high probability that the scientist chose a narrow field of study for the experiment, such that the design of the facility does not share any common components with other possible experiments in the same field. Note that while this possibility reflects back to the Principle of Enabling a Field of Study, the Principle of Modularity remains separate. An experiment that enables a field of study does not necessarily have to be modular; or vice versa, a fully modular facility may not enable a whole field of study, but it may allow deep understanding of a small area of study.

5.5 Principle of Remote Operation & Usability

Descriptive:

A remotely operated laboratory, such as those in the ISS, must consider the fact that remote operators perform the everyday operation of the facility while research scientists, who do not have direct access to the hardware, are examining data and creating hypothesis and experiments for use on the facility.

Prescriptive:

An ISS experiment must accommodate the needs for a remote operator and a research scientist not in direct contact with the experiment.

Basis:

These principles are specifically intended to support the development of laboratories for operations aboard the International Space Station. As Chapter 1 explains, the development of all ground based laboratories, even those in remote locations, stresses the need to allow scientists to be present in the laboratory. The use of the ISS not only precludes the idea that the scientist be present at the laboratory, but Chapter 2 even presents several challenges to the effective use of the ISS crew time. Therefore, as opposed to the development of ground-based laboratories, ISS-based laboratories must provide the necessary facilities to account for remote operations and provide the correct usability for both the operator and the scientists in the ground.

Explanation:

Remote laboratories are based on remote locations because they offer a limited resource that researchers cannot obtain in their home locations. The design of remotely operated laboratories must account for the following facts about the operation:

- Operators

  - Are usually not experts in the specific field.
  - Are a limited resource.
- Research Scientists
  - Have little or no experience in the operational environment.
  - Are unable to modify the experiment in real-time.
  - Are usually an expert in the field but not in the development of facilities and testing environments.
  - May not have full knowledge of the facility design, especially when multiple scientists are invited to participate as part of a larger project.

The goal of a remote facility is to allow for a virtual presence of the research scientist in the operational environment. This includes the need for continuous communications between the operator and the research scientist, preferably in real-time. The availability of real-time two-way video is an important resource that benefits remote operations. In all cases, the use of high bandwidth communication systems, even if not real-time, should maximize the transfer of knowledge between the operator and researcher, especially when that is required to operate the facility successfully. In general the operator should have some idea of the expected results of each experiment in order to quickly transmit to the researcher information. In other words, the researcher should not solely depend on the communication of data, but also use the operator for feedback on the experiment.

Ultimately, the remote environment should allow a full virtual presence of the research scientist, where the operator becomes an extension of the scientist.

5.6 Principle of Incremental Technology Maturation

Descriptive:

A successful ISS laboratory for technology maturation allows technology maturation to transition smoothly between 1-g development and the microgravity operational environment in terms of cost, complexity, and risk.

Prescriptive:

Provide a representative μ-g environment that allows researchers to maturate technology in incremental steps between earth-based prototypes and flight equipment.

Basis:

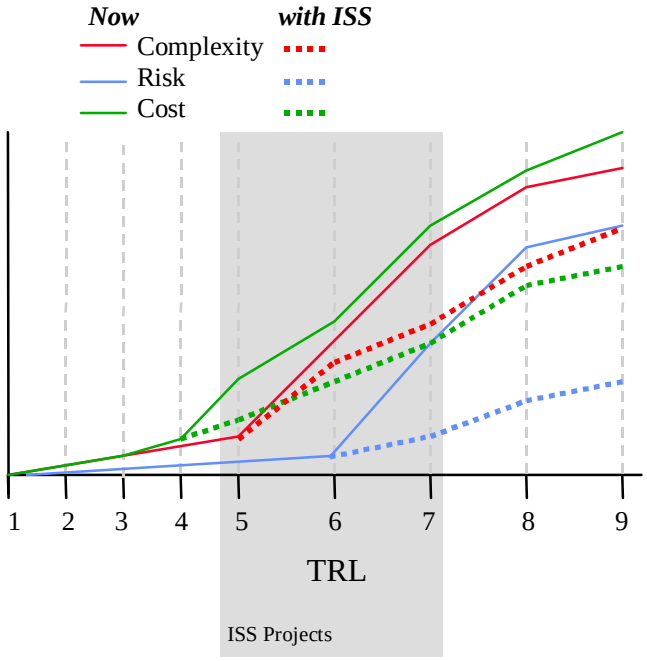
Chapter 2 identifies the primary challenges of microgravity research as risk, complexity, cost, remote operations, and visibility. Chapter 1 presents the concept of Technology Readiness Levels; Figure 1.2 on page 39 illustrates the general trend of three of these challenges (risk, complexity, and cost) to increase substantially as a project progresses through the TRLs. This principle emerges from the need to mature technology with limited and smooth increments of risk, complexity, and cost as the technology matures. The steepest increases originate from the need to provide a *relevant environment*; this principle calls for the correct use of the ISS environment, presented in Chapter 2, to create said environment without the current steep jumps pictured in Figure 1.2.

Explanation:

Technology maturation is an essential step for space programs. Current Technology Readiness Levels are used as a baseline to evaluate when a new technology is ready for flight. Due to the large jumps in cost, complexity, and risk between TRLs, they are not always followed systematically. Higher TRLs call for operations in a *relevant* environment to demonstrate maturation. A *relevant* environment is representative of the final operational environment in space; creating such an operational environment usually causes the steep jumps in cost, complexity, and risk. The lack of access to a representative space environment hinders the ability of scientists to demonstrate technologies at all TRLs. Therefore, there is a need to better support the maturation of technologies by enabling access to a relevant environment without steep jumps in complexity, risk, and cost, allowing incremental technology maturation.

The goal of incremental technology maturation is to make the complexity, risk, and cost increase smoothly as one moves across TRL levels, while being realistic of the changes in the environment required. With current test environments, excluding the ISS, there is an important steep jump when moving from the component level (TRL 4) to the system level (TRL 5) in a relevant environment, and a similar, if not steeper, jump when moving from a relevant (TRL 6) to a space environment (TRL 7). Further, the definition of relevant environment is not exact, sometimes leading to a relevant environment being a high-fidelity simulation and analytical model, rather than physical exposure to the system. Therefore, in many cases, the jump from TRL 6 to TRL 7 is very steep. The ISS provides an environment that can closely, if not fully, satisfy the requirements for a space environment; yet the presence of humans in the ISS can greatly reduce the risks involved, and the existence of the ISS itself can reduce the costs. Further, successful tests in the ISS may lead to less complexity when moving to higher TRL levels by providing scientists with a better understanding of the system.

Figure 5.2 builds on Figure 1.2 to present a pictorial representation of the increase in challenges as technology matures through the TRLs both with and without the use of the ISS as a host. The goal of incremental technology maturation is presented in the dotted lines: as one enters TRLs 5, 6, and 7, the ISS provides an environment where cost, risk, and complexity do not go through substantial jumps. The major increases should only be seen as the project leaves the benign environment of the ISS and enters the space environment. These increases should not be as pronounced as before, since the technology has been demonstrated in full microgravity conditions; the increases should be due to technical requirements, the need for new hardware, and the inherent challenges of launching a spacecraft into orbit; but the increases should no longer be due to any remaining need for further scientific knowledge of the problem.



TRL 1-2	Basic principles & concept
TRL 3-4	Proof-of-concept & laboratory breadboard
TRL 5	Component validation in relevant environment
TRL 6	System prototype demonstration in relevant environment
TRL 7	System prototype demonstration in space environment (usually skipped due to cost)
TRL 8	Flight system demonstration in relevant environment
TLR 9	Mission Operations

Figure 5.2 Smoothing TRL transitions

5.7 Principle of Requirements Balance

Descriptive:

The requirements of a laboratory are balanced such that one requirement does not drive the design in a way that it hinders the ability to succeed on other requirements; further, the hard requirements drive the majority of the design, while soft requirements enhance the design only when possible.

Prescriptive:

Maximize the hard requirements of a design and balance their effect on the design; minimize the soft "decirements" and ensure they don't drive substantial portions of the design.

Basis:

Chapter 2 presents the use of the ISS; Chapter 3 calls for the implementation of multiple features to satisfy the MIT SSL Laboratory Design Philosophy. The two chapters do not necessarily call for the same design to be created. Further, neither chapter accounts for the viability or cost to create a laboratory which implements the features called for. This principle arises from the lessons learned in the development of the SPHERES laboratory, which fulfills the majority of the ideas of Chapters 2 and 3. The design of SPHERES required several iterative design cycles to implement the features called for in Chapters 2 and 3 while remaining within the necessary cost and implementation constraints. The development necessitated that the different requirements which arise from the use of the ISS and the MIT SSL Laboratory Design Philosophy to be continuously reviewed so that no single requirement drove the project outside of its constraints.

Explanation:

Hard requirements are usually set at the start of a project to determine the goals that must be met; they are mostly quantitative. Soft 'decirements' are features desired by the scientists but which do not necessarily have a specific value or which are not essential for the success of the mission. A successful design creates a realistic set of requirements, maximizing the number of hard requirements, while taking into account the other principles presented herein:

- Balance the need for depth and breadth of a laboratory.
- Determine the correct amount of modularity needed.
- Prevent use of resources that are not needed; utilize the useful resources to their maximum.

Developing requirements is an iterative process just like any other system design problem, therefore to meet this principle the scientist is expected to iterate on the requirements of the other principles and then balance them. The other principles should be evaluated first, so as to develop a set of basic requirements for the facility. Using the requirements created

from the other principles, this principle calls for the balance of effort into each of the other principles.

This principle does not call for all the requirements to be perfectly balanced or to necessarily eliminate the soft requirements; rather, this principle calls for the scientist to proactively pursue a realistic justification for each requirement and to ensure that a substantial part of the effort into the development of the facility goes towards clearly defined needs.

5.8 The Design Principles, the Design Philosophy, and the ISS

These chapter incorporates all the features of the MIT SSL Laboratory Design Philosophy (Chapter 3) for use in experiments which operate aboard the International Space Station (Chapter 2) into a set of concise design principles which broaden the scope of their applicability into a wide range of space technology maturation missions. Table 5.1 relates the design philosophy and use of the ISS to the design principles, demonstrating the ability of the principles to not only incorporate all of the features presented in Chapters 2 and 3, but also to account for critical design issues which were not directly present in the previous chapters.

All the features of the MIT SSL Laboratory Design Philosophy are accounted for in the Microgravity Laboratory Design Principles. The high-level feature of *facilitating the iterative design process* translates directly into the *Principle of Iterative Research*, with the majority of the features within the group *support of experiments* also being part of the iterative research principle. The high level feature of *supporting multiple investigators* joins several reconfiguration features to form the *Principle of Enabling a Field of Study.* The larger group to *support reconfiguration and modularity* is part of both the *Principle of Enabling a Field of Study* and the *Principle of Focused Modularity.* The principle of focused modularity describes why these features form part of both principles, since a laboratory could potentially support a field of study without being modular. The *Principle of Operations and Usability* is based on features of the MIT SSL Laboratory Design Philoso-

TABLE 5.1 Design Principles, the Laboratory Design Philosophy, and the ISS

Principle	SSL Design Philosophy	Support of Experiments					Reconfiguration and Modularity					
	Facilitating Iterative Research Process	Data Collection and Validation	Repeatability and Reliability	Human Observability and Manipulation	Supporting Extended Investigations	Risk Tolerant Environment	Supporting Multiple Investigators	Generic versus Specific Equipment	Physical End-to-End Simulation	Hardware Reconfiguration	Software Reconfiguration	ISS
Iterative Research	✓	✓	✓		✓	✓						
Enabling a Field of Study					✓		✓		✓	✓	✓	
Optimized Utilization												✓
Focused Modularity								✓		✓	✓	
Remote Operations & Usability	✓	✓		✓								✓
Incremental Technology Maturation						✓						✓
Requirements Balance												

phy as well as the operations of the ISS to ensure that a remotely operated facility utilizes the ISS correctly and enhances research at the same time. The use of resources available in the ISS is captured within the *Principle of Optimized Utilization*. The challenges of microgravity research, presented in Chapter 2, are addressed together with the need to create a *risk-tolerant environment* within the *Principle of Incremental Technology Maturation.* Lastly, the *Principle of Requirements Balance* glues together all the other principles beyond what the MIT SSL Laboratory Design Philosophy and the literature research on the ISS call for. The principle of requirements balance is an oversight of the other principles to ensure that a mission is successful.

5.9 Science in the ISS to Date: Applicability of the Principles

This section reviews the science conducted aboard the ISS so far to identify common designs and operations implementations to identify if the principles presented in this thesis are exhibited in past experiments, even if not specifically designed to do so. The existence of the traits of the principles in past experiments provide insight into how the principles should be applied to future experiments.

The ISS is currently hosting the crew of Expedition 10. Expeditions 1-7 consisted of three crew members; expeditions 8-10 have two crew members. The smaller crew on the later expeditions has limited the ability to conduct science aboard the ISS, therefore it is more relevant to study the science conducting during a 'full' expedition. Expedition 6, which was the last expedition to operate with a standard crew of three and performed the expected number of experiments that will take place in the long-term, has been fully researched. Table 5.2 shows all the experiments in Expedition 6. The NASA White Papers [NASA, URL1] about each experiment were reviewed to understand the design and operations of each project. The white papers provide sufficient information to identify the general characteristics of the experiments and determine whether the design follows a specific principle. These reviews do not evaluate the experiments, the reviews identify if past experiments exhibit the characteristics of a principle to determine the applicability of the principles.[1]

The Expedition 6 results demonstrate some important trends related to the principles. As Chapter 2 explains, the thesis concentrates on iterative space technology maturation experiments. Expedition 6 conducted 21 different experiments: ten in the bioastronautics area, six in the physical sciences, two in space product development, and three in space flight technologies. Out of the ten bioastronautics experiments, six are *exposure* experi-

1. The Principle of Requirements Balance is **not** used in this review since the deployment of the project aboard the ISS implies the mission successfully met its principal requirements. Further, the basic information presented in the white papers does not provide enough insight to determine specific requirements of a mission.

TABLE 5.2 Experiments in Expedition 6

Id	Field	Experiment	*Repeat*	*Size*	Iterative	Field	Utilization	Modularity	Usability	Maturation
1	Bioastronautics Research	The Effects of EVA on Long-term Exposure to Microgravity on Pulmonary Function (PuFF)	✓	*M*			✓	✓	✓	
2		Renal Stone Risk During Space Flight: Assessment and Countermeasure Validation (Renal Stone)	✓	*M*			✓		✓	
3		Study of Radiation Doses Experienced by Astronauts in EVA (EVARM)	✓	*M*			✓		✓	
4		Subregional Assessment of Bone Loss in the Axial Skeleton in Long-term Space Flight (Subregional Bone)	✓	*S*	Only Pre/Post flight					
5		Effect of Prolonged Spaceflight on Human Skeletal Muscle (Biopsy)	✓	*S*	Only Pre/Post flight					
6		Promoting Sensorimotor Response Generalizability: A Countermeasure to Mitigate Locomotor Dysfunction After Long-duration Space Flight (Mobility)	✓	*S*	Only Pre/Post flight					
7		Spaceflight-induced Reactivation of Latent Epstein-Barr Virus (Epstein-Barr)	✓	*S*	Only Pre/Post flight					
8		*Entry Monitoring*			*DELAYED*					
9		Chromosomal Aberrations in Blood Lymphocytes of Astronauts (Chromosome)		*S*	Only Pre/Post flight					
10		Foot/Ground Reaction Forces During Space Flight (Foot)		*L*			✓		✓	
11	Physical Sciences	Protein Crystal Growth-Single-locker Thermal Enclosure System (PCG-STES)	✓	*L*	✓	✓	✓	✓	✓	✓
12		Microgravity Acceleration Measurement System (MAMS)	✓	*M*						
13		Space Acceleration Measurement System II (SAMS-II)	✓	*M*						
14		Investigating the Structure of Paramagnetic Aggregates from Colloidal Emulsions for the Microgravity Sciences Glovebox (MSG-InSPACE)	✓	*L*	✓		✓		✓	
15		Vibration Isolation System for the Microgravity Sciences Glovebox (MSG-g-LIMIT)	✓	*M*	*No data*					
16		Coarsening in Solid-Liquid Mixtures for the Microgravity Science Glovebox (MSG-CSLM)		*M*			✓		✓	
17	Space Product Development	Zeolite Crystal Growth Furnace (ZCG)	✓	*L*		✓		✓	✓	✓
18		Microencapsulation Electrostatic Processing System (MEPS)	✓	*L*		✓		✓	✓	
19	Space Flight	Crew Earth Observations (CEO)	✓	*S*			✓			
20		Earth Knowledge Acquired by Middle-School Students (EarthKAM)	✓	*S*						
21		Materials International Space Station Experiment (MISSE)	✓	*L*		✓	✓	✓	✓	✓

ments (one was delayed), and the principles do not apply to them, as they do not create the facilities to implement a laboratory, but rather only use the fact that humans are exposed to the microgravity environment. Of the remaining four experiments, three exhibit the characteristics of the *optimal utilization* and *operations and usability* principle. The last experiment (PuFF) makes use of *modularity.*

The six physical science experiments are more evenly divided in the use of the principles. Two experiments (PCG-STES and MSG-InSPACE) exhibit many characteristics of the principles. It is interesting to see that these two experiments are the only ones that clearly exhibit the ability to perform iterations aboard the ISS. The experiments provide the necessary facilities for astronauts to repeat experiments in a manner that advances the iterative research process. While MSG-InSPACE appears limited in scope, PCG-STES provides research facilities for a large number of scientists to conduct a wide range of protein crystal growth experiments. Further, PCG-STES exhibits modularity. Both experiments utilize the resources available on the ISS to simplify the design of their facilities and enhance their capabilities by utilizing the astronaut time efficiently.

On the other hand, several physical sciences experiments on this expedition were effectively exposure experiments. The MAMS and SAMS-II experiments simply collect data for analysis later on. They do not exhibit any of the characteristics of the principles.

Space product development shows a growing trend toward exhibiting the characteristics of the design principles. One experiment, ZCG, exhibits a large number of the principles, only lacking enabling iterative research (while the utilization does not appear optimized, since it appears that the experiment could benefit from further crew time utilization and better interfaces, it correctly uses the standard ISS experiment rack supplies). The MEPS experiment also lacks the ability to enable the iterative research process, and it seems it would benefit strongly from better use of the ISS resources. But, the experiment does provide modular facilities for multiple researchers and has been designed to operate remotely with ease.

The space flight experiments of Expedition 6 consisted of an observation experiment (CEO), an educational experiment (EarthKAM), and an space technology research experiment (MISSE). The first two experiments do not exhibit a substantial number of characteristics from the principles. MISSE, on the other hand, exhibits several characteristics of the principles. MISSE is an exposure experiment, in that its samples are located outside the ISS and left unattended for an extended period of time; therefore, MISSE does not enable iterative research. But the facilities of MISSE do provide a modular setup where a large number of scientists can study a substantial amount of materials science. Further, the design directly accounts for several of the resources of the ISS: power, benign environment (exposed), and long-term experimentation. MISSE even accounts for the use of crew time since the exposed facility is designed to be accessed by EVA in case changes are needed. Lastly, MISSE was designed to allow cheap access to the space environment to better understand material science, effectively creating a facility to enable incremental technology maturation for space materials.

A trend identified in the operational description of a majority of these projects is that many researchers are proud that their facilities practically do not use crew time. In many cases the crew simply turns the experiment on and does not interact with it again until samples or data must be returned to the scientists. The extremely limited crew time has clearly pushed experiments to operate autonomously, and the fact that a human is present in the operational environment has not been utilized correctly. As a consequence of requiring autonomous experiments due to limited crew time, all these autonomous experiments do not enable iterative research aboard the station. Rather, the experiment provides data for one iteration; subsequent operations require delivery of further hardware to or from ground and direct interaction of the scientist.

The experiments of Expedition 6 confirm the stated intent of the principles: to guide in the development of iterative space technology maturation experiments. The review of the experiments clearly indicates that the principles do not apply to observation or education. On the other hand, the experiments of Expedition 6 which required multiple samples, a

wide range of scientists, or interfaces with the crew exhibited the characteristics of a large number of the principles. Further, it is interesting to note that these experiments are physically and operationally *large* compared to the other experiments; the need to provide the necessary facilities requires the experiment to utilize more space.

Past experiments of the ISS show that the principles presented in this chapter are applicable to space technology maturation experiments conducted aboard the ISS. While not all of the experiments conducted aboard the ISS will benefit directly from these principles, it is clear that a substantial portion of the larger experiments conducted aboard the ISS will. Therefore, a researcher who identifies a new technology need requires clear and concise guidelines on how to apply these principles to achieve technology maturation. The next section presents a design framework to aid scientists in following these design principles.

5.10 Design Framework

The design framework concentrates on allowing a research scientist to design a laboratory, with its necessary ISS and ground based facilities, that meets the principles. The framework consists of several design steps which sequentially detail the requirements of the facilities that comprise the laboratory. The framework also provides general guidelines to evaluate if a specific design principle is being satisfied by the laboratory and determine whether there are benefits towards the maturation of the space technology by operating aboard the International Space Station. Through this framework the scientists can introduce their perspective of the science goals as well as the constraints of the project.

The application of the principles onto a new design does not occur in parallel for all the principles. As explained in the *Principle of Requirements Balance*, the design process is itself iterative, and therefore composed of several steps. With this in mind, the strategy presented in Figure 5.3 was developed. The figure groups the principles into the following main actions: determination of mission objective, identification of a field of study, initial design of a facility, identification of modular elements and design of operational elements, and balancing of the requirements. The application of each principle has been ordered so

at to create an incremental set of requirements for the design of the facility. The order should help refine the requirements at every step. These actions are iterated until a final design is achieved.

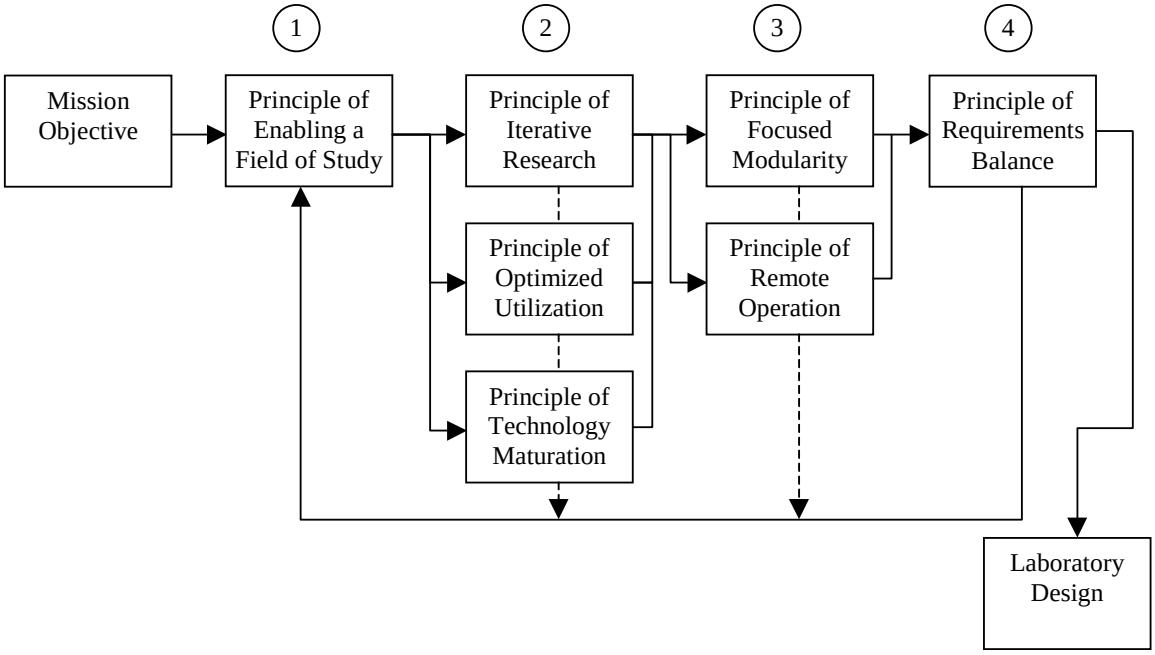


Figure 5.3 Design principles application strategy

The mission objective is determined by the customer. These objectives determine the science requirements of the mission. To satisfy these requirements, a laboratory must usually demonstrate results which cover a large area of study and which compare several designs to identify the best solution. The design principles of this thesis provide benefits when the research scientist charged with the mission determines that technology maturation is necessary and believes that the mission may benefit from operating in the ISS. In that case, the following steps should be taken to create a facility which will benefit from being in the ISS and which will facilitate the maturation of the space technology:

Step 1 - Identify a Field of Study

• Principle of Enabling a Field of Study

The first step is to identify the field of study the facility will support. The initial attempt is to select a large enough area in the field of study that the experiment can support technology maturation, but not so large that it is impossible to identify a clear set of science requirements.

Step 2 - Identify Main Functional Requirements

- Principle of Iterative Research
- Principle of Optimized Utilization
- Principle of Technology Maturation

The next step combines three principles that allow identification of the main functional requirements for the facility.

The principle of iterative research sets several requirements for the facility. Through this principle the scientist can determine the need for inputs and outputs, data capture and transfer rates, and requirements on the repeatability and reliability of the system. The principle also calls for the scientist to set requirements on the scheduling of experiment sessions and needs for data analysis.

The principle of optimized utilization provides an essential piece of information: should this facility be part of the ISS program or not? The principle requires the scientist to study the reasons for operating in the ISS and how the resources made available by the ISS are being used by the facility. It also gives the scientist an idea of which resources affect the facility heavily and which do not. Once it is determined which ISS resources should be used, a clear set of interface requirements to the ISS can be created.

For a facility to achieve technology maturation it must provide a representative environment for assembled sub-systems and/or prototypes. The definition of that environment and those systems are cast into clear requirements for the facility.

After all these requirements have been created it should be possible to identify a limited set of design options for the facility. These designs will be further studied in step 3.

Step 3 - Refine Design

• Principle of Focused Modularity

• Principle of Remote Operations and Usability

Step 2 identified all the major requirements for the system to achieve the science goals and helps create a limited set of candidate designs. Step 3 identifies those designs that best meet the call for modularity and ensures that the designs meet the need for remote operations.

As described within the principle of modularity, its goal is not that the initial objectives are for a modular system, but rather to identify those parts of a design that can be modular. Therefore, the principle of modularity is applied once a set of designs has been selected to search for those elements of the facility that meet the principle.

The principle of remote operation and usability requires a minimum set of information to provide valuable feedback, specifically: knowing what type of data the operator needs and what processing tools the scientist needs.

Step 4 - Review Requirements and Design

• Principle of Requirements Balance

Once the set of requirements has been finalized and a preliminary design conceived, the principle of balanced requirements calls for a review of the requirements prior to going on. At this point the scientist must evaluate whether the proposed facility has any requirements that have too much effect on the cost of the mission, and if they should be changed.

If the scientist determines that the requirements should be reviewed, the process should restart at step one to maintain objectivity. If the scientists agrees with the

weight of each requirement and determines that the mission objectives will be met, then the process is finalized.

These steps provide a general overview of the laboratory design process to implement the design principles. The following sections presents general guidelines to determine functional requirements which satisfy the design principles.

5.10.1 Step 1 - Identify a Field of Study

This step utilizes the *Principle of Enabling a Field of Study* to determine the breath of the research. The following section presents guidelines to determine the range of the research that should be possible to research in a space technology maturation laboratory.

Principle of Enabling a Field of Study

Identifying the field of study and the areas which comprise it is a subjective process conducted by the research scientist to ensure the mission science objectives are fulfilled. The science objectives sometimes immediately identify the field of study; for example, when the mission objective is to mature the technology of a specific spacecraft subsystem as defined by [Larson, 1992]. In this case, the field of study may be propulsion, avionics, or structures. The research scientist can then identify the areas of study which comprise this sub-system, and select those areas which the laboratory allows to be studied. Other times, the mission objectives may not immediately identify the field of study; the mission objectives may be too broad to clearly identify a field of study or too narrow to be considered a complete field. For example, the mission may call for the observation of stellar objects, in which case the scientist must determine what part of a spacecraft for space observation requires technology maturation, and select that part as the field of study. On the other hand, a mission objective may simply call for the optimization of a specific control algorithm; in those cases the scientist needs to determine if its possible to study a substantial part of the controls field, rather than concentrate on the single algorithm. If the objective of the research scientist is to develop a laboratory for space technology maturation, rather

than a single facility to test a specific concept, the research scientist must identify the field of study for the laboratory, even if that differs slightly from the science mission.

Once a field of study has been clearly defined, the subject areas which comprise the field of study must be identified. Each area must be complimentary to another, rather than replicate efforts on gaining the same type of knowledge. A laboratory allows the study of a meaningful number of the areas such that, when the knowledge gained from tests in all areas covered by the laboratory is brought together significant steps are taken to mature the technology.

The identification of specific areas of study allows the designer to determine if multiple scientists will need to conduct research in the laboratory in order to cover all the areas. Determining the need to support multiple scientist is necessary at this point, since the facilities will have different requirements depending on the number of scientists involved. Guidelines for these requirements are presented below. If at this point the determination is made that one scientist can perform all required research, there is a high probability that the initial field selection was too narrow; specific area of a field of study was selected, rather than a field of study. The designer should return to the first step and ensure that a field of study is being covered.

The facilities to support a field of study, researched by multiple scientists, must provide the following functionalities:

- Allow all scientists to create models of their work in their home locations.
- Create simple interfaces for the scientists to utilize the testing facilities.
- Ensure efficient data transfer between the scientists and the testing facilities.
- Provide flexible operations plans for scientists to conduct experiments.
- Enable software and/or hardware reconfiguration to cover all subject areas.

At this point it is useful to calculate an initial cost of creating the laboratory environment by using existing guidelines such as those presented in [Larson, 1992] or existing design tools such as those developed by [Matossian, 1996], [Shaw, 1998], [Mosher, 2000], and

[Jilla, 2002]. These initial costs can be used to determine the number of areas of study which should be covered by the laboratory to obtain a benefit by studying a substantial part of a field, rather than developing individual test facilities for each area of study. The following equation, based on the methods proposed in [Meyer, 1997] to measure the efficiency of product platforms, provides a general idea of the efficiency of using a laboratory rather than multiple facilities:

$$J = \frac{m}{n} \cdot \frac{K_{lab} + \sum_{i=1}^{n} k_i}{\sum_{i=1} K_i} \tag{5.1}$$

where:

m = total number of areas in the field of study

n = number of areas covered by the laboratory

K_{lab} = cost of the common laboratory facilities

k_i = cost of enabling each area of study

K_i = cost of developing an independent facility for each area of study

The numerator considers the costs of implementing derivative products; the numerator represents the cost of implementing new products every time. The factor m/n relates the total number of areas of study with the ability of the laboratory to support multiple areas; as more areas are supported, the factor decreases. The goal is to ensure that the cost J is less than 1.

Using the guidelines presented in [Larson, 1992] and the experience of the MIT SSL microgravity projects, it is possible to make the following assumptions:

- The costs K_i will always be larger than the costs k_i, since developing complete facilities requires substantial launch cost and development of common equipment.
- The costs k_i will not be constant for all ares of study.

- The cost K_{lab} will be larger than the costs K_i, since it requires the addition of resources to support multiple scientists.

Under these assumptions, the general trends of cost J can be represented graphically, as shown in Figure 5.4. While the figure shows only a pictorial representation of cost J, it provides the designer with important information:

- Utilizing this measure, a laboratory must cover multiple areas of study; even if the cost of a facility is small, not covering multiple areas of study results in a high cost.
- As the cost of supporting areas of study increases, a facility will need to support more areas to result in a cost factor less than 1.
- It is possible that supporting certain areas of study results too high and drives the cost beyond the threshold as it is added.

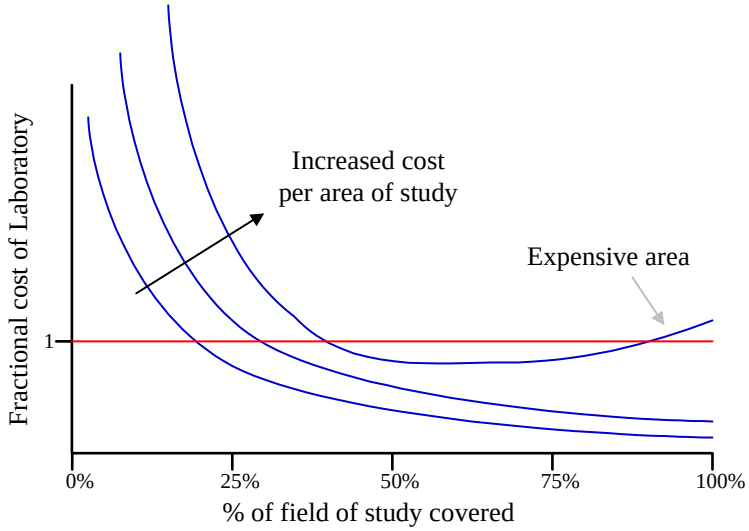


Figure 5.4 General trend of cost *J* using cost function *5.1*

5.10.2 Step 2 - Identify Main Functional Requirements

This step calls for scientists to determine the main functional requirements of the laboratory once the field of study has been defined. Three principles are used to determine these requirements: *Principle of Iterative Research, Principle of Optimized Utilization,* and

Principle of Technology Maturation. The guidelines to determine the functional requirements using these principles are presented next.

Principle of Iterative Research

Designing a laboratory to satisfy the Principle of Iterative Research requires that the facilities of the laboratory permit to close at least one of the iterative loops presented in Figure 5.1 on page 204; preferably, a laboratory which operates aboard the ISS not only allows repetitions of experiments (loop 1), but also modifications of experiments (loop 2) and the hypothesis (loop 3). For this to occur, a facility design must exhibit efficient and sufficient *data collection and analysis tools*, as well as the *ability to reconfigure* the facilities with new experiments that reflects the knowledge obtained during previous iterations. Further, the laboratory needs to develop a *flexible operations plan* which provides scientists with sufficient and flexible science time.

Data Collection and Analysis Tools

Data collection is part of the overhead time; data analysis is part of science time. The design of a laboratory should minimize the work to collect data, and ensure that the collected data are of the appropriate quality. At the same time, the laboratory must provide the correct tools for efficient data analysis to support or refute experiments and hypothesis.

Data collection consists of four main parts: capture bandwidth, precision, accuracy, and data transfer. For successful data collection, the following guidelines should be met:

- **Bandwidth -** The data sampling rate indicates the maximum bandwidth of the collected data. The facilities must be designed with the maximum bandwidth possible, so that they can be used for as yet unforeseen research. In no case should the Nyquist criterion (sample at twice the frequency of the highest mode of the system which should be observed) not be met for known system bandwidths; further, as per [Larson, 1992], the sampling factor should be 2.2 to account for model uncertainties.
- **Precision -** Data precision tells the scientist how small a change in the physical system can be measured. A high precision system can measure small

changes; a low precision system can only measure big changes. As a general guideline, the precision should be a fraction of the "impulse bit" (the smallest actuation possible) of a system, such that the sensors can successfully measure the effects of one impulse bit.

- **Accuracy -** Data accuracy accounts for how close a measurement is to the actual physical event. The higher the accuracy, the closer the measurement is to reality. Because accuracy greatly benefits from in-flight calibration [Larson, 1992], the scientist must design a laboratory so that its measurement systems can be calibrated once operational aboard the ISS. If in-flight calibration is not possible, the scientist must account for the physical effects of launch and deployment on the ultimate accuracy of the system and ensure it is of good enough quality for successful research.
- **Transfer -** Once captured, that data must reach the scientist. Data transfer does not necessarily need to be in real-time in order to meet this criteria, although real-time data could benefit experiments. But the transfer times must not substantially affect the available science time; data transfer times should never be a substantial fraction of data analysis time. Therefore, a facility must interface correctly to available communications resources to minimize the data transfer time.

Once the data reaches the scientist they must be analyzed. A complete laboratory provides scientists with data analysis tools which minimize the time between receiving the data and the start of analysis. It is not sufficient to make the data available to the scientist, they must be able to use it efficiently. For this, the laboratory must clearly define the type of data transferred to the scientist and the format in which the data is transferred. Further, tools to convert the data into those formats which best suit the needs of the scientist must be created before experiments are conducted, so that data analysis time is spent in examining the data, rather than in making the data available and presentable in the correct format.

Enable reconfiguration

In designing a laboratory which facilitates the iterative design process, it is necessary to determine the level of reconfiguration necessary to close each of the three iterative loops presented in Figure 5.1. The design process should include the identification of what is needed to close each of the three loops, and subsequently, based on the resources available

for the project, determine which loop is to be closed. The following points serve as guidelines to determine what is required to close each of the loops:

- Repeat Experiments
  - Ensure the operator's interfaces permit starting tests with minimal overhead from repetitive tasks.
  - Facilitate the repetition and/or measurement of initial conditions.
  - Enable resupply of consumables.
  - Provide sufficient space for data storage.
- Modify Experiments
  - Enable initial conditions to change sufficiently for the difference between experiments to be relevant.
  - Allow scientists to specify a different set of variables for the system, be it via hardware or software changes, consistent with DOE techniques ([Fisher, 1935],[Mead, 1988],[Antony, 2003]).
  - Allow changes in the response of the system to the same actuation by the addition or removal of hardware or software
- Modify the Hypothesis
  - Allow for the modification of both sensors and actuators so that different types of data are available as research progresses.
  - Ensure that new models of the problem are supported by the facilities that support the operator and the researcher.
    - The facilities aboard the ISS must allow any software or hardware which utilizes models of the problem to be fully reconfigured.
    - The data analysis tools of the researcher must allow its models to be updated.
  - Provide for the ability to modify the operational plans of the laboratory to accommodate for the need to develop new facilities; allow these new facilities to operate under new plans.

The capability of a laboratory to allow modifications must be bound by the limit of the available resources. For example, if a program only calls for one launch to the ISS, then it is not possible to claim that the system can be fully reconfigured by a second launch; reconfiguration must be enabled in the original facility. On the other hand, if a program has secured several flights to the ISS, it can utilize those to modify the facilities suffi-

ciently to enable modification of the hypothesis without having to implement more complex features in the original hardware.

Flexible Operations Plan

Enabling data collection and reconfiguration features in a laboratory does not guarantee successful iterative research. The iterative research process greatly depends on the availability of time for the scientist to conduct the necessary research. Therefore, a flexible operations plan which provides this science time is essential. Figure 5.5 illustrates this concept. The experience from MODE, DLS, and MACE has shown that too short or too long a time between iterations has a negative effect on the iterative research process. This concept is captured in the bottom plot. On the other hand, an experiment quickly benefits as iterations start; but the benefits from each iteration decreases each time; ultimately, the benefit asymptotes as the number of iterations increases. This concept is illustrated in the left side plot. The center plot shows that there exists a middle area where both the time between iterations and the number of iterations provide substantial benefits to the science. The goal is to design all laboratories to allow iterations in this area.

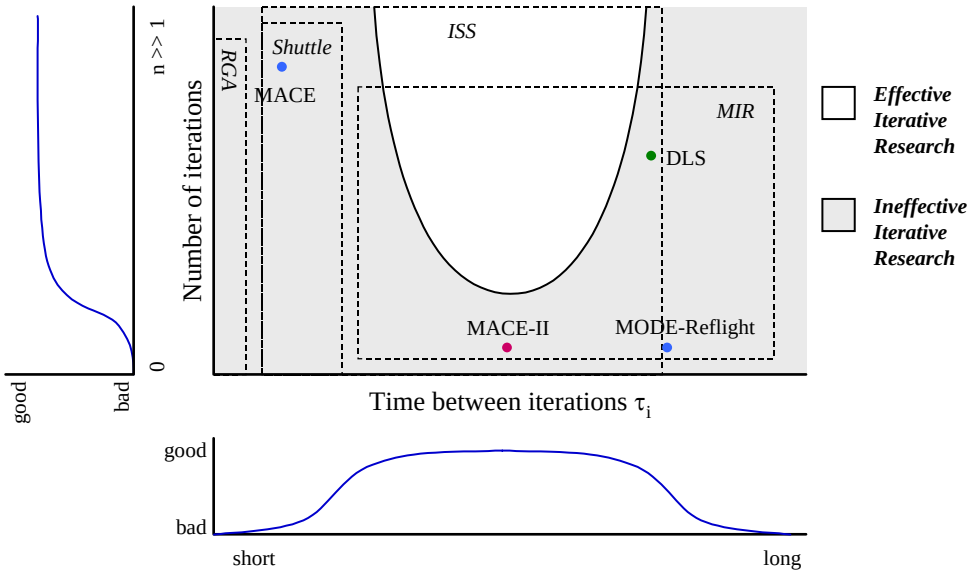


Figure 5.5 Achieving effective iterations though flexible scheduling.

Reduced gravity airplanes (RGA) provide very short time periods between iterations, only allowing the capture of data. MACE, MODE, and MODE-Reflight all operated aboard the Space Shuttle, which provides only short periods to analyze data and iterate. MODE-Reflight is shown twice, since the second flight was effectively a single iteration of the first flight, since lessons were learned, but the time between the two flights was exceedingly long. MACE-II operated aboard the ISS for an extended period of time, but the lack of communications at the time prevented a significant number of iterations to take place. DLS, which operated aboard MIR, was close to operating in an effective region of the iterative research process, but the long delays in communications between the researchers, NASA, and their Russian counterparts proved to have some negative effect on the research.

Figure 5.5 does not indicate quantitative figures for the actual time between iterations or the number of iterations that must be accomplished. Each scientist must determine the quantitative values for their specific research projects, since they can vary widely. The following guidelines should be followed in determining these values:

- An iteration consists of all the actions between conducting an experiment, collecting the data, achieving scientific knowledge, and applying that knowledge to mature a technology. An iteration is *not* the time it takes an operator to repeat an experiment; it includes all the steps to conduct the experiment, analyze the data, and determine the next step to take.
- Do *not* force the time between iterations, τ_i, to be fixed; find a minimum and maximum time between iterations which provides enough flexibility to the research scientist.
  - The minimum time between iterations must account for the need to analyze data and upload, at least, new experiments.
  - The maximum time between iterations must account for the resources available to the program. Also, it must account for other research which could reduce the value or replace the research of the laboratory.
  - Increasing the number of scientists using the laboratory enables the creation of a fixed time between iterations, as it increases the possibility that any one scientist will make use of the facilities when available, even if the others need further analysis time.

- It is beneficial to maximize the number of iterations which can be executed in the laboratory, especially if multiple scientists are involved in the research.

Principle of Optimized Utilization

A primary goal of the principle of Optimized Utilization is to change the way in which people think of the resources available in the International Space Station. The review of ISS experiments presented in Section 5.9 indicates that the majority of projects currently consider the cost of utilizing a resources; the design of existing facilities attempts to minimize the cost by reducing the use of resources. While in general this is a common goal of space missions, the use of the ISS should not follow those standards. ISS resources provide *value* to missions, and the correct use of these resources should maximize the value obtained by the project from using the correct resources for the specific science goals. Therefore, the development of a laboratory must consider all the resources available and optimize their use with respect to the research needs.

The first step in designing a laboratory which correctly utilizes the resources available aboard the ISS is to understand the resources and select those that are useful to the research:

- *Understand the resources and limitations of the ISS.* [NASA, 1998] and [NASA, 2000b] provide substantial information on the resources available for research aboard the ISS; the finding from these references are summarized in Chapter 2.
- *Determine the needs of the research.* Based on the mission objectives and the other principles, the designer can determine the general operational processes of the facilities which will operate aboard the ISS and at Mission Control. The general operations provide insight into which resources benefit the mission.
- *Realize which resources do not provide a benefit to the research.* While the goal is to maximize the value obtained from the resources aboard the ISS, there may be cases in which utilizing a resource presents more challenges than benefits to the science. If a resource cannot provide positive value to the mission, the scientist must realize this early in the design process, and decide not to use that resource.

A set of guidelines based on value models presented in [Cook, 1997] was developed to help better understand the availability of resources aboard the ISS. The value curves presented in [Cook, 1997] utilize Taguchi functions which consist of Nominal is Better (NIB), Larger is Best (LIB), and Smaller is Best (SIB) models. The original Taguchi NIB functions are centered around a central value (the curves are symmetrical about the best value), which is not necessarily the case for the ISS resources. Therefore, these functions have been adapted to generate value curves which better represent the availability of resources aboard the station.

The Taguchi method normalizes the value obtained from a resource between 0 (least value) and 1 (most value). The parameters which define the shape of the value curves represent the availability of the ISS resources identified in Chapter 2 based on the information provided in [Hagopian, 1998], [NASA, 1998], [NASA, 2000b], [NRC, 2000] and [Durham, 2004]. The curves represent the ability of the ISS to provide that resource; if the resource can benefit the laboratory, then the amount of resource to utilize should be consistent with the value curve.

Table 5.3 summarizes the findings of the availability of the special ISS resources and Figure 5.6 presents the resulting value curves. The table indicates the type of modified Taguchi value curve used for each resource (NIB, LIB, or SIB); in one case a value curve is not appropriate, and Yes/No (Y/N) values are recommended. The *base* value indicates a starting value for the design. The *critical* value represents the point at which the ISS can no longer provide that resource, and further usage would negatively affect other research operations aboard the station; therefore, experiments should not use these amounts of resources. The *ideal* column indicates the point at which the ISS resources can be best shared among research facilities. The determination of these values was based on the following information about each resource:

- **Crew -** Human presence is one of the most important characteristics that separate ISS operations from standalone spacecraft. Therefore, the value of this resource is based on a nominal is better (NIB) curve - the astronauts

TABLE 5.3 Summary of ISS special resources

Resource	Type	Unit	Base	Critical	Ideal
Crew	NIB	hours/month	10	144	18
Power Sources - usage	SIB	kW	4	20	2
- percentage	LIB	%	50	0	100
Telemetry	NIB	Kbps	7	1000	100
Duration	NIB	months	6	120	6
Benign Env't / Atmosphere	Y/N	-	[1 0]	-	-

should perform tasks for the experiment, but only for a limited time. The parameters were obtained as follows:

- The review of Expedition 6, with three astronauts aboard the ISS, provided a total of 2160 hours per month of astronaut time. According to NASA approximately 10% of astronaut time can be used for science experiments Therefore there are a total of 216 hours per month available for science. Review of ISS science up to date reveals that there are approximately a dozen experiments in the ISS at any one time, leaving approximately 18 hours per experiment per month as an ideal value

The recommended NIB value curve drops sharply as less than eight hours per month of astronaut time are used, since it results in a waste of an important resource of the ISS. If more than 50 hours per month are used, the facility starts to reduce the available resources to other projects, also reducing its value.

- **Power sources -** The value of the ISS with respect to power is presented using two measures:
  - *Minimize total power.* The curve for absolute power utilization is a smaller-is-better (SIB) curve since the total power consumed should be minimized. The ISS will provide up to 46kW of electrical power; each locker provides an average of 2kW of power, making that the ideal value. The base value of 4kW is based on an average of 12 experiments present in the station at any one point. The critical value of 20kW is based on the maximum available power using special resources.
  - *Maximize power fraction used from ISS.* This is a larger-is-better (LIB) measure - a facility should maximize the percentage of power used from the ISS with respect to the total facility power.
- **Telemetry upload/download -** The ISS communications system value is based on the NIB value curve. The available data bandwidth is 1.5Mbps, with expected upgrades to 15Mbps in the next couple of years. Given that

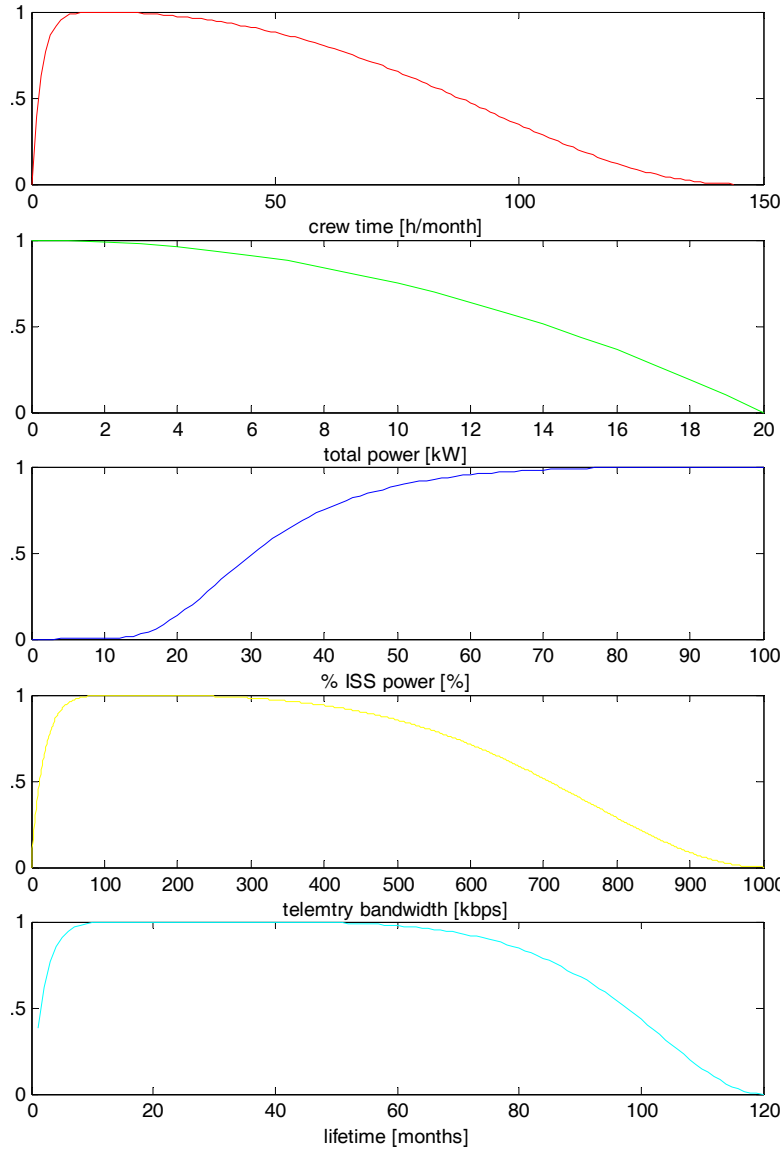


Figure 5.6 Value curves for ISS unique resources

there are approximately a dozen experiments in the ISS, the current ideal value is for each project to use approximately 100Kbps of bandwidth on average.

- **Long-term experimentation** - The parameters for long term experimentation take into account the presence of astronauts in a schedule of every six months and an ISS life expectancy of 15 years. An NIB curve is recommended for long-term experimentation. The ideal value for the longevity of an experiment is one expedition (six months), due to training and other operational constraints. The other limit is the station's life expectancy of approxi-

mately 10 years from now. The value between six months and several years of operation is almost the same, since the limitations from the ISS perspective lie solely on astronaut training and availability. The value drops for shorter periods, since the expenses in the deployment of a project to the ISS should be amortized over longer operations. The value curve is not an LIB because experiments that reside too long in the ISS restrict the availability of resources, preventing other facilities from operating and limiting the science conducted aboard the station.

- **Benign Environment / Atmosphere -** The value of utilizing the benign environment of the ISS does not follow a Continuos curve; rather, it is a binary yes/no determination. A scientist should consider the following points to ensure the laboratory's facilities benefit from this resource:
  - The monitoring capabilities of the ISS should be utilized to safeguard the facilities aboard the station.
  - Astronauts should be able to prevent and/or repair damage to a facility to ensure long-term operations.
  - Facilities aboard the ISS should utilize the standard interfaces of the ISS available for research, including the availability of structural mounting points, supply lines, and data connections.
  - Facilities which operate inside the ISS should reduce their cost by utilizing standard components without special treatments to account for the space environment.

Principle of Incremental Technology Maturation

The achievement of technology maturation in an experimental setup suggests that the operational environment of the experiment resembles the final application enough so that the technology is considered reliable for operations in the next level of technology readiness. Therefore, in order to determine how well a design meets this principle, one must examine how far into the TRLs we can go using this facility and how much the cost and risk can be reduced by developing technologies in the ISS.

To measure how far into the TRLs a technology can be advanced, we use the tests for each TRLs that apply to operations in the ISS: TRL 5, TRL 6, and TRL 7. An ISS experiment should provide advancement at least to TRL 5, preferably TRL 6. Some facilities may even provide TRL 7 advancements if the final technology is compatible with the ISS envi-

ronment. To measure how far into the TRL's a technology can be advanced, we use the established tests for each TRL [Lindensmith, 2003]:

TRL 5:

1. The "relevant environment" is fully defined.
2. The technology advance has been tested in its "relevant environment" throughout a range of operating points that represents the full range of operating points similar to those to which the technology advance would be exposed during qualification testing for an operational mission.
3. Analytical models of the technology advance replicate the performance of the technology advance operating in the "relevant environment"
4. Analytical predictions of the performance of the technology advance in a prototype or flight-like configuration have been made.

TRL 6:

1. The technology advance is incorporated in an operational model or prototype similar to the packaging and design needed for use on an operational spacecraft.
2. The system/subsystem model or prototype has been tested in its "relevant environment" throughout a range of operating points that represents the full range of operating points similar to those to which the technology advance would be exposed during qualification testing for an operational mission.
3. Analytical models of the function and performance of the system/subsystem model or prototype, throughout its operating region, in its most stressful environment, have been validated empirically.
4. The focus of testing and modeling has shifted from understanding the function and performance of the technology advance to examining the effect of packaging and design for flight and the effect of interfaces on that function and performance in its most stressful environment.

TRL 7:

TRL 7 requires both an actual system prototype and its demonstration in a space environment. The prototype should be at the same scale as the planned operational system and its operation must take place in space.

The use of the ISS should allow the challenges of microgravity research to be reduced by by providing a representative environment to mature technology. The researcher must determine whether obtaining this advance in technology by using the ISS provides an advantage, especially in terms of the risk and cost involved in a space mission. This concept is illustrated in Figure 5.7. The figure shows that there is a certain risk and cost for each step towards developing new missions. The cost/risk can be that incurred in the deployment of a technology in ground-based facilities followed by one or more flight programs; this cost is represented by $\$_3$ and $Risk_3$ in the figure. The risk/cost can also be split by using the ISS as an intermediate step, with one or more steps taken aboard the ISS ($\$_1$, $Risk_1$) before deployment of the space mission ($\$_2$, $Risk_2$). Incremental technology maturation should allow the total cost and risk of a mission to be lower by using the ISS as an intermediate laboratory to develop technologies incrementally.

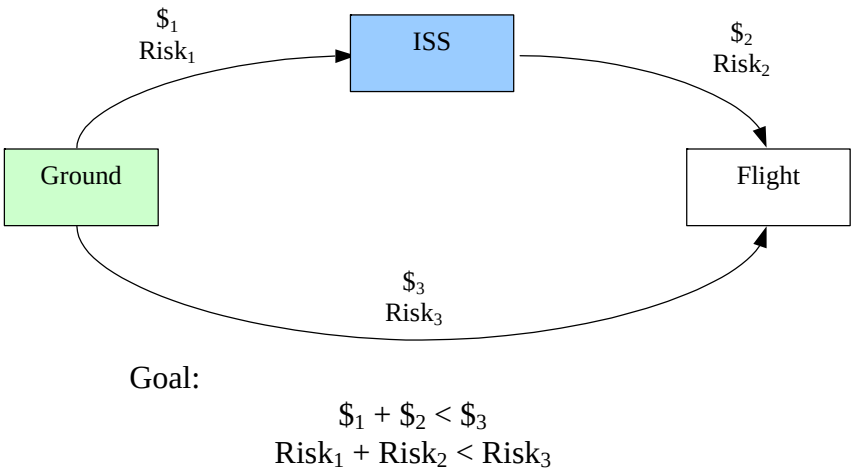


Figure 5.7 Two paths to flight operations

Reducing risk and cost are not independent of each other, but they are not necessarily proportional to each other. To reduce risk the cost may need to go up, but the reduced risk may result in a lower program cost if problems are prevented. Therefore, the following cost function accounts for the fact that lowering risk can reduce total program costs even at a higher individual cost, while at the same time penalizing extreme cost increments. It

quantifies the risk of a mission in terms of the cost of mission failure, the probability of failure, and weigh it by the possible advances from the mission in terms of TRL levels:

$$R = \frac{\$_{TRL} \cdot \rho}{w(TRL_{\#})} \tag{5.2}$$

where

$\$_{TRL}$ = cost of the facility to allow a specific TRL advance

ρ = probability of failure of the mission such that a TRL advance could no longer take place (if a facility allows multiple TRL advances, each of them should be considered independently)

$w(TRL_{\#})$ = the weight assigned to the TRL numbers 5, 6, or 7. Achieving each TRL brings the technology closer to maturation at different levels, therefore the weight accounts for the level of technology advancement:

$w(5) = 1$

$w(6) = 1.5$

$w(7) = 3$

The resulting "risk" is presented in units of cost, such that it can be combined easily with the total cost of the mission to determine the cost/risk value for a step:

$$J(\$, R) = \$ \cdot \left(\frac{\$ + R}{\$ - R}\right) \tag{5.3}$$

where

$\$$= total cost of the facility

R = total risk/cost of the facility

If a laboratory allows maturation of a technology for multiple TRL's, the cost of achieving each level should be accounted independently, and then the total cost added together. For example, if a laboratory allows a technology to mature through both TRL 5 and 6, then its total cost should have two parts: first, the cost to deploy the laboratory for operations

aboard the ISS and costs related to achieving TRL 5; second, the operational costs (without deployment costs, such as launch, if none are incurred) for achieving TRL 6.

5.10.3 Step 3 - Refine Design

Once the primary functional requirements of the system are defined in Step 2, Step 3 identifies possibilities to get added value from the laboratory's facilities by providing guidelines to decide on the level of modularity in the design and ensure that remote operations take place efficiently. The principles of *Focused Modularity* and *Remote Operations & Usability* apply in this step.

Principle of Focused Modularity

The Principle of Focused Modularity calls for the identification of parts of a facility which are generic equipment that could be utilized by other facilities with similar needs. Of outmost importance in the application of this principle is to remember that the creation of a modular system must not deviate the project from its original mission objective. Modular systems must be identified after the initial design of the facility has been created to meet the science goals of the mission, not designed a-priori as part of the mission objectives.

In some sense it is obvious whether a system is modular or not; a modular system has multiple components that can be interchanged to create different configurations. The following criteria, helps to further identify the applicability of modularity to parts of the system; it also helps in the design process, to ensure that modularity is thought off as an integral part of the system while ensuring that the science goals are met:

- **Inter-disciplinary use** - if the component can be used in different disciplines within the field it should be generic equipment
- **Reconfiguration** - if the component easily allows the experiment to change the general area of study of the experiment, while supporting all other functions, it should be generic equipment
- **Obsolescence** - only those components that are not expected to be obsolete by the time of re-use should be made generic

- **Life-time** - only those components whose life-time is over the anticipated time to re-use should be made generic
- **Cost amortization** - if the component cost is likely to be amortized by future use in different experiments it should be generic equipment
- **Maintain original goals** - the immediate research should not be compromised by making a specific equipment generic

The goal of the criteria helps to ensure that the component will not change the original goals, not be obsolete, and will be fully functional at the time of re-use. Further, it guides towards the ability of generic equipment to add value to the facility by ensuring the component will expand the functionality of the system. Table 5.4 presents a truth table that indicates whether equipment should be generic or not: based on this criteria.

TABLE 5.4 Modularity criteria truth table

Inter-discipline	Reconfigure	Obsolete	Life-limited	Cost Amortize	Original Goals	Make Modular
x	x	x	x	x	0	0
x	x	0	x	x	x	0
x	x	x	0	x	x	0
1	x	1	1	x	1	1
x	1	1	1	x	1	1
x	x	1	1	1	1	1

Key: 0 = no, the criteria is not met
1 = yes, the criteria is met
x = irrelevant

Principle of Remote Operation and Usability

A successful laboratory exhibits the following characteristics to support both operators and research scientists by creating the necessary facilities both at the ISS and the ground-based location of the scientists:

- **Operators**
  - Provides the operator with the necessary controls to conduct research efficiently
    - Available controls must ensure the operator can actuate or command the facility in every way necessary to perform the experiments
    - Extraneous controls should not be present to minimize the distractions of the operator.
    - Repetitive tasks not directly related to conducting experiments and obtaining relevant data should be minimized.
    - The controls must meet the requirements for interfaces as defined by the ISS boards and to account for human abilities. The controls must consider the user of tools such as quickening, predictors, and virtual displays in the case where the operator must perform real-time maneuvers or commands.
    - The operator must always feel safe to stop an experiment.
  - Ensures that the data is safely transferred to the scientists regardless of operator actions.
    - A clear data-path must be defined prior to operations so that data is not lost or delayed inadvertently
  - Presents relevant information to the operator for successful run of experiments.
    - Allows operators to conduct full cycles of the inner-most iterative research process loop presented in Figure 5.1: they can repeat experiments to obtain substantial data without the intervention of the scientists.
    - Provides the operator with enough information to make decisions about the course of the experiment while minimizing the risk to the research.
    - Reduces the dependency of real-time audio/video contact between the scientist and the operator.
    - Presents only the necessary information for the successful run of experiments, but does not burden the operator with data not essential to the conduction of experiments.
  - Enables operators to provide feedback to the research scientists from their observations in the operational environment.
  - Provides methods to conduct real-time communications with the research scientists under pre-specified circumstances.

  - Does not require real-time communications under standard operating conditions.

- **Research Scientists**
  - Minimizes the efforts and time to collect the data obtained in the remote environment.
    - The scientist(s) must know in advance the expected delays to obtain the data
    - The structure of the data must be fully documented to allow for quick data parsing when the data is received
    - Data can be separated into two levels of importance; this fact must be especially considered if the experiment requires a high amount of data
      - Critical data needed to continue experiments after one set of operations must reach the scientist efficiently and in a short period as compared to the iteration period
      - Support data that is not essential for continued operations can be transferred in a slower fashion
    - Provides data management tools to extract the data relevant to the scientist with minimal overhead.
  - Allows scientists to upload information and data to the operator
    - The scientists must be able to contact the operator to ensure correct operations of the facilities aboard the ISS.
    - After reviewing the data and modifying their hypothesis, the scientists must have efficient interfaces to upload new experiments to the ISS.
  - Provides real-time communications under pre-specific circumstances but does not require real-time communications for the scientist to evaluate the progress of the research.
  - Allows scientists to predict results and compare them with the collected data.
    - Ground based facilities should be made readily available to scientists so that they can predict the results of experiments conducted aboard the ISS.
      - These facilities can include simulations, 2D testing environments, and the use of the microgravity testing facilities presented in Chapter 1.
    - The facilities should provide methods to compare predicted results and results from the data obtained aboard the ISS.

Because the interfaces for the operator and the scientist serve a different purpose, there is no need for both interfaces to be the same; rather, the facility design should consider each interface separately to meet their specific needs. Further, a laboratory may utilize several facilities to help the scientist, each with its own interface. (The operator aboard the ISS should only be required to use on interface.)

5.10.4 Step 4 - Review Requirements and Design

The final step before starting a new iteration (or concluding the functional requirement identification process) is to ensure that none of the previous steps creates requirements which drive the system outside of its constraints and/or create conflicts between the principles. The *Principle of Requirements Balance* is utilized in this step.

Principle of Requirements Balance

This principle exists to incorporate two important concepts into the design of a facility beyond those presented in the previous principles: the need to maintain a missions within its constraints; and to ensure that no single requirement is driving the design of a laboratory, but rather there is a balance in the design. Therefore, this principle necessitates that the design requirements for the laboratory be specified before it can be applied.

Developing requirements is an iterative process just like any other system design problem, therefore to meet this principle the scientist is expected to iterate on the requirements of the other principles and then balance them. The other principles should be evaluated first, so as to develop a set of basic requirements for the facility. Using the requirements created from the other principles, this principle calls for the balance of effort into each of the other principles.

Once the first iteration of a design achieves a clear set of requirements, the first step is to determine which are hard requirements and which are soft *decirements*:

- *Hard requirements* are usually set at the start of a project to ensure that the science objectives are met. Hard requirements are essential for the success of

a mission. These requirements are mostly quantitative, and their values seldom change after the mission has been defined. When these requirements are qualitative, they clearly define a feature or characteristic that must be present in the mission for its success.

- *Soft 'decirements'* are features desired by the scientists but which are not essential to the success of the mission. These requirements can add value to the mission, but usually due to secondary objectives. Their realization usually occurs after the hard requirements have been set and a scientists sees other possibilities beyond the primary mission objectives. Soft requirements are not usually quantitative, but rather describe another feature desired in the mission. These soft requirements should be treated carefully, as they are the usual source of *requirements creep.*

Once the requirements have been identified as hard and soft, this principle calls to ensure that the majority are hard requirements. If the majority of the requirements are soft, then the scientist must review the design by restarting the design process described in Figure 5.3 on page 225.

Once a clear set of hard requirements drives the design, one can improve on the measure of requirements balance by taking into consideration the effort put into meeting each of the other principles. This effort should be a combination of quantitative facts of the facility (e.g. cost, personnel, design time, etc.) as well as a subjective evaluation of the scientist (e.g. expected research value, expected time to maturation, etc.). The scientist should collect as much data as possible on the proposed designs to obtain reasonable expected effort to satisfy each requirement and assign each requirement a value representative of the effort.

This principle does not call for all the requirements to be perfectly balanced or to necessarily minimize the soft requirements. A project with balanced requirements will exhibit limited variations between the efforts to meet each requirement. This principle calls for the scientist to proactively pursue a realistic justification for each requirement and to ensure that a substantial part of the effort into the development of the facility goes towards clearly defined needs.

5.11 ISS NGO Evaluation Framework

The call for the creation of a Non-Governmental Organization (NGO) to institutionalize research aboard the ISS creates a unique scenario in the evaluation of programs that are to be operated aboard the station. Among the driving principles in the recommendation of the NRC is that:

> "Basic and applied scientific and engineering users should be selected on the basis of their scientific and technical merit, as determined by peer review." [NRC, 1999]

Current procedures at NASA separate the safety, training, and funding/selection processes; rarely do these processes take into account their effects on the other. The NGO scientists who select projects for the space station are challenged with the need to consider all parts of a project in its ability to successfully operate aboard the ISS. In order to concentrate on the scientific and technical merit of a mission, NGO scientists will need to fully understand the safety and operational limitations of the ISS. When reviewing a mission proposal, the technical limitations of the ISS should not immediately be considered something to prevent a program with high scientific value from taking place. The NGO scientists must care about the success of a mission as something that benefits the ISS.

The design principles provide important guidelines to allow and NGO scientist to determine the ability of a mission to successfully mature space technologies aboard the ISS. In arriving to this determination, the NGO scientist will concentrate on the following points:

- Correct utilization of the ISS
- Technology advancement
- Mission probability of success
- Mission scope

The following sections present a framework for the ISS evaluator to use in determining the correct application of the design principles in a mission design, taking into account that the evaluator is likely to only have high-level knowledge of the project.[1]

Principle of Iterative Research

The evaluation of iterative research must first consider the need for the experiment to allow iterations aboard the ISS in order for the technology to mature. In some cases a single experiment may be all that is needed, and in those case the evaluation should not penalize the experiment. In the cases where the ISS reviewer determines iterations would benefit the research, then they must also review the proposed facility to ensure it can conduct successful iterations. The ISS evaluator must be able to determine from the proposal that a facility will allow enough iterations to achieve technology maturation.

The following questions provide the insight necessary to determine if a facility supports the iterative research process:

1. Does the experiment collect the data necessary to support or refute the hypothesis?
2. Do the operational plans of the facility provide sufficient flexibility for efficient iterations?

3a. Can the facility perform multiple experiment runs with repeatability and reliability?

3b. Can the facility be reconfigured while in the ISS in such a way to provide new meaningful results and/or reflect changes in the hypothesis?

A positive answer to the first question is essential since the iterative research process requires the ability to collect sufficient data to validate or refute the hypothesis. The second question requires that the facility operations allow enough time for scientists to examine the data and present meaningful results. Note that the question does not set a timeframe for the data analysis, but the requirement that the time exists. The third question

1. The *Principle of Requirements Balance* is not used by the NGO evaluators. The balance of requirements is directly related to the design of the laboratory; it does not directly affect the effective use of the ISS nor a project's suitability to operate aboard the station. When a design change occurs due to this process, the research scientist should communicate that to the ISS evaluators when addressing the specific principle where the change occurred.

addresses the ability of a laboratory to close at least one of the iterative research loops. Part 3a is the ability to close the first loop of figure Figure 5.1 on page 204; part 3b addresses the possibility to close the second and third loops.

Principle of Enabling a Field of Study

The ISS evaluator should be able to readily identify the space technology to be matured and the selected field of study that the laboratory will cover. It should be clear how research on that field will directly allow maturation of the space technology. The evaluator must also be able to identify the specific areas of study which the laboratory enables to be researched. The areas of study must be complementary to other science already conducted aboard the space station.

Upon identifying the areas, the evaluator should make their own determination on the need to support multiple scientists in order for the laboratory to be successful. The determination of the ISS evaluator should be the same as that proposed by the research scientist, otherwise the proposal should be returned for review.

If multiple scientists are to be supported, then the ISS evaluator must determine the ability of the laboratory to successfully host them. The proposal should include the development of facilities both aboard the ISS and ground-based to support the scientists. Specifically, the ISS evaluator should look for:

- The existence of efficient data paths for the transfer of data to/from the ISS and multiple scientists.
- The ability of scientist to analyze the data in their home facilities.
- Flexible operations plans for scientists to conduct experiments within the limits of the ISS.
- The need to reconfigure hardware and/or software in the ISS based facility and the existence of support mechanism to allow said reconfiguration.

The ISS evaluator is concerned with the success of the mission, but must also ensure that a mission does not create undue burden upon the ISS program by not having the necessary facilities to support multiple scientists. The research scientist who proposes a mission

must always provide the means to distribute the data to and from the multiple scientists, while the ISS staff must ensure the data is readily available to the scientist in charge of the laboratory. Similarly, the need to change operational plans and reconfigure the facilities aboard the ISS should be determined by the scientist, and only that scientists should communicate the changes to the ISS staff.

Principle of Optimized Utilization

The ISS reviewer should give priority to the use of the special resources of the ISS over the needs of the project. The research scientists should present their model of the cost/benefit of the resources, allowing the ISS staff to better understand why resources are used (or not). As the principle prescribes, the goal is to optimize the use of resources, not maximize them; therefore, a good proposal will clearly define when the use of a unique ISS resource has a negative effect on the proposal. If a valuable resource is not used at all, but the ISS evaluator determines the project could make use of that resource, the project should be sent back for review by the researcher. In the case where very few or none of the special ISS resources are used by a project, the ISS reviewer may recommend that the project is better suited for operation as an independent satellite.

Principle of Focused Modularity

The evaluation of modularity from the ISS perspective consists solely on what the project provides. It should be expected that the scientist already made the trade-off between what should be modular or not. The ISS staff may identify further modular systems and recommend that the project be re-designed if needed. A modular project should receive higher priority, especially if its modular items can benefit a large number of scientists in the future.

Principle of Remote Operation & Usability

The ISS evaluator will concentrate on the operators' point of view. The mission’s ability to succeed, given the requested data transfer capacities and real-time interaction with the crew, must also be considered, since the ISS evaluator must care about the success of the

project as well. Therefore, the ISS evaluator must see the following characteristics in the laboratory:

- The operator has the necessary interfaces to control the facilities aboard the ISS in an efficient and safe manner.
- The operator is presented the information necessary to successfully evaluate experiments.
- The operator can provide feedback to the research scientists from their observations in the operational environment.
- A clear data path has been established for the download and upload of data.
- The need for real-time communications has been clearly defined.

Principle of Incremental Technology Maturation

Using the guidelines for the TRL levels, the ISS staff must evaluate the level of the technology maturation the project provides. Balancing the need for maturation with the other principles (such as a wide field of study), the ISS reviewer should give priority to those projects that provide the most technology maturation. The evaluation must take into account the project's ability to succeed in achieving that level of maturation.

5.12 Summary

The lessons learned in the development of the SPHERES Laboratory for Distributed Satellite Systems (Chapter 4) following the guidelines of the MIT SSL Laboratory Design Philosophy (Chapter 3) for use aboard the International Space Station (Chapter 2) resulted in the development of seven design principles for microgravity laboratories for space technology maturation. These seven principles are:

- *Principle of Iterative Research* - enable scientists to conduct iterative research through repetition of experiments to obtain the necessary data to support or refute a hypothesis; provide the capability for scientists to analyze that data and modify their theories on a flexible schedule, and allow reconfiguration of the facilities to allow for changes in experiments and hypothesis.
- *Principle of Enabling a Field of Study* - a laboratory allows research in a field of study, which consists of multiple research areas. To enable the study

of a field, it is almost always true that multiple scientists will participate in the mission. Therefore, to enable a field of study a laboratory must provide the tools necessary to support multiple scientists: the ability for scientists to create models and analyze data in their home location; simple operational interfaces; and efficient data transfer mechanisms.

- *Principle of Optimized Utilization* - the ISS provides several special resources not available in any other space research environment: crew, power, long-term experimentation, and a benign environment/atmosphere. Successful laboratories must use these resources effectively, with the idea tat the use of the resources adds value to the mission, rather than being a cost.
- *Principle of Focused Modularity* - the facilities of a laboratory almost always include common parts that can be used by a wide range of applications within the field of study of the laboratory. Those parts, the generic equipment, should be identified and designed in a modular fashion so that they can be utilized by as yet unforeseen research.
- *Principle of Remote Operation & Usability* - operations aboard the ISS occur in a remote environment where it is practically impossible for the research scientist to be present in the operational environment. Therefore, it is essential that the operators have the necessary tools and information to conduct effective runs of experiments, while the scientists have efficient access to data obtained from the experiments for analysis. Ultimately, the operator should become a virtual extension of the scientists aboard the ISS.
- *Principle of Incremental Technology Maturation* - the ISS provides a representative space environment for a large number of missions, capable of pushing the TRL of a technology between levels TRL 5 and TRL 7. Utilizing the ISS should achieve this technology levels with the risk and cost increasing incrementally, without steep jumps as the technology level increases. Successful use of the ISS should allow technology maturation with a lower cost and risk than deployment of the mission directly from ground-based tests to the space environment.
- *Principle of Requirements Balance* - the previous principles create a wide range of functional requirements. For a laboratory to succeed, these requirements must be balanced, ensuring that the hard requirements, which directly affect the ability to succeed in the mission, drive the mission efforts. Soft requirements, desired features not directly affecting the success of the mission, should only be implemented when they do not cause the mission to break its constraints and do not contradict any hard requirements.

Two frameworks are presented for the use of these principles: a design framework for research scientists who will develop new space technology laboratories, and an evaluation

framework for members of a proposed NGO that will manage research activities aboard the ISS. The design framework provides scientists with guidelines to determine the functional requirements of the laboratory's facilities (both ground-based and aboard the ISS). The evaluation framework presents guidelines for an NGO scientist to determine the effective use of the ISS while taking into account the success of the mission and the achievement of technology maturation.

The next chapter presents the results obtained so far with the SPHERES testbed and uses the design framework presented in this chapter. The chapter evaluates the success of SPHERES in fulfilling the design principles, even if it was designed prior to their development.

Chapter 6

ASSESSMENT OF SPHERES

This chapter starts by presenting an overview of the programs supported by SPHERES and the results obtained to date in several operational environments. Next, the chapter uses the design framework presented in Chapter 5 to make an assessment of the design of SPHERES with respect to the microgravity laboratory design principles. Although the framework is applied to an existing design, the application of the design framework to the SPHERES testbed illustrates the process which would take place in iterating the design through one full cycle of the design framework. It demonstrates the ability of the framework to capture all the features expected in a successful microgravity laboratory by identifying issues not considered in the initial design. Lastly, the evaluation framework is applied to the SPHERES testbed. The evaluation provides insight into how future ISS evaluators must consider the success of a mission and balance it with the need to utilize the ISS correctly.

6.1 SPHERES Results to Date

SPHERES satellites have operated continuously since the Spring of 2000. The prototype satellites were designed and built between the Spring of 1999 to the Spring of 2000. They were used to conduct proof-of-concept and initial research from the Spring of 2000 to the Summer of 2002, at which point the prototype units were retired. The flight units were designed and built from the Fall of 2000 to the Spring of 2002, and are currently in opera-

tion. The following sections present the current programs supported by the SPHERES laboratory, future programs expected to take place in the short term, and results obtained in the three operational environments currently supported.

6.1.1 Current Programs

This section presents overviews of the three programs currently supported by SPHERES at the MIT-SSL. These three programs include supporting guest researchers from NASA Ames to implement Mass Property Identification algorithms onboard the SPHERES testbed, algorithm development for Autonomous Spacecraft Rendezvous and Docking funded by DARPA, and spacecraft formation flight work in support of the Terrestrial Planet Finder mission. Algorithms from these programs are scheduled to be tested during the first SPHERES flight onboard the ISS; they do not require additional hardware or payload development, allowing the algorithms to be tested upon deployment aboard the station.

Mass Property Identification

The idea of using a characterized model of a system to augment a controller becomes much more powerful if one can perform on-line real-time characterizations. This method allows the use of changing system parameters to be tracked (e.g., center of mass and moment of inertia due to fuel depletion or docking of two spacecraft), thus allowing for better controller performance. The identification of these parameters using only gyroscope measurements is proposed in [Wilson, 2002]. Online mass property identification algorithms have been implemented and tested at MIT-SSL and aboard the RGA (KC-135). The first set of algorithms for testing onboard the ISS has been successfully implemented on the ground-based facilities. Figure 6.1 shows an example of estimating the z-axis inertia of a satellite when it is attached to the air carriage during a test session performed at the MIT-SSL. Future research includes updated filter coefficients for determining angular acceleration, using accelerometer data to improve the identification, and combining it with other autonomy algorithms such as thruster Fault Detection Identification and Recovery (FDIR).

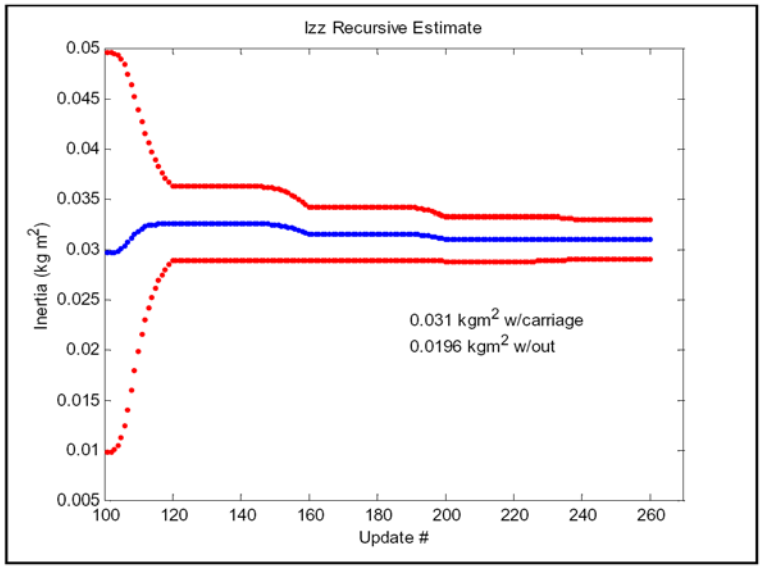


Figure 6.1 Z-axis inertia estimate from ground-based tests

Autonomous Rendezvous and Docking

The ultimate goal of the SPHERES ARD research, supported by the DARPA Orbital Express program [Shoemaker, 2004], is to develop a control architecture consisting of various algorithms that will enable safe and fuel efficient docking of a thruster based spacecraft with a free tumbling target in presence of obstacles and contingencies. Three classes of algorithms have been developed: metrology, control and autonomy. Metrology class algorithms consist of a series of extended Kalman filters that derive the state vector from the different sensor suites available for spacecraft. The control class algorithms include path planning [Hablani, 2001] as well as close-loop control algorithms. A series of PD controllers coupled with a pulse-width modulator control the attitude and the lateral alignment during the approach. Figure 6.2 shows sample results of this approach. Autonomy algorithms are used to determine the mode of operation (type of docking and phase), as well as executing the plan generated by the control class algorithms [Nolet, 2004].

Future work in this program focuses on the integration of optimal path planning algorithms that account for constraints such as obstacle avoidance and plume impingement

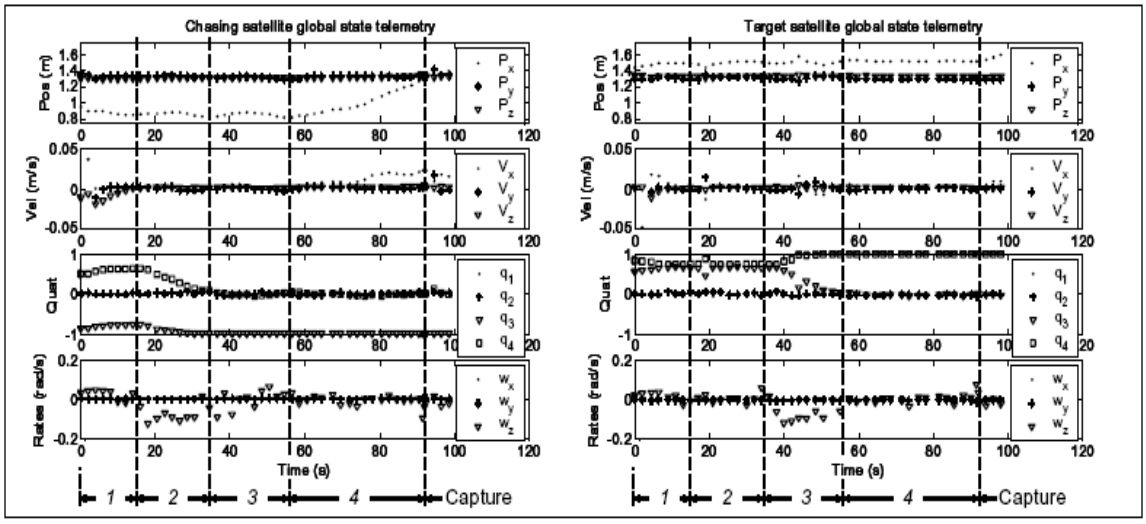


Figure 6.2 Sample results of docking algorithms at the MIT SSL

using techniques such as Model Predictive Control and parametric programming [Bemporad, 2002]. Integration of FDIR algorithms will also be of interest [Wilson, 2003].

Terrestrial Planet Finder Multiple Spacecraft Maneuvers

The TPF Mission [Beichman, 1999] will support a long baseline separated interferometer for space observation. The coordination between the spacecraft in such a system is crucial. To this end, the MIT-SSL, under the sponsorship of NASA JPL, has developed and tested algorithms for several key TPF maneuvers on the RGA and also on the MSFC flat floor facility. These key TPF maneuvers include:

- *lost in space* - the spacecraft in the array are to determine their orientations with respect to each immediately after deployment
- *array spin-up* - the array is spun up to the desired rotation rate
- *array rotation* - continuous control actuation will be required to maintain the separations between the spacecraft
- *array re-sizing* - the array size is tuned to survey the different extra-solar systems
- *array re-target* - the most complicated maneuver where the line-of-sight of the array is changed during capture to allow for different systems to be surveyed without having to stop the entire array

To date, SPHERES has been used to demonstrate a limited version of the lost-in-space maneuver, array spin-up, array rotation and array re-sizing maneuvers; Figure 6.3 shows a five satellite setup ready for tests at MSFC. The array re-target maneuver has yet to be tested due to the limited zero-gravity period currently available. Once array maneuvers are successful, plans call to add an optical pointing payload and develop multi-staged control algorithms.

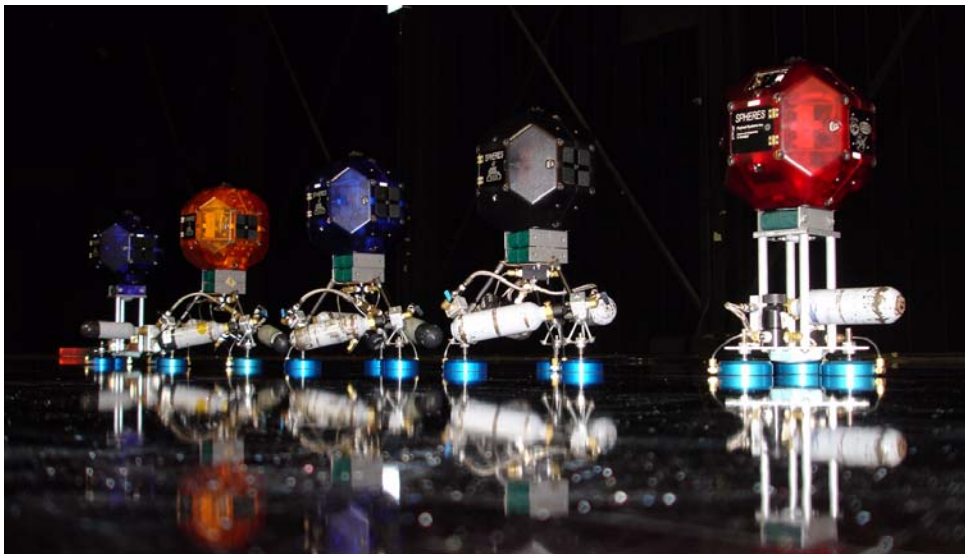


Figure 6.3 Five satellite TPF maneuvers at the MSFC Flat Floor

6.1.2 Future Programs

The SPHERES expansion port enables additional testing capabilities with the SPHERES laboratory. In most cases, only incremental payload development work is needed since the core facilities (satellites and beacons) remain onboard the ISS. This section presents three new programs for potential testing onboard the ISS. The first is the addition of a precision pointing payload to compliment the TPF maneuvers program. Second, the SPHERES team expects to study the dynamics and control of tethered spacecraft. Lastly, SPHERES will support tests of the Mars Orbit Sample Retrieval mechanism.

TPF Multi-staged Control

The TPF work described in the previous section provides only the coarse actuation of a SSI system. As the follow-on work to the TPF maneuvers demonstration, NASA JPL has funded an optical pointing payload for use with the SPHERES satellites' expansion ports, to facilitate the development of a multi-staged control testbed onboard the ISS. The ultimate goal will be to perform the TPF maneuvers through thruster actuations while maintaining precision pointing between the satellites onboard the ISS. Note that only the incremental optical pointing payload will need to be launched to the ISS to complement the core facilities.

Tethered Formation Flight

A tethered system is a trade-off between using a structurally connected interferometer, which allows for very limited baseline changes, and a separated spacecraft system where the usage of propellant can be prohibitively expensive. A tethered system is currently being considered for NASA's Sub-millimeter Probe of the Evolution of Cosmic Structure (SPECS) mission [Mather, 1998] to maneuver the sub-apertures out to separations of a kilometer, thereby achieving very high resolution. Under the guidance of NASA Goddard Spaceflight Center, the SPHERES program will be used to research tethered systems by the addition of two major components:

- tether deployment and retraction mechanism with tether tension sensors, latch plate, and momentum wheel package
- momentum wheel package

Initial tests at the MSFC Flat Floor facilities (Figure 6.4) took place in 2004 with a prototype deployment and retraction mechanism.

Mars Orbit Sample Retrieval

To obtain and analyze samples of Mars surface elements, the Mars Orbit Sample Return program (MOSR) must overcome the challenge of autonomous search, acquisition, rendezvous, and docking of the sample return spacecraft with the sample. Terminal-phase

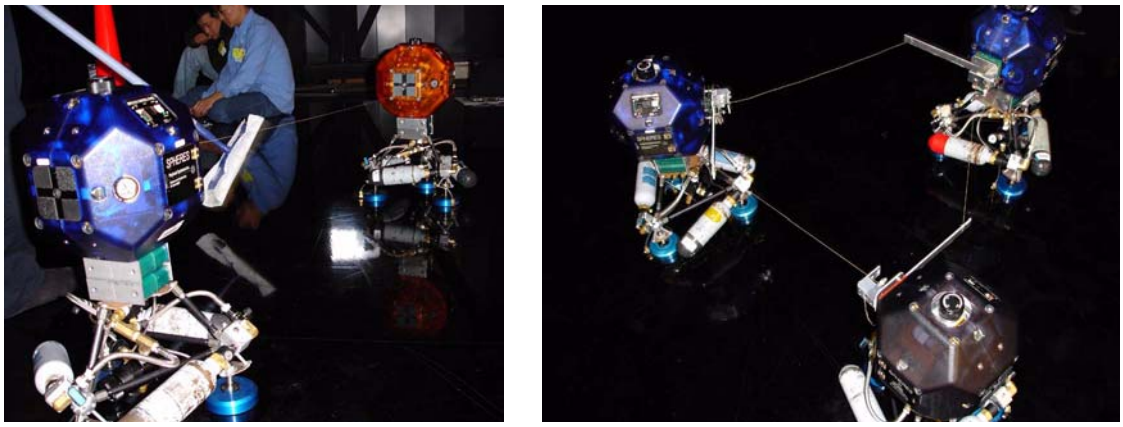


Figure 6.4 Two and three satellite tethered setups at the MSFC Flat Floor

multi-body trajectories and physical contact dynamics between the orbital sample and retrieval system can only be represented with high fidelity in a 6 DOF physical environment. Under the guidance of JPL, the SPHERES program is being utilized to test the capture mechanism of the Mars Orbit Sample Retrieval (MOSR) system (Figure 6.5). Force and torque sensors will be placed on the capture mechanism to measure the impact of the satellite on the cone as the velocity and rotation speed changes. The orbit sample in this experiment is represented by a SPHERES satellite. Since the satellite has the dimensions and mass properties similar to those expected for the final system, full scale emulation of a sample by the satellite can be achieved.

Figure 6.5 Artist's conception of MOSR aboard the ISS

6.1.3 Experimental Results

Appendix I presents the results of experiments conducted using the SPHERES laboratory at the MIT SSL, aboard the RGA, and at the MSFC Flat Floor facilities. Table 6.1 summarizes the experiments conducted with the SPHERES laboratory since 2000. The experiments included tests of formation flight and ARD control algorithms at all three locations. The RGA was used considerably to aid in the design and demonstration of the global metrology system. As the table shows, guest scientist involvement began in 2003 with the participation of NASA Ames, Goddard, and JPL staff in several reduced gravity campaigns.

TABLE 6.1 Summary of SPHERES Experimental Results

Date	Research	Location	Application	Guest Scientist
2000	F.F. Communications	SSL	DSS	
2000	F.F. Control	SSL	TPF	
Feb. 2000	Satellite Demonstration	RGA	SPHERES	
Mar. 2000	Metrology System Test F.F. Control	RGA	SPHERES DSS	
Oct. 2001	Metrology System Test Satellite System ID	RGA	SPHERES	
2002 +	Docking Control	SSL	Orbital Express (DARPA)	
Jul. 2002	Metrology System Test Docking Control	RGA	SPHERES DARPA	
2003 +	Mass ID / FDIR	SSL	Modeling	Ames
Feb. 2003	FDIR Global Frame Control	RGA	Modeling TPF	Ames
Nov. 2003	F.F. Communications F.F. Control FDIR	RGA	DSS TPF Modeling	Goddard Ames
2003 +	Tethers	SSL	SPECS	Goddard
2004 +	MOSR	SSL	Mars Sample Return	
June 2004	F.F. Control Docking	MSFC	TPF DARPA	JPL
Nov. 2004	F.F. Control Tethers	MSFC	TPF SPECS	JPL

6.2 Design Framework

Chapter 4 describes all the features of the SPHERES Laboratory for Distributed Satellite Systems which enable it to fulfill the definition of a laboratory. The previous section presents the range of research conducted with SPHERES to date; it also shows the ability of the SPHERES facilities to operate in several locations to accomplish different research goals. This information enables a thorough examination of the SPHERES Laboratory's ability to fulfill the design principles based on the design framework presented in Chapter 5 and suggest design changes if SPHERES could go through one more design iteration.

6.2.1 Step 1 - Identify a Field of Study

- *Principle of Enabling a Field of Study*

Principle of Enabling a Field of Study

At its conception, SPHERES was planned to be a testbed for the development of spacecraft docking and autonomous rendezvous algorithms. At that point, the SPHERES team identified several areas of study necessary to develop these types of algorithms:

- Metrology
- Control
- Autonomy
- Artificial Intelligence
- Communications
- Human/Machine Interfaces

These areas of study are described in Section 4.3.3.

As the design of SPHERES matured to fulfil the MIT SSL Laboratory Design Philosophy the field of study progressed from docking and rendezvous to distributed satellite systems. The areas of study supported by the laboratory should not only cover those topics which allow docking and rendezvous, but also the different configurations that comprise DSS. The SPHERES team identified the following configurations:

- Docking and rendezvous
- Formation flight
- Separated spacecraft telescopes
- Tethered spacecraft
- Sample capture

For each of these areas, the SPHERES laboratory must allow, at least, the study of the metrology, control, autonomy, and communications requirements to mature the technology.

To support this range of areas of study, SPHERES clearly needs to allow the participation of multiple scientists. Therefore, the SPHERES team created the Guest Scientist Program (Section 4.3.3.1) to provide scientists with:

- A simulation to create models of their experiments in their home locations and the ability to conduct experiments at the MIT SSL as the models mature.
- The SPHERES Core software which features a high-level applications programming interface (API) and multiple libraries to support scientists in the implementation of their algorithms.
- The ability to define their own telemetry data structures.
- A flexible schedule with continuous support by the SPHERES team.

Further, SPHERES allows full software reconfiguration (Section 4.3.4.7), which has enabled scientist to conduct research in multiple areas of study without any hardware changes (docking and rendezvous, formation flight, and sample capture on the high-level areas; metrology, control, autonomy, communications within the low-level areas). The SPHERES Expansion Port (Section 4.3.3.2) enables hardware reconfiguration. Through the use of the expansion port, SPHERES has already enabled ground-based research on docking and rendezvous with an advanced docking port, tethered spacecraft formations, and complex formation flight maneuvers. The areas of artificial intelligence, human/machine interfaces, and separated spacecraft telescopes have not had experiments at this point; their study with SPHERES will require the addition of hardware and/or creation of special software.

This information allows the calculation of the costs for the development of the SPHERES Laboratory for DSS. Table 6.2 summarizes the areas of study supported by SPHERES in two groups: high level configuration of distributed satellite systems, and low-level areas of study within each configuration. The *guests* column indicates that a guest scientists is currently conducting research on the subject or that the SPHERES team expects a guest scientist to be a primary researcher for that area. The *current* column indicates an area of study currently being researched with SPHERES. The last two columns provides information on the cost to enable each area of study within SPHERES (based on existing contracts) and or as standalone ISS projects (based on past MIT SSL projects).

TABLE 6.2 Areas of study supported by SPHERES

Area of Study	Guests	Current	SPHERES[a]	Standalone[a]
Docking and rendezvous		✓	$2.5	$2.0
Formation flight	✓	✓	$0.6	$2.0
Separated spacecraft telescopes	✓		$1.0	$4.0
Tethered spacecraft		✓	$0.6	$3.0
Sample capture	✓	✓	$1.2	$3.0
Metrology		✓	$0.0	$0.5
Control	✓	✓	$0.0	$0.5
Autonomy	✓	✓	$0.0	$0.5
Artificial Intelligence			$0.5	$2.0
Communications		✓	$0.5	$2.0
Human Machine/Interface			$1.0	$4.0

a. Costs in US $ millions

The costs to enable docking and rendezvous research represent the original cost to develop the SPHERES Laboratory of approximately $2.5m. This initial cost included the ability to test metrology, control, and autonomy algorithms. It is estimated that enabling research on each of these specific areas in a standalone project will cost at least $0.5m. The cost to support formation flight with SPHERES is covered by contracts approximating $0.6 million; but development of a standalone facility would require a complete new project to be

delivered to station; the project cost would be similar to that of SPHERES, at $2m. The development of the optical systems to model a separated telescope has been proposed at a cost of approximately $1.0m; the complexity of a standalone system would require no less investment than that used for MACE, at $4.0m. The development of expansion port items to support tethered spacecraft is done under a project funded with $0.6m; the complexity of this project is estimated between that of SPHERES and MACE, at $3.0m, due to the added hardware requirement. The sample capture system used for MOSR requires the development of the capture station, of a new satellites with a fully spherical shell, and the launch of these items to the ISS. Therefore, the cost of this system within SPHERES is based on contracts for $1.2m. The deployment of a standalone system is expected at $3.0m. SPHERES lacks the data storage capacity for successful artificial intelligence (AI) tests; therefore, it requires an investment of approximately $0.5m to develop the expansion port items to provide the increased storage space necessary to support AI. A standalone project would require no less investment than that used for SPHERES. While tests on the area of communications have already taken place with SPHERES, these tests are limited to the default hardware provided. The expansion port can be used to provide different types of communications hardware to test different technologies and protocols. This expansion would require approximately $0.5m. A standalone project would require an investment similar to SPHERES at $2.0m. The area of human/machine interfaces has not been considered for testing with SPHERES in the short term, but initial estimates require approximately $1.5m to develop expansion port hardware for the satellites as well as new interfaces for the operators. The complexity of this project as a standalone experiment would be closer to that of MACE, at $4.0m.

Figure 6.6 shows the fractional cost of SPHERES with respect to launching standalone projects to study the areas of study identified in Table 6.2 utilizing equation 5.1. The figure shows that at least five, preferably six areas of study must be covered to obtain a reasonable benefit from supporting multiple investigators in the laboratory. It is also noticeable how adding the last area of study (human/machine interfaces) adds little value, given its higher cost. The SPHERES team has demonstrated the ability to conduct science

on at least the following areas: docking and rendezvous, formation flight, tethered spacecraft, metrology, controls, autonomy, and communications. SPHERES is further expected to be used to demonstrate sample capture and separated spacecraft telescope systems. Therefore, the SPHERES laboratory allows research in a sufficient number of research areas to warrant the costs to make it a laboratory, rather than a docking and rendezvous testbed.

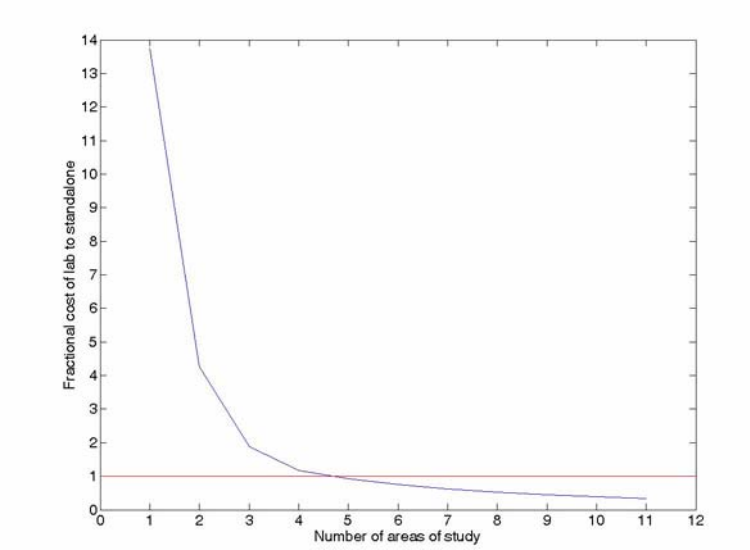


Figure 6.6 Fractional cost of enabling multiple areas of study

6.2.2 Step 2 - Identify Main Functional Requirements

- *Principle of Enabling Iterative Research*
- *Principle of Optimized Utilization*
- *Principle of Incremental Technology Maturation*

Principle of Enabling Iterative Research

The principle of iterative research is composed of three parts: development of data collection and analysis tools, enabling reconfiguration, and having a flexible operations plan. The following section describe how SPHERES fulfills these requirements.

Data Collection and Analysis Tools

Section 4.3.2.1 describes the metrology sub-system, which is used for all data collection in the satellites. The metrology sub-system provides a 6DOF IMU system with a bandwidth of 300Hz, and the precision to observe an impulse bit of the propulsion solenoids. The global metrology system, which measures the state of the satellites with respect to a reference frame, has a bandwidth of up to 2Hz with 0.5cm linear and 2.5° angular precision.

SPHERES counts with several features to ensure the integrity of data and minimize the transfer time. As explained in Section 4.3.2.3, the laptop programs (both ground-based and ISS) save all received data; data files are not corrupted if an experiment terminates unexpectedly. Further, the GSP program provides a clearly defined set of data packages as well as user-defined packages. This allows scientists to quickly identify the data necessary to perform analysis. For ISS operations, SPHERES stylizes the existing communications resources of the station to minimize data transfer times.

Enable Reconfiguration

The iterative research process presented under this principle consists of three iterative loops:

- Repetition of experiments
- Modification of experiments
- Modification of the hypothesis

This section analyzes the ability to close each of those three loops with the SPHERES laboratory.

Repetition of experiments. By following the MIT SSL Laboratory Design Philosophy, the SPHERES design considers the repetition of tests as an essential aspect of its facilities. Section 4.3.1.4 details the features of SPHERES which directly enable efficient test repetitions. The software sub-system most directly facilitates test repetitions by providing operators with simple tools to start and stop tests. Section 4.3.2.8 presents the two separate

user interfaces, each designed to simplify repetitions of tests in their respective operational environments. Section 4.3.2.7 explains test synchronizations to help guarantee initial conditions of tests with multiple units. Lastly, the ability of SPHERES to re-supply all of its consumables (Section 4.3.2.9) allows for multiple repetitions with reduced risk that a single test will deplete all available consumables.

Modification of experiments. The ability to run families of tests, explained in detail in Section 4.3.1.3, allows each operating session to test a range of algorithms, allowing multiple experiments to be conducted during each iteration. Section 4.3.4.7 presents the ability of SPHERES to change the software easily. The use of the ISS communications system (Section 4.3.1.5) to upload new experiments and the lack of NASA safety controls on software (Section 4.3.1.6) minimize the time to reconfigure the satellites. Lastly, the physical nature of SPHERES allows to easily change initial conditions. The addition of passive hardware is easily performed by using the velcro of the docking port; adding active hardware can be done via the expansion port (Section 4.3.3.2).

Modification of the hypothesis. Modification of they hypothesis implies that substantial changes can be made to the facilities of a laboratory. The principle calls for the ability to modify sensors and actuators, to enable software and hardware changes to represent new models derived by the scientists, and to allow modification of the operation plans. Software modifications can be made if the desired dynamics of the new sensors and/or actuators are within the limits of the avionics used in SPHERES (Section 4.3.4.5). Further, the SPHERES sensors and actuators can potentially be modified by using the expansion port (Section 4.3.3.2), although these changes require delivery of new hardware.

The satellites can be modified to represent new models, with certain limitations. SPHERES provides the ability to fully change the software (Section 4.3.4.7), which allows software based model to be fully modified. As presented above, the docking port and expansion port can be used to add hardware, but this will require the delivery of the

expansion items to the ISS. Further, hardware modifications are limited to the general capabilities of the satellites basic design (Section 4.2.1).

Flexible Operations Plan

SPHERES operates in a multitude of ground based facilities, all of which have demonstrated its capability to produce multiple iterations. The locations where experiments have been conducted include: the MIT SSL laboratory facilities, the KC-135 reduced gravity airplane, and the Marshal Space Flight Center flat floor facility. Research operations at the MIT SSL are described in Section 4.3.1.1; iterative loops are presented for the cases where the researcher is both on-site and off-site. These loops show the ability of SPHERES to provide a flexible operations plan for ground-based research at the MIT SSL. Scientists have the ability to determine the time they need for data analysis, while the SPHERES team minimizes the time to transfer data and update algorithms. The only hard limitation on ground-based tests at the MIT SSL are due to the limited test time of approximately 20 minutes (operation of the air carriages). Similar iterative loops can be created for the two operational environments not considered an integral part of the ISS operations, but which appeared during ground-based operations of SPHERES:

Iterative Research Utilizing the KC-135. The KC-135 operational environment (described in Appendix B) provides the ability to perform 6DOF tests with the presence of the researcher. But it is a relatively harsh environment, where test time is heavily constrained. The SPHERES operations in this platform required a pre-specified plan to be strictly followed during each test session; only one or two programs were planned for testing each day, without the ability to modify the programs. After the tests are performed, video and data analysis occurs and programs are modified in the evening, for testing the next day. Therefore, while multiple tests are performed each day in the KC-135 itself, the process has a minimum iteration period of one day. In some cases, the iterations occurred over two days, as one day was left in between for data analysis. A further limitation of the KC-135 is that tests can only be performed over a one week period, and subsequent tests, which require further sponsorship of new campaigns, are usually no less than six months

apart. The KC-135 follows the four steps of the iterative process (as presented in Figure 4.8 on page 118) as follows:

1. **Running tests** - Limited to 20 seconds; useful data of 5-10s. 60s between tests, with three 5-10 minute periods every ten parabolas.
2. **Data collection** - Data is collected in real-time or between tests within the KC-135; available to the researcher until after the flight.
3. **Data analysis and algorithm modification** - Inflexible: average time between iterations is less than 24 hours and maximum of 72 hours.
4. **Algorithm implementation and update** - Algorithms cannot be modified aboard the RGA; updating the satellites can only be performed during the three long pauses (five to ten minutes).

Figure 6.7 presents the modified iterative research process aboard the ISS. Of special note is the addition of data evaluation outside the standard loop, and the separation of the data analysis and algorithm modifications into a different location than where tests are conducted. The figure illustrates the need to maximize the science time aboard the KC-135, while leaving the data collection, analysis, and algorithm modification for a later time.

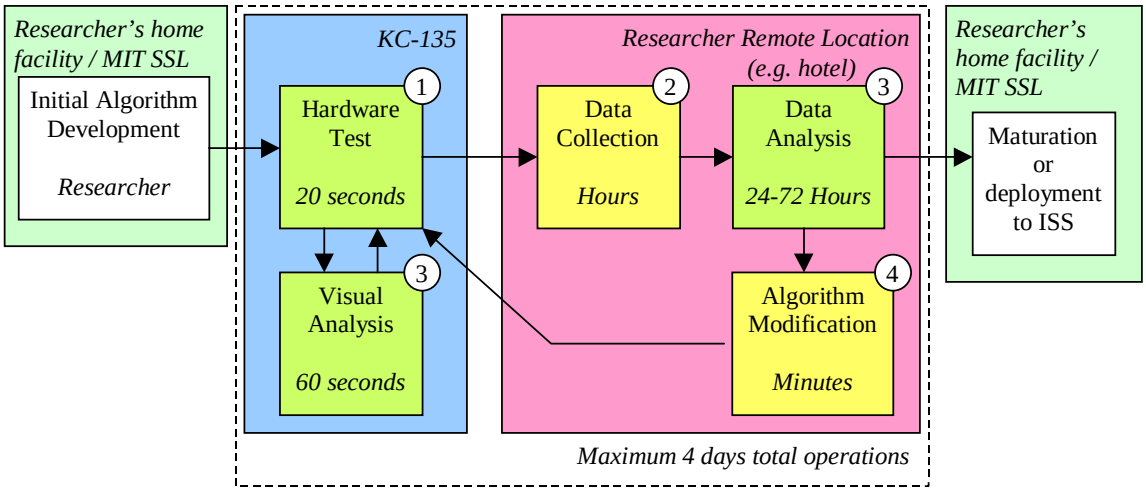


Figure 6.7 KC-135 iterative research loop

Table 6.3 summarizes the *research iterations* conducted during the five week-long campaigns at the KC-135 reduced gravity airplane. Although all experiments were repeated

multiple times (between 5 and 80 times each week), the table shows the number of research iterations after data was analyzed each night and new algorithms were uploaded for tests on a subsequent flight. The maximum number of research iterations is three; several experiments achieved this number of iterations, although the majority only had one or two iterations.

TABLE 6.3 Research iterations aboard the KC-135

Flight	Test Topics	Research Iterations
March 2000	Global System ID	1
	Global Frame Control	3
	Angular regulation (Euler vs. Quaternions)	2
	KC Frame ID	1
	Formation Flight Tests	3[a]
	Minimum Gas Turn	-
October 2001	Inertia Measurement	1
	Closed Loop Inertial Control	-
	Hardware Tests	-
	Global Frame Control	3
July/August 2002	Global Frame Control	3
	Docking	1
February 2003	1DOF System ID	3
	Global Frame Control	2
	Thruster ID	n/a
November 2003	Beacon Track	1
	Docking	2
	Lost in Space	2
	Inertia ID	3
	Distributed Control Architecture	2

a. *KC frame identification and angular regulation tests culminated in the ability to perform formation flight tests.*

Research on the KC-135 also had iterations at a different scale. The metrology system design went through three major iterations, with cycles of approximately twelve months each. These revisions were directly affected by the data and results obtained from operations aboard the RGA.

Iterative Research at the MSFC Flat Floor. A description of the facilities and benefits of the MSFC Flat Floor are presented in Appendix B. The MSFC Flat Floor environment is relatively stress free. The schedule test time is usually in terms of full days, allowing scientists to iterate on their algorithms after every test run. Scientists are not required to run one test after another. Further, the facility also allows all consumables to be replenished with ease and resupply is practically unlimited. While time is not as critical as in the case of the KC-135, the number of tests and data analysis/algorithm modification times are limited to the length of the visit to MSFC; scheduling of the facility usually requires a few months of advance notice. Lastly, tests are again limited by the air carriages ability to operate friction-less; in the case of the MSFC installations the operational time is approximately 10 minutes, since the conditions of the flat floor are different than those at MIT. The steps of the iterative research process (as presented in Figure 4.8) at the MSFC Flat Floor are as follows:

1. **Running tests -** Up to 10 minutes (carriage gas limitations).
2. **Data collection -** Two possible time scales: can take a few minutes while at MSFC or after the end of the work day.
3. **Data analysis and algorithm modification -** Two possible options: full quick iterations on-site at MSFC or extended analysis off-site overnight or over a few days. Limited by travel time.
4. **Algorithm implementation and update** - Updated within minutes at both the MSFC Flat Floor location or at the researcher's remote location.

Two possible iterative research loops result from operating at the MSFC Flat Floor; these are presented in Figure 6.8. A research loop can be closed at the MSFC facilities, in a similar fashion to on-site research at the MIT SSL. If more time is necessary, a second

research loop can be closed with data analysis taking place at the researcher's remote location (e.g. hotel) in increments of days.

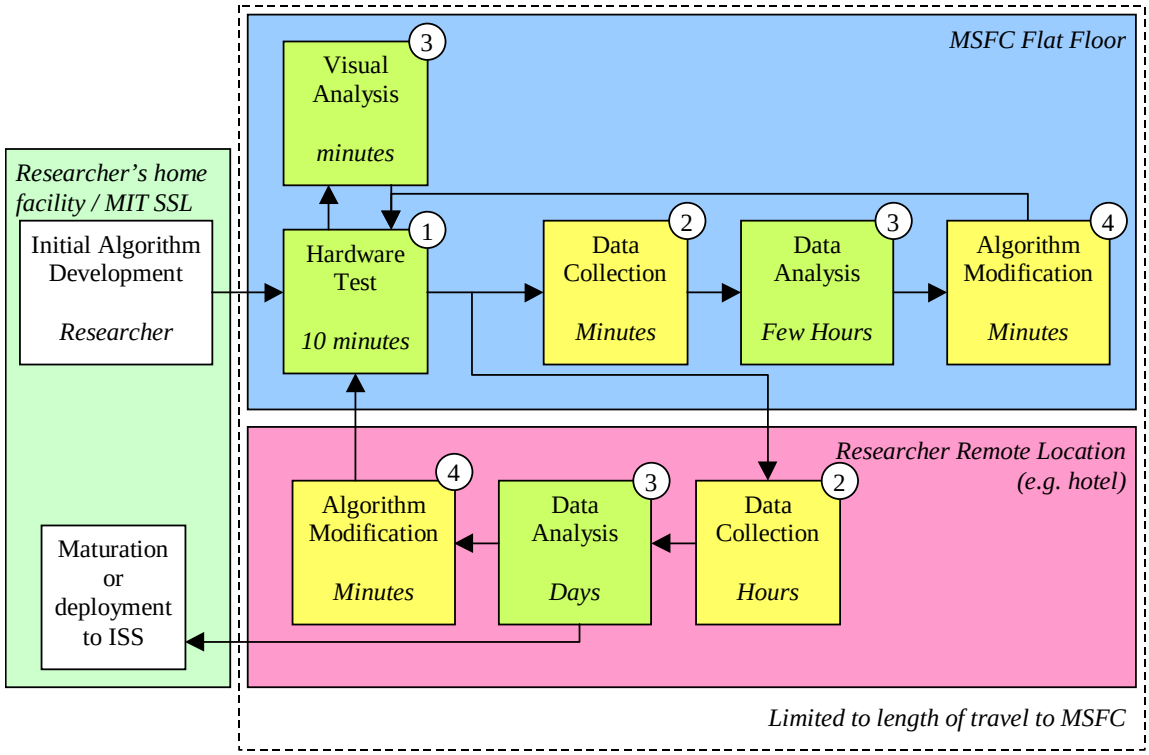


Figure 6.8 MSFC Flat Floor iterative research loops

Table 6.4 presents the iterations that took place during the two weeks of operations at the MSFC Flat Floor. TPF rotations were iterated twice each week; the iterations required a substantial amount of repetitions to collect the necessary data, therefore, although tests were conducted daily, only two iterations took place each week. Docking algorithms, tested during the first week only, were iterated once as tests were done the first day, data analyzed during the third day, and new algorithms tested the third day. Tether experiments iterated four times during the second week of tests at the MSFC Flat Floor. Data was analyzed every night and new algorithms tested each day. These first two weeks of tests did not take advantage of the on-site iterative options for research iterations, but the ability to modify experiments on site was essential to debug the algorithms used each day.

TABLE 6.4 MSFC flat floor iterations

Algorithm	Iters
TPF Rotations	2/2
Docking	1
Tether	4

Operations Summary. SPHERES provides a wide range of iterative loops at different fidelity levels. The operational plans make the steps of improving the fidelity of the test manageable by always keeping the researcher in the loop with minimal overhead times. The availability of the MIT SSL facilities allows scientists to test their algorithms in flight-identical hardware prior to deployment to the ISS. The operational plans for the ISS calls for a flexible iteration time with minimal overhead in the order of days, compared to weeks of science time. Further, the portability of SPHERES has allowed a wider range of operational environments than the three principal locations, further expanding the range of science and overhead times. A summary of the demonstrated science and overhead in ground-based facilities, and the expected times of ISS operations is presented in Table 6.5.

TABLE 6.5 Summary of operational environments and iterative research

	Step				
Location	**1**	**2**	**3**	**4**	**Comments**
Simulation	Researcher	Minutes	Researcher	Hours	Low fidelity models
MIT SSL - Off Site	20 min	Hours	Researcher	Days	SPHERES team member runs tests
MIT SSL - On Site	20 min	Minutes	Travel	Minutes	Maximum level of support
ISS	30 min	2 days	2-4 weeks	2 days	Analysis time in increments of 2 weeks
KC-135	20 sec	Hours	24-72 Hours	Minutes	Challenging environment provides operational feed-back
MSFC Flat Floor	10 min	Minutes / Hours	Hours / Days	Minutes	Possibility of two iterative loops: on site at MSFC and at remote location

Step 1: Test Duration (science time)
Step 2: Data Collection (overhead time)
Step 3: Data Analysis and Hypothesis Update (science time)
Step 4: New Algorithm Upload (overhead time)

Figure 6.9 shows where each of these locations lie within the curve of effective iterations. The simulation provides a large number of iterations with very flexible time. Operations at the MIT-SSL with the research on-site provide many iterations with the time limited by experiment time and researcher travel, neither being critical. Off-site research at the MIT-SSL can provide a larger number of iterations, only limited by test time, although overhead time does become larger. The ISS schedule is expected to allow a reasonable number of iterations (although less than those available in ground facilities), with flexible science time and manageable overhead time. The KC-135 provides up to four iterations (KC-135-1) once a day, or one iteration every year (KC-135-2). Similarly, tests at the MSFC allow a small number of iterations over short periods of time, or one iteration every several months.

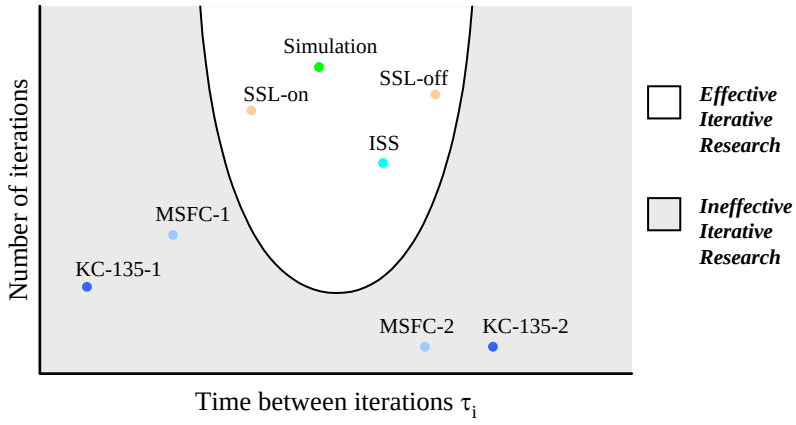


Figure 6.9 Effectiveness of iterations with SPHERES

Iterative Research Conclusions

After several iterations in the design of the SPHERES facilities (the satellites and different user interfaces), the resulting laboratory closely follows the guidelines of the Principle of Iterative Research. The metrology and communications systems provide sufficient data collection and transfer tools to facilitate iterative research. While the systems do have hard

limitations, and their operation in the ISS still must be demonstrated, research in several ground facilities has shown the ability of SPHERES to collect the necessary data.

SPHERES clearly allows not only repetition of experiments, but also modification of both the experiments and the hypothesis. While these changes are limited to the capabilities of the satellites to accept new software and hardware, they have proven enough to iterate on the hypothesis behind several areas of study.

The SPHERES operations plan has demonstrated great flexibility. Not only has iterative research been conducted at the MIT SSL, but also at two remote facilities. At all locations, the SPHERES operations plans work to minimize the overhead time to collect data and update modifications. The available science time varies greatly between facilities, each providing wide ranges of experiment time and data analysis. Each of the facilities has been used to successfully accomplish iterations.

Principle of Optimized Utilization

The use of the ISS resources is as follows:

- **Crew** - Interaction with the crew is an essential element of the SPHERES facilities aboard the ISS as presented in Sections 4.2.1.4 and 4.3.1.2. The presence of the crew is essential to allow scientists to push their algorithms to the limits; if the algorithms fail, the crew can stop a test. The SPHERES program has been designed so that astronauts can provide substantial feedback to the SPHERES team. The astronaut will be allowed to make decisions on the progression of tests, based on information provided by the scientists.

  Test sessions at the ISS have been scheduled for two hours of science every two weeks, plus setup and brakedown. Therefore, SPHERES expects to use approximately six hours per month of astronaut time.
- **Power** - The SPHERES facilities at the ISS utilize a minimal amount of power, but this power is provided by custom battery packs. A full system with three satellites, five beacons, and one laptop transceiver consumes at most 51W. This amount of power is well below the *standard* power supplies of 3kW provided for each ISPR.

  The SPHERES flight hardware does not utilize rechargeable batteries. Therefore, out of the 51W used by a full setup, the only power supplied by the ISS is that of the laptop transceiver (1W), which accounts for less than

2% of the total power. The use of disposable batteries increased the upload mass of SPHERES by approximately 20kg, more than a 30% increase in total upload mass.

The use of liquid carbon dioxide as propellant was a decision made after substantial trade-offs. Fans, air compressors, and available gases in the ISS (mainly nitrogen) did not prove feasible solutions. Therefore, although the CO_2 represents an additional lack of use of available ISS resources, it was selected as the only propellant which provided the necessary combination of operations time, volume, and thrust.

- **Telemetry -** The SPHERES interface operates directly on a laptop computer supplied by the ISS (Station Support Computer, SSC); SPHERES doe not use any other type of data storage. The SPHERES user interface places all the data files directly on the drive shared between the ISS and the ground control center. Therefore, all the experiment data is available as soon as the drives are synchronized.

  The SPHERES team requested real-time video of the first two operating sessions aboard the ISS in order to ensure correct operations of the facilities the first time they are used. The facility has been designed so that future operations do not require (but could use) real-time communications with the astronauts. Therefore, SPHERES will not utilize an undue amount of bandwidth during its operations.

  Based on operations at ground-based facilities, the expected total size of the data files to be downloaded each test session will be 1MB; new programs to upload are expected to be less than 5MB. These transfers can easily take place over several seconds at data rates between 100-200kbps. There is no real-time data download requirement from the ISS to ground.

- **Duration -** The base mission has been defined as ten two hour sessions every two weeks; the consumables have been sized for this operation. Therefore, the basic SPHERES mission is six months long, with the ability to extend the program if consumables can be delivered to the ISS.

- **Benign Environment / Atmospheres -** SPHERES makes full use of those aspects of the benign environment of the ISS that affect it directly: the ability to use a low-cost ultrasound-based metrology system; simple structural design; low-pressure propulsion system; and use of COTS avionics. Further, astronauts have limited access to the SPHERES satellites hardware and software is available to correct problems with the satellites. But the astronauts do not have the ability to correct hardware malfunctions.

SPHERES obtains substantial value from the correct use of most of the resources available at the ISS. Table 6.6 shows the *value* obtained from the use of each resource based on

the charts presented in Figure 5.6. SPHERES slightly under utilizes crew time, for a value of 0.8. The total power of SPHERES is minimal, for a value of 0.99; but because it does not use ISS power sources, it obtains no value from the percentage power. The correct use of telemetry, with flexible download data rates and limited data sizes, give it a value of 0.99. The duration is considered slightly short, although well within the expected lifetime of an ISS mission, for a value of 0.9. Lastly, SPHERES utilizes the ISS environment to a large extent; this subjective measure is given a value of 0.8 since astronauts cannot fix hardware malfunctions. As a result, the SPHERES facilities obtain a value of 4.48 out of a possible 6.0, or a 75%, indicating an acceptable use of ISS resources.

TABLE 6.6 SPHERES value from ISS resource utilization

Resource	Amount	Value
Crew	6	0.8
Power (total)	0.051W	0.99
Power (%)	2%	0
Telemetry	100-200kbps	0.99
Duration	6 months	0.9
Environment	Used	0.8

Principle of Incremental Technology Maturation

The first step to evaluate the design of SPHERES is to determine how far up the TRL levels SPHERES allows a technology to mature. As presented in the definition of this principle, TRL's 5, 6, and 7 will be considered.

TRL 5:

1. *The "relevant environment" is fully defined.*

   SPHERES defines the relevant environment as that available at the ISS US Laboratory: a pressurized microgravity environment with a volume of approximately three meters cubed, full 6DOF dynamics, *no* orbital/celestial dynamics, *no* exposure to the radiation, vacuum, and external elements of a full space environment.

2. *The technology advance has been tested in its "relevant environment" throughout a range of operating points that represents the full range of operating points similar to those to which the technology advance would be exposed during qualification testing for an operational mission.*

   The ability to run families of tests and update the algorithms used for those tests allows scientists to conduct tests throughout the necessary range of operating points to represent qualification for an operational mission.

3. *Analytical models of the technology advance replicate the performance of the technology advance operating in the "relevant environment"*

   The SPHERES simulation has been used to create preliminary models of experiments, prior to testing on physical facilities; the simulation has provided relevant results, with tests replicating the results several times. Therefore, it is expected that the results from models derived in the simulation and ground-based facilities will be able to be replicated in operations aboard the ISS, but this has not been demonstrated yet.

4. *Analytical predictions of the performance of the technology advance in a prototype or flight-like configuration have been made.*

   SPHERES provides an unique opportunity to test the metrology, control, and autonomy technologies of distributed satellite systems in a flight-like configuration for a wide range of missions. Two satellites fully represent docking, rendezvous, and sample capture missions. Three satellite missions provide flight-like configuration for separated space telescopes and the study of cluster formations.

Therefore, SPHERES allows a wide range of DSS technologies to mature to TRL 5.

TRL 6:

1. *The technology advance is incorporated in an operational model or prototype similar to the packaging and design needed for use on an operational spacecraft.*

   The SPHERES satellites are an operational model similar to the design of an operational spacecraft for the maturation of coarse metrology and control algorithms for formation flight, docking, and sample capture.

   The base satellites are *not* representative models for more complex missions, such as stepped control of optical telescopes, the use of active docking ports, or tethered spacecraft. Additional hardware is required to enable SPHERES to fully model the packaging and design of an operational spacecraft. These elements can be added to the SPHERES satellites through the Expansion Port, requiring only small investments in terms of design and launch costs.

2. *The system/subsystem model or prototype has been tested in its "relevant environment" throughout a range of operating points that represents the full range of operating points similar to those to which the technology advance would be exposed during qualification testing for an operational mission.*

   As with TRL 5, the ability to run families of tests and change the programs that run these tests allows scientists to conduct all the necessary tests to cover a range of operating points representative of qualification of an operational mission.

3. *Analytical models of the function and performance of the system/subsystem model or prototype, throughout its operating region, in its most stressful environment, have been validated empirically.*

   The SPHERES satellites have been designed to represent general spacecraft; they do not model any specific mission. The capabilities of SPHERES allow it to demonstrate the capabilities of algorithms empirically, by creating a fully observable and controllable environment which provides data to validate the algorithms. The risk-tolerant environment created by the SPHERES facilities used inside the ISS allow scientists to push these algorithms to their most stressful environment, allowing for technology maturation.

   But SPHERES is not intended to demonstrate specific hardware equipment for use in a mission. While software can help model specific sensors and actuators, and additional hardware can be added to better model a system, the SPHERES facilities are not designed to demonstrate hardware technologies.

4. *The focus of testing and modeling has shifted from understanding the function and performance of the technology advance to examining the effect of packaging and design for flight and the effect of interfaces on that function and performance in its most stressful environment.*

   The SPHERES satellites present realistic limitations in the implementation of algorithms, including finite forces in actuators, bandwidth limited sensors, and constraints in the data processing system similar to that of other spacecraft buses. Therefore, SPHERES does allow scientists to start to concentrate on how to integrate their algorithms into a full system. The data collected can help evaluate the effects of interfaces between the different spacecraft bus sub-systems and ultimately help determine the performance requirements of the flight equipment based on the coupling between sub-systems.

SPHERES enables the maturation of metrology, controls, and autonomy algorithms, implemented through software, to reach TRL 6. The satellites provide the necessary understanding of the interactions between the sub-systems of a satellite through empirical

tests under stressful operating conditions. But the facilities do not allow maturation of hardware technologies to TRL 6 unless these hardware elements can be operated through the SPHERES Expansion Port and the resources exist to deliver them to the ISS.

TRL 7:

> *TRL 7 requires both an actual system prototype and its demonstration in a space environment. The prototype should be at the same scale as the planned operational system and its operation must take place in space.*
>
> SPHERES has not been designed to be an actual system prototype; further, it operates inside the station, so experiments are not exposed to a *full* space environment. In general, SPHERES will not enable technologies to achieve TRL 7 by itself.
>
> The case of MOSR is special, since the SPHERES satellites are of the same scale as the planned operational system, and the capture mechanism will be a prototype of the actual system. In this special case, SPHERES can allow MOSR to achieve TRL 7.

In summary, SPHERES allows a wide range of technologies to mature to TRL 5 with the baseline hardware and software provided in the current design. Projects which only require maturation of software technologies (e.g., algorithms, some artificial intelligence) can mature to TRL 6. Missions that can provide the resources to develop and launch expansion port modules to create the necessary operational models can also mature to TRL 6 with relatively minor investments. SPHERES allows only a limited set of missions to reach TRL 7 maturation, since only missions of the same scale as the SPHERES facilities (satellite size, communications bandwidth, and operations inside the ISS) can reach that level.

6.2.3 Step 3 - Refine Design

- *Principle of Focused Modularity*
- *Principle of Remote Operations and Usability*

Principle of Focused Modularity

The design of the SPHERES facilities consists of the following clearly delineated elements (or sub-systems) to be considered for modularity and reconfiguration:

- SPHERES satellites
  - Propulsion
  - Structures
  - Metrology
  - Data processing
  - Communications
  - Software
- Metrology Beacons
- Laptop Transceiver

The ability to make any of these systems modular and/or allow reconfiguration through them was balanced with the primary science objectives and constraints of operation aboard the ISS:

- Develop a set of multiple distinct spacecraft that interact to maintain commanded position, orientation, and direction.
- Allow reconfigurable control algorithms, data acquisition and analysis, acquisition of a truth measure.
- Enable the testbed to perform array capture, static array maintenance under disturbances (attitude control and station keeping), and retargeting maneuvers.
- Enable testing of autonomy tasks, including fault-detection and recovery, health and status reporting, and on-board replanning.
- Ensure traceability to flight systems via communication, propulsion, structural, avionics, guidance, control, and power capabilities.
- Design for operation in the KC-135, shuttle mid-deck, and ISS.
  - Allow full operations with only one astronaut.
  - Meet all NASA safety requirements.
  - Meet mass & volume requirements for launch aboard one MLE.
  - Account for remote operations.

The science objectives directly call for the software sub-system, through which algorithms are implemented, to be reconfigurable. But the other sub-systems required further analysis, to determine whether making them modular could provide a benefit without interfering with the original mission objectives.

The Principle of Reconfiguration and Modularity provides six specific criteria to test for modularity: interdisciplinary use, reconfiguration, obsoleteness, life-time, cost amortization, and maintenance of the original objectives. The design of the satellites was strongly driven by the constraints for operations aboard the ISS. Each of its sub-systems was developed almost independently of each other, resulting in four different modules (propulsion, communications, data processing, and metrology) with simple interfaces between them, physically put together using the structures sub-system and logically connected through the software sub-system. The interfaces can be easily replicated by other hardware implementations.

Safety constraints prevented any reasonable modularity or reconfiguration of the propulsion system. Not only does the physical reconfiguration of the propulsion system add little value to the main science requirements (since the original configuration allows full 6DOF operations), but physical changes of the propulsion system would require additional hardware (especially to meet safety requirements) which would have prevented the satellites from fitting inside one MLE, which directly conflicts with the original goals.

The communications sub-system interfaces through standard serial ports (UART) to the data processing stack and to the laptop computer. Internally within the satellite the communications hardware is fixed, it does not allow any modularity or reconfiguration because no added value was seen from allowing these elements of the satellites to change. But the development of the external laptop transceiver as an autonomous module which can communicate with any standard PC serial port and use power from a standard USB port does add value to the mission, since the operator’s control station is not limited to any specific computer and the life-time of the module is unlimited. Therefore, the modularity

of the communications transceiver, together with the ground-based user interface, allows SPHERES operations by scientists in multiple areas (interdisciplinary use), while using the same hardware (prevent obsolescence and allow cost amortization) in multiple operating conditions.

Substantial effort was put into allowing the data processing unit to allow reconfiguration and provide a modular interface. Being able to reconfigure the main processing unit (the TI C6701 DSP board) would have been beneficial if the processor could easily be upgraded to newer DSP's. But the microprocessor is not modular itself because enabling direct physical access to the DSP board would have forced the satellites to be larger than the one MLE constraint for launch to the ISS. Further, there was no reason to develop the DSP as a modular system to be used in other projects because the DSP unit does not have any common interfaces (therefore it does not easily allow inter-disciplinary use) and its time to obsolescence is not long enough to warrant use in systems designed in future years. Therefore, rather than allowing upgrades of the DSP board itself, the avionics team created the Expansion Port, which provides several common interfaces to the DSP. The Expansion Port makes the satellites modular, as it allows the satellites, which were designed with a life-span of multiple years, to enable inter-disciplinary use and take advantage of cost amortization as multiple scientists use the facilities.

The metrology system interfaces with the data processing unit with simple time-of-flight signals, but their use requires custom hardware and algorithms to collect and process the data correctly. Further, correct metrology information depends on precise positioning of the sensors in the satellites, and allowing physical reconfiguration presented many challenges to ensure the data was collected correctly. Therefore, it was not easy to make the metrology hardware of the satellites modular for inter-disciplinary use nor allow its reconfiguration. On the other hand, the external beacons enable easy reconfiguration of the global metrology system to accommodate a wide range of operating environments. Their design uses standard track-mounts available in the ISS and space shuttle, and the SPHERES team acquired several of these tracks for use at the MIT SSL and the KC-135.

This allows easy reconfiguration of the global system. To enable modularity, all the beacons are identical. Selection of the beacon number is done through an operator accessible selection switch. This allows the beacons to be interchanged and to operate in different configurations.

As seen, the sub-systems of the SPHERES satellites are not modular elements. Their implementation as separate modules, rather than a single integrated satellites, would have violated the mass/volume constraints to fit within one MLE without adding substantial value to the science goals, to inter-disciplinary use, or to cost amortization. But the satellites as an element do take advantage of modularity. The satellites do allow inter-disciplinary use within the field of study; they are reconfigurable through the Expansion Port, docking port, and the software sub-system; the satellites are not expected to reach obsolescence before re-use with new programs; the life-time is expected to be several years; and the cost of the mission is amortized by allowing multiple scientists to use the equipment.

The software sub-system is reconfigurable to meet the mission's science goals. The software also clearly supports inter-disciplinary use. It has no finite life-time/obsolescence, as it depends on the operations of the satellites only, no other factors affect the time it is usable. The software is modular (Section 4.3.3.1, [Hilstad, 2003a]). It clearly identifies the modules which enable the controls, metrology, communications, and support functions. Scientists can select to use standard modules provided by the SPHERES team or develop their own.

Table 6.7 summarizes how each of the SPHERES sub-systems meets the criteria set forth in the Principle of Focused Modularity and Reconfiguration. The satellites as a whole provide modularity and reconfiguration by being identical satellites, interchangeable with each other, and by using the docking port and expansion port to allow reconfiguration. The propulsion and structures internal sub-systems would have violated the 1MLE constraint if they had been designed as modules, rather than integrated components. The internal metrology hardware requires precise alignment and special hardware to use the

signals, therefore, it is not easily to use in an inter-disciplinary fashion and could cause violation of the 1 MLE constraint. The DSP unit suffered from both ISS constraints and the danger of obsolescence to warrant being a module, although allowing upgrades of the DSP would have been a positive effect of a modular data processing system. Making the communications system modular does not provide a clear value to the system; it does not truly enable reconfiguration. On the other hand, it could provide inter-disciplinary use for other projects, and its time to obsolescence and life-time are not of great concern. But the system was not designed in a modular fashion since it provided no benefits for the SPHERES project. The software sub-system is highly reconfigurable and modular as a direct result of the mission goals. The metrology beacons are modular in their ability to be interchanged and reconfigured with ease to provide accommodate different operational environments of the global metrology system. The laptop transceiver enables the use of the SPHERES facilities through any standard PC serial port at many locations.

TABLE 6.7 Modularity of SPHERES

Sub-System	Inter-discipline	Reconfig.	Obsolete	Life-time	Cost-Amortize	Original Goals	Implementation
Satellite	1	1	1	1	1	1	Identical, interchangeable satellites; docking port; expansion port.
Propulsion		0				0	Not Modular
Structures						0	Not Modular
Metrology	0	0				0	Not Modular
Data Processing			0			0	Not Modular
Communications		0					Not Modular
Software	1	1	1	1		1	FLASH memory for reconfiguration. Guest Scientist Program for modularity.
Metrology Beacons		1	1	1	1	1	Identical beacons with user-selectable configuration
Laptop Transceiver		1	1	1		1	Use of standard interface (UART) and power (USB).

Principle of Remote Operation and Usability

The SPHERES laboratory was specifically designed for operations aboard the International Space Station, where the operators and researchers are distinct individuals; it was also designed for operations at the MIT SSL and NASA's Reduced Gravity Airplane, where the operators are sometimes the researchers. Therefore, as presented in Section 4.3.2.8, there are different interfaces of the SPHERES laboratory to satisfy operations at the different locations.

The Principle of Remote Operations and Usability separates the requirements for operators and researchers:

- Operator
  O1. Provide necessary controls to conduct research efficiently
  O2. Ensure safe data transfer regardless of operator actions
  O3. Present relevant information for successful run of experiments
  O4. Enable operators to provide feedback
  O5. Allow real-time communications for selected operations
- Researcher
  R1. Minimize efforts to collect data
  R2. Allow upload of information
  R3. Enable real-time communications for selected operations
  R4. Allow scientists to predict results and compare with collected data

The prototype interface concentrated on the development of the facilities and immediate science feedback, rather than the operation at any specific location. While the interface had the ability to present custom data in real-time, the data did not aid in operations, rather, it distracted operators in environment such as the RGA (violating O3). This interface also violated requirement O2, since it saved only recognized data. Because the interface was used only by the SPHERES team, it required no direct feedback mechanism (O4) or real-time communications (O5, R3). The interface did meet requirement O1, as it allowed easy operation of the units, informed the operator when tests were running, and

when data was received. By collecting processed data the interface attempted to satisfy requirement R1; data was easy to read from the stored files. The design tools necessary to load new programs (R2) were available since the prototype design. But the prototype systems did not include a simulation to allow scientists to predict their results and compare them, violating requirement R4.

The prototype interface evolved into two separate programs: a ground-based interface and an ISS interface (Section 4.3.2.8). Further, the SPHERES simulation (Section 4.3.3.1) was developed to account for the remote location of scientists who are not members of the SPHERES team. In this manner, the SPHERES laboratory meets all the requirements of this principle.

The ground-based interface was designed for operations at the MIT SSL, NASA RGA, MSFC Flat Floor, and other facilities where the operators are either the researchers and/or members of the SPHERES team. This interface addresses requirement O1 (control of the facilities) by enabling simple operations for all common tasks and incorporating program upload (R2) directly into the interface. The availability of optional windows with real-time state and debug data allows the interface to provide relevant data (O3) when the operators are the research scientist; otherwise the presented data is only that essential for the operation of the satellites. This interface saves data in its raw format, so that scientists can do substantial post-processing and do not loose any information (O2). The interface *does* require the operator to initiate data storage, therefore creating the potential situation where data is not stored due to operator error. To address R1, minimize data collection time, the SPHERES team developed several Matlab functions to collect the data from the raw data files created by the interface. The SPHERES simulation and the information provided with the Guest Scientist Program fulfills requirement R4, allowing the scientist to create models of their experiments and compare the information. Because this interface was designed for use in ground-facilities with scientists or SPHERES team members present, requirements O4, O5, and R3 are not applicable.

While ground operations depend almost entirely on the scientist and the SPHERES Team, ISS operations depend on more parties: the SPHERES Team, PSI, STP, NASA/ISS Mission Control, and the astronaut. Therefore, the interfaces for operations aboard the ISS satisfy the requirements in different ways, since they also must meet requirements set forth by other parties.

The design for control of the satellites (O1) had to meet NASA requirements, apart from the needs of SPHERES. Therefore, the ISS interface requires the use of several steps to start a test. These steps take into account the need to ensure the operator select the correct program and test and is aware of the expected results of each experiment. While the ground-based interface allows test and maintenance tasks to be performed from the main window, the ISS interface presents separate windows/processes.

The ISS interface stores data immediately upon starting. Regardless of the operator's actions, the program will save all outgoing and incoming raw data, ensuring the data is safe regardless of the operator's actions (O2). If a test terminates unexpectedly or is canceled by the operator, the file is saved automatically.

The flight GUI presents information to the operator (O3) in several sections. The state information of the satellites are presented permanently through a status bar. This ensures the operator is always aware of which units are operating and what program is in use. Descriptions of the tests allow the operators to know expected results and make decisions on the test performance. By providing sufficient details on the test, the interface reduces the dependency of real-time communications with the researcher.

The ISS interface presents a questionnaire to the astronaut at the end of each test, requiring the astronaut to provide feedback (O4). The questions are written specifically for each test so that the feedback from the astronaut provides the maximum amount of information to the scientist. Further, the astronaut is allowed to enter notes freely after the questionnaire, allowing feedback on topics not originally considered by the scientist. This feature effectively creates an electronic laboratory notebook.

Operations of the SPHERES laboratory does not require real-time communications in general (O5, R3). Through the data download and astronaut feedback mechanisms, the scientist can determine progress of the research. By interfacing directly with the ISS communications system, the SPHERES facilities can potentially download and upload data and programs in real-time (if the ISS channel is available at the time), even if telecommunications are not established.

Multiple steps were taken to minimize the data download time (R1). The flight interface packages all the data from each session. The use of the ISS telemetry system simplifies the operator's tasks. At that point the data transfer time is dependent on the NASA command center availability to distribute the data to PSI/STP and the SPHERES team. Once the data reaches the SPHERES team, it can be interpreted with the same Matlab tools that were used for initial testing in the ground facilities, since the flight interface uses the same file structures as the ground based interface.

The utilities to upload new programs (R2) are fully integrated into the flight interface. Because a multi-satellite test may require different executable files for each satellite, the interface maintains the structure of the program, making the existence of separate executables transparent to the operator. The interface also manages all the preview files and questionnaires as a single file, so that the astronaut does not have to manage any individual files.

The use of the simulation and MIT SSL ground-based facilities as an integral part of the ISS iterative loop (Figure 4.12 on page 127) allows scientists to predict their results prior to operations aboard the ISS (R4). The facilities are also available after the flight to reproduce allow comparison of results. Further, the availability of raw data allows the results of both ground-based tests and ISS tests to be compared analytically using tools such as Matlab.

6.2.4 Step 4 - Review Requirements and Design

- *Principle of Requirements Balance*

Principle of Requirements Balance

The current design of SPHERES consists of several dozen system-level functional requirements and over one hundred functional requirements for the sub-systems. This assessment concentrates on the system-level requirements derived from the mission objectives summarized in Figure 6.10 [SPHERES, 1999].

The system functional requirements consist of 21 hard requirements (those essential for mission success) and 10 soft requirements (those that would enhance the mission). The hard requirements stem directly from the need to demonstration formation flight algorithms in 6DOF within the KC-135 and the ISS while facilitating iterative research and allowing the study of several areas. The soft requirements derive from desire to demonstrate specific capabilities not fully defined at the time of development (e.g., the mission objective to demonstrate autonomy tasks) or the need to use ground-based facilities (e.g. the KC-135) prior to deployment aboard the ISS. Taken numerically, this is an acceptable division of hard and soft requirements; but one must ensure that the hard requirements drive the mission, while the soft requirements require only limited resources and effort to implement.

The following descriptions illustrate how SPHERES achieved requirement balance, even though it required trade-off's between the functional requirements, including the desire to implement several soft requirements.

> **One-time Use Alkaline Batteries.** Not only did the power sub-system team have a need for re-chargeable batteries, they had the capability to build a ground-based system which has been used continuously since 2001. But the flight hardware utilizes one-time use alkaline batteries. This decision was not a trivial one, but one considered necessary due to the high costs associated with certifying a recharging system for flight aboard the ISS. The SPHERES project team had to balance the need for battery power with the available development resources (both time and money were limited at the time of certification); therefore, while not ideal (especially when the *Principle of Optimized Utilization* is applied), the use of alkaline batteries balanced the efforts required to certify the power system for use aboard the ISS with the resources available.

- Develop a set of spacecraft that interact to maintain position, orientation, and direction.
  - Satellites require translational, rotational, and attitude control capabilities. (Hard)
  - Each satellite must contain its own propulsion, avionics, software, power, communication, and GNC systems within its own structure. (Hard)
  - Satellites must be able to communicate their relative positions. (Hard)
  - Satellites should employ handshaking and negotiation for decision-making. (Soft)
  - Array should consist of at least three distinct satellites. (Soft)
- Allow reconfigurable algorithms, data acquisition and analysis, and provide a truth measure.
  - Satellite must be able to receive control algorithms. (Hard)
  - Satellite must be able to acquire, analyze and send data. (Hard)
  - Allow measurement of relative orientations and positions between satellites. (Hard)
  - Allow measurement of the satellite states relative to the KC-135 / ISS. (Hard)
  - Some of the downloaded data must provide health status information. (Hard)
- Enable the testbed to perform array capture, static array maintenance under disturbances (attitude control and station keeping), and retargeting maneuvers.
  - Should perform self-diagnostic on power up. (Soft)
  - Must determine relative and absolute position. (Hard)
  - Must provide sufficient control authority to counteract environmental effects. (Hard)
- Enable testing of autonomy tasks, including fault-detection and recovery, health and status reporting, and on-board replanning.
  - Compensate for the failure of any other satellite(s). (Soft)
  - Detect a total failure of one of the others. (Soft)
  - Recognize and compensate for minor failures in its subsystems. (Soft)
  - Able to report any minor failures back to an external monitor. (Soft)
  - Able to regularly report the status of each of its subsystems. (Soft)
  - All of the satellites should be physically identical. (Soft)
- Ensure traceability to flight systems via communication, propulsion, structural, avionics, guidance, control, and power capabilities.
  - Enable traceable control algorithms to future missions. (Hard)
  - Provide representative dynamics of the propulsion. (Hard)
  - Provide precision metrology system equivalent to actual applications. (Hard)
  - Enable data communications equivalent to real missions. (Hard)
- Design for operation in the KC-135, shuttle mid-deck, and ISS.
  - ***KC-135***
  - Functionality needs to be demonstrated in <20 sec. (Hard)
  - Operate within the space confines of the KC-135. (Soft)
  - Allow for retrieval and restraint during 2g fall section of flight. (Hard)
  - Meet all applicable KC-135 safety requirements. (Hard)
  - ***ISS***
  - Satellites must fit into shuttle mid-deck locker. (Hard)
  - Enable demonstrations within the confines of the ISS. (Hard)
  - Must allow protocol test time of two hours. (Hard)
  - Meet all applicable ISS safety requirements. (Hard)

Figure 6.10 SPHERES Functional Requirements

Custom Metrology System. The need for a metrology system that identified the full state of the satellites for formation flight control is a hard requirement, but one that is not fully quantitative. The requirements for the metrology system were originally specified in "sub-centimeters" (range) and

"degrees" (rotation), but the actual values were determined by the selected system. The final selection of a custom ultrasound and infrared time-of-flight system was mainly a trade-off between acquiring a COTS product and building a custom one. At the time of the development of SPHERES, only a handful of COTS products existed; the majority of them had a cost beyond 15% of the total SPHERES budget, making them unattainable. Further, all the reviewed systems required modifications from their original configuration. Therefore, the SPHERES team decided to develop a custom metrology system (Section 4.2.1.1) which would incorporate directly with the other sub-systems. The final design has demonstrated the ability to meet the design requirements, with specific quantitative resolutions provided (0.5cm, 2.5°) in ground-based 2D operations; 3D operations are yet to be demonstrated aboard the ISS. Ultimately, as the results of tests aboard the KC-135 show, the development of a custom metrology system utilized a substantial amount of resources and effort beyond that of any other sub-system. While metrology itself is part of the science being conducted with the SPHERES laboratory, this requirement did create an unbalance between metrology and all the other sub-systems.

Propellant Selection: CO_2. The Selection of carbon dioxide as propellant does not appear to be optimal. The use of CO_2 means increased safety requirements, including the use of a toxic gas and development of a pressurized system. Further, it is not possible to replenish CO_2 aboard the ISS. But this selection was due to the fulfillment of several other requirements: occupy at most one MLE (mass and volume), provide thrust to perform the strawman maneuvers, allow traceability to spacecraft, and maintain development costs in control. The selection of CO_2 over any other pressurized gas allowed the propulsion system to maintain the amount of resources and effort invested on it balanced with the other sub-systems, as little custom work was required and the technologies to handle carbon dioxide were clearly understood. While it required substantial more time investment in the safety process, it did not require substantial development efforts, which gave balance to the selection.

Expansion Port. While modularity and reconfiguration was initially built into SPHERES, it originally was only conceived as part of the software system. The development of the expansion port (Section 4.3.3.2), which enables hardware reconfiguration, came late in the process. Therefore, its implementation required that only minor changes be needed from the existing sub-systems. It was essential that the addition of an expansion port did not drive the mission beyond its constraints, especially the need to meet launch deadlines (such as CDR, safety reviews, etc.). This required the expansion port to use existent data channels and to fit within space available in the system. The resulting expansion port attempts to provide for future projects by using both

simple and complex data channels, as well as several power voltages. The serial and power lines have been utilized in several projects, and their usefulness demonstrated; no projects have utilized the global memory bus as of yet. This discrepancy in the use of the expansion port data lines is due to an imbalance in the resources and effort put to develop the expansion port. The requirement for hardware modularity was given low priority and assigned only limited resources, while the other requirements drove the mission.

Communications Channel Frequency Selection. Enabling iterative research has always been a primary functional requirement for SPHERES. Yet, to meet this requirement two potentially conflicting requirements existed: de-couple the software from safety reviews to minimize the overhead time to upload new algorithms and provide the necessary tools to collect substantial data. The communications channel has a conflict between these two requirements since the use of an 802.11b wireless LAN interface card can provide over 10Mbps of bandwidth utilizing standard COTS equipment and publicized protocols; but such a system requires that the software be controlled, since the 802.11b network is part of the ISS controlled environment. Therefore, the SPHERES team required that the communications system utilize an uncontrolled frequency range. At the time of development of the SPHERES testbed, the simplest integration was through the use of 916MHz technologies. This required substantial effort in the development of a custom communications protocol and limited the bandwidth to at best 56.6kbps. But, the use of the custom system allowed the software sub-system to remain decoupled from any safety requirements.

6.2.5 Design Framework Assessment Summary

Having been designed to exhibit the features of the MIT SSL Laboratory Design Philosophy, SPHERES closely follows the principles which derived directly from the philosophy. The principles of *Enabling a Field of Study* and *Iterative Research* have been successfully implemented in SPHERES. The Guest Scientist Program enables research by multiple scientists by providing the necessary tools for scientists to conduct research in their home locations (simulation, data analysis) and at remote facilities, with option to be present at several testing facilities (MIT SSL, NASA RGA, MSFC Flat Floor). The laboratory facilitates iterative research by allowing the necessary reconfiguration of algorithms, minimizing overhead time to repeat experiments and upload new algorithms, and providing scientists with flexible operations plans. The SPHERES laboratory exhibits the major

traits called for in the principle of *Focused Modularity.* While modularity did not play a major role in the design of the individual satellite sub-systems, it did guide the overall design: the satellites represent a standard satellite bus. Further, the software exhibits modularity throughout.

The principles of *Optimized Utilization* and *Remote Operations and Usability* derive from the need to operate aboard the ISS, a task inherent in the design of the SPHERES laboratory. SPHERES makes an acceptable use of the resources aboard the ISS, although several resources were not utilized correctly due to the inability to fulfill all existent NASA requirements with the resources available. SPHERES accounts for all the considerations raised in the principle of *Remote Operations and Usability*, providing both the operators and scientists with the tools to conduct remote research. These include the ability of the operator to control the experiment, preview expected results, and provide feedback to the scientist. The scientist has tools to predict results, analyze the data, and compare results. The correct use of ISS resources enable real-time communications between the operator and the scientist if necessary.

SPHERES provides the ability for scientists to test metrology, control, and autonomy algorithms in a representative environment, achieving TRL 5 or TRL 6 in most cases, and TRL 7 in a selected few. This ability allows SPHERES to meet the principle of *Incremental Technology Maturation* because the cost of SPHERES is minimal compared to the full cost of an operational mission. By providing a representative environment at low costs, where the technologies can be demonstrated and the risks reduced, SPHERES can allow the total cost and risk of an operational mission to be reduced.

The principle of *Requirements Balance* originated from the observation that the two guidelines behind the development of SPHERES did not specifically account for the limited resources available for a mission. The successful development of the SPHERES laboratory indicates that enough requirements balance took place to create the design. But review of the implementation demonstrates that further requirements balance could have

occurred to prevent incosistencies in the effort and resources put into the development of a few sub-systems (e.g., metrology), while others (e.g., power, expansion port) could have used further resources to provide important benefits.

6.3 Evaluation Framework

An ISS NGO evaluator would be provided with a proposal that describes the design of a mission with enough details to address all of the microgravity laboratory design principles. The review by the NGO addresses the following points:

- Correct utilization of ISS resources
- Technology advancement
- Mission scope

In the case of SPHERES an ISS NGO would receive information similar to that presented in the SPHERES Critical Design Reviews (Technical: [SSL, 2002] and Science: [SSL, 2002a]) and the Safety Data Packages [SPHERES, 2001]. This information is now used to asses SPHERES using the evaluation framework presented in Section 5.11.

Principle of Iterative Research

The stated science objectives of SPHERES (develop metrology, control, and autonomy algorithms for distributed satellite systems) clearly indicate the need for iterations. To demonstrate the maturation of algorithms requires running multiple tests and evaluating the results until the desired performance is met. The ISS evaluator must asses the ability of SPHERES to support iterative research based on the evaluation framework:

1. *Does the experiment collect the data necessary to support or refute the hypothesis?*

   It is not the job of the ISS NGO evaluator to guarantee that the proposed facilities collect all the data necessary for mission success, but it is in the best interest of the ISS program to determine if a project has the ability to collect relevant data through either custom hardware or systems available aboard the ISS. In reviewing SPHERES, the evaluator can see the existence of two measurement systems (inertial and global metrology systems), the use of the communications system to download data to an ISS SSC, and the use

of the ISS telemetry system to download the data to ground.

The SPHERES CDR clearly indicates that the sensors of the satellites provide the necessary measurements to perform tests of the different algorithms; the evaluator need not question the proposal unless there is an obvious question on the ability to collect the data. In the case of SPHERES there is no obvious failure to collect the data, and the resources aboard the ISS have been used correctly to this purpose.

2. *Do the operational plans of the facility provide sufficient flexibility for efficient iterations?*

The SPHERES operational plans clearly include wide flexibility, but the ISS NGO should notice that a lot of this flexibility depends on periodic operations aboard the ISS. The ground-based tests prior to ISS deployment are outside of the control of the ISS program, and allow scientists substantial research time to analyze data and come up with new algorithms. But the plan for operations aboard the ISS depends on having test sessions every two weeks, at which point the ISS staff will play a critical role in the effectiveness of iterations. Therefore, the ISS NGO staff must, at least, make note that the ability of this facility to perform iterations requires allocation of resources by the ISS program. In the case of SPHERES the resources are not mission critical; that is, if one or two sessions are missed, the program will not fail. Therefore, it is possible to allocate the resources without undue stress on the ISS program.

3a. *Can the facility perform multiple experiment runs with repeatability and reliability?*

The facilities of SPHERES have been designed specifically to allow for repetitions of tests aboard the ISS, as well as multiple other locations. The operator can start tests with minimal setup time; the interface provides simple controls to start and stop tests. Further, the reliability of SPHERES has been demonstrated in ground-based facilities. But the project does have a major limitation in its ability to support repetition: depletable propellant and batteries. The ISS NGO needs to evaluate the ability of SPHERES to perform repetitions over an extended period of time, and get assurances that the consumables can be resupplied to the ISS for continued operation. SPHERES has limited launch capabilities for consumables; currently they account for up to six months of operation. This is a reasonable time frame, although the ISS NGO should make note of this limitation in the review of SPHERES.

3b. *Can the facility be reconfigured while in the ISS in such a way to provide new meaningful results and/or reflect changes in the hypothesis?*

The primary scientific objective of SPHERES is to develop algorithms for

distributed satellite systems. The SPHERES facilities aboard the ISS allow reconfiguration of the algorithms being tested via wireless links between the satellites and the ISS SSC. Therefore, SPHERES allows reconfiguration of both individual algorithms and of high-level hypothesis.

The proposed expansion port of SPHERES allows further reconfiguration, including the change of hardware. But because these changes require launch of further hardware to the ISS, an ISS NGO evaluator must first see that the program has the resources available to deliver these items to the ISS. Because SPHERES has not yet demonstrated this capability, the evaluator should only consider software reconfiguration under this question.

Based on these reviews, an ISS NGO evaluator can see that the SPHERES laboratory enables iterative research at all the levels presented in Figure 5.1 for software-based research. While several reservations exist on the ability of the laboratory to support iterative research in the long-term, the proposed initial mission should be successful.

Principle of Enabling a Field of Study

Review of the Science CDR [SSL, 2002a] immediately identifies the field of study covered by the SPHERES laboratory: metrology, control, and autonomy algorithms for distributed satellite systems. The development of these algorithms is essential for several upcoming missions; their maturity through demonstration in a space environment can greatly reduce the risk of the operational missions. The same presentation contains an overview of the Guest Scientist Program, which provides the capabilities to support multiple scientists:

- *Efficient data paths* - Data transfer to and from the ISS includes the use of the ISS data system to minimize the delay in delivery to the SPHERES program. The ISS NGO evaluator wants to ensure that if the ISS systems are used, the program does not create undue delays for the data to reach the scientists, countering efficient use of ISS resources. The SPHERES team does not create any further delays in the delivery of data, which is immediately forwarded to the scientists.
- *Data analysis tools* - The evaluator can determine from the Science CDR that guest investigators have a simulation to predict results at their home facilities. Further, the GSP clearly defines all standard SPHERES data packets and allows scientists to define their own structures.

- *Flexible operations* - As with iterative research, the SPHERES plans do provide substantial flexibility, but require the allocation of resources by the ISS program. This allocation of resources is within the capabilities of the ISS, although it may have to be reviewed in the long term.
- *Reconfiguration* - As before, demonstrated reconfiguration of the SPHERES facilities aboard the ISS is limited to software changes. SPHERES does allow multiple scientists to utilize the ISS facilities as long as their tests require only software. To allow hardware reconfiguration, the SPHERES program will need to demonstrate the capability to deliver new hardware to the ISS.

From the ISS program point of view, the SPHERES laboratory has been correctly designed to allow research on a substantial field of study (algorithms for distributed satellite systems) and has created the necessary programs and facilities to support multiple scientists. These programs do utilize ISS resources, but not beyond the capabilities of the ISS program. Therefore, the ISS NGO would have no recommendations to change any aspect of SPHERES with respect to the principle of *Enabling a Field of Study.*

Principle of Optimized Utilization

An ISS NGO evaluator must determine if a project makes proper use of the special resources available at the ISS. The design of SPHERES demonstrates utilization of many resources available at the space station. Further, the presented trades between operational environments indicate that the program would not have the resources to operate as a standalone space mission; that it does not have the need for such operations; and most importantly, that it would loose many of the benefits obtained by operating in the controlled environment of the station with human interaction (i.e., would no longer create a risk-tolerant environment for algorithm research) if it were a free flyer program. Next, the evaluator should asses whether the utilized resources of the ISS are used correctly and if other resources exist to benefit the program and at the same time make better use of the station:

- *Crew* - SPHERES is programmed to utilize approximately six hours of crew time each month, accounting for half an hour of setup, two hours of tests, and half an hour of breakdown. In terms of the absolute amount of time, this is an acceptable amount, towards the lower side of expected crew use.

The importance of the crew involvement as an integral part of the science conducted with SPHERES stands out. SPHERES requires the astronaut to truly become an extension of the researcher, requiring the astronaut to understand some of the science aspects and make decisions based on their own observations. From the ISS NGO perspective, this is both an asset and a concern. It is an asset because astronauts will be researchers in space, directly in line with the main objective of the station. It is a concern because the astronaut may be responsible for the scientific success of the mission. Therefore, the ISS NGO evaluator will need to put extra emphasis on the need to satisfy the principle of *Remote Operation and Usability.*

- *Power* - The SPHERES facilities aboard the ISS utilize minimal amount (50W) of power compared to that available for experiments. But SPHERES requires the delivery of custom batteries to the station, rather than using existing power resources. For the ISS program this means the use of both launch, storage, and disposal resources for a resource widely available at the station. Upon review with the SPHERES team, the evaluator would learn about the lack of resources to develop a recharging station and put it through the necessary safety review processes. The evaluator should then present the problem to the ISS NGO leaders, who have the responsibility to review the processes required to allow better utilization in the future. Upon review of the procedures, the ISS NGO staff should continue in contact with the SPHERES team to encourage them to better utilize the power resources.
- *Telemetry / communications* - SPHERES utilizes a small amount of telemetry during normal operations; its real-time communications are limited to pre-specified operating sessions. The use of the existing SSC, which communicates directly with the ISS network, simplifies integration of the project with existing resources. Therefore, SPHERES successfully utilizes the telemetry and communications resources of the ISS.
- *Long term experimentation* - The expected mission life of six months to one year presented in the critical design review is a valid utilization of the ISS. But this longevity depends strongly on the availability of the two consumables used in SPHERES: batteries and propellant. The ISS NGO evaluator should be interested in knowing the capabilities of the facilities in case each of the resources run out, therefore should ask the SPHERES team to present contingency plans in case consumables run out.

  Note that utilizing rechargeable batteries would better ensure long-term experimentation, reinforcing the need for the SPHERES team to reconsider the use of disposable batteries.
- *Benign environment / atmosphere* - The presented design for the SPHERES facilities aboard the ISS require a pressurized atmosphere; the project takes advantage of this resource to greatly reduce its costs and development time.

Further, SPHERES makes use of the controlled environment of the station to reduce the risk involved in developing new algorithms.

SPHERES successfully utilizes four of the five ISS resources identified for special consideration. The crew is involved directly with the science, beyond mechanical or repetitive activity. The ISS telemetry data system is well integrated to the facilities. Assuming no unexpected problems with delivery of consumables, the longevity of the project is appropriate. The project utilizes the benign environment to reduce both its costs and risks. While the power consumption is minimal in terms of available ISS power, SPHERES requires its own power source (disposable batteries). The project must be reviewed to correct this problem, which would not only reduce delivery costs to the station, but also help ensure the longevity of the mission.

Principle of Focused Modularity

To review the modularity of SPHERES, the ISS NGO considers the three main elements of the facilities delivered to the ISS:

- *Satellites* - The satellites do not exhibit any modularity of their sub-systems. The structure does not allow separation of the sub-systems, therefore it is not possible to utilize any specific part of the satellites in other projects. A review of the sub-systems would identify the usefulness of some modularity in the satellites. The communications system could be modular, providing a simple wireless communications interface to other projects. If the batteries were rechargeable, they could provide power to new facilities. While in some cases the microprocessor could be made modular, the selected SPHERES processor is too specific to the application, therefore there is no need to make it modular from the ISS perspective.

  The satellites do provide the ability for reconfiguration via their expansion port (requires hardware delivery) and the update of the software and user interfaces (via the ISS communications channels). Therefore, the satellites can be used for as yet unforeseen projects.

- *Metrology beacons* - The metrology beacons are identical and can be reconfigured easily. While they require specific signals to operate, there are no limitations which restrict their use with other projects beyond the SPHERES facilities. Therefore, these beacons could become a tool available for other project which needs a ranging system and can accommodate the signal timing requirements.

- *Communications* - The transceiver provided by SPHERES operates with any module that has an RS232 serial interface (UART). This allows the module to be used with a wide range of projects beyond SPHERES. But it requires that each new project utilize exactly the same communications hardware, with its custom firmware, greatly reducing the ability to consider the communications transceiver a modular entity. Like with the satellites, the SPHERES communications transceiver could be reviewed to allow for better modularity.

From the ISS perspective, SPHERES offers little modularity. The satellites provide substantial reconfiguration capabilities, allowing their use in future projects. But this depends on the ability to launch hardware and develop new software. Further, these new uses depend on the SPHERES team cooperating with new scientists, and does not directly provide new capabilities for the station. Therefore, the satellites do not provide any substantial modularity to the station. The communications system, while modular on the ability to operate with any RS232 serial line, requires the use of special hardware and firmware, limiting its usefulness. The metrology beacons do provide a new system which could be made available to new users. A clear interface exists and there are no physical limitations to their use by other projects.

Principle of Remote Operations and Usability

This principle calls for the reviewer to determine if the proposed program has clearly identified the data and information requirements to provide scientists with sufficient data and operators with the necessary information to conduct science. The SPHERES program clearly identify the data paths, allows scientists to create custom data, and provides methods to test the selected data through actual experiment runs prior to deployment to the ISS. Therefore, the ISS program has reasonable assurances that the data collected aboard the station will be sufficient for the scientist. The requested real-time communications are not intended for the scientists directly, but rather for the SPHERES team to support the operator during the first two operating sessions. Thereafter, the operator is expected to use the facility independently, so the ISS NGO must ensure that the operator has enough information to perform the required tests in the future.

The principle provides the following guidelines to evaluate the capability of an operator to successfully run tests:

- *The operator has the necessary interfaces to control the facilities aboard the ISS in an efficient and safe manner.*

  The SPHERES interface provides the operator with a clear procedure to load programs, select tests, and start experiments. Starting an experiment is contingent on the astronaut enabling the necessary satellites, which implies the astronaut has control of them. The interface ensures that whenever a test is in progress, the stop command is always available to the astronaut. Therefore, the operator does have the necessary controls to conduct tests in an efficient and safe manner.

- *The operator is presented the information necessary to successfully evaluate experiments.*

  The SPHERES documentation indicates that the operator is to be presented with a preview (written explanation with optional images and/or animations) of the experiment before a test is started. The description of each test will provide the astronaut with a clear idea of the expected results. A result code will allow the astronaut to compare their observations with the determination of the algorithm, providing further information to determine if the test was successful. Using this information, the operator can make the decision to repeat a test or move on with the program.

  Therefore, upon review of the existing documentation, the SPHERES program appears to provide the necessary information for astronauts to evaluate experiments. Still, the ISS NGO must clearly establish that this depends on the SPHERES team providing the information continuously. Further, because astronauts conduct a wide range of experiments on many areas, they should not be held responsible for any misinterpretations of the presented previews.

- *The operator can provide feedback to the research scientists from their observations in the operational environment.*

  The SPHERES project accounts for the ability of the operator to provide feedback in two manners: the use of a survey at the end of each experiment, with questions directly related to the science; and the ability of the operator to provide open feedback after the survey. This information is stored together with the data collected during the experiment, and transferred at the same time. Therefore, the scientist can always correlate the collected data with astronaut feedback to better evaluate the success of an experiment.

From the ISS NGO perspective, it appears that the SPHERES team has accounted for all the necessary tools to allow successful remote operations of the SPHERES facilities, even

without real-time communications between the scientists and the operator. The design does depend on the correct interpretation of the descriptions, images, and movies provided in the previews and result codes presented after each test. Therefore, the possibility for operator error exists, but it has been reduced by the providing the astronaut information both before and after each test.

Principle of Incremental Technology Maturation

By correctly utilizing the environment of the ISS, SPHERES provides scientists with a relevant environment to test new metrology, control, and autonomy algorithms aboard the space station. The ability to run multiple tests, change operating conditions, and reconfigure the software covers the necessary operating points to demonstrate the technology. The fact that scientists can present previews demonstrates the ability of SPHERE to enable predictions of performance; the data collection allows comparison of these predictions with actual results. Therefore, SPHERES satisfies the requirements to advance technologies to TRL 5.

SPHERES can achieve TRL 6 and TRL 7 only for those programs where the hardware represents either operational models of the sub-systems (TRL 6), or full scale prototypes of the operational system (TRL 7). Therefore, the ability of SPHERES to advance technologies beyond TRL 5 depends on the specific mission being considered. The ISS NGO evaluator can consider that TRL 6 may be achieved for a limited set of missions; the evaluation should not consider the capability of SPHERES to mature technologies to TRL 7 unless specifically addressed by the SPHERES documentation. Therefore, SPHERES is considered a laboratory which allows technology maturation to TRL 5 and TRL 6.

6.3.1 ISS NGO Evaluation Summary

Overall the SPHERES laboratory successfully implements the features called for in the design principles. The deployment of the SPHERES facilities to the ISS present a unique opportunity to expand the science conducted aboard the ISS for more than a single research program. A clearly established Guest Scientist Program opens up research aboard

the ISS to multiple scientists conducing research on distributed satellite systems metrology, control, and autonomy algorithms. The laboratory clearly accounts for the need to facilitate iterative research through its capabilities to repeat tests with minor overhead and the ability to reconfigure the software to enable modifications of algorithms and hypothesis. Utilizing SPHERES, these algorithm technologies can be matured to TRL 5 or TRL 6 in most cases.

SPHERES makes correct utilization of the facilities aboard the ISS. The crew, telemetry, long-term experimentation, and benign environment resources are used appropriately. While SPHERES minimizes its power consumption, to levels almost negligible to the availability of power aboard the ISS, it requires the launch of power sources (batteries) which have mass and volume not negligible to the ISS. Therefore, an ISS NGO would strongly urge the SPHERES project to upgrade their facilities to make use of the power sources available in the ISS. Further, the use of consumable CO_2 as propellant creates important concerns about the ability of the facilities to operate over the long-term; while the ISS NGO cannot provide a reasonable substitute, the SPHERES team should provide the evaluator with contingency plans on how to obtain partial use of the SPHERES facilities aboard the ISS in the case where the propellant is not available.

The facilities provided to the astronauts aboard the ISS provide substantial tools to make the operator an extension of the scientist. Not only does the operator start and stop tests, they also evaluate the experiments to determine, without real-time scientist interaction, whether a test was successful or not and when to repeat tests or move on with a program. The interface provides operators with substantial information to make these decisions. Further, they allow the astronaut to provide feedback to the scientist, effectively creating an electronic laboratory notebook where the astronaut can make annotations about each test.

Chapter 7

CONCLUSIONS

7.1 Thesis Summary

The objective of this thesis is to *create a design methodology for the development of microgravity laboratories for the research and maturation of space technologies*. This goal is motivated by the desire to combine the experiences learned by staff of the MIT Space Systems Laboratory through more than two decades of microgravity research aboard the Space Shuttle, MIR, and the International Space Station. This research includes experiments with the Middeck 0-g Dynamics Experiment (MODE), the Dynamic Load Sensors (DLS), and the Middeck Active Control Experiment (MACE). Further motivation arises from the call by the National Research Council to create a non-governmental organization to *manage all aspects of research on the ISS* in such a way that *the research community [has] early, substantive, and continuing involvement in all phases of planning, designing, implementing, and evaluating the research use of the ISS* such that *basic and applied scientific and engineering uses [of the ISS are] selected on the basis of their scientific and technical merit, as determined by peer review.* Further, the NRC report states that *the organization must be flexible and capable of adapting over time in response to changing needs and lessons learned* [NRC, 1999].

The development of the design methodology follows the steps of the scientific method: objective definition, hypothesis formulation, experimentation, results analysis, and conclusions determination. Based on the objective defined above, the hypothesis states that

the use of the International Space Station as a host and following the MIT SSL Laboratory Design Philosophy as design guidelines enable the development of a low-cost environment for the development and operation of facilities to conduct space technology research. The experimentation consists in the design, implementation, and operation of the SPHERES Laboratory for Distributed Satellite Systems. The laboratory consists of several facilities, including a simulation, ground-based operations for 2D tests, and operations aboard the ISS for 3D tests. Analysis of the design of SPHERES, the MIT SSL Laboratory Design Philosophy, and the resources and operations of the ISS results in the creation of the Microgravity Laboratory Design Principles, a design framework for scientists to develop new laboratories, and an evaluation framework for ISS staff to determine if a project utilizes the station correctly. The thesis concludes with the application of both frameworks to the SPHERES laboratory.

In order to better understand the existing resources available to mature space technologies and how research is performed at remote locations, Chapter 1 presents an overview of the NASA Technology Readiness Levels used to determine technology maturation; it reviews existing facilities that allow research in actual or simulated microgravity conditions; and reviews existing remote laboratories. The NASA TRLs provide an example of an existing methodology to evaluate a technology after its testing environment has been designed and the tests conducted. They also provide specific information on the requirements of a laboratory designed to demonstrate space technology maturation.

Chapter 1 reviews the microgravity facilities and remote research facilities summarized in Table 7.1. The review of existing facilities includes facilities located at the home location of researchers, such as simulations, robots, and air tables. Remote ground-based facilities, including flat floors, drop towers, neutral buoyancy tanks, and reduced gravity airplanes, are researched to determine their capabilities and operations. The Space Shuttle and International Space Station are the only space-based environments currently available for research in a microgravity environment. Their capabilities and operations are reviewed. The operations, capabilities, and research conducted aboard previous space based research

TABLE 7.1 Research facilities studied in the thesis

In-house	3rd Party/Full μ-g	Space	Remote
Robot Helicopters	RGO (KC-135)	Free Flyer	Ocean Exploration
6 DOF Robot Arms	Neutral Buoyancy	ISS	Antarctic Research
Helium Balloons	Drop Towers	Shuttle Payload	
Robot Cars		Shuttle Middeck	
Flat Floor			
Air table			
Simulation			

stations (Salyut, US Skylab, SpaceLab, and MIR) are also researched. A comparison of the capabilities and operations of the available facilities, with an understanding on how research was conducted aboard previous space stations, indicates that the ISS provides many unique opportunities for research which allow technology maturation through TRL 5 and up to TRL 7. Ground based facilities are not able to simulate the required environment; the use of the Space Shuttle is now highly restricted. Therefore, further research on the ISS is presented in Chapter 2.

Because operations aboard the ISS require that scientists not be present in the operational environment, Chapter 1 also presents research on two existing remote facilities: antarctic research and ocean based research. These facilities create a laboratory environment in remote locations, with strict operational plans and limited resources. The research on these facilities showed that the main goal of the designers is to enable the presence of the scientist at the research locations. Both facilities continuously increase the resources available for humans, specifically the scientists, to be present at the research locations. Therefore, even though these locations are remote, they do not require an operator external to the mission to conduct tests and report results. Rather, they provide operators to assist the scientists in conducting the research in an efficient manner. Given that the presence of the scientist aboard the ISS is practically impossible at this point, this review indicates that research conducted aboard the ISS must stress the need to extend the capabilities of the operator such that they can become an extension of the scientist. The operator should not

only conduct repetitive tasks which could be automated; the ability of the human operator to process information and make educated decisions must be used as an extension of the scientist.

Chapter 2 identifies the challenges of microgravity research observed in the review of present and past research facilities in Chapter 1. These challenges are:

- *Risk* - Every space mission has inherent risks, the possibility of failure at some step of the mission. The risk considered in this thesis involves the possibility of a facility failing in such a way that research can no longer be conducted and technology maturation is not achieved. The goal in the development of the design methodology for microgravity research is to allow a laboratory to reduce the risk of research for technology maturation and of the operational mission.
- *Complexity* - The complexity of space missions grows increasingly as a mission approaches operations. The need to account for system integration without the ability to test all the components usually leads to increased complexity to reduce the possibility of failure; but the increased complexity adds cost to the mission, as well as more failure modes. Therefore, it is desirable to reduce the complexity by allowing tests of integrated systems at every step.
- *Cost* - As a mission approached operational state, its costs increase substantially due to the increase of risk and cost involved, as well as the need to account for the space environment. The use of a microgravity laboratory should allow the cost to be controlled by providing model environments that better represent the operational environment.
- *Remote Operations* - Operations aboard the ISS are necessarily remote; the scientist remain in the Earth, while the operator conducts tests aboard the ISS. To efficiently conduct research, the operator must be provided with the necessary tools to assess experiments and make decisions that help to advance the science, even if it is not possible for the scientist to be in real-time communications with the operator.
- *Visibility* - Space missions usually stand at the extremes of visibility by the public (and funding sources). A mission that is very visible cannot take any risks that could result in failure, many times leading to conservative science goals and/or an increase in the cost of development to minimize the risk in the use of new technologies. On the other hand, missions with low visibility usually don’t receive the necessary resources. Therefore, it is desirable that missions receive resources due to their scientific merits, rather than their visibility.

Next, the chapter presents an in-depth review of the environment and resources of the International Space Station that can help in overcoming these challenges. The chapter introduces the modules that form the ISS, concentrating on those that have been specifically designed to support research: US Destiny Laboratory, US Centrifuge Accommodations Module, US Integrated Truss Attachments, Japanese Experiment Module, ESA Columbus Module, and two Russian Research Modules. Further, the ISS provides scientists with several shared resources: controlled environment, power, communications, thermal management, cryofreeze systems, and payload stowage. The station will also provide up to 22 (most under development) shared research facilities for specific areas such as biotechnology, materials processing, human physiology, and more. But no space technology facilities are planed. The research identifies five special resources of the International Space Station which directly help to overcome the challenges of microgravity research:

- *Crew* - The presence of humans aboard the ISS helps reduce the risk, complexity, and cost of missions to research space technology maturation. Humans can determine incorrect operation of an experiment, preventing critical failures in facilities aboard the ISS, therefore reducing the risk. The use of the crew allows scientists to reduce the complexity and cost of facilities, since they do not require expensive automation tools to operate and collect data. Further, humans can perform direct observations and provide feedback to the scientists.
- *Communications* - Existing communications equipment at the ISS reduces the cost of missions. More importantly, the availability of real-time communications and a substantial bandwidth for data transfer allows scientists to communicate with the operator as needed. The scientist can upload data for the operator to review, and download their data in an efficient manner. The operator can ask questions to the scientist with little delay.
- *Long-term experimentation* - The ability to conduct research over extended periods of time lowers the effects of visibility on a mission. A project which is researched aboard the space station for a long time will not have the public visibility and impact of other missions, but the presence of the project aboard the ISS adds visibility among the scientific community.
- *Power* - The ISS provides substantial amounts of electrical power, as well as several gases. This helps missions reduce their cost, since power sources and storage are no longer required of each project. Further, this reduces the complexity and risk of a mission, due to the simplified power sub-system and guaranteed availability of power over extended periods of time.

- *Benign Environment / Atmosphere* - The benign environment of the ISS creates a risk-tolerant environment for research projects. The availability of humans to oversee operations, combined with the ability to repair faults, reduces the risk of a mission. The availability of an atmosphere allows projects which operate in the pressurized modules to develop their hardware in standard ground-based facilities, without the need to account for the effects of an actual space environment. This results in reduced cost and complexity of the mission.

The benefits of utilizing the ISS are summarized in Table 7.2.

TABLE 7.2 Benefits of the ISS for microgravity research

Resource	Risk	Complexity	Cost	Remote Operations	Visibility
Crew	↓	↓	↓		
Communications			↓	↓	
Long-term experimentation					↓
Power Sources	↓	↓	↓		
Atmosphere	↓	↓	↓		

↓ = reduces challenge

Research aboard the ISS covers multiple fields through several operational modes. Three types of operations were identified: observation, exposure, and iterative experiments. Observation experiments involve equipment and/or astronauts making observations of the Earth or space and providing that information to scientists on Earth. Exposure experiments require the presence of the test artifact in the microgravity environment of the ISS, either in the pressurized areas (e.g., effects of microgravity on humans) or exposed areas (e.g., materials exposure to space radiation). These experiments do not process any data while the artifacts are aboard the ISS; all data is analyzed in ground-based facilities after the exposed artifacts return to Earth. Iterative experiments collect data while aboard the ISS;

this data is used by either the astronauts or the scientists to review the experiment and conduct new tests until the desired results are obtained.

Three main ares of research have been identified: educational, pure science, and space technology. Educational experiments provide methods for students at all levels to become involved with space experiments. Pure science missions provide insight into how systems behave differently in microgravity and full gravity conditions; they address the better understanding of physical laws and biology. Space technology experiments allow research to develop products for use in space missions and facilitate access to space. The thesis specifically addresses the design of iterative space technology missions aboard the ISS.

Chapter 3 introduces three past MIT SSL microgravity programs. The Middeck 0-gravity Dynamics Experiment flew on STS-48 and STS-62 to measure the non-linear dynamics of fluid slush and jointed structures. The Dynamics Load Sensor experiment operated aboard the MIR space station for approximately three years to measure loads of humans as part of the ISS risk mitigation program. The Middeck Active Control Experiment flew on STS-67 and was operated by Expedition One aboard the ISS. The first mission developed dynamics and controls tools for predicting and refining robust, multi-variable control algorithms. The ISS mission of MACE studied dynamics and controls technologies ranging from neural networks to nonlinear dynamic characterization to adaptive reaction wheel isolation.

The MIT SSL Laboratory Design Philosophy was developed from the lessons learned in the development and operation of these experiments. The philosophy calls for new laboratories to exhibit a specific set of features that facilitates research of dynamics and controls experiments. The philosophy is based on the need of a mature control algorithm to be demonstrated in a valid environment; produce repeatable and reliable results; allow the determination of simulation accuracy; and identify performance limitations, operational drivers, and new physical phenomena. For a laboratory to enables demonstration that show characteristics are met, the MIT SSL Laboratory Design Philosophy calls for a laboratory to exhibit the following features:

- **Facilitating Iterative Research** - A laboratory must allow scientists to conduct research following the scientific method: formulate a hypothesis, design an experiment, run tests, collect and analyze data, and compare the results with the hypothesis to demonstrate the problem has been solved or modify the hypothesis to run new tests. This is an iterative process; the research iterates on multiple hypothesis until the desired performance is achieved. Specifically for the dynamics and control experiments conducted at the MIT SSL, a laboratory must allow formulation of new models (dynamics) and algorithms (control) based on the analysis of collected data and is comparison with predicted results.
- **Support of Experiments** - The features that support experiments ultimately facilitate the iterative research process by reducing the time to conduct experiments and providing relevant data. Several features encompass the support of experiments:
  - *Data Collection and Validation* - A laboratory must provide the facilities necessary to collect substantial data which can support or refute the hypothesis. The facility must provide methods to validate the accuracy and precision of the data independently of the original sensors used to collect the data.
  - *Repeatability and Reliability* - The facilities must allow easy repetition of experiments with minimal time invested in setting up and starting new tests. Further, the facility must be reliable such that each test is conducted with reasonable expectations that the environment is controlled and any changes external to the experiment are observed.
  - *Human Observability and Manipulation* - A dynamics and controls laboratory allows humans to directly or indirectly observe every step of the experiment without the need for expensive data analysis when an observation can be used to determine results. Further, the ability of humans to manipulate the facilities allows humans to modify the experiment conditions.
  - *Supporting Extended Investigations* - To allow iterative research, a laboratory must provide scientists with substantial time to analyze data and determine the need of new hypothesis and/or design of new experiments.
  - *Risk Tolerant Environment* - A laboratory for research must provide an environment where a scientist can take risks in the design of their experiments. Specifically in the case of dynamics and control algorithms, the scientists should be able to push the limits of their algorithms to determine the performance limitations without the possibility of a critical failure to the facilities.

- **Support Multiple Investigators** - Aerospace research involves the study of a wide range of areas. Even within the dynamics and control fields, many scientists come up with different hypothesis, all of which could achieve the desired performance. Therefore, a laboratory must support multiple scientists to test their different hypothesis. This enables the use of a laboratory to determine the best solution among the field, rather than provide a single data point within the field.

  For collaborative science to be effective it must allow each scientist to achieve goals they would otherwise not be able to do on their own. The experiments developed for collaborative research must support multiple investigators by design. These experiments must identify the common elements shared between scientists, and allow individuals to add their own components for their specific research. This requires a systematic approach which must clearly define the goals and structures of the collaboration while creating trust between the parties. Effective inter-personal and data communications channels must be established

- **Reconfiguration and Modularity** - The ability to reconfigure an experiment provides benefits to facilitate the iterative research process (redefine the experiments and/or hypothesis) and to support multiple investigators (allow individuals to utilize science-specific components). The use of modular systems allows for simpler methods to allow reconfiguration. Several features are considered for reconfiguration and modularity:

  - *Generic versus Specific Equipment* - The facilities that support collaborative research must identify the generic equipment which supports all scientists involved in the research. Provisions must be made for scientists to provide science-specific equipment. This can be in the form of either hardware or software.

  - *Hardware Reconfiguration* - Hardware reconfiguration refers to the ability to change the hardware not only to support multiple scientists, but also to change the configuration of the facility for individual tests. In the area of dynamics and control the hardware configuration of a test apparatus directly affects the results by changing the dynamics of the system. Hardware reconfiguration also allows the addition of new sensors and actuators to better represent the intended system.

  - *Software Reconfiguration* - The need to support iterations and multiple scientists requires that facilities provide software reconfiguration. The correct use of software reconfiguration can lead to the development of a good platform where multiple scientists can implement their own modules for their specific research area. This development must utilize simple APIs that enhance the collaborative efforts.

- *Physical End-to-End Simulation* - An experiment designed to mature a space technology must exhibit all the relevant physical characteristics of the operational system to reach a TRL 5 or greater. Requiring an experiment to fulfill end-to-end simulation means that the experiment includes the necessary sub-systems and operates in the correct environment to provide realistic operations. No critical elements of a program can be missing in the tests, otherwise the experiment does not satisfy being an end-to-end simulation and the technology cannot advance.

The SPHERES Laboratory for Distributed Satellite Systems was designed specifically for operations aboard the ISS (Chapter 2), following the guidelines of the MIT SSL Laboratory Design Philosophy presented in Chapter 3. Chapter 4 presents detailed information about the features exhibited by the SPHERES testbed to satisfy the philosophy and enable efficient remote operations aboard the ISS.

SPHERES consists of five nano-satellites, metrology and communications hardware, a researcher interface, an astronaut interface, and a guest scientist program to allow multiple researchers to use the facility. In its final configuration, three of the satellites will be aboard the ISS, where the astronauts will conduct tests in 6DOF. Two units will remain in the ground facilities of the SSL where MIT researchers will tests algorithms prior to uplink to the ISS. The guest scientist program provides a simulation which allows researchers outside of the MIT SSL to develop their initial algorithms in house. Operation of the SPHERES satellites (prototypes, left; flight right) aboard the KC-135 RGA is illustrated in Figure 7.1.

Figure 7.1 SPHERES operations aboard the KC-135 RGA

The SPHERES features to fulfill the MIT SSL Laboratory Design Philosophy are:

- **Facilitate the Iterative Research Process**
  - *Multi-layered operations plan* - The SPHERES operations plan directly accounts for the need to conduct iterative research. It provides scientists with multiple steps to incrementally increase the fidelity of their tests, while always ensuring that complete iterations are possible. The need for efficient data collection and ability to update the experiments and/or hypothesis are accounted for at every step.
  - *Continuous visual feedback* - Visual observation of the experiments is available at all steps. In most cases where experiments are conducted in ground-based facility the scientist can directly observe the tests. All ISS missions will be recorded; the video will be made available to scientists within a few days of the test.
  - *Families of tests* - The SPHERES software allows scientist to program a large number of tests into each of their programs to be used in a test session. In this manner, scientists can have ready several experiments which incrementally demonstrate more features of their algorithms, even before the first test. As successful experiments are conducted, the operators can conduct more complex tests, allowing a single session to substantially advance knowledge of a technology.
  - *Easy repetition of tests* - The hardware and software have been designed to allow quick repetitions of tests. Ground-based experiments can be restarted by simply positioning the satellites and using a one-key command. ISS experiments require that the satellite be positioned and then enabled, followed by two step process to ensure the astronaut is ready for the test to start.
  - *Direct link to ISS data transfer system* - By using a direct link to the ISS data transfer system, the SPHERES operational plan minimizes the lag for scientists to obtain the collected data. It also simplifies the operations needed to upload new programs to the ISS facilities.
  - *De-coupling of software from NASA safety controls* - Software that is controlled by NASA requires lengthy reviews before being deployed to the ISS. By not requiring safety reviews, SPHERES allows scientists to make any modifications necessary and upload them to the ISS efficiently.
- **Support of Experiments**
  - *Data Collection and Validation Features*
    - *Layered metrology system* - The SPHERES metrology system provides both high-frequency inertial measurements and low-frequency

absolute measurements to provide scientists with the necessary data to determine the full state of a 6DOF satellite.

  - *Flexible communications: real-time & post-test download* - The communications mechanism allows scientists to define the amount of data they need. If the real-time bandwidth is not sufficient, scientists can download large amounts of data after a test has finished.
  - *Full data storage* - To minimize the effects of errors during wireless transmission, all the data received by the control laptop is stored intact. While the interfaces process some of that data to show information to the operators and scientists, all of the data is available for post-processing.
  - *32 bit floating point DSP* - The use of a 32-bit floating point DSP provides scientists with high precision data and quick calculations to meet the precision needs of their research.

- *Redundant communications channels* - While the use of two communications channels is directly related to creating an end-to-end simulation, it also increases the reliability of the system since the two channels are interchangeable.
- *Test management & synchronization* - Tests with SPHERES can be repeated easily by using the test management software, part of the implementation to support families of tests. The software also synchronizes all the satellites in a multi-unit test.
- *Location specific GUI's* - SPHERES provides two separate interfaces: one for use in ground-based facilities, where the researchers are likely to be the operators, and one for ISS research, where astronauts are the operators. The use specific interfaces allows scientist maximum observability by providing real-time state and debug information in ground-based operations. The ISS interface provides astronauts with the information necessary to understand the objectives of a test and to determine the state of the satellites at all times.
- *Re-supply of consumables* - The ability to re-supply consumables allows tests to be repeated multiple times to collect sufficient data. It creates a risk-tolerant environment because the scientist can design their experiments to push the limits of their algorithms knowing that an error in the tests will not result in the end of the mission due to depletion of all consumables. Further, it enables supporting extended investigations.
- *Operations with three satellites* - The use of three satellites creates redundancy for a large number of DSS experiments which require only one or two satellites to demonstrate the technology (e.g., docking, tethers).

- *Software cannot cause a critical failure* - The design of the satellites guarantees that the software cannot cause a critical failure of the facilities, such that scientists can program any algorithms, regardless of how aggressive it is, enhancing the risk-tolerant environment of SPHERES.

- **Support Multiple Investigators**
  - *Guest Scientist Program* - The SPHERES Guest Scientist Program provides scientists with several tools to support their research both at their home locations and remotely. Specifically, the GSP provides:
    - *Information Exchange* - A clearly defined data path for scientists to collect the data. Further, members of the SPHERES team can provide substantial feedback when operations are conducted at the MIT SSL without the scientists.
    - *SPHERES Core Software* - The core software of SPHERES has been designed with modularity and simple interfaces in mind. It provides all the basic functions to control the satellites, while allowing scientists to program only those parts they are most interested in (e.g., metrology, control, or autonomy).
    - *GSP Simulation* - The GSP simulation allows scientists to predict results of their experiments at their home location.
    - *Standard Science Libraries* - These libraries provide scientists with several routines to simplify the development of their algorithms. Scientists can utilize math, metrology, and control functions developed by the SPHERES team.
  - *Expansion port* - The SPHERES Expansion Port fulfills the need for scientists to utilize custom hardware for their specific science goals.
  - *Portability* - The facilities required to operate SPHERES are highly portable, allowing their operation in a multitude of facilities, including at the home locations of several scientists and special environments such as NASA's RGA and MSFC's Flat Floor. This allows multiple scientists to conduct ground-based tests in the environment that best meets their needs.
  - *Schedule flexibility* - The schedule to conduct research aboard the ISS has been set at periodic intervals of two weeks. While that time cannot be reduced, the SPHERES team is able to manage that time so that scientists can take more time to analyze their data.

- **Reconfiguration and Modularity**
  - *Generic satellite bus* - The SPHERES team identified the generic equipment required for research on DSS algorithms and developed the satellites to represent a generic satellite bus. In this manner, the satellites can

be used to model the operations of a 6DOF satellite representative of standard space missions.

- *Science specific equipment: on-board beacon and docking face* - As an example of specific equipment, and to fulfill the original requirements of the SPHERES project, the satellites provide an on-board beacon and a docking face to demonstrate docking technologies.
- *Generic Operating System* - The core software not only provides with modular interfaces and simple API's, it also creates a generic real-time operating system for dynamics and controls experiments similar to commercially available RTOS's (based on a COTS RTOS).
- *Physical Simulation of Space Environment*
  - *Operation with three units* - The operation with three units aboard the ISS allows SPHERES to model the exact operations of mission that utilize one, two, or three satellites which constitutes a substantial portion of DSS research. Further, operations with three satellites models the essential complexity of missions that utilize more units.
  - *Operation in 6DOF* - SPHERES is able to simulate the space environment with 6DOF operations aboard the ISS.
  - *Two communications channels* - The use of two communications channels in SPHERES directly models distributed satellite systems which are expected to utilize at least one channel for inter-satellites communications and one channel for satellite-to-ground communications.
- *Software interface to sensors and actuators* - While sensors and actuators can be added via the expansion port, SPHERES allows scientists to define their own software interfaces to sensors and actuators. In this manner, a scientist can model a sensor or actuator representative of the operational mission they are researching (of lower bandwidth than the sensors and actuators available in SPHERES).
- *Hardware expansion capabilities* - Hardware elements can be added to the SPHERES satellites via two different locations: the docking port supports passive elements, while the expansion port supports active elements. This allows scientists to develop hardware which better represents the operational mission.
- *FLASH memory and bootloader* - To enable software reconfiguration a custom bootloader was developed. The bootloader allows the full software of a SPHERES satellite to be reprogrammed and stored in FLASH. This enables software changes to range from simple corrections to algorithms to completely new hypothesis.

After development of the SPHERES Laboratory (Chapter 4), it was possible to review the different ways to fulfill the MIT SSL Laboratory Design Philosophy (Chapter 3) for a project specifically intended to operate aboard the International Space Station (Chapter 2). The lessons learned resulted in the development of seven design principles for microgravity laboratories for space technology maturation. These principles, based on the fundamentals of the scientific method, collaborative research, and existing resources of the ISS, apply to laboratories beyond the areas covered in the MIT SSL Laboratory Design Philosophy by identifying those traits that are almost always true of all laboratories, as required by the definition of a principle. The seven principles are:

- *Principle of Iterative Research* - A laboratory must enable scientists to conduct iterative research through repetition of experiments to obtain sufficient data; provide the capability for scientists to analyze that data and compare it with predicted results while on a flexible schedule; and allow reconfiguration of the facilities to allow for changes in experiments and hypothesis.
- *Principle of Enabling a Field of Study* - To enable research in a field of study, it is almost always true that a laboratory needs to support multiple scientists. This includes the need to provide: the ability for scientists to create models and analyze data in their home location; simple operational interfaces; and efficient data transfer mechanisms.
- *Principle of Optimized Utilization* - This principle calls for the correct utilization of the special resources available aboard the ISS in such a way that they add value to the mission. The use of these facilities should not be considered a cost to the mission. The principle identifies five special resources: crew, power, long-term experimentation, and a benign environment/atmosphere.
- *Principle of Focused Modularity* - The facilities of a laboratory almost always include common parts that can be used by a wide range of applications within the field of study of the laboratory. Those parts, the generic equipment, should be identified and designed in a modular fashion so that they can be utilized by as yet unforeseen research.
- *Principle of Remote Operation & Usability* - Because operations aboard the ISS occur in a remote environment where it is practically impossible for the research scientist to be present in the operational environment, a laboratory for research aboard the ISS must provide tools and information to conduct effective runs of experiments, while the scientists need efficient access to data obtained from the experiments for analysis. Ultimately, the operator should become a virtual extension of the scientists aboard the ISS.

- *Principle of Incremental Technology Maturation* - Utilizing the ISS should allow technology maturation to TRL 5 or TRL 6 with the risk and cost increasing incrementally rather than in steep jumps. Successful use of the ISS should allow operational missions with a lower total cost and risk than deployment of the mission directly from ground-based tests.
- *Principle of Requirements Balance* - For a laboratory to succeed, the requirements which arise from the previous principles must be balanced, without one single requirement driving the majority of the cost and effort in development of the mission. The hard requirements, which directly affect the ability to succeed in the mission, must drive the mission efforts. Soft requirements, desired features not directly affecting the success of the mission, should only be implemented when they do not cause the mission to break its constraints and do not contradict any hard requirements.

To enable scientists to use the design principles when developing a new laboratory, a design framework is presented. The framework involves a four-step iterative process. First, the scientist must identify the field of study covered by the laboratory. Next the scientist can determine the functional requirements by designing the laboratory such that it facilitates the iterative research process, enables incremental technology maturation, and utilizes the resources of the ISS correctly. This design can then be refined by identifying the modularity of the system which enhances its capabilities, but does not hinder the ability to satisfy all the mission objectives. The design must also consider the need to support remote operations. The fourth step is to identify the hard and soft requirements, ensure that the hard requirements drive the mission, and then iterate on the design to ensure that the requirements use a balanced amount of cost and effort.

An evaluation framework for members of the proposed NGO that will manage research activities aboard the ISS is also presented. This framework presents guidelines for an NGO evaluator to determine the effective use of the ISS while taking into account the success of the mission and the achievement of technology maturation.

Chapter 6 presents the results from operations of the SPHERES laboratory up to date and assesses the laboratory through the design and evaluation frameworks. SPHERES currently supports multiple research programs, including: mass property identification,

autonomous rendezvous and docking, TPF formation flight maneuvers, TPF multi-stage control, tethered formation flight, and the Mars orbit sample retrieval (MOSR) program. The first three programs are scheduled to be studied during the first phase of ISS research, while the later three programs will require delivery of new hardware to the ISS. The results to date include tests at the MIT SSL, NASA's RGA, and MSFC's Flat Floor. Tests at the MIT SSL have included formation flight algorithms and communications, docking control, mass ID, fault detection identification and recovery, tethered formation flight, and tests for MOSR. Five week-long campaigns aboard NASA's RGA (KC-135) consisted of tests on metrology (inertial and global), single satellite 6DOF controls, formation flight, docking, mass system ID, thruster ID, tracking, TPF maneuvers, and distributed control algorithms. The two week-long campaigns at the MSFC Flat Floor consisted of TPF formation flight, docking, and tethered tests.

The review of these operations and results provide the necessary information to evaluate the design of SPHERES utilizing the design framework; the existing design can be considered a first iteration of the design process, while this evaluation provides guidelines for further iterations. The review of the SPHERES design based on the design framework provides the following conclusions:

- **Step 1 - Identify a Field of Study**

  For SPHERES to obtain a benefit from supporting multiple investigators, it must allow research in at least five of the research areas identified within DSS. These include specific missions (docking and rendezvous, formation flight, separated spacecraft telescopes, tethered spacecraft and sample capture) and several areas of study (metrology, control, autonomy, artificial intelligence, communications, and human machine/interfaces). The first ISS mission of SPHERES is scheduled to conduct research on metrology, control, and autonomy algorithms. These will be specifically applied to docking missions and TPF formation flight. Future missions will support tethered spacecraft and sample return, as well as artificial intelligence programs. It is unclear if SPHERES will be able to support separated spacecraft telescope and human machine/interfaces research aboard the ISS. Therefore, SPHERES does allow study in a sufficient number of research areas.

- **Step 2 - Determine Functional Requirements**

The SPHERES laboratory closely follows the guidelines of the Principle of Iterative Research. The metrology and communications systems provide sufficient data collection and transfer features to facilitate iterative research. SPHERES clearly allows not only repetition of experiments, but also modification of both the experiments and the hypothesis. The SPHERES operations plan has demonstrated great flexibility. Not only has iterative research been conducted at the MIT SSL, but also at two remote facilities. At all locations, the SPHERES operations plans work to minimize the overhead time to collect data and update modifications. Each of the facilities has been used to successfully accomplish iterations.

SPHERES obtains substantial value from the correct use of most of the resources available at the ISS. SPHERES slightly under utilizes crew time, but the benefits obtained from its use greatly enhance the risk-tolerant environment, its ability to facilitate the iterative research process, and provide substantial feedback to scientists. The total power of SPHERES is minimal, at only approximately 50W for the complete system; but because it does not use ISS power sources, it obtains no value from that resource of the station. The correct use of telemetry, with flexible download data rates and limited data sizes, allow it to minimize data transfer times and maximize the capabilities obtained from communications. The duration is considered slightly short, although well within the expected lifetime of an ISS mission. Lastly, SPHERES utilizes the ISS environment to a large extent; reducing its development and operational costs substantially, while ensuring a risk-tolerant environment.

SPHERES enables the maturation of metrology, controls, and autonomy algorithms implemented through software to reach TRL 6. The satellites provide the necessary understanding of the interactions between the sub-systems of a satellite through empirical tests under stressful operating conditions. But the facilities do not allow maturation of hardware technologies to TRL 6 unless these hardware elements can be operated through the SPHERES Expansion Port and the resources exist to deliver them to the ISS. Only a limited number of missions (those of the same scale and properties as SPHERES) can utilize the laboratory to advance to TRL 7.

- **Step 3 - Refine Design**

  The SPHERES satellites as a whole provide modularity and reconfiguration by being identical satellites, interchangeable with each other, and by using the docking port and expansion port to allow reconfiguration. The satellite sub-systems, on the other hand, do not provide modularity since, in most cases, that would violated the 1 MLE mass/volume constraint for the system with little added value. The software sub-system is highly reconfigurable and modular as a direct result of the mission goals. The metrology beacons are modular in their ability to be interchanged and reconfigured with ease to

provide accommodate different operational environments of the global metrology system. The laptop transceiver enables the use of the SPHERES facilities through any standard PC serial port at many locations.

To satisfy the Principle of Remote Operations and Usability, SPHERES provides two separate interfaces to operate the facilities: a ground-based interface and an ISS interface. Further, the SPHERES simulation was developed to account for the remote location of scientists who are not members of the SPHERES team. The ground-based interface was designed for operations at the MIT SSL, NASA RGA, MSFC Flat Floor, and other facilities where the operators are either the researchers and/or members of the SPHERES team. This interface provide simple, single-key-stroke, operations for all common tasks and incorporating program upload directly into the interface. The availability of optional windows with real-time state and debug data allows the interface to provide relevant data when the operators are the research scientist. The interface does require the operator to initiate data storage, therefore creating the potential situation where data is not stored due to operator error, but to minimize that the interface clearly indicates when data is being saved. Further, the SPHERES team developed several Matlab functions to analyze the data. The SPHERES simulation and the information provided with the Guest Scientist Program allows the scientist to create models of their experiments and compare the information.

The ISS interface utilizes several steps to ensure the operator selects the correct program and test and is aware of the expected results of each experiment. The ISS interface stores data immediately upon starting. Regardless of the operator's actions, the program will save all outgoing and incoming raw data, ensuring the data is safe regardless of the operator's actions. The flight GUI presents information to the operator to always be aware of the state of the satellites through a status bar. Descriptions of the tests are presented to allow the operators to know expected results and make decisions on the test performance. By providing sufficient details on the test, the interface reduces the dependency of real-time communications with the researcher. Further, the interface presents a questionnaire to the astronaut at the end of each test, requiring the astronaut to provide feedback The astronaut is also allowed to enter notes freely after the questionnaire effectively creating an electronic laboratory notebook.

- **Step 4 - Review Requirements and Design**

  A total of 31 system-level functional requirements were identified for this iteration of the SPHERES laboratory design. Of those 21 are hard requirements and 10 soft requirements. Further review into the efforts needed to implement the requirements resulted in the following conclusions.

  - The requirements relevant to 6DOF operations did not require substantial trade-off's in the implementation of the sub-systems.

- The hard requirements to *operate aboard the ISS* and to *facilitate iterations* by allowing the *collection of substantial data* and *enabling reconfiguration* forced trade-offs in the following implementation decisions:
  - *One-time Use Batteries* - The need to operate aboard the ISS and meet safety requirements prevented the use of rechargeable batteries which would have provided benefits to iterative research, long-term experimentation, and modularity.
  - *Custom Metrology System* - The decision to develop a custom metrology system resulted in the investment of too many resources and effort to this sub-system, while the resources allocated to other sub-systems (such as propulsion and the expansion port) were not balanced.
  - *Propellant Selection (CO_2)* - The selection of the propellant correctly balanced the need to support iterative research and create a representative environment for technology maturation even though it required increased resources to operate aboard the ISS.
  - *Expansion Port* - The development of the expansion port towards the end of the program implementation prevented sufficient resources to be used in developing a simpler interface. Still, the port has allowed the development of several expansion items to increase the research possibilities of the facility.
  - *Communications Channel Frequency* - The need to operate aboard the ISS without creating any bureaucratic issues due to safety concerns by NASA required the selection of wireless communications in the 900MHz range, which is limited to 56.6kbps, rather than the use of 802.11b hardware which allows over 1Mbps. This trade-off was necessary to ensure the iterative research process is efficient when using SPHERES.
- The soft requirements did not account for substantial effort or use of resources, since the most important design decisions were not driven by requirements not essential to the mission.

The current iteration of the SPHERES laboratory design meets the principles to a large extent. The facility enables research on multiple areas within Distributed Satellite Systems. The facilities enable iterative research with few limitations, minimal overhead, and flexible schedules. The iterative nature of the research in a microgravity environment allows technologies to mature incrementally to TRL 5 or TRL 6. SPHERES achieves this by utilizing many of the resources available aboard the ISS. Both the software and hardware allow reconfiguration, although hardware changes do require delivery of the new

items to the ISS. The software is highly modular, allowing scientists to work only on their specific area of study. Overall, the resources and effort put into the implementation of the features are correctly balanced.

A new iteration of SPHERES would consider utilizing rechargeable batteries to benefit the iterative research process by allowing more repetitions, ensure long-term experimentation, and lower its need for launch mass and storage volume aboard the ISS. If the metrology system were still in the design stages, the selection to use a custom system would be revised; since it is already implemented, the program should ensure that substantial metrology research occurs to warrant the amount of resources used to implement the subsystem. Lastly, the expansion port should be reviewed to provide better interfaces that allow easier development of expansion items.

Finally, the evaluation framework was applied to SPHERES. An ISS evaluator concentrates on the correct utilization of ISS resources, the ability to demonstrate technology maturation, and the scope of the program. The review of SPHERES concludes that:

- SPHERES enables iterative research by allowing repetitions, experiment modification, and hypothesis modifications. Reservations exist due to the consumables required for operations.
- SPHERES has been correctly designed to support multiple investigators conducing research aboard the ISS. The resources necessary from the ISS are acceptable, and the necessary programs exist for efficient data transfer to scientists.
- The utilization of crew, communications, long-term experimentation, and benign environment resources is acceptable. The crew time is small but provides important benefits to the mission. The use of ISS telemetry systems has been well incorporated into the program. Long-term experimentation, while limited by consumables, is of the correct length (six months to a year). The benign environment is fully accounted for. SPHERES does not use the power resources of the ISS correctly, and its design should be reviewed to analyze the possibility to make better use of this resource.
- SPHERES offers little modularity from the perspective of the ISS. The global metrology system is the only component which can be easily used by other programs beyond SPHERES. The communications system could provide modularity, but utilizes too much custom hardware. The satellites

enable research by multiple scientists, but cannot be considered modules for use by other programs independently from SPHERES.

- From the ISS NGO perspective, the SPHERES team has accounted for all the necessary tools to allow successful remote operations of the SPHERES facilities, even without real-time communications between the scientists and the operator.
- SPHERES can allow technology demonstrations to TRL 5 for a large number of programs. TRL 6 can also be achieved, for a limited number of programs, especially if new hardware can be delivered. SPHERES cannot be considered a laboratory to demonstrate a technology to TRL 7.

As a result, an ISS NGO evaluator following the evaluation framework would conclude that SPHERES can operate aboard the ISS, but several reservations exist which should be addressed. The long-term experimentation is greatly challenged by the consumables required in SPHERES, and steps should be taken to minimize the dependence on consumables. Specifically, the use of rechargeable batteries should be considered. Further, the SPHERES team should develop contingency plans which detail operations and science benefits from the use of SPHERES without propellant. While SPHERES provides little modularity for the ISS, any changes to the modularity would be too large and costly for the program; therefore, the evaluator should not propose the changes, but should consider the limited modularity when evaluating other projects. None of the reservations are substantial enough to preclude the use of SPHERES aboard the ISS; to the contrary, SPHERES fulfills a substantial part of the design principles. It utilizes the crew of the ISS wisely, allowing astronauts to become scientists in space. Further, it opens the use of the ISS to multiple scientists who would otherwise not be able to conduct research in a microgravity environment. Therefore, the SPHERES laboratory enhances the science capabilities of the ISS allowing it to better achieve its primary objective: *to provide and "Earth orbiting facility that houses experiment payloads, distributes resource utilities, and supports permanent human habitation for conducting research and science experiments in a microgravity environment."* [NASA, 1998]

7.2 Contributions

The research presented in this thesis extracts the fundamental characteristics that a laboratory must exhibit and condenses them into a set of design principles for the development of space technology maturation laboratories. The principles also derive from the lessons learned in the design, implementation, and operation of the SPHERES laboratory, which provides a special research environment for the maturation of technologies related to distributed satellite systems. The following are the specific contributions identified from the research:

- Identified the fundamental characteristics of a laboratory for space technology maturation and formalized the features required in the design of a laboratory to exhibit said characteristics.
  - Identified the need for a laboratory to support iterative research utilizing the definition of the scientific method. While laboratories are continuously created at research institutions, no previous literature specifically addresses the need for a laboratory to enable every step of the scientific method. This thesis formally calls for a laboratory to support iterative research, including development of a hypothesis, design and operation of experiments, data collection and analysis, and the ability to modify the hypothesis until the desired performance is met.
  - While literature on the Design of Experiments (DOE) emphasizes the need to collect data at operationally interesting points, this thesis goes a step beyond DOE towards the development of a laboratory that allows repetitions of experiments at all data points. A laboratory reduces the dependency on DOE techniques to minimize the number of experiment runs.
  - The review of existing research facilities, definition of a laboratory, and collaborative science identifies the need for a laboratory to enable research of a field of study by supporting multiple scientists. Given the restricted access to a space environment, projects which support space technology maturation must enable research of a field of study.
  - Determined the use of the International Space Station as a location which helps reduce the major challenges of space technology research: risk, complexity, cost, remote operations, and visibility.
- Established a set of principles to guide the design of research laboratories for space technology maturation aboard the International Space Station.

- The principles address the need to facilitate the iterative research process, enable a field of study, and enable incremental technology maturation, coupled with the need to account for remote operations aboard the International Space Station.
  - Enables the use of the ISS to incrementally mature technologies in such a manner the total risk and cost of a mission can be reduced.
  - The principles are based on a comprehensive review of ISS research, existing resources, and programed upgrades to ensure they account for the correct utilization of the station.
- Developed a design framework for application of the principles by research scientists.
- Developed an evaluation framework which can be used to respond in part to the call by the National Research Council to define the functions of a non-governmental organization which manages the science aboard the International Space Station.
  - The application of these principles in the design of new laboratories for research aboard the ISS can result in an expanded base of scientists who can conduct research under the microgravity environment of the ISS.
  - Calls for a change in attitude towards the use of resources aboard the International Space Station: rather than treat the use of those resources as a cost which must be minimized, the resources should be treated as added value, which must be maximized.

• Designed, implemented, and operated the SPHERES Laboratory for Distributed Satellite Systems.

- Provides a facility for multiple researchers to advance metrology, control, and autonomy algorithms in a microgravity environment able to meet the requirements for TRL 5 or TRL 6 maturation.
- Demonstrates the implementation of miniature embedded systems to support research by multiple scientists through the creation of a *flexible* embedded system which allows implementations of multiple types of algorithms.
  - Developed a real-time operating system with modular and simple interfaces for use by multiple scientists.
- Demonstrates the ability to create generic equipment while enabling future expansion through both hardware and software.
- Approaches virtual presence of the scientists in a remote location by providing the necessary interfaces to present the operator with the necessary

knowledge to be an integral part of the research process. Further, it creates a laboratory environment by allowing astronauts to provide feedback to the scientists.

7.3 Future Work

The recommendations for future work concentrate on two main areas: further development of the design principles and operations of the SPHERES laboratory. While the concepts behind the laboratory design principles have been reviewed with more than twenty experiments conducted during Expedition 6 aboard the ISS, the actual application of the principles lacks data points. Validation of the design principles by their application and evaluation in several new programs would provide the necessary validation. The design framework provides a few initial quantitative measures to utilize in the application of the principles. These measures are extracted from available information about product platforms and the review of the International Space Station existing capabilities. These quantitative models should be validated with new data as experiments are designed following the guidelines of the design principles.

These principles concentrate on the creation of the engineering and technical aspects of the program. This thesis does not introduce the need to also provide financial support for the laboratories, as it assumes that a well designed laboratory will have high demand within the scientific community. A parallel study, which considers both the development and maintenance costs of these type of laboratories should be created.

The ability of scaled experiments aboard the ISS to demonstrate the maturity of space technologies depends directly on the ability of scientists to prove that these models are representative of actual mission. To this end, research should be conducted to develop a set of *scaling laws*, equivalent to those used in aeronautical engineering when testing with wind-tunnels, which enable scientists to scale the results of models used aboard the ISS to for application in full-scale space missions.

Due to circumstances outside the control of the SPHERES program, its operations aboard the ISS have not taken place yet. Therefore, future work must evaluate ISS operations to corroborate the expected results presented in the evaluation of SPHERES in Chapter 6. While the usefulness of the Guest Scientist Program has been demonstrated through multiple ground-based programs, the ability of SPHERES to facilitate the iterative research process through complete ISS iterations are yet to be demonstrated. Future work must support or refute the ability of SPHERES to minimize the overhead time and maximize the science time available during each research cycle. The lack of demonstrations in a space environment has also prevented any algorithms developed with the use of SPHERES to mature to TRL 5 or TRL 6. Therefore, continued research is necessary to confirm that SPHERES can help mature technologies to those levels.

Work should also be performed to enhance the capabilities of the SPHERES facilities:

- The SPHERES metrology system requires continued research to ensure it can provide the necessary accuracy and precision in a 3D environment. While the system has been tested and its capabilities demonstrated in 2D environments, 3D tests have been limited to operations aboard the reduced gravity airplane (KC-135) and limited tests at the MIT SSL.
- Future work to upgrade the software capabilities of SPHERES is also recommended. The current software concentrates on the ability to test metrology, control, and autonomy algorithms. Future work should allow scientists to test different communications protocols and test the real-time requirements of operating systems for distributed satellite systems.
- As clearly indicated by the frameworks, the SPHERES satellites should be upgraded to use rechargeable batteries.

Future research to be conducted with the SPHERES laboratory already includes tethered formation flight, multi-stage optical formation flight telescopes, and the Mars orbital sample return. The development of active docking ports is also planned. An area missing to research with SPHERES is human/machine interfaces. SPHERES provides the unique capability for humans aboard the ISS to control a 6DOF satellite with minimal risks in a full microgravity environment. This enables the creation of training environments where humans can use advanced interfaces (e.g. special joysticks) to control 6DOF satellites.

SPHERES allows research on the human factors engineering aspects (e.g. design of the joysticks) as well as actual training of astronauts for general or specific missions once an interface has been selected. The possibility for the SPHERES satellites to support astronaut activity aboard the ISS via the use of wireless video should also be considered.

7.4 Concluding Remarks

This thesis concentrates on the development of seven design principles for the design of space technology maturation laboratories to operate aboard the International Space Station. The research required substantial background information on previous space programs, other microgravity facilities, other remote facilities, the scientific method, collaborative research, and NASA Technology Readiness Levels. The resulting principles address the most basic mission of the International Space Station: to conduct research in a microgravity environment.

But all this research would not have been complete without the lessons learned from the design, implementation, and operation of the SPHERES testbed. The words *design, implementation, and operation* appear multiple times in this thesis. They are the force behind the development of the SPHERES prototype system: teaching undergraduate students the complete design process in the form of CDIO - conceive, design, implement, operate. The road traversed through the development of SPHERES was so rich with information on the requirements of a laboratory because the teaching staff was learning together with the students. While staff could provide simple implementation answers, the staff continuously evaluated the *why's* of the design; this knowledge represents the lessons learned from SPHERES which resulted in the development of the design principles. Therefore, the work in this thesis represent the knowledge acquired not only by the author, but also by thirteen undergraduate students at MIT, over a dozen staff members, and the several guest scientists who participate in the SPHERES Guest Scientist Program.

The laboratory environment created by SPHERES will provide many scientists with the ability to demonstrate their algorithms in space, something they would not be able to do

otherwise. Expansion items will reach the ISS so that the SPHERES facilities can better represent new missions. This will allow the study of the origins of our galaxy through missions such as TPF, make autonomous re-supply of spacecraft a reality, and even enable the development of technologies, such as tethers, to permit extended human travel into space.

REFERENCES

[Adler, 2004] Adler, N, Shani, A.B., Styhre, A, "Collaborative Research in Organizations", SAGE Publications, London, 2004

[Amir, 1999] Amir, A R, Baroni, G, Pedrocchi, A, Newman, D J, Ferrigno, G, Pedotti, A, "Measuring astronaut performance on the ISS: advanced kinematic and kinetic instrumentation", Proceedings of the 16th Instrumentation and Measurement Technology Conference, IEEE, May 1999, Vol 1, pp 397 - 402

[Amir, 2000] Amir, A R, Newman, D J, "Research Into the Effects of Astronaut Motion on the Spacecraft: a Review", Acta Astronautica, Elsevier Science Ltd, Great Britain, 2000, Vol 47, No 12, pp 859-869

[Antony, 2003] Antony, Jiju, "Design of experiments for engineers and scientists", Butterworth-Heinemann, Burlington MA, 2003

[ARF, URL] Antarctic Research Facility, Florida State University "Home"

- http://www.arf.fsu.edu/

[Ashley, 2004] Ashley, M.C.B., Burton, M.G., Lawrence, J.S. & Storey, J.W.V., "Robotic telescopes on the Antarctic plateau", School of Physics, University of New South Wales, Sydney, Australia, 2004

[Astronautix] Encylopedia Astronautica, "Salyut", astronatuix.com

- http://www.astronautix.com/project/salyut.htm

[BAS1] "Globa Science in the Antarctic Context", British Antactic Survey, Cambridge, UK

[BAS2] "Antarctic Science in the Global Context", British Antarctic Survey, Cambridge, UK

[Beach, 1992] Beach, David P, Alvager, Torsten K E, "Handbook for Scientific and Technical Research", Prentice Hall, Englewood Cliffs JH, 1992

[Beichman, 1999] Beichman, C A, Woolf, N J, and Lindensmith, C A, “The Terrestrial Planet Finder (TPF): A NASA Origins Program to Search for Habitable Planets,” JPL Publication 99-3, 1999

[Bemporad, 2002] Bemporad, A, Borrelli, F, and Morari, M, “Model Predictive Control Based on Linear Programming - The Explicit Solution,” IEEE Transaction on Automatic Control, Vol. 47, No. 12, pp. 1974-1985, December 2002

[Berkovitz, 2003] Berkovitz, Dustin, Kong, Edmund, Miller, David W, "System Identification of the SPHERES Autonomous Rendezvous and Docking Testbed", AIAA Space 2003 Conference, Long Beach CA, 2003

[Belew, 1977] Belew, Leland F, "Skylab, Our First Space Station", NASA Scientific and Technical Information Office, Washing DC, 1977

[Blaurock, 1999] Blaurock, C, Kenny, S, Miller, D, Yung, J, “Nonlinear Modeling and Control for the Middeck Active Control Experiment Reflight,” AIAA Space Technology Conference & Exposition, AIAA 99-4589, Albuquerque, NM, Sept. 28-30, 1999

[Burrough, 1998] Burrough, B, "Dragonfly: NASA and the Crisis Aboard Mir", Harper Collins Publishers, New York, NY, 1988

[Burton, 2004] Burton, M.G., "Astronomy in Antarctica", in Organizations and Strategies in Astronomy, Volume 5. Series editor Andre Heck, Kluwer, 2004

[Campbell, 1995] Campbell, M E, Grocott, S C O, How, J P, Miller, D W, Crawley, E F, "Verification procedure for on-orbit controllers for the MIT Middeck Active Control Experiment", Proceedings of the American Control Conference, Seattle WA, June 1995, Vol 5, pp 3600 - 3605

[Campbell, 1999] Campbell, M E, How, J P, Grocott, S C O, and Miller, D W, “On-Orbit Closed-Loop Control Results for the Middeck Active Control Experiment,” AIAA Journal of Guidance, Control, and Dynamics, Vol. 22, No. 2, Mar-Apr. 1999.

[Chen, 2001] Chen, A, Saenz-Otero, A, Hilstad, M, Miller D, ""Development of Formation Flight and Docking Algorithms Using the SPHERES Testbed", 15th Annual USU Conference on SMall Satellites, Utah State University, UT, SSC01-VIIIa, August 13-26, 2001

[Cook, 1997] Cook, H. E., "Product Management: Value, Quality, Cost, Price, Profits, and Organization", Chapman & Hall, London & New York, 1997

[Crawley, 2003] Crawley, Ed, "Introduction to System Architecture: Architecture to Value", Massachusetts Institute of Technology, ESD.34J Lecture notes, January 9, 2003, Rev 2.0

[Cunninghman, 1970] Cunningham, R E, Cooper C G, Dilham, H B, et al, "Ocean Engineering Research Ship Systems", Massachusetts Institute of Technology, Report No 70-9, vol 1, Cambridge MA, 1970

[Davenport, 1999] Davenport, S, Davies, J, Grimes, C, "Collaborative Research Programmes: Building Trust from Differences", Technovation v19, Pergamon, 1999

[Durham, 2004] Durham, H J, "ISS Program Infrastrure Upgrades to Enhance Science Return: Current, Planned & Potential", 2004 IEEE Aerospace Conference, Montana, USA, IEEAC paper #1083

[deSouza, 2004] de Souza, C R B, Redmiles, D, Cheng, L T, et al, "How a Good Software Practice Thwarts Collaboration - The Multiple Roles of APIs in Software Development", SIGSOFT '04, ACM, Newport Beach, CA, Oct 31-Nov 6, 2004

[Elzinga, 1993] Elzing, A, "Changing Trends in Antarctic Research", Kluwer Academic Publishers, Dordrecht, The Netherlands, 1993

[Emond, 2000] Emond, J, "The Spacelab Accomplishments Forum", NASA, Washington DC, 2000

[Enright, 2004] Enright, J, Saenz-Otero, A, et. al., "The SPHERES Guest Scientist Program: Collaborative Science On the ISS", IEEE Aerospace Conference 2004, Big Sky, Montana, Paper #1296, March 2004

[Ernst, 2003] Ernst, Michael, "Retrospective and Iterative Design", Science of Design: Software-Intensive Systems Workshop, NSF Computer and Information Science and Engineering Directorate, University of Virginia, 2003

[Fisher, 1935] Fisher, Ronald A, "The Design of Experiments", Oliver and Boyd, London UK, 1935

[Gauch, 2003] Gauch, Hugh G Jr, "Scientific Method in Practice", Cambridge University Press, Cambridge UK, 2003

[Graettinger, 2002] Graettinger, C P, Garcia, S, Siviy, J, et al, "Using the Technology Readiness Levels Scale to Support Technology Management in the DoD's ATD/STO Environments", Carnegie Mellon Software Engineering Institute, Pittsburgh, PA, 2002

[Grocott, 1994] Grocott, S, How, J, Miller, D, MacMartin, D and Liu, K, "Robust Control Design and Implementation on the Middeck Active Control Experiment," AIAA Journal of Guidance, Control, and Dynamics, Vol. 17, No. 6, pp. 1163-1170, Nov-Dec 1994

[Hablani, 2001] Hablani, H B, Tapper, M, Dana-Bashian, D, "Guidance Algorithms for Autonomous Rendezvous of Spacecraft with a Target Vehicle in Circular Orbit," Proc. of the AIAA Guidance, Navigation, and Control Conference and Exhibit, A01-37204, AIAA 2001-4393, Montreal, Canada, August 2001

[Hagopian, 1998] Hagopian, J, Maxwell, T, Nahay, E, "NASA/MIR Phase 1: A Lesson in Long Duration Mission Planning and Operations", SpaceOps 98, Tokyo, Japan, paper 2m018, June 1-5, 1998

[Haux, 1982] Haux, G F K, "Subsea Manned Engineering", Baillière Tindall, London, UK, 1982

[Hilstad, 2003] Hilstad, M O, "A Multi-Vehicle Testbed and Interface Framework for the Development and Verification of Separated Spacecraft Control Algorithms", Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, June 2003

[Hilstad, 2003a] Hilstad, M., Enright, J., Richards, A., "SPHERES Guest Scientist Program Interface Document", MIT Space Systems Laboratory, 2003

[How, 1997] How, J, Glaese, R, Grocott, S, Miller, D, "Finite element model-based robust controllers for the middeck active control experiment (MACE)", IEEE Transactions on Control Systems Technology, IEEE, January 1997, Vol 5 , Issue 1, pp 110 - 118

[Hupfer, 2004] Hupfer, S, Cheng, L, Ross, S, Patterson, J, "Introducing Collaboration into an Application Development Environment", Computer Supported Cooperative Work '04, ACM, Chicago IL, November 6-10, 2004

[Huxham, 1996] Huxham, C., "Creating Collaborative Advantage", SAGE Publications, London, 1996

[IMBOSS, URL] IMBOSS Documentation

- http://www.rm.iasf.cnr.it/ias-home/IMBOSS/imbossdocs.html

[Intel, 2003] "Intel Research Network: A New Model of Industry-Unversity Collaboration", Intel Research and Development, USA, 2003

[JAMIC, URL] Japan Microgravity Center, Page automatically translated from Japanese

- http://www.jasma.info/SisetsuShoukai/JAMIC.html

[Jilla, 2002] Jilla, C, "A Multiobjective, Multidisciplinary Design Optimization Methodology for the Conceptual Design of Distributed Satellite Systems", Doctoral Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 2002

[Kraut, 1988] Kraut, R, Egido, C, "Patters of Contact and Communication in Scientific Research Collaboration", Proceedings of the 1988 ACM conference on computer-supported cooperative work, p1-12, Portland Oregon, 1988

[Kong, 2004] Kong, Edmund M C, Saenz-Otero, Alvar, et al, "SPHERES as a Formation Flight Algorithm Development and Validation Testbed: Current Progress and Beyond", 2004 Formation Flight Symposium, Washington DC, September 2004

[Kong, 2004a] Kong, E M C, Hilstad, M O, Nolet, S, and Miller, D W, "Development

and verification of algorithms for spacecraft formation flight using the SPHERES testbed: application to TPF," SPIE Astronomical Telescope 2004, Paper No. 5491-34, Glasgow, Scotland, June 2004

[Larson, 1992] Larson, Wiley J, Wertz, James R, "Space Mission Analysis and Design", Microcosm Inc, Torrance CA, 1992

[LeGris, 2000] LeGris, J, Weir, R, Browne, G, Gafni, A, Stewart, L, and Easton, S, "Developing a model of collaborative research: the complexities and challenges of implementation", International Journal of Nursing Studies, Volume 37, Issue 1, February 2000

[Lindensmith, 2003] Lindesmith, C A (Editor), "Technology Plan for the Terrestrial Planet Finder", NASA and Jet Propulsion Laboratory, California Institute of Technology, JPL Publication 03-007, Rev A, March 2003

[Littler, 1995] Littler, D, Leverick, F, Bruce, M, "Factors Affecting the Process of Collaborative Product Development: A Study of UK Manufacturers of Information and Communications Technology Products", Journal of Product Innovation Management, v12-1, p16, January 1995

[Lorenzen, 1993] Lorenzen, Thomas J, Anderson, Virgil L, "Design of Experiments, A No-name Approach", Marcel Dekker, New York NY, 1993

[Mankins, 1995] Mankins, J, "Technology Readiness Levels", Advanced Concepts Office, Office of Space Access and Technology, NASA, April 1995

[Mather, 1998] Mather, J C, et al., "The Submillimeter Frontier: A Space Science Imperative," Astrophysics 1998, astro-ph/9812454

[Matossian, 1996] Matossian, M., "Design for Success: Optimizing the Earth Observing System for Performance and Cost with Managed Risk," Proceedings of the 46th International Astronautical Congress, IAF-95-Q.4.05, 1995

[Mead, 1988] Mead, R, "The design of experiments", Cambridge University Press, Cambridge UK, 1988

[Merriam-Webster, URL] Merriam-Webster Online Dictionary

- http://www.webster.com/

[Merrill-Sands, 1996] Merril-Sands, D., Sheridan, B., "Developing and Managing Collaborative Research Alliances: Lessons from a Review of the Literature", Simmons Institute for Leadership and Change, Simmons College, Boston, MA, 1996

[Meyer, 1997] Meyer, M, Lehnerd, A, "The power of Product Platforms", The Free Press, New York, 1997

[Miller, 1992] Miller, D W, "The MODE Family of Facility Class Experiments: Conducting Cost Effective Engineering Research in the Shirt Sleeve Environment of the Middeck," Flight Experiment Technical Interchange Meeting, Monterrey, CA, Oct., 1992

[Miller, 1995] Miller, D W, deLuis, J, Stover G, How, J P, et al, "The Middeck Active Control Experiment (MACE): using space for technology research and development" Proceedings of the American Control Conference, Seattle WA, June 1995, Vol 1, pp 397 - 401

[Miller, 1996] Miller, D. W., et al, "The Middeck Active Control Experiment (MACE): Summary Report", MIT Space Engineering Research Center, June 1996, SERC #7-96

[Miller, 1998] Miller, D W, How, J P, Campbell, M E, Grocott, S C O, Liu, K, Glaese, R M, and Tuttle, T, "Flight Results from the Middeck Active Control Experiment (MACE)," AIAA Journal, Vol. 36, No. 3, March 1998, pp. 432-440

[Montgomery, 1991] Montgomery, Douglas C, "Design and analysis of experiments", Wiley, New York NY, 1991

[Mosher, 2000] Steuer, R E.” Multiple Criteria Optimization: Theory, Computation and Application”. Wiley, New York, 1986

[Myers, 1996] Myers, J.D.; Fox-Dobbs, C.; Laird, J.; Dai Le; Reich, D.; Curtz, T., "Electronic laboratory notebooks for collaborative research", Proceedings of the 5th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, vI., p47-51, 19-21 June 1996

[Myers, 2001] Myers, J, Mendoza, E, Hoopes, B, "A Collaborative Electronic Notebook", Proceedings of the IASTED International Conference on Internet and Multimedia Systems and Applications (IMSA 2001), Honolulu, Hawaii, August 13-16, 2001

[NASA, 1998] "International Space Station Familiarization", Mission Operations Directorate, Space Flight Training Division, NASA, Houston TX, July 1998

[NASA, 2000] "Science and Technology Research Directions for the International Space Station", Office of Life & Microgravity Sciences and Applications, NASA, Houston, TX, January 2001

[NASA, 2000a] "Space Shuttle Program, Payload Bay Payload, User’s Guide", NASA Lyndon B. Johnson Space Center, Houston TX, NSTS 21492, December 2000

[NASA, 2000b] "ISS User’s Guide - Release 2.0", NASA, Houston TX, 2000

[NASA, 2004] NASA, "ISS Technical Confguration", July 23, 2004

[NASA, 33897] "Experiment Design Requirements and Guidelines, NASA 931 KC135A", Aircraft Operations Division, NASA Lyndon B. Johnson Space Center, Houston TX, AOD 33897, February 2003

[NASA, 33898] "Interface Control Document, NASA 931 KC135A", Aircraft Operations Division, NASA Lyndon B. Johnson Space Center, Houston TX, AOD 33898, May 2004

[NASA, 33899] "JSC Reduced gravity Program, User's Guide", Aircraft Operations Division, NASA Lyndon B. Johnson Space Center, Houston TX, AOD33899, March 2004

[NASA, URL1] NASA, "International Space Station: Science", accessed URLS:

- Home: http://spaceflight.nasa.gov/station/science/index.html
- Expedition 6 Experiments: http://spaceflight.nasa.gov/station/science/experiments/exp6_expmt.html

[NASA, URL2] NASA, "Research on Station --- PCG Single-locker Thermal Enclosure Systems":

- http://spaceresearch.nasa.gov/research_projects/ros/pcgstes.html
- http://spaceresearch.nasa.gov/research_projects/ros/pcgstesop.html

[NASA, URL3] NASA Glenn 2.2 Second Drop Tower

- http://microgravity.grc.nasa.gov/drop2/

[NASA, URL4] NASA Space Shuttle Payload Information Source

- http://shuttlepayloads.jsc.nasa.gov/flying/overview/overview.htm

[NASA, URL5] NASA: History of Shuttle-MIR Home Page

- http://spaceflight.nasa.gov/history/shuttle-mir/index.html

[NASA, URL6] NASA NBL Website

- http://www1.jsc.nasa.gov/dx/dx12/

[NASA, URL7] "The Neutral Buoyancy Simulator", NASA Marshall Space Flight Center Fact Sheets

- http://www1.msfc.nasa.gov/NEWSROOM/background/facts/nbs.htm

[NASA, ULR8] "HSF - International Space Station", NASA Human space Flight

- http://spaceflight.nasa.gov/station/index.html

[Newman, 2001] Newman, D J, Amir, A R, Beck, S M, "Astronaut-Induced Distrubances to the Microgravity Environment o the Mir Space Station", Journal of Spacecraft and Rockets, AIAA, July-August 2001, Vol 38, No 4, pp 578-583

[NMP, 2003] "Technology Readiness Levels for the New Millennium Program", New Millennium Program, NASA, version 1, May 2003

[NRC, 1999] "Institutional Arrangements for Space Station Research", National Research Council, Washington D.C., 1999

[NRC, 2000] "Engineering Challenges to the Long-Term Operation of the International Space Station", National Research Council, Washington DC, 2000

[NRC, 2001] "Readiness Issues Related to Research in the Biological and Physical Sciences on the International Space Station", National Research Council, Washington DC, 2001

[NRC, 2002] "Factors Affecting the Utilization of the International Space Station for Research in the Biological and Physical Sciences", National Research Council, Washngton DC, 2002

[NRC PRB] Polar Research Board, "Antarctic Treaty System: An Assessment", National Research Council, Washington DC, 1985

[Nolet, 2004] Nolet, Simon, Kong, Edmund, Miller, David W, "Autonomous docking algorithm development and experimentation using the SPHERES testbed", SPIE Defense and Security Symposium, Orlando FL, 2004

[NSTS, XIV] "Space Shuttle System Payload Accommodations", Space Shuttle Customer and Flight Integration Office, NASA, Houston, TX, Rev L.1, 2001

[O'Neill, 1999] O'Neill, J, Wales, R, "CSCA Issues Raised by Mission Control for the International Space Station", CSCL 1999, UK

[OPP, URL] Antarctic Science Section, Office of Polar Programs

- http://www.nsf.gov/od/opp/antarct/start.htm

[Pololu, URL] "Micro Dual Serial Motor Controller User's Guide", Pololu Corporation, Las Vegas, NV

- http://www.pololu.com/products/pololu/0410/smc02b_guide.pdf

[PSI, 2003] "Middeck Active Control Experiment (MACE) and MACE-II", Payload Systems Inc, Cambridge, MA, October 2003

[Penzias, 1973] Penzias, W, and Goodman M W, "Man Beneath the Sea", Wiley Interscience, New York, 1973

[Portree, 1995] "Mir Hardware Hritage", Information Services Division, Lyndon B. Johnson Space Center, Houston TX, NASA Reference Publication 1357, March 1995

[RFM, URL] "RFM DR2000 Development Board", RFM Monolithics Inc

- http://www.rfm.com/products/data/dr2000manual.pdf

[Saccoccia, 2000] Saccoccia, G, Gonzales del Amo, J, Estublier, D, "Electric Propulsion: A Key Technology for Space Missions in the New Millenium", ESA Bulletin 101, February 2000

[Saenz-Otero, 2000] Saenz-Otero, Alvar, "The SPHERES Satellite Formation Flight Testbed: Design and Initial Control", Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Masters of Engineering Thesis, Cambridge MA, 2000

[Saenz-Otero, 2002] Saenz-Otero, A, Miller, D W, "The SPHERES ISS Laboratory for Rendezvous and Formation Flight", 5th International ESA Conference On Guidance, Navigation and Control Systems, Frascati, Italy, 22-25 October 2002, paper #29

[Saenz-Otero, 2002a] Saenz-Otero, A, Chen, A, Miller, D, Histad, M, "SPHERES: Development of an ISS Laboratory for Formation Flight and Docking Research", IEEE Aerospace Conference 2002, MT, March 8-16, 2002

[Saleh, 2004] Saleh, J H, Hassan, R, Torres-Padilla, J P, et al, "Impact of Subsystem Reliability on Satellite Revenue Generation and Present Value", MIT Center for Technology, Policy, and Industrial Development, Cambridge MA, 2004

[Saleh, 2002] Saleh, J H, "Weaving Time into System Architecture: New Perspectives on Flexibility, Spacecraft Design Lifetime, and On-orbit Servicing", Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Ph.D. Thesis, Cambridge, MA 2002

[Schur, 1998] Schur, A, Keating, K A, Payne, D A, Valdez, T, et al, "Collaborative Suites For Experiment-Oriented Scientific Research", Interactions May/June volume v.3 1998

[Shaw, 1998] Shaw, G B, "The Generalized Information Network Analysis Methodology for Distributed Satellite Systems", Doctoral Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 1998

[Shoemaker, 2004] Shoemaker, J, Wright, M, “Orbital express on-orbit satellite servicing demonstration,” Proc. of the SPIE Defense and Security Symposium 2004, Vol. 5419-09, Orlando, Florida, April 2004

[SPHERES, 1999] "Requirements Document", SPHERES, Spae Systems Laboratory, MIT, Cambridge MA, Ver 2.1, Dec 1999

[SPHERES, 1999a] "Design Document", SPHERES, Space Systems Laboratory, MIT, Cambridge MA, Ver 2.0, Nov 1999

[SPHERES, 2001] "Phase 0/I/II Payload Safety Review Data Package", SPHERES, Payload Systems Inc, Cambridge MA, 2001

[SSL, 2002] "SPHERES Critical Design Review", MIT Space Systems Laboratory, Cambridge, MA, February 2002

[SSL, 2002a] "SPHERES Science CDR", MIT Space Systems Laboratory, Cambridge, MA, November 2002

[SSL, URL] SPHERES Website, MIT Space Systems Laboratory, Cambridge, MA

- http://ssl.mit.edu/spheres

[Steuer, 1986] Steuer, R E.” Multiple Criteria Optimization: Theory, Computation and Application”. Wiley, New York, 1986

[Stoewer, 1985] Stoewer, H, Binum, P M, (Editors), "From Spacelab to Space Station", American Asronautical Society, Proceedings of the Fifth AAS/DGLR Symposium held October 3-5, 1984 in Hamburg Germany, AAS, San Diego, CA, 1985

[Sundance, 2003] SMT335/SMT375 User Manual, Sundance Multiprocessor Technology Ltd, Version 3.4, QCF42, 2003

[TI, SPRS067E] "TMS320C6701 Floating Point Digial Singal Processor", Texas Instruments, Houston TX, SPRS067E, 2000

[TI, SPRU159A] TMS320C4x General-Purpose Applications", Texas Instruments, Houston TX, SPRU159A, 1999

[TI, SPRU189F] "TMS320C6000 CPU and Instruction Set Reference Guide", Texas Instruments, Houston TX, SPRU189F, 2000

[TI, SPRU303B] "TMS320C6000 DSP/BIOS User’s Guide", Texas Instruments, Houston TX, SPRU303B, 2000

[TI, SPRU328B] "Code Composer Studio User’s Guide", Texas Instruments, Houston TX, SPRU328B, 2000

[TI, SPRU403E] "TMS320C6000 DSP/BIOS Application Programming Interface (API) Reference Guide", Texas Instruments, Houston TX, SPRU403E, 2002

[TI, SPRU423B] "TMS320 DSP/BIOS User's Guide", Texas Instruments, Houston TX, SPRU423B, 2002

[UoM SSL, URL] Space Systems Laboratory, University of Maryland

- http://www.ssl.umd.edu/

[van Schoor, 1993] van Schoor, M C, Crawley, E F, Miller D W, "Results from the Middeck 0-gravity Dynamics Experiment", NASA STI/Recon Technical Report A, 1993, Vol 95, p 33-56

[WHOI, URL] Marine Operations, Woods Hole Oceanographic Institute

- http://www.whoi.edu/marops/index.html

[Wilson, 2002] Wilson, E., Lages, C., and Mah, R., "On-line gyro- based mass-property identification for thruster-controlled spacecraft using recursive least squares", Proceedings of the 45th IEEE International Midwest Symposium on Circuits and Systems, Tulsa, Oklahoma, Aug 2002

[Wilson, 2003] Wilson, E, et al., "Motion-Based System Identification And Fault Detection And Isolation Technologies For Thruster Controlled Spacecraft," Proc. of the JANNAF 3rd Modeling and Simulation Joint Subcommittee Meeting, Colorado Springs, CO, December 2003

[Wycoff, URL] Wycoff, J, "Defining Innovation", Innovation Network, Denver CO

- http://www.thinksmart.com/2/articles/MP_3-3-3.html

[Xilinx, DS001-1] "Spartan-II 2.5V FPGA Family: Introduction and Ordering Information", Xilinx Inc., DS001-1, 2001

- http://www.xilinx.com

[Yung, 2001] Yung, J H, "Gain Scheduling for Geometrically Nonlinear Flexible Space Structures", Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Ph.D. Thesis, Cambridge, MA, 2001

[ZARM, 2000] "ZARM Drop Tower Bremen, General Information", Drop Tower Operation and Service Company, ZAR FABmbH, Am Faliturm, Bremen, Germany, ,ver 28, April 2000

[ZARM, 2003] "ZARM Drop Tower Bremen, User Manual", Drop Tower Operation and Service Company, ZAR FABmbH, Am Faliturm, Bremen, Germany, ver 10, July 2003

Appendix A

NASA TECHNOLOGY READINESS LEVELS

The following is the description of the Technology Readiness Levels presented in the TPF Technology plan [Lindensmith, 2003]:

> Technology Readiness Levels (TRLs) are a systematic metric/measurement system that supports assessments of the maturity of a particular technology and the consistent comparison of maturity between different types of technology. The TRL concept is based on a general model for technology maturation that includes: (a) research in new technologies and concepts (targeting identified goals, but not necessary specific systems), (b) technology development addressing specific technologies for one or more potential identified applications, (c) technology development and demonstration for each specific application before the beginning of full system development of that application, (d) system development (through first unit fabrication), and (e) system 'launch' and operations.
>
> **TRL 1: Basic principles observed and reported**
>
> Transition from scientific research to applied research. Essential characteristics and behaviors of systems and architectures. Descriptive tools are mathematical formulations or algorithms.
>
> **TRL 2: Technology concept and/or application formulated**
>
> Applied research. Theory and scientific principles are focused on specific application area to define the concept. Characteristics of the application are described. Analytical tools are developed for simulation or analysis of the application.

TRL 3: Analytical and experimental critical function and/or characteristic proof-of-concept

Proof of concept validation. Active Research and Development (R&D) is initiated with analytical and laboratory studies. Demonstration of technical feasibility using breadboard or brassboard implementations that are exercised with representative data.

TRL 4: Component/subsystem validation in laboratory environment

Standalone prototyping implementation and test. Integration of technology elements. Experiments with full-scale problems or data sets.

TRL 5: System/subsystem/component validation in relevant environment

Thorough testing of prototyping in representative environment. Basic technology elements integrated with reasonably realistic supporting elements. Prototyping implementations conform to target environment and interfaces.

TRL 6: System/subsystem model or prototyping demonstration in a relevant end-to-end environment (ground or space)

Prototyping implementations on full-scale realistic problems. Partially integrated with existing systems. Limited documentation available. Engineering feasibility fully demonstrated in actual system application.

TRL 7: System prototyping demonstration in an operational environment (ground or space)

System prototyping demonstration in operational environment. System is at or near scale of the operational system, with most functions available for demonstration and test. Well integrated with collateral and ancillary systems. Limited documentation available.

TRL 8: Actual system completed and "mission qualified" through test and demonstration in an operational environment (ground or space)

End of system development. Fully integrated with operational hardware and software systems. Most user documentation, training documentation, and maintenance documentation completed. All functionality tested in simulated and operational scenarios. Verification and Validation (V&V) completed.

TRL 9: Actual system "mission proven" through successful mission operations (ground or space)

Fully integrated with operational hardware/software systems. Actual system has been thoroughly demonstrated and tested in its operational environment. All documentation completed. Successful operational experience. Sustaining engineering support in place.

Appendix B

MICROGRAVITY RESEARCH FACILITIES

Chapter 1 identifies the facilities presented in Table B.1 to showcase an important range of existing facilities to conduct microgravity research. This appendix presents an in-depth review of 3rd party ground based (second column) and space based (third column) facilities. This appendix also presents a review of the four major space stations which operated in the past, including their operations and the research conducted during the programs.

TABLE B.1 Sample of available facilities for μ-g research

In-house	3rd Party / Full μ-g	Space
Robot Helicopters	RGO (KC-135)	Free Flyer
6 DOF Robot Arms	Neutral Buoyancy Tank	ISS
Helium Balloons	Drop Towers	Shuttle Payload
Robot Cars		Shuttle Middeck
Flat Floor		
Air table		
Simulation		

B.1 3rd Party Ground-based Facilities

The available terrestrial facilities include: simulations, air tables, flat floors, robot cars, helium balloons, robot arms, robot helicopters, drop towers, neutral buoyancy, and reduced gravity airplanes (NASA's Reduced Gravity Office, as well as Russian and Euro-

pean facilities). Simulations, air tables, robot cars, helium balloons, robot arms, and helicopters, are all tools available to researchers at home. Their operations and capabilities depend on the design created by each researcher, and therefore it is not possible to identify their characteristics in general; further, these facilities only provide limited microgravity conditions, not necessarily creating a representative environment. Flat floors can be created by individual researchers, but some general use facilities do exist; and examples are presented below. Drop towers, neutral buoyancy tanks, reduced gravity airplanes, and the space shuttle require special attention, as they are facilities which usually involve a third party but which closely meet the need for a representative environment.

B.1.1 Flat Floors

Flat floors utilize an air cushion to float an experiment in such a way that frictionless motion is provided in a two dimensional environment. A basic flat floor setup provides simulated microgravity in two translational and one rotational dimension. While flat floors restrict the operations to 3DOF in most cases, they do present a viable intermediate step for technology maturation if their size is large enough to provide a representative environment. Further, the use of special carriages can allow limited 6DOF operations, as expected from the TPF experiments at JPL.

In the United States NASA operates a large flat floor facility at Marshall Space Flight Center and Boeing operates a privately owned facility. Both facilities are available for research by scientists at large. The facilities provide scientists with up to eight hours per day of operations, limited only by the operational nature of their experiments. While some limitations exist to ensure the safety of the flat floor, researchers can operate relatively freely in the facilities, maximizing the interaction with their experiments.

B.1.2 Drop Towers

Drop towers simulate microgravity by allowing the experimental item to free fall for a short period of time in a controlled environment. There is a large number of drop towers

around the world which range in size from providing only fractions of a second to almost ten seconds of microgravity time. Important drop towers exist in the United States, Europe, and Japan. Two of them stand out in our discussion due to their openness for current research by a wide number of scientists and their relatively large size: the NASA Glenn 2.2 Second Drop Tower and the ESA's ZARM Drop Tower Bremen.

The NASA tower provides 2.2 seconds of microgravity, as its name implies. The drop takes place off a 100ft tower (the actual drop is 79'1"). A drag shield is used to minimize air drag. The tower allows up to 12 drops each day, with a clearly defined operational plan. The center provides a range of support facilities for assembly of the experiment and integration to the drag shell. The center also provides support hardware such as cameras, data acquisition equipment, and batteries. After integration the package is lifted to the top of the tower, at which point the investigator can perform any preparations necessary. The experiment is provided with electrical control signals which will indicate the exact time of the drop, such that experiments utilize as much as possible of the drop time and do not need to waste resources before the micro gravity time. The drop is initiated by cutting the cables that hold the experiment, achieving microgravity conditions within one-third of a second. The drop ends when the drag shield falls onto an airbag at the bottom of the tower; the impact peak values are 15 to 30g.

ZARM has a vertical drop of over 100m, providing 4.74s of free fall; the use of a catapult to allow a parabolic path of the payload gives up to 10s of free fall. A picture and diagram of ZARM are shown in Figure B.1 [ZARM, 2000]. But ZARM allows only a maximum of three drops in one day. Further, the ZARM operations are more complex, requiring the investigators to be on site at least ten days prior to the tests in order to prepare their payloads. Once the payload is integrated, researchers have remote access to the payload once it reaches the platform, but before the drop. The small number of drops is due to the fact that ZARM evacuates its drop tube to under 10Pa (the experiment is pressured), a process that takes two hours, but remote access to the experiment is available during this time. The experiment experiences up to 25g at impact. It takes approximately one hour to retrieve

the experiment. A full set of experiments at ZARM usually consists of 8 to 24 drops, which takes approximately three to four weeks to complete.

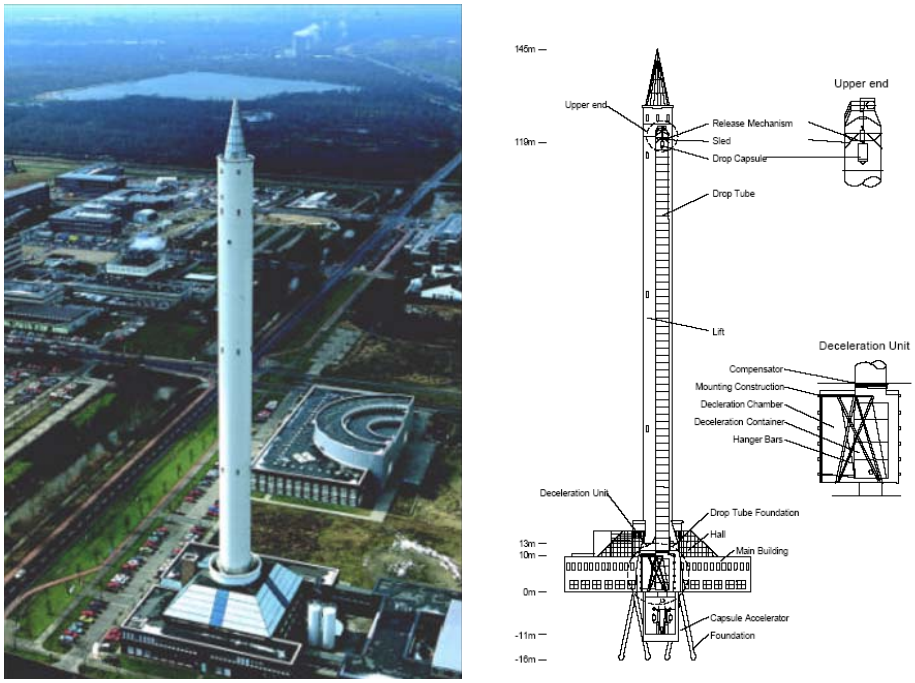


Figure B.1 ZARM drop tower

Drop towers present good opportunities for research in terms of the ability of investigators to work directly with their experiment both before and after the microgravity test. In general the size of the experiment is not restricted; the limitations are one to two meters in diameter and a mass of over 100kg. But:

> "...not all types of scientific inquiry are appropriate for the drop tower. For example, meaningful microgravity research of the life sciences, biotechnology, and materials sciences can seldom be conducted in drop facilities (living things and crystals grow too slowly). On the other hand, flames can spread very quickly, which explains why combustion experiments account for approximately 90% of the experiments conducted in the drop tower." [NASA, URL3]

The applicability of drop towers for space technology maturation is limited. Only those tests that can conclude well within five seconds will benefit from drop towers.

B.1.3 Neutral Buoyancy Tanks

Neutral buoyancy tanks are operated in multiple places around the world. The major tanks are operated by NASA, the Russian Space Agency, and the University of Maryland (UoM). NASA and the Russian Space agency manage major neutral buoyancy facilities. The UoM Space Systems Laboratory operates a neutral buoyancy facility for research purposes. All of the tanks are large enough to allow full-sized tests of major spacecraft. For example, the NASA Neutral buoyancy Laboratory at the Johnson Space Center can hold several full-sized mock-ups of modules of the International Space Station (Figure B.2 [NASA, URL6]).

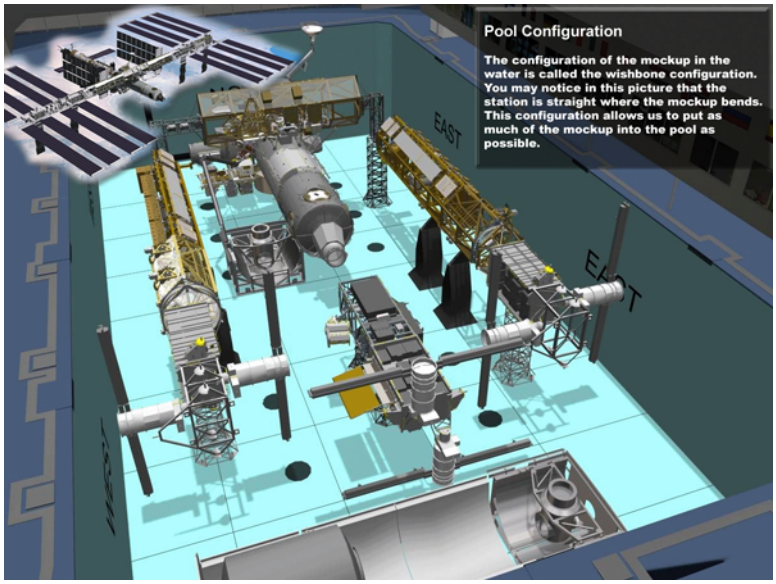


Figure B.2 NASA Neutral Buoyancy Laboratory

NASA manages facilities at JSC and at the Marshal Space Flight Center. The Russian Space Agency manages a tank at the Gagarin Cosmonauts' Training Center outside Moscow. The primary purpose of these facilities is to train astronauts for extra-vehicular activities (EVA). Full scale mock-ups of spacecraft that will require assembly in space are created; astronauts and cosmonauts use the same tools they will use in space for training. The only difference usually present is that the astronauts supply of breathable air and

power for the EVA suits is provided by tethers, rather than through the backpacks of EVA suits.

The UoM SSL tank is the only neutral buoyancy facility dedicated to research. It supports experiments by undergraduates, graduates, and faculty of the university. The UoM SSL has also established a program by which external parties can conduct research at the tank. Research has concentrated on EVA operations and tools, the Ranger vehicle for telerobotics, and the SCAMP project for additional video during EVAs.

Conducting operations in the tanks requires certified SCUBA divers to either perform the activities and/or support the test subjects. In the case of astronaut training, the astronauts are supported by large teams of SCUBA divers who monitor their health and progress. In the case of research, certified divers must perform the experiments. This operational scenario introduces the need for a third party to sometimes perform the experiment, rather than the scientist always being directly at the controls, since not all research scientists will have the required certification. This trend will continue from this point forward as the operational environments get more complex. Given the nature of SCUBA diving, the facilities usually perform one major session a day. The test session can be several hours long, allowing a substantial amount of research to be conducted and minimizing the impact of setup times. The high level of support at the locations, such as SCUBA support, machine shops, and work areas, allow research to be conducted with low risk. Because they are based in ground facilities the research is not strictly limited to autonomous operation, and supplies can be replenished easily. Through the course of a few weeks, a scientist could get a substantial number of tests performed; the tanks are readily available for continued research.

The type of activities conducted in neutral buoyancy tanks gives a clear idea of their best use: human interactions and large spacecraft mock-ups. While neutral buoyancy allows full 6DOF maneuvers, the effects of drag in water prevent the dynamics from being equivalent to space. Therefore neutral buoyancy tanks are not practical for tests that will be

affected by drag, such as propulsion tests or precision spacecraft control. Therefore, the applicability of neutral buoyancy tanks is best suited for that range of research which does not require precise representation of a microgravity environment, especially with respect to the dynamics of the system.

B.1.4 Reduced Gravity Airplanes

NASA and the Russian Space Agency operate the most commonly used reduced gravity airplanes, although other national space programs and private ventures also exist. NASA operates a program out of its Reduced Gravity Office at the Johnson Space Center. The Russian aircraft operates out of the Gagarin center outside Moscow. Both programs continuously support research by government, academic, and private agencies, as demonstrated by the NASA Reduced Gravity Program Mission Statement:

> "To provide a world-class, reduced gravity research platform that emphasizes user compatibility, quality reduced gravity levels, and a customer-oriented support organization."

Reduced gravity is achieved by following a parabolic curve with an amplitude of approximately 10,000 feet, providing approximately 20 seconds of microgravity. Each microgravity flight consists of 15 to 40 parabolas, for up to 400 seconds of micro gravity time in a day (Figure B.3, [NASA, 33899] [NASA, 33898]). This period more than doubles the available time from drop towers per drop, and is four times that per day; however, it is substantially less than that of neutral buoyancy tanks. The airplanes provide another important benefit over the drop towers: the scientists can directly interact with their experiments; it is also an improvement over neutral buoyancy tanks because they do not need SCUBA certification or equipment, allowing easier interaction with the equipment.

Like with drop towers, operations on a RGA are very structured and time critical. The following example is based on the NASA RGP. Usually experiments operate on the aircraft one week at a time, with one day for setup and then four days of operations (up to 160 parabolas), requiring scientists to travel to JSC for a minimum of three days and up to a week. The first day involves integration of the experiment to the aircraft, usually within

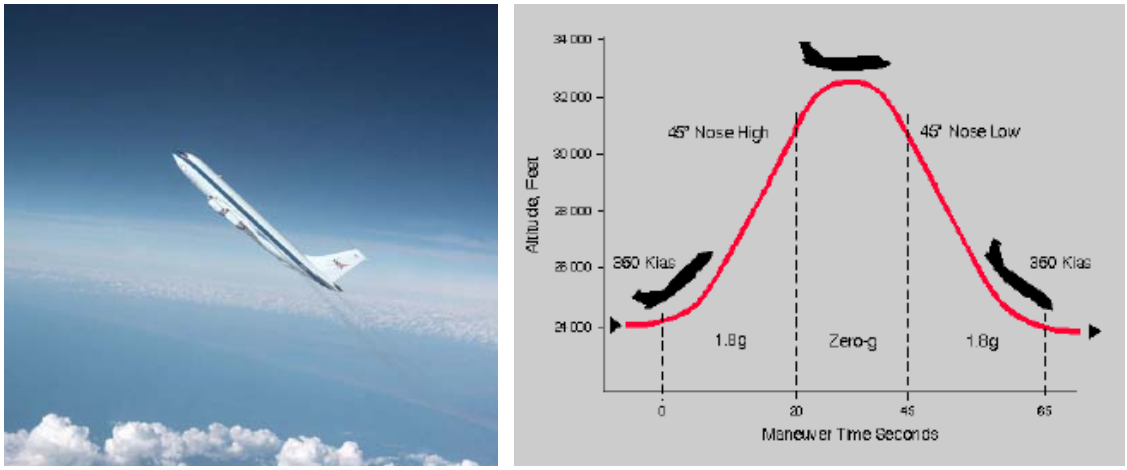


Figure B.3 NASA KC-135 airplane and flight path

three or four hours. The flight days are especially time critical; the experiments are accessible for about one hour before flight; after takeoff there is a period of about 15 minutes to start experiments; then parabolas start. The parabolas consist of approximately 40 seconds of "pull-up", times with 1.8g when the airplane flies towards the top of the parabola, and then 20 seconds of microgravity as the airplanes flies over the top; 10 parabolas are repeated consecutively. Five minute breaks are available after parabolas 10 and 30; a 10 minute break is available after parabola 20. While the scientists have full access to their experiments and can interact with them, this environment is not susceptible to substantial modifications or repairs without wasting valuable microgravity time. Therefore scientists must be prepared for successful operation and have backup plans in case of equipment failure or incorrect assumptions in their setups.

Research conducted at NASA's RGP covers a wide range of areas including human factors, medicine, space technology, astronaut training, and combustion. The range of science is substantially more than that of drop towers or neutral buoyancy tanks for two main reasons: human presence over a substantial period of time. Yet, the RGAs are not suitable for every type of science either. Like with drop towers, research on biological sciences, some human factors, and spacecraft control usually requires prolonged exposure to micrograv-

ity. Further, the effects of turbulence and the rotational motion as the airplane goes over the parabola prevent it from providing completely clean dynamics.

B.2 Space Shuttle

> "The United States developed the Space Shuttle system to improve its access to space. Since the first flight in April 1981, the Shuttle has carried more than 1.5 million pounds of cargo and over 600 major payloads into orbit. The Shuttle is the first and only reusable space vehicle, and is the world's safest, most reliable and versatile launch system. It is designed and operated to support a variety of space-based activities from the delivery of large payloads to orbit, the capability of spacecraft retrieval and servicing, to providing a versatile platform to conduct research and development experiments in a "shirt-sleeve" laboratory environment. The Shuttle also provides for experiment return and re-flights.
>
> "Among the Shuttle's greatest strengths, in addition to its amazing array of capabilities, is its ability to adapt and evolve to meet new mission requirements. Whether it means installing a spare part, or sending astronauts on a spacewalk to retrieve an errant satellite, the Shuttle is unrivaled in its ability to adapt real-time to get the job done....:
>
> - Payload Deployment and Retrieval
> - On-Orbit Assembly
> - On-Orbit Repair and Servicing
> - On-Orbit Research
> - Technology Testbed
> - Crew Transfer
> - Cargo Return " [NASA, URL4]

NASA’s Space Shuttle Program (SSP) provides service to a wide range of payloads, from small experiments operated inside the crew compartment area (middeck) to the deployment of large satellites into orbit from the payload bay (Figure B.4 [NSTS, XIV]). Among the most unique features of the SSP is not only the availability of humans, but also the return of the vehicle, crew, equipment, and products to Earth in a short period of time. Further, the SSP supports two main operational areas: a payload bay for experiments fully exposed to the space environment, and the middeck area for experiments that require human interaction and/or a pressurized environment.


Figure B.4 Space Shuttle payload bay and middeck

A typical SSP program starts two years before flight by presenting the necessary documentation to NASA. During the first year scientists present safety and interface documentation to NASA while they develop their experiments/products. The year before flight involves the development of the flight hardware and at least six months of integration. During this time researchers must also train astronauts.[NASA, 2000a]

A typical Space Shuttle mission starts with launch at KSC. After launch the orbiter can reach altitudes of 100-600nm and inclinations of 28-51deg. Approximately 10 minutes after liftoff the shuttle performs its final main engine firings to reach the approximate orbit. Next the payload bays are opened to allow the space radiators to dissipate heat; the doors remain open for the duration of the mission. The base mission is seven days long, but a mission can be extended for up to 16 days. The orbiter returns to earth by firing its main engines to reduce its speed, after which it glides back to earth un-powered.

The standard operating procedure for experiments grows in complexity from previously presented facilities. While the RGA carry two pilots, three to five operations support personnel, and 15 to 20 scientists, the space shuttle carries at most seven astronauts that share research and operational duties throughout the mission. Therefore, conducting experiments begins with the training of astronauts; this training must be substantial enough that astronauts could conduct the research independently in case there is no real-time link

between the shuttle and ground. Due to limited communications between the shuttle and ground the experiment must also be ready to operate without further assistance from the scientist once it gets integrated into the spacecraft, usually no less than four weeks prior to launch. While software updates are possible once in flight, the process must involve, as a minimum, the payload integration office and mission control, adding layers of complexity to the ability to change experiments while in flight. As a result, in general, experiments are fully designed months prior to being conducted in the shuttle.

Once actual operations start, astronauts follow established procedures for the experiments. These procedures guide the astronauts through the complete experiment, and usually the scientist is not involved in real-time. In the cases where a real-time link is available, the communications is handled through mission control; the scientists communicate with mission control, who forward the instructions from the scientists to the astronauts. The scientist does directly observe and/or hear what the astronaut is doing, but mission control must be involved in any communications to the astronauts.

After the experiment is performed the scientists will usually wait for the return of the shuttle to ground before accessing their data. Limited communications do exist to allow scientists access to the data prior to return of the shuttle, but the use of these systems will further add to the complexity of the integration process (a trade-off which scientists must consider).

Each space shuttle mission carries three types of payloads: primary, secondary, and middeck. The primary payloads are those that justify the flight; each flight can have one or multiple primary payloads. Secondary payloads, in general, do not define the critical path of the integration process, but use a significant amount of SSP resources. It is possible for a number of secondary payloads to together justify a flight as a primary payload. A middeck payload does not define the critical path of the integration process, but still requires significant SSP resources. [NSTS, XIV]

In general SSP missions conduct dozens of experiments. The astronaut time must be carefully divided throughout the one to two weeks of operations. To accomplish this the SSP has a clearly defined integration process which dictates how payloads are added to the program and their priority. Therefore, even though each mission has hundreds of man-hours available, the operations in the shuttle are even more structured than those in RGAs. Once a research session is scheduled on the shuttle, scientists must ensure that they will fully utilize their assigned time. Scientists have to consider the time to set up their experiment, the amount of interaction with astronauts while the experiments run, and the time to take it apart. Still, each experiment can take on the order of hours (attended), and even days or weeks (unattended) to complete, rather than a few seconds.

The space shuttle exposes payloads to full microgravity conditions, with almost perfect dynamics for spacecraft. External payloads are further exposed to space conditions. The shuttle does orbit the earth in LEO, which means that the orbital dynamics to which experiments are exposed are not necessarily identical to that of final missions (for example, some experiments may be precursors of earth-trailing satellites). Further, the shuttle does have orbit correction maneuvers during its mission, at which point experiments are exposed to non-realistic forces from thrusters. Still, the SSP provides one of the cleanest microgravity conditions available for research.

The large number of experiments conducted so far in the SSP demonstrates its success in conducting microgravity research. Projects have covered almost every area of space research, including astronomy, biological experiments, material science, space technology development, human factors, and space propulsion; the shuttle has also been used for deployment and capture of a wide range of spacecraft which have conducted their own science.

The major challenges in conducting research aboard the space shuttle lie in the integration process. Because the SSP is a precious facility with limited operations, the integration process is not only time-consuming, but also requires substantial investments by the scientists

in both work hours and money. A decision to use the SSP as a research environment means the scientists are wiling to compromise between obtaining almost ideal microgravity conditions in exchange for a substantial jump in operational complexity. The SSP requires scientists to go through substantial safety approvals and astronaut training. Further, it requires that a third party always be involved in operations, since the researcher cannot communicate directly with the astronauts conducting the experiments.

B.3 The International Space Station

> "The purpose of the ISS is to provide an “Earth orbiting facility that houses experiment payloads, distributes resource utilities, and supports permanent human habitation for conducting research and science experiments in a microgravity environment.” (ISSA IDR no. 1, Reference Guide, March 29, 1995)
>
> "This overall purpose leads directly into the following specific objectives of the ISS program:
> - Develop a world-class orbiting laboratory for conducting high-value scientific research
> - Provide access to microgravity resources as early as possible in the assembly sequence
> - Develop ability to live and work in space for extended periods
> - Develop effective international cooperation
> - Provide a testbed for developing 21st Century technology."
>
> [NASA, 1998]

The ISS is the only existing facility which provides a true microgravity environment whose goal is specifically to support scientific research. While its configuration and specific research goals have changed over time, the basic concept remains the same: to provide a manned microgravity environment for scientific research and technology development.

The idea of a permanent space station began in 1984 when President Ronald Reagan invited Canada, Europe and Japan; Russia joined the program in 1993. ISS development has been split into three phases. ISS Phase I took place before assembly and consisted of a series of cooperative research flights between the United States and the ISS partners. Most

notably, this involved a series of rendezvous flights between the Space Shuttle and the Russian space station MIR, cosmonaut flights on the Space Shuttle, and U.S. astronaut stays on MIR; research on MIR was expanded to US and other international partners. In ISS Phase II, the knowledge gained from Phase I operations is being applied to the on-orbit assembly of the ISS. ISS Phase II will conclude with the successful assembly of the U.S. and Russian components of the ISS that are necessary to begin Station research. ISS Phase III development consists of the final research outfitting of the ISS, as the European, Japanese, and Canadian elements are transported to orbit and the Station becomes fully operational. [IMBOSS, URL]

Assembly of the ISS began in 1998 and will reach "US Core Complete" after mission 10A (predicted for the end of 2006), marking the end of Phase II. Figure B.5 ([NASA, ULR8]) shows a picture of the ISS on October 2002. At that point the ISS habitable modules will consist of the Zarya control module, two nodes, three docking modules, the US Destiny Laboratory, the ESA Columbus laboratory, the Japanese Experiment Module (including an exposed facility), a centrifuge module, and two airlocks. The exposed elements will include solar arrays to provide up to 30kW of power for research, multiple express pallet mounting points, a science power platform, and the CSA robotic arm. At core complete, the ISS will be permanently inhabited by three people.

Like the space shuttle, the ISS offers extremely clean microgravity conditions in both its pressurized and exposed modules. The only limits of the ISS lie in the need for orbit correction maneuvers, which introduce artificial forces on the experiments, and the orbit location in LEO. As compared to the space shuttle, though, the ISS long-term deployment allows experiments to be exposed to microgravity conditions even for years at a time. The constant number of servicing flights to the ISS allow experiments to be returned to earth within a reasonable time. The expanded number of flights, beyond the space shuttle, allow the hardware of experiments to be upgraded in shorter time periods than possible with SSP.

Figure B.5 The ISS on October 2002

As with the space shuttle, ISS operations are highly structured, time-critical, and require the presence of third parties throughout a substantial part of the program. Even with three humans permanently present in the ISS, their time is extremely precious. NASA and its partners spend incredible amounts of time organizing the schedules of astronauts. Even when an astronaut completes their duties early, a number of activities are always ready to be performed. Therefore, many scientific experiments are likely to be performed at random times, preventing the scientists to interact in real-time with the astronauts. As such, like before, all experiments must go through enough training and integration that they can be conducted independently by the astronauts. As with the SSP, when real-time communications are possible, the scientist must interface with mission control and the payload integration office, rather than directly with the astronaut. The ISS does provide a substantial improvement on communications, such that when properly planned, real-time audio, video, and data transfer (both downlink and uplink) are possible. Even though the operations of the ISS are clearly more complex than a scientist conducting research in their own facility, the trade-off between operational complexity and availability of microgravity is

not as important because of the availability of substantial astronaut time, real-time interaction, and ability to upgrade both hardware and software in a reasonable amount of time.

B.4 Past Space-based Laboratories

The ISS had both US and Russian predecessors: the US Skylab, Space Lab, and the Russian Sályut and MIR Space Stations. This section presents a quick historical review of these four different facilities, as well as their contributions to scientific research in space and the development of the ISS.

B.4.1 Sályut

Multiple Sályut space stations were built through the 1970's and into the 1980's. The first station was launched in May 1971. Its first crew worked on the station for 21 days; unfortunately the crew died after a de-pressurization accident upon reentry. The second mission was destroyed in an explosion of the launch vehicle. The third attempt reached orbit in 1973, but contact was lost before a crew could reach the station. The fourth version flew successfully in 1974. The sixth and seventh missions flew in 1977 and 1982, respectively. The sixth mission was the most successful, recording 27,785 orbits of Earth and five manned expeditions over four years and ten months of operations. The longest crew duration was 185 days. The seventh mission was aloft for four years and two months; two to six crew members were aboard at any one time through six main expeditions and four secondary flights (including French and Italian cosmonauts). Mission durations of up to 237 days were achieved. The seventh and last Sályut mission ended in June 1986; the spacecraft burned up in the Earth's atmosphere in 1991.

The Sályut program featured important capabilities seen in the ISS. The stations performed docking of both manned and unmanned spacecraft for crew rotation and resupply missions. For this purpose, the stations had two separate docking ports. The stations had an airlock, which allowed dozens of EVA's to take place. Of special importance to the ISS program is that a number of international cosmonauts flew to the Sályut stations over the

years, the first time that international cooperation would occur in a space station. Lastly, the objective of the Sályut program was directly in line with that of the ISS:

> "Objectives: Continuation of scientific research on board manned space complexes in the interests of science and the Soviet national economy; testing of advanced systems and apparatus for orbital stations. Continuation of the scientific research in progress on board manned space complexes in the interests of science and the national economy; testing of advanced systems and apparatus for orbital stations." [Astronautix]

The sixth mission completed many of its objectives with a wide range of scientific equipment: multi spectral camera system, high resolution topographical camera system, 10 m diameter radio telescope, alloying/materials processing furnace, containerless processor for semiconductor materials, 1.5 m diameter cryogenic submillimeter/ultraviolet/infrared telescope, Refraktion and Zarya spectrometers for sun/moon views through Earth's limb, experiment for coating of plates with materials, gamma ray telescope, plant growth unit, and cardiovascular monitoring system. The seventh mission was dominated by military research after cutbacks in other military programs forced the use of the Sályut for this purpose.

B.4.2 US Skylab [Belew, 1977]

Skylab was the United States' first space station. It was more than the actual space station, it was a comprehensive scientific program. The program consisted of four launches: one unmanned delivery of the space station and three separate manned missions. The station was launched in May 1973 and remained in operation through its third man mission which ended in February 1974. Through these months in space Skylab completed more than 100 experiments in a wide range of topics.

The Skylab station consisted of five major elements (Figure B.6, [Belew, 1977]): command and service module (CSM), orbital workshop, airlock module, docking adapter, and the Apollo Telescope Mount for solar observations. The unmanned launch put into orbit all the modules except the command and service module; the CSM, a standard Apollo command module, carried the crew to the station and returned them to Earth at the end of

their mission. The majority of the activities occurred in the orbital workshop, which included the living and working quarters.

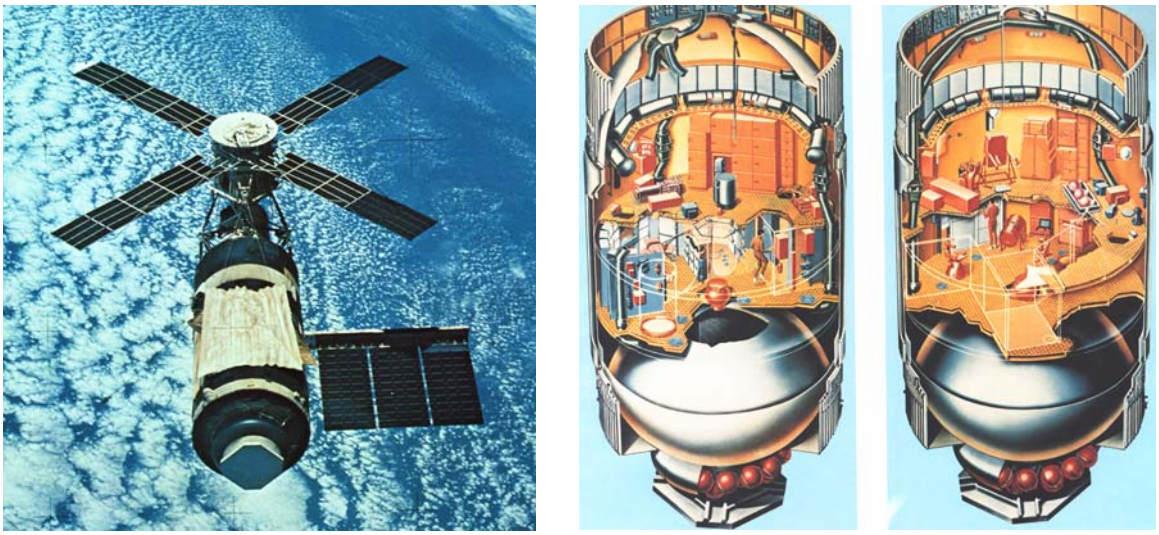


Figure B.6 US Skylab

The Skylab science covered multiple disciplines. The Apollo Telescope provided unprecedented observations of the sun. The location of Skylab outside the atmosphere also allowed studies of stellar astronomy, including the study of comet Kohoutek. Man's adaptability to long-duration space-flights was studied; this included both physiological and behavioral research. Observation of the Earth provided more than 40,000 pictures for scientists to study agriculture and forestry, geological applications, the oceans, coastal zones, water resources, atmospheric phenomena, regional planing and development, and remote sensing technologies. Substantial research was also conducted in the area of materials science, including crystal formation, homogeneity in semiconductors, diffusion in liquid metals, and solidification of metals. Lastly, Skylab also served as an educational environment; a large number of student projects were completed by the astronauts.

As a predecessor to the ISS, Skylab started the goal to develop space facilities dedicated to scientific research. While the scientific goals of Skylab were strictly defined to understand

the human physiological responses to long-term microgravity conditions, with a pre-defined set of additional experiments, the major goals were always to support science.

B.4.3 Space Lab [Emond, 2000]

While not a true space station, the Spacelab program helped pave the way for the ISS in many ways. Spacelab was not a spacecraft launched to space for long-term microgravity research; rather, the Spacelab program was a successful attempt to open the use of the space shuttle to a broad range of research by the international community. Through a system of modules, Spacelab provided both pressurized and un-pressurized platforms. NASA operated the program, including providing the space shuttle to carry the modules. The ESA designed the modules and supported the operations of the laboratory's payloads.

The program lasted 17 years, from 1981 to 1998. Spacelab modules flew on 36 space shuttle missions; 16 missions were with pressurized modules, the rest with exposed pallets. A total of 375 days of flight were logged. Throughout its time it supported over 800 distinct experiments, leading to thousands of research articles and hundreds of graduate theses. Four principle components formed the core hardware of the Spacelab program. The Spacelab Module provided a pressurized environment. The Spacelab Pallet allowed large instruments to be in direct exposure to the space environment and broad fields of view. The Instrument Pointing System provided a high-accuracy mount for space telescopes. The Mission-Specific Experiment Support Structure supported up to 3000 pounds of payload.

Spacelab was a milestone towards the ISS in two important respects. It began unprecedented international space collaboration. Due to the size of the program, researchers, engineers, scientists, and peacemakers had to learn how to properly use the facilities that Spacelab provided. As a special part of the Spacelab program, the Spacelab 1 mission docked a Spacelab Module to the Russian MIR space station, providing a true laboratory environment for MIR, and creating, albeit for a short time, the first truly international space station for research.

By developing complex systems, which were different with each mission, many lessons were learned on how to use the facilities; sometimes what was learned was that you have to fly it to learn how to use it. Spacelab provided experience on integrating a wide range of experiments from multiple disciplines and countries. It also provided an environment where the scientist was directly involved in the actual process of conducting the science; scientists knew when execution would take place and had input on what would happen. Spacelab provided a clear perspective on what to expect from the operations of the ISS.

Spacelab fostered a broad range of research activities spanning widely separate fields of study. While research on the Sályut and Skylab missions was pre-defined and strictly scheduled, Spacelab opened the doors for a much broader range of science to be conducted. No longer was the science for the full program pre-defined, but rather unsolicited research could be conducted. Spacelab provided both the hardware and operational support needed to broaden the research spectrum. The program welcomed experiments exposed to space and pressurized in the module. The timeline allowed development of new experiments over time. New technologies that appeared after the program was launched could be integrated to enable new science.

Spacelab created a cultural change in the ways to perform microgravity research. The space stations by both the US and Russia demonstrated the ability of humans to reside in space for long periods of time. Spacelab demonstrated the ability to create an international program which welcomed a wide range of science.

B.4.4 MIR

MIR was constructed from 1986 to 1996. The MIR Core Module was launched in 1986 and provided living accommodations and station control. Two scientific modules, Kvant I and II, were launched in 1987 and 1989, respectively. A third module, Kristall, was added in 1990. The Spektr module, added in 1995, added space for a US astronaut to live in MIR. In 1996 the Docking module was added to provide a connection port to the US space shuttle. Figure B.7 ([NASA, URL5]) shows a picture of MIR taken from the Space Shut-

tle. MIR was burned in the Earth's atmosphere in March 2001. Through its lifetime, MIR was serviced by Soyus (manned) and Progress (supplies) vehicles.

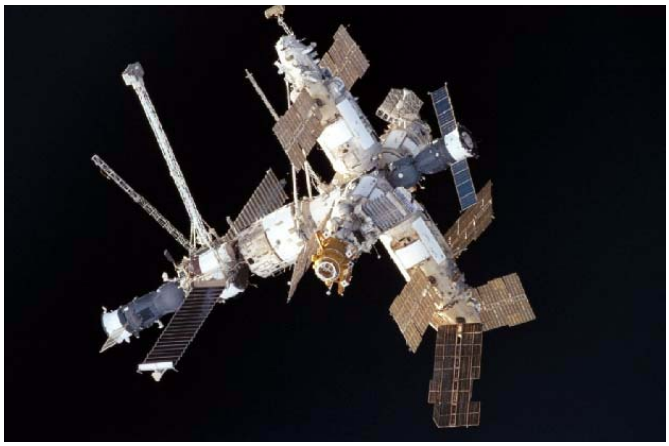


Figure B.7 The MIR Space Station

As a whole the MIR program surpassed all of its expectations. Originally scheduled for five years of operations, it was operational for almost 15 years. Even after the original Russian plans were complete, the station continued to grow to welcome US astronauts and the space shuttle. By the end of its life, when docked with the Space Shuttle, the joined MIR station and space shuttle formed history's largest spacecraft.

But MIR was not without problems. Through its history MIR had to be serviced continously. Assembly of the station required several unexpected EVAs after failed automatic dockings. The station suffered from fires, broken computers, oxygen emergencies, and a collision with a Progress vehicle in 1997. Unfortunately for the research community these problems meant that a high percentage of cosmonaut time was spent in maintaining MIR. The time spent on research in MIR was greatly affected by its history of problems. Yet, all of these experiences have served as lessons for the ISS.

Research on MIR was conducted in a broad number of areas. During the Shuttle-MIR program time, research took place on: advanced technologies, earth sciences, fundamental biology, human life sciences, ISS risk mitigation, life support risk mitigation, micrograv-

ity, and space sciences. Advanced technologies included materials science and the characterization of the micro-accelerations in MIR, which would help in the development of science facilities in the ISS. Combustion experiments also took place. Earth sciences concentrated on remote sensing and space photography to make detailed assessments of the Earth. Both the biosphere and the atmosphere were studied. Biology research concentrated on studying the effects of microgravity on the development of plants and animals; research on radiation effects was also performed. Extensive human life sciences studies took place, among the main research areas were: the cardiovascular system, endocrinology, hematology, human factors, immunology, microbiology, muscle and bone, neuroscience, pharmacology, and radiation studies. Space science experiments collected cosmic dust over extended periods for analysis on Earth.

MIR played a critical role in the development of the ISS. Phase I of the ISS involved the interactions of NASA and the RSA in using MIR and the Space Shuttle as a test bed for the future development of the ISS. Multiple shuttle flights took place to demonstrate assembly of the ISS by assembling new parts to MIR. EVAs which utilized both the Russian and the US hardware took place. The living environment of MIR was fully analyzed to develop standards for the new station, including air quality, noise levels, and water quality. The effects of crew motions on the structure were measured to help the design of ISS experiment support systems. The external environment of MIR provided valuable information on the external design of the ISS.

Quite possibly the most important benefit from Phase I was to force the internal partners, including the astronauts and cosmonauts, to work in a long-term relationship in a permanently manned space station. While Sályut had international astronauts visit the stations and Spacelab established a long-term relationship between international partners, neither required two separate organizations to work together to support life in space for a long period of time. The ISS Phase I program, using MIR, forced NASA and RSA to solve critical problems together; that required understanding each other.

Phase I drove significant designs of the ISS, including improvements to the rendezvous and docking instruments, "quick-disconnect" cables that come apart in seconds in the case of emergencies, and prevention of moisture buildup. It also provided important lessons for the administration of the ISS. NASA would work on more flexible schedules for astronauts. NASA surgeons learned how to perform medicine on a space station. And, perhaps the most important lesson learned from Phase I was humility; learning that, as with Spacelab, there is a lot to learn after launch. NASA learned how different operating a space station is from the shuttle. [Burrough, 1998]

Appendix C

OTHER REMOTE RESEARCH FACILITIES

This appendix studies remote research facilities currently under operation. Specifically, research in the Antarctica and sub-sea/ocean exploration facilities are reviewed. In both of these cases it is common for the investigations to occur by a limited set of scientists; the full research team is not always present where the research is being conducted. Further, the operators in charge of the facilities are not always the researchers. While sometimes the full research team can be present for Antarctic/Ocean research missions, these facilities present the best models for remote operations of shared facilities.

C.1 Antarctic Research

While human exploration of the Antarctic region dates as far back as the travels of Magellan in 1520, there are two important periods in Antarctic scientific history highly relevant to the ideas of cooperative research and the establishment of remote scientific bases: the years around the 1957-58 International Geographical Year (IGY) and today. The years around IGY are of special importance since they resulted in the creation of the Antarctic Treaty System and the Special Committee for Antarctic Research (SCAR). Before the treaty and SCAR, Antarctica exploration, while significant, had no overall plan, and was mostly driven by commercial interests. A realistic potential of conflict existed by several nations making claim to the land, while other nations simply went through it without making claims, but expecting to be able to cross again. The Antarctic Treaty is a political

agreement created between 12 nations to guarantee the non-military use of the Antarctic continent; over 30 nations now adhere to the treaty (Figure C.1 - from [BAS2] [BAS1] - shows a map of current stations and a picture of research being conducted in a British station). SCAR is a scientific tool for research in the region; it works in parallel and as an advisor to its political counterparts in the Antarctic Treaty to promote science and guide the politics of the region.

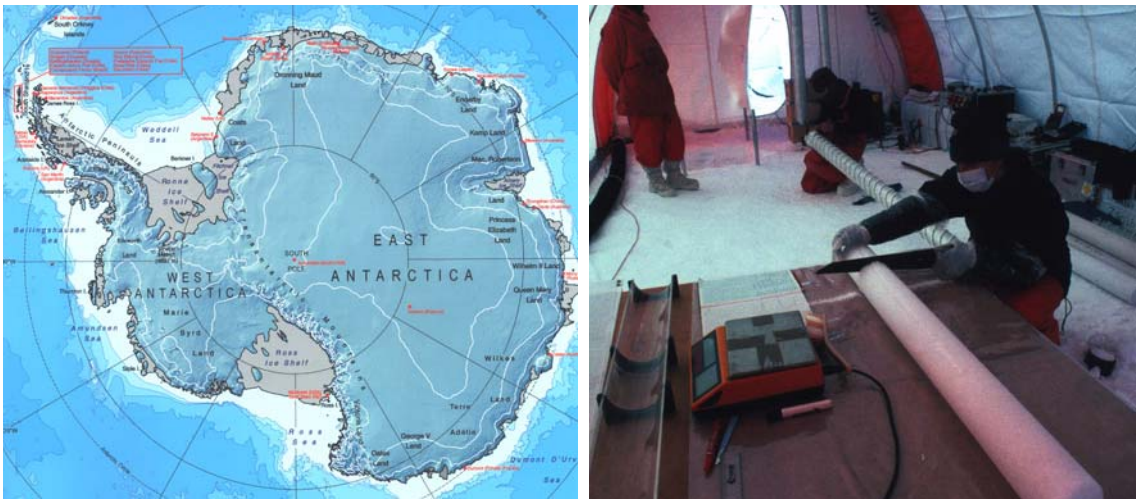


Figure C.1 Antarctic research stations

When discussing the Antarctic Treaty System as a mechanism for scientific research, William F. Budd starts with the following thesis:

> "The main thesis of this chapter is that humankind's quest for knowledge needs to be recognized as the primary motivation for the high level of continued interest and activity in the Antarctic. The treaty nations, through the Antarctic Treaty System, have supported this objective." [NRC PRB]

Like the ISS today, the Antarctic Treaty System and SCAR were formed with science as their primary goal. At a Conference in honor of the 30th Anniversary of the Antarctic Treaty System six primary motives were identified:

1. Basic Research
2. Political - national presence and prestige

3. Economic - natural resources and technology development
4. Military - although against the treaty, recognized as an important military arena
5. Jurisdictional
6. Environmental

All of these motives continue to spur developments and research programs in the Antarctic Region. At the same conference, though, "Finn Sollie [who] was intimately involved in the drafting of the Antarctic Treaty... [expressed that] His major point was that science in fact was the crucial element that made the treaty possible. Without science there wouldn't have been an Antarctic Treaty." [Elzinga, 1993]

The requirements of science ultimately set the guiding principles of the Antarctic Treaty System:

- free access to Antarctic territory
- free use
- free exchange of information
- allowance of inspections of any base by any member country's scientist
- joint planning and execution of activities
- peaceful uses only
- no territorial claims

"Antarctic politics is unique, and truly in a world of its own. While on a national level local issues may determine how a country runs its Antarctic programme and funds its science, on an international level Antarctic politics and science are about co-operation." [Burton, 2004] Nations fund Antarctic science either individually or in collaboration, therefore funding processes differ greatly from nation to nation. But the lack of national boundaries allows collaborations to occur easily. It is not uncommon for scientists to work in bases of countries other than their own; as long as the science is in-line with the standards of the host country (and its scientists agree), bases welcome investigators from across the world.

While the scientific community has achieved an environment which reduces the politics in the Antarctic substantially, they still must endure the physical conditions of the southern continent. To this purpose each country that has a base has established vast logistical support for scientific research. In the case of the United States, for example, there are three different bases (McMurdo Station at 77°53’S, Amundsen-Scott South Pole Station at 90°S, and Palmer Station at 64°46’S). Temporary camps can be setup during the summer months out from McMurdo Station. Automated unmanned data collection can be established; the University of Wisconsin has placed automatic weather stations and supports ice coring and drilling. Several research ships are available (the *Laurence M Gould* and the *Nathaniel B. Palmer* are the primary ones). Several support instruments are also available, such as differential GPS and radars. A vast database of maps, aerial photographs, and bibliographic documents exist to help prepare research missions.

In these large organizations there are multiple challenges. "Logistics has a hardware side and a software side. The latter is the more important, covering the know-how and competence of the people operating vessels, the ship and crew that are there to support the scientists, the helicopter pilots, technicians, consultants, etc. It was pointed out that a ship should always be under the command of the captain, and not of the scientist." [Elzinga, 1993] Scientists conducting research in Antarctica must face social and cultural factors, understanding that issues such as safety may override the scientific needs. The same holds true for the ISS, where even though the astronaut conducts the science *and* manages the spacecraft, they must balance their priorities, and science may not always be the highest. The conclusion in Antarctic research is that the ship’s captain must always be in charge; in the ISS there is always going to be someone in charge over the scientist.

It is also of use to understand the challenges posed by the Antarctic environment, as some of these closely resemble some of the features that define the uniqueness of the ISS. When developing an experiment for operations in the Antarctica, the following factors will play an important role in the design [Ashley, 2004]:

- Transporting large structures is a cumbersome and slow process

- High altitude environment reduces the effects of convection and fans
- Large temperature fluctuations throughout the day and night cause multiple problems
  - Need to insulate for temperatures as low as -80°C
  - Must account for changes of up to 30°C
  - Batteries loose capacity
  - O-ring become brittle
  - Thermal compression and expansion of materials
  - Metals become brittle
- High relative humidity approaching 100%
- Electrostatic damage at room temperature (humidity quickly drops)
- Difficult to maintain exposed equipment

An example of a manner to overcome the difficulties of Antarctic research comes from the development of a robotic autonomous telescope for use in the Antarctica by the University of New South Wales, Australia. The approach has been to minimize the number of computers, addressing many of the issues. Both hardware and software watchdogs watch over the computers. A priority of the computer systems is to prevent reaching ambient temperature (as low as -80°C). They have taken a modular approach to the software environment, allowing scientists to create their own programs for telescope control. Still, even for the design of an autonomous telescope, they conclude that "for the foreseeable future humans will be an essential component in building, operating and maintaining telescopes in Antarctica." [Ashley, 2004]

> "Without pre-existing infrastructure and support capability, conducting frontier science is impossible... for the explorer, engaged in comprehending their surroundings this [be an adventurer] is no longer possible. Nations conducting Antarctic science go to great lengths to provide facilities that are safe and practical for their inhabitants. For every scientist present, four or five people are there to support them... Nations that don't put this effort into their stations... are not in Antarctica to do science." [Burton, 2004]

Researchers in Antarctica do have the opportunity to be present where they plan to conduct research, although the conditions are not ideal. During the summer months the bases

are overcrowded; the winter months have very limited daylight. Schedules are an important part of the scientists routines. To start, every instrument must be tested at home before deployment to Antarctica. The base stations do have supplies to repair equipment, although it may take time to schedule the available support personnel or the facilities needed to conduct the repairs. Antarctic researchers have the benefit that obtaining parts from their home laboratories is only a week away during the summer months. The schedule plays an important part for many scientists that must leave before winter, since it is essential to finish their research before then.

For those that reside in the bases during the winter, they will face inter-personal challenges, where it is impossible to avoid those living in the base. Communications have highly improved the conditions of conducting research in the Antarctic. All staff in the base can communicate with the rest of the world and be informed. When scientific teams get split those residing in the base can contact the rest of the team via video conference. Through extended communications research occurs at Antarctica year-round.

Antarctic research that does not require the scientist to conduct fieldwork also occurs. Locations such as the Antarctic Research Facility (ARF) in Florida State University have been setup to provide access to Antarctic research. The ARF services include vast literature through journals and books, photography and map collections. The facility also provides access to specialized equipment such as x-ray sensors, diffractors, and imagers, digital and analog photography, and sediment processing facilities. The ARF collects samples from the Antarctica through the US vessels and makes them available to scientists; the ARF even supports special requests for samples.

Even if some facilities exist to permit off-site research, the vast investment in supporting fieldwork in the Antarctica clearly points to the fact that the presence of humans in the research environment remains critical. Every nation that has a scientific presence in the Antarctica provides a large number of support personnel and equipment. In many cases thousands of support staff help hundreds of scientists.

There are several lessons to be learned from Antarctic research. Possibly the most important lesson lies in the development of the Antarctic Treaty, where science played the driving force, rather than politics. The vast success in cooperation between nations that conduct science in the Antarctic is overwhelming. There are also lessons regarding remote operations. The practices to develop experiments which operate in a harsh environment, where simple repairs and operations are not possible, provide valuable lessons for the design of ISS experiments. Technological improvements in communications have helped both the operations and science in the Antarctic. But, at the same time, the environments have been set up to ensure that at least a portion of the scientists involved in the research are present. Therefore, Antarctic Research history says that when possible it is best to have the researcher conduct the experiments. This means that the ISS must provide enhanced capabilities such that astronauts conducting research remotely can communicate effectively with the scientists on Earth.

C.2 Ocean Research and Exploration

The challenges incurred in ocean research closely resemble those of space research. The design and operation of ocean facilities must allow for humans to conduct research in a harsh environment: life support is essential; the presence of humans is limited; mechanical tools must enhance the human ability to manipulate the environment; in many cases the facility operates separate from its home station - it must ensure safe return to its base; communications play an essential role in the ability to conduct science; and the vessel must support all of these functions on its own power.

Ocean engineering manned vehicle systems can be grouped into four primary areas, just like microgravity facilities. These areas are [Penzias, 1973]:

- Conventional diving systems - these range from sponge divers and SCUBA gear to hard-hat body suits which allow humans to explore in shallow depths. Humans are usually exposed to the environment, and as such the human performs most of the actuation directly. The gear’s primary task is for life support.

- Saturation diving systems - these systems are composed of large pressurized chambers which allow humans to live and work at pressure continuously for weeks or months at a time. They are mostly used for observation and study of human physiology in pressurized environments.
- Fixed bottom stations - fixed stations create an under-water shirt-sleeve environment for long-term human presence in the bottom of the ocean. As with saturation systems, these stations provide spaces for observation and exposure. The larger size of the stations allows more instruments which can interact with the ocean environment, but the location of the station is fixed, so the exploration capabilities are limited.
- Submersible vehicles - while these systems include large submarines, this research concentrates on submersible work-boats for research. These vessels usually carry: a crew of two to seven in a shirt-sleeve environment. Each time the vessel is deployed it is intended to carry a specific experiment. To accomplish this mission the vessels can carry special actuators and sensors to conduct the necessary investigations of the ocean environment. It is also possible for the vehicles to have a lock to allow human to exit and re-enter.

A closer look at the functional requirements of submersible research vehicles further points out the similarities between Ocean research vessels and spacecraft. Ocean research vessels must have the ability to traverse across the oceans. They need to pick up and reposition objects and samples. Scientists require the ability to see, via both visual and electronic methods. The vehicles must endure their missions over extended periods of time. For all missions there is a need to have special instruments. In some cases vehicles must be able to mate/dock with each other and/or with their bases. Researchers may require that humans be able to leave the vehicle, requiring a lock. [Penzias, 1973]

A review of the major sub-systems of a manned submersible [Penzias, 1973], presented in Table C.1, shows the close relationships between ocean and space research facilities. Both types of vehicles must have a pressurized cabin with their respective life support systems. The propulsion of the vehicle must provide both coarse control and fine maneuvering control; this control must be supported by navigation systems. In both cases a ground/surface station is necessary. The principal difference is that, while Ocean vehicles require flotation control systems, space vehicles require a launch system to place them into orbit.

Overall, though, the similarities between the two predict that the challenges are answered in similar ways.

TABLE C.1 Major sub-systems of space and ocean research vehicles

Ocean Vehicle	Space Vehicle
Pressure Hull or Cabin	Pressurized Cabin
Structure (beyond pressure hull)	Truss elements
Flotation	Launch Vehicle
Power	Power
Propulsion	Propulsion
Maneuvering	Maneuvering
Life Support	Life Support
Navigation	Navigation
Work Space	Research Space
Surface Support	Ground Station

[Cunninghman, 1970] studied the design of an ocean engineering vessel in the 1970's. The study included a survey of researchers to identify the most important tasks that needed to be performed in such a vessel. This survey identified the functional requirements of the vessel. There were some interesting results relevant to the design of microgravity laboratories:

- Four features were considered luxuries, that is, they are not high priority for implementation; of these features one is relevant to microgravity laboratories: real-time data analysis. The survey concluded that many scientists preferred to collect the highest amount of data possible for post-analysis, rather than invest in real-time data analysis.
- Large laboratory areas were of low priority. Instead, the scientists preferred to have large deck areas. Deck areas are large common spaces for the deployment of experiments. Rather than limit the capabilities of their equipment due to size limitations, scientists preferred to have smaller analysis areas in the vessel, and perform more in depth analysis off the vessel. This is not to say that laboratory areas were not needed. Five specific laboratory areas (structures and materials, instrumentation, chemical and biological,

electrical and electronic, and human engineering) and two general purpose laboratory areas (machine shop and computer center) were identified.

- It appeared from the analysis that scientists would like to conduct more than one experiment at a time; the results called for multiple hoists to deploy experiments.

Current ocean research vehicles at the Woods-Hole Oceanographic Institute [WHOI, URL] show that the results of the early survey still apply in general. New technology allows integration of more tools for data analysis (both for real-time and post-processing) into new vessels, but the primary goal of the vessels remains to provide human support and the deployment of equipment for exploration and research. The most important functions of these vessels are to:

- support missions of days to months
- provide navigation and communication systems
- support for multiple projects
  - availability of multiple winches for experiment deployment
- carry submersibles
  - provide space for scientists: manned submersibles require only one pilot and have space for one or two scientists.
  - allow unmanned observation and sample recovery

Figure C.2 WHOI research vessels Knorr (left) and Alvin

The design of submersible research vessels closely matches the design needed for a space-based research laboratory, such as the ISS. The challenges and results of the design of sea vessels can be applied to some aspects of the designs for microgravity facilities. Yet, the review of both past and current ocean engineering systems shows a trend similar to that of Antarctic research: take the scientists to the place of research.

Appendix D

THE INTERNATIONAL SPACE STATION

This appendix presents an review of the International Space Station Program and the facilities that are or will be available for research. First, the appendix reviews the objectives of the ISS and the identified research directions for the station. Next, it presents an overview of all the ISS components. That is followed by a more in depth review of the components which directly support research aboard the ISS. The appendix ends with a presentation of the identified challenges of the ISS and expected upgrades to the program to overcome these challenges. Chapter 2 utilizes this review to identify the most important resources provided by the ISS.

D.1 Objectives and Research Directions

The objectives of the ISS as stated in the ISS Familiarization Manual developed by NASA are:

> "The purpose of the ISS is to provide an “Earth orbiting facility that houses experiment payloads, distributes resource utilities, and supports permanent human habitation for conducting research and science experiments in a microgravity environment.” (ISSA IDR no. 1, Reference Guide, March 29, 1995)
>
> "This overall purpose leads directly into the following specific objectives of the ISS program:
> - Develop a world-class orbiting laboratory for conducting high-value scientific research

- Provide access to microgravity resources as early as possible in the assembly sequence
- Develop ability to live and work in space for extended periods
- Develop effective international cooperation
- Provide a testbed for developing 21st Century technology."

[NASA, 1998]

After creating these objectives, NASA worked to further detail the research objectives of the ISS. To this purpose, NASA has created an ongoing program to determine the "research directions" of the ISS. During the development of these directions, NASA first defined the ISS as a special type of laboratory, one which has three special purposes:

- "an *advanced testbed* for technology and human exploration;
- a *world-class research facility*; and
- a *commercial platform* for space research and development." [NASA, 2000]

As of January 2000 the NASA Office of Life and Microgravity Science Applications had identified a number of research fields which can directly use the resources provided by the ISS to advanced human knowledge and provide benefits to the people in the ground. The identified fields include (adapted from [NASA, 2000]):

- **Biomedical research and countermeasures / Advanced human support technology** - what knowledge and technology are necessary for humans to live and function productively beyond the Earth’s surface.
- **Biotechnology** - produces and characterizes biological molecules and assemblies important to basic and clinical research.
- **Combustion science** - the basic mechanisms of combustion can be more easily studied in microgravity, where scientists can make observations and measurements of combustion and the systems and processes it enables.
- **Fluid physics** - the universal nature of fluid phenomena, which affect everything from transport dynamics in the human body to the mixing characteristics of the atmosphere, makes this research fundamental to all areas of science and engineering.
- **Fundamental physics** - microgravity enables the development of uniform samples, the free suspension of objects, and the lack of mechanical disturbance on experimental subjects to test the fundamental physical laws.

- **Gravitational biology and ecology** - studies in gravitational biology and ecology seek to advance our understanding of how the ubiquitous force of gravity affects the many stages of plant and animal life.
- **Materials science** - the ISS gives researchers to study the relationships between the structure, properties, and processing of materials without buoyancy-induced convection, sedimentation, or hydrostatic pressure. Microgravity also provides the chance to investigate “containerless processing”.
- **Space science** - from its orbital position, the ISS affords researchers a long-term “window on the universe” from which to study the structure and evolution of the cosmos.
- **Engineering research and technology development** - advances in engineering research and technology development (ERTD) can help reduce costs and improve the performance of future government and commercial activity in space, and enhance the quality of life on Earth. ISS research helps validate the technologies for long-duration space exploration, power generation and storage, robotic manipulation capabilities, automatic maintenance, and spacecraft control. New applications, processes, and technologies promise to benefit the telecommunications, water and power, construction, and other industries on Earth.
- **Space product development** - commercial researchers will springboard off of basic science and engineering to use the knowledge gained from the ISS research to create new products and processes to benefit the medical and pharmaceutical fields, the electronics and chemical industries, and the engineering community, among others.
- **Earth science** - the orbit of the ISS will cover about 85% of our planet’s surface, making it a useful platform for ongoing Earth science research to assess the global trends such as: atmospheric and climate change; weather patterns; vegetation and land use patterns; and food, water, and mineral resource use.

D.2 Components of the ISS

Figure D.1 shows the expected configuration of the ISS at US Core Complete, as of July 23, 2004. Once this configuration is achieved, the ISS will be composed of the following major modules (the following descriptions were adapted from [NASA, 1998] to reflect US Core Complete):

- **Node** - The Node is a U.S. element that provides six docking ports (four radial and two axial) for the attachment of other modules. It also provides

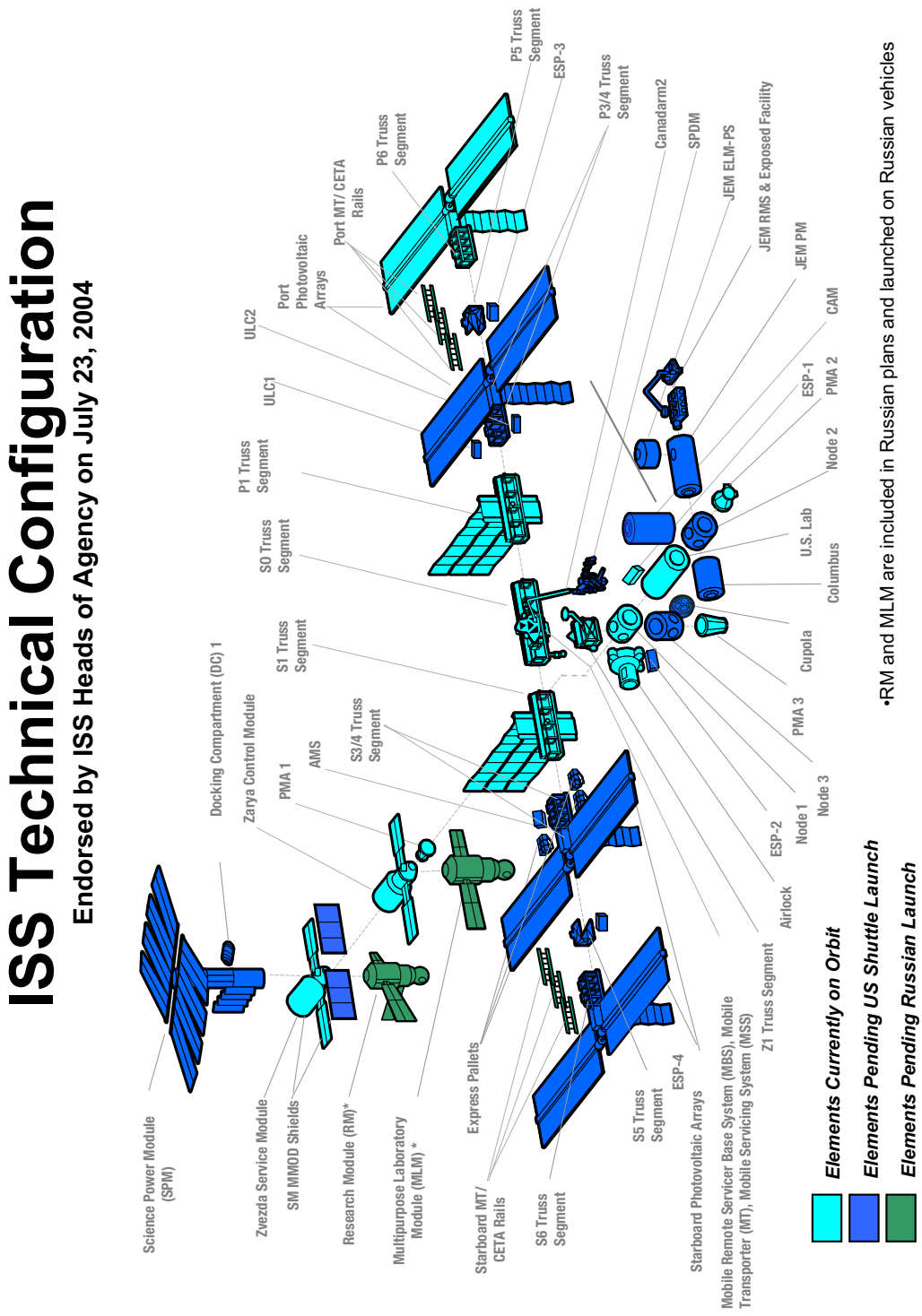


Figure D.1 The ISS at US Core Complete

external attachment points for the truss. The Node provides internal storage and pressurized access between modules. There are three Nodes.

- **Service Module** - The Service Module (SM) provides the Station living quarters, life support system, communication system, electrical power distribution, data processing system, flight control system, and propulsion system. Living accommodations on the Service Module include personal sleeping quarters for the crew; a toilet and hygiene facilities; a galley with a refrigerator/freezer; and a table for securing meals while eating. Spacewalks using Russian Orlan-M spacesuits can be performed from the SM by using the Transfer Compartment as an airlock.
- **Soyuz** - Besides being an Earth-to-Orbit Vehicle (ETOV) used for crew rotations, Soyuz is the Russian element that provides the crew emergency return ("lifeboat") capability. As such, there is always a Soyuz docked to the Station whenever the Station crew is onboard. At least every 6 months, the docked Soyuz is replaced with a "fresh" Soyuz.
- **Laboratory** - The Lab is a U.S. element that provides equipment for research and technology development. It also houses all the necessary systems to support a laboratory environment and control of the U.S. Segment.
- **Multi-Purpose Logistics Module** - The MPLM allows transfer of pressurized cargo and payloads. It is launched on the Shuttle and berthed to the Node, where supplies are off-loaded and finished experiments are loaded. The MPLM is then re-berthed in the Shuttle for return to Earth.
- **Joint Airlock** - The Joint Airlock is a U.S. element that provides Station-based Extravehicular Activity (EVA) capability using either a U.S. Extravehicular Mobility Unit (EMU) or Russian Orlon EVA suits.
- **Docking Compartment** - The Russian element Docking Compartment (DC) is used during the assembly sequence to provide egress/ingress capability for Russian-based EVAs and additional docking ports.
- **Truss** - Built over numerous flights, the truss is a U.S. element that provides the ISS "backbone" and attachment points for modules, payloads, and systems equipment. It also houses umbilicals, radiators, external payloads, and batteries.
- **Science Power Platform** - The Science Power Platform (SPP) is a Russian element that is brought up by the Shuttle to provide additional power and roll axis attitude control capability.
- **Japanese Experiment Module** - The Japanese Experiment Module (JEM) is a Japanese element that provides laboratory facilities for Japanese material processing and life science research. It also contains an external platform, airlock, and robotic manipulator for in-space ("exposed") experiments and a separate logistics module to transport JEM experiments.

- **Cupola** - The Cupola is a U.S. element that provides direct viewing for robotic operations and Shuttle payload bay viewing.
- **Research Module** - The Research Module (RM) is a Russian element that provides facilities for the Russian experiments and research. It is analogous to the U.S. Lab.
- **Columbus Orbital Facility, Also Known as the Attached Pressurized Module** - The Columbus Orbital Facility (COF) is an European Space Agency (ESA) element that provides facilities for the ESA experiments and research. It is analogous to the U.S. Lab.
- **Centrifuge Accommodations Module** - The Centrifuge Accommodation Module (CAM) is a U.S. element that provides centrifuge facilities for science and research. It also houses additional payload racks.
- **Logistics Vehicles** - Logistics flights are required throughout the life of the ISS and will be accomplished using a variety of vehicles.
  - The Shuttle will be used to bring water, and pressurized cargo. When the Mini-Pressurized Logistics Module (MPLM) is used, the Shuttle can bring nearly 9 metric tons of pressurized cargo to ISS. The Shuttle is also the only means for returning items intact from ISS.
  - The Progress M1 is provided by RSA and used to accomplish three primary tasks: orbital reboost, attitude control fuel resupply, and pressurized cargo resupply. It will be launched on a Soyuz booster. Pressurized cargo includes oxygen, nitrogen, food, clothing, personal articles, and water. The Progress is filled with trash as its stores are consumed, and when exhausted, undocks, deorbits, and re-enters the atmosphere over the Pacific Ocean.
  - The Autonomous Transfer Vehicle (ATV) is provided by ESA and is scheduled to be completed in 2006. It will be launched on an Ariane V launch vehicle. It is roughly three times as large as the Progress M1, but is functionally the same as described above.
  - The H-2 Transfer Vehicle (HTV) will be provided by National Space Development Agency of Japan (NASDA) and is under design. It will launch on a H-2A launch vehicle. Its purpose is to carry pressurized cargo only. Unlike the Progress M1 and ATV, the HTV doesn’t carry resupply fuel, and it doesn’t dock. It rendezvous to the forward end of the Station and is grappled by a robotic arm and berthed.

D.3 ISS Facilities for Research

The previous section presented an overview of all the components that compose the International Space Station. Several of those elements have been designed specifically to support research aboard the ISS to fulfill the research directions listed in Section D.1. This section describes the special elements of the ISS which support research, summarizes their capabilities, and lists the available resources for scientists. The section first presents further description of the modules; then the section describes the resources available to scientists.

D.3.1 ISS Modules for Research

The following summaries of the research modules were adapted from [NASA, 2000b]

- **US Destiny Laboratory -** The U.S. laboratory, named Destiny, is the module where a significant portion of the pressurized U.S. research will take place. The module overview is presented in Figure D.2 [NASA, 2000b]. Destiny will accommodate up to 13 ISPR research racks. Destiny is the first research module installed on the Station. The side of Destiny that faces Earth for the majority of possible ISS flight attitudes contains a circular window of very high optical quality design where the Window Observational Research Facility (WORF, see below) will reside.
- **US Centrifuge Accommodation Module (CAM)** - The CAM (pictures in Figure D.3 [NASA, 2000b]) is a laboratory dedicated to U.S. and cooperative international gravitational biology research. It houses a 2.5m diameter centrifuge, which is the essential component of a larger complement (multi-user facilities) of research equipment dedicated to gravitational biology. In addition, 9 locations are provided for passive stowage racks.
- **US Integrated Truss Attachments** - There are four dedicated sites on the starboard side of the ISS truss where external payloads can be attached. The general location of these attach points is indicated in Figure D.4 [NASA, 2000b]. There are two attachment points on the nadir, or Earth-facing, side of the truss, and two on the opposite, or zenith, side of the truss. Physically the attach points consist of a system of three guide vanes and a capture latch used to secure the payload, as well as an umbilical assembly to mate utilities and connections. Resources are given for a single payload occupying an entire truss-site; an EXPRESS pallet, which occupies an entire truss site, provides an equally sub-divided surface for up to six individual payloads with standardized interfaces for payload integration.

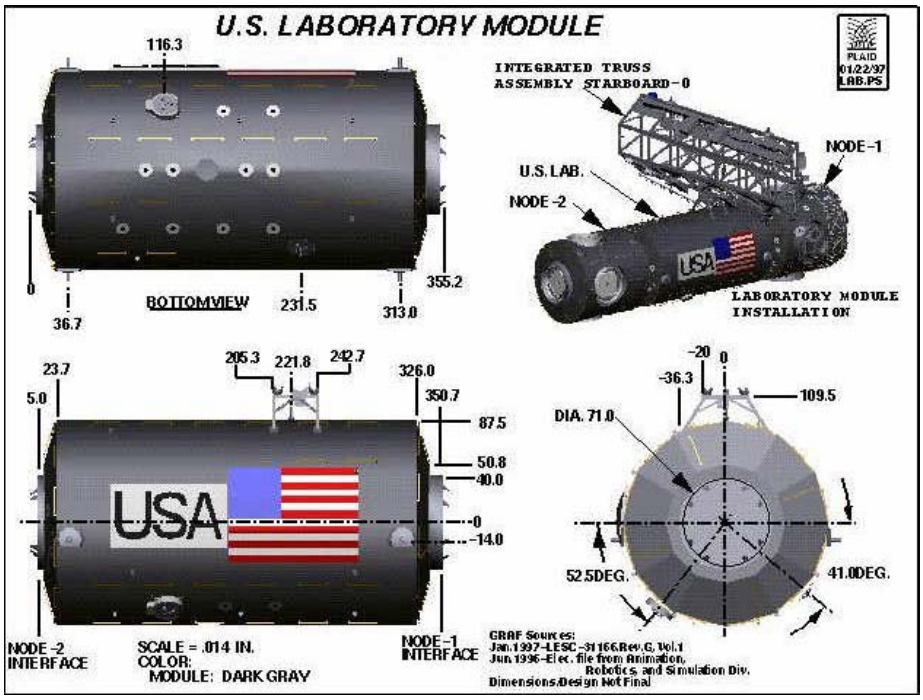


Figure D.2 US Destiny laboratory

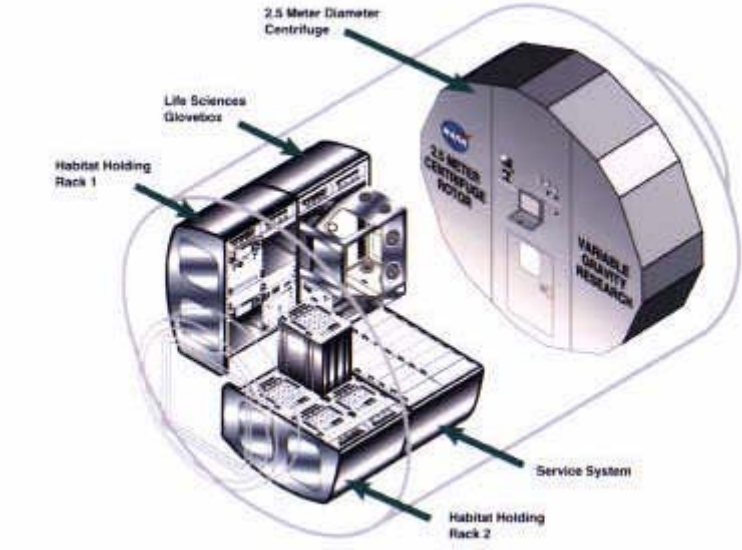


Figure D.3 US Centrifuge Accommodation Module

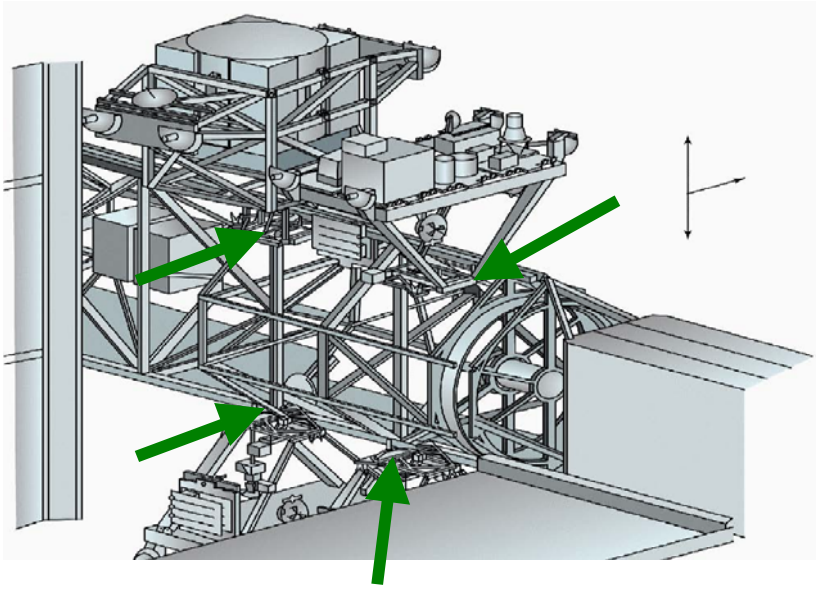


Figure D.4 US Truss Attachment Points (4)

- **Japanese Experiment Module (JEM)** - The Japanese Experiment Module (JEM), also known by the name Kibo, is the segment of ISS developed by the National Space Development Agency (NASDA) of Japan for the purpose of supporting research and development experiments in Earth orbit. Figure D.5 [NASA, 2000b] shows the several major systems of the JEM:
  - JEM Pressurized Module (JEM-PM) is a laboratory for experimental research in areas such as space medicine, life sciences, materials processing, and biotechnologies. It contains an airlock to transfer experiments. The module provides 10 ISPRs for research payloads.
  - JEM Exposed Facility (JEM-EF) is an un-pressurized pallet structure exposed to the environments of space to support user payloads for the purpose of experimental research in areas such as communications, space science, engineering, materials processing, and earth observation. A total of 10 payload sites are provided
  - The Pressurized Section (ELM-PS) and Exposed Section (ELM-ES) serve as a pressurized and exposed passive storage, respectively.
- **Colombus Module** - The European Space Agency (ESA) Columbus module, pictured in Figure D.6 [NASA, 2000b], provides 10 ISPRs for research payloads. Columbus is designed as a general-purpose laboratory to support ESA-defined scientific disciplines in the areas of materials and fluid sciences, life sciences and technology development. An Exposed Payload

Facility with two separate support structures attached is expected to be added to the Columbus Pressurized Module.

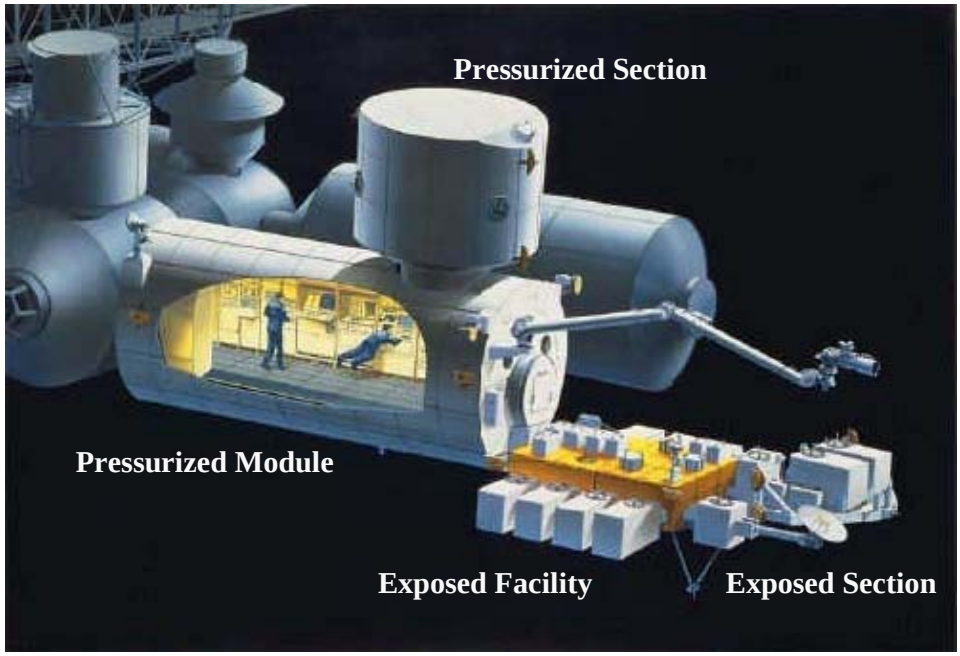


Figure D.5 Japanese Experiment Module

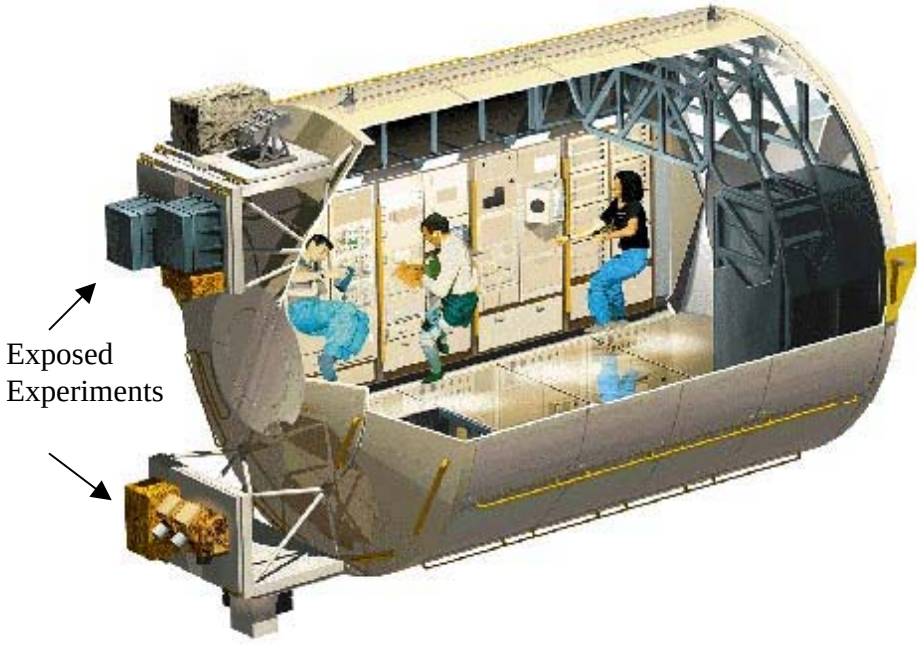


Figure D.6 Columbus Module

- **Russian Segment** - The Russian segment is slated to have two research modules to support research payloads generated by Russian researchers. The segment will have power, data, and other systems separate from the rest of the ISS.

D.3.2 ISS Resources for Research

The following resources are available to scientists at one or more of the research modules provided for research. While some resources are specifically intended for operations in the pressurized area of the modules, several are also available to exposed experiments.

- **Controlled environment** - the ISS operates an altitude of 350-460km and an inclination of 51.6° to the equator. It flies over 85% of the globe and 95% of the Earth's population. Steady state accelerations are maintained between 1-2μg during standard operations in the laboratory modules; vibration is controlled in the 0.01-300Hz frequency range, with a maximum RMS of 1mg at 300Hz. The pressure, air composition, relative humidity, and temperature are also controlled and monitored. The ISS also provides multiple sensors to monitor the external environment as well as controls to prevent contamination of the "exposed" experiments.
- **Power** - At US Core Complete the ISS will provide 26 kW minimum continuous and 30 kW average power during Standard and Microgravity modes.
- **Payload Data Handling and Communication** - A 72 kbps S-band forward link is used to send commands for payloads, while a 150 Mbps Ku-band system is used for the payload downlink. Three types of connections are available for this distribution: 1) a MIL-STD- 1553B Payload Bus, 2) an 802.3 Ethernet, or 3) a fiber-optic High-Rate Data Link (HRDL). Included in the data rate are four compressed channels of video downlink, and three channels of video uplink. The Payload Multiplexer/Demultiplexer provides 300 megabytes of nonvolatile mass storage for payloads.
- **Thermal Management** - Thermal radiators are positioned on the Integrated Truss Structure. Using H_2O internally, the radiators pick up heat from the ISS through the environmental control system, which then transfers that heat to the radiators using NH_4 as the active heat transfer fluid.
- **Cryofreezer System** - the Cryogenic Freezer System will maintain samples at or below -183 °C (-297.4 °F) throughout a mission life cycle. It will be used to preserve plant and animal cell fine anatomy, ultra structure and genetic material. The 35-liter (1.2 ft^3) internal volume of the storage freezer will accommodate 1000 or more 2- to 5- ml (0.1- to 0.3- in^3) sample vials.

- **Minus Eighty-degree Laboratory Freezer** (MELFI) - The Minus Eighty-degree Laboratory Freezer for ISS will maintain samples below -68 °C for experiments which all biochemical action to be stopped but do not require cryogenic temperatures. Cell culture media, bulk plant material, and blood, urine and fecal samples are examples of the types of items that will be stored in the MELFI.
- **Payload Stowage** - The majority of stowage on ISS is accommodated by International Standard Payload Rack (ISPR)-size racks that provide compartmentalized storage. The types of stowage racks currently being developed are the Resupply Stowage Rack (RSR, 1.1m^3) for storage of miscellaneous items on individual storage trays; the Zero-G Stowage Rack (ZSR, 1.2m^3), which uses a collapsible shell and a fabric insert to store things within the ISS only, but not for transport; and the Resupply Stowage Platform (RSP, 1.2m^3) to transport cargo using the fabric shells of the ZSR. The ISPR racks provide a number of resources to scientists:
  - 1.6 m3 (55.5 ft^3) of internal volume to accommodate up to 700kg of equipment.
  - Support of half-sized payloads, such as the Spacelab Standard Interface Rack (SIR) and the Space Shuttle Middeck Locker.
  - A 3 kW power feed and a 1.2 kW auxiliary feed for the payloads at 120 Vdc/25A average. Selected locations provide 6kW and 12kW power capability.
  - A moderate-temperature water loop is provided at an inlet temperature range of 16-24 °C (61-75 °F).
  - A MIL-STD-1553B Payload Bus provides command and data processing capabilities at each ISPR location.
  - EIA-RS-170A optical pulse frequency modulated video signals are available in the US and ESA modules; the JEM distributes video using twisted shielded wire pairs. Seven video recorders and multiple video monitors can be connected to the ISPR signals.
  - All of the laboratory modules support a waste gas exhaust system that is vented to space via a 2.5 cm gas line.
  - Selected locations provide a 2.5 cm vacuum resource line.
  - A 0.95 cm nitrogen line is provided as a standard service
  - Carbon dioxide, argon, and helium are provided to selected locations in the JEM.
  - The Active Rack Isolation System (ARIS) is designed to isolate payload racks from vibration such that the on-rack environment will meet the sys-

tem vibratory specifications. Currently, eleven ARIS-equipped racks are planned for investigator use.

- Sub-Rack Accommodations and EXPRESS Rack - Scientists can use the standard stowage presented above, or build their own equipment based on a Middeck Locker Equivalent (MLE). The EXpedite the Processing of Experiments (EXPRESS) rack concept allows quick integration of sub-rack payloads to the ISS resources described above ($0.06m^3$, 2kW @ 28Vdc, 2kW head rejection, some with ARIS).

D.3.3 Multi-user Facilities

The ISS program originally planned for the development of a wide variety of multi-user facilities. Multi-user facilities provide general equipment for research on a specific area, but are not individual experiments on their own. They are intended to form a key part of the ISS infrastructure, providing scientists with basic equipment for use in their experiments. The design of these facilities is modular, such that individual scientists can design components uniquely suited for their experiment needs which attach to the ISS provided facilities. Table D.1 [NASA, 2000b] presents the original list of multi-user facilities intended for the ISS. While some of the following facilities are no longer expected to be deployed, it is relevant to list all of the concepts developed by NASA and its international partners originally, as it showcases the need to develop multi-user facilities for a wide range of scientific fields.

It is interesting to see that out of the 22 multi-user facilities originally planned for the ISS, only one is for the advancement of space technology (AHSTF). Further, that facility is directly related to the life of humans in space, and not to the advancement of space technology outside of the human physiology. This thesis will study the creation of multi-user facilities for space technology, rather than the biological and physical sciences; the planning of the multi-user facilities for the biological and physical sciences in the ISS is a positive reinforcement to the idea of multi-user facilities for space technology.

TABLE D.1 Originally planned multi-user facilities for the ISS

Facility Name	Sponsor	Description
Human Research Facility	NASA	Physical and physiological changes in humans due to space flight.
Gravitational Biology Facility	NASA	Effect of space environment on biological systems.
Biotechnology Research Facility	NASA	Mammalian cell culture, tissue engineering, biochemical separations, and protein crystal growth.
Fluids and Combustion Facility	NASA	Fluid physics and combustion science research.
Microgravity Sciences Glovebox	NASA/ESA	Crew-manipulated investigations for a variety of experiments.
Materials Science Research Facility	NASA	Solidification of metals and alloys, thermophysical properties, polymers, crystal growth, and ceramics.
Window Observational Research Facility	NASA	Geologic, climatologic, atmospheric, and geographic research.
X-Ray Crystallography Facility	NASA	Protein crystal analysis facility for macromolecular crystals.
Advanced Human Support Technology Facility	NASA	Research on technology to sustain human life during long duration space missions.
Low-Temperature Microgravity Physics Facility	NASA	Low temperature exposed laboratory to study fundamental physics in space.
Fluid Science Laboratory	ESA	Study dynamic phenomena of fluid media.
European Physiology Modules	ESA	Respiratory, cardiovascular, hormonal, body fluids and bone conditions, as well as neuroscience.
Gradient Heating Furnace	NASDA	High temperature (up to 1600°C) zone-type furnace with vacuum.
Advanced Furnace for Microgravity Experiments with X-ray Radiography	NASDA	In-situ observation with X-ray radiography of heated/melted samples.
Electrostatic Levitation Furnace	NASDA	Study containerless sample processing.
Isothermal Furnace	NASDA	Solidification and diffusion of samples with uniform temperature profile.
Cell Biology Experiment Facility	NASDA	Controlled environment for research on small plants, animals, cells, tissues, and microorganisms
Clean Bench	NASDA	Closed workspace for aseptic operations with life sciences and biotechnology.
Fluid Physics Experiment Facility	NASDA	Support fluid physics experiments.
Solution/Protein Crystal Growth Facility	NASDA	Protein crystal growth in ground and microgravity.
Image Processing Unit	NASDA	Image capture for other experiment facilities.
Aquatic Animal Experiment Facility	NASDA	Accommodates freshwater and saltwater organisms in microgravity.

D.4 Engineering and Operational Challenges of the ISS

In order to better understand the resources available in the ISS, and more specifically to identify those resources which set it apart from other types of microgravity facilities, it is useful to understand which issues have been identified as the most important limiting factors to the correct utilization of the ISS for research purposes. Reports by NASA ([Durham, 2004], [O'Neill, 1999]) and the NRC ([NRC, 2000], [NRC, 2001], [NRC, 2002]) have identified specific issues related to the ISS which directly affect research. The resources identified by these issues can greatly benefit research aboard the ISS; limitations on the availability of these resources limit research opportunities. Because of this direct relationship, these resources are of special importance to make the best use of the ISS for research.

The major challenges and issues identified over the past years, directly related to conducting research on the ISS and listed as recommendations within the mentioned reports, are:

- Communications:
  - Increase of communications bandwidth
  - Enable video communications
  - Establish continuous communications
  - Allow crew to communicate directly with principal investigators (PI's)
  - Provide PI's with direct electronic access to their experimental data
  - Develop better communications tools and interaction processes to support the multi-national, multi-time-zone utilization of the ISS
- Facilities
  - Refine payload computing architectures continuously
- Crew
  - Reassess the crew's activities; allow the crew to take part of daily timeline development
  - Reconsider the use of payload-specialists for ISS missions
  - Reconsider the posibility to use some components of the ISS as a "safe-haven" to enable the return of a crew of seven

  - Use robotics to increase crew availability
- Approach to Research
  - Pursue revolutionary approaches to develop new EVA technologies
  - Create an inter-disciplinary research prioritization plan
  - Utilize Space Shuttle missions to maintain interest by the science community until the ISS is ready for operations
  - Expand cooperation with international partners to maintain their interest

The recommendations are grouped into four main areas: communications, facilities, crew, and approach to research. The emphasis on communications and crew among the several reports is of importance to this thesis, since it indicates that those two special features of the ISS are relevant to a wide range of ISS users and NASA itself. It is interesting to see that the recommendations on communications are guided towards creating a stronger and more direct presence of the investigators in the everyday operations of experiments aboard the ISS. The recommendations call for the use of real-time video for the PI to be virtually present in the ISS; they also intend to make data more available to scientists.

The recommendations also put emphasis on the crew availability and expertise. One of the reports ([NRC, 2002]) goes as far as to argue that NASA reassess the ability to support seven astronauts, even without a crew return vehicle. Two reports call for the re-evaluation of payload-specialists to become part of ISS crews (when the crew size was expected to be seven astronauts). All the reports argue that the crew should have an important role in deciding daily timelines to maximize their productivity. As a whole, the reports put heavy emphasis on utilizing the crew as best as possible. Minimizing their involvement in repetitive tasks should be automated, so as to maximize their time involved in conducting science. It is important to note the difference between simply minimizing the astronaut time needed for an experiment and minimizing time involved in repetitive tasks. The reports do not call for the automation of all experiments, rather they call to maximize the crew time availability to perform productive research.

The recommendation to "pursue revolutionary approaches to develop new EVA technologies" also stands out. Through this recommendation, a large number of important scientists not only push for research, they call for the ISS to become a laboratory where technology matures substantially. The recommendation argues that the experiments conducted aboard the ISS must help to substantially advance the scientific knowledge and technological capabilities of space flight. As related to the TRLs presented in Chapter 1, this recommendation calls for technology maturation up to TRLs 6/7/8, demonstrating full technologies, ready for deployment to their final missions.

Lastly, we can look at the upgrades that NASA has planned for the ISS after the US Core Complete assembly changes have been decided. In [Durham, 2004] members of the ISS Payloads Office list the latest infrastructure upgrades to the ISS; a summary of these upgrades is presented in Table D.2 [Durham, 2004].

Once again, we note that the intended upgrades to the ISS concentrate on the areas of communications and crew. The projects stress the need to transfer data better, both within the ISS and to ground; increases in bandwidth of up to 30 times are expected. Some of these upgrades are to help the crew to improve their interfaces with ISS equipment and with ground personnel, ultimately maximizing their useful time to perform research and minimizing the time they need to perform procedural tasks.

TABLE D.2 ISS Infrastructure Upgrades

Project	Description	Current Constraint	Upgrade Goal
Integrated Station LAN (ISL)	Replace the existing three separate networks in the ISS into a single "all-ISS" network to enhance data delivery to ground and enable astronaut access from all modules.	Data rates & Limited Connectivity	Flexible 1Gbps network
Timeliner	Develop a procedural and logical scripting tool to automate execution of procedures aboard the ISS with three levels of crew intervention (manual, crew-controlled, and automatic).	Manual procedures execution	Automated procedure execution
ISS Downlink Enhancement Architecture (IDEA)	Remove the ground-based and DOMSAT bottlenecks which restrict the Ku band downlink to 50Mbps and implement a high bandwidth WAN.	50Mbps downlink	150Mbps downlink
Mission Operational Voice Enhancement (MOVE)	Standardize voice communications among all NASA centers and the ISS with the option to add features such as video teleconference and multiple site support.	Obsolete/non-maintainable voice communications	Common all-NASA mission communications systems
Ku→Ka Band Transition	Initial study to increase the downlink bandwidth of the ISS to a higher-frequency band.	Ku band capacity & int'l use agreements	Ka band data capacity > 1.5Gpbs

Appendix E

MIT SSL PREVIOUS MICROGRAVITY EXPERIMENTS

The MIT SSL has designed, built, and operated a multitude of flight experiments in the past. The lessons learned from these experiments led to the development of the MIT SSL Laboratory Design Philosophy presented in Chapter 3. The experiments utilized to reach the philosophy are:

- Mid-deck 0-g Dynamics Experiment (MODE), which flew on STS-48 in September 1991 and its re-flight on STS-62 in March 1994.
- Dynamic Load Sensors (DLS), which flew on MIR for about three years.
- Middeck Active Control Experiment (MACE), which flew on STS-67 in March 1995.
- MACE re-flight, which was the first crew-interactive space technology experiment conducted aboard the ISS by Expedition 1 in December 2000.

This appendix describes each family of experiments in further detail, with emphasis on the identification of the features of the MIT SSL Laboratory Design Philosophy.

E.1 MODE & MODE Re-Flight

MODE. The Middeck 0-Gravity Dynamics Experiment (MODE) flew as a facility for measuring the nonlinear dynamics of fluid slosh and jointed truss structures. Scaled deployable trusses of different geometries, developed by ABLE Engineering, were excited at different forcing levels to measure the amplitude-dependent shifts in frequencies and

damping as compared to equivalent 1-g tests. McDonnell Douglas Astronautics collaborated with MIT on this part of the experiments.

The fluid slosh portion of the experiments were conducted to determine the predictability of nonlinear fluid slosh in 0-g, where surface tension provides the stiffness terms in the governing equations, based upon models and tests in 1-g where gravity provides the displacement-dependent restoring forces. Silicon oil and water, contained in different geometry tanks, were excited at different amplitudes. Both experiments calibrated the ability to predict 0-g nonlinear behavior from 1-g tests and analyses. [Miller, 1992] and [van Schoor, 1993] present the objectives and results of the MODE project.

The first laboratory attribute incorporated in MODE was the separation of test-specific hardware from generic infrastructure. The test-specific hardware consisted of the truss segments and the tanks that contained the silicon oil and water. Since the objectives of the tests required that these elements be tested in different configurations, the generic test equipment was placed in an Electronic Support Module (ESM in Figure E.1 with fluid tank and shaker attached). This included data storage media, the operator interface, power amplifiers used to drive actuators, sensor signal conditioning, power conditioning, and experiment control computers. The ESM was designed to fit into one standard mid-deck locker. The ESM design was versatile enough to not only test different truss and fluid tank geometries, but to also allow testing of other as yet unforeseen test-specific hardware on subsequent Shuttle flights.

Hardware reconfigurability was the second laboratory attribute built into MODE. The truss segments included two four-bay deployable segments, an erectable segment, and a rotary alpha joint with operator-selectable friction (Figure E.2). These segments were attached by the Shuttle crew in different combinations and geometries to allow tests on increasingly complex and nonlinear systems. Furthermore, a total of four fluid slosh tanks allowed two different fluids to be tested in tanks with flat and hemispherical bottoms. The ability of the crew to reconfigure the hardware was instrumental in enabling the test of a

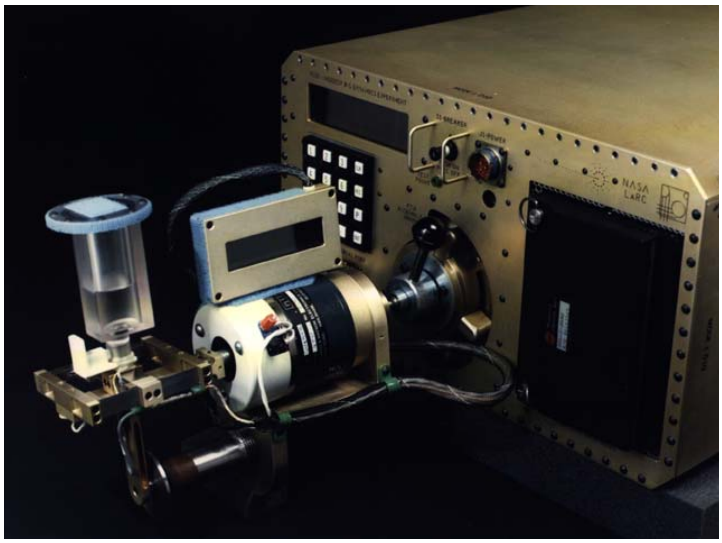


Figure E.1 MODE Experiment Support Module w/ Fluid Test Article

range of test articles that was much wider than can be tested in non-human rated on-orbit facilities. The crew facilitated the changes of test articles and reconfigured the Structural Test Article with Alpha joint without the need for costly automation equipment which would have been required to perfom these tasks remotely. Even with such equipment, it may not have been possible to develop the ESM in the modular fashion it was designed. Instead, the ESM would have been special equipment for a single automated mission.

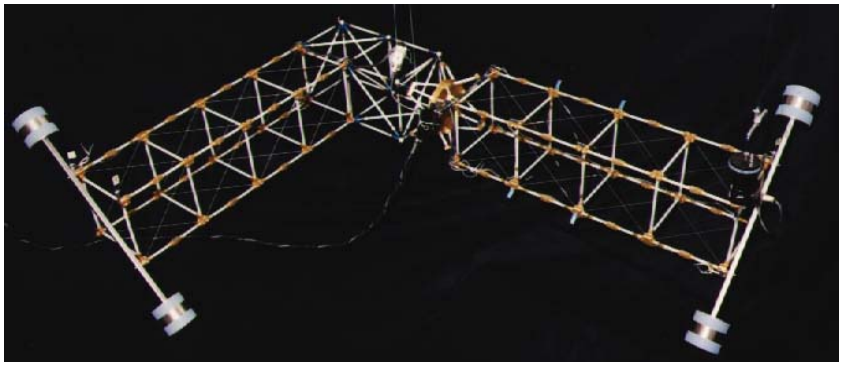


Figure E.2 MODE Structural Test Article with Alpha joint

MODE-re-flight. MODE-re-flight flew on STS-62 in March 1994 to perform additional truss structure tests and initiate the testing of the Dynamic Load Sensors (DLS, pictured in Figure E.3). DLS consisted of a hand hold, foot restraint, and push pad each instrumented to measure the forces and torques that a crew member imparts on the vehicle as they move through the Mid-deck using these devices. This provided a database for quantifying the loads that crew members would impart on the International Space Station (ISS).

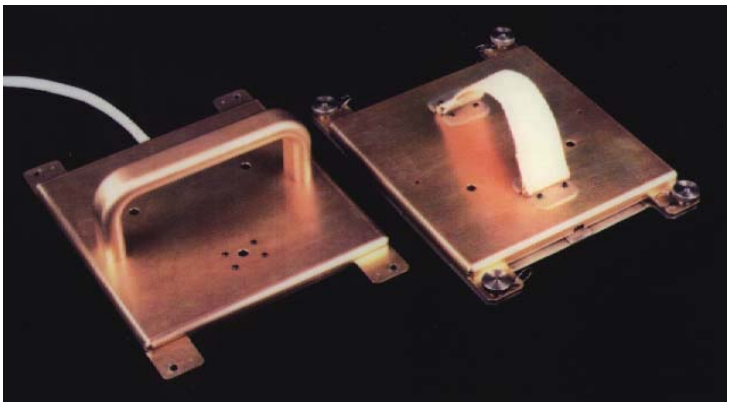


Figure E.3 DLS handhold and foot restraint

The generic versus specific nature of the MODE design was exploited in two ways during re-flight. First, the fact that the MODE hardware flown on the original flight was retrievable allowed it to be refurbished and reflown to support additional tests at a fraction of its original cost. Second, MODE-re-flight saw the introduction of a new series of test articles known as the Dynamic Load Sensors. Since only the DLS test articles, and not the entire test support equipment, needed to be built, the data collected on the DLS test articles was acquired very cost-effectively. The MODE design enabled reusability by containing the generic test equipment in the ESM.

E.2 DLS

The MODE ESM and DLS test articles re-flew on MIR for about three years as a part NASA's ISS Risk Mitigation program. Re-flight allowed a more extensive database to be acquired. This allowed crew motion force and torque statistics to be correlated with crew flight experience as well as with current time on orbit. By testing over weeks to months, rather than days on MODE-re-flight, crew adaptation time constants could be identified. [Amir, 1999], [Amir, 2000], and [Newman, 2001] present the methodology and results obtained over the 40 weeks of DLS operations aboard MIR.

DLS on MIR was the first introduction of extended duration testing in the MODE family of dynamics and controls laboratories. Extended duration testing allows long time constant dynamics to be identified (e.g., crew adaptation), many cycles of the iterative research process to be completed, and new test article configurations to be introduced (e.g., different crew members conducting maneuvers using DLS).

DLS on MIR was the MIT-SSL's first experience with space station operations. Shuttle science operations are typically characterized by the science team traveling to the NASA Johnson Space Center and assisting in the operation of the experiment while residing in one of the Customer Support Rooms. Essentially, the science team needs to travel to Houston and constantly be "on call" for the duration of the flight. This is an expensive process. For DLS on MIR, no such travel was conducted, and the crew and science team were given days to weeks, rather than minutes or hours, to work through test anomalies. This more relaxed operational environment allowed anomalies and their solutions to be analyzed more carefully. Unfortunately, the remote nature of the communication made it difficult for the science team to clearly understand the issues with the tests, particularly when the communication was by email and through several layers of flight operations.

E.3 MACE & MACE-re-flight

MACE. The Mid-deck Active Control Experiment (MACE) flew on STS-67 in March, 1995 to develop dynamics and controls tools for predicting as well as refining robust, multi-variable control algorithms on systems that cannot be realistically tested in 1-g due to gravity couplings. [Grocott, 1994], [Campbell, 1995], [Miller, 1996], and [How, 1997] present the methodologies used to develop and verify controllers during the mission. [Miller, 1998] and [Campbell, 1999] presents the results of the mission. [Miller, 1995] present the high-level design goals of MACE to build upon the design of the MODE family.

Figure E.4 shows the MACE multi-body platform test article being operated on the mid-deck of the Shuttle. The test article consisted of a reaction wheel triax attached to a flexible backbone over a meter and a half in length. A two axis motorized gimbal was attached at both ends of the backbone to perform pointing and slewing maneuvers. MACE consisted of twenty sensors, nine actuators, an ability for the crew to reconfigure its geometry into three different shapes, and a system for connecting the MACE ESM with the Shuttle's Ku-Band communication system.

The Ku-Band Interface System (KIS) gave the ability to downlink test results and uplink new control algorithms. Five times during the mission, the crew downlinked system identification and closed-loop test data, the science team refined dynamic models and control algorithms, and uplinked the new algorithms for test.

MACE continued the design philosophy developed under MODE by separating the test article from the generic laboratory equipment. Again, an ESM (Figure E.5) was built to house the generic equipment in a standard mid-deck locker. This ESM, however, included a high speed real-time computer, an interface for a laptop, and an interface for the Ku-band system. MACE also incorporated hardware reconfigurability in the test article design to maximize the science return.

Figure E.4 MACE operations on shuttle mid-deck

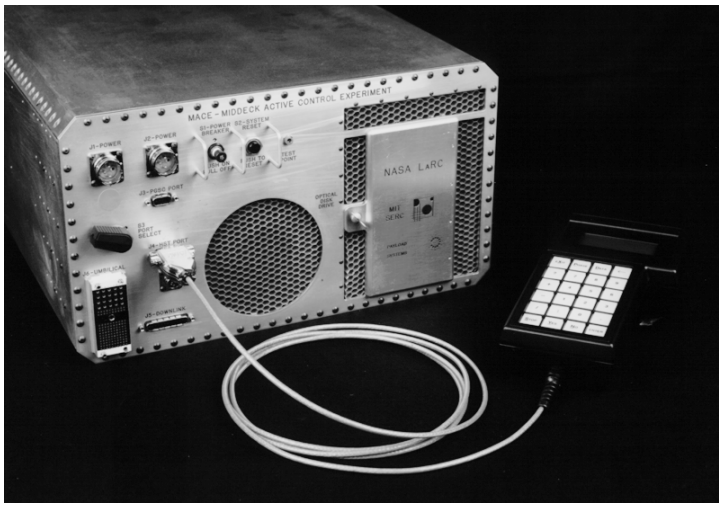


Figure E.5 MACE Experiment Support Module

MACE also introduced a number of important and advanced laboratory attributes. MACE was designed to be risk-tolerant. Unlike dynamics experiments, control experiments have the potential to result in unstable behavior leading to large amplitude motion. The team realized that since the technology being investigated was control and that the desire to explore the limits of control capability dramatically increased the probability that instability would occur, safety assurance through software was not an option. Instead, the MACE team took the approach of ensuring that instability could not break the hardware and that

the crew member was provided with a switch that could immediately cut off power to the test article. These features allowed the boundaries of instability to be explored using a test article that was tolerant of the associated risk. Few if any other on-orbit systems could view unstable behavior as routine operation.

Unlike the MODE laboratory, MACE allowed the science team to change the software during the mission. Software reconfiguration gives the researcher many more degrees of freedom that can be manipulated to alter the conditions of the test. Software that performs dynamic characterization of the test article can be changed in order to investigate unforeseen phenomena in more detail. Control algorithms can be modified based upon models of the unforeseen phenomena. Sensor and actuator groupings can be changed to explore new disturbance and performance topologies. Furthermore, it allows experimental research into methodologies that facilitate on-orbit validation for precision space systems that cannot be accurately tested prior to launch.

The MACE design was the first in the MODE family to exploit human observation and manipulation of the test. Through training and on-orbit documentation, the crew developed the capability to identify, describe, and alter test conditions. In the event that the crew observed unexpected motion of the test article, they described the motion to the science team. A nomenclature was developed to allow the crew to describe the behavior. The crew was asked to estimate whether it exhibited broad or narrow-band frequency content, the frequency range in which the dominant motion occurred, and any directional preference in terms of an agreed upon coordinate system.

Since MACE verbal communications only occurred twice a day and data down-links occurred once a day, the crew then needed the ability to manipulate the test sequence if a clear instability or hardware failure was observed. The training and on-orbit documentation allowed the crew to maximize the number of successful algorithms tested between communication opportunities. To this end, the Flight Data File, used by the crew to guide them through experiment operations, contained families of algorithms. Each family corre-

sponded to a specific geometric axis of motion, sensor-actuator suite, control design approach, and designation as to whether that family was a primary family or a backup. Each family consisted of six to eight algorithms or individual tests. Each succeeding algorithm would have slightly higher control gain.

Backup families would be tested if its corresponding primary family exhibited hardware problems (e.g., failed actuator or sensor). This allowed them to explore the same control features while isolating failed hardware from the control.

When a clear instability was observed during two successive tests within a family, the crew could ignore all remaining higher gain algorithms in that family. This allowed the crew to maximize science productivity.

The last important attribute introduced in MACE was the facilitation of the iterative research process. The science team used the verbal observations communicated by the crew to prepare a plan of attack for the next control redesign phase that would start after the next down-link of data. In addition, they would assist the crew in re-planning their next sequence of tests as well as selecting data files that should be down-linked during the next opportunity. Data down-links occurred once every one to two days. Dynamic characterization data was used to generate models and closed-loop response data was used to alter control parameters. During a single crew sleep period, the science team built models, developed new families of control algorithms, and wrote updates to the Flight Data File. The latter two were up-linked with the morning mail. The existence of the KIS interface for (up)down-link, crew observations and test sequence manipulation, and science team facilities were instrumental in shortening the research "question & answer" cycle time to one day.

One day cycle time was a little too short. The team basically had enough time to regurgitate algorithm refinements in response to features observed by the crew and in the data. Allowing several days to weeks to digest the downlinked test results and develop updates to algorithms would allow a more methodical approach to posing new research questions

and putting test plans in place to address those questions. Clearly, this requires the laboratory to reside on orbit for longer periods of time.

MACE-re-flight. On MACE's second flight, it was launched to the International Space Station in September 2000. Operations started in December of that year with the arrival of the Expeditionary One Crew making MACE-re-flight the first crew-interactive experiment on ISS. Figure E.6 shows operations of the MACE article inside the ISS US Node. MACE-re-flight was a collaborative effort between the Air Force Research Laboratory (AFRL) and the MIT-SSL. Each brought with them a team of researchers studying dynamics and controls technologies ranging from neural networks to nonlinear dynamic characterization to adaptive reaction wheel isolation. Modifications for re-flight included an upgraded real-time control computer and operating system, upgraded data storage media, and meter long flexible appendages attached to the gimbal faces giving the test article an appearance of a free-floating, multi-link robot. MACE-re-flight returned to the ground in August 2001. [Blaurock, 1999] presents the modeling of MACE-re-flight prior to deployment, while [Yung, 2001] uses the results of the mission.

The laboratory attribute first featured in MACE-re-flight was opening the laboratory to multiple investigators. A number of government research agencies, industries, and academic institutions participated in MACE-re-flight. Unfortunately, due to the early stage of ISS assembly, data downlink and crew time were limited. This severely limited the iterative research process.

Figure E.6 MACE operations aboard the ISS

Appendix F

SPHERES AVIONICS DESIGN

This appendix presents detailed descriptions of the SPHERES avionics. The SPHERES laboratory avionics sub-systems implements electronics for the satellites, communications with the control computer, metrology beacons, and a beacon tester. This appendix also presents the design of several expansion port items already in use. Each section presents the functional block diagram and the complete schematics (when applicable) for all the electronic components of the SPHERES laboratory:

- SPHERES nano-satellites
  - Power & control panel
  - Data processing unit (C6701 DSP / SMT375)
  - Metrology
  - Communications
  - Propulsion
  - Expansion Port
  - Internal beacon
- Laptop communications
- Metrology Beacons
- Metrology Beacon Tester
- Expansion Port Items

F.1 SPHERES nano-satellites

The electronics of the SPHERES nano-satellites, shown in Figure F.1, are implemented in two primary "electronics stacks", with several peripheral electronic boards. The groupings are as follows (third level bullets indicate peripheral electronic boards which support the primary board of that stack):

- First stack (power & propulsion)
  - Power
    - Batteries (2x)
    - Power Switch
    - Circuit Breaker
    - Control Panel
  - Propulsion
    - Propulsion LEDs
- Second stack (data processing)
  - SMT375 (C6701 DSP)
  - Metrology (Motherboard 1)
    - Metrology 6 (ultrasound / IR)
    - Accelerometer Boards
    - On-board beacon
  - Communications (Mother board 2)
    - DR200x wireless boards (2x)
    - Expansion Port

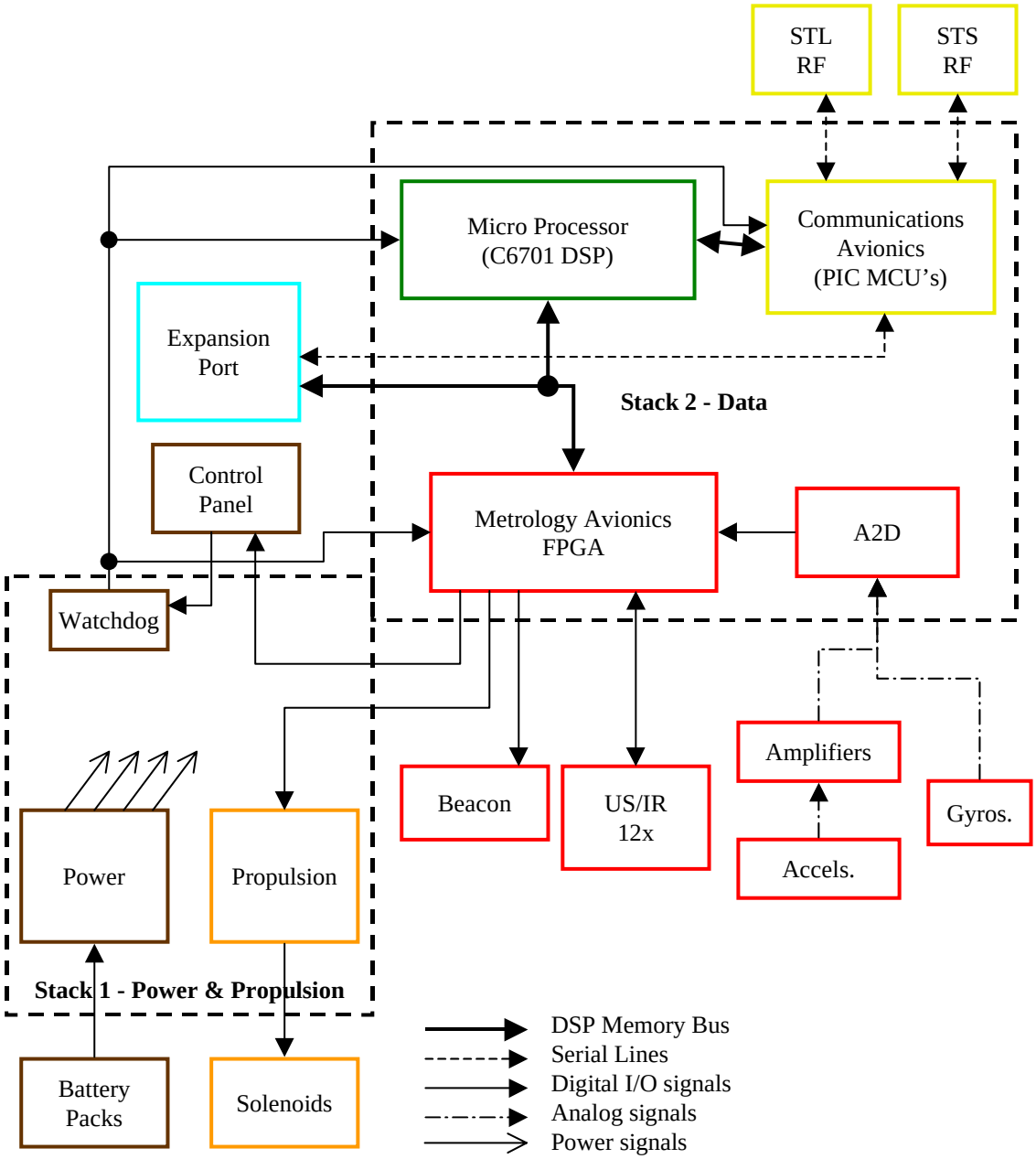


Figure F.1 SPHERES avionics overview

F.1.1 Power & Control Panel

Design Drivers

- Provide the necessary power and voltages for all sub-systems
  - 3.3 V: DSP, Metrology, Communications
  - 5 V: DSP, Metrology, Communications, Propulsion
  - ±15 V: Metrology (gyros and accelerometers)
  - 22 V: Propulsion
- Meet applicable ISS safety guidelines
- Maximize battery utilization

Functional Block Diagram

The power sub-system is comprised of three main type of electronic boards: battery packs, control panel, and the power regulation board. The circuit breaker and power switch are wired independently. The functional description, inputs, and outputs of each component are presented below.

Battery packs

There are two types of battery packs: flight and rechargeable. The functions of the two types are:

- **Flight:** Provides up to two hours of operations to the SPHERES nano-satellites through 8 AA alkaline batteries. It also provides diode and fuse protection to meet NASA Safety requirements (triple redundancy).
- **Rechargeable:** Provides up to two hours of operations to the SPHERES nano-satellites through 8AA NiMH rechargeable batteries. The battery charging circuit resides within the packs themselves, requiring only a 15Vdc external supply. The external supply can have an optional LED to indicate charging status. The board provides the same diode and fuse protection as the flight packs.

Its inputs and outputs are listed in Table F.1

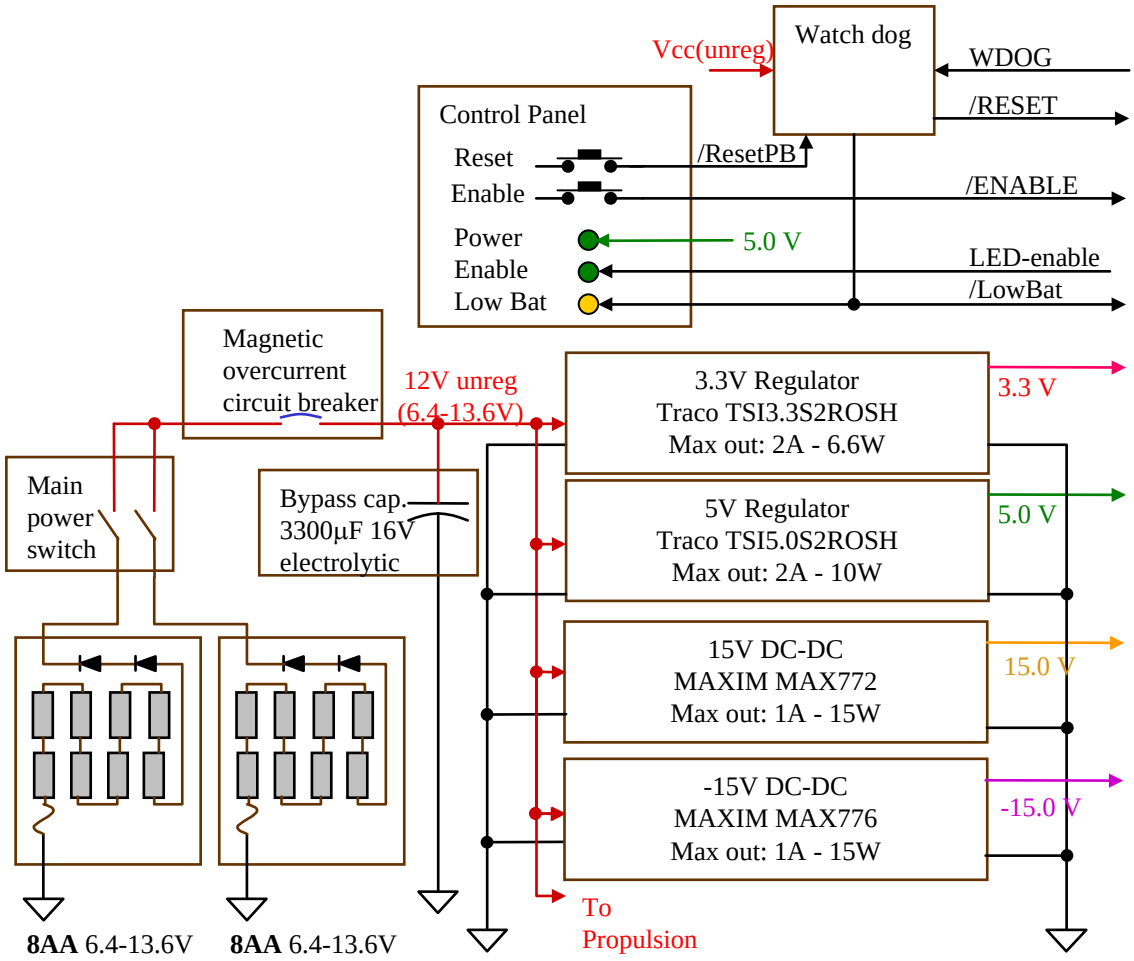


Figure F.2 Power sub-system functional block diagram

Power Switch

The power switch is a two phase mechanical switch. The switch is two phase so that the battery positive power connections are isolated when the switch is open (the SPHERES are off), preventing any current from flowing between the battery packs. While not necessary for NASA safety requirements, it allows the battery packs to remain inserted in the satellites without risk. Its inputs and outputs are listed in Table F.2

TABLE F.1 Battery packs signals description

Signal	Type	Description
Flight		
Vin	In	Unregulated input voltage from 8AA batteries (6.4-13.6V)
Vout	Out	Protected, unregulated voltage (5.8-13.0V due to 0.6V drop through diodes)
GND	Pwr	Common reference ground
Rechargeable		
Vin	In	Unregulated input voltage from 8AA batteries (6.4-13.6V)
Vout	Out	Protected, unregulated voltage (5.8-13.0V due to 0.6V drop through diodes)
Vsupply	In	15V input voltage for recharging circuit
FT	Out	Signal to external LED which indicates charging in process (blinking) or done (solid on)
GND	Pwr	Common reference ground

TABLE F.2 Power Switch signals description

Signal	Type	Description
		Two phase power switch which isolates the battery packs when turned off
Vin_1, Vin_2	In	Protected, unregulated voltage from battery packs
$Vout_1$, $Vout_2$	Out	Switched, unregulated voltage to circuit breaker

Circuit Breaker

The magnetic circuit breaker provides 5A current protection. A thermistor is connected in series with the circuit breaker to prevent power-surges larger than 5A when the satellites are turned on and the large bypass capacitor (3300μF) charges. Once heated the thermistor has a resistance of approximately 0.1Ω. The inputs and outputs of this board are listed in Table F.3

TABLE F.3 Circuit breaker signals description

Signal	Type	Description
Vin_1, Vin_2	In	Switched, unregulated voltage from power switch
Vout	Out	Switched and protected, unregulated voltage to power board

Control Panel

The control panel is the primary manual interface of the satellites. The panel mechanically holds the power switch, although it is not connected electrically. The panel electronics only include digital I/O lines powered through the regulated 5V supply. The elements in the panel are:

- Reset button - creates a negative logic signal which connects directly to the watchdog module, which in turn generates a correctly timed reset signal for the rest of the electronics.
- Enable button - creates a negative logic signal which is sent directly to the SPHERES FPGA as a general I/O signal; the SPHERES Core Software checks the state of this button to enable operations (go from "idle" to "ready" or "running" mode).
- Power LED - driven directly off the regulated 5V supply indicates when the power is on; since it is driven directly off the 5V supply, it is only on when the supply operates correctly, giving a reasonable indication that the power regulation module is operating correctly.
- Low Battery LED - the watchdog measures the unregulated battery voltage and indicates a low battery condition when there are approximately 20 minutes remaining of operation.
- Enabled LED - the LED is driven directly off the SPHERES FPGA as a general I/O signal; it is turned on by the SPHERES Core Software when the satellites are in a "ready" or "running".

The inputs and outputs of this board are listed in Table F.4

Power Regulation Board

The power regulation board is the most complex board of the power sub-system. It provides power regulation for the data stack and all other avionics[1], contains the watchdog, and serves as a bypass for the propulsion signals. The board outputs four voltages to the

TABLE F.4 Control panel signals description

Signal	Type	Description
Vcc	Pwr	Input +5V dc
GND	Pwr	Common reference ground
LED-enable	In	Enable LED control signal
LED-lowbat	In	Low battery LED indicator control signal
/Reset	Out	Reset signal to power board watchdog module
/Enable	Out	Enable signal for SPHERES FPGA

second stack: +3.3V, +5V, +15V, and -15V. The +3.3V and +5V signals are used throughout the system to power electronic components. The ±15V powers the accelerometers and gyroscopes. The power board is the mechanical attachment point for two of the gyroscopes, although no electrical signals from the gyroscopes pass through the board. Table F.5 lists the inputs and outputs of this board (or refers to other tables as applicable).

TABLE F.5 Power regulation board signals description

Section	Signal	Type	Description
Power to second electronic stack	Vcc(+5V)	Pwr	+5V output
	Vcc(+3.3V)	Pwr	+3.3V output
	Vcc(+15V)	Pwr	+15V output
	Vcc(-15V)	Pwr	-15V output
	GND	Pwr	Common ground
Data connector to second electronics stack	THR 1-12	In	Pass through signals for thrusters 1-12
	/Enable	Out	Enable button pass through signal
	/LED-enable	In	Enable LED pass through signal
	/Batlow	Out	Low battery output to DSP
	WDOG	In	Watchdog control signal from DSP
	/RESET	Out	Reset control line to second electronics stack

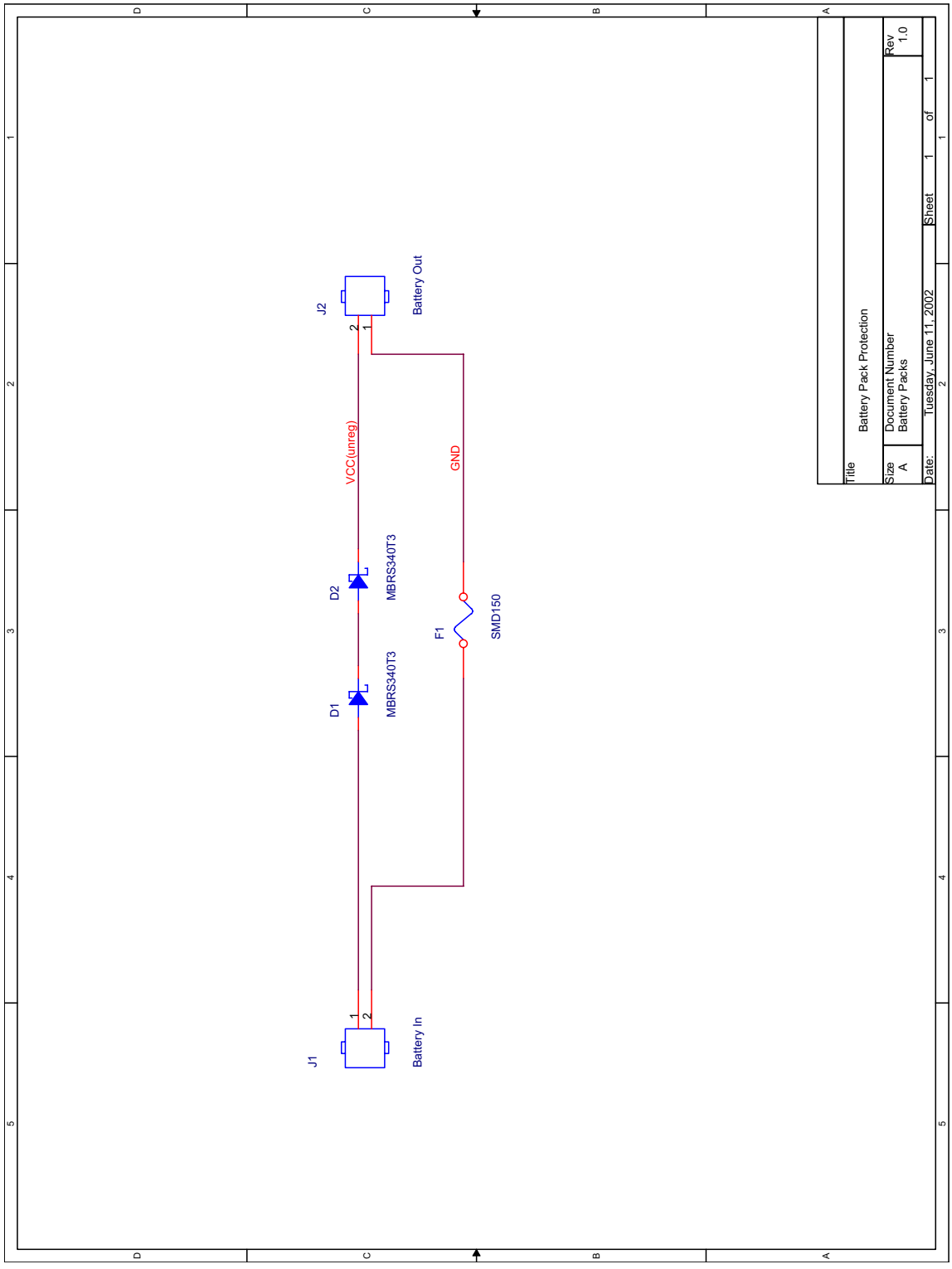
1. The propulsion sub-system increases the unregulated voltage to 20V; the power board does not provide the higher voltage required by the propulsion circuit.

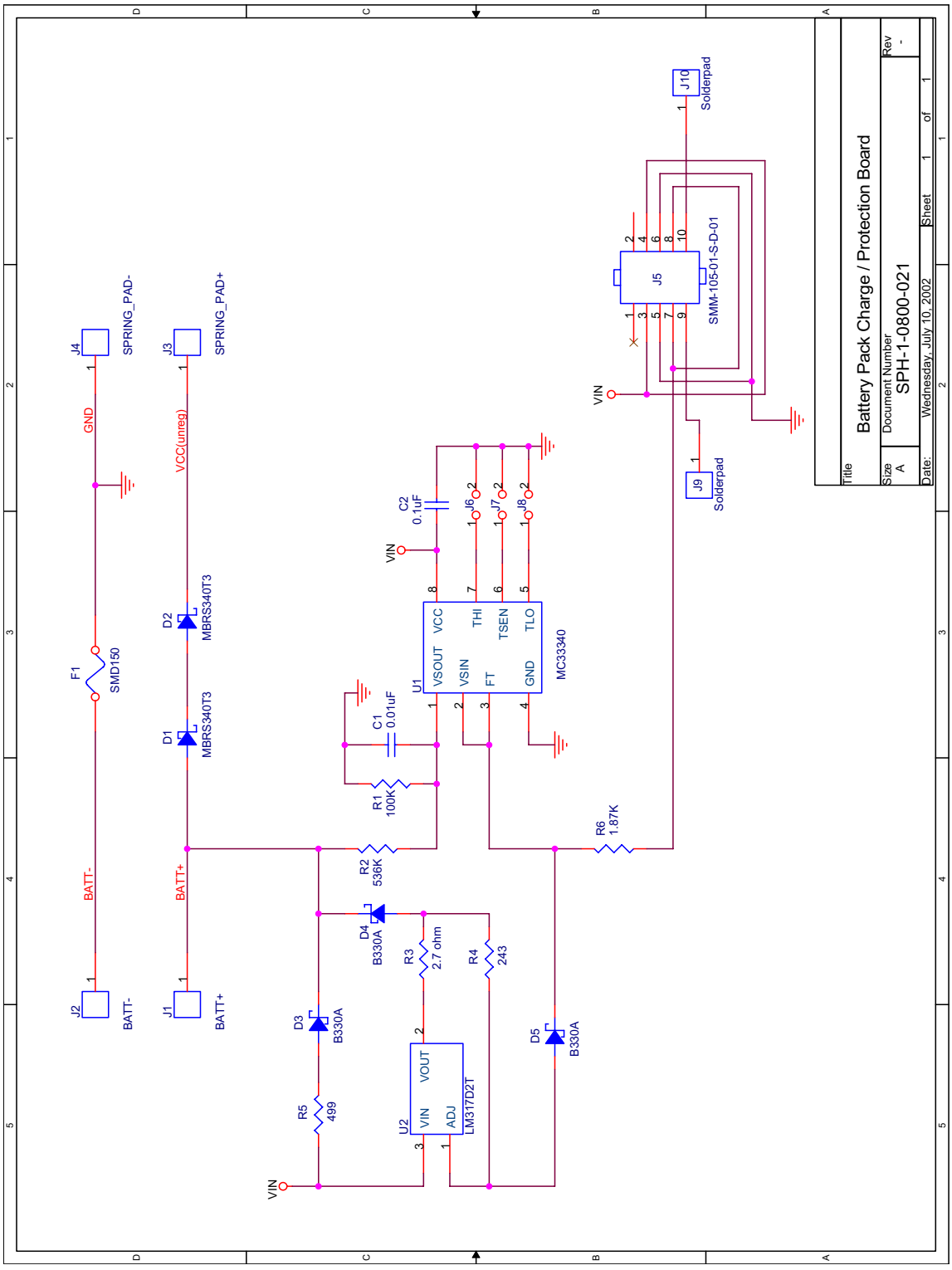
TABLE F.5 Power regulation board signals description

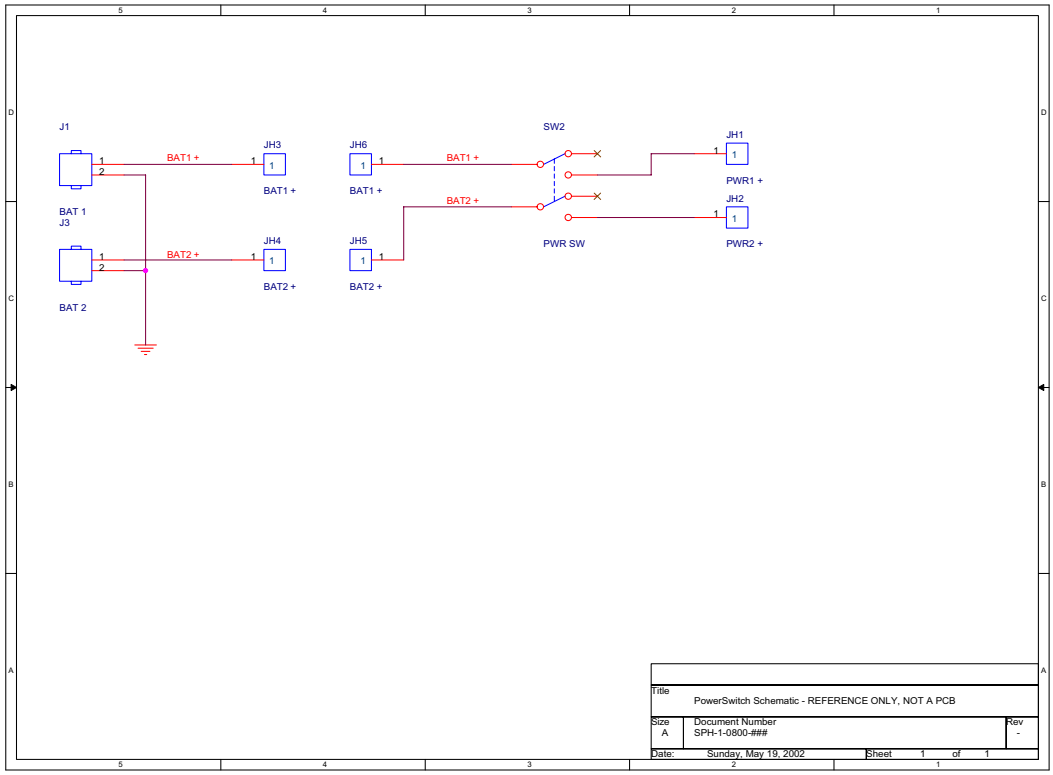
Section	Signal	Type	Description
Data and power to propulsion board	THR 1-12	Out	Pass through signals for thrusters 1-12
	Vcc(5V)	Pwr	5V power for propulsion board
	Vcc(unreg)	Pwr	Switched, protected, unregulated voltage for propulsion board
	GND	Pwr	Common ground
Data to/ from control panel	See Table F.4		

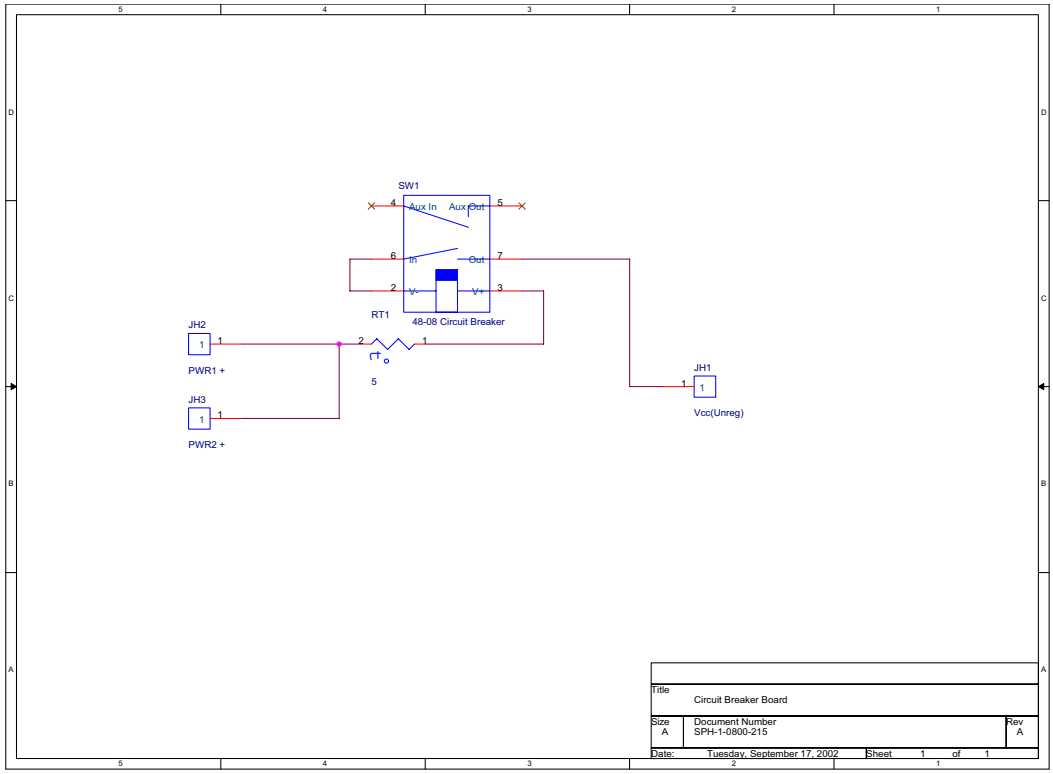
Schematics

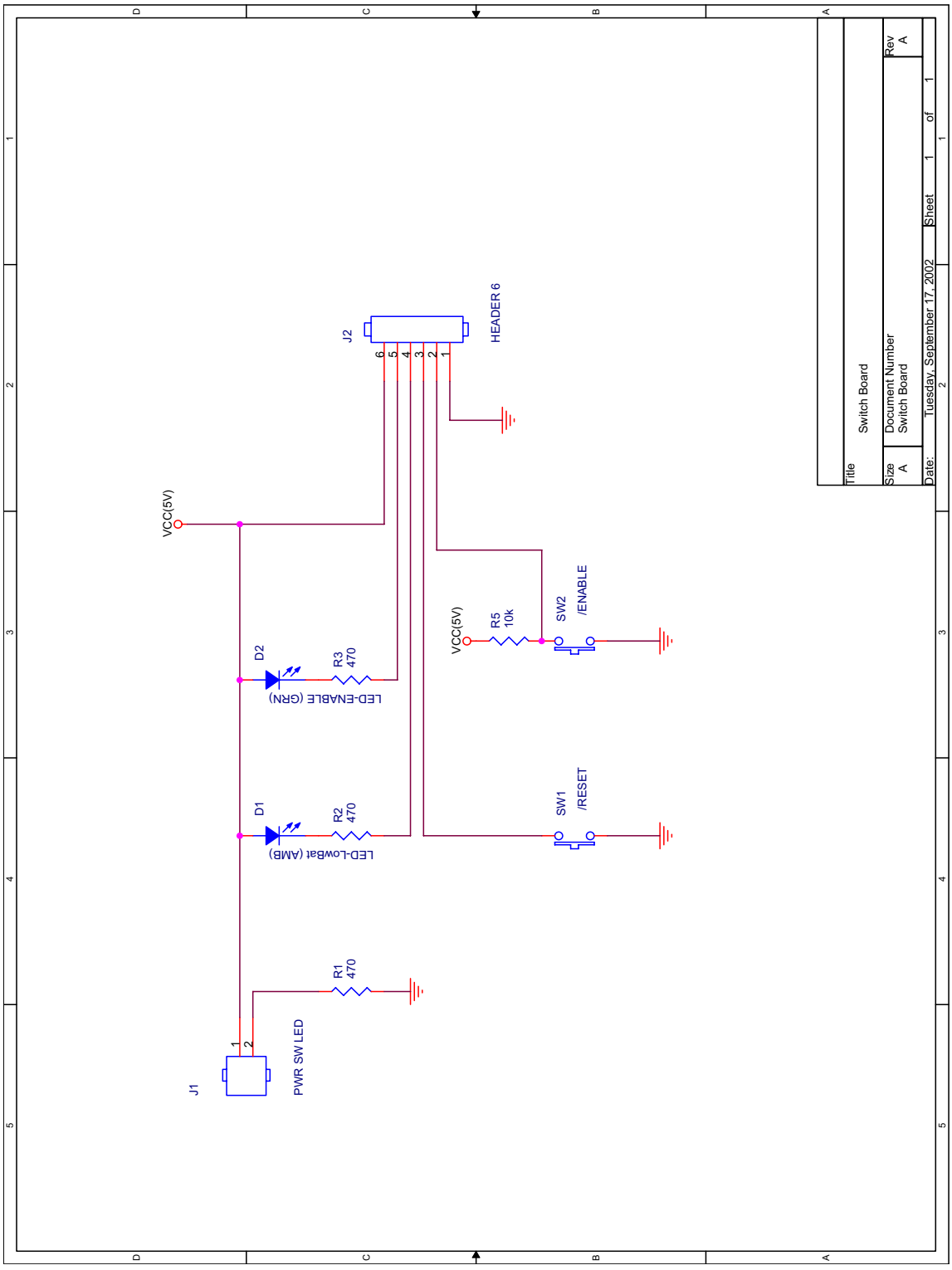
The schematics of the power sub-systems follow.

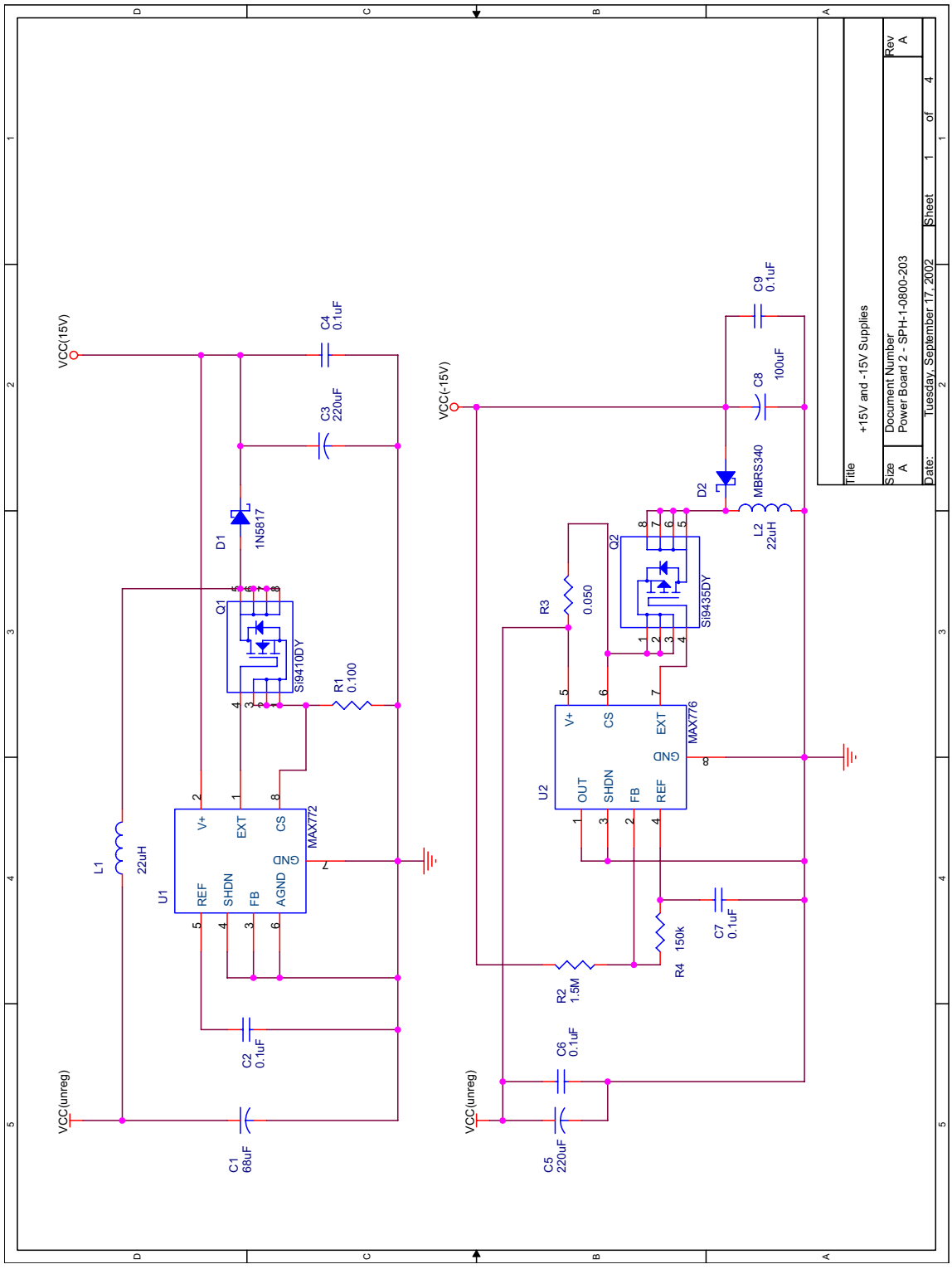

J2
Battery Out
VCC(unreg)
GND
D2
MBRS340T3
F1
SMD150
D1
MBRS340T3
J1
Battery In
Title
Battery Pack Protection
Size
A
Document Number
Battery Packs
Rev
1.0
Date:
Tuesday, June 11, 2002
Sheet
1
of
1

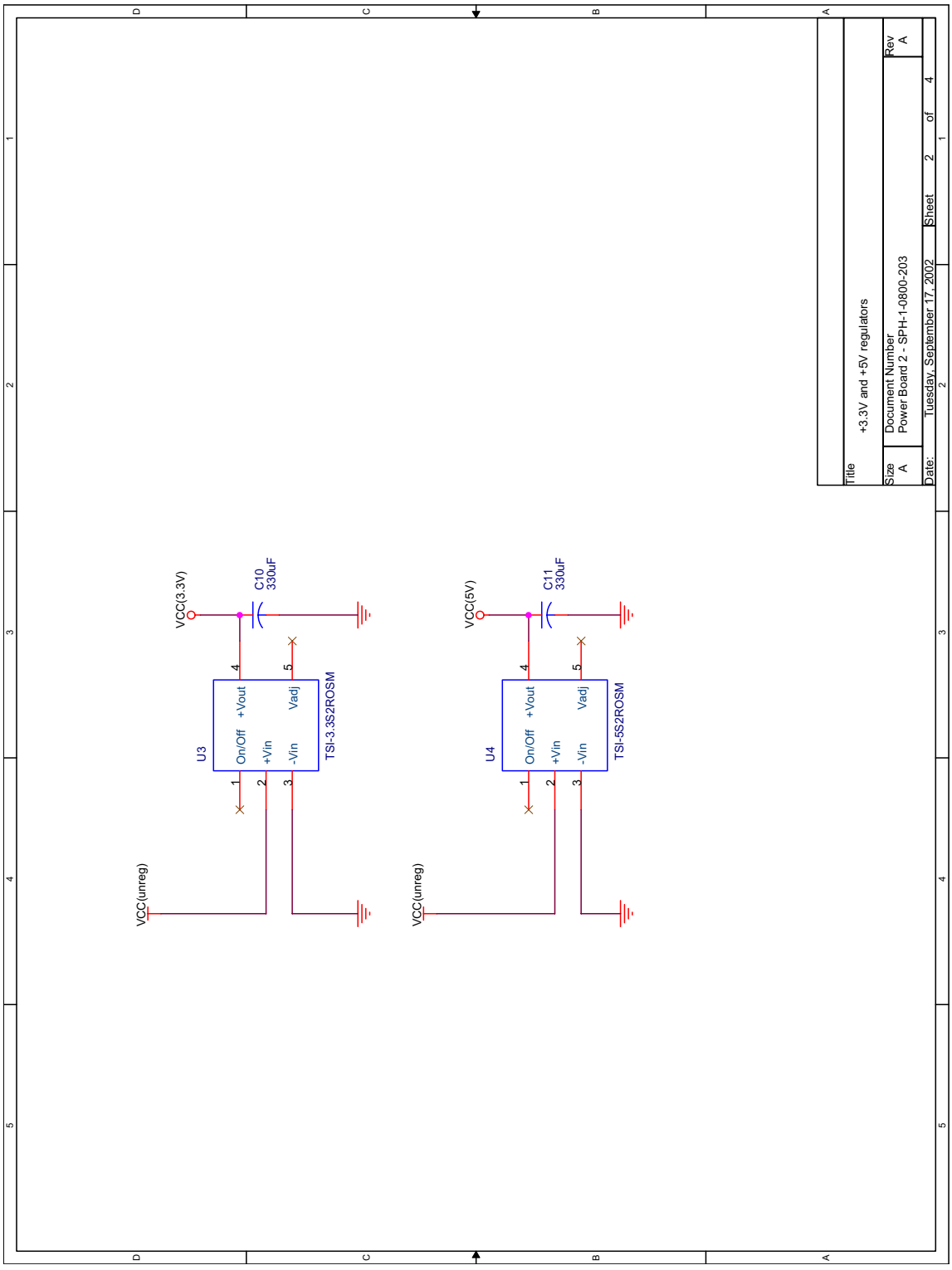

Battery Pack Charge / Protection Board
Document Number
SPH-1-0800-021
Rev
-
Size
A
Date:
Wednesday, July 10, 2002
Sheet
1
of
1
Title
J1
BATT+
J2
BATT-
J3
SPRING_PAD+
J4
SPRING_PAD-
J5
SMM-105-01-S-D-01
J6
J7
J8
J9
Solderpad
J10
Solderpad
F1
SMD150
D1
MBRS340T3
D2
MBRS340T3
D3
B330A
D4
B330A
D5
B330A
U1
MC33340
VCC
THI
TSEN
TLO
VSOUT
VSIN
FT
GND
U2
LM317D2T
VIN
VOUT
ADJ
R1
100K
R2
536K
R3
2.7 ohm
R4
243
R5
499
R6
1.87K
C1
0.01uF
C2
0.1uF
GND
VCC(unreg)
BATT-
BATT+
VIN

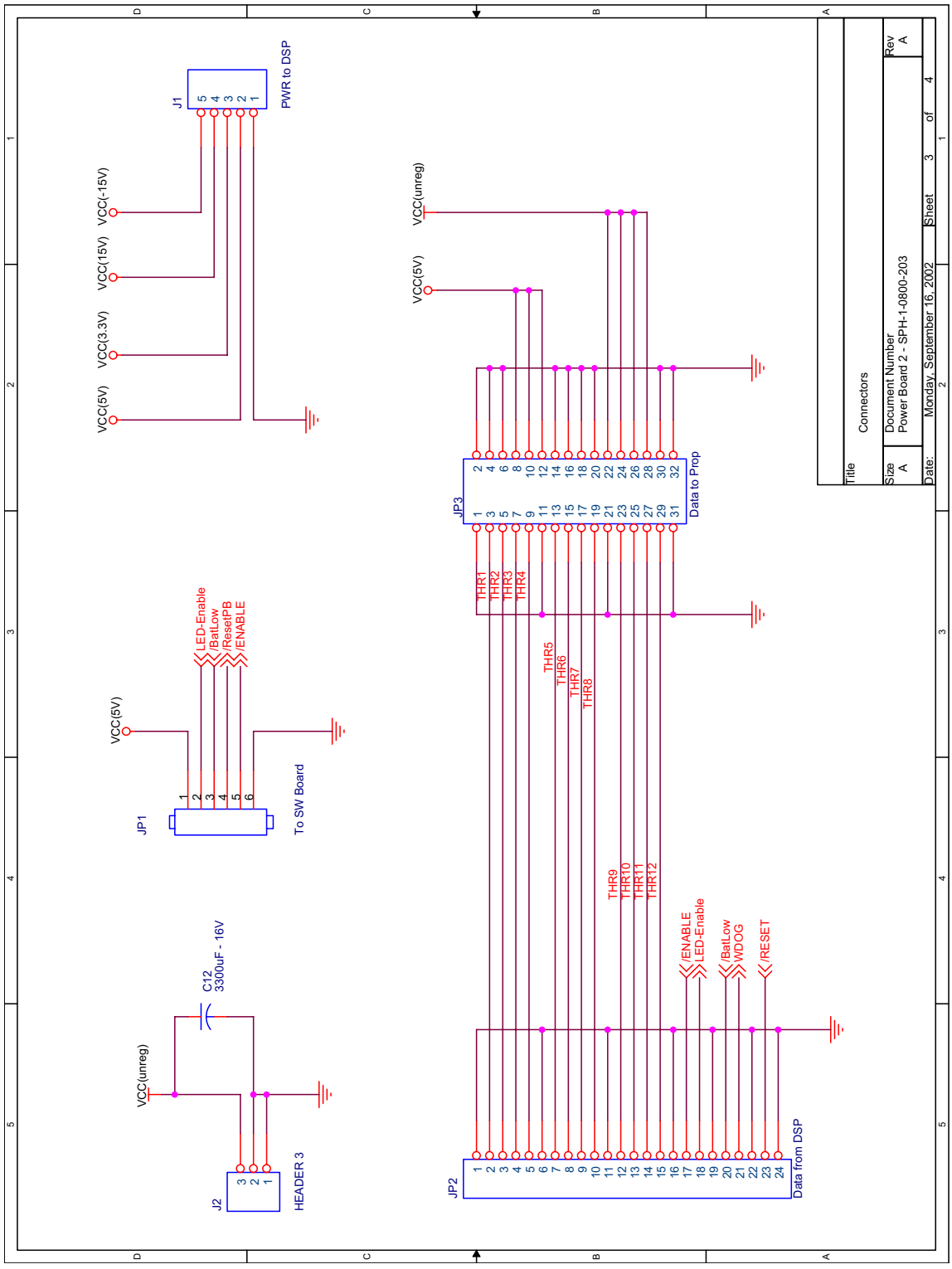

J1
BAT 1
J3
BAT 2
BAT1 +
BAT2 +
JH3
JH4
JH6
JH5
SW2
PWR SW
JH1
JH2
PWR1 +
PWR2 +
Title
PowerSwitch Schematic - REFERENCE ONLY, NOT A PCB
Size
A
Document Number
SPH-1-0800-###
Rev
-
Date: Sunday, May 19, 2002
Sheet 1 of 1

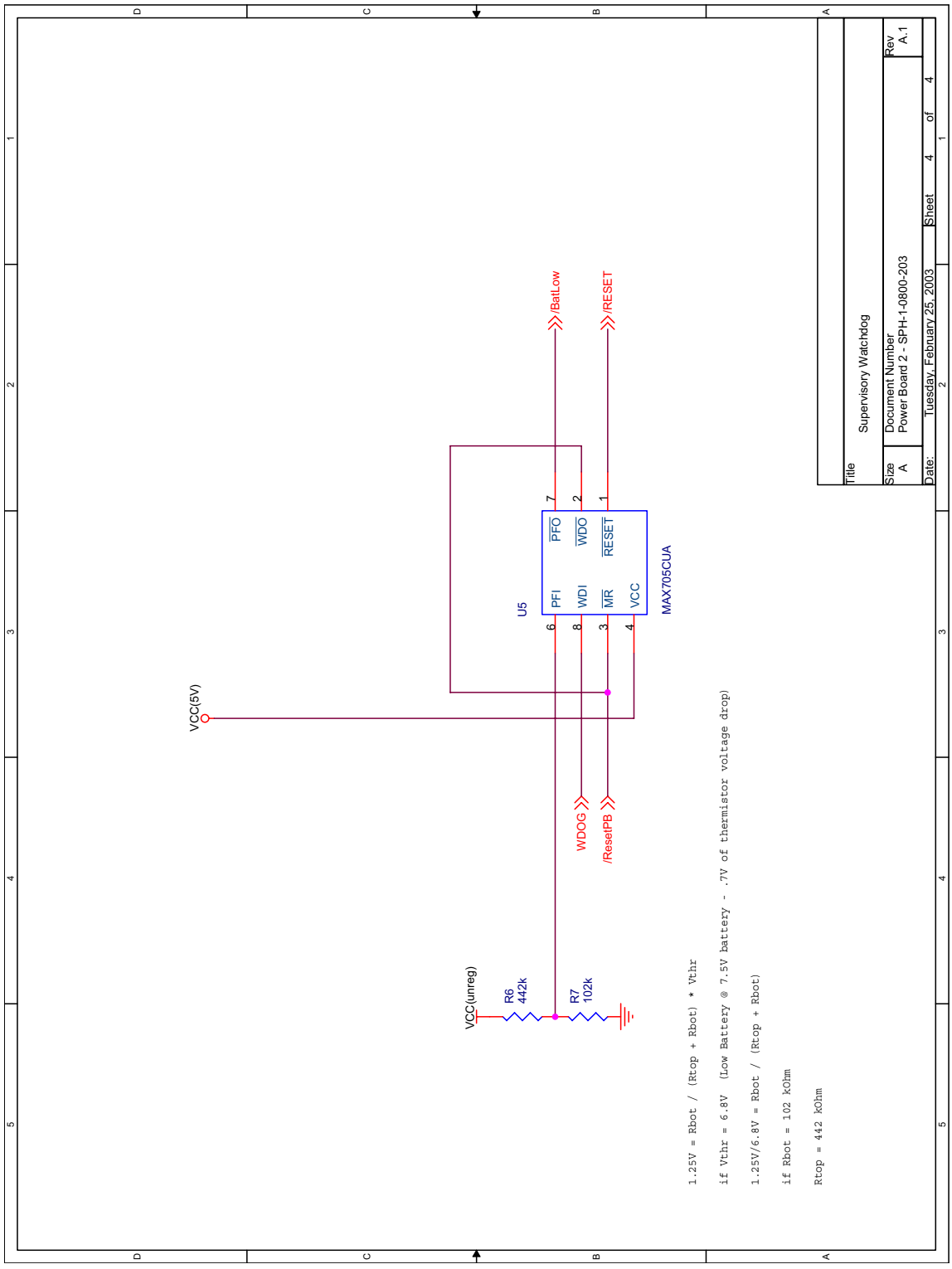

SW1
Aux In
Aux Out
In
Out
V-
V+
RT1
48-08 Circuit Breaker
JH2
PWR1 +
JH3
PWR2 +
JH1
Vcc(Unreg)
Title
Circuit Breaker Board
Size
A
Document Number
SPH-1-0800-215
Rev
A
Date: Tuesday, September 17, 2002
Sheet 1 of 1


Title
Switch Board
Size
A
Document Number
Switch Board
Rev
A
Date:
Tuesday, September 17, 2002
Sheet
1
of
1
J2
HEADER 6
VCC(5V)
R5
10k
SW2
/ENABLE
D2
R3
470
LED-ENABLE (GRN)
D1
R2
470
LED-LowBat (AMB)
SW1
/RESET
R1
470
J1
PWR SW LED


VCC(15V)
C4 0.1uF
C3 220uF
D1 1N5817
Q1 Si9410DY
R1 0.100
U1 MAX772
V+ EXT CS GND REF SHDN FB AGND
L1 22uH
C2 0.1uF
C1 68uF
VCC(unreg)
VCC(-15V)
C9 0.1uF
C8 100uF
D2 MBRS340
L2 22uH
Q2 Si9435DY
R3 0.050
U2 MAX776
V+ CS EXT GND OUT SHDN FB REF
R4 150k
C7 0.1uF
R2 1.5M
C6 0.1uF
C5 220uF
Title +15V and -15V Supplies
Size A
Document Number Power Board 2 - SPH-1-0800-203
Rev A
Date: Tuesday, September 17, 2002
Sheet 1 of 4


Title
+3.3V and +5V regulators
Size
A
Document Number
Power Board 2 - SPH-1-0800-203
Rev
A
Date:
Tuesday, September 17, 2002
Sheet
2
of
4
VCC(3.3V)
C10
330uF
U3
On/Off
+Vout
Vadj
+Vin
-Vin
TSI-3.3S2ROSM
VCC(unreg)
VCC(5V)
C11
330uF
U4
On/Off
+Vout
Vadj
+Vin
-Vin
TSI-5S2ROSM
VCC(unreg)


J1
PWR to DSP
VCC(-15V)
VCC(15V)
VCC(3.3V)
VCC(5V)
VCC(unreg)
VCC(5V)
JP3
Data to Prop
THR1
THR2
THR3
THR4
THR5
THR6
THR7
THR8
THR9
THR10
THR11
THR12
/ENABLE
LED-Enable
/BatLow
WDOG
/RESET
JP2
Data from DSP
LED-Enable
/BatLow
/ResetPB
/ENABLE
JP1
To SW Board
C12
3300uF - 16V
VCC(unreg)
J2
HEADER 3
Title
Connectors
Size
A
Document Number
Power Board 2 - SPH-1-0800-203
Rev
A
Date:
Monday, September 16, 2002
Sheet
3
of
4


VCC(5V)
U5
MAX705CUA
PFI
WDI
MR
VCC
PFO
WDO
RESET
/BatLow
/RESET
WDOG
/ResetPB
VCC(unreg)
R6
442k
R7
102k
1.25V = Rbot / (Rtop + Rbot) * Vthr
if Vthr = 6.8V (Low Battery @ 7.5V battery - .7V of thermistor voltage drop)
1.25V/6.8V = Rbot / (Rtop + Rbot)
if Rbot = 102 kOhm
Rtop = 442 kOhm
Title
Supervisory Watchdog
Size
A
Document Number
Power Board 2 - SPH-1-0800-203
Rev
A.1
Date:
Tuesday, February 25, 2003
Sheet
4
of
4

F.1.2 Propulsion

Design Drivers

- Create a *spike and hold* signal for the solenoids as presented in Figure F.3
  - Spike at +22V
  - Spike hold time: 7ms
  - Hold at +5V
  - Minimum impulse bit: 5ms
  - Maximum impulse bit: infinite
- Provide visual indication of solenoid states (open or closed)

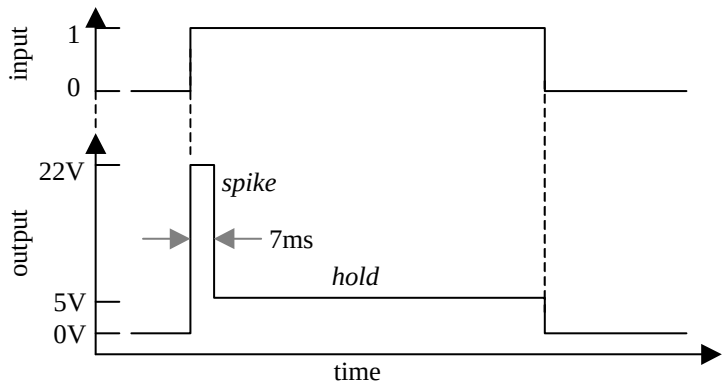


Figure F.3 Propulsion *spike and hold* timing diagram

Functional Block Diagram

Figure F.4 presents the functional block diagram of the propulsion system. A propulsion board receives the propulsion inputs from the data processing stack (pass through in the power board) and creates a spike and hold signal independently for each solenoid. The driver signals are sent to the solenoid via shielded wire to reduce the EMI effects of the spike on other circuits. A board with an LED attaches to the front face of the nozzle of each solenoid to indicate its state (off = closed, on = open). The propulsion board creates the +22V supply needed for the spike from the unregulated voltage input.

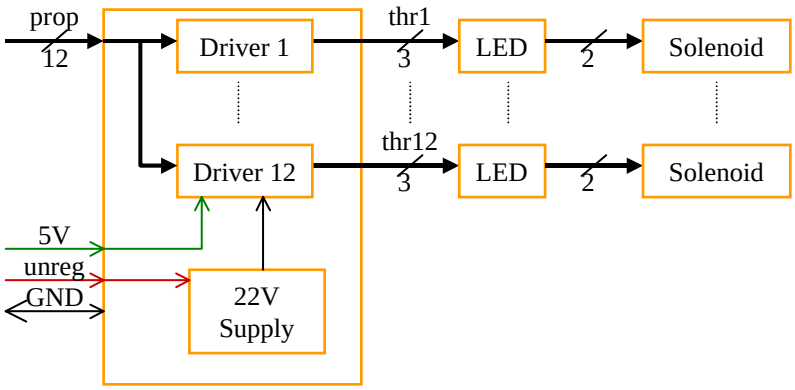


Figure F.4 Propulsion avionics functional block diagram

Propulsion Board

The propulsion board creates the spike and hold signal necessary to operate the solenoids. The board utilizes a Maxim MAX668 step-up switching regulator to create 22V from the variable 8-13V unregulated input. Each spike and hold circuit uses the schematic presented in Figure F.5 (with the necessary current limiting resistors and reverse voltage protection diodes). The inputs and outputs of this board are described in Table F.6.

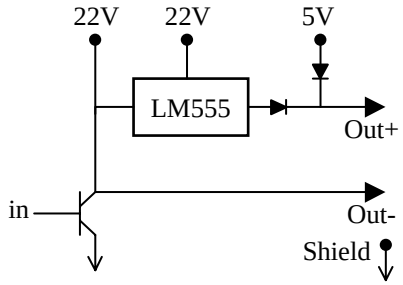


Figure F.5 Propulsion *spike and hold* circuit

Solenoid Board

The solenoid boards provide visual indication of the state of their corresponding solenoid. Their small size allows them to be mounted directly on top of the connector side of each nozzle, ensuring immediate correlation between an LED and a solenoid. The inputs and outputs of this board are listed in Table F.7.

TABLE F.6 Propulsion board signals description

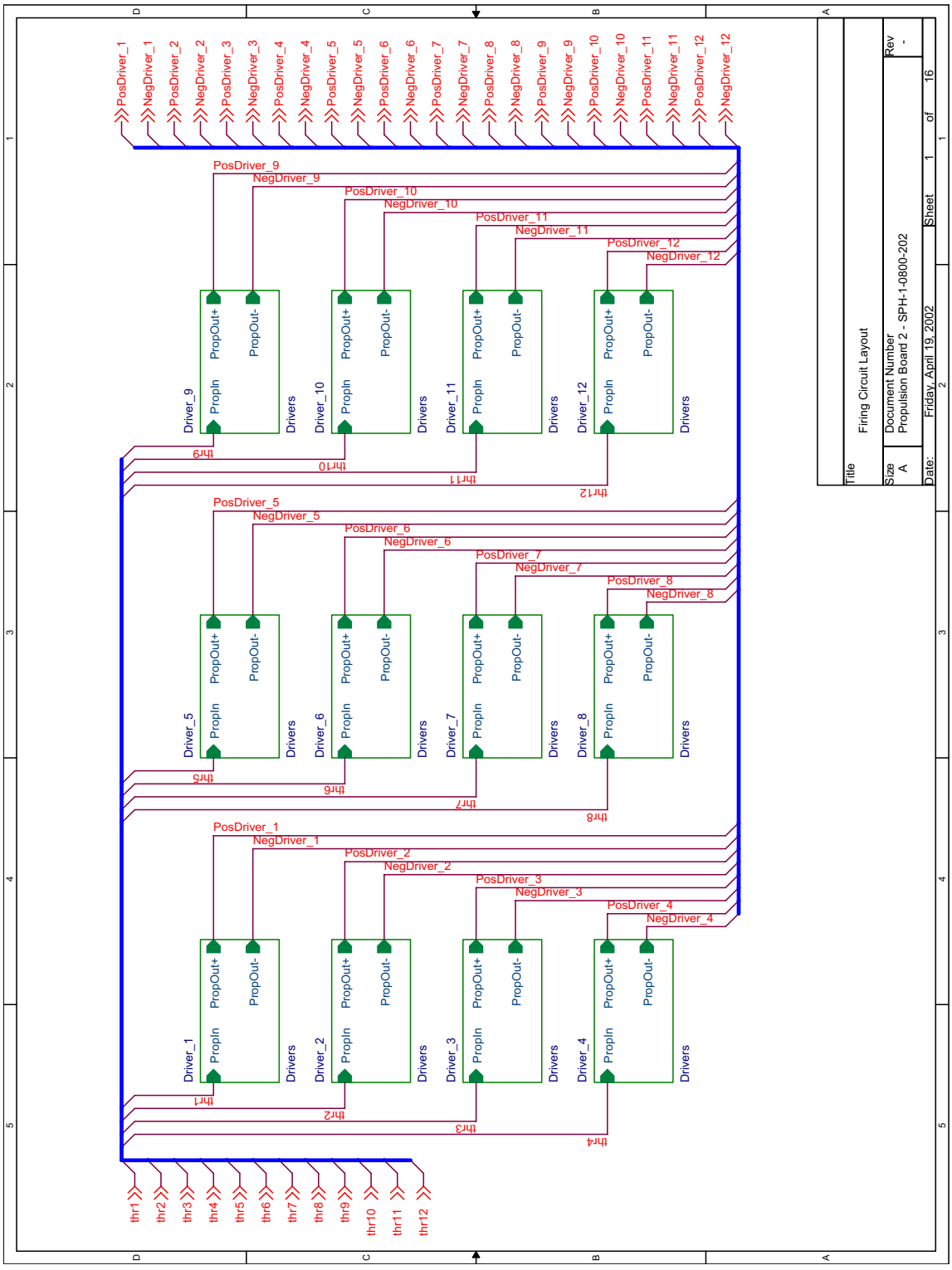
Signal	Type	Description
GND	Pwr	Common ground
Vcc(5V)	Pwr	+5V supply
Vcc(unreg)	Pwr	Unregulated (8-13V) power
Prop[1-12]	In	Command signals from the data processing stack
PosDriver[1-12]	Out	Positive terminal for each solenoid
NegDriver[1-12]	Out	Negative terminal for each solenoid

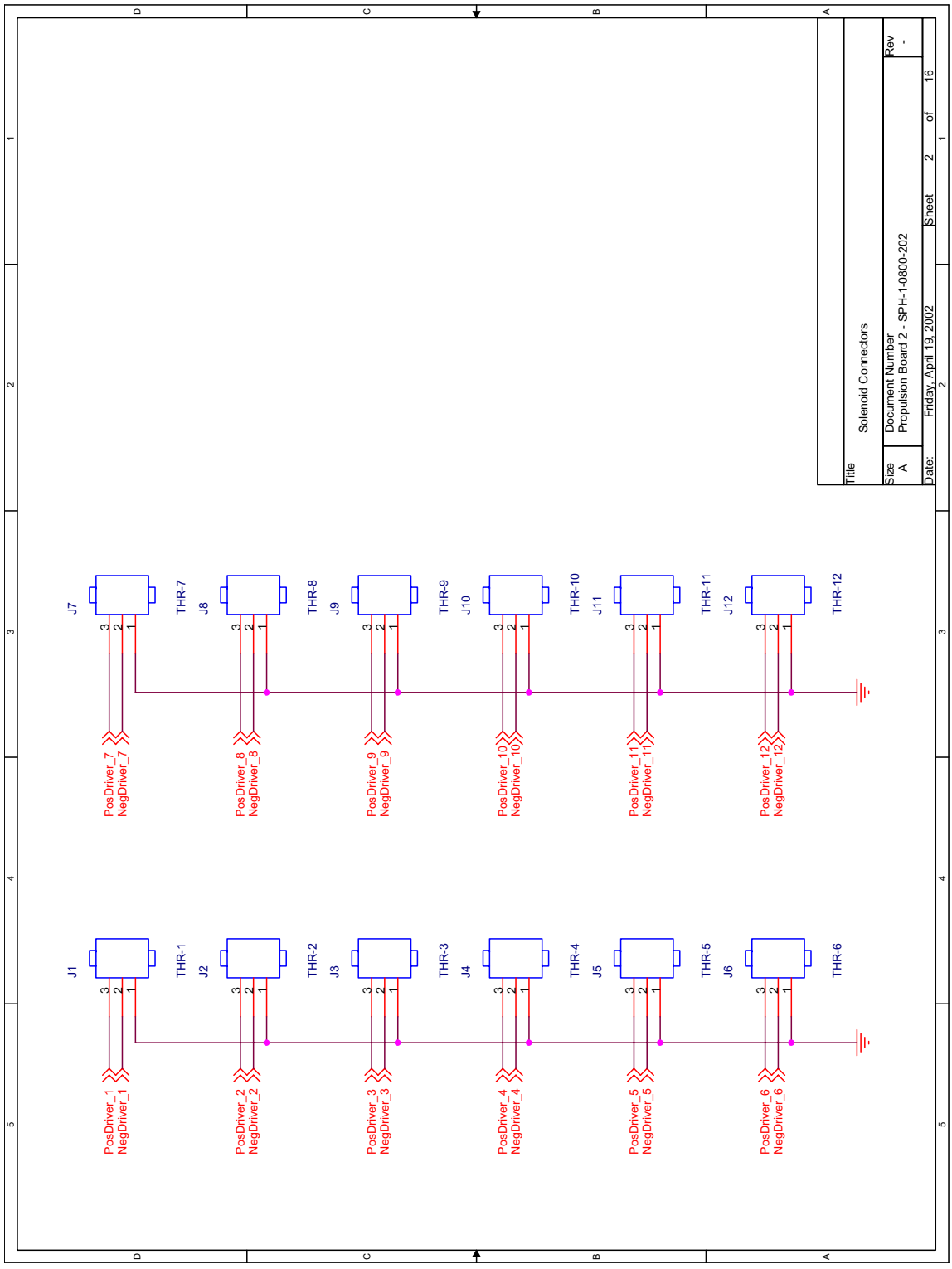
TABLE F.7 Solenoid board signals description

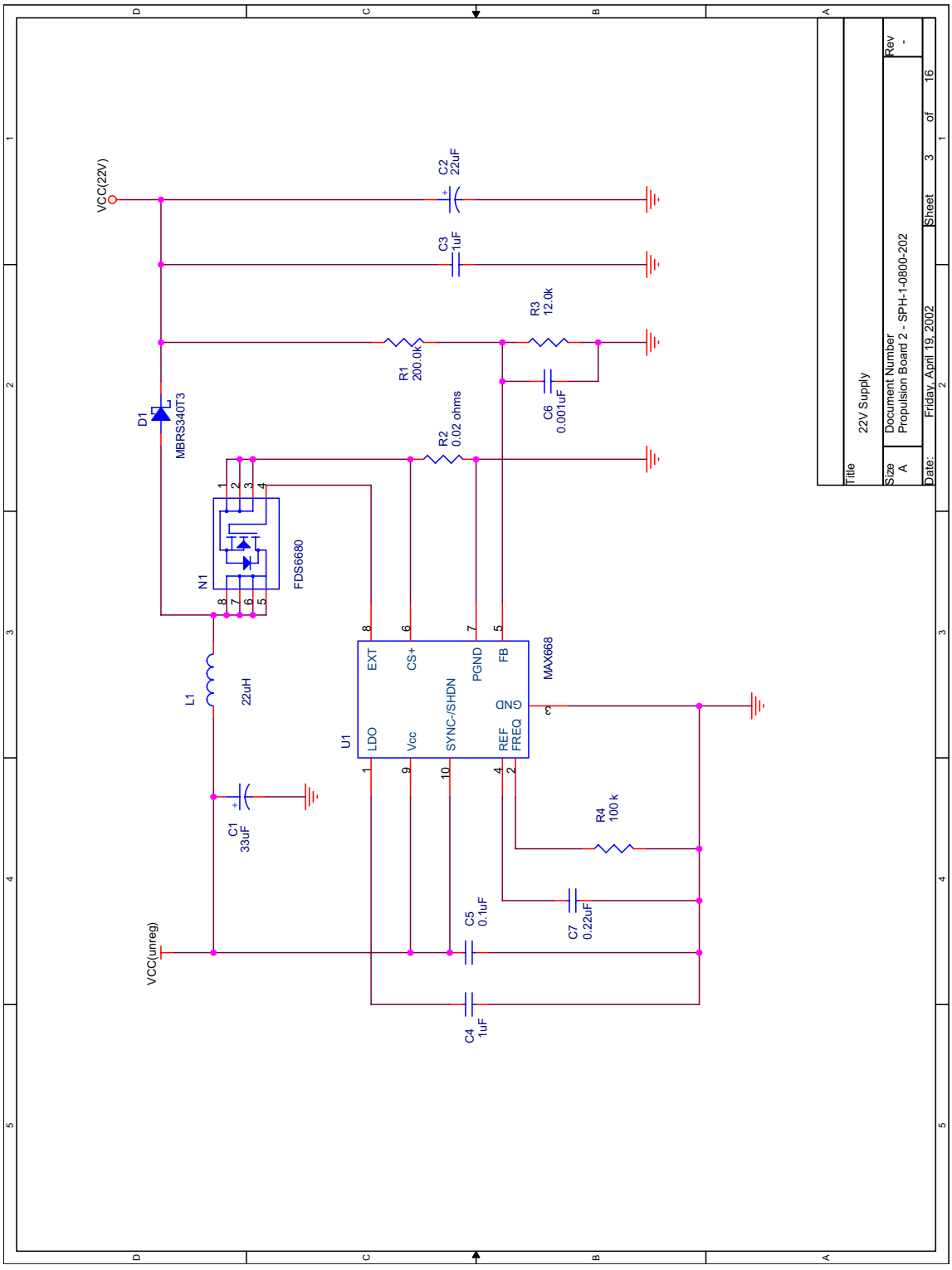
Signal	Type	Description
Thr +	I/O	Positive terminal of signal / solenoid
Thr -	I/O	Negative terminal of signal / solenoid

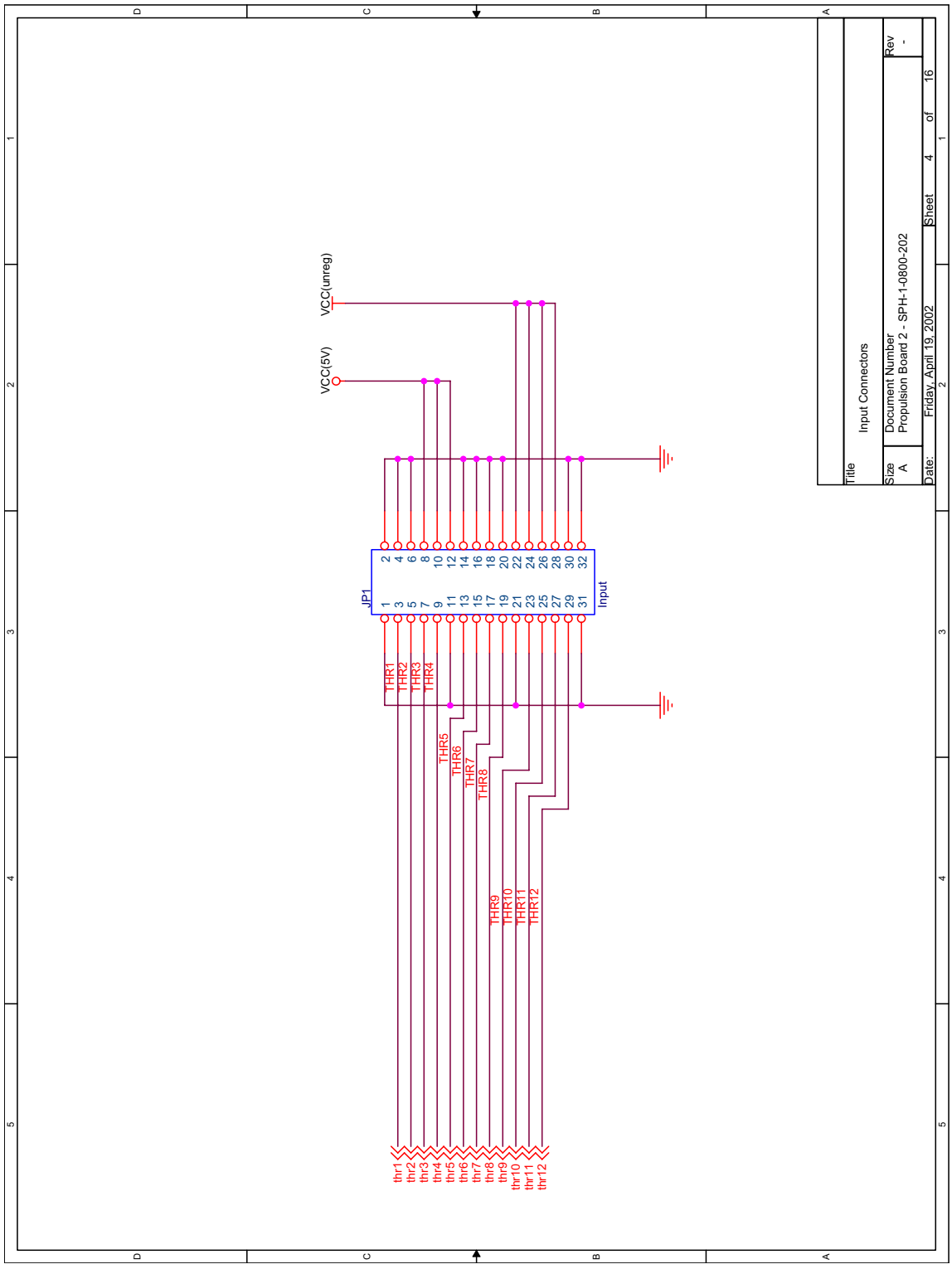
Schematics

The schematics for the propulsion sub-system follow.

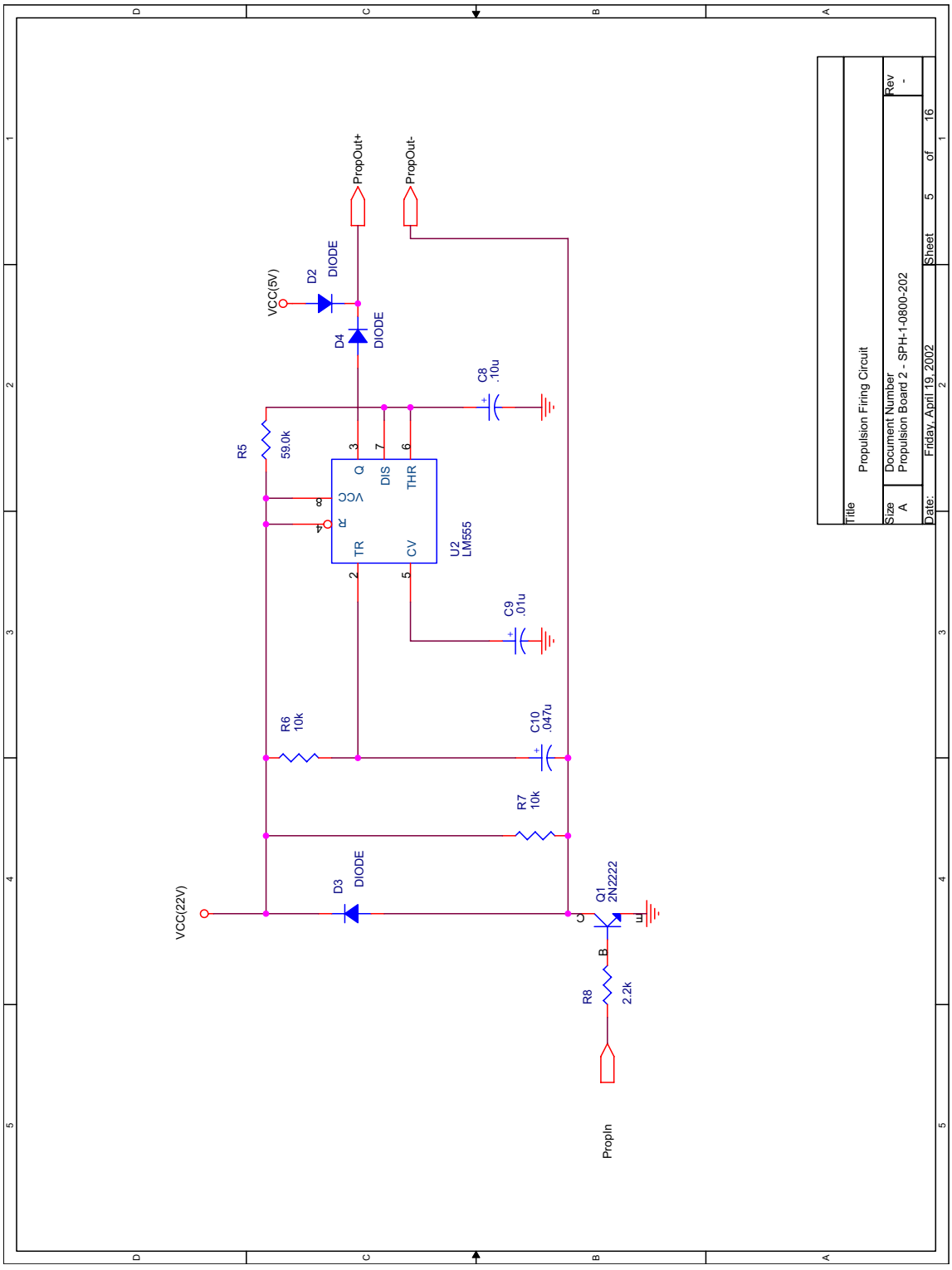

PosDriver_1
NegDriver_1
PosDriver_2
NegDriver_2
PosDriver_3
NegDriver_3
PosDriver_4
NegDriver_4
PosDriver_5
NegDriver_5
PosDriver_6
NegDriver_6
PosDriver_7
NegDriver_7
PosDriver_8
NegDriver_8
PosDriver_9
NegDriver_9
PosDriver_10
NegDriver_10
PosDriver_11
NegDriver_11
PosDriver_12
NegDriver_12
Driver_1
Driver_2
Driver_3
Driver_4
Driver_5
Driver_6
Driver_7
Driver_8
Driver_9
Driver_10
Driver_11
Driver_12
PropIn
PropOut+
PropOut-
Drivers
thr1
thr2
thr3
thr4
thr5
thr6
thr7
thr8
thr9
thr10
thr11
thr12
Title
Firing Circuit Layout
Size
A
Document Number
Propulsion Board 2 - SPH-1-0800-202
Rev
-
Date:
Friday, April 19, 2002
Sheet
1
of
16

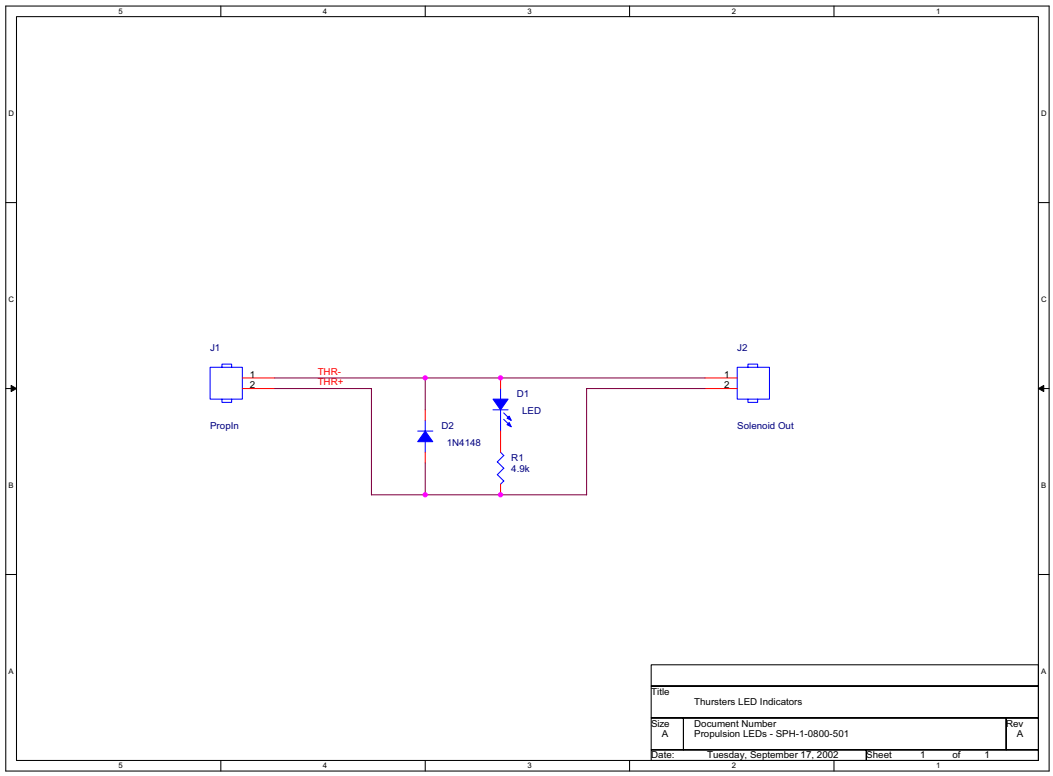

Title
Solenoid Connectors
Size
A
Document Number
Propulsion Board 2 - SPH-1-0800-202
Rev
-
Date:
Friday, April 19, 2002
Sheet
2
of
16
J1
THR-1
J2
THR-2
J3
THR-3
J4
THR-4
J5
THR-5
J6
THR-6
J7
THR-7
J8
THR-8
J9
THR-9
J10
THR-10
J11
THR-11
J12
THR-12
PosDriver_1
NegDriver_1
PosDriver_2
NegDriver_2
PosDriver_3
NegDriver_3
PosDriver_4
NegDriver_4
PosDriver_5
NegDriver_5
PosDriver_6
NegDriver_6
PosDriver_7
NegDriver_7
PosDriver_8
NegDriver_8
PosDriver_9
NegDriver_9
PosDriver_10
NegDriver_10
PosDriver_11
NegDriver_11
PosDriver_12
NegDriver_12


VCC(22V)
C2 22uF
C3 1uF
R3 12.0k
R1 200.0k
C6 0.001uF
D1
MBRS340T3
R2 0.02 ohms
N1
FDS6680
L1 22uH
U1
MAX668
EXT
CS+
PGND
FB
LDO
Vcc
SYNC-/SHDN
GND
REF
FREQ
C1 33uF
R4 100 k
C5 0.1uF
C7 0.22uF
VCC(unreg)
C4 1uF
Title
22V Supply
Size A
Document Number Propulsion Board 2 - SPH-1-0800-202
Rev -
Date: Friday, April 19, 2002
Sheet 3 of 16


VCC(unreg)
VCC(5V)
JP1
Input
THR1
THR2
THR3
THR4
THR5
THR6
THR7
THR8
THR9
THR10
THR11
THR12
thr1
thr2
thr3
thr4
thr5
thr6
thr7
thr8
thr9
thr10
thr11
thr12
Title
Input Connectors
Size
A
Document Number
Propulsion Board 2 - SPH-1-0800-202
Rev
-
Date:
Friday, April 19, 2002
Sheet
4
of
16

The following schematic repeats twelve times, once per signal/solenoid:





F.1.3 Data Processing (C6701 DSP / SMT375)

Design Drivers

- Support other subsystems' data processing needs
  - Communications data processing
  - Metrology computational support
  - Propulsion thruster actuation
  - Provide house-keeping information to user
    - Battery information
    - Tank usage
- Allow reconfiguration of control algorithms
  - Enable the complete software to be changed to allow testing of programs with different configurations and goals
- Maximize processing power for available volume and power

- Minimize processing needs of ‘bus’ system to maximize processing power available for control algorithms

Functional Block Diagram

A COTS product was selected to perform the data processing within each SPHERES satellite. The selected product is a Sundance Multiprocessor Technology Ltd SMT375 board which features a Texas Instruments TMS320C6701 Digital Signal Processor (DSP). The SMT375 board utilizes the Texas Instruments Module standard for the C40 DSP (TIM40), used in the prototype design of SPHERES. The standard implements a 32 bit global data bus with 31 address lines, plus six TI communications ports which split 32 bit data into bytes for a maximum data speed of 20MBps. A block diagram (simplified from [Sundance, 2003]) of the SMT375 board is presented in Figure F.6. The SPHERES metrology FPGA interfaces via the global data bus, while the communications system utilizes the commports.The features of the SMT375 board are summarized in Table F.8

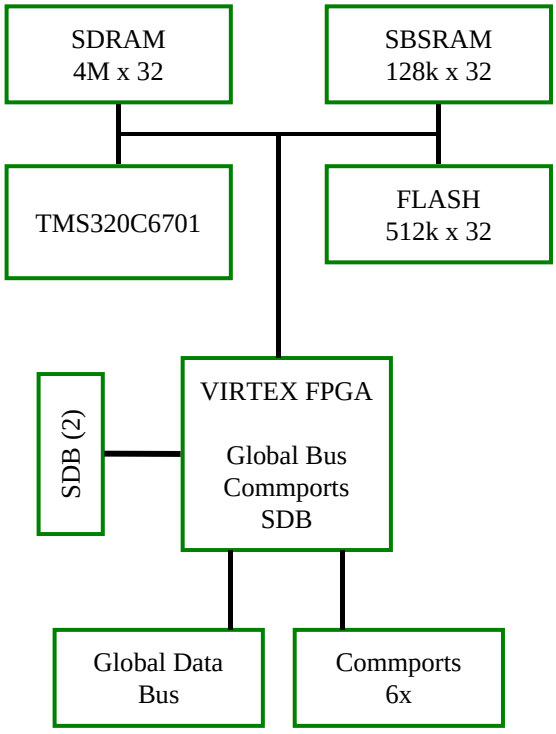


Figure F.6 SMT375 functional block diagram

TABLE F.8 Features of the SMT375 board

Form Factor	Single-width TIM40
CPU	TMS320C6701
Speed	167MHz
FLOPS	1 GFLOPS peak
RAM	16MB (4M x 32)
Cache	512k (128k x 32)
Commports	6 x 20MBps
Programming	C and C++
Power Consumption	7 W

The SMT375 interfaces to the rest of the sub-systems through the metrology FPGA board via three 80-pin connectors. The signal descriptions of these connectors are explained in the metrology and communications sections which utilize them.

F.1.4 Metrology

Design Drivers

- Provide real-time position and attitude information of each satellite
- Implements measurement circuitry for metrology
  - One infrared transmitter command (protected)
  - 12 infrared receiver channels
  - 24 ultrasound receiver channels
  - Six 12-bit A/D channels, up to 1KHz
- Propulsion register (to control solenoid valves)
  - 12 outputs with read-back capability
- General output register
  - Two outputs with read-back capability
- General input register
  - Two inputs

Functional Block Diagrams

Because the metrology motherboard is the only board which interfaces directly with the SMT375 DSP board, it functions not only to support metrology, but also to provide general input/output signals and pass-through of the commports and power to the communications system. To implement these functions the metrology system centers its design around an FPGA as pictured in Figure F.7. The system interfaces to the twelve US/IR boards, the internal beacon, three gyroscopes, and three accelerometers (with amplifiers). An EEPROM stores the configuration of the FPGA and interfaces to a JTAG port to update the programming as necessary.

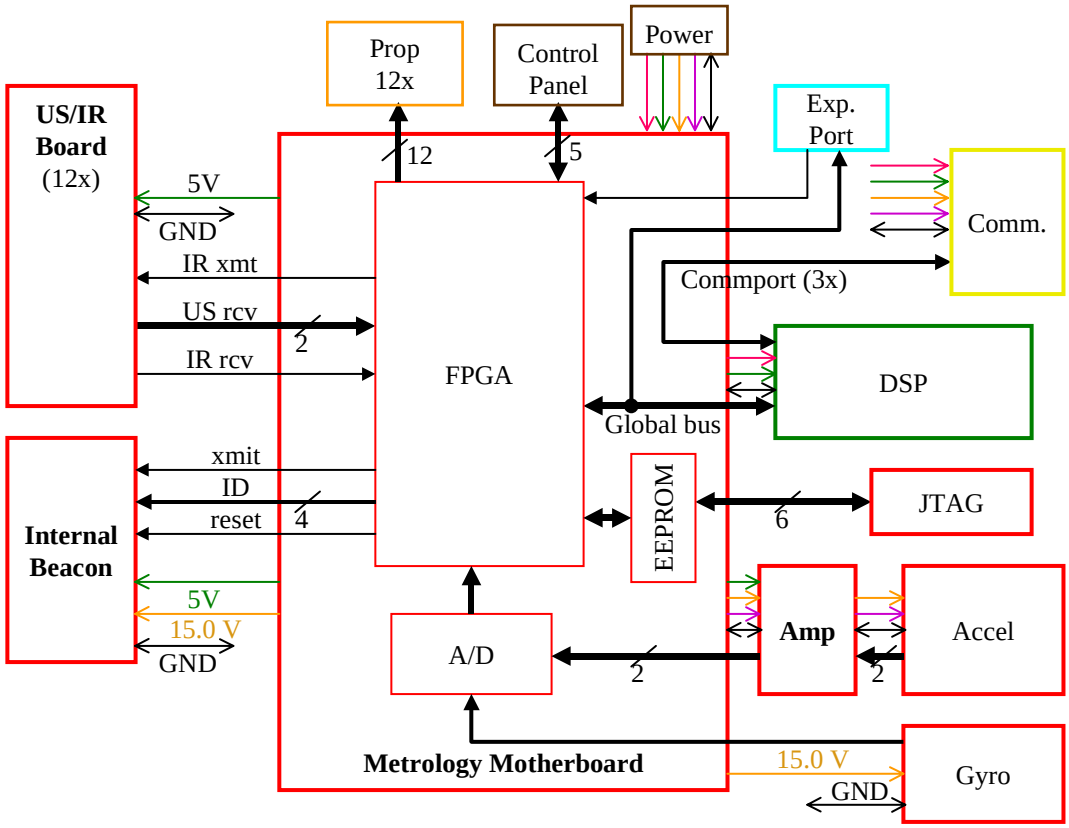


Figure F.7 Metrology sub-system functional block diagram

The four boards that support the metrology sub-system (motherboard, US/IR, accelerometer amplifier, and internal beacon) are described next.

Metrology Motherboard

The metrology system utilizes a Xilinx FPGA which interfaces with the DSP via the global bus and implements general input/output for the propulsion and control panel boards. The FPGA also collects the range measurements of the US/IR system independently of the DSP, to minimize the load of the metrology system on the DSP. The FPGA implements its own 25MHz counter which is reset automatically when an IR signal is received; registers on the FPGA store the time taken to receive an ultrasound signal at each of the 24 receivers without interrupting the DSP. The DSP is only interrupted once after each beacon transmits its signal (i.e., a total of five times when five beacons are in use, but not 120 times if each receiver interrupted). The parallel processing of the FPGA ensures that the signals from each receiver are recorded correctly. The FPGA interfaces with a 12-bit analog to digital converter to provide up to 1kHz data from the gyroscopes and accelerometers. The board provides a single pole low-pass filter with a drop-off frequency at 300Hz implemented with a simple RC circuit. Because the metrology motherboard is oriented in the X plane, it hosts one accelerometer and its amplification circuit. Figure F.8 shows a schematic of the FPGA firmware design.

Table F.9 describes the inputs and outputs of the metrology mother board.

TABLE F.9 Metrology motherboard signals description

Section	Signal	Type	Description
From Power	Vcc(+5V)	Pwr	+5V power
	Vcc(+3.3V)	Pwr	+3.3V power
	Vcc(+15V)	Pwr	+15V power
	Vcc(-15V)	Pwr	-15V power
	GND	Pwr	Common ground

TABLE F.9 Metrology motherboard signals description

Section	Signal	Type	Description
Data to power and propulsion stack	THR 1-12	Out	Pass through signals for thrusters 1-12
	/Enable	In	Enable button signal
	/LED-enable	Out	Enable LED signal
	/Batlow	In	Low battery indicator signal
	WDOG	Out	Watchdog control signal
	/RESET	In	Reset
Data and power to SMT375	Vcc(+5V)	Pwr	+5V power
	Vcc(+3.3V)	Pwr	+3.3V power
	GND	Pwr	Common ground
	C[0-5] D[0-7]	I/O	Commport data lines
	CACK[0-5]	I/O	Commport acknowledge signal
	CRDY[0-5]	I/O	Commport ready signal
	CREQ[0-5]	I/O	Commport request signal
	STRB[0-5]	I/O	Commport strobe
	/RESET	Out	Reset Line
	IR_RCV_INT	Out	Infrared reception interrupt. It is asserted when an IR is received so the DSP can prepare for a global metrology cycle
	PADS_INT	Out	1kHz interrupt. Provides timing for the DSP and indicates IMU data is available.
	A0-A30	Out	Global bus address lines
	D0-D31	I/O	Global bus data lines
	RDY1	Out	Global bus ready
	PAGE1	In	Global bus page select
	STRB1	In	Global bus strobe
	R/W1	In	Global bus read/write
	/CE1	Out	Global bus control lines enable
	/OE	Out	Global bus data lines enable
	/AE	Out	Global bus address lines enable

TABLE F.9 Metrology motherboard signals description

Section	Signal	Type	Description
Data and power to communications board	Vcc(+5V)	Pwr	+5V power
	Vcc(+3.3V)	Pwr	+3.3V power
	Vcc(+15V)	Pwr	+15V power
	Vcc(-15V)	Pwr	-15V power
	GND	Pwr	Common ground
	A0-A30	Out	Global bus address lines (expansion port)
	D0-D31	I/O	Global bus data lines (expansion port)
	RDY1	I/O	Global bus ready (expansion port)
	PAGE1	I/O	Global bus page select (expansion port)
	STRB1	Out	Global bus strobe (expansion port)
	R/W1	I/O	Global bus read/write (expansion port)
	/RESET	Out	Reset line
	/Exp_port_in	In	High when an expansion port selects to bypass the satellite US/IR metrology boards
	IR_XMIT	Out	IR transmit command
	US-RX[11-12]-[1-2]	In	Input ultrasound signals from the expansion port board
	IR-RX[11-12]	In	Input infrared signals from the expansion port board
	EXP A2D [0-2]	In	Input analog signals from the expansion port board
	C[1,2,4] D[0-7]	I/O	Commport data lines
	CACK[1,2,4]	I/O	Commport acknowledge signal
	CRDY[1,2,4]	I/O	Commport ready signal
	CREQ[1,2,4]	I/O	Commport request signal
	STRB[1,2,4]	I/O	Commport strobe
US/IR Boards (12x)	Vcc(+5V)	Pwr	+5V power
	GND	Pwr	Common ground
	IR_XMIT	Out	IR transmit command
	US-RX[1-2]	In	Input ultrasound signals
	IR-RX	In	Input infrared signal

TABLE F.9 Metrology motherboard signals description

Section	Signal	Type	Description
Onboard beacon	Vcc(+5V)	Pwr	+5V power
	Vcc(+15V)	Pwr	+15V power
	GND	Pwr	Common ground
	IR_RCV_INT	Out	IR Received interrupt signal - commands the beacon to initiate its US transmit process
	B-MCLR	Out	Beacon enable/reset
	B-NUM[0-3]	Out	Beacon configuration number
Gyros. (3x)	Vcc(+15V)	Pwr	+15V power
	GND	Pwr	Common ground
	[X,Y,Z] Gyro	In	Gyroscope analog signal
Accels. (2x)	Vcc(+5V)	Pwr	+5V power
	Vcc(+15V)	Pwr	+15V power
	Vcc(-15V)	Pwr	-15V power
	GND	Pwr	Common ground
	[Y,Z] Accel-Amp	In	Y and Z accelerometer amplified analog signals
	[Y,Z] Temp-Temp	In	Y and Z accelerometer temperature sensor signal
JTAG	Vcc(+3.3V)	Pwr	+3.3V power
	GND	Pwr	Common ground
	PTCK	In	Clock
	PTDO	Out	Data out
	PTDI	In	Data in
	PTMS	In	Select

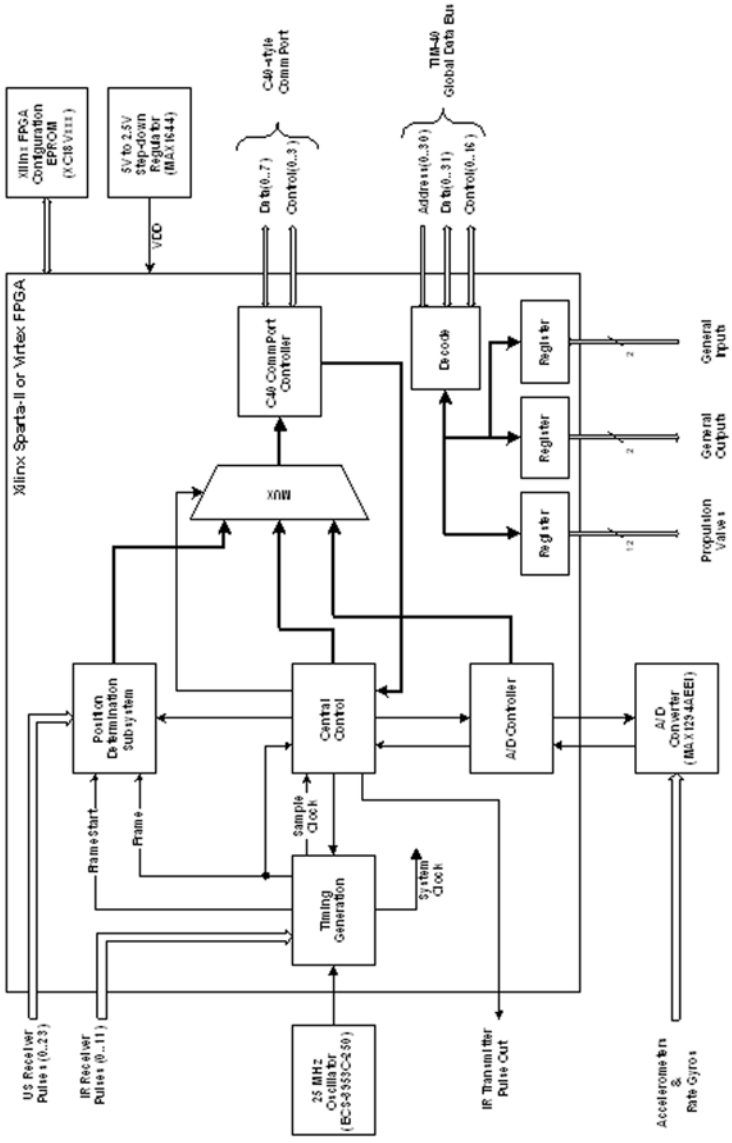


Figure F.8 FPGA firmware design

Metrology US/IR Boards

The metrology ultrasound/infrared board receive and amplify the ultrasound signals emitted by the global metrology beacons. They also transmit and receive the infrared signal which initiates the global metrology process. Therefore, as pictured in Figure F.9, each board consists of three main elements: infrared transmit, infrared receive, and ultrasound receive. The infrared transmit circuit uses a transistor to drive 1A of current through an

infrared LED; the FPGA ensures the 1A pulse is no longer than 10µs and has a maximum duty cycle of 10% to protect the LEDs. The infrared receive utilizes a COTS receiver which directly outputs a logic signal to the FPGA. The ultrasound receive signal required the use of a quad op-amp to amplify, rectify, and digitize the signal. A schematic of the circuit is presented in Figure F.10. The signals of the US/IR boards are described in Table F.9, reversing inputs and outputs.

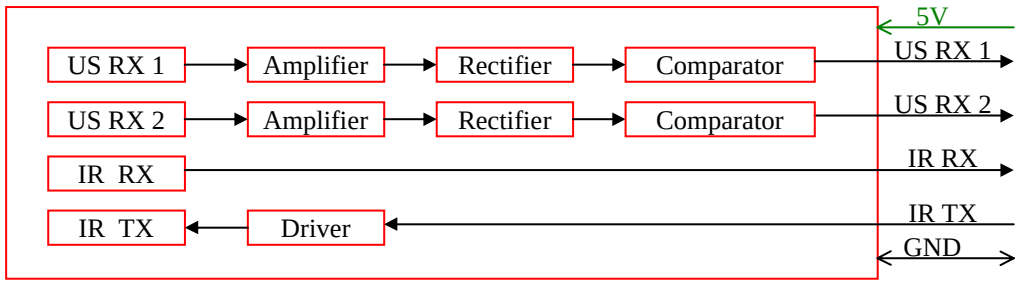


Figure F.9 US/IR boards functional block diagram

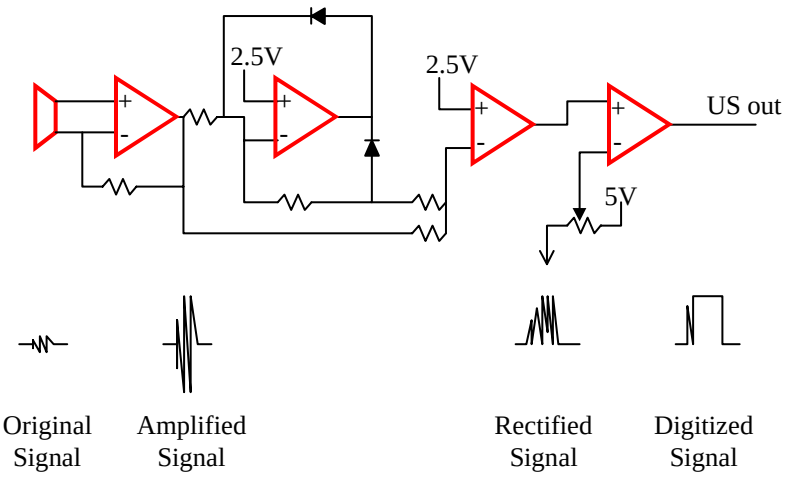


Figure F.10 Ultrasound amplification schematic

Accelerometer Amplifier Boards

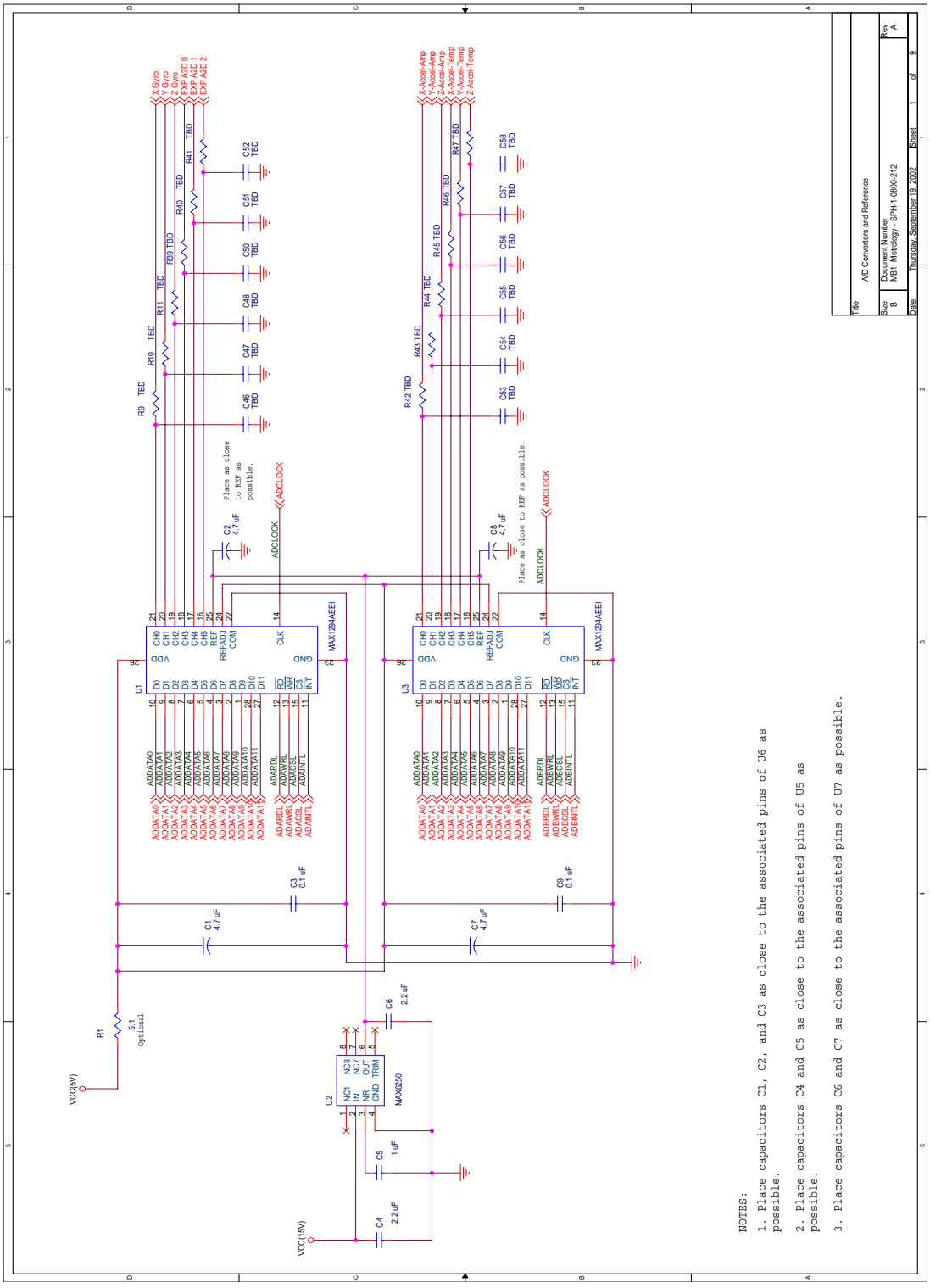
The selected accelerometers have an operational range of ±30g but are capable of measuring accelerations with milli-g precision. Because the SPHERES thrusters create accelerations in the milli-g range, the accelerometer require custom amplifiers to provide the necessary 0-5V analog signal required by the A2D converter. The accelerometer operate like current sources, therefore they require a sense resistor at the output to create a voltage for measurement. The circuit selected a sense resistor which allows the use of a high-frequency op-amp to amplify the input signal 40 times. The accelerometers also output a temperature measurement signal from a thermistor, so that software can take into account temperature changes. While not used in the initial implementation of SPHERES, the accelerometer boards send the temperature signal to the FPGA in case it proves necessary in the future. A description of the signals of the accelerometer boards is presented in Table F.9, reversing inputs and outputs.

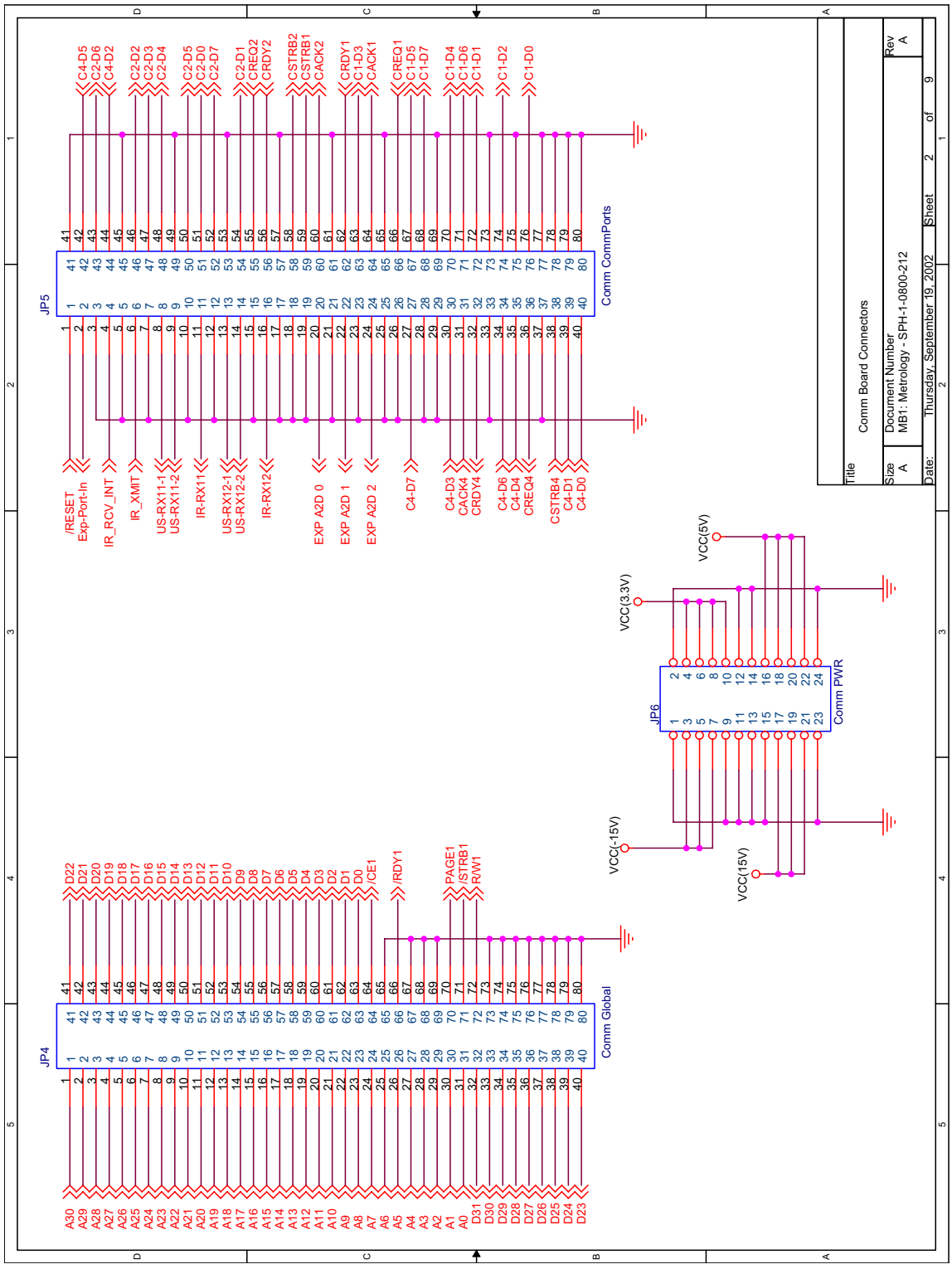
Internal Beacon

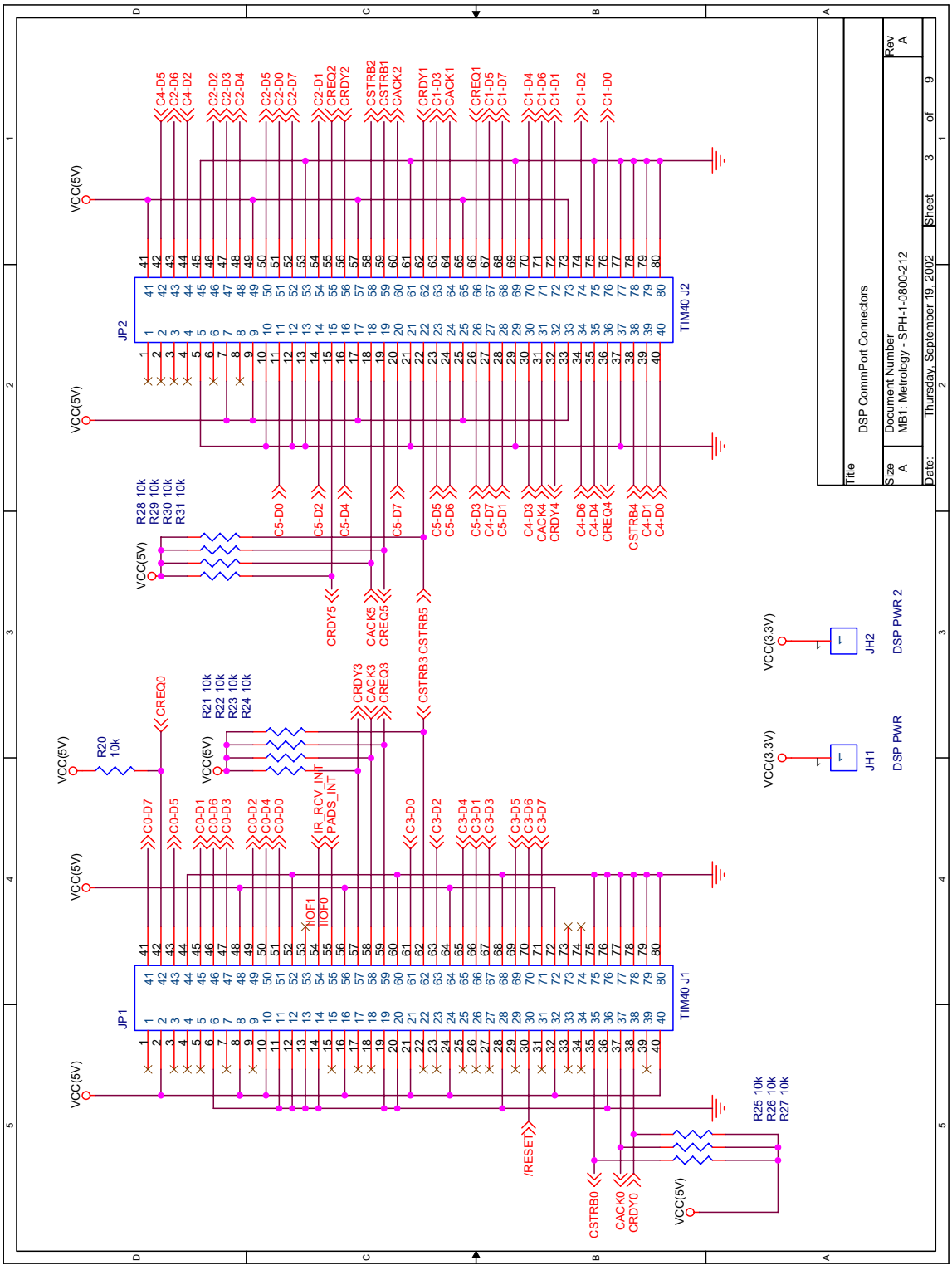
The internal beacon replicates an external beacon (see Section F.3 below), but omits the infrared reception circuitry, does not use a manual switch, and utilizes custom power regulation circuitry. The infrared circuitry and ID number selector are replaced by signals from the FPGA. The power sub-system provides regulated 5V. The internal beacon regulates +15V to +12V using a linear regulator. Descriptions of the signals are presented in Table F.9, reversing inputs and outputs.

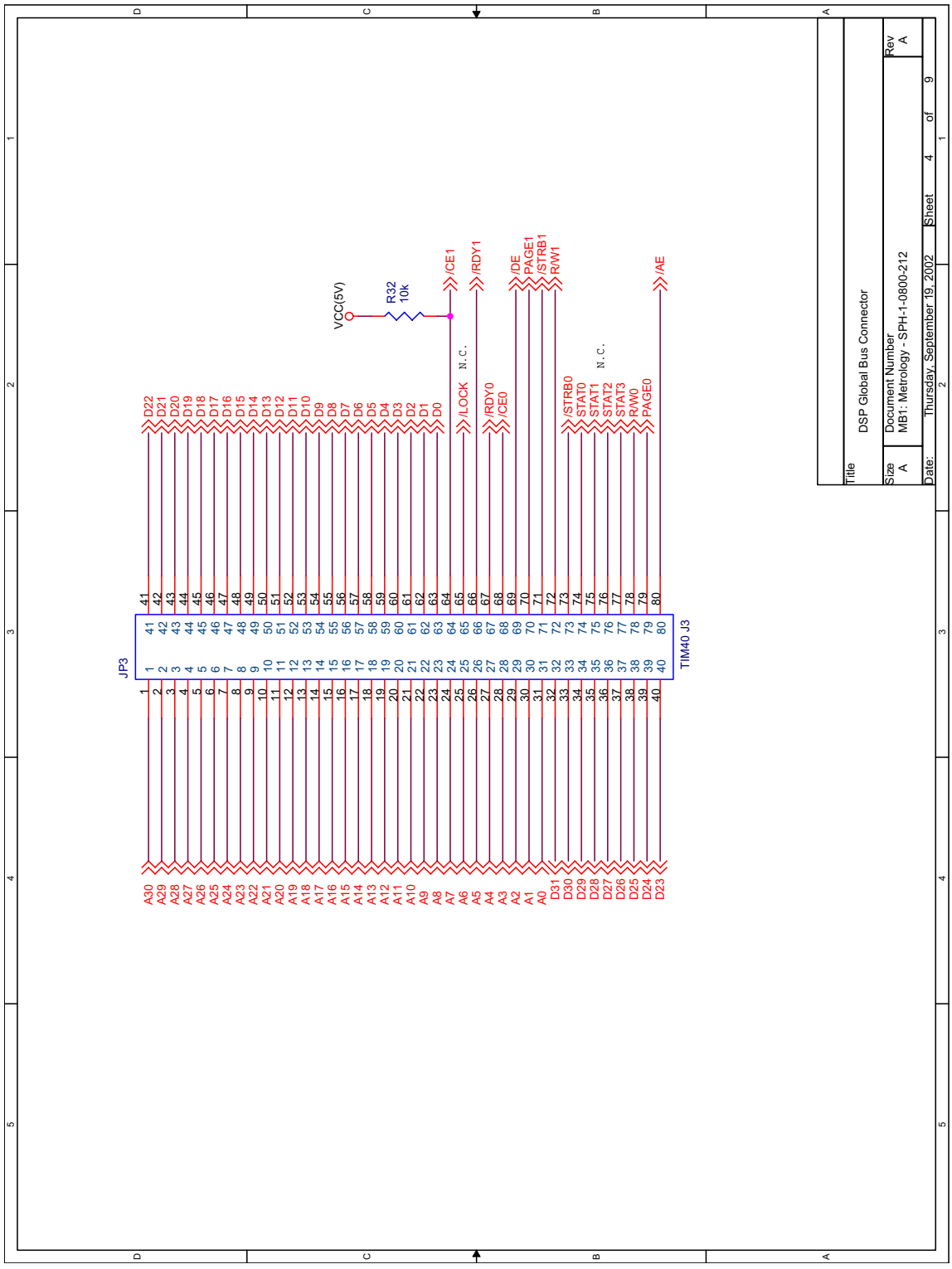
Schematics

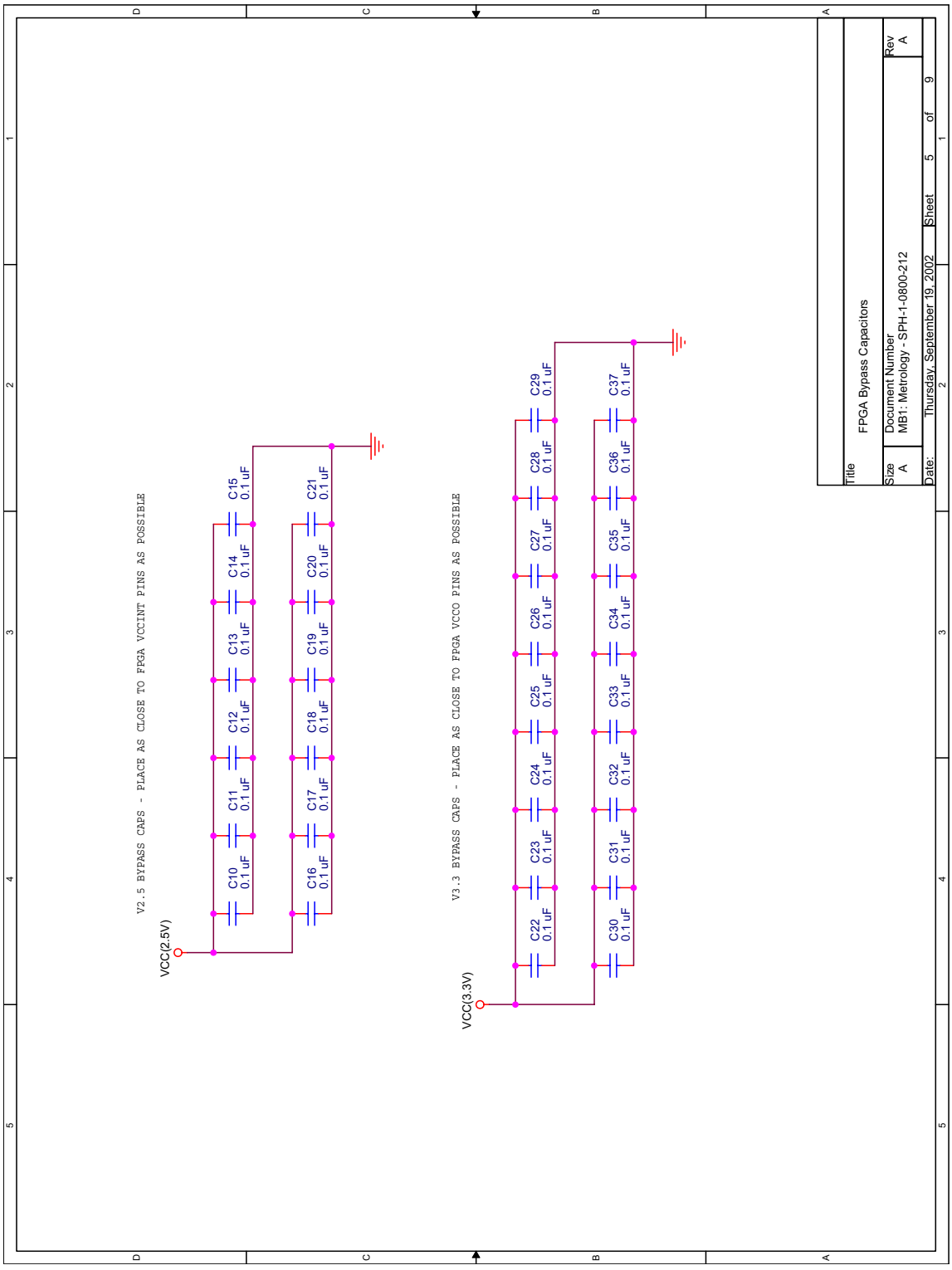
The schematics for all the metrology boards are presented next.

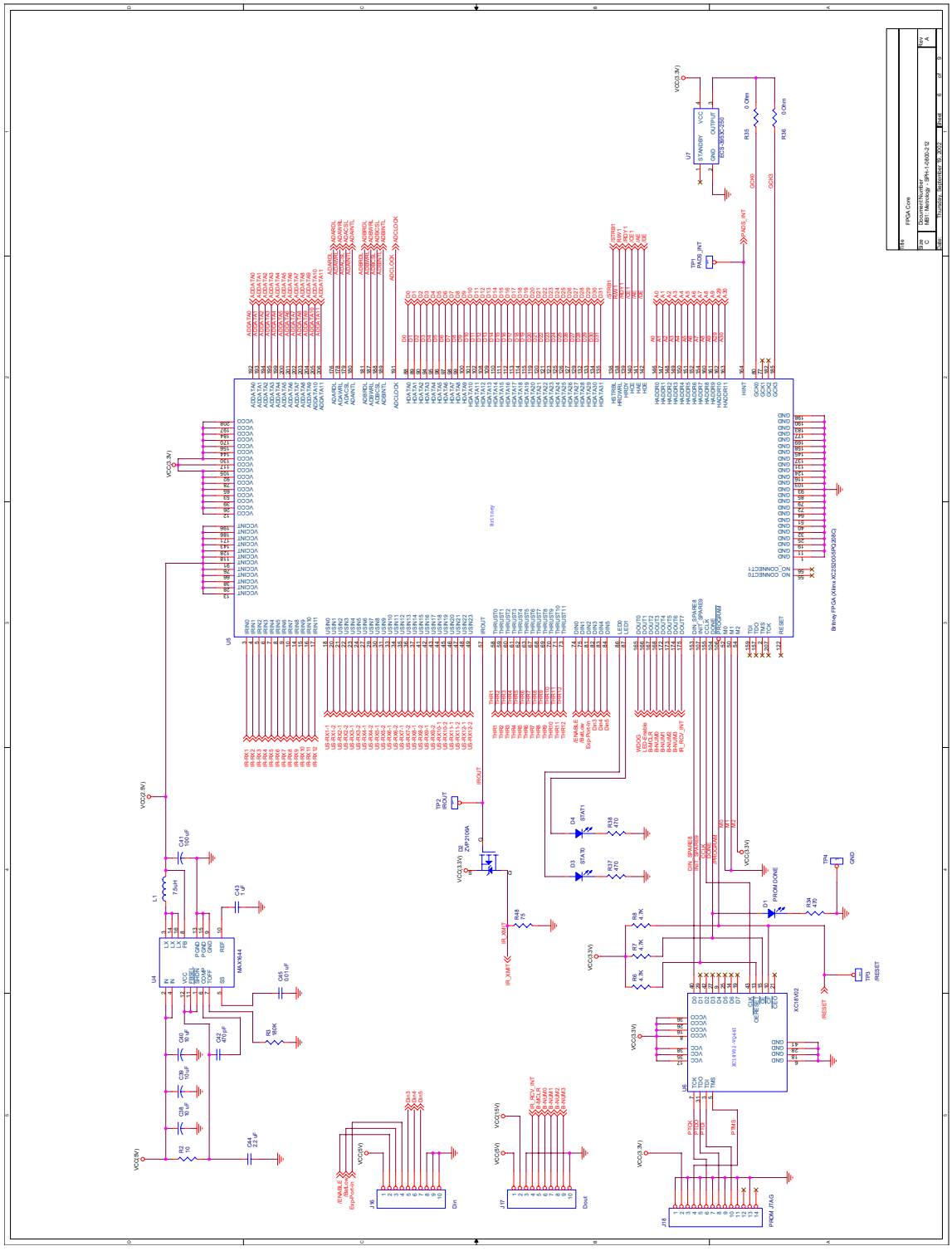

NOTES:
1. Place capacitors C1, C2, and C3 as close to the associated pins of U6 as possible.
2. Place capacitors C4 and C5 as close to the associated pins of U5 as possible.
3. Place capacitors C6 and C7 as close to the associated pins of U7 as possible.
A/D Converters and Reference
Document Number
MB1: Mekology - SPH-1-0800-212
Rev A
Thursday, September 19, 2002
Sheet 1 of 9
MAX1294AEEI
MAX6250
ADCLOCK
Place as close to REF as possible.
X Gyro
Y Gyro
Z Gyro
EXP ADD 0
EXP ADD 1
EXP ADD 2
X-Accel-Amp
Y-Accel-Amp
Z-Accel-Amp
X-Accel-Temp
Y-Accel-Temp
Z-Accel-Temp
VCC(5V)
VCC(15V)

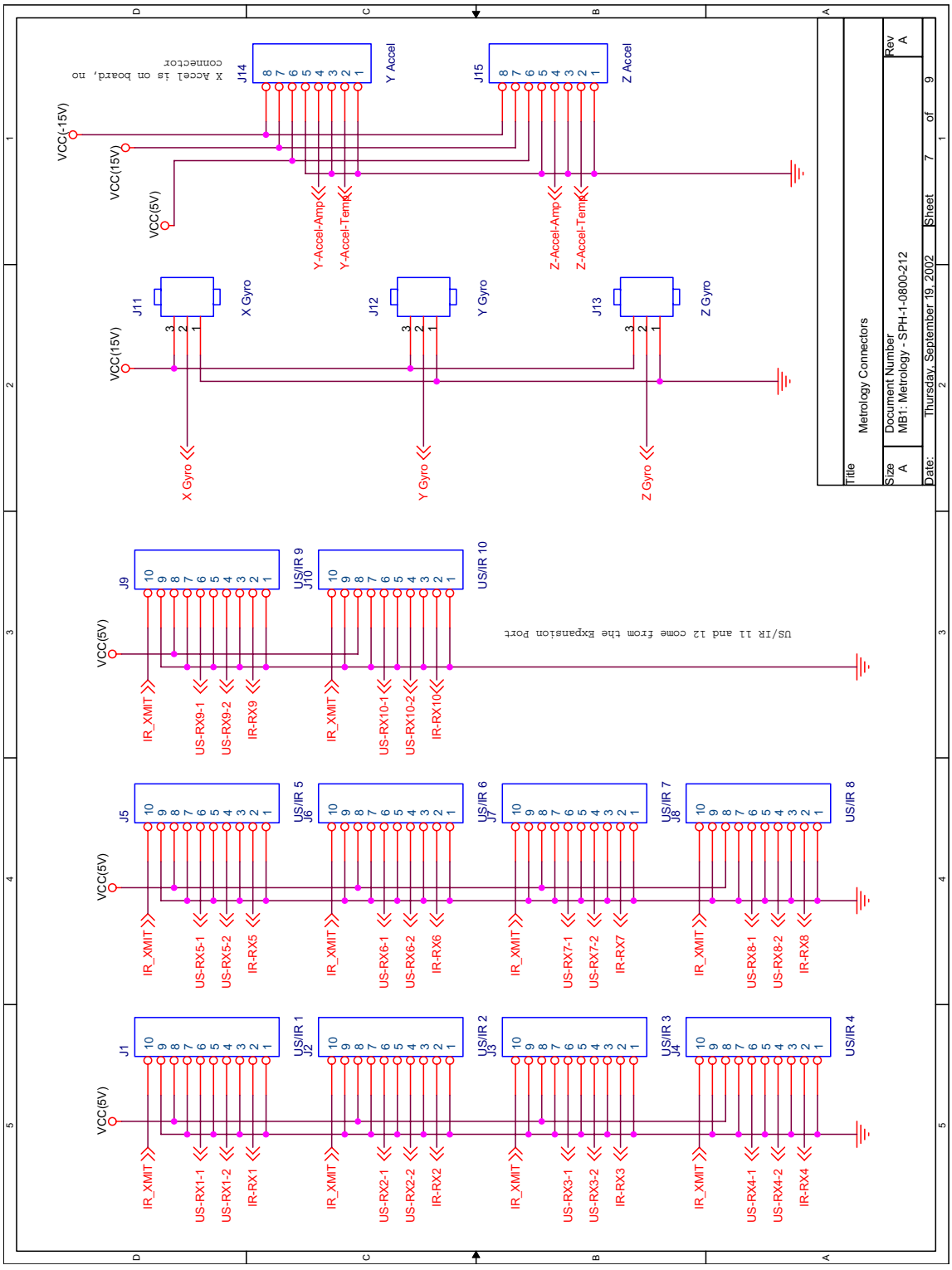

JP4
Comm Global
JP5
Comm CommPorts
JP6
Comm PWR
/RESET
Exp-Port-In
IR_RCV_INT
IR_XMIT
US-RX11-1
US-RX11-2
IR-RX11
US-RX12-1
US-RX12-2
IR-RX12
EXP A2D 0
EXP A2D 1
EXP A2D 2
C4-D7
C4-D3
CACK4
CRDY4
C4-D6
C4-D4
CREQ4
CSTRB4
C4-D1
C4-D0
C4-D5
C2-D6
C4-D2
C2-D2
C2-D3
C2-D4
C2-D5
C2-D0
C2-D7
C2-D1
CREQ2
CRDY2
CSTRB2
CSTRB1
CACK2
CRDY1
C1-D3
CACK1
CREQ1
C1-D5
C1-D7
C1-D4
C1-D6
C1-D1
C1-D2
C1-D0
D22
D21
D20
D19
D18
D17
D16
D15
D14
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
/CE1
/RDY1
PAGE1
/STRB1
R/W1
A30
A29
A28
A27
A26
A25
A24
A23
A22
A21
A20
A19
A18
A17
A16
A15
A14
A13
A12
A11
A10
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
D31
D30
D29
D28
D27
D26
D25
D24
D23
VCC(3.3V)
VCC(5V)
VCC(-15V)
VCC(15V)
Title
Comm Board Connectors
Size
A
Document Number
MB1: Metrology - SPH-1-0800-212
Rev
A
Date:
Thursday, September 19, 2002
Sheet
2
of
9

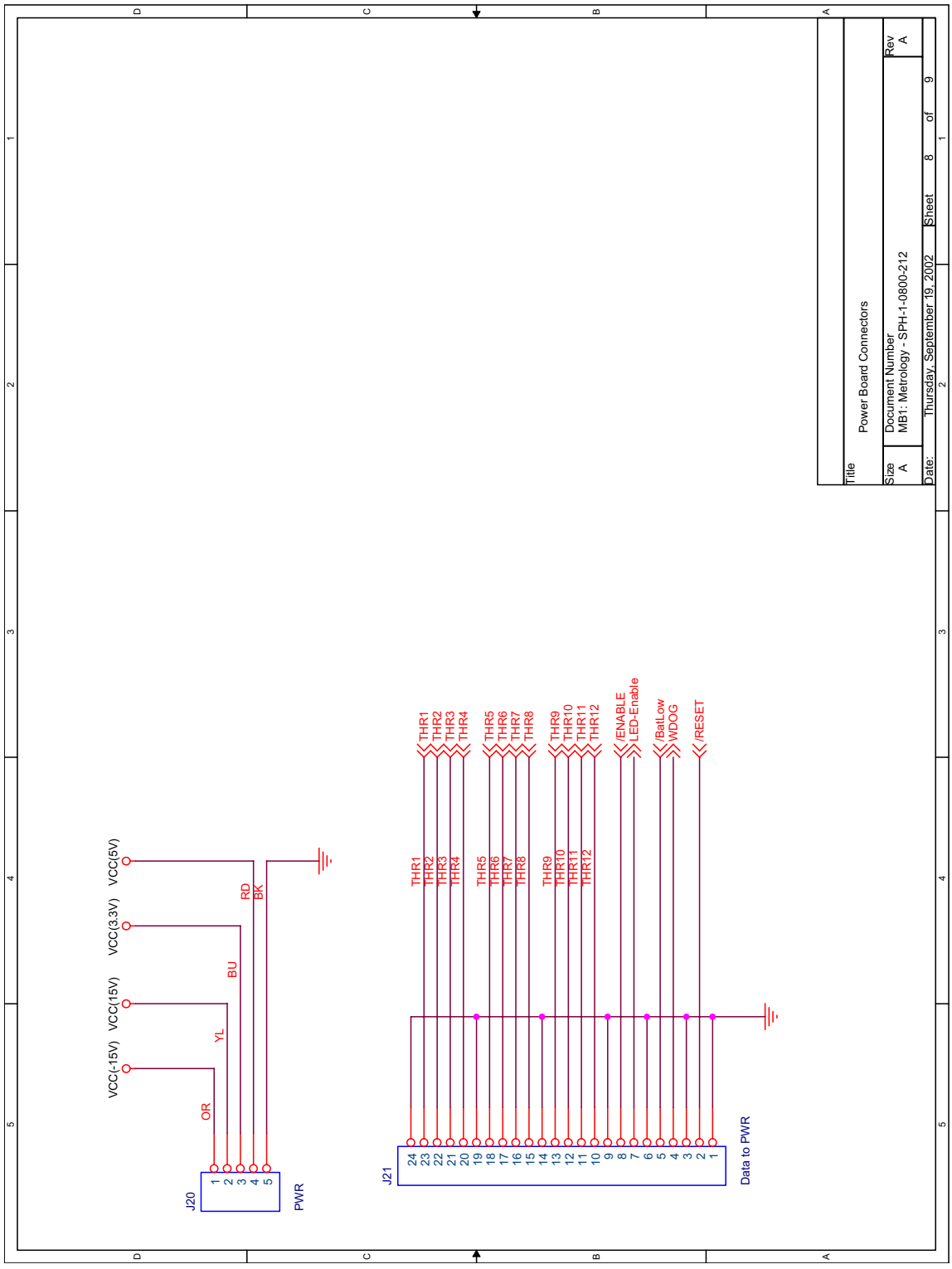

JP1
TIM40 J1
JP2
TIM40 J2
VCC(5V)
VCC(3.3V)
R20 10k
R21 10k
R22 10k
R23 10k
R24 10k
R25 10k
R26 10k
R27 10k
R28 10k
R29 10k
R30 10k
R31 10k
JH1
DSP PWR
JH2
DSP PWR 2
/RESET
IR_RCV_INT
PADS_INT
Title
DSP CommPort Connectors
Size A
Document Number
MB1: Metrology - SPH-1-0800-212
Rev A
Date: Thursday, September 19, 2002
Sheet 3 of 9

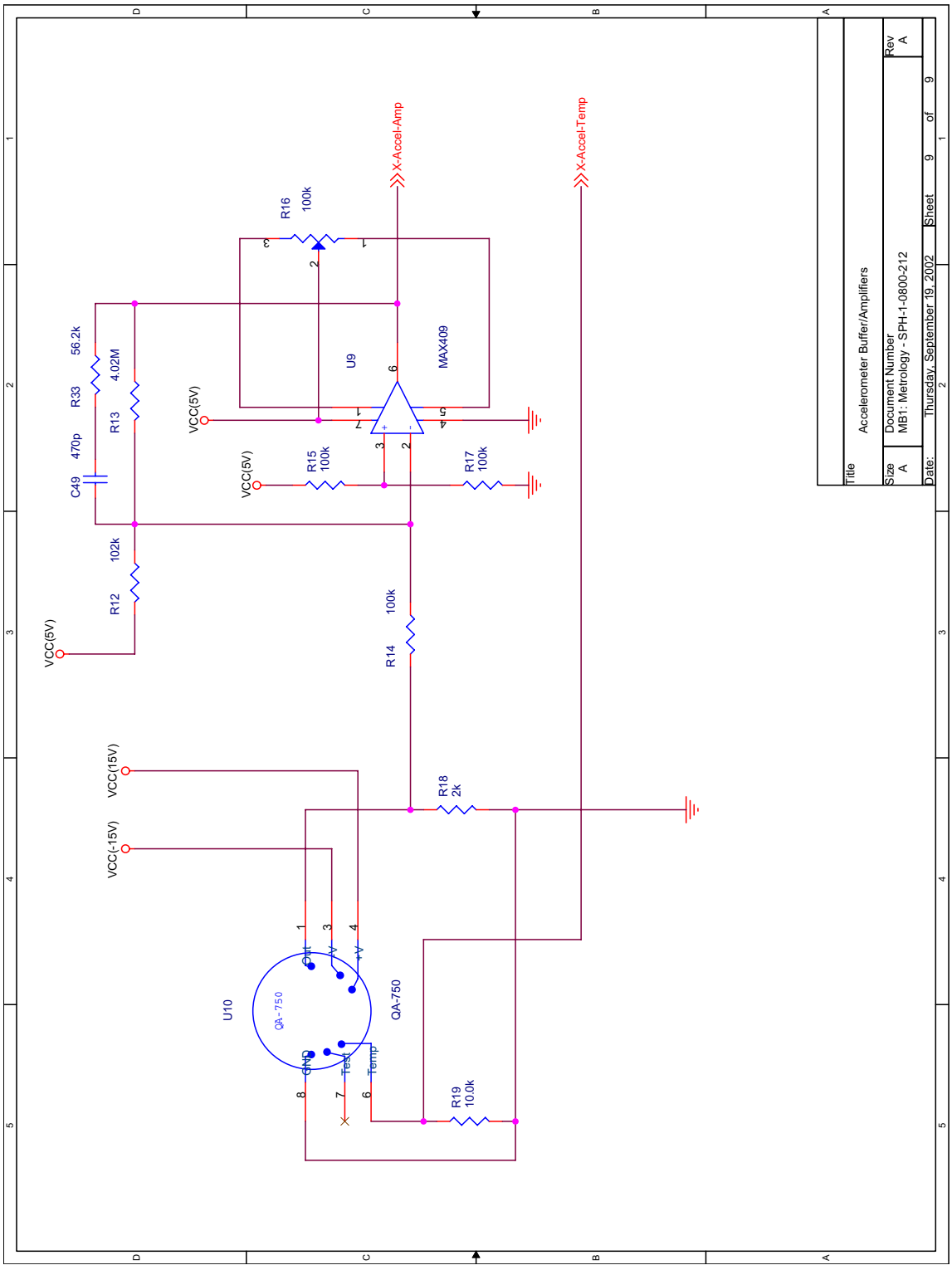

JP3
TIM40 J3
VCC(5V)
R32
10k
/CE1
/RDY1
/DE
PAGE1
/STRB1
R/W1
/AE
/LOCK
/RDY0
/CE0
/STRB0
STAT0
STAT1
STAT2
STAT3
R/W0
PAGE0
N.C.
N.C.
D22 D21 D20 D19 D18 D17 D16 D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0
A30 A29 A28 A27 A26 A25 A24 A23 A22 A21 A20 A19 A18 A17 A16 A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0 D31 D30 D29 D28 D27 D26 D25 D24 D23
Title
DSP Global Bus Connector
Size A
Document Number MB1: Metrology - SPH-1-0800-212
Rev A
Date: Thursday, September 19, 2002
Sheet 4 of 9

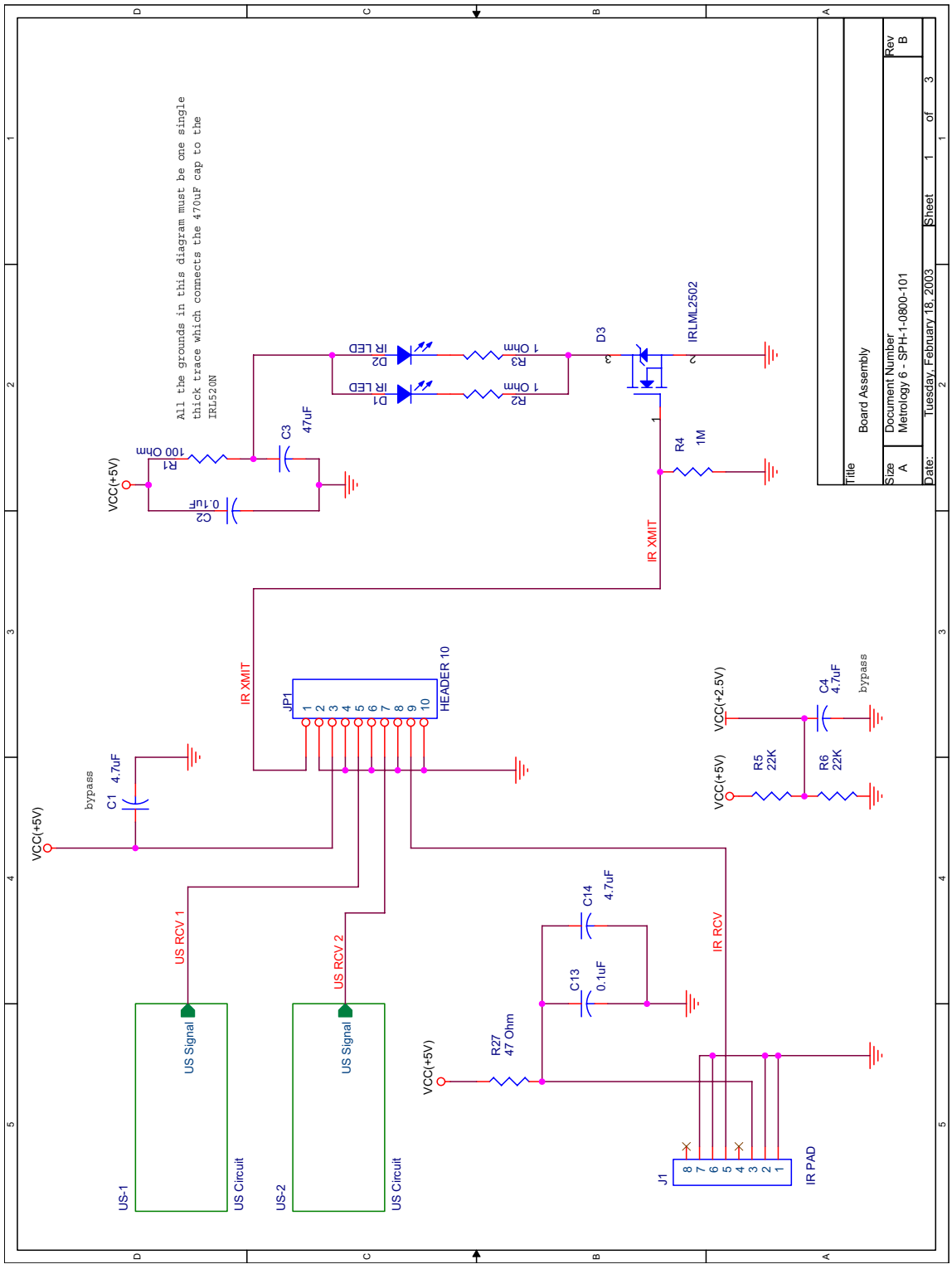

V2.5 BYPASS CAPS - PLACE AS CLOSE TO FPGA VCCINT PINS AS POSSIBLE
VCC(2.5V)
C10 0.1 uF
C11 0.1 uF
C12 0.1 uF
C13 0.1 uF
C14 0.1 uF
C15 0.1 uF
C16 0.1 uF
C17 0.1 uF
C18 0.1 uF
C19 0.1 uF
C20 0.1 uF
C21 0.1 uF
V3.3 BYPASS CAPS - PLACE AS CLOSE TO FPGA VCCO PINS AS POSSIBLE
VCC(3.3V)
C22 0.1 uF
C23 0.1 uF
C24 0.1 uF
C25 0.1 uF
C26 0.1 uF
C27 0.1 uF
C28 0.1 uF
C29 0.1 uF
C30 0.1 uF
C31 0.1 uF
C32 0.1 uF
C33 0.1 uF
C34 0.1 uF
C35 0.1 uF
C36 0.1 uF
C37 0.1 uF
Title
FPGA Bypass Capacitors
Size
A
Document Number
MB1: Metrology - SPH-1-0800-212
Rev
A
Date:
Thursday, September 19, 2002
Sheet
5
of
9



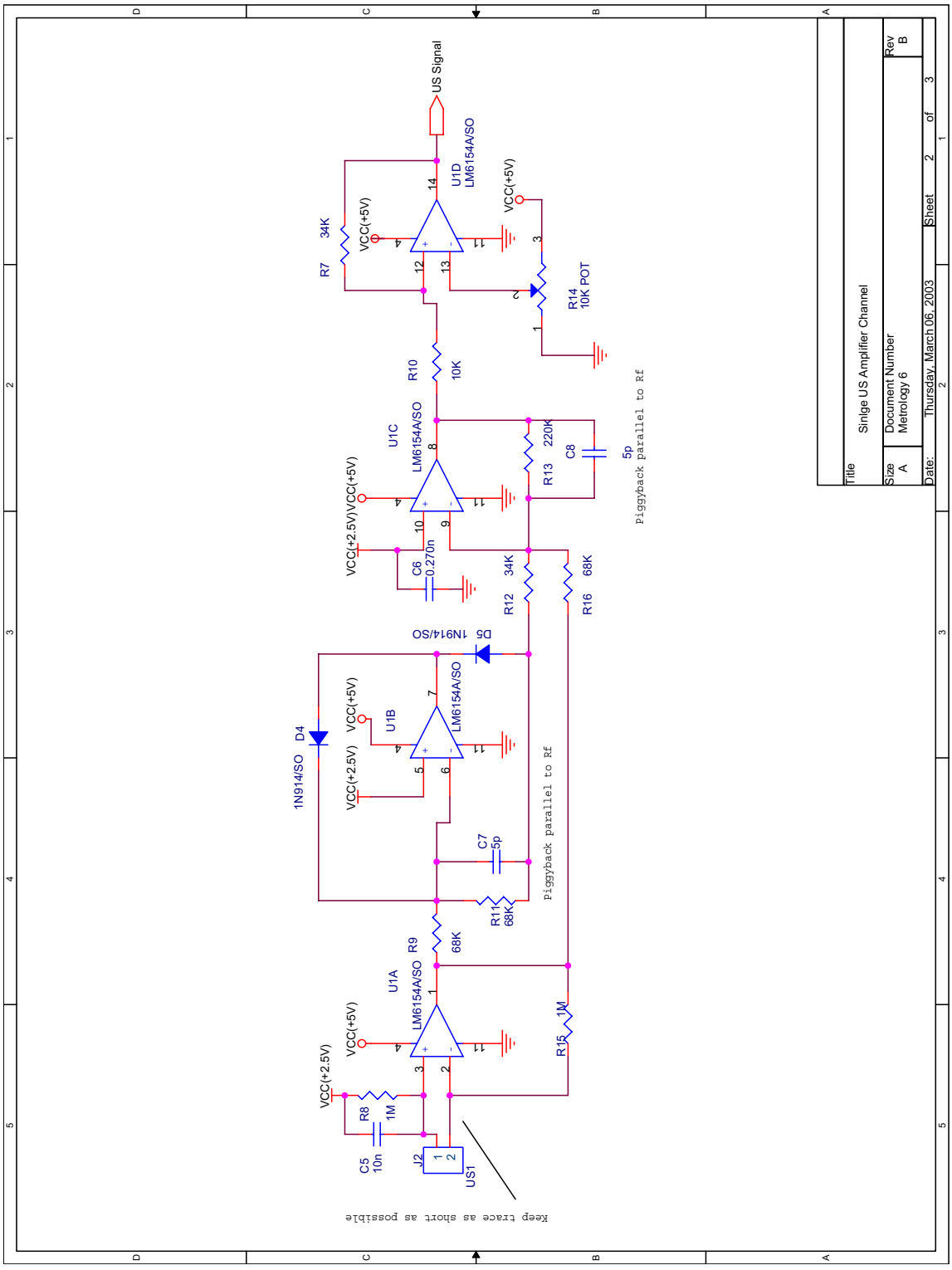

X Accel is on board, no connector
VCC(-15V)
VCC(15V)
VCC(5V)
J14
Y Accel
J15
Z Accel
Y-Accel-Amp
Y-Accel-Temp
Z-Accel-Amp
Z-Accel-Temp
J11
X Gyro
J12
Y Gyro
J13
Z Gyro
US/IR 9
US/IR 10
US/IR 5
US/IR 6
US/IR 7
US/IR 8
US/IR 1
US/IR 2
US/IR 3
US/IR 4
IR_XMIT
US-RX9-1
US-RX9-2
IR-RX9
US-RX10-1
US-RX10-2
IR-RX10
US-RX5-1
US-RX5-2
IR-RX5
US-RX6-1
US-RX6-2
IR-RX6
US-RX7-1
US-RX7-2
IR-RX7
US-RX8-1
US-RX8-2
IR-RX8
US-RX1-1
US-RX1-2
IR-RX1
US-RX2-1
US-RX2-2
IR-RX2
US-RX3-1
US-RX3-2
IR-RX3
US-RX4-1
US-RX4-2
IR-RX4
US/IR 11 and 12 come from the Expansion Port
Title
Metrology Connectors
Size
A
Document Number
MB1: Metrology - SPH-1-0800-212
Rev
A
Date:
Thursday, September 19, 2002
Sheet
7
of
9

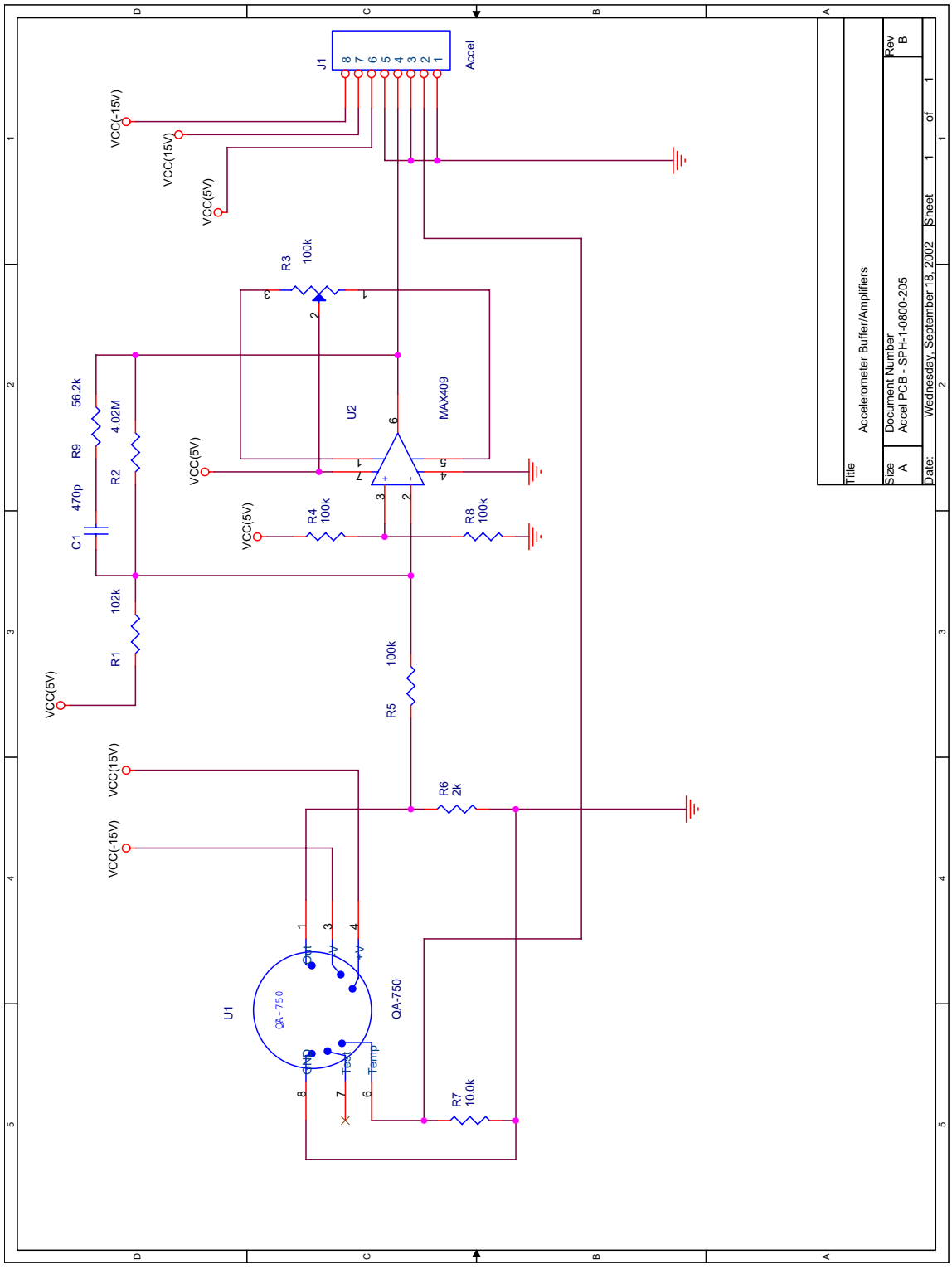

Power Board Connectors
Document Number
MB1: Metrology - SPH-1-0800-212
Rev
A
Size
A
Date:
Thursday, September 19, 2002
Sheet
8
of
9
VCC(5V)
VCC(3.3V)
VCC(15V)
VCC(-15V)
RD
BK
BU
YL
OR
J20
PWR
THR1
THR2
THR3
THR4
THR5
THR6
THR7
THR8
THR9
THR10
THR11
THR12
/ENABLE
LED-Enable
/BatLow
WDOG
/RESET
J21
Data to PWR

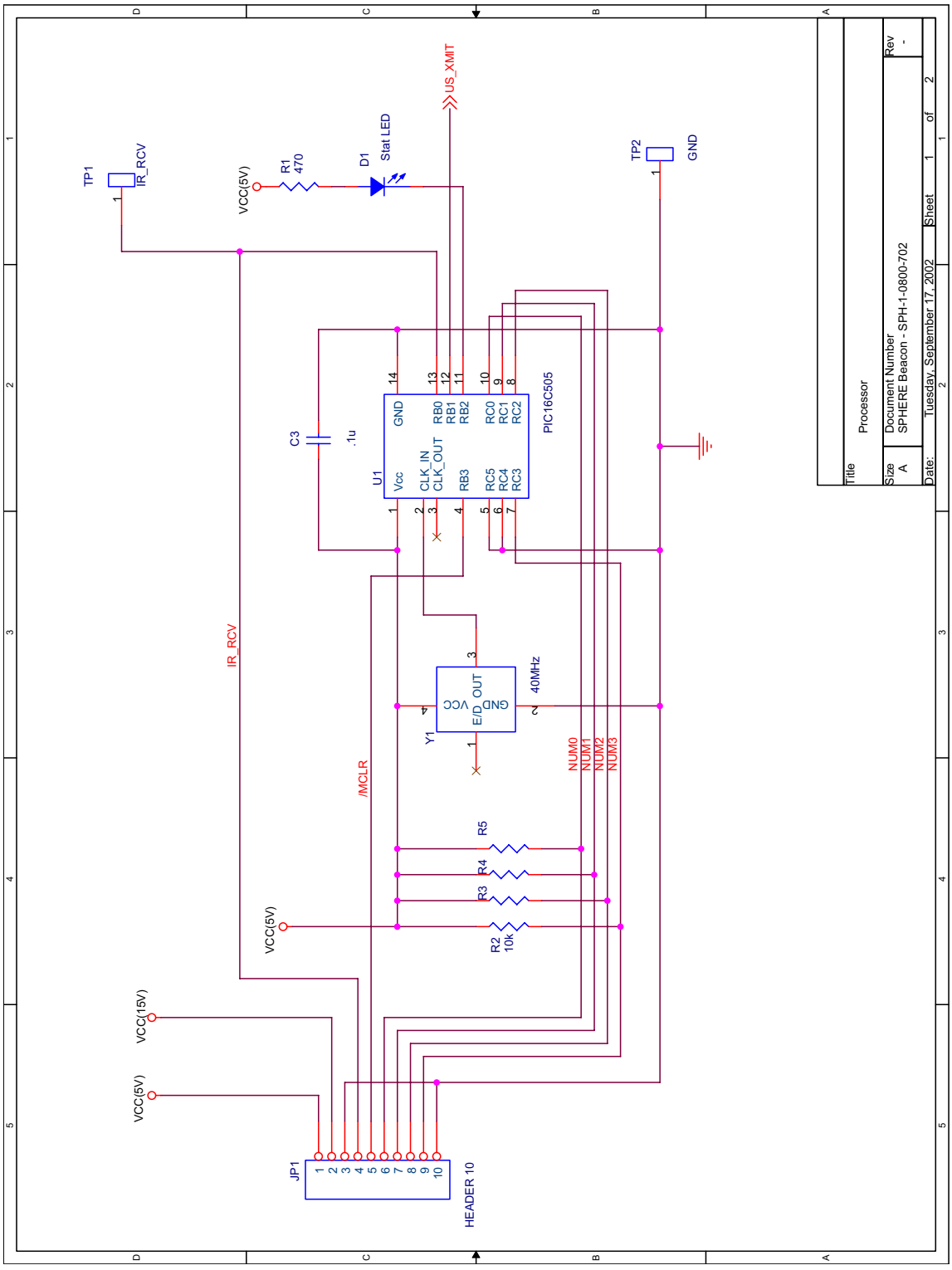

X-Accel-Amp
X-Accel-Temp
R16 100k
U9
MAX409
R33 56.2k
R13 4.02M
C49 470p
R12 102k
R14 100k
R15 100k
R17 100k
R18 2k
R19 10.0k
VCC(5V)
VCC(15V)
VCC(-15V)
U10
QA-750
Out
-V
+V
GND
Test
Temp
Title
Accelerometer Buffer/Amplifiers
Size A
Document Number MB1: Metrology - SPH-1-0800-212
Rev A
Date: Thursday, September 19, 2002
Sheet 9 of 9

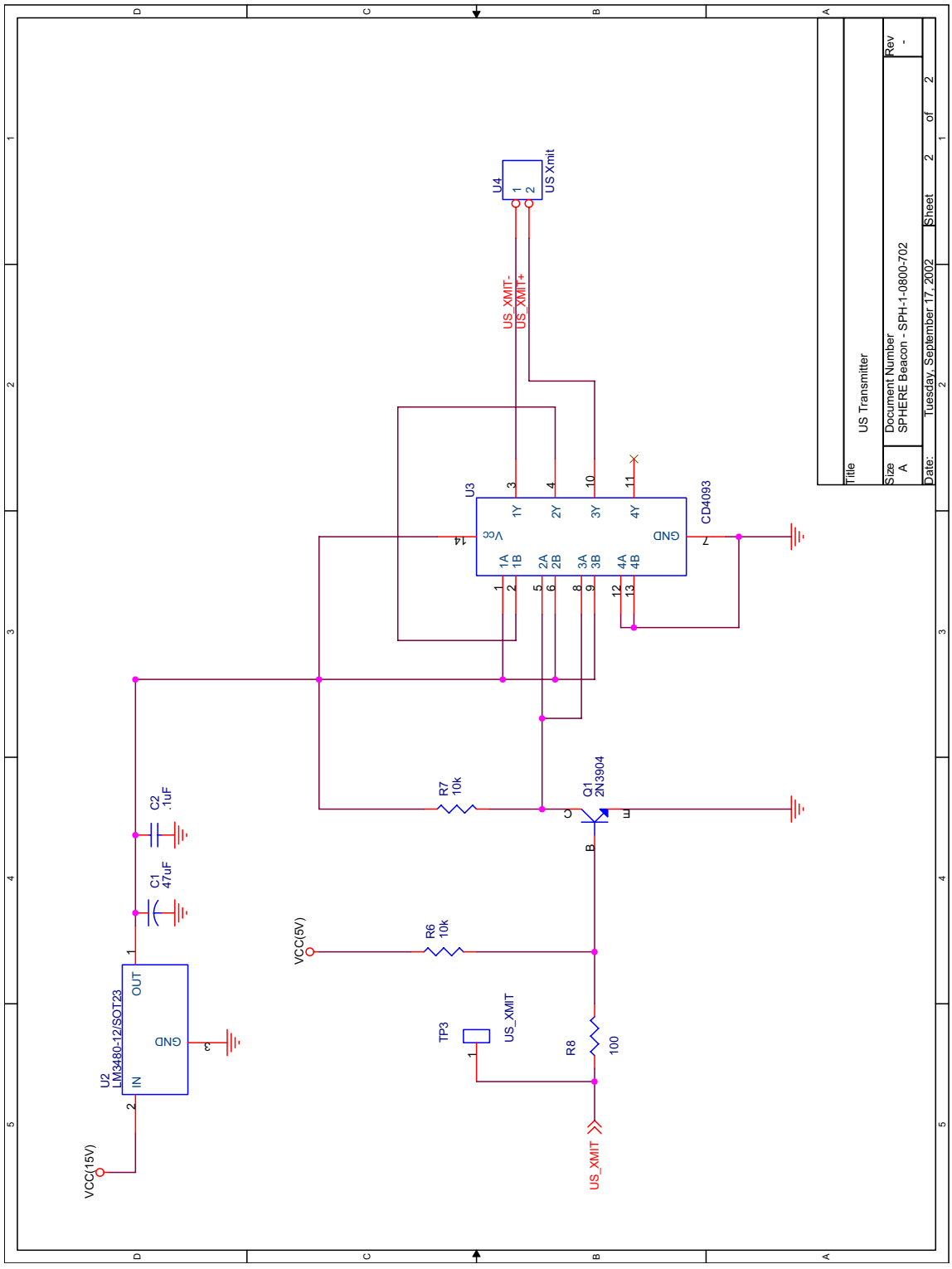

All the grounds in this diagram must be one single thick trace which connects the 470uF cap to the IRL520N
Title
Board Assembly
Size A
Document Number Metrology 6 - SPH-1-0800-101
Rev B
Date: Tuesday, February 18, 2003
Sheet 1 of 3
VCC(+5V)
R1 100 Ohm
C2 0.1uF
C3 47uF
D1 IR LED
D2 IR LED
R2 1 Ohm
R3 1 Ohm
D3
IRLML2502
R4 1M
IR XMIT
JP1
HEADER 10
C1 4.7uF
bypass
US RCV 1
US RCV 2
US Signal
US Circuit
US-1
US-2
VCC(+2.5V)
R5 22K
R6 22K
C4 4.7uF
C13 0.1uF
C14 4.7uF
R27 47 Ohm
IR RCV
J1
IR PAD

Each metrology board includes two of these circuits:




J1
Accel
VCC(-15V)
VCC(15V)
VCC(5V)
R3
100k
R9
56.2k
R2
4.02M
C1
470p
U2
MAX409
R4
100k
R8
100k
R1
102k
R5
100k
R6
2k
R7
10.0k
U1
QA-750
Out
-V
+V
GND
Test
Temp
Title
Accelerometer Buffer/Amplifiers
Size
A
Document Number
Accel PCB - SPH-1-0800-205
Rev
B
Date:
Wednesday, September 18, 2002
Sheet
1
of
1


TP1
IR_RCV
VCC(5V)
R1
470
D1
Stat LED
US_XMIT
TP2
GND
C3
.1u
U1
PIC16C505
GND
Vcc
CLK_IN
CLK_OUT
RB0
RB1
RB2
RB3
RC0
RC1
RC2
RC3
IR_RCV
/MCLR
Y1
GND
VCC
E/D
OUT
40MHz
NUM0
NUM1
NUM2
NUM3
R5
R4
R3
R2
10k
VCC(5V)
VCC(15V)
VCC(5V)
JP1
HEADER 10
Title
Processor
Size
A
Document Number
SPHERE Beacon - SPH-1-0800-702
Rev
-
Date:
Tuesday, September 17, 2002
Sheet
1
of
2


U4
US Xmit
US_XMIT-
US_XMIT+
U3
CD4093
1Y
2Y
3Y
4Y
1A
1B
2A
2B
3A
3B
4A
4B
Vcc
GND
R7
10k
Q1
2N3904
C2
.1uF
C1
47uF
U2
LM3480-12/SOT23
OUT
GND
IN
VCC(15V)
VCC(5V)
R6
10k
TP3
US_XMIT
R8
100
US_XMIT
Title
US Transmitter
Size
A
Document Number
SPHERE Beacon - SPH-1-0800-702
Rev
-
Date:
Tuesday, September 17, 2002
Sheet
2
of
2

F.1.5 Communications

Design Drivers

- Two wireless communications channels
  - Satellite to Laptop (STL) - Telemetry and Commands
  - Satellite to Satellite (STS) - Control and Commands
- Support at least three satellites
- Should be expandable
- Accommodate a minimum volume of 6’ x 6’ x 6’
- Highest data rate possible
- Low power

Functional Block Diagrams

Figure F.11 presents the functional block diagram of the communications sub-system. The major elements of the communications sub-system are three PIC processors that translate TI Commport signals into standard 8-bit (plus start and stop bit) UART serial data and two DR200x modules (one each for Satellite-to-Laptop {STL} and for Satellite-to-Satellite {STS} communications) which convert the 8-bit UART data into 14-bit bit-balanced words to minimize errors during wireless transmissions. The two elements are described below.

DR200x Wireless Boards

The DR2000 development kit is a COTS product available from RFM Monolithics in. The kit utilizes an ARM DSP to manage data for wireless transmissions. The ARM performs four functions:

- Creates 12-bit bit-balanced words for every byte to be transmitted. Bit balanced words contain the same number of ones and zeros to reduce the error rate in wireless transmission.
- Manages packets of a pre-set size. The DR200x can be configured to send fixed sized packets immediately once the fixed number of bytes are received; alternatively, it will transmit a packet if there is a pause longer than 2ms between bytes.

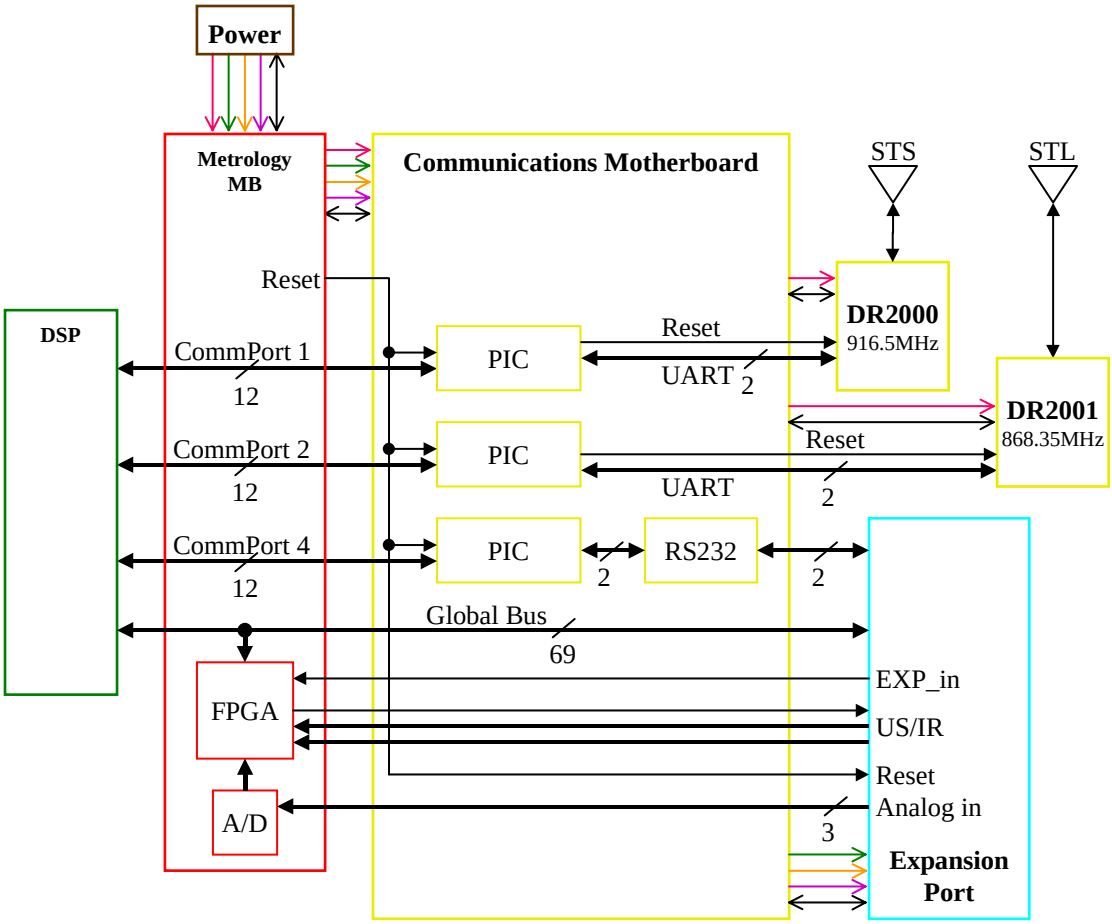


Figure F.11 Communications sub-system functional block diagram

- Adds a start header to all transmissions which allows the crystals in the receiving end to resonate at the correct frequency before the actual data arrives.
- Allows identification of each module individually, so that data can be directed to a specific DR200x board.

The basic features of the DR200x boards are listed in Table F.10. Table F.11 describes the signals of the DR2000x.

To improve the bandwidth of the system, though, SPHERES uses custom firmware. Therefore, while [RFM, URL] provides an overview of the hardware used in the board, Appendix H should be consulted to understand the operations of firmware. Further, to

TABLE F.10 DR200x specifications

	DR2000	DR2001
Frequency	916.5MHz	868.35MHz
Maximum wireless data rate	115.2kbps	
Implemented wireless data rate	56.6kbps	
UART data rate	115.2kbps	
Buffer	64 bytes	
RF Mode	ASK	
Available Addresses	1-255 (0x01-0xFF)	
Broadcast mode	Yes (to address = 0x00)	
Packet size	1-255 (0x01-0xFF)	
Input Voltage	3.1V-3.6V	
Power	<050mW	

TABLE F.11 DR200x signals descriptions

Signal	Type	Description
Vcc(+3.3V)	Pwr	+3.3V power
GND	Pwr	Common ground
RX	Out	Serial data receive line
TX	In	Serial data transmit line
/RST	In	Reset

minimize power consumption, the DR200x board used inside the satellites have been modified by removing the power regulation circuit (because the satellite power system provides regulated 3.3V) and the RS232 level converter, since the boards can connect directly through TTL to the PIC processors.

Communications Interface Board

The communications interface board hosts the PIC processors which translate the DR200x serial data to the TI commport standard. The DR200x utilizes a standard UART signal at 115.2kbps; the selected PIC (16C66) contains a serial port capable of handling up to 1.25Mbps communications with support via special registers. The implemented firmware

provides 96-byte input and output buffers. The TI commport standard is a parallel bi-directional data bus with token handshaking. The data consists of 32-bit words split into four bytes. Four control lines are used to pass the token (REQ and ACK) between the two units (in this case the DSP and the PIC) and to indicate that data is available (STRB) and has been read (RDY). Chapter 8 of [TI, SPRU159A] describes the operations of the communications ports in full.

Table F.12 describes the input and output signals of the communications motherboard.

TABLE F.12 Communications motherboard signals description

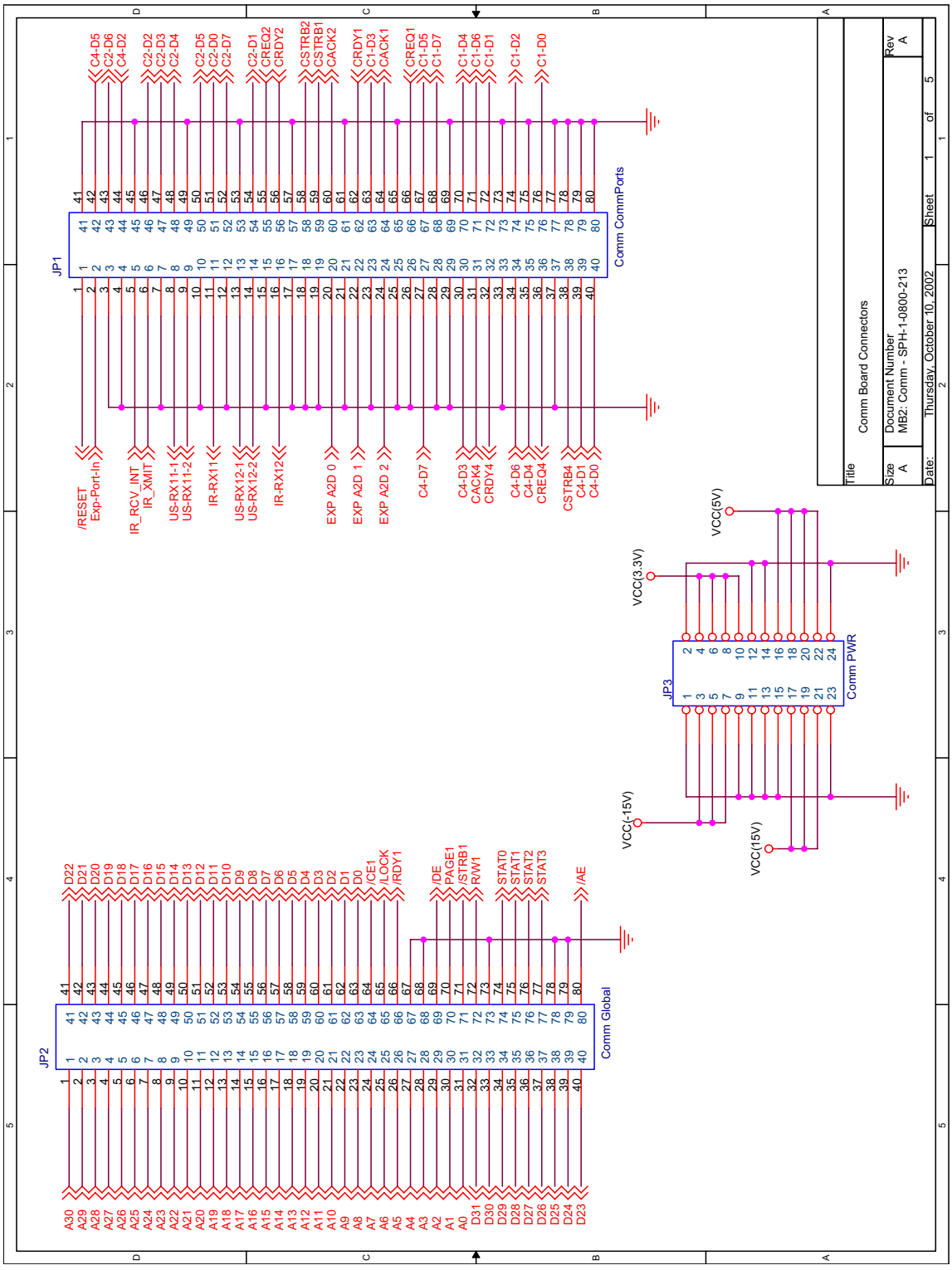
Section	Signal	Type	Description
To/From Metrology Mother-board	Vcc(+5V)	Pwr	+5V power
	Vcc(+3.3V)	Pwr	+3.3V power
	Vcc(+15V)	Pwr	+15V power
	Vcc(-15V)	Pwr	-15V power
	GND	Pwr	Common ground
	A0-A30	In	Global bus address lines (expansion port)
	D0-D31	I/O	Global bus data lines (expansion port)
	RDY1	I/O	Global bus ready (expansion port)
	PAGE1	I/O	Global bus page select (expansion port)
	STRB1	In	Global bus strobe (expansion port)
	R/W1	I/O	Global bus read/write (expansion port)
	/RESET	In	Reset line
	/Exp_port_in	Out	Expansion port item indicator
	IR_XMIT	In	IR transmit command
	US-RX[11-12]-[1-2]	Out	Input ultrasound signals from the expansion port board
	IR-RX[11-12]	Out	Input infrared signals from the expansion port board
	EXP A2D [0-2]	Out	Analog signals from expansion port
	C[1,2,4] D[0-7]	I/O	Commport data lines
	CACK[1,2,4]	I/O	Commport acknowledge signal
	CRDY[1,2,4]	I/O	Commport ready signal

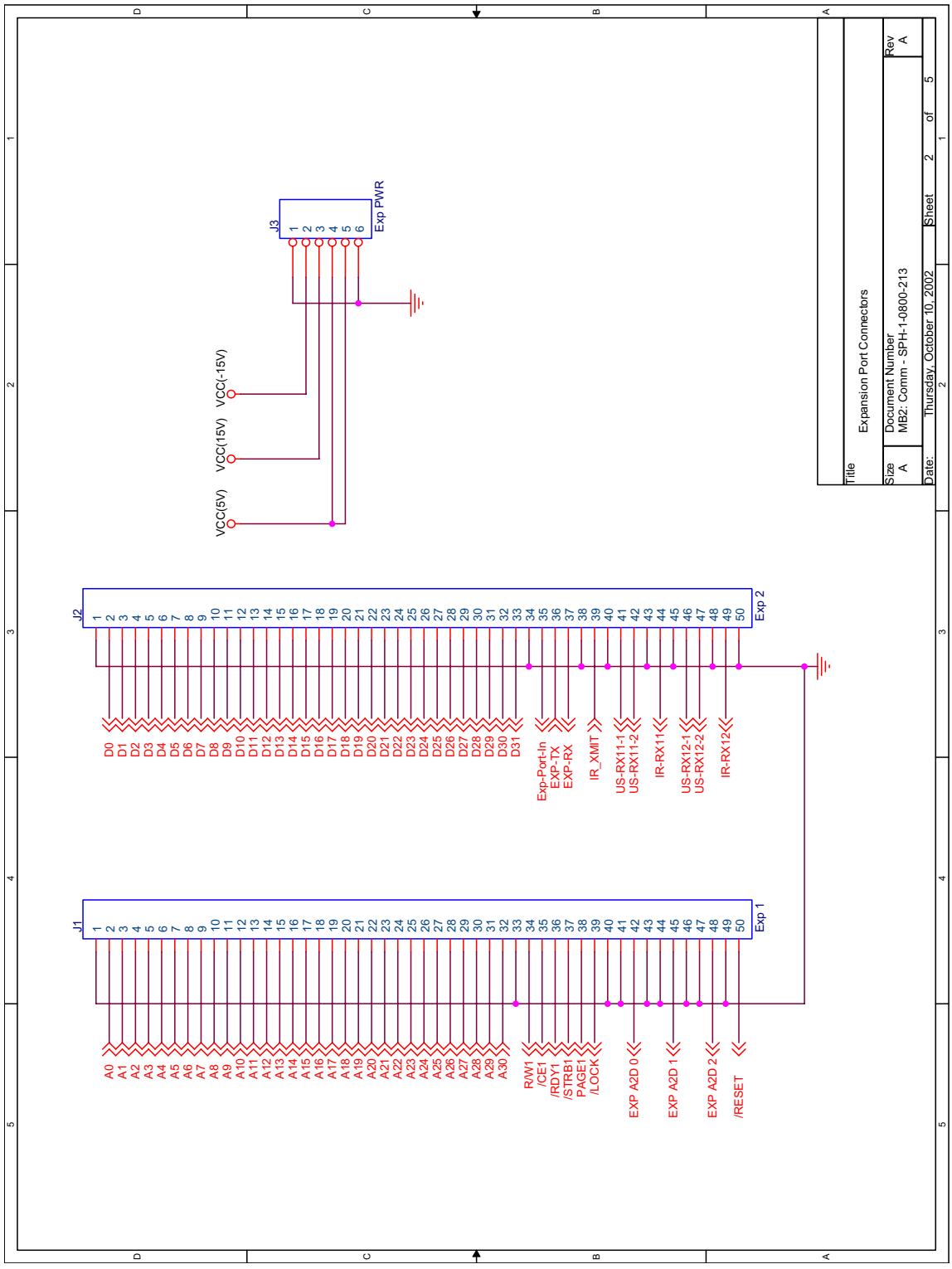
TABLE F.12 Communications motherboard signals description

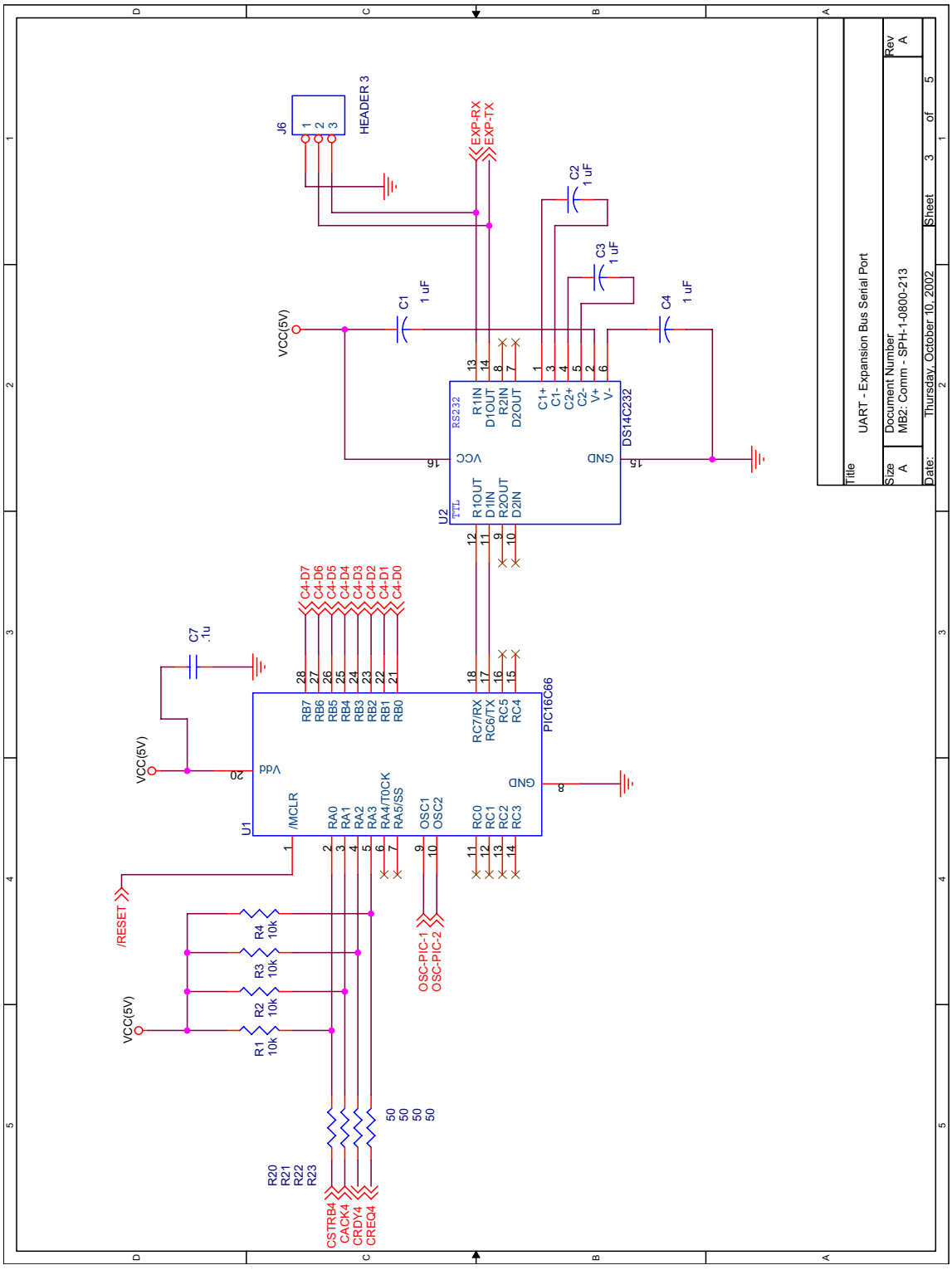
Section	Signal	Type	Description
Met. MB (cont)	CREQ[1,2,4]	I/O	Commport request signal
	STRB[1,2,4]	I/O	Commport strobe
Expansion Port Connector	Vcc(+5V)	Pwr	+5V power
	Vcc(+15V)	Pwr	+15V power
	Vcc(-15V)	Pwr	-15V power
	GND	Pwr	Common ground
	A0-A30	Out	Global bus address lines
	D0-D31	I/O	Global bus data lines
	RDY1	I/O	Global bus ready
	PAGE1	I/O	Global bus page select
	STRB1	Out	Global bus strobe
	R/W1	I/O	Global bus read/write
	/RESET	In	Reset line
	/Exp_port_in	In	High when an expansion port selects to bypass the satellite US/IR metrology boards
	IR_XMIT	Out	IR transmit command
	US-RX[11-12]-[1-2]	In	Input ultrasound signals
	IR-RX[11-12]	In	Input infrared signals
	EXP A2D [0-2]	In	Input analog signals
	EXP RX	In	Serial data receive (RS232)
	EXP TX	Out	Serial data transmit (RS232)
Wired Serial Connector	EXP RX	In	Serial data receive (RS232)
	EXP TX	Out	Serial data transmit (RS232)
	GND	Pwr	Common Ground
DR200x (2x)	Vcc(+3.3V)	Pwr	+3.3V power
	GND	Pwr	Common ground
	RX	Out	Serial data receive line
	TX	In	Serial data transmit line
	/RST	In	Reset

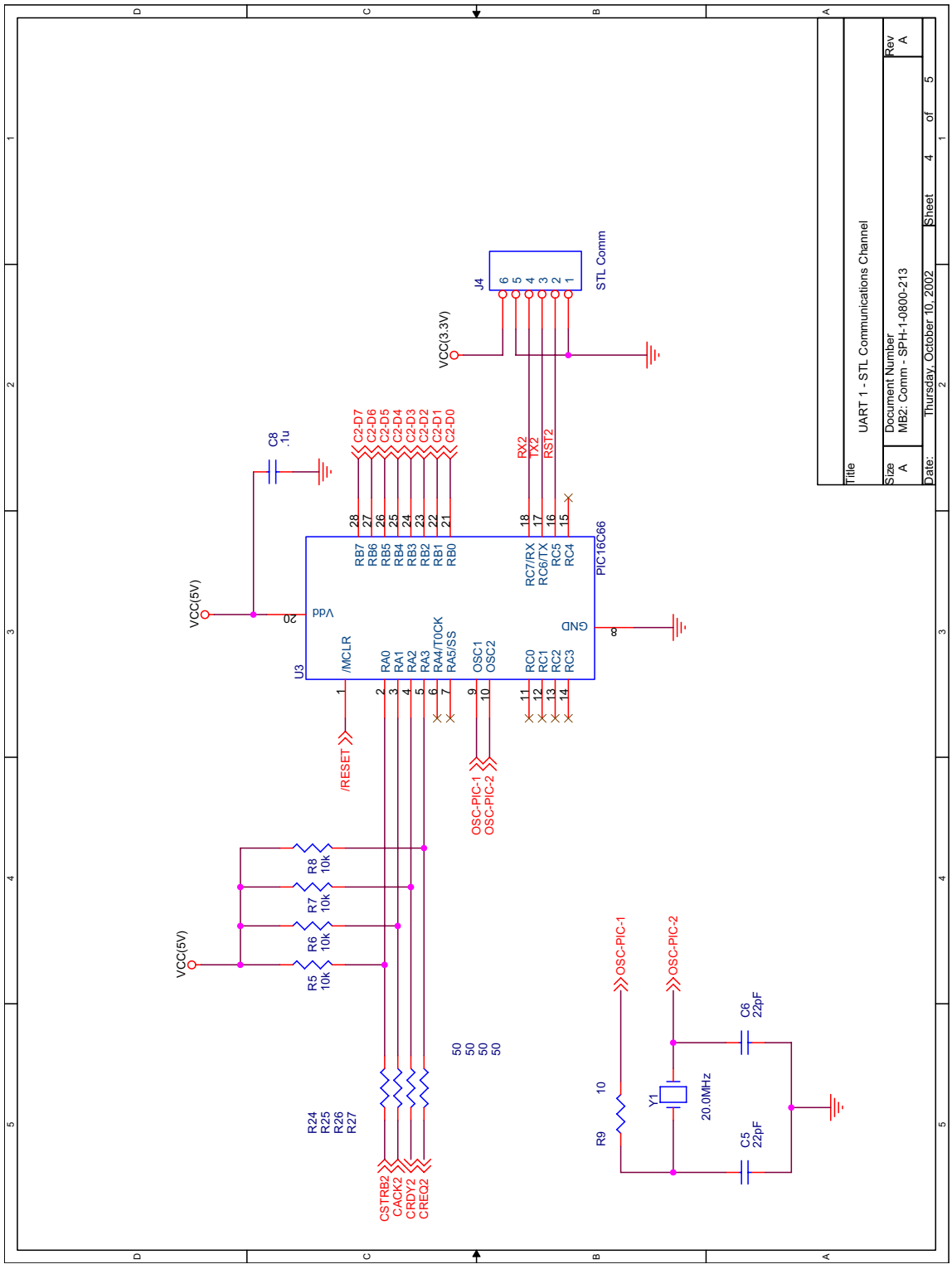
Schematics

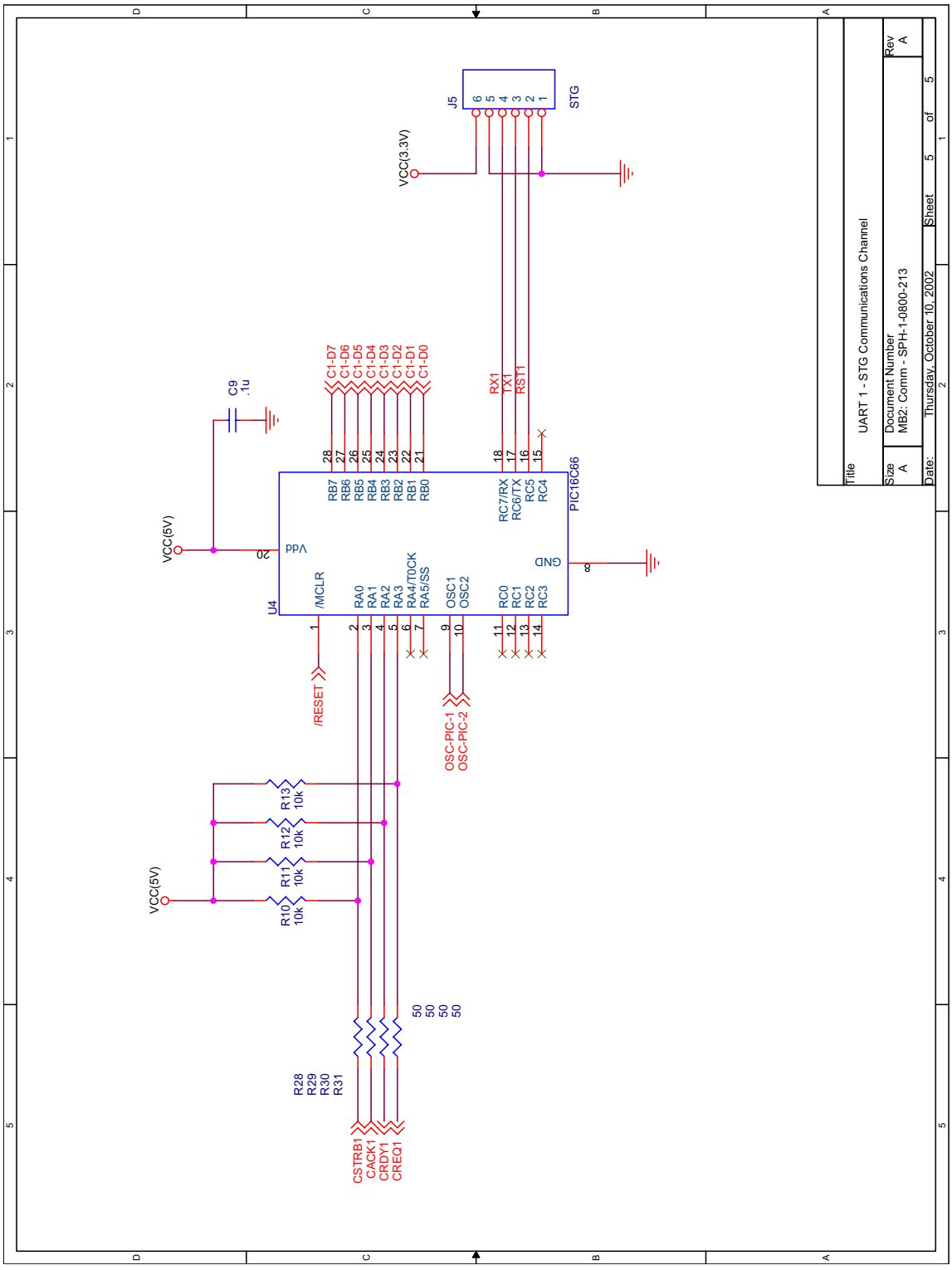
The schematics of the DR200x boards are available in [RFM, URL]. The schematics of the SPHERES communications motherboard are presented next.


JP1
Comm CommPorts
/RESET
Exp-Port-In
IR_RCV_INT
IR_XMIT
US-RX11-1
US-RX11-2
IR-RX11
US-RX12-1
US-RX12-2
IR-RX12
EXP A2D 0
EXP A2D 1
EXP A2D 2
C4-D7
C4-D3
CACK4
CRDY4
C4-D6
C4-D4
CREQ4
CSTRB4
C4-D1
C4-D0
C4-D5
C2-D6
C4-D2
C2-D2
C2-D3
C2-D4
C2-D5
C2-D0
C2-D7
C2-D1
CREQ2
CRDY2
CSTRB2
CSTRB1
CACK2
CRDY1
C1-D3
CACK1
CREQ1
C1-D5
C1-D7
C1-D4
C1-D6
C1-D1
C1-D2
C1-D0
JP2
Comm Global
A30
A29
A28
A27
A26
A25
A24
A23
A22
A21
A20
A19
A18
A17
A16
A15
A14
A13
A12
A11
A10
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
D31
D30
D29
D28
D27
D26
D25
D24
D23
D22
D21
D20
D19
D18
D17
D16
D15
D14
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
/CE1
/LOCK
/RDY1
/DE
PAGE1
/STRB1
R/W1
STAT0
STAT1
STAT2
STAT3
/AE
JP3
Comm PWR
VCC(3.3V)
VCC(5V)
VCC(-15V)
VCC(15V)
Title
Comm Board Connectors
Size
A
Document Number
MB2: Comm - SPH-1-0800-213
Rev
A
Date:
Thursday, October 10, 2002
Sheet
1
of
5


Title
Expansion Port Connectors
Size
A
Document Number
MB2: Comm - SPH-1-0800-213
Rev
A
Date:
Thursday, October 10, 2002
Sheet
2
of
5
J3
Exp PWR
VCC(-15V)
VCC(15V)
VCC(5V)
J2
Exp 2
D0
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
D14
D15
D16
D17
D18
D19
D20
D21
D22
D23
D24
D25
D26
D27
D28
D29
D30
D31
Exp-Port-In
EXP-TX
EXP-RX
IR_XMIT
US-RX11-1
US-RX11-2
IR-RX11
US-RX12-1
US-RX12-2
IR-RX12
J1
Exp 1
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
A16
A17
A18
A19
A20
A21
A22
A23
A24
A25
A26
A27
A28
A29
A30
R/W1
/CE1
/RDY1
/STRB1
PAGE/
/LOCK
EXP A2D 0
EXP A2D 1
EXP A2D 2
/RESET


HEADER 3
J6
EXP-RX
EXP-TX
C2
1 uF
C3
1 uF
C1
1 uF
C4
1 uF
VCC(5V)
U2
RS232
TTL
R1IN
D1OUT
R2IN
D2OUT
C1+
C1-
C2+
C2-
V+
V-
VCC
GND
DS14C232
R1OUT
D1IN
R2OUT
D2IN
C4-D7
C4-D6
C4-D5
C4-D4
C4-D3
C4-D2
C4-D1
C4-D0
C7
.1u
U1
/MCLR
Vdd
Vss
RB7
RB6
RB5
RB4
RB3
RB2
RB1
RB0
RC7/RX
RC6/TX
RC5
RC4
PIC16C66
RA0
RA1
RA2
RA3
RA4/T0CK
RA5/SS
OSC1
OSC2
RC0
RC1
RC2
RC3
/RESET
R4
10k
R3
10k
R2
10k
R1
10k
OSC-PIC-1
OSC-PIC-2
R20
R21
R22
R23
50
50
50
50
CSTRB4
CACK4
CRDY4
CREQ4
Title
UART - Expansion Bus Serial Port
Size
A
Document Number
MB2: Comm - SPH-1-0800-213
Rev
A
Date:
Thursday, October 10, 2002
Sheet
3
of
5


Title
UART 1 - STL Communications Channel
Size
A
Document Number
MB2: Comm - SPH-1-0800-213
Rev
A
Date:
Thursday, October 10, 2002
Sheet
4
of
5
STL Comm
J4
VCC(3.3V)
PIC16C66
U3
VCC(5V)
C8
.1u
/MCLR
/RESET
OSC-PIC-1
OSC-PIC-2
C6
22pF
C5
22pF
Y1
20.0MHz
R9
10
R5
10k
R6
10k
R7
10k
R8
10k
R24
R25
R26
R27
50
CSTRB2
CACK2
CRDY2
CREQ2
C2-D7
C2-D6
C2-D5
C2-D4
C2-D3
C2-D2
C2-D1
C2-D0
RX2
TX2
RST2


J5
STG
VCC(3.3V)
C9
.1u
VCC(5V)
U4
PIC16C66
/MCLR
VDD
VSS
RB7
RB6
RB5
RB4
RB3
RB2
RB1
RB0
RC7/RX
RC6/TX
RC5
RC4
RC0
RC1
RC2
RC3
OSC1
OSC2
RA0
RA1
RA2
RA3
RA4/T0CK
RA5/SS
C1-D7
C1-D6
C1-D5
C1-D4
C1-D3
C1-D2
C1-D1
C1-D0
RX1
TX1
RST1
/RESET
OSC-PIC-1
OSC-PIC-2
R13 10k
R12 10k
R11 10k
R10 10k
R28 50
R29 50
R30 50
R31 50
CSTRB1
CACK1
CRDY1
CREQ1
Title
UART 1 - STG Communications Channel
Size A
Document Number MB2: Comm - SPH-1-0800-213
Rev A
Date: Thursday, October 10, 2002
Sheet 5 of 5

F.1.6 Expansion Port

Design Drivers

- Provide digital interface for future expansions

Functional Block Diagrams

The broad requirements in the definition of the expansion port resulted in a design which provides a simple but limited serial line capable of up to 1.25Mbps data rates as well as the very flexible but complex global bus. Figure F.12 presents the functional block diagram for the Expansion Port.

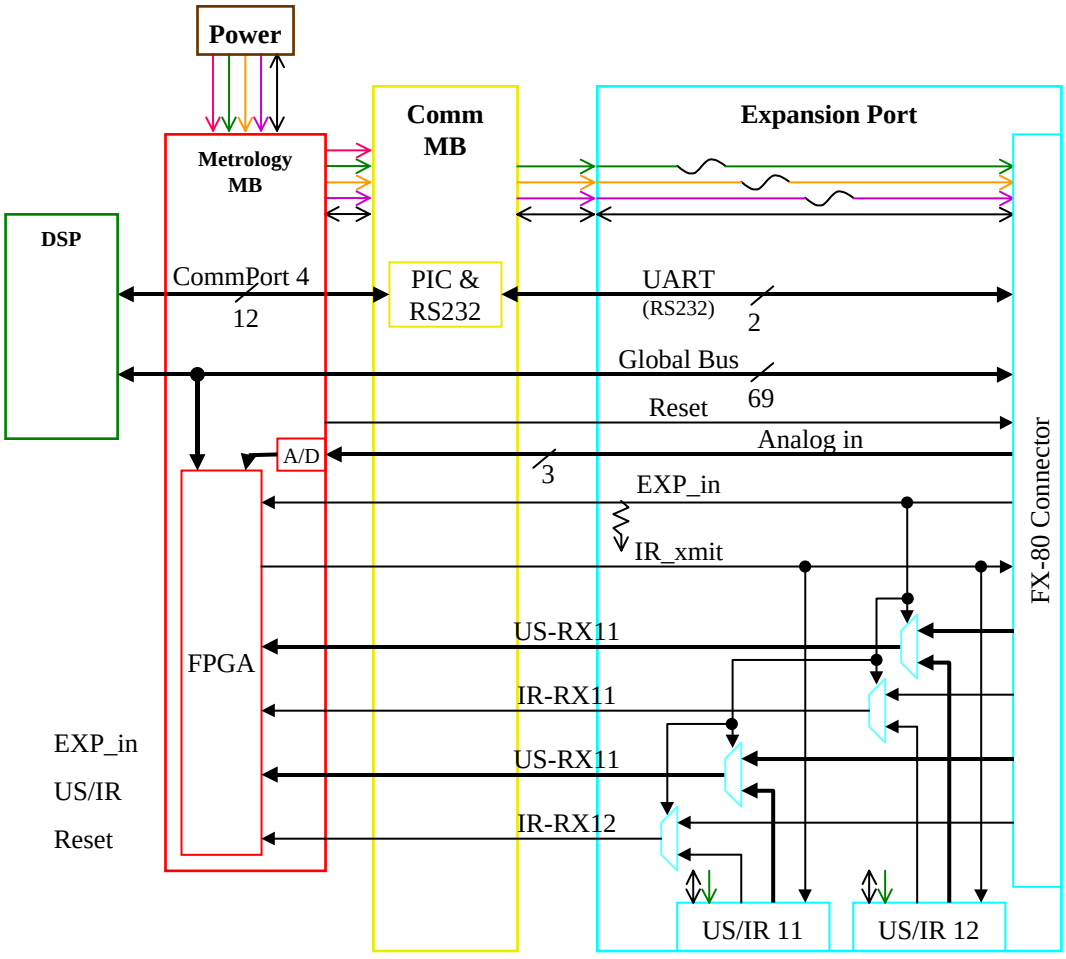


Figure F.12 Expansion port functional block diagram

Apart from providing the required digital data lines, the Expansion Port also supports three other functions:

- Provides power to expansion items via +5V, +15V, and -15V power lines. It protects these lines with 0.5A self-resetable fuses.
- Because three analog lines were available from the basic metrology design, the expansion port makes these lines available to expansion items.
- Allows an expansion item to *bypass* the internal US/IR metrology boards located on the expansion port face (+X face). This allows an expansion item to replace the functionality of those boards if the expansion item covers the sensors. The expansion board uses high-speed multiplexers so that the signals received by the FPGA are equivalent to any other US/IR signals. The EXP_in line allows the DSP to account for the new physical locations (which must be programmed) of the US/IR boards when the signals have been bypassed.

Table F.13 describes the inputs and outputs of the Expansion Port.

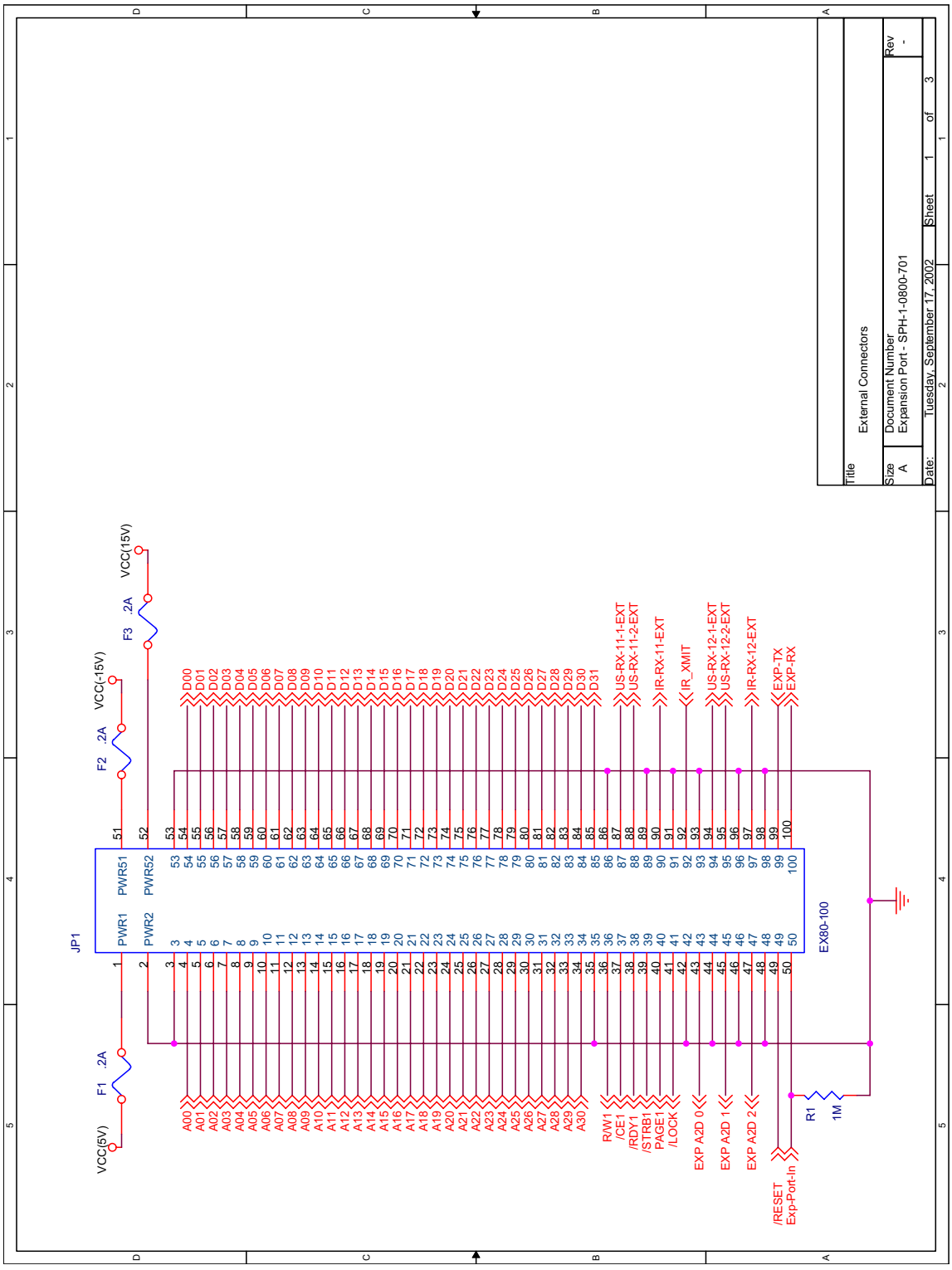
TABLE F.13 Expansion port signals description

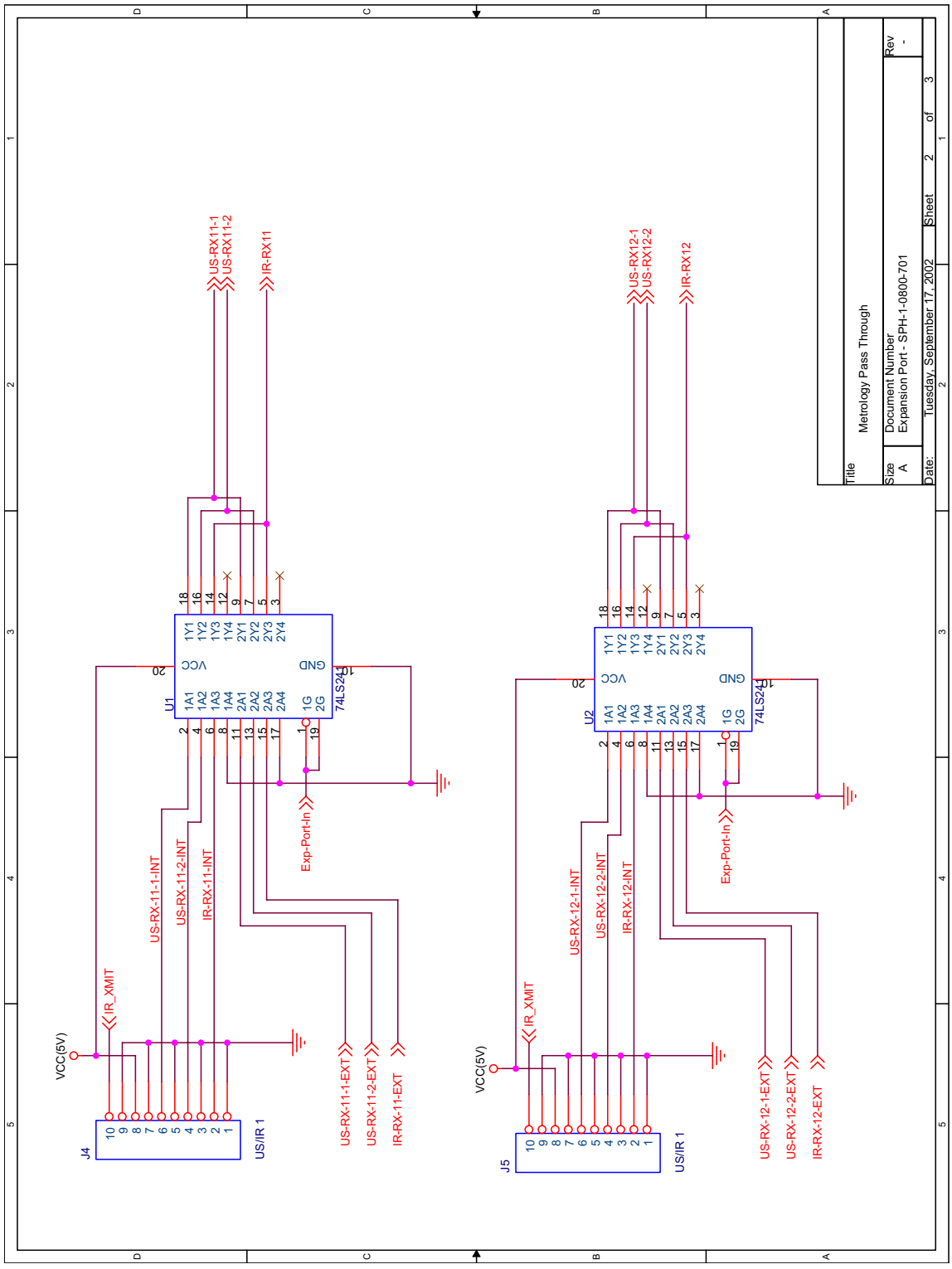
Section	Signal	Type	Description
To/From Comm. Mother-board	*See Table F.12*		
Expansion Port Con-nector	Vcc(+5V)	Pwr	+5V power
	Vcc(+15V)	Pwr	+15V power
	Vcc(-15V)	Pwr	-15V power
	GND	Pwr	Common ground
	A0-A30	Out	Global bus address lines
	D0-D31	I/O	Global bus data lines
	RDY1	I/O	Global bus ready
	PAGE1	I/O	Global bus page select
	STRB1	Out	Global bus strobe
	R/W1	I/O	Global bus read/write
	/RESET	In	Reset line

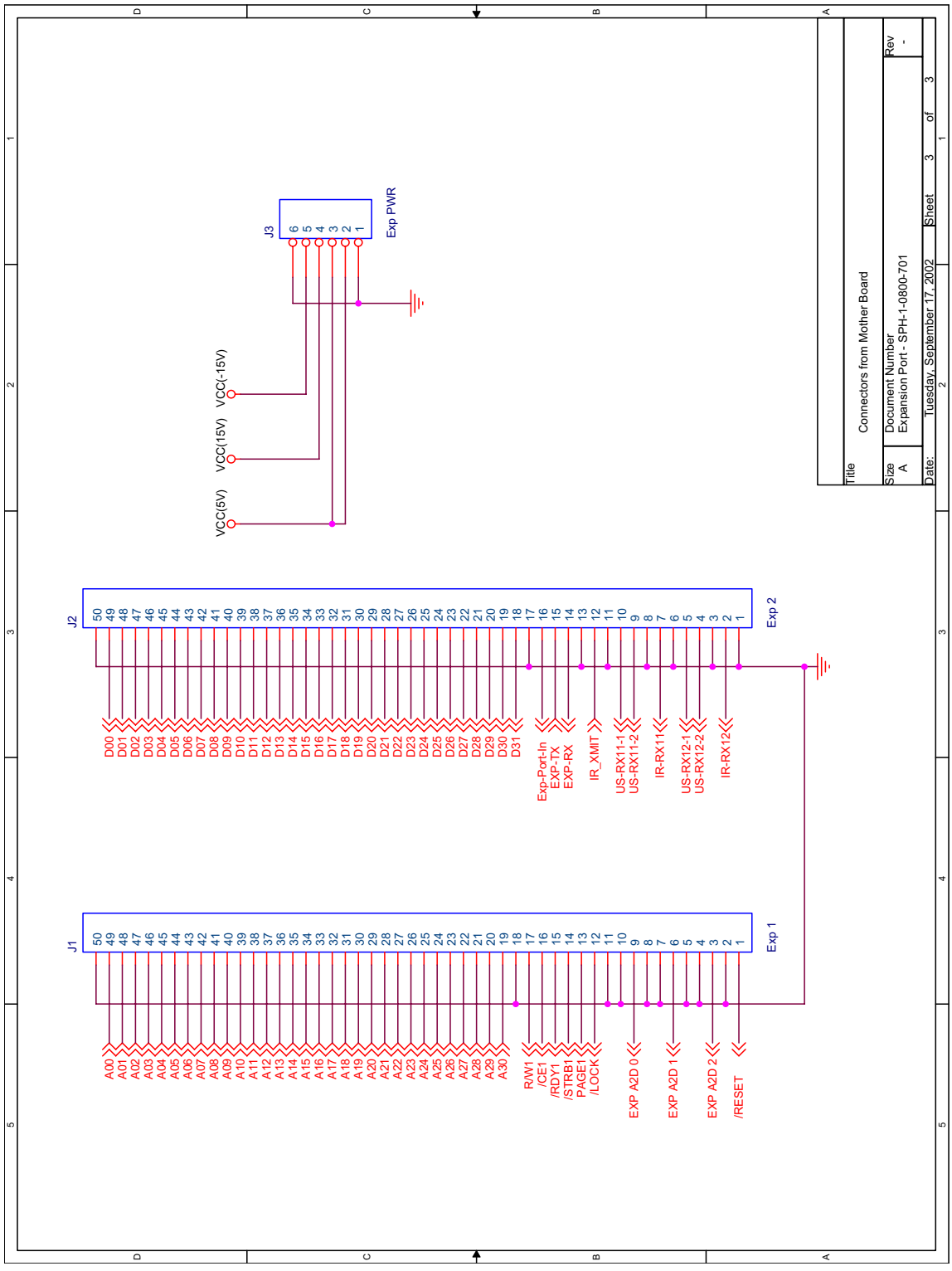
TABLE F.13 Expansion port signals description

Section	Signal	Type	Description
Expansion Port (cont)	/Exp_port_in	In	High when an expansion port selects to bypass the satellite US/IR metrology boards
	IR_XMIT	Out	IR transmit command
	US-RX[11-12]-[1-2]-EXT	In	External sensor input ultrasound signals
	IR-RX[11-12]-EXT	In	External sensor input infrared signals
	EXP A2D [0-2]	In	Input analog signals
	EXP RX	In	Serial data receive (RS232)
	EXP TX	Out	Serial data transmit (RS232)
Metrology Pass-through (2x)	Vcc(+5V)	Pwr	+5V power
	GND	Pwr	Common ground
	IR_XMIT	Out	IR transmit command
	US-RX[11-12]-[1-2]-INT	In	Internal sensor input ultrasound signals
	IR-RX[11-12]-INT	In	Internal sensor input infrared signals

Schematics


JP1
EX80-100
PWR1
PWR2
PWR51
PWR52
F1 .2A
F2 .2A
F3 .2A
VCC(5V)
VCC(-15V)
VCC(15V)
R1
1M
A00
A01
A02
A03
A04
A05
A06
A07
A08
A09
A10
A11
A12
A13
A14
A15
A16
A17
A18
A19
A20
A21
A22
A23
A24
A25
A26
A27
A28
A29
A30
D00
D01
D02
D03
D04
D05
D06
D07
D08
D09
D10
D11
D12
D13
D14
D15
D16
D17
D18
D19
D20
D21
D22
D23
D24
D25
D26
D27
D28
D29
D30
D31
R/W1
/CE1
/RDY1
/STRB1
PAGE1
/LOCK
EXP A2D 0
EXP A2D 1
EXP A2D 2
/RESET
Exp-Port-In
US-RX-11-1-EXT
US-RX-11-2-EXT
IR-RX-11-EXT
IR_XMIT
US-RX-12-1-EXT
US-RX-12-2-EXT
IR-RX-12-EXT
EXP-TX
EXP-RX
Title
External Connectors
Size
A
Document Number
Expansion Port - SPH-1-0800-701
Rev
-
Date:
Tuesday, September 17, 2002
Sheet
1
of
3


Title
Metrology Pass Through
Size
A
Document Number
Expansion Port - SPH-1-0800-701
Rev
-
Date:
Tuesday, September 17, 2002
Sheet
2
of
3
US-RX11-1
US-RX11-2
IR-RX11
U1
74LS241
VCC
GND
1A1
1A2
1A3
1A4
2A1
2A2
2A3
2A4
1G
2G
1Y1
1Y2
1Y3
1Y4
2Y1
2Y2
2Y3
2Y4
Exp-Port-In
US-RX-11-1-INT
US-RX-11-2-INT
IR-RX-11-INT
IR_XMIT
VCC(5V)
J4
US/IR 1
US-RX-11-1-EXT
US-RX-11-2-EXT
IR-RX-11-EXT
US-RX12-1
US-RX12-2
IR-RX12
U2
US-RX-12-1-INT
US-RX-12-2-INT
IR-RX-12-INT
J5
US-RX-12-1-EXT
US-RX-12-2-EXT
IR-RX-12-EXT


Connectors from Mother Board
Document Number
Expansion Port - SPH-1-0800-701
Rev
-
Size
A
Date: Tuesday, September 17, 2002
Sheet 3 of 3
J3
Exp PWR
VCC(-15V)
VCC(15V)
VCC(5V)
J2
Exp 2
D00 D01 D02 D03 D04 D05 D06 D07 D08 D09 D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D20 D21 D22 D23 D24 D25 D26 D27 D28 D29 D30 D31
Exp-Port-In
EXP-TX
EXP-RX
IR_XMIT
US-RX11-1
US-RX11-2
IR-RX11
US-RX12-1
US-RX12-2
IR-RX12
J1
Exp 1
A00 A01 A02 A03 A04 A05 A06 A07 A08 A09 A10 A11 A12 A13 A14 A15 A16 A17 A18 A19 A20 A21 A22 A23 A24 A25 A26 A27 A28 A29 A30
R/W1
/CE1
/RDY1
/STRB1
PAGE1
/LOCK
EXP A2D 0
EXP A2D 1
EXP A2D 2
/RESET

F.2 Laptop Communications

Design Drivers

- Interface with standard equipment available on the ISS SSC

Functional Block Diagrams

The laptop transceiver is a modified DR2001 (868.35MHz) development kit (the backup is a DR2000, 916.5MHz). The transceiver uses the custom firmware developed for SPHERES and does not use the power regulation circuit. The power regulation has been replaced with two diodes which step down the +5V voltage of a USB port of the SSC to approximately 3.6V, the maximum allowed by the DR200x without power circuitry.Figure F.13 presents the functional block diagram of the laptop communications. [RFM, URL] provides information on the hardware design, and Appendix H on the firmware and operations.

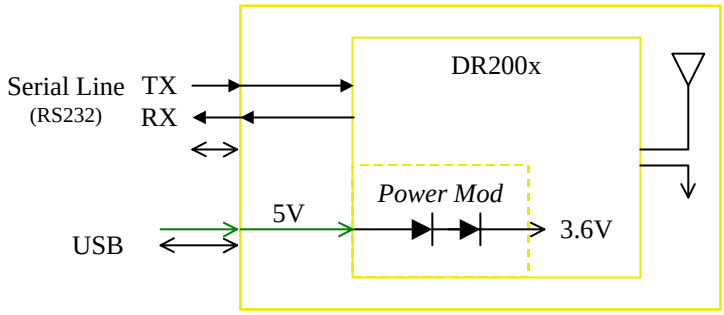


Figure F.13 Laptop communications functional block diagram

F.3 Metrology Beacons

Design Drivers

- Trigger on IR reception
- Transmit ultrasonic (US) pulse at
  - $\Delta t = \{(N-1)\cdot 20 + 5\}$ ms
  - N = beacon number (1-5)
- Provide selectable beacon number

- Battery operation
  - On/off switch
  - Low battery LED

Functional Block Diagrams

The metrology beacons operate on two AA batteries (alkaline aboard the ISS, which provide approximately 24 hours of operation, and rechargeable in ground-based facilities). The approximately 3V from the batteries are stepped-up to 5V to power a PIC microcontroller and 12V to drive the ultrasound transmitter. The PIC operates at 40MHz to create the pulses required for the ultrasound with a timing accuracy (as per the function presented in the design drivers) of 1μs (0.3mm error). The PIC creates a pulse with 16 oscillations at 40kHz. The driver circuitry effectively produces 24V pulses at the ultrasound transmitter by alternating the two leads of the transmitter between ground and +12V during the pulses (rather than using a 12V signal by holding one lead constant and only alternating the other lead). When there are no pulses a constant 12V differential exists between the leads, but the transmitter does not produce any ultrasound since it is a resonator which must be exited to its resonant frequency (40kHz). The beacon uses the same IR receiver/amplifier used in the satellite US/IR boards.

Two LED’s provide feedback on the status of the beacon. A green LED indicates the status of the beacon. A solid green indicated the beacon is powered on; a flashing green indicates the beacon is active (is transmitting ultrasound). An amber LED indicates a low-batter condition.

The functional block diagram of the metrology beacons is presented in Figure F.14.

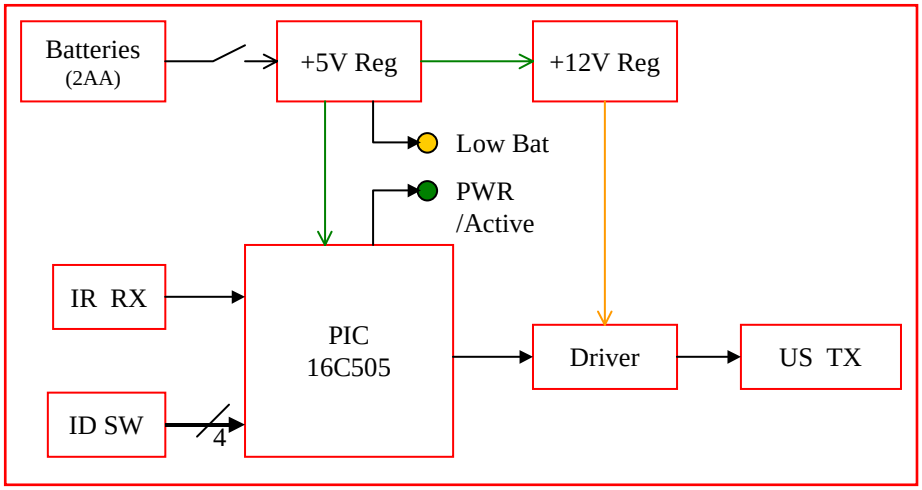
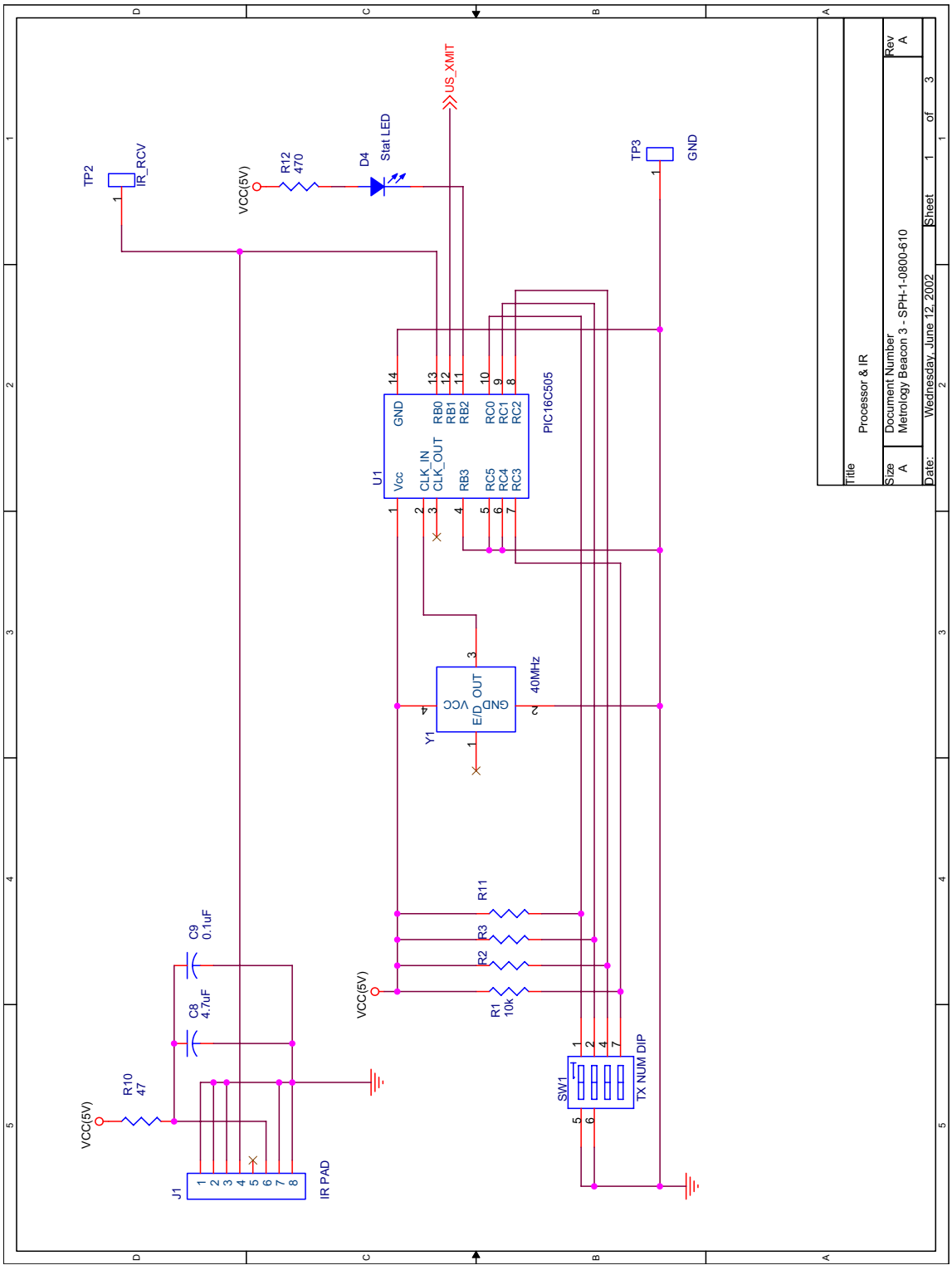
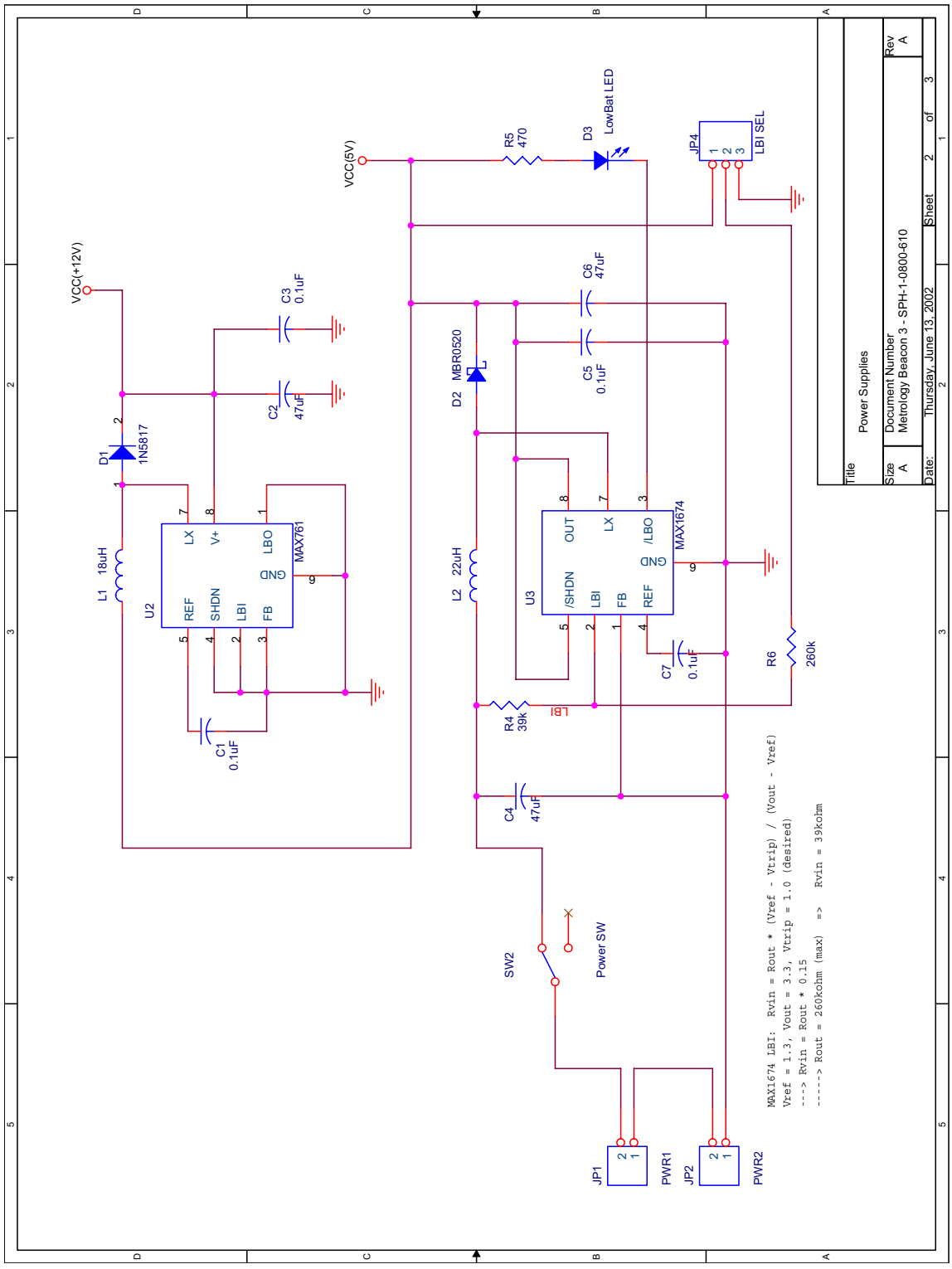
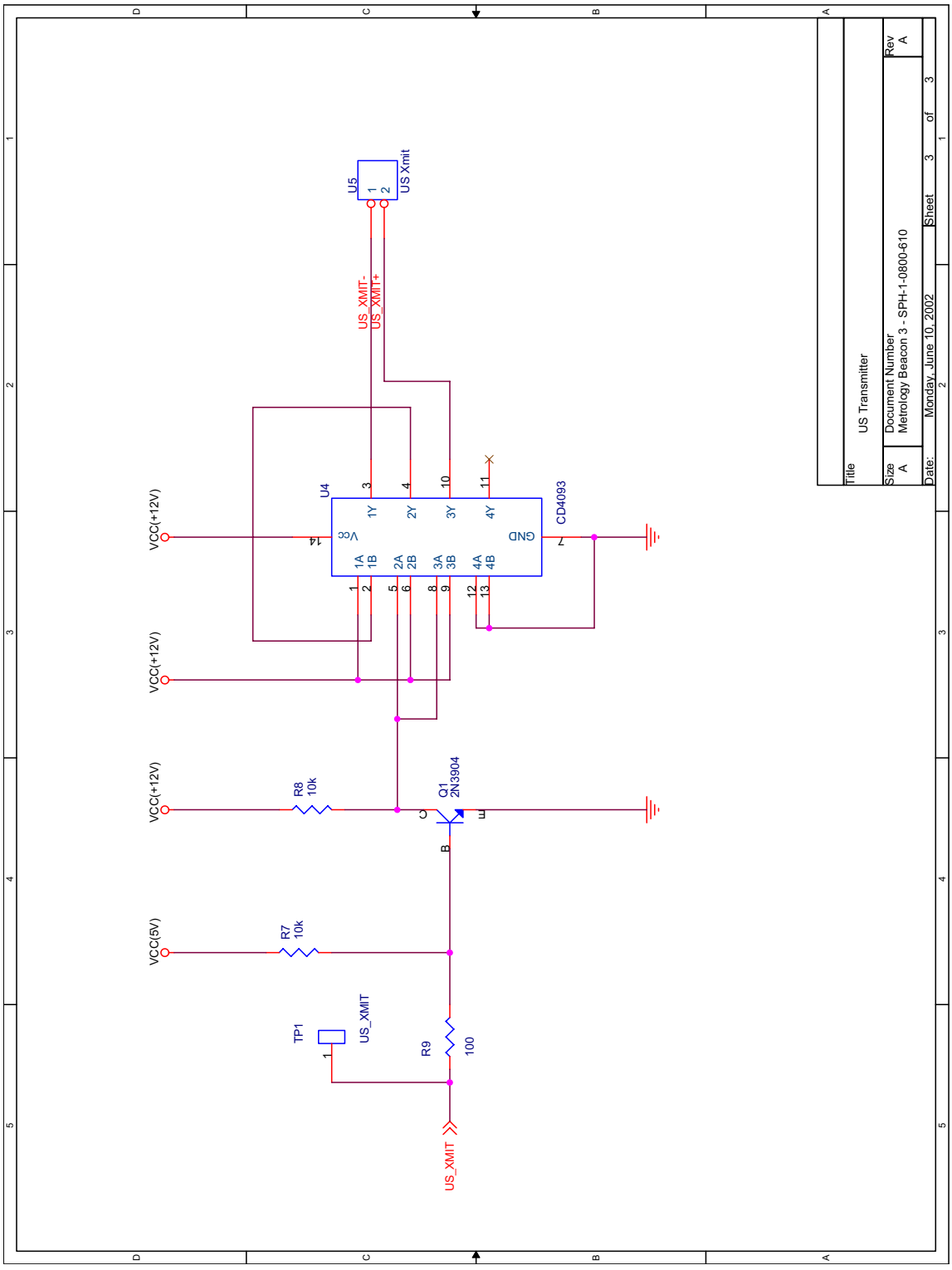


Figure F.14 Metrology beacon functional block diagram

Schematics


US_XMIT
TP2
IR_RCV
VCC(5V)
R12
470
D4
Stat LED
TP3
GND
U1
PIC16C505
GND
Vcc
CLK_IN
CLK_OUT
RB0
RB1
RB2
RB3
RC0
RC1
RC2
RC3
RC4
RC5
Y1
40MHz
E/D
OUT
R11
R3
R2
R1
10k
SW1
TX NUM DIP
C9
0.1uF
C8
4.7uF
R10
47
J1
IR PAD
Title
Processor & IR
Size
A
Document Number
Metrology Beacon 3 - SPH-1-0800-610
Rev
A
Date:
Wednesday, June 12, 2002
Sheet
1
of
3


VCC(+12V)
VCC(5V)
D1 1N5817
D2 MBR0520
D3 LowBat LED
R5 470
JP4 LBI SEL
C3 0.1uF
C2 47uF
C6 47uF
C5 0.1uF
L1 18uH
L2 22uH
U2 MAX761
U3 MAX1674
LX V+ LBO GND REF SHDN LBI FB
OUT /SHDN LX LBI FB /LBO GND REF
C1 0.1uF
C4 47uF
C7 0.1uF
R4 39k
R6 260k
LBI
SW2 Power SW
JP1 PWR1 JP2 PWR2
MAX1674 LBI: Rvin = Rout * (Vref - Vtrip) / (Vout - Vref)
Vref = 1.3, Vout = 3.3, Vtrip = 1.0 (desired)
---> Rvin = Rout * 0.15
-----> Rout = 260kohm (max) => Rvin = 39kohm
Title Power Supplies
Size A
Document Number Metrology Beacon 3 - SPH-1-0800-610
Rev A
Date: Thursday, June 13, 2002
Sheet 2 of 3


U5
US Xmit
US_XMIT-
US_XMIT+
U4
CD4093
VCC(+12V)
VCC(+12V)
VCC(+12V)
VCC(5V)
R8
10k
R7
10k
Q1
2N3904
TP1
US_XMIT
R9
100
US_XMIT
US Transmitter
Document Number
Metrology Beacon 3 - SPH-1-0800-610
Rev
A
Size
A
Date:
Monday, June 10, 2002
Sheet
3
of
3

F.4 Metrology Beacon Tester

Design Drivers

- Allow operator to test each metrology beacon individually
  - Manual operation
  - Indicate beacon number
- Detect extraneous infrared and ultrasound signal in operational environment

Functional Block Diagrams

The beacon tester merges the design of the satellites US/IR boards and the metrology beacons to allow an operator to determine correct operations of the metrology beacons. The design uses the same infrared receiver product as the satellites and beacons. It uses the same ultrasound amplification and infrared transmission electronics as the satellites. The beacon tester uses the same PIC processor used in the beacons to create an infrared signal, detect the ultrasound signals, and measure the time of flight to identify the beacon number. Figure F.15 presents the functional block diagram of the beacon tester.

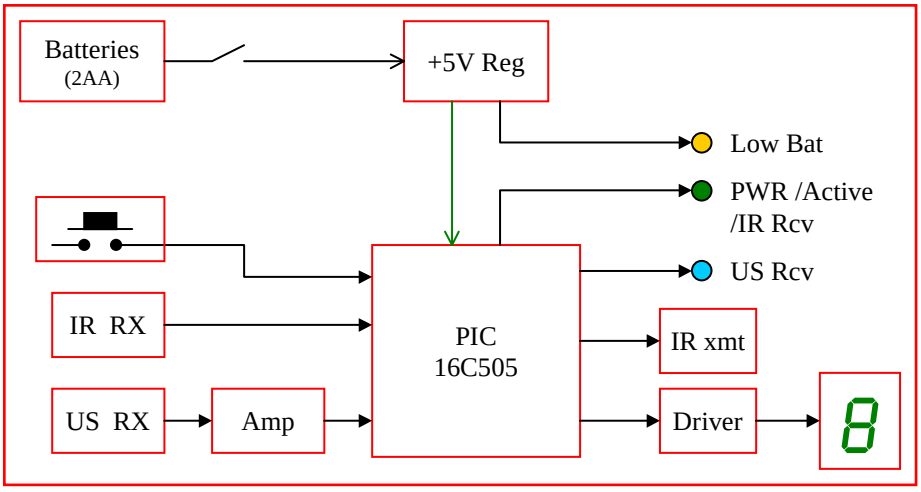
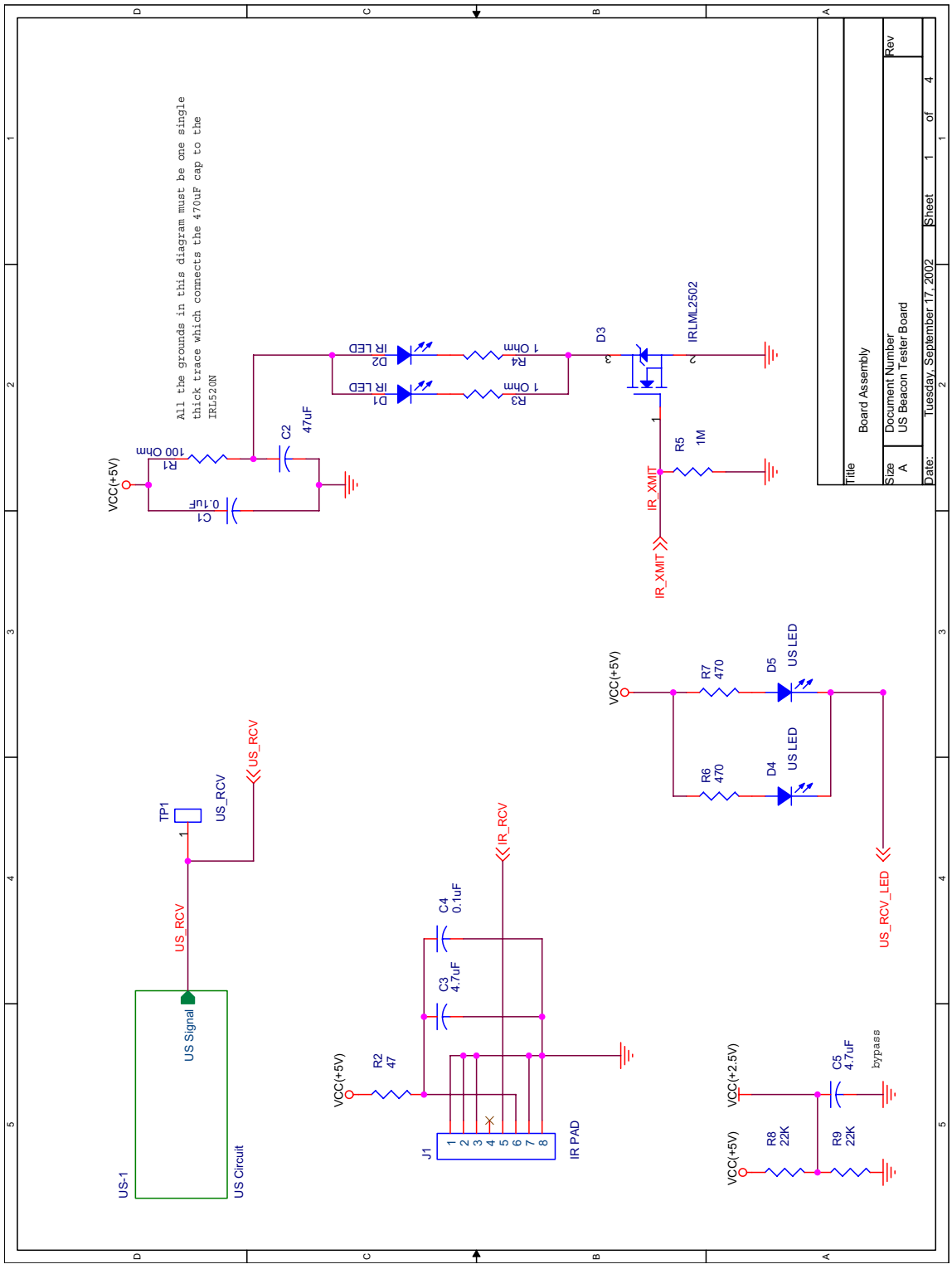


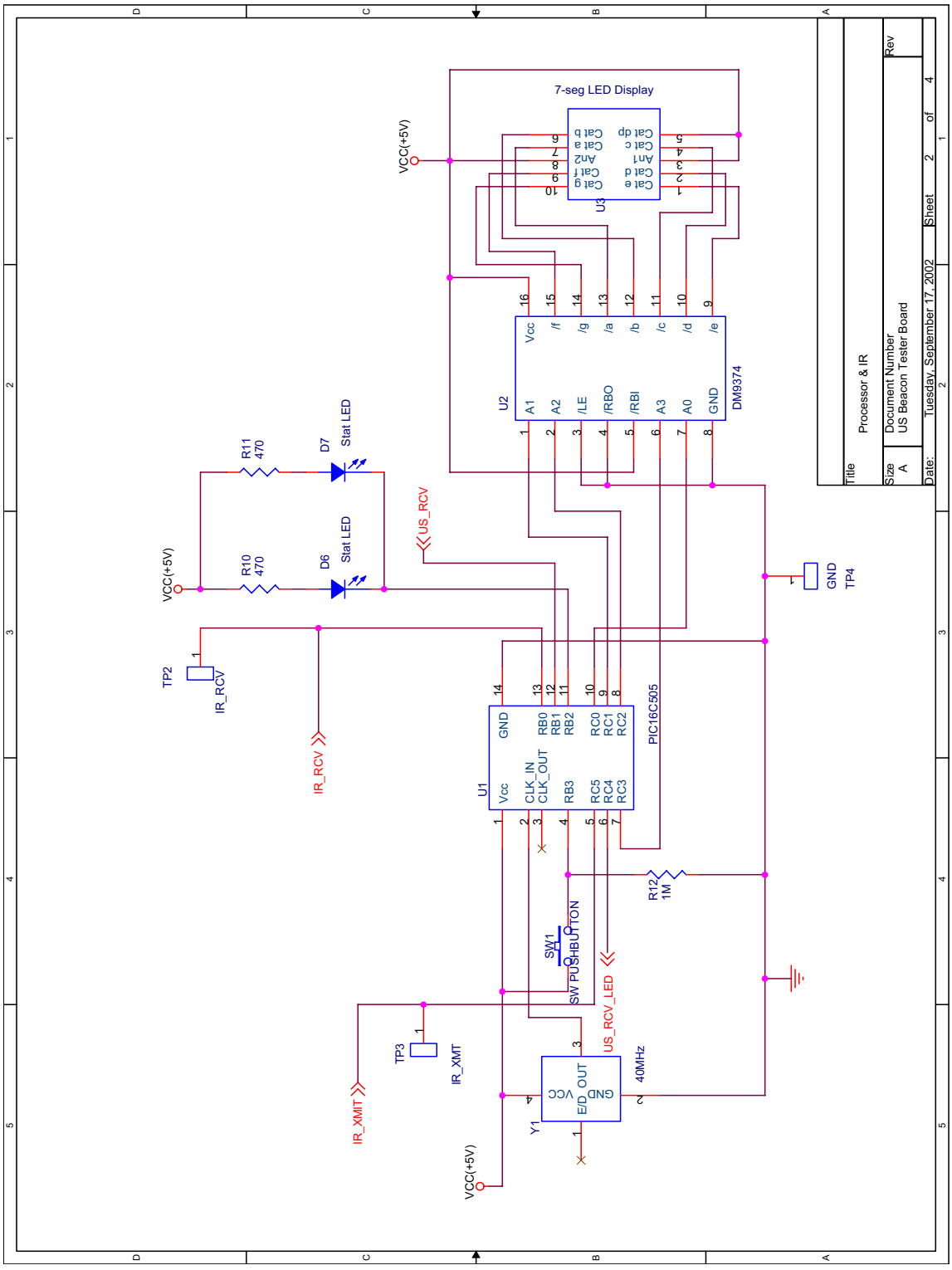
Figure F.15 Beacon tester functional block diagram

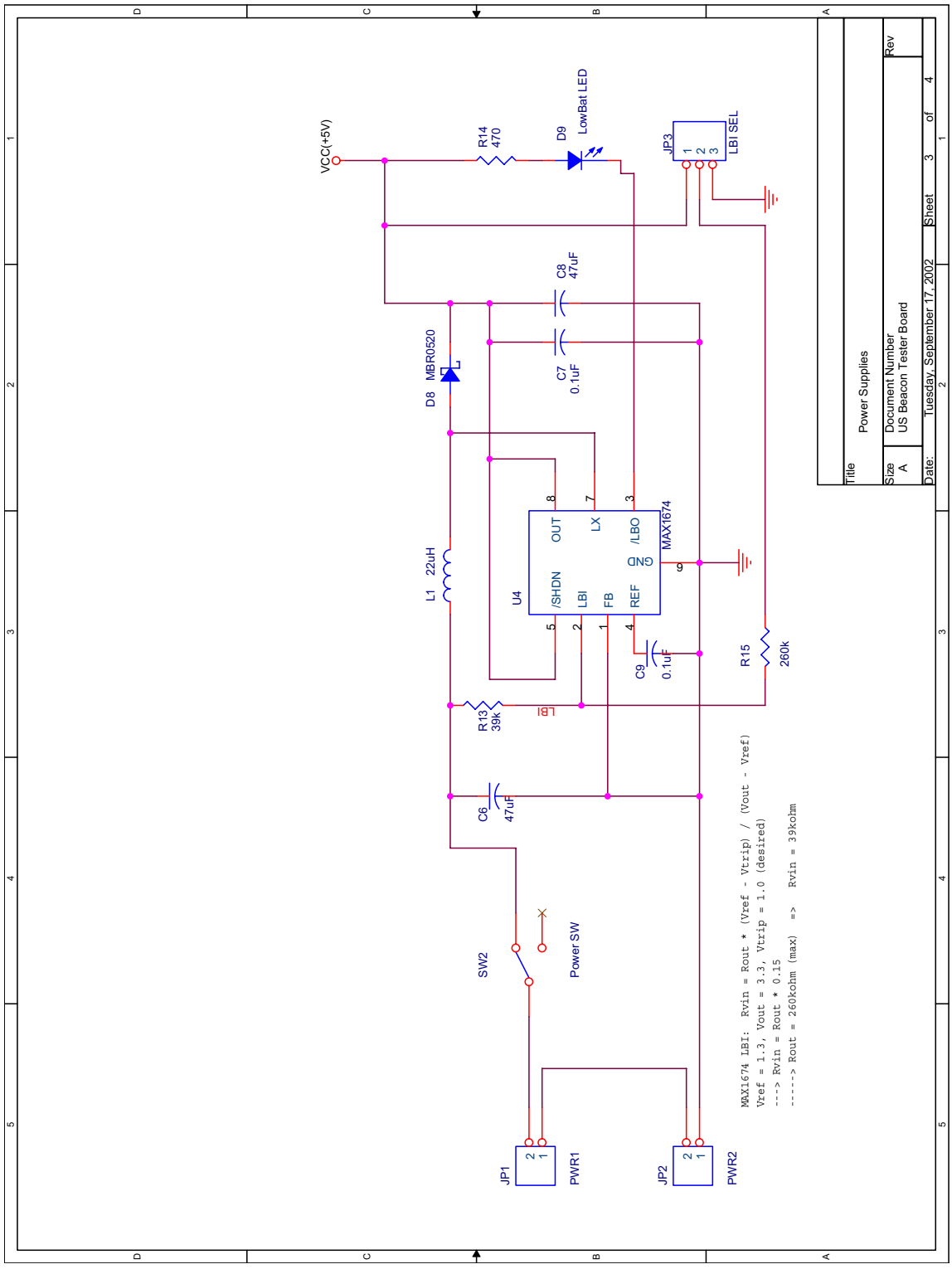
The beacon tester has a manual push-button used to command an IR transmission, equivalent of a satellite starting a global metrology cycle. If the beacon tester receives infrared or

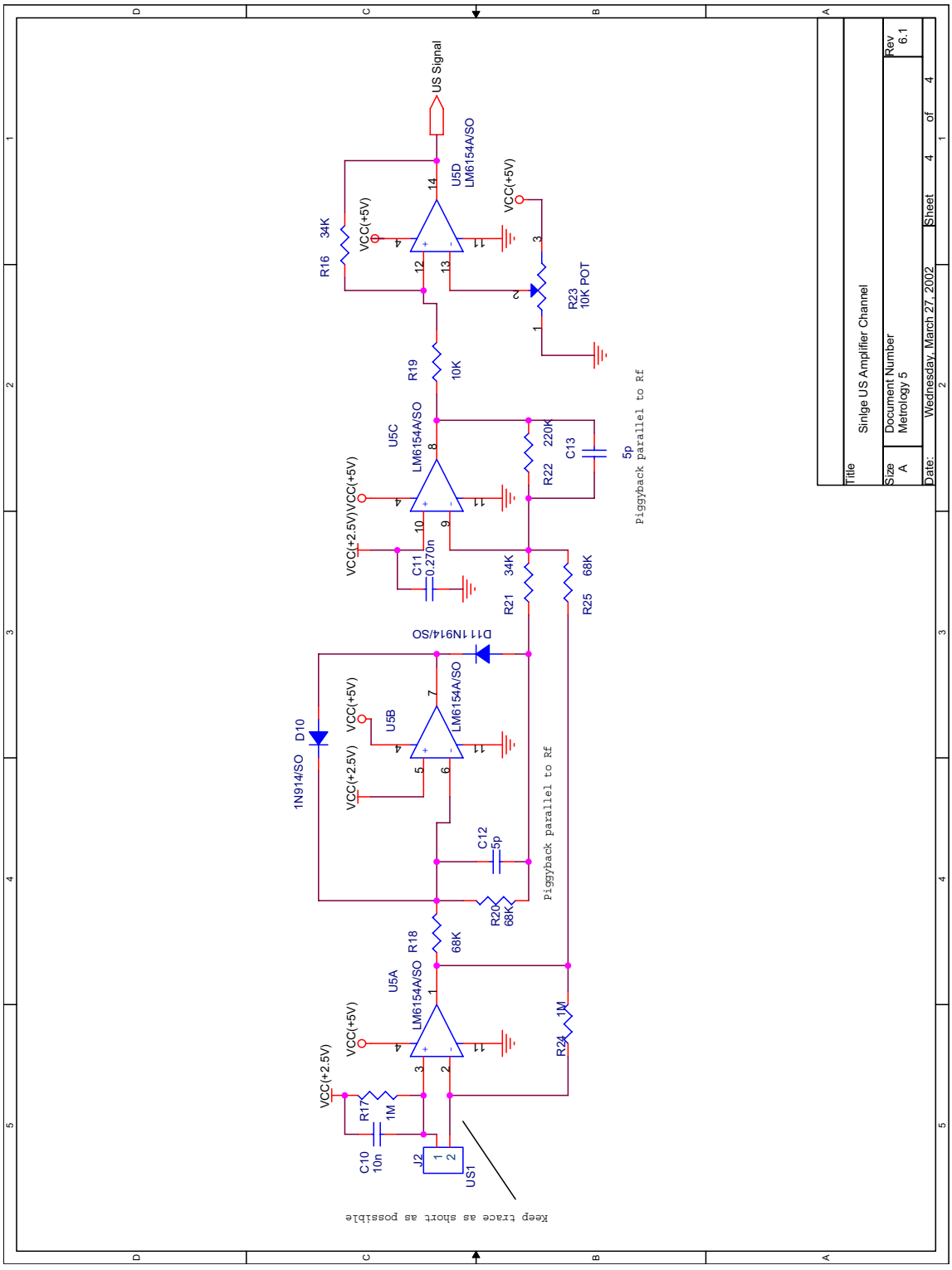
ultrasound signals when the push button has not been operated, it indicates an error (extraneous signals) by showing an E in the display. If the button is pressed and the beacon tester receives a valid ultrasound signal it will display the beacon number (1-9). If the beacon receives too many ultrasound signals or no signal at all, it will indicate an error (E). Any time an ultrasound signal is received the tester flashes the blue LEDs. Whenever an infrared signal is received the beacon tester flashes the green status LEDs. An amber LED indicates a low-battery condition.

Schematics


All the grounds in this diagram must be one single thick trace which connects the 470uF cap to the IRL520N
VCC(+5V)
R1 100 Ohm
C1 0.1uF
C2 47uF
IR LED D2
IR LED D1
R4 1 Ohm
R3 1 Ohm
D3
IRLML2502
R5 1M
IR_XMIT
IR_XMIT
R7 470
R6 470
D5 US LED
D4 US LED
US_RCV_LED
TP1
US_RCV
US_RCV
US_RCV
US Signal
US-1
US Circuit
IR_RCV
C4 0.1uF
C3 4.7uF
R2 47
J1
IR PAD
VCC(+2.5V)
VCC(+5V)
R8 22K
R9 22K
C5 4.7uF
bypass
Title
Board Assembly
Size A
Document Number US Beacon Tester Board
Rev
Date: Tuesday, September 17, 2002
Sheet 1 of 4


7-seg LED Display
U3
VCC(+5V)
U2
DM9374
R11 470
R10 470
D7 Stat LED
D6 Stat LED
US_RCV
TP2
IR_RCV
TP4
GND
U1
PIC16C505
R12 1M
SW1
SW PUSHBUTTON
US_RCV_LED
TP3
IR_XMT
IR_XMIT
Y1
40MHz
Title
Processor & IR
Size A
Document Number
US Beacon Tester Board
Rev
Date: Tuesday, September 17, 2002
Sheet 2 of 4


VCC(+5V)
R14
470
D9
LowBat LED
JP3
LBI SEL
C8
47uF
C7
0.1uF
D8 MBR0520
L1 22uH
U4
MAX1674
OUT
LX
/LBO
GND
/SHDN
LBI
FB
REF
C9
0.1uF
R15
260k
R13
39k
LBI
C6
47uF
SW2
Power SW
JP1
PWR1
JP2
PWR2
MAX1674 LBI: Rvin = Rout * (Vref - Vtrip) / (Vout - Vref)
Vref = 1.3, Vout = 3.3, Vtrip = 1.0 (desired)
---> Rvin = Rout * 0.15
-----> Rout = 260kohm (max) => Rvin = 39kohm
Title
Power Supplies
Size
A
Document Number
US Beacon Tester Board
Rev
Date:
Tuesday, September 17, 2002
Sheet
3
of
4


US Signal
U5D
LM6154A/SO
VCC(+5V)
R16 34K
R23
10K POT
R19
10K
U5C
LM6154A/SO
R22 220K
C13
5p
Piggyback parallel to Rf
C11
0.270n
VCC(+2.5V)
R21 34K
R25 68K
D11 1N914/SO
D10
1N914/SO
U5B
LM6154A/SO
C12
5p
R20
68K
R18
68K
U5A
LM6154A/SO
R24 1M
R17
1M
C10
10n
J2
US1
Keep trace as short as possible
Title
Sinlge US Amplifier Channel
Size
A
Document Number
Metrology 5
Rev
6.1
Date:
Wednesday, March 27, 2002
Sheet
4
of
4

F.5 Expansion Port Items

F.5.1 Expansion Port Beacon

The Expansion Port beacon was developed to allow formation flight algorithms where each satellite can transmit an ultrasound signal from two opposite sides: one using the on-board beacon (-X) and another using the expansion port (+X). The expansion port beacon replicates the internal beacon, but uses the serial line in the expansion port instead of it own IR or the use of the IR_rcv_int line which is not available to expansion items. The firmware in the expansion port beacon accounts for the extra delay in serial communications to initiate the command so that the receiving units do not need to account for the delay. The functional block diagram of the expansion port beacon is presented in Figure F.16. Table F.14 describes the inputs and outputs of this expansion item.

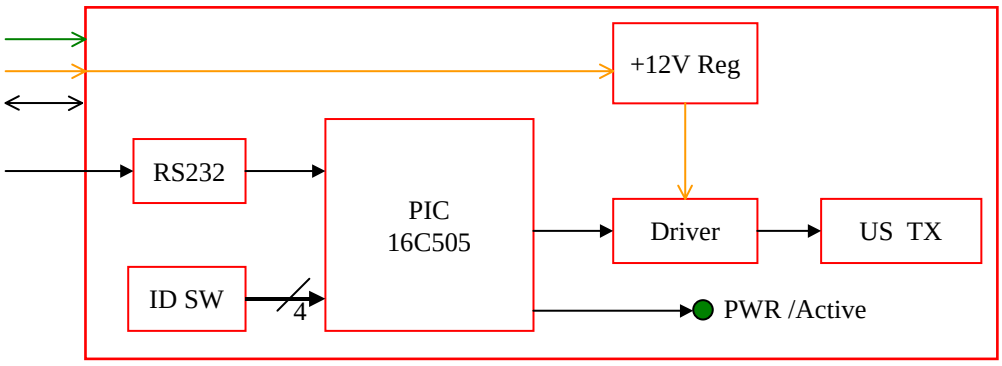
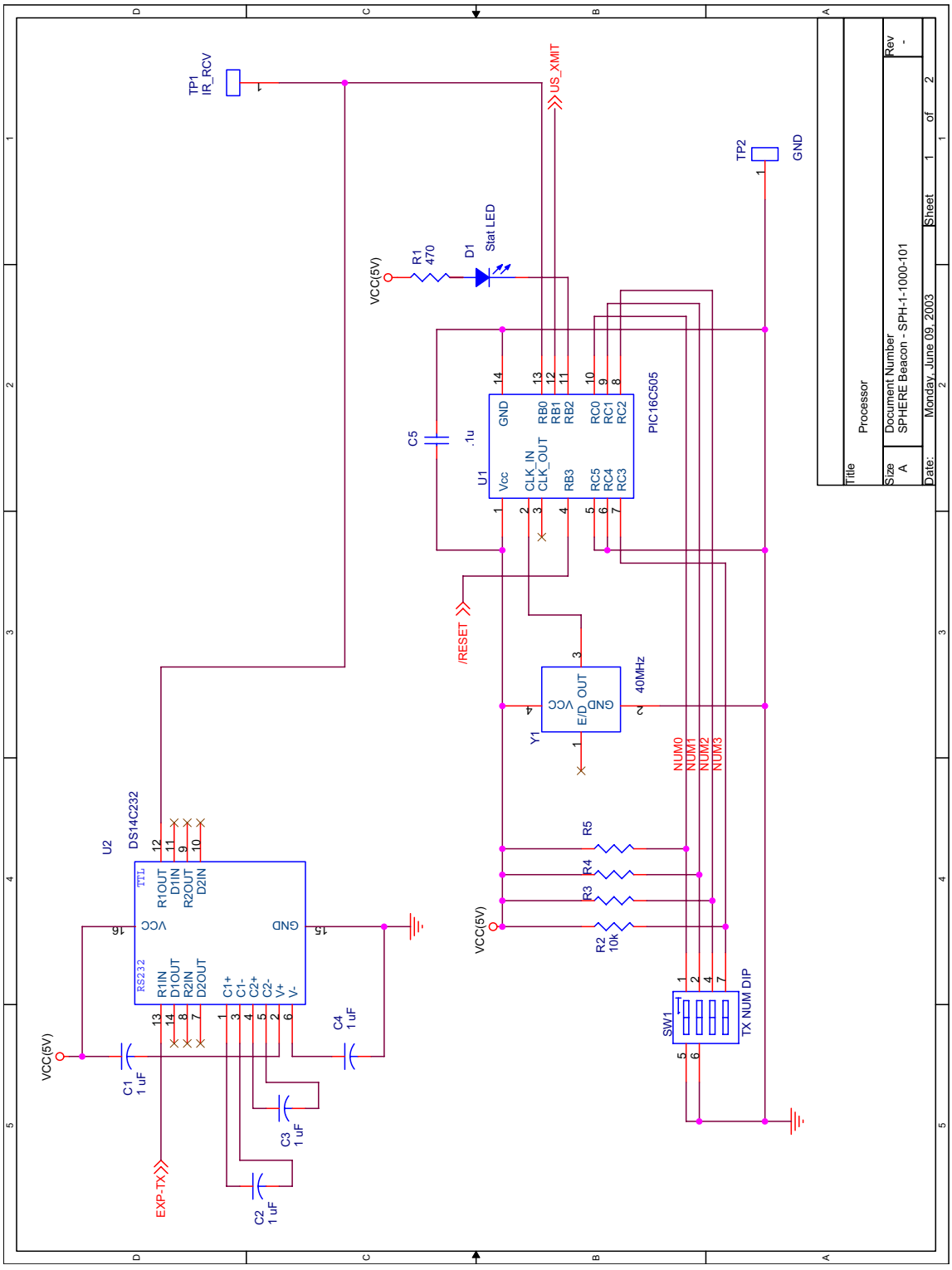


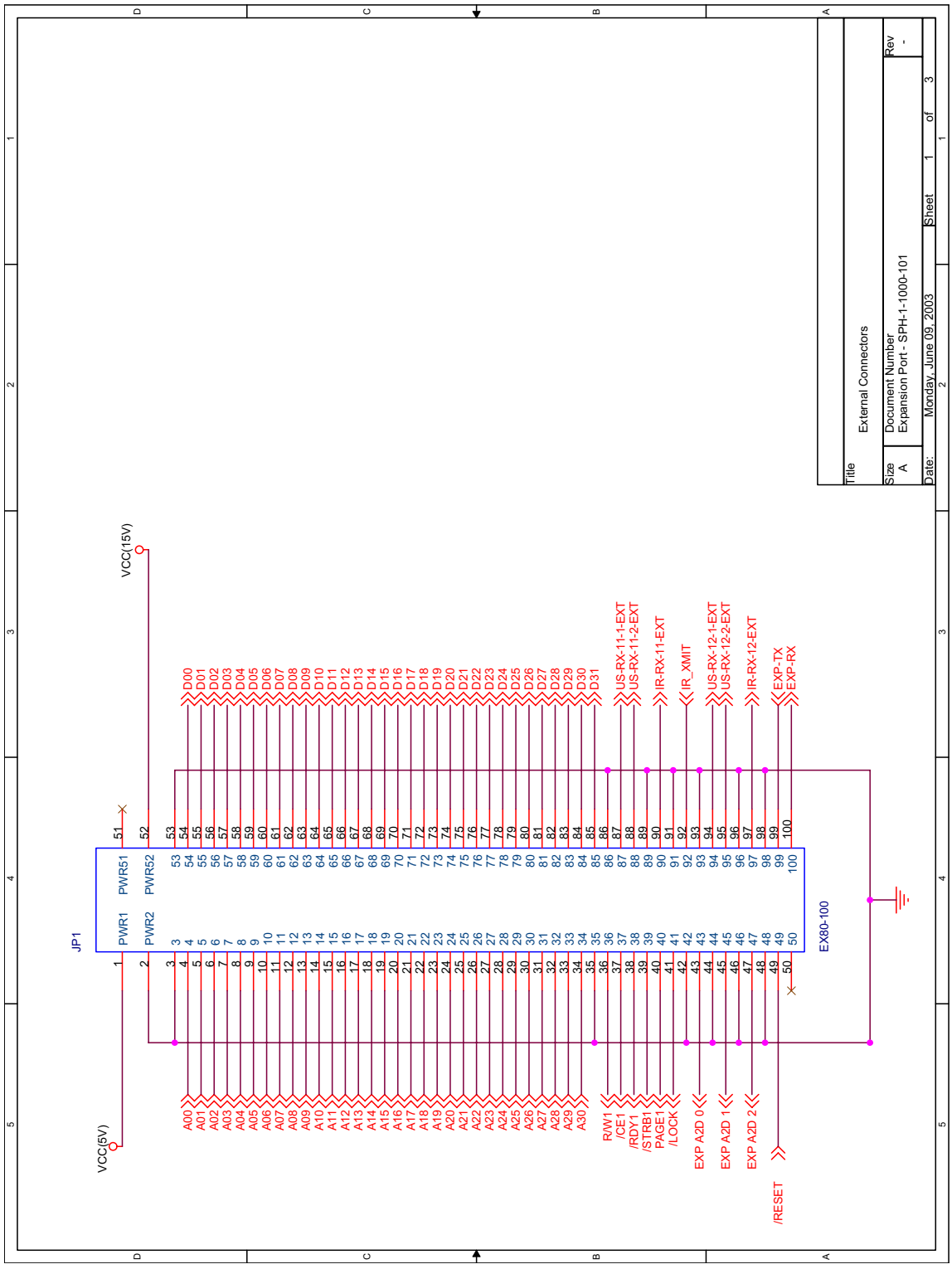
Figure F.16 Expansion port beacon functional block diagram

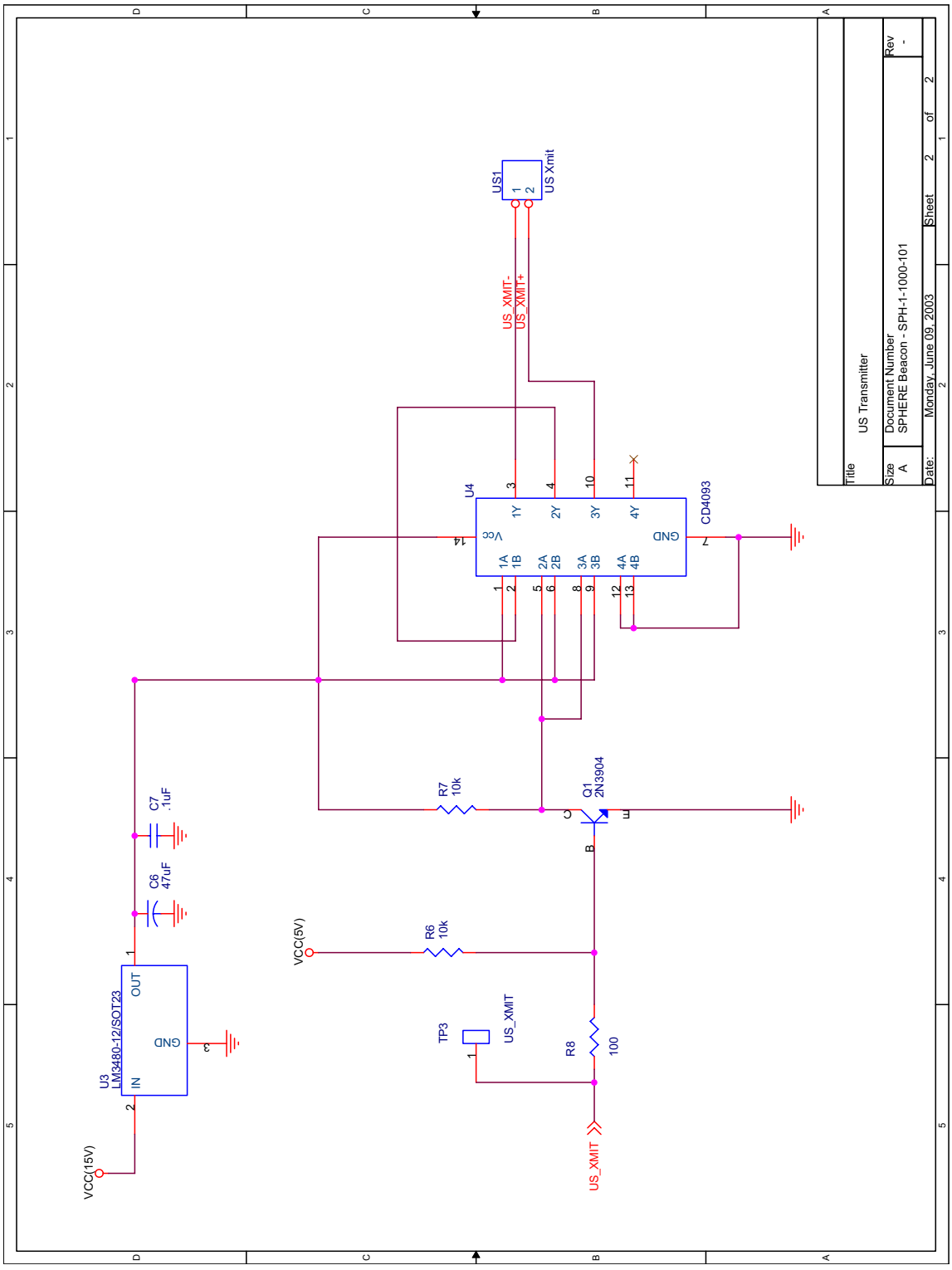
TABLE F.14 Expansion port beacon signals description

Signal	Type	Description
Vcc(+5V)	Pwr	+5V power
Vcc(+15V)	Pwr	+15V power
GND	Pwr	Common Ground
TX	In	Transmit data line (RS232)
/RESET	In	Reset line

Schematics


TP1
IR_RCV
US_XMIT
TP2
GND
VCC(5V)
R1
470
D1
Stat LED
C5
.1u
U1
PIC16C505
GND
Vcc
CLK_IN
CLK_OUT
RB0
RB1
RB2
RB3
RC0
RC1
RC2
RC3
RC4
RC5
/RESET
Y1
40MHz
E/D
OUT
NUM0
NUM1
NUM2
NUM3
R5
R4
R3
R2
10k
SW1
TX NUM DIP
U2
DS14C232
TTL
RS232
R1OUT
D1IN
R2OUT
D2IN
R1IN
D1OUT
R2IN
D2OUT
C1+
C1-
C2+
C2-
V+
V-
VCC
GND
C1
1 uF
C2
1 uF
C3
1 uF
C4
1 uF
EXP-TX
Title
Processor
Size
A
Document Number
SPHERE Beacon - SPH-1-1000-101
Rev
-
Date:
Monday, June 09, 2003
Sheet
1
of
2


JP1
PWR1
PWR2
PWR51
PWR52
EX80-100
VCC(15V)
VCC(5V)
/RESET
R/W1
/CE1
/RDY1
/STRB1
PAGE1
/LOCK
EXP A2D 0
EXP A2D 1
EXP A2D 2
IR_XMIT
EXP-TX
EXP-RX
Title
External Connectors
Size
A
Document Number
Expansion Port - SPH-1-1000-101
Rev
-
Date:
Monday, June 09, 2003
Sheet
1
of
3


US1
US Xmit
US_XMIT-
US_XMIT+
U4
CD4093
1Y
2Y
3Y
4Y
Vcc
GND
1A
1B
2A
2B
3A
3B
4A
4B
R7
10k
Q1
2N3904
C7
.1uF
C6
47uF
U3
LM3480-12/SOT23
OUT
GND
IN
VCC(15V)
VCC(5V)
R6
10k
TP3
US_XMIT
R8
100
US_XMIT
Title
US Transmitter
Size
A
Document Number
SPHERE Beacon - SPH-1-1000-101
Rev
-
Date:
Monday, June 09, 2003
Sheet
2
of
2

F.5.2 Expansion Port Tether

The Expansion Port Tether mechanism is a prototype system to test control algorithms for tethered formation flight spacecraft. The mechanism uses the expansion port serial line to interface with a COTS pulse-width-modulation driver board, which drives a motor to extend or retract a monofilament tether which connects two satellites. This prototype does not have any sensors, allowing the design to be very simple.

Figure F.17 shows the functional block diagram of the expansion port tether board. The board uses a SMC02B micro serial motor controller by Pololu Corporation ([Pololu, URL]). A serial line commands the micro controller the speed and direction of the motor. An adjustable (manual) voltage regulator in the expansion board allows testing of several motors at voltage ranges between 6.5V to 9V. Table F.15 describes the expansion port signals used by the expansion tether board.

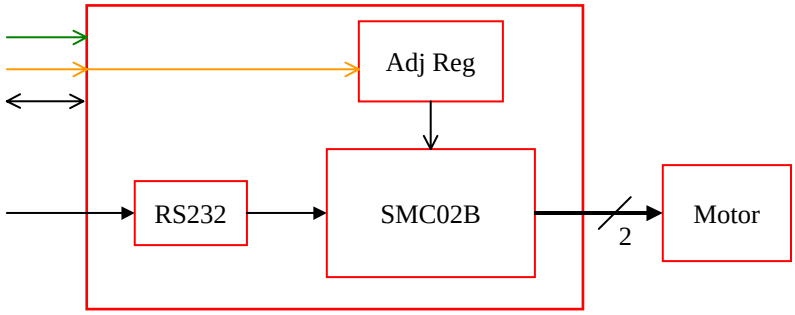
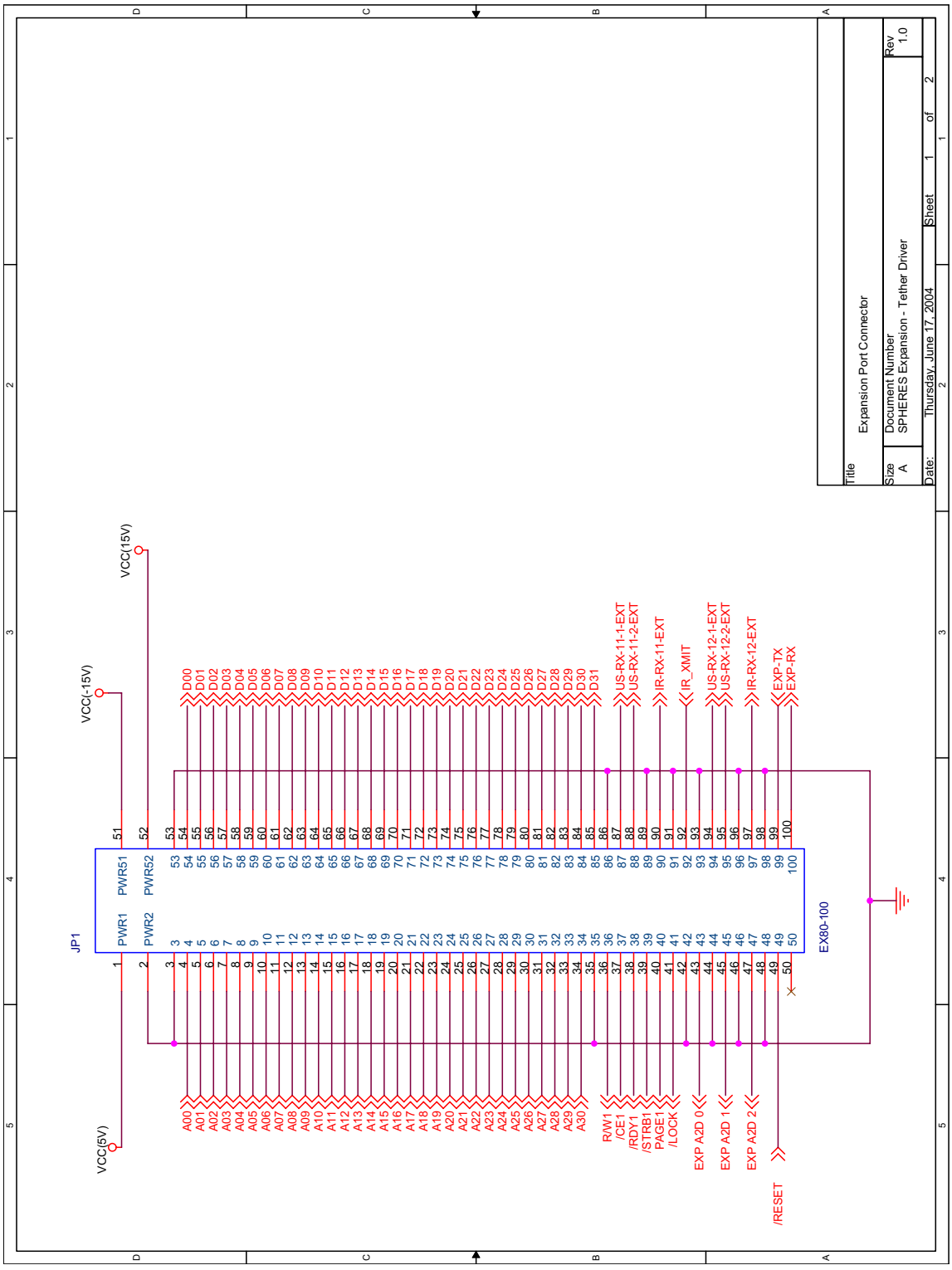


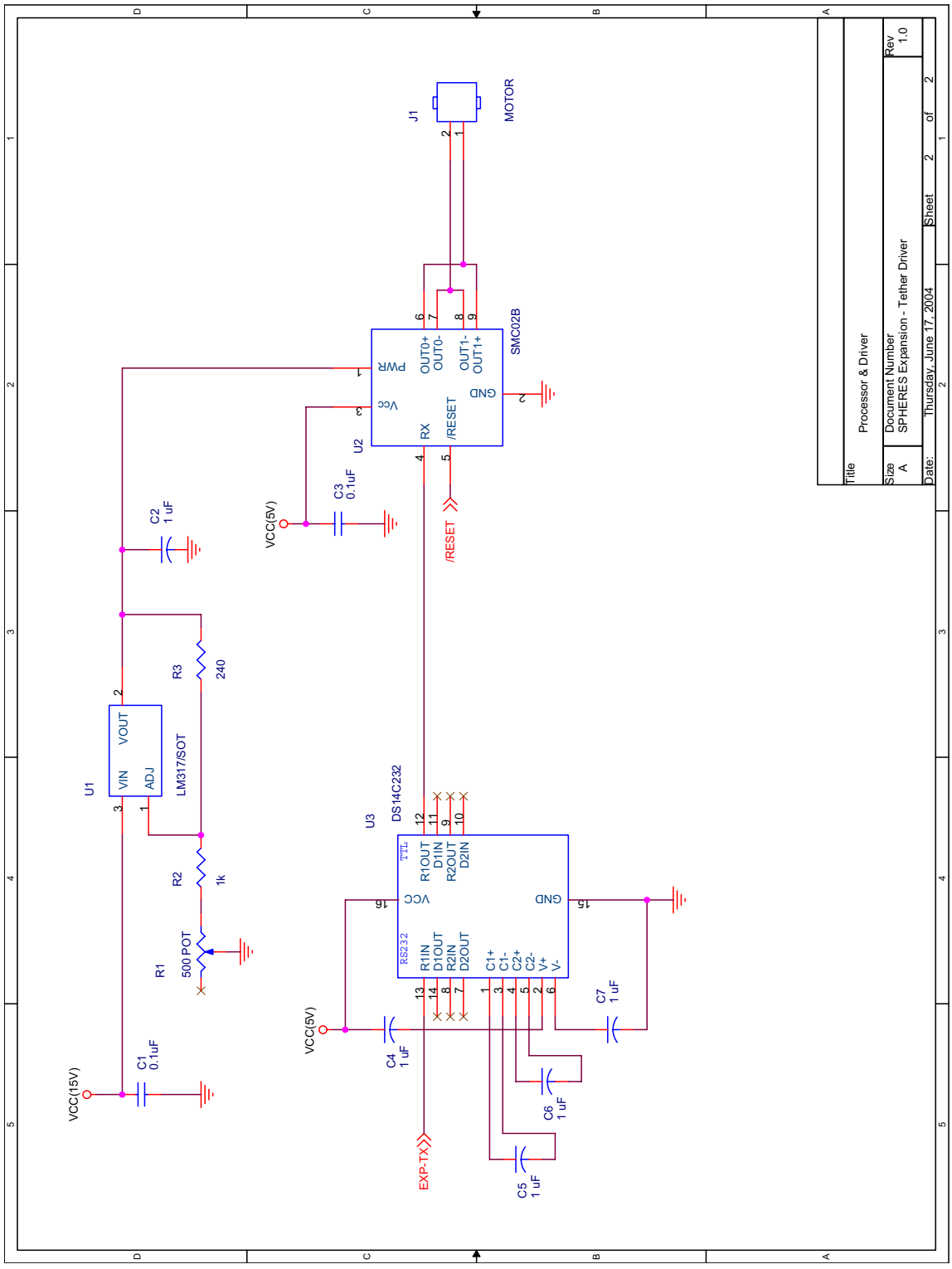
Figure F.17 Expansion port tether functional block diagram

TABLE F.15 Expansion port tether signals description

Signal	Type	Description
Vcc(+5V)	Pwr	+5V power
Vcc(+15V)	Pwr	+15V power - regulated to +7V for motor
GND	Pwr	Common Ground
TX	In	Transmit data line (RS232)
/RESET	In	Reset line

Schematics


JP1
PWR1
PWR2
PWR51
PWR52
EX80-100
VCC(-15V)
VCC(15V)
VCC(5V)
D00
D01
D02
D03
D04
D05
D06
D07
D08
D09
D10
D11
D12
D13
D14
D15
D16
D17
D18
D19
D20
D21
D22
D23
D24
D25
D26
D27
D28
D29
D30
D31
A00
A01
A02
A03
A04
A05
A06
A07
A08
A09
A10
A11
A12
A13
A14
A15
A16
A17
A18
A19
A20
A21
A22
A23
A24
A25
A26
A27
A28
A29
A30
R/W1
/CE1
/RDY1
/STRB1
PAGE1
/LOCK
EXP A2D 0
EXP A2D 1
EXP A2D 2
/RESET
US-RX-11-1-EXT
US-RX-11-2-EXT
IR-RX-11-EXT
IR_XMIT
US-RX-12-1-EXT
US-RX-12-2-EXT
IR-RX-12-EXT
EXP-TX
EXP-RX
Title
Expansion Port Connector
Size
A
Document Number
SPHERES Expansion - Tether Driver
Rev
1.0
Date:
Thursday, June 17, 2004
Sheet
1
of
2


J1
MOTOR
U2
SMC02B
PWR
OUT0+
OUT0-
OUT1-
OUT1+
GND
Vcc
RX
/RESET
C3
0.1uF
VCC(5V)
/RESET
C2
1 uF
U1
LM317/SOT
VIN
VOUT
ADJ
R3
240
R2
1k
R1
500 POT
C1
0.1uF
VCC(15V)
U3
DS14C232
TTL
RS232
R1OUT
D1IN
R2OUT
D2IN
R1IN
D1OUT
R2IN
D2OUT
VCC
GND
C1+
C1-
C2+
C2-
V+
V-
C4
1 uF
C5
1 uF
C6
1 uF
C7
1 uF
EXP-TX
Title
Processor & Driver
Size
A
Document Number
SPHERES Expansion - Tether Driver
Rev
1.0
Date:
Thursday, June 17, 2004
Sheet
2
of
2

Appendix G

SPHERES SOFTWARE DESIGN

This appendix presents the software design of the SPHERES satellites. The software developed by the SPHERES team consists of two main elements: the boot loader program and SPHERES Core. Figure G.1 illustrates the four pieces of software which operate in a satellite: the Texas Instruments boot firmware readies the DSP for operations; the Sundance custom firmware initializes the interfaces of the SMT375 board with the SPHERES avionics; the SPHERES boot loader allows to reprogram the operating system and loads the program into memory; the SPHERES Core Software (SCS) is the operating system which runs the satellites during normal operations.

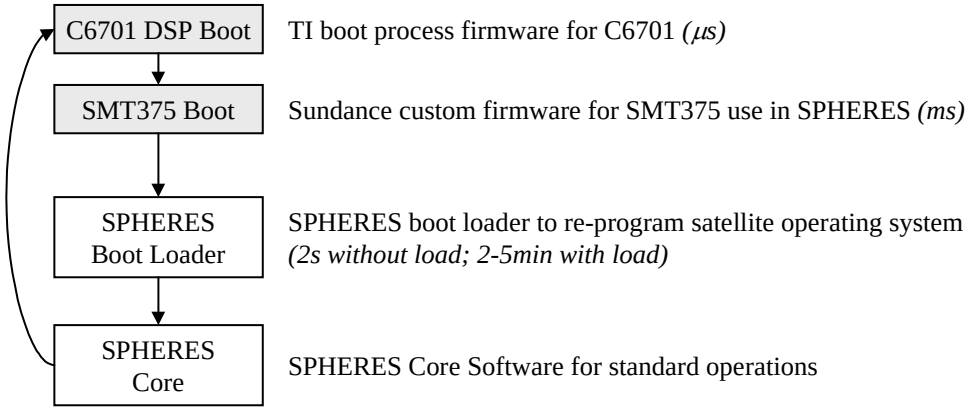


Figure G.1 Satellite software components

This appendix is divided into three main sections: the boot loader, the SPHERES Core Software operating system design, and the Standard Support Libraries.

G.1 Boot Loader

Figure G.2 shows the program development process for SPHERES. The SCS is programmed in assembly, standard ANSI C, or C++ using the Texas Instruments Code Composer Interface [TI, SPRU328B]. After compilation, the executable binary is translated into a text file and organized to enable storage in FLASH memory with `maketext` (this process is incorporated into the ground-based interface). This file is then transferred to the satellite via wireless communications. The boot loader reads the file from FLASH and loads it into RAM for operations. For ISS operations the file is packaged in a compressed file format for delivery to the ISS interface, which then decompresses the file and transmits it to the satellite in the same way that the ground-based interface does.

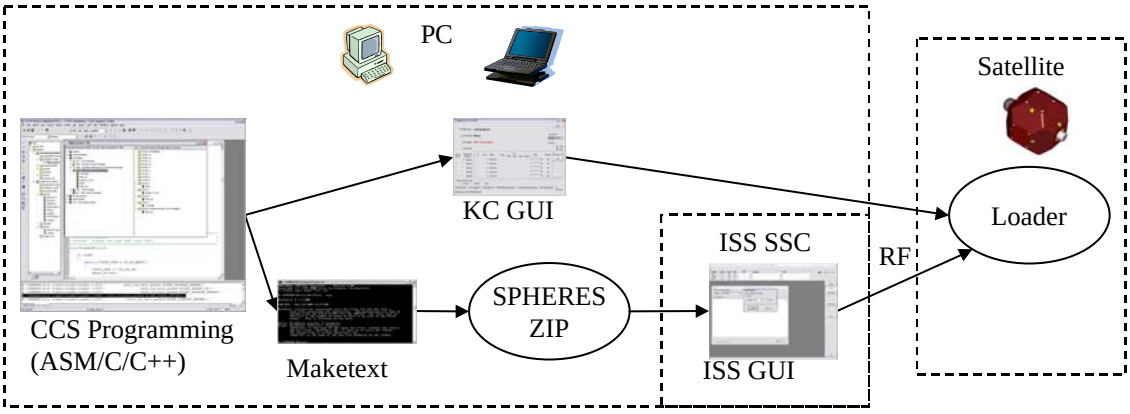


Figure G.2 SPHERES program development sequence

The TI Code Composer (CCS) interface assembles, compiles, and links the source code of the scientist and the DSP/BIOS kernel. The output from CCS is normally loaded directly into the RAM of a DSP system, therefore it requires some modifications for storage in FLASH. The function `maketext` utilizes two programs to enable transfer of the files to the satellites and their storage in FLASH. First, the TI provided program `hex6x.exe` is

used to convert the RAM ready output to hexadecimal format for storage in FLASH. This process involves separating the program into several sections and identifying the intended destination and size of those sections in RAM during operations. Next, the Sundance provided `hex2text.exe` is used to convert the binary hexadecimal into ASCII an text file. The text file is used in the wireless data transfer. The transferred file is converted to 32-bit words by the software on the satellites and stored in FLASH. When ready to run, the boot loader program on the satellites reads the sections written in FLASH and copies them to RAM.

G.1.1 Boot Loader Transfer Protocol

To balance the need for minimum transfer time together with the requirement for no data errors in the stored program, the boot loader uses packages of 20 packets to transfer the data. Each package must be acknowledged as received correctly or with a request to repeat the packet. The need to only acknowledge reception every 20 packets reduced the program upload time by both reducing the amount of data transferred and not requiring as many changes between transmission and reception in a unit.

Figure G.3 illustrates the boot loader data transfer protocol. The first step of both the PC and the satellite programs is to configure their DR200x transceivers (See Appendix H). Because programs are intended for specific satellites, the boot loader master program (on the PC) configures the DR200x to transmit only to a specific satellite; all other satellites will ignore the program. The satellite boot loader configures the DR200x for general operations. Table G.1 shows the boot-time configurations of the DR200x.

After initialization the master programs transmits a "start" packet to indicate to the satellite that data is available. It will transmit this packet at 1Hz until it receives an acknowledgement from the satellite. The satellite boot process is described below.

After the acknowledgement is received, the master program transmits the first packet individually, and awaits for a response. This packet is special because it includes the total pro-

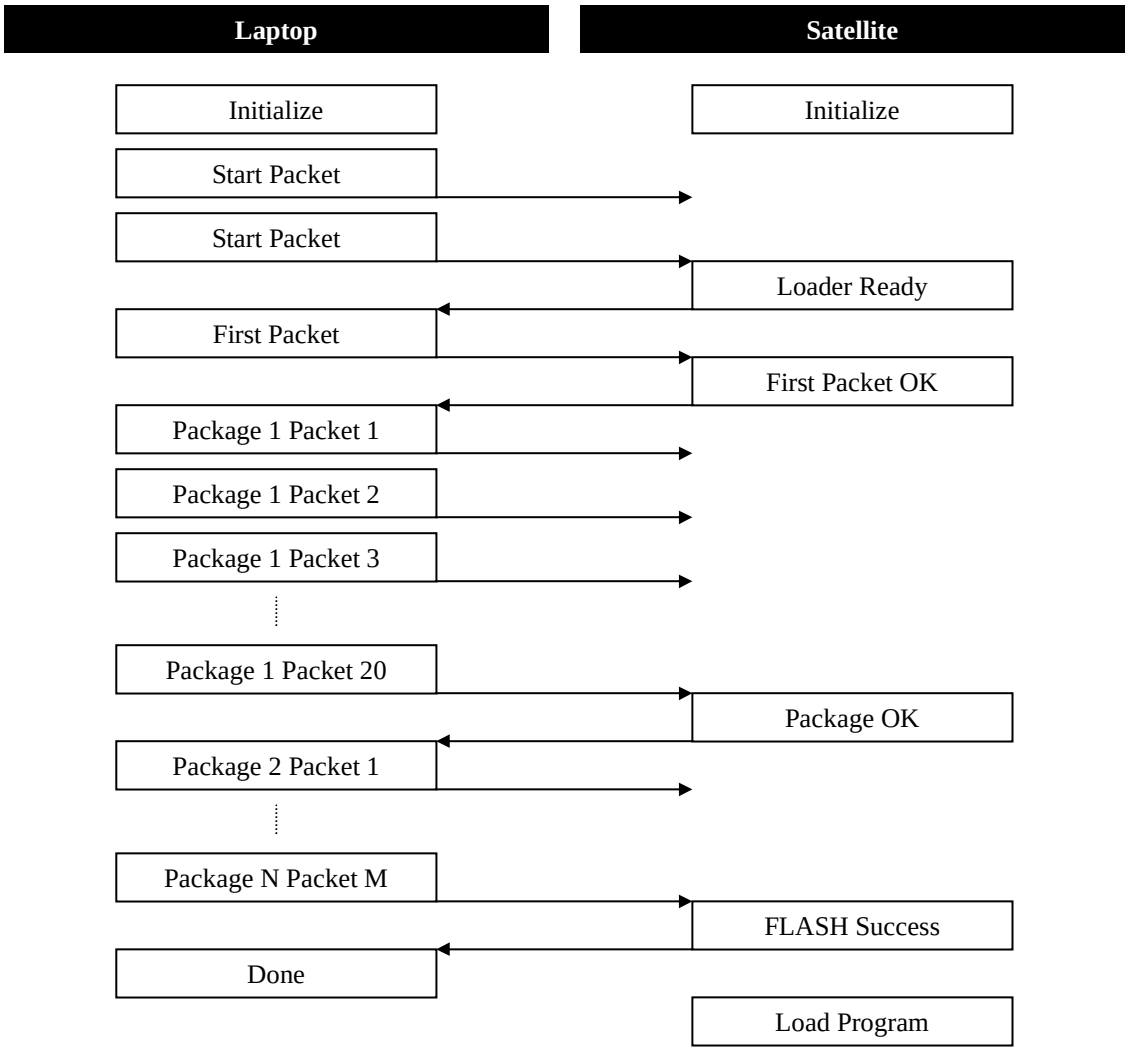


Figure G.3 Boot loader transfer protocol

TABLE G.1 DR200x configurations for boot load process

	PC	Satellite
TO	Satellite ID	0x00
FROM	0x30	Satellite ID
MODE	ASK	
RF data rate	56.6kpbs	
Packet size	6 then 75	6

gram size, that is, the amount of data to be transferred. The satellite will use this value to determine when it has received the full program, before overwriting the current program in FLASH. After the first packet has been acknowledged, the master program starts to transmit packages of 20 packets each. If a packet is not received correctly, the satellite requests a repeat, as illustrated in Figure G.4. This can be due to a data error (e.g. checksum error) or a timeout. Similarly, if the master does not sees a response after a timeout period, it transmits the packet again. Once all data has been received and confirmed with the checksums, the satellites unlocks the FLASH write function and overwrites the old program with the new one.

The boot loading process uses three types of packets: command/reply packets, first packet, and general packets. All packets contain a standard five byte header which indicates the destination (to), origin (from), packet number (pkt), command (cmd), and data size (len). The command/reply packets (Figure G.5, Table G.2) have one single byte of data, the command. The data packets (first and general) follow the header with a package number (pkg) and a packet number (pkt). The first packet (Figure G.6, Table G.3) has a 32-bit integer which indicates the total size of the program (in 32-bit words), followed by 15 32-bit words. General packets (Figure G.7, Table G.4) contain 16 program words (64 bytes). Data packets end with a 32-bit CRC (4 bytes).

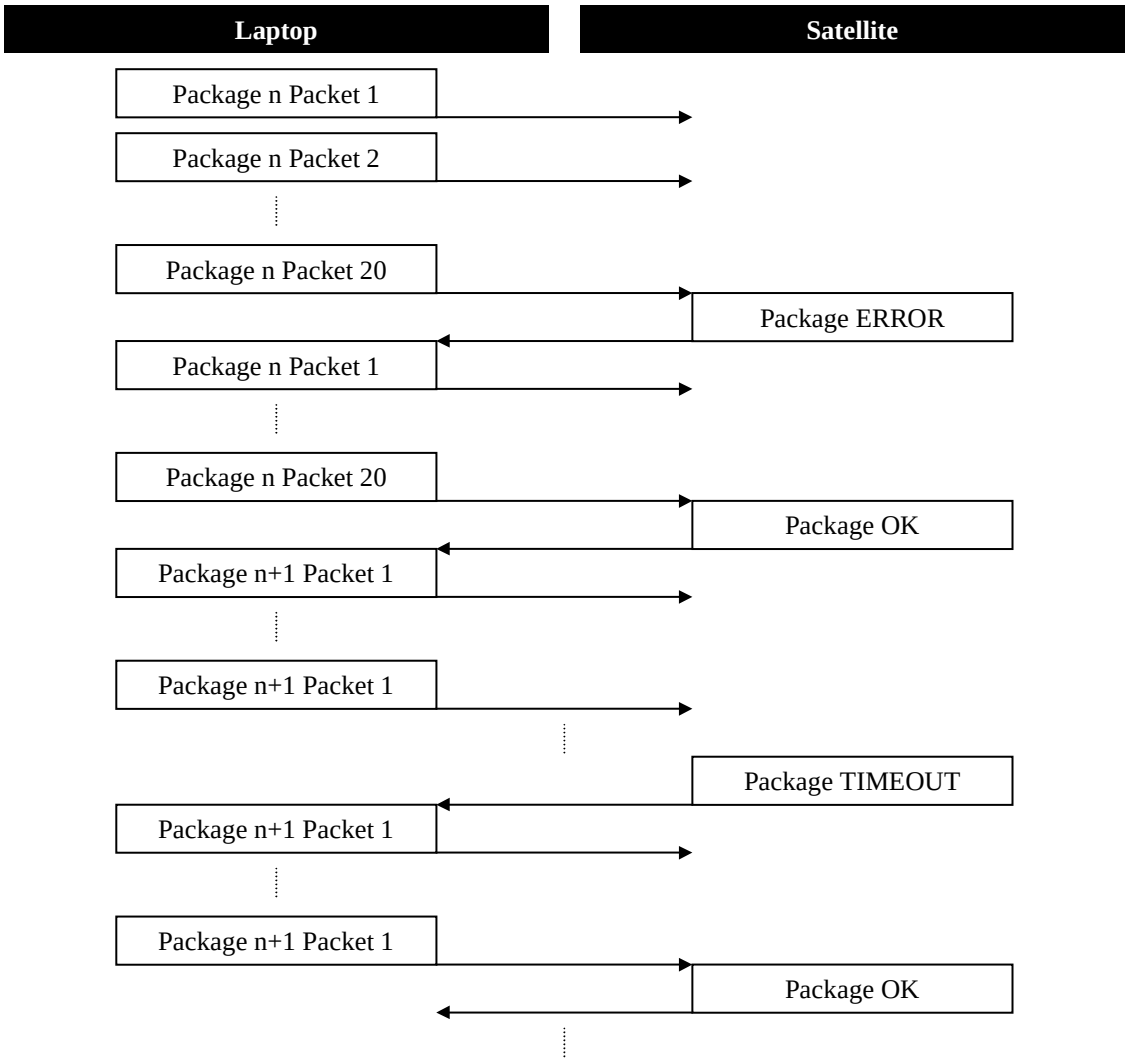


Figure G.4 Boot loader transfer protocol error handling

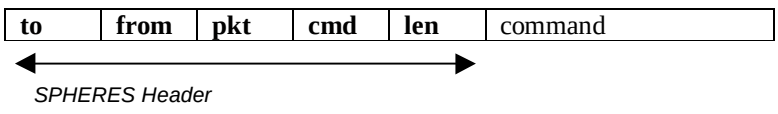


Figure G.5 Boot loader command/reply packets

TABLE G.2 Boot loader command/reply packet structure.

Byte	Width	Name	Function
1	1	to	The intended recipient.
2	1	from	This satellite's ID.
3	1	pkt	Packet counter byte, should change with every packet.
4	1	cmd	Command byte shared with SCS, therefore it **must** be 0x7F
5	1	len	The length of the data in the packet = 0x01

6	1	command	From satellite to laptop:
			`FL2EX_READY`
			`FL2EX_NO_PROGRAM`
			`FL2EX_INVALID_COMMAND`
			`FL2EX_PACKAGE_REPEAT`
			`FL2EX_FLASH_SUCCESS`
			`FL2EX_PACKET_OK`
			
			From laptop to satellite:
			`EX2FL_PROGRAM`
			`EX2FL_RUN`

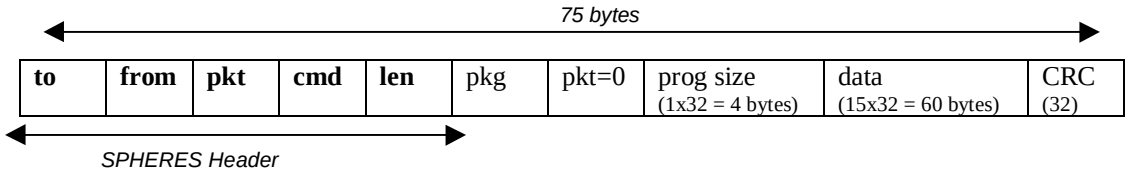


Figure G.6 Boot loader first data packet structure

TABLE G.3 Boot loader first packet structure

Byte	Width	Name	Function
1	1	to	The intended recipient
2	1	from	This satellite's ID
3	1	pkt	Packet counter byte
4	1	cmd	Command byte (0x7F)
5	1	len	The length of the data in the packet = 0x46

TABLE G.3 Boot loader first packet structure

Byte	Width	Name	Function
6	1	pkg	Package number
7	1	pkt	Packet number (1-20)
8-11	4	prog size	Total program size (in 32-bit words)
12-72	60	data	15 program words
72-75	4	crc	4 byte checksum

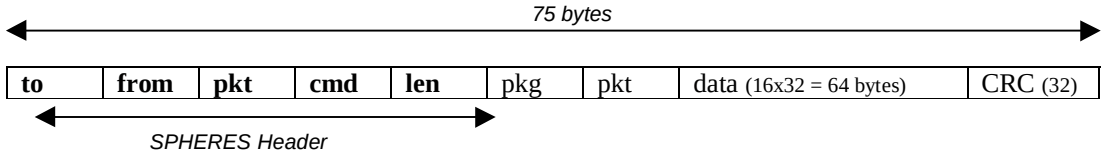


Figure G.7 Boot loader general data packets structure

TABLE G.4 Boot loader general packet structure.

Byte	Width	Name	Function
1	1	to	The intended recipient
2	1	from	This satellite's ID
3	1	pkt	Packet counter byte
4	1	cmd	Command byte (0x7F)
5	1	len	The length of the data in the packet = 0x46
6	1	pkg	Package number
7	1	pkt	Packet number (1-20)
8-71	64	data	16 program words
72-75	4	crc	4 byte checksum

G.1.2 Master

The implementation of the master program which runs on the PC to upload the programs has four main states as shown in Figure G.8. After initializing the DR200x as described above, the program enters idle mode and waits for a reply from a satellite. The actions of the master program in each state are as follows:

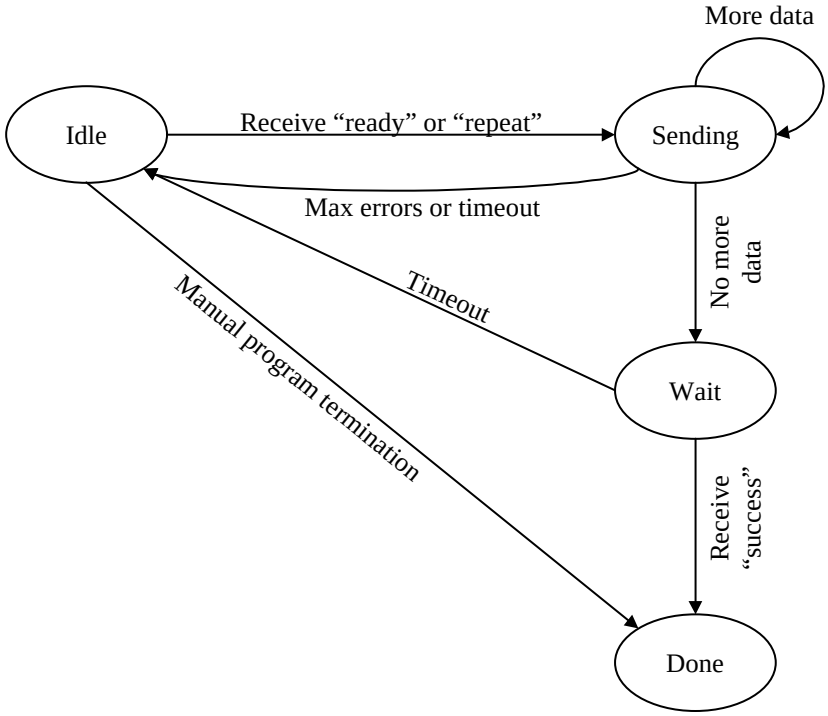


Figure G.8 Master program state diagram

- **Idle** - sends the `EX2FL_PROGRAM` command at 1Hz to the satellite until it gets a response or the program is terminated manually. If it receives the `FL2EX_READY` response it starts sending the program.
- **Sending** - sends packets until it is done or a pre-specified number of errors is reached (data errors or time outs). It transmit the first packet only, and awaits a response for that one packet the first time. Thereafter it transmits (or re-transmits) packages of 20 packets. When the last packet has been acknowledged it switches to the *wait* state. If the maximum number of error is reached it returns to *idle*.
- **Wait** - awaits a `FL2EX_SUCCESS` packet because all the program has been transferred. If the response is not received after a timeout it returns to *idle* to start again, since a failure in the burning the FLASH is assumed.
- **Done** - returns the DR200x configuration to normal operations (broadcast mode, 32-byte data) and exits.

G.1.3 Loader

The boot loader program on the satellites constitutes the only critical software element in the SPHERES project. Therefore, it has been programmed with minimal elements to reduce the sources of errors. Further, it uses some redundancy and a fallback mechanism to return the unit to operational conditions in case of a critical failure in the SCS. Figure G.9 presents the state diagram of the boot loader program. The states are explained below.

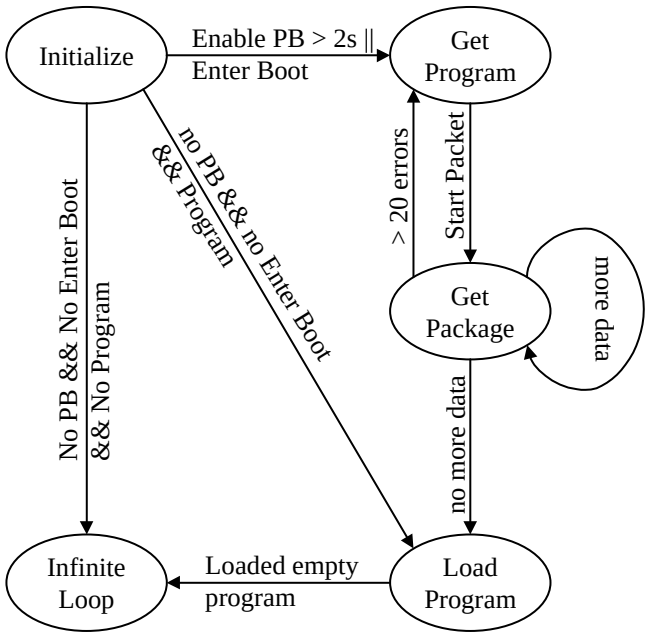


Figure G.9 Satellite boot loader state diagram

- **Initialize** - upon boot the loader initializes its hardware timers, reads the satellite identity and other information from a dedicated space in FLASH, initializes the DR200x modules and then checks to see if the *boot loader mode* should be entered or not. *Boot loader mode*, the *get program* state, waits for a program. The decision to enter boot loader mode depends on:
  - A register in FLASH being previously set by the boot loader itself or SCS.
  - The operator depressing the Enable push button in the SPHERES control panel for two seconds immediately after power on or upon reset.

If the boot loader mode is not entered, the boot loader checks if a program already exists in the FLASH memory. If it does, it will go into load program mode, otherwise it sends an `FL2EX_NO_PROGRAM` response and enters an infinite loop (slowly dimming Enable LED) to indicate no program exists in the FLASH.

- **Get Program** - in this state the satellite will flash the Enable LED slowly until it receives an `EX2FL_PROGRAM` command from the PC. It will then go to the *Get Package* state. If a satellite enters this state, it remains in this state until a program is received, an `EX2FL_RUN` command is received, or the satellite is reset.
- **Get Package** - executes the program transfer protocol described above (obtains program size from the first packet, acknowledges the packet, and then receives and acknowledges packages or 20 packets each). If there are more than 20 errors during the program upload (data error or timeouts) the loader returns to the *Get Program* state. If all the data is received, it stores the data in FLASH and then goes to the *Load Program* state.
- **Load Program** - this state reads the FLASH memory and copies each section of the program to its respective location in program RAM. After the program is placed in memory, it sets the program pointer to the location of `c_int00` (main) to execute the program. If the stored program length is zero or an error is detected with the program in FLASH, it will enter the infinite wait loop to indicate that no program exists.
- **Infinite Loop -** this state indicates that no program can be run with the satellite and therefore a new program must be loaded. The only way to exit this mode is to reset the satellite and enter boot loader mode.

The boot loader program has a single main thread which to perform its functions. Figure G.10 presents the general algorithm of the boot loader.

The redundancy and fallback mechanism include the use of hardware timers to control communications timing and to write data to the FLASH. The FLASH requires special timing sequences to start writing and between writing each sector, therefore the bootloader uses hardware timers to guarantee the timing. The timers do not create interrupts, they are polled to maintain a single-thread program. As discussed above, the boot loader initializes the DR200x modules. The reason is that the SCS programs can modify the DR200x configurations. The *load program* state will return the DR200x configurations to the standard SCS protocol, so that new programs are not jeopardized by configurations set in previous

Loader

```
initialize timers
read FLASH variables
initialize DR2000
check enable PB & FLASH variables
if enter loader
  wait for data in commport
  call get program
if application in FLASH
  call load application
else
  enter infinite loop
```

```
get program
  get first packet
  set transfer size
  while more to transfer
    get next package
      get 20 packets
      if error repeat package
  save data to FLASH
```

```
load application
  for each section in FLASH
    copy section to RAM
  set program vector to 0x0
```

Figure G.10 Satellite boot loader general algorithm

ones. Lastly, the use of redundant sectors of FLASH to store critical values to identify the satellite and its configuration creates redundancy in the system. The use of these sections is explained further below.

G.1.3.1 FLASH Variables

Critical values of the satellite are maintained in redundant FLASH sectors. There are two copies of these variables (referred to as locations ’0’ and ’1’), such that if one copy is corrupted, the second copy can be used. The variables stored in these special sectors are:

```
unsigned FlagAppInFlash;      // existence of a program
unsigned AppLength;           // length (words) of program
unsigned EnterBoot;           // command to enter boot
unsigned ID;                  // satellite ID
unsigned BattTime;            // time satellite has been on
```

```
unsigned TankTime;            // estimate tank usage
float BeaconPos[6][3];        // global metrology setup
float BeaconDir[6][3];
float AccelScale[3];          // accelerometer scale factors
float AccelBias[3];           // accelerometer bias
float GyroScale[3];           // gyroscope scale factors
float GyroBias[3];            // gyroscope bias
unsigned count;               // count of FLASH storage
unsigned checksum;            // checksum of FLASH memory
unsigned ver;                 // boot loader version
unsigned Padding[7];          // unused
```

The most important variables to the boot process are:

- `FlagAppInFlash` - indicates if a program has been stored in FLASH memory. A value of `0x12ABCDEF` indicates a program is present. Any other value indicates a program is not present (or is corrupt).
- `EnterBoot` - if set to 0x1 it will enter the boot loader mode and wait for a program even if one already exists and the enable push button has not been depressed during boot time. This allows the SCS to command the satellite to enter boot loader mode through the wireless communications systems, without the need for the operator to use the control panel. But, since the boot loader only changes this byte back to zero after a program has been loaded, it also effectively erases any current program in the satellite.
- `ID` - the satellite ID, which will always be matched to the packets to ensure the data is for this satellite.
- `Count` - a count of how many times these special FLASH variables have been saved. The count is used to identify the latest information between the two storage locations, since the boot loader always guarantees to the SCS that the latest information will be in one location ’0’. Therefore, if location ’1’ contains the latest information, it is copied over to location ’0’.
- `Checksum` - a checksum of all the previous values stored in the sector. If the latest version of the sector is corrupted, the boot loader checks the other sector. If the other sector is correct, it is copied to the corrupted sector, and the boot loading process continues. If both sectors are corrupt, the boot loader creates a new sector with the default values shown in Table G.5

TABLE G.5 Boot loader FLASH variables default values

Variable	Value
FlagAppInFlash	0;
AppLength	0;
EnterBoot	1;
ID	0x39;
BattTime	0;
TankTime	0;
BeaconPos[6][3]	{0};
BeaconDir[6][3]	{0};
AccelScale[3]	{0.0};
AccelBias[3]	{0.0};
GyroScale[3]	{0.0};
GyroBias[3]	{0.0};
count	0;
checksum	<new checksum>;
ver	0x31;

G.2 SPHERES Core

The SPHERES Core Software (SCS) layer acts as a buffer between the user-provided experiment code, the DSP/BIOS operating system, and the satellite hardware. The Texas Instruments DSP/BIOS [TI, SPRU423B] real-time operating system is used as the operating system on the SPHERES satellites. This kernel provides multi-processing capability, inter-process communication, and a number of input/output management tools. Through multiple execution threads created using DSP/BIOS, SCS controls the scientist provided functions which implement their specific algorithms. In this manner, the SCS creates a generic real-time operating system for the development of metrology, control, and autonomy algorithms. The core software encapsulates the hardware interfaces of the satellites by providing software interfaces to them. The SCS also preforms several housekeeping and data management tasks. The main functions of the SCS are:

- **Control.** The SCS implements a digital real-time controller and executes the code provided by scientists at a specified rate. The control module implements the test-management functions which allow a program to have multiple tests.
- **Propulsion.** SCS interfaces between the user code and the digital outputs to the propulsion hardware. The basic interface implements standard on/off pulses commanded through an array of on/off times.
- **Communications.** The SCS manages the TDMA communications protocol, handles incoming communications, and prepares standard and custom packets for transmission. The SCS also provides a module which allows scientists to transfer data of variable lengths (longer than a standard packet).
- **Metrology.** The SCS implements several threads to capture data and run metrology algorithms. High-priority threads collect the data. Middle priority threads can process data at high frequencies. Two low priority threads are implemented to run extended procedures in the background.
- **Housekeeping.** The SCS performs a number of routine tasks automatically in the background: it monitors the tank usage, reports health status to the control station, and downloads telemetry information.

The organization of the SCS modules with respect to the guest scientists modules, the DSP/BIOS kernel, and the SPHERES hardware is depicted graphically in Figure G.11.

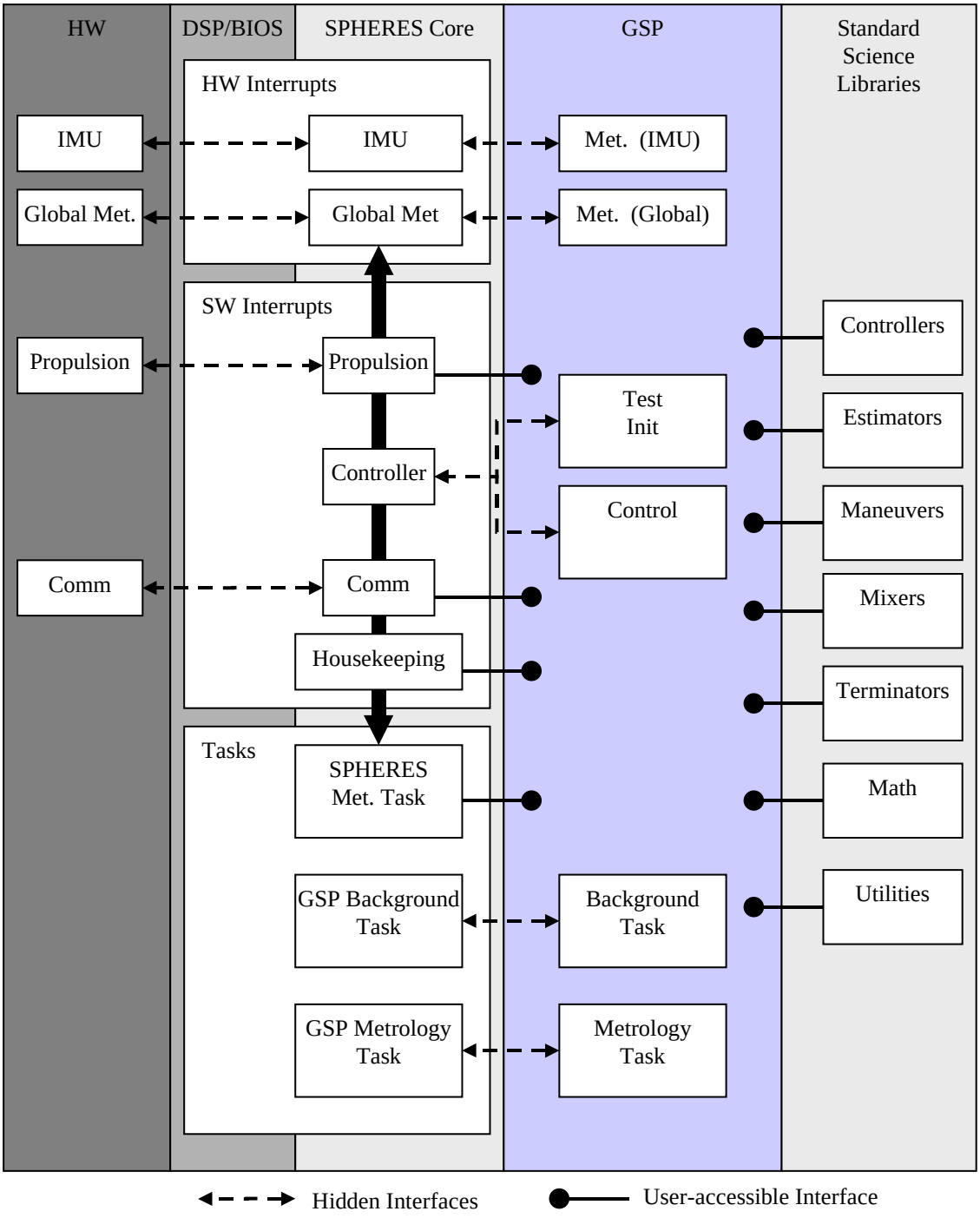


Figure G.11 SPHERES Core Software overview

SCS makes use of five types of threads available in the DSP/BIOS kernel (listed from highest to lowest priority): hardware interrupts (HWI), hardware clock interrupts (CLK), software interrupts (SWI), periodic software interrupts (PRD), and semaphore driven tasks (TSK). Figure G.12 presents the threads implemented by the SCS to create the SPHERES generic operating system. The following threads are used:

- **Hardware interrupts (HWI)**
  - **IR Rcv** - reception of a global metrology infrared signal.
  - **PADS Int** - interrupt by the metrology FPGA indicating data availability.
  - **COMM Rx** - reception of data through one of the commports
  - **CLK** - an interrupt created by DSP/BIOS to drive the periodic hardware clock interrupts
- **Periodic Hardware Interrupts (CLK)**
  - **Propulsion** - creates the propulsion solenoid signals at 1kHz
  - **Comm TDMA Mgr** - manages the TDMA transmission windows for the satellite
- **Software Interrupts (SWI)**
  - **PADS** - triggered from the PADS Int HWI, it collects the data, performs some data processing, and stores the results for other processes
  - **COMM Tx** - transmits data out through the commports one full packet at a time during the TDMA transmission window
  - **PRD** - a thread created by DSP/BIOS to manage periodic SWI's
  - **Control Dispatch** - manages the timing of the control software interrupt at 1kHz, allowing simple changes in the rate of the controller interrupt
  - **Fast Housekeeping** - keeps track of the state of health of the satellite and triggers the watchdog to prevent a hardware reset
  - **Telemetry** - downloads state information periodically
  - **Controller** - implements the test management routines and runs the periodic control algorithm specified by scientists
  - **KNL** - the kernel process created by DSP/BIOS
- **Background Tasks (TSK)**
  - **Comm Mgr** - provides the interface for all incoming and outgoing data
  - **PADS Global TX** - used to trigger an infrared pulse for global metrology

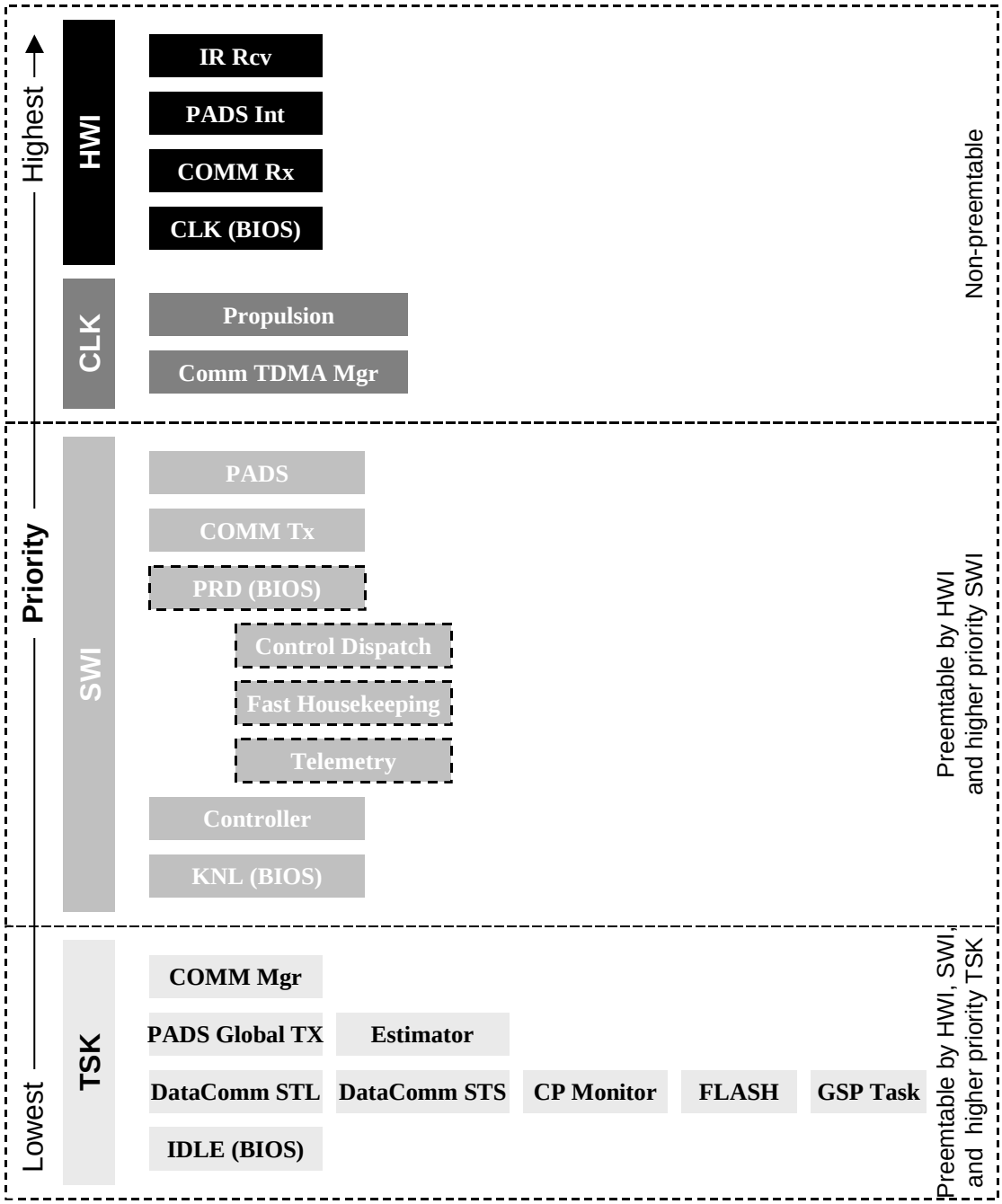


Figure G.12 SCS threads

- **Estimator** - runs the SPHERES standard estimator
- **DataComm STL/STS** - manages large data transfers by dividing it up into standard SPHERES packets on the transmitting satellite and rebuilding the data in the receiving satellite

- **CP Monitor** - monitors the state of the commport interrupts upon boot
- **FLASH** - writes to the FLASH memory in the background, since the flash requires several milliseconds between writing each sector
- **GSP Task** - allows guest scientists to run extended tasks in the background
- **IDLE** - the idle process created by DSP/BIOS

The SCS utilizes several global timers to maintain a system time, to trigger timed threads, and to control timed events. Figure G.13 presents the major timers used in the SCS. Their description follows.

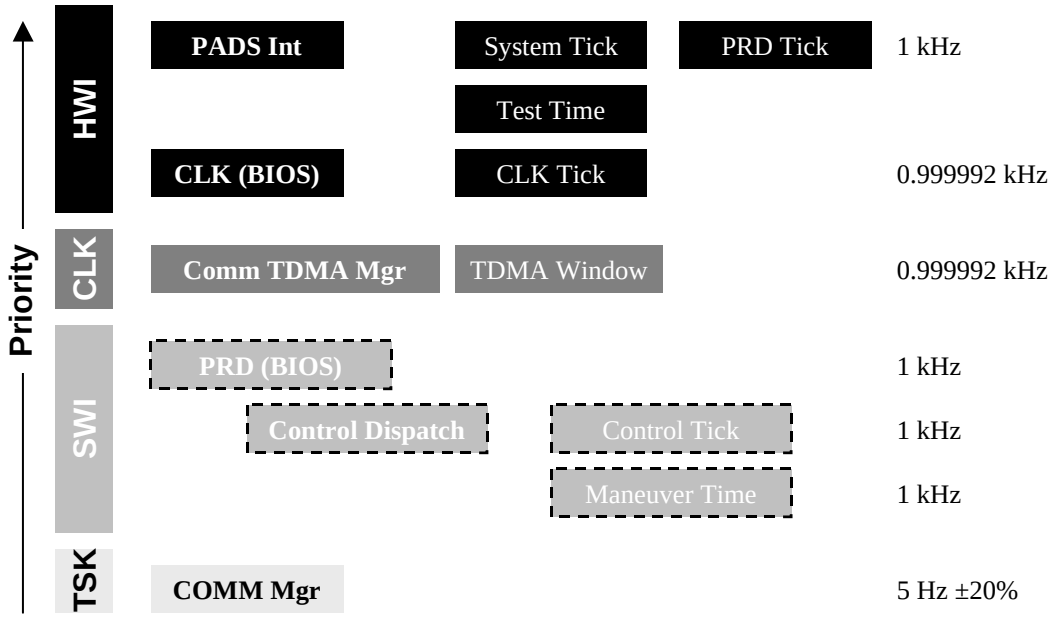


Figure G.13 SCS real-time clocks

- **PADS Int -** From the Metrology FPGA 1kHz hardware interrupt
  - **System Tick** - Maintains the global system time in milliseconds. Reset to zero when the satellite is reset (hardware or software reset).
  - **Control Tick** - The control tick is used by the control dispatcher thread to post new controller SWI's. Because the control tick is based on the PRD and the PRD on the PADS Int, the controller is always synchronized with the system tick.

  - **Test Time & Maneuver Time** - The test time and maneuver time reset whenever a new test or maneuver starts, respectively. They are synchronized with the system time.

- **CLK** - From the DSP hardware timer based on the SMT375 clock of 166.67MHz, which results in a period of 1.0078ms which gives a frequency of 0.999998kHz
  - **Propulsion & Comm TDMA Window** - The propulsion and communications TDMA manager functions utilize the SMT375 clock, rather than the PADS clock. Therefore, these functions are slight off-sync from the global system time. Because they operate through relative times (active = off_time - on_time in both cases) the errors are always minimal.
- **COMM Mgr** - The communications manager task processes the packets received from the control laptop, which include the command to start a new TDMA frame. Because the laptop can have errors as large as 20ms, this process operates at 5 Hz ±20%. This requires the satellites to store all the data until a valid window appears, and breaks any correlation between the data reception time and the time it was created, therefore packets are identified with the system time when they are created.

These threads and timers support the modules of the SCS. The next sections describe the modules of the SCS in detail:

- System
- Control
- Propulsion
- Communications
- Metrology
- Housekeeping
- Guest Scientist Program Interface

G.2.1 System

The system module includes the system initialization routines and maintains the global satellite time and the satellite's physical parameters.

The system is initialized using the standard C function `void main()`. It is a thread which executes **once** after a reset to initialize the satellite and exits after completion. The DSP/BIOS kernel starts the real-time environment threads after the main function completes. The function performs the following actions:

- Initializes the hardware timers, global bus, DR200x, metrology FPGA, and satellite state estimate.
- Loads the FLASH variables to identify the satellite.
- Sets physical parameters.
- Runs the GSP initialization routines.
- Sets up hardware interrupts.
- Loads the DSP/BIOS kernel.
- The module is comprised of the following files:

The system module implements the local variables used to maintain the time of the satellite and tests. These times are kept as follows:

- **System time** - Set to zero during program initialization. Updated in the PADS Int HWI
- **Test time** - Set to `SYS_FOREVER` when a test is not running. It is increment whenever it is *not* `SYS_FOREVER` (the control module starts the clock by asking the system module to set it to zero) through the PADS Int HWI.

The system module also maintains the *logical* identity of the satellite. The logical identity tells the satellite what role it plays within a distributed system. This allows a scientist to decouple the physical serial number (i.e., the hardware ID used in the communications system) from the logical role played by the unit during a test. In this manner, any satellite can be used to perform any role.

Every program is identified by a unique number. This number is controlled by the SPHERES team so that the ISS GUI can identify the program currently loaded in a satellite and indicate the name of the program to the astronaut. The ground-based GUI shows the integer identifier so that scientists also know which program is loaded. Neither the SCS nor the interfaces control the sequence of the program ID, which means that the SPHERES team members must manually ensure the numbers uniquely identify a program readied for tests aboard the ISS.

The system module also provides a high-level interface to enable and disable interrupts following the guidelines of the DSP/BIOS kernel. It includes a function to perform atomic memory copy by handling the interrupts automatically.

The module also provides an interface to the physical properties of the satellite, including:

- satellite dry mass
- full tank propellant mass
- estimated total wet mass given current propellant consumption
- satellite inertia matrix and its inverse
- satellite center of mass (dry and wet)
- model of the thrusters: strength, direction, and location

Source Files

- init_sphere.c
- fpga.c
- main.c
- system.c
- spheres_physical_parameters.c

Internal Header Files

- init_sphere.h
- system_internal.h

Public Header Files

- fpga.h

- spheres_physical_parameters.h
- SMT335Async.h
- spheres_constants.h
- spheres_types.h
- system.h

G.2.2 Control

The control module uses two threads to implement a periodic routine which allows substantial calculations at a user-selectable rate. Rather than using a hardware interrupt based on the hardware times, which is commonly done in embedded control systems, SCS utilizes two software interrupts. Using HWI would give the controller a high priority and prevent any other threads from executing while the control algorithm executes, unless special steps are taken to enable certain levels of preemption within the hardware interrupts. By using the software interrupts provided by DSP/BIOS, the SCS can easily configure the priority of the control interrupts and enable preemption by processes with more strict real-time requirements or higher rates. The use of SWI also makes a hardware timer available for other functions (it is used to control the timing of writes to FLASH memory).

The two threads which implement the control module are presented in Figure G.14. Their functions are:

- **Control dispatch** - This thread executes at a constant 1kHz regardless of the state of the satellite. Its purpose is to provide a simple interface to change the rate of the controller with period increments of 1ms independently of the controller itself. It also forms part of the synchronization routines so that multiple satellites start the tests at the same time.

  When a test start command is received by the communications module, it indicates to the software dispatcher that a test will start. The dispatcher then waits one second (1000 cycles) before starting the test, which gives the communications module enough time to acknowledge the command.

  The thread maintains a local timer to control the period at which the controller software interrupt is posted. Once the dispatcher posts the controller SWI, the controller will execute as soon as no other higher priority tasks are pending, usually within micro seconds. Because the dispatch thread executes continuously, the controller interrupt will be posted regardless of the state of the satellite. It is the controller SWI that determines what action to take, not the dispatcher.

  The dispatcher also maintains the maneuver time when a test is running.

- **Controller** - This thread executes upon being posted by the dispatcher. It is a state machine which implements the test management functions of the SCS. Figure G.15 illustrates the state machine used by the controller thread. It has the following states:

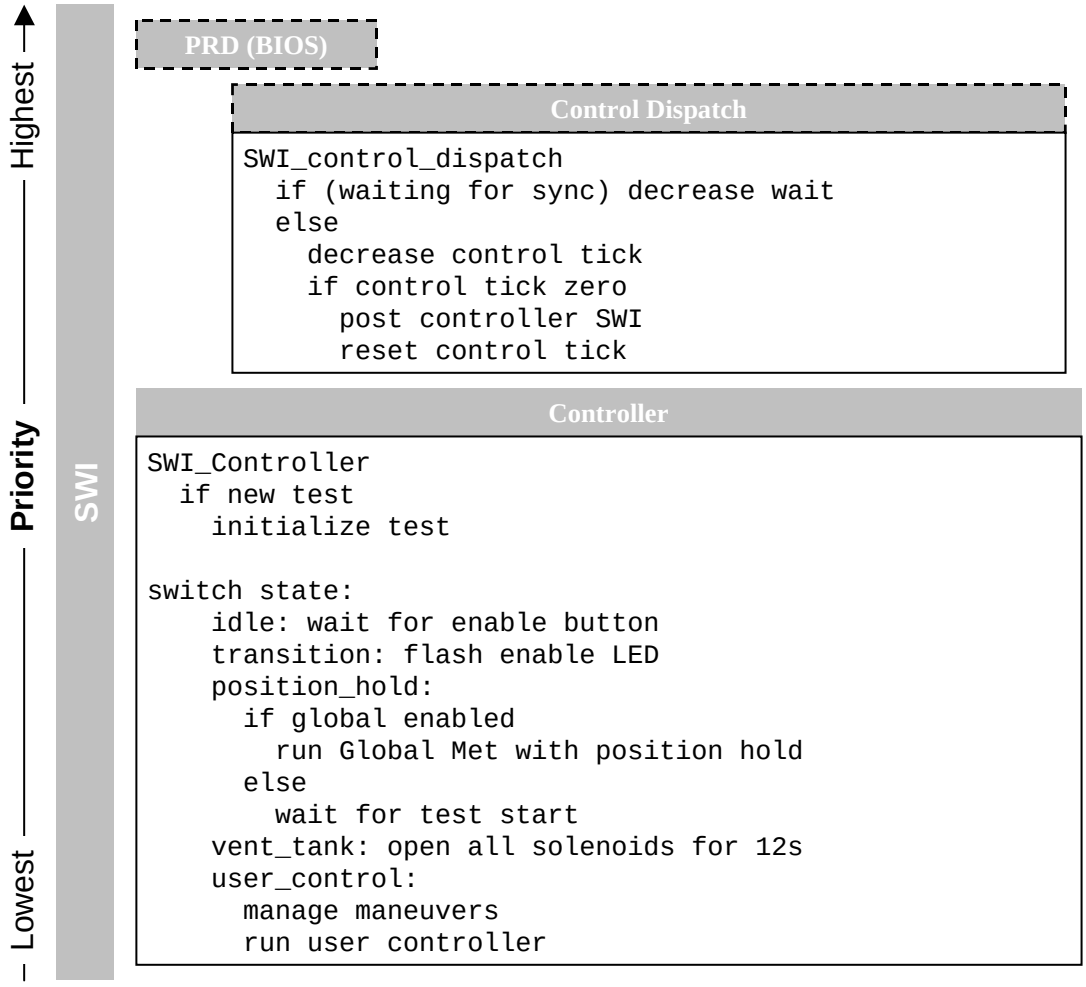


Figure G.14 SCS controller module threads and general algorithm

- **Idle** - When idle, the controller performs no actions. It waits for a new test to start or the Enable PB to either enable the satellite (depressed for one but no more than two seconds) or command a tank vent (depressed for more than five seconds). The idle mode is indicated in the SPHERES control panel by the Enable LED being off.

  To start a test the operator must first enable the satellite by using the Enable PB so the state goes to the Position Hold / Ready state. Scientists can bypass the need to enable a satellite for tests in ground-based facilities where starting a test is time-critical (e.g., the RGA). This feature is also useful during operations at the MIT SSL where dozens of tests are conducted in series to debug a program. The SPHERES team will ensure that enabling a satellite is not bypassed for tests aboard the ISS.

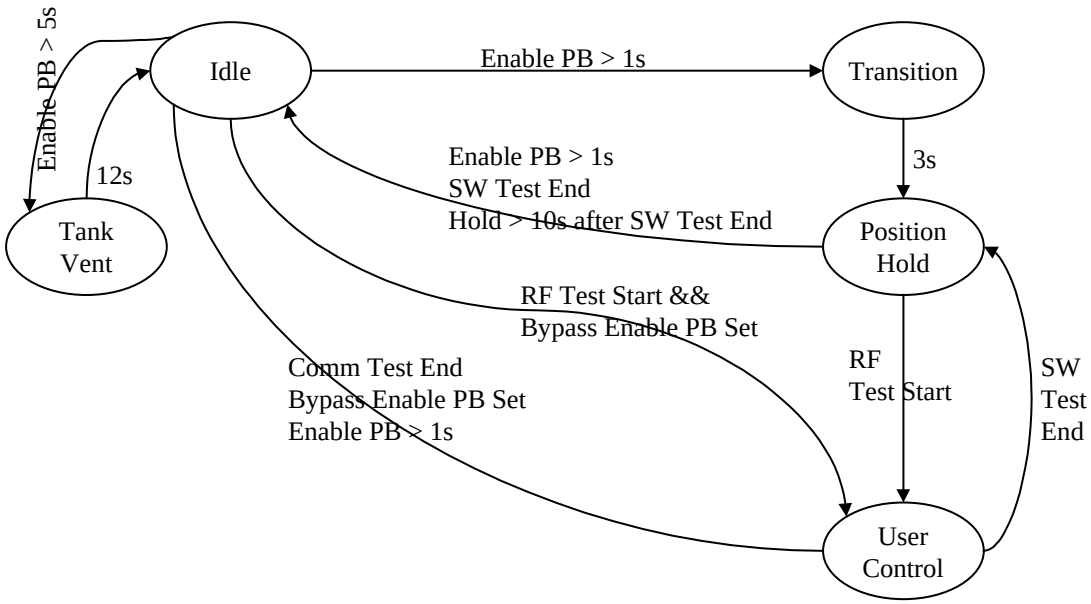


Figure G.15 SCS controller state diagram

- **Transition** - If the Enable PB is used to enable a satellite for tests, the controller enters a transition state to allow the operator to deploy (let go of) the satellite prior to entering the Position Hold mode. The transition state is fixed at three seconds, and changes automatically to the Position Hold state after the pause. The transition mode is indicated in the SPHERES control panel by a flashing Enable LED.
- **Position Hold / Ready** - This state can operate in two ways, depending on whether the global metrology system and the MIT estimator are available or not.

  If the global metrology system and the estimator are available, then the satellite will measure its position when released by the operator (after the three second transition) and maintain that position until a test is started. This helps operators locate multiple satellites without having to worry about drift.

  When the global system and/or estimator are not available, the satellite performs no actions.

  If the Enable PB is not explicitly bypassed in the program, the satellite must be in Position Hold / Ready mode before a test is started.
- **User Control** - Once a new test is commanded and the unit is enabled (or bypass enable is selected) the controller first runs the test initialization routines. These include management of local variables, including the change of state to user control, as well as executing the GSP test initial-

ization functions. Because these functions are executed within the controller SWI, they must run in the time allotted to one control period.

After the test is initialized, the controller performs maneuver management functions to maintain the maneuver number and time, it then executes the GSP controller functions.

The controller changes state again after a test has ended. If the GSP algorithm indicates a successful test and the global metrology and MIT estimator are available, the satellite will enter Position Hold mode for 10 seconds to cancel any residual velocity from the test. After the 10 second position hold, the satellite returns to Idle.

If the test terminates successfully but there is no global metrology and/or the MIT estimator is not available, the satellite returns to Idle.

If the test is interrupted using the Enable PB or via a wireless command, the satellite returns to Idle immediately.

- **Tank Vent** - This state opens all the solenoid valves so that a tank is completely empty before its removal. The state terminates automatically after 12 seconds of firing the thrusters and always returns to Idle.

Source Files

- control.c

Internal Header Files

- control_internal.h

Public Header Files

- control.h

G.2.3 Propulsion

The propulsion module creates a software interface to control the thruster solenoid valves with one millisecond increments. The module utilizes one main thread, CLK Propulsion, to control the state of the solenoids. Because the thrusters create white noise which triggers the ultrasound receivers of the metrology system, the thrusters cannot be active during a global metrology cycle. The global metrology process is initiated by an infrared pulse. Therefore, the IR Rcv HWI affects the propulsion module directly because upon IR reception the thrusters are turned off until the Metrology module indicates that the global metrology cycle has finished. Figure G.16 presents the two processes used in the propulsion module.

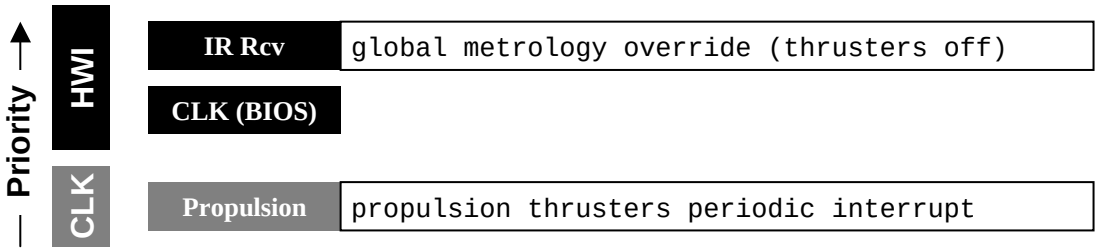


Figure G.16 SCS propulsion module threads

The propulsion CLK interrupt interfaces with the rest of the modules using an array of on/off times for each thruster. These times are relative to the first time the array is read by the propulsion interrupt, they are not relative to the satellite, test, or maneuver times. Therefore, when any other module (usually the controller) commands a set of on/off times to the propulsion module, it is equivalent of sending the command to an external module which is not synchronized with the rest of the system. The timing of the propulsion module is illustrated in Figure G.17.

By interfacing the propulsion module though an array of on and off times, the scientist can simulate several types of discrete control actuators. In its simplest form, as illustrated in Figure G.17, it commands on/off pulses which start at the same time and end at different

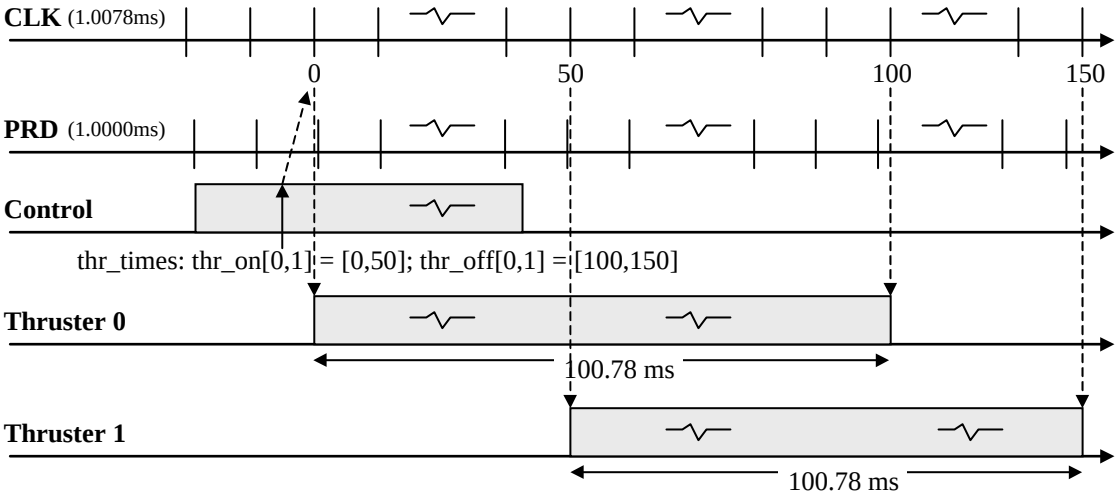


Figure G.17 Propulsion module timing diagram

times. But the scientist can also create algorithms to center the pulses, have them at the end of a period, or mix them. This allows the implementation of several types of modulation, including pulse width and frequency modulation. These possibilities are pictured in Figure G.18

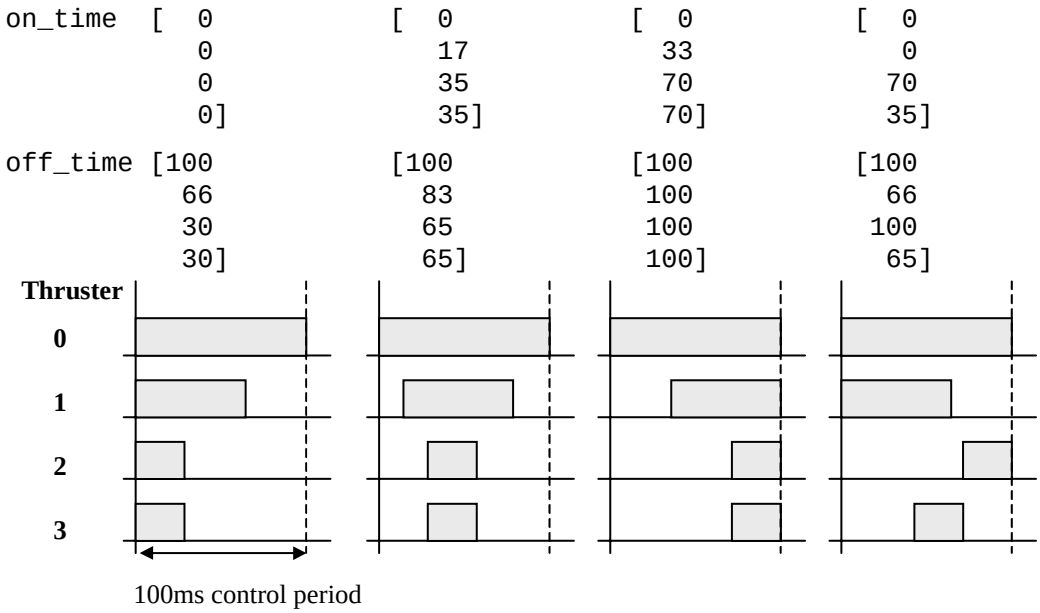


Figure G.18 Propulsion modulation options

Source Files

- prop.c

Public Header Files

- prop.h

G.2.4 Communications

The communications module implements the TDMA protocol and provides both high priority and low priority queues for data transmission. The module uses several synchronization and data management tools provided by DSP/BIOS to manage the data securely in a multi-thread environment. Appendix H presents details on the TDMA protocol implementation and the SPHERES data packets. This section describes the data transfer and processing between threads of the SCS communications module.

Figure G.19 shows the threads used by the communications module. The module separates the reception and transmission tasks in high priority interrupts, but joins them in the background communications management task which provides the actual interfaces to the communications module.

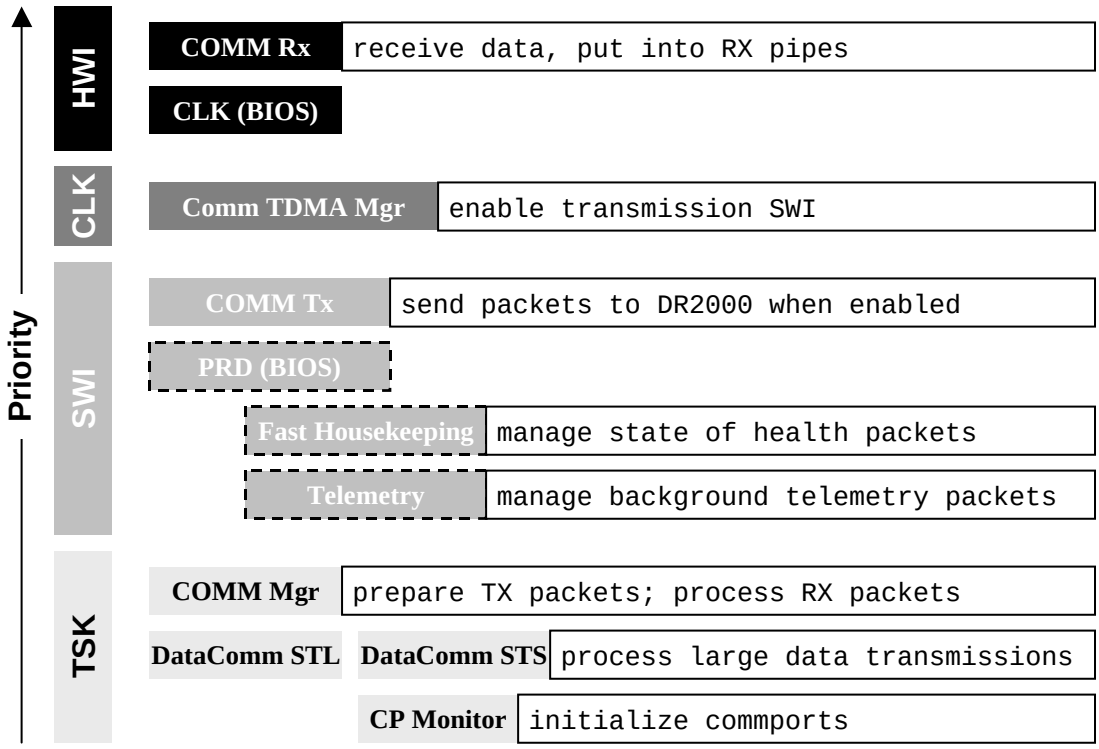


Figure G.19 SCS communications module threads

Data Reception

Figure G.20 presents the data reception processes. When data is received by the comports, a hardware interrupt is posted. The reception hardware interrupt collects the data and stores it in a temporary buffer. This procedure *does* process the data to identify complete packages, and only places complete packages in the pipes of lower priority processes. The reception interrupt also identifies commands to start a new test, since it resets the synchronization of the control periods between multiple units (the wait in the control dispatcher). It does not perform any other data processing; interpreting all other commands and data is done by other threads.

The reception HWI places complete packets in a *pipe* construct. Pipes are data management tool provided by DSP/BIOS which implement queues and call data processing functions automatically. The DSP/BIOS PIP module, used by the SCS, accounts for the multithreaded nature of the system. When a packet is placed in the pipe, a semaphore is posted to the communications management task to indicate that new data is waiting.

The communications management task works in the background of all real-time processes to handle commands from the laptop and identify packets from other satellites. The communications module of the SCS handles the following types of packets:

- **Telemetry** - the state information of other satellites is placed in local variables
- **State of health** - the state of health of other satellites, including their role, is stored in local variables
- **General commands** - commands from the ground station, described below
- **Beacon initialization** - the configuration of the global metrology system is uploaded by the ground station
- **Large data transfer** - packets that form a custom data transmission of the guest scientist

All other types of packets are ignored by the SCS.

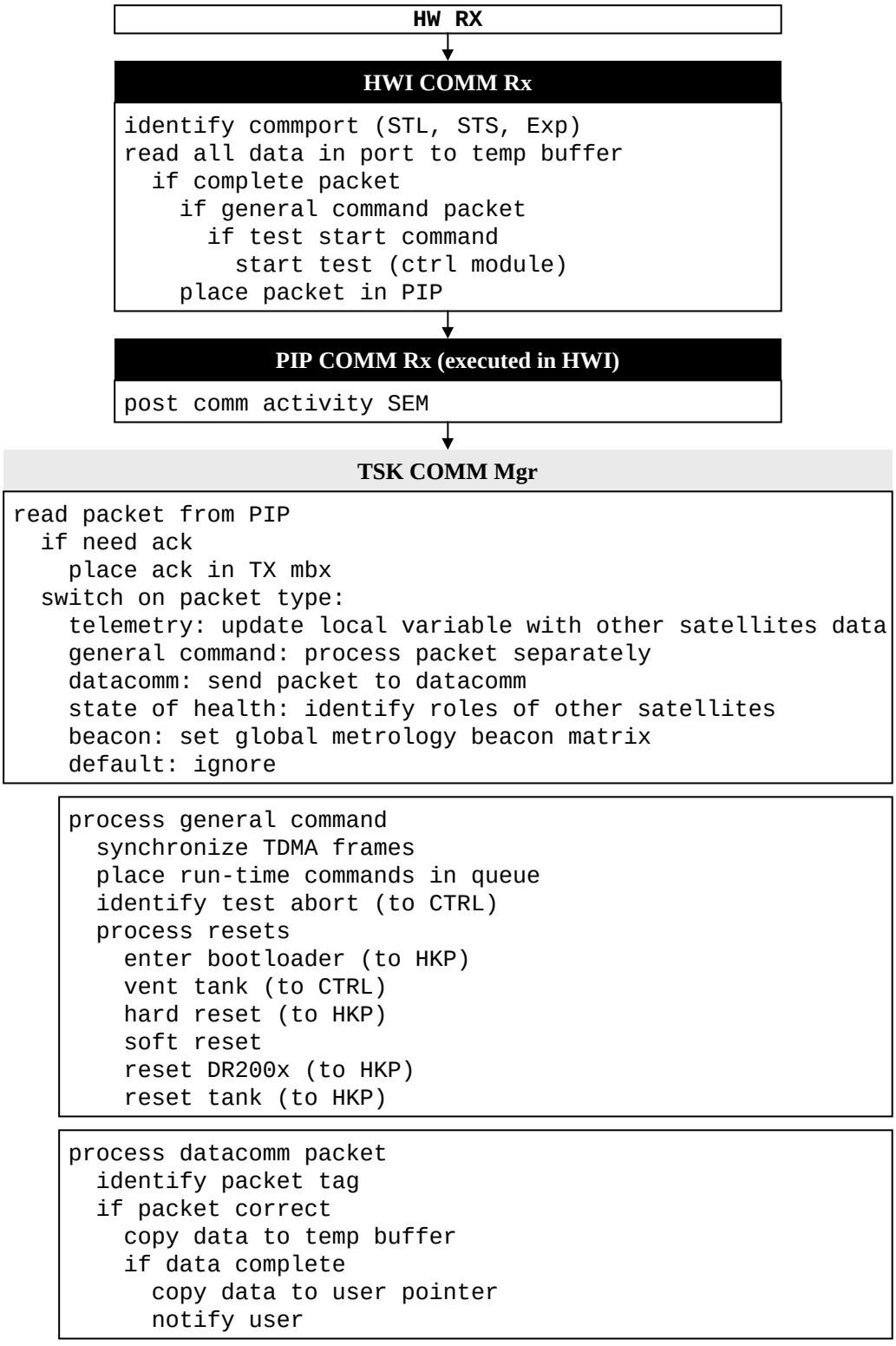


Figure G.20 Communications data reception process

Scientists can transfer data between satellites by using the large data transfer tools of the communications module, even if their data is smaller than a standard SPHERES packet. These tools automatically format the data for transmission and re-incorporate the data upon reception by the intended satellite. Scientists must use this tool, since unknown packet types are ignored by the communications module and are not available to the scientist.

General command packets from the ground station are processed in this task, except for a test start, which is handled in the hardware interrupt to synchronize the satellites. The commands are actually executed by other modules (housekeeping, control), but are triggered by this process.

Data Transmission

Figure G.21 presents the processes used for data transmission. The transmission of data must only occur during the TDMA window assigned to the satellite to prevent contention in the wireless network. A periodic hardware interrupt, the Comm TDMA Mgr CLK process, times the length of the TDMA windows. The start time of a TDMA cycle is commanded by the reception of a general command packet from the ground station; the TDMA manager process records this start time and opens the transmission window by posting a semaphore to the communications manage task. The communications management task then posts the Comm TX SWI if data needs to be transmitted. The Comm TX SWI sends one complete packet at a time, since the hardware requires that packets be delivered without long interruptions (no more than 2ms between bytes).

As illustrated in Figure G.21, transmission data can be created by a wide range of processes. The periodic housekeeping and telemetry tasks create state of health and state telemetry packets continuously. The controller send a confirmation every time a test is started. The scientist can create data in practically any process, including the periodic controller, program and test initialization, and background tasks. This data can be in the form of standard SPHERES packages by using the publicly available function

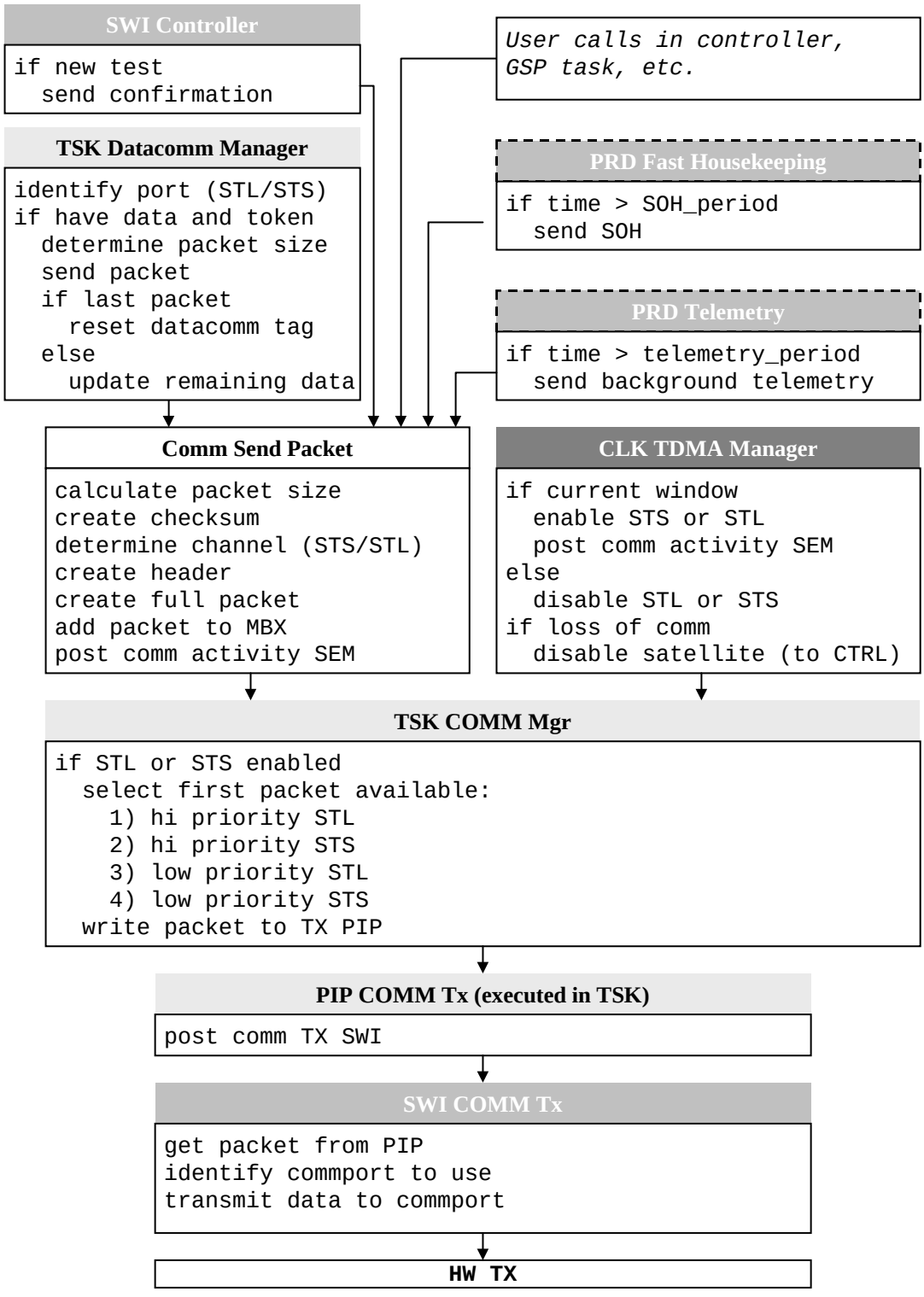


Figure G.21 Communications data transmission process

`CommSendPacket`, or by using the large data transfer tools (datacomm, which used send packet itself). The interface function `CommSendPacket` is the only interface to the communications transmission procedures to ensure that the TDMA protocol is maintained.

The `CommSendPacket` function, which can execute in any type of thread, utilizes the DSP/BIOS supplied mailboxes to store the data for the communications management thread to process. Mailboxes provide the necessary atomic operations and controls to maintain the data safe regardless of what type of thread made the post to the mailbox. After placing the data on the mailbox the function posts a semaphore to indicate to the communications management task that new data is ready for transmission.

The communications management task checks four mailboxes for data transmission when a TDMA window is available: a high and allow priority mailbox for each of the STS and STL channels. Figure G.21 indicates the order in which the mailboxes are searched. When a packet is found, the task sends the data to the transmission pipe. This causes a Comm TX SWI to be posted until the pipe is empty; the Comm TX SWI transmits the packet to the hardware.

Source Files

- comm.c
- comm_interrupt.c
- comm_process_rx_packet.c

Internal Header Files

- comm_internal.h
- comm_interrupt.h
- comm_process_rx_packet.h
- commands.h

Public Header Files

- comm.h

G.2.4.1 DR200x Driver

The communications module also implements several procedures to configure and interface with the commports and the DR200x modules. The functions of the communication driver sub-section are to:

- Manage the communications port
  - Read input buffers
  - Empty input buffers
  - Write to output buffers
  - Prevent output buffer overflow
- Send initialization commands to the DR200x
  - Send commands
  - Wait for response
- Reset DR200x modules

Source Files

- comm_driver.c

Internal Header Files

- comm_driver.h

G.2.4.2 Background Telemetry

The background telemetry sub-module manages the transmission of state information between satellites automatically at a specified rate. The scientist can define the state vector variable to use for state transmission at the start of a program or test. The background telemetry functions will then transmit that state periodically without further intervention by the scientist. The procedures automatically stores the received information in local structures, which can be accessed by the guest scientist using the function `commBackgroundStateGet`. The background telemetry utilizes the following thread:

- **PRD Telemetry** - sends the telemetry information at a periodic rate
- **TSK Comm Mgr** - processes received packets by executing the background telemetry unpack function

Source Files

- comm_background.c

Internal Header Files

- comm_intrnal.h

Public Header Files

- comm.h

G.2.4.3 Datacomm

The datacomm sub-module allows scientists to transfer data or arbitrary size and format to ground and between satellites. The datacomm functions split large packets into standard SPHERES packets, manage the transmission of the multiple packets, and (if necessary) assembly the original data structure in the receiving satellite. To use the datacomm functions the scientist must initialize the transmission and reception buffers at the start of a program, then they only need command a new transmission; reception occurs automatically. The datacomm system allows scientists to poll the state of a transmission. The module triggers the GSP task when new data has been received.

The datacomm sub-module utilizes a background task for each channel to divide and transmit packets:

- **TSK_gspdata_manager** - Implements a heuristic leaky-bucket scheme to manage flow control between multiple datacomm transmission requests. This allows multiple datacomm request to be made simultaneously.

Data reception procedures are executed completely within the general communications management task.

To utilize datacomm scientists use the following procedures:

- Sending satellite
  - `datacommSendData (tag, *buffer, size, channel, to, mode, *TX_done_flag);`

    `tag` - unique identifier of the data

    `*buffer` - pointer to data location

`size` - size of the data in bytes

`channel` - STS or STL channel

`to` - destination (use 0 to broadcast)

`mode` - low or high priority

`*TX_done_flag` - user flag to poll until transmission is done

- Receiving satellite (optional)
  - In the program or task initialization (but **not** in test initialization), allocate space to assemble new data:

    ```
    datacommInitializeTag(tag, buffer_size, timeout);
    ```

    `tag` - unique identifier of the data

    `buffer_size` - size, in bytes, of assembly buffer (size of data transfer)

    `timeout` - timeout, in milliseconds, to cancel assembly of this tag

  - Before the data is received (i.e., preferable in the initialization routines), the scientist must allocate space for the assembled data to be copied into by using the following function:

    ```
    int datacommRegisterBuffer(tag, *buffer);
    ```

    `tag` - unique identifier of the data

    `*buffer` - pointer to the destination memory space allocated by the scientist

Source Files

- comm_datacomm.c

Internal Header Files

- comm_datacomm_internal.h

Public Header Files

- comm_datacomm.h

G.2.5 Metrology

The metrology module performs two tasks: collects metrology data and provides the necessary threads to process this data. To achieve this, the metrology module utilizes high priority interrupts to collect the data and low priority tasks to enable processing, as illustrated in Figure G.22. The metrology module utilizes the following threads:

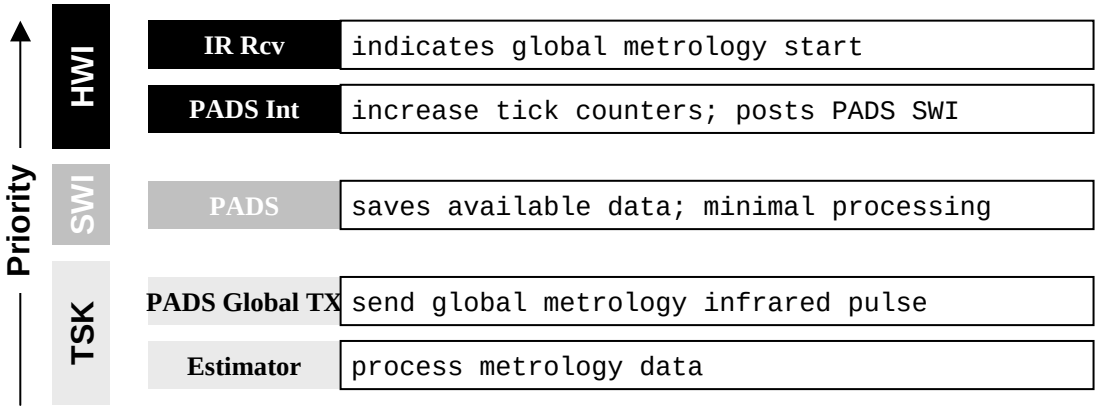


Figure G.22 SCS metrology module threads

- **Hardware Interrupts**
  - **IR Rcv -** The reception of an infrared pulse indicates the start of a global metrology cycle. The metrology FPGA collects the time-of-flight measurements of the ultrasound signals; to achieve the highest precision possible, the IR signals are connected to the FPGA externally via hardware, independently of the DSP. Actually, it is the FPGA which creates the signal that interrupts the FPGA. The IR Rcv hardware interrupt allows the SCS to transmit commands to the onboard beacon or external expansion items and to turn the thrusters off. Because no other tasks are performed, this interrupt is relatively short and very fast.
  - **PADS Int** - The hardware metrology interrupt is essential for the operations of the SCS. The FPGA creates a precise 1kHz clock used to control the satellite system time, which is maintained through the PADS Int HWI. The same interrupt drives the periodic software interrupts created by DSP/BIOS (PRD). But the interrupt does not perform any metrology data handling or processing directly, instead, it posts the PADS software interrupt, which collects the data.
- **Software Interrupts**

  - **PADS -** The PADS SWI collects the data available in the FPGA. The FPGA will always interrupt at 1kHz and will always have IMU data available. But it will only have global metrology data available during a global cycle. Therefore, the PADS SWI checks the status of the FPGA to determine whether to save global data or not.

- **Tasks** - The availability of two tasks for data processing enable scientists to compare their algorithms with the standard SCS estimators.
  - **Estimator** - The estimator task is reserved for use with the SPHERES standard estimator which is part of the SCS. The estimator task, which runs in the background, perform both long calculations with the global metrology data and short updates with the IMU data.
  - **GSP Task** - The GSP task is provided to scientists to handle multiple events, including the reception of metrology data (the full range of events is described below, in the GSP section). Scientists can implement their own algorithms in this thread utilizing any combination of inertial and global data.
  - **Global TX** - This task sleeps in the background until triggered to command an infrared transmission. The transmission of an IR should only be done by one satellite, although there are no restrictions from the SCS programming.

The flow of data through the metrology system is illustrated in Figure G.23. The inertial and global data are transmitted through the FPGA to the DSP via the same HWI/SWI combination; the software checks which data is available. The SWI makes the data available to the scientist. The PADS SWI collects all the inertial data and immediately converts the digitized analog values to floating point values in units of [m/s^2] for accelerations and [rad/s] for rotation rates. The inertial data is stored in buffers configured during program initialization; the scientist can choose to receive the data at any rate in period increments of one millisecond. Further, the scientist can receive all the data (e.g., an array of 10 data sets every 10ms) or just the single data collected at that rate (e.g., one array of data every 10ms). The global metrology data is transferred as N packets of 24 measurements and one time stamp (i.e. 25 words total), where N is the number of beacons programmed during program initialization. The GSP interface functions to the inertial and global data execute within the SWI, therefore these functions must completely quickly.

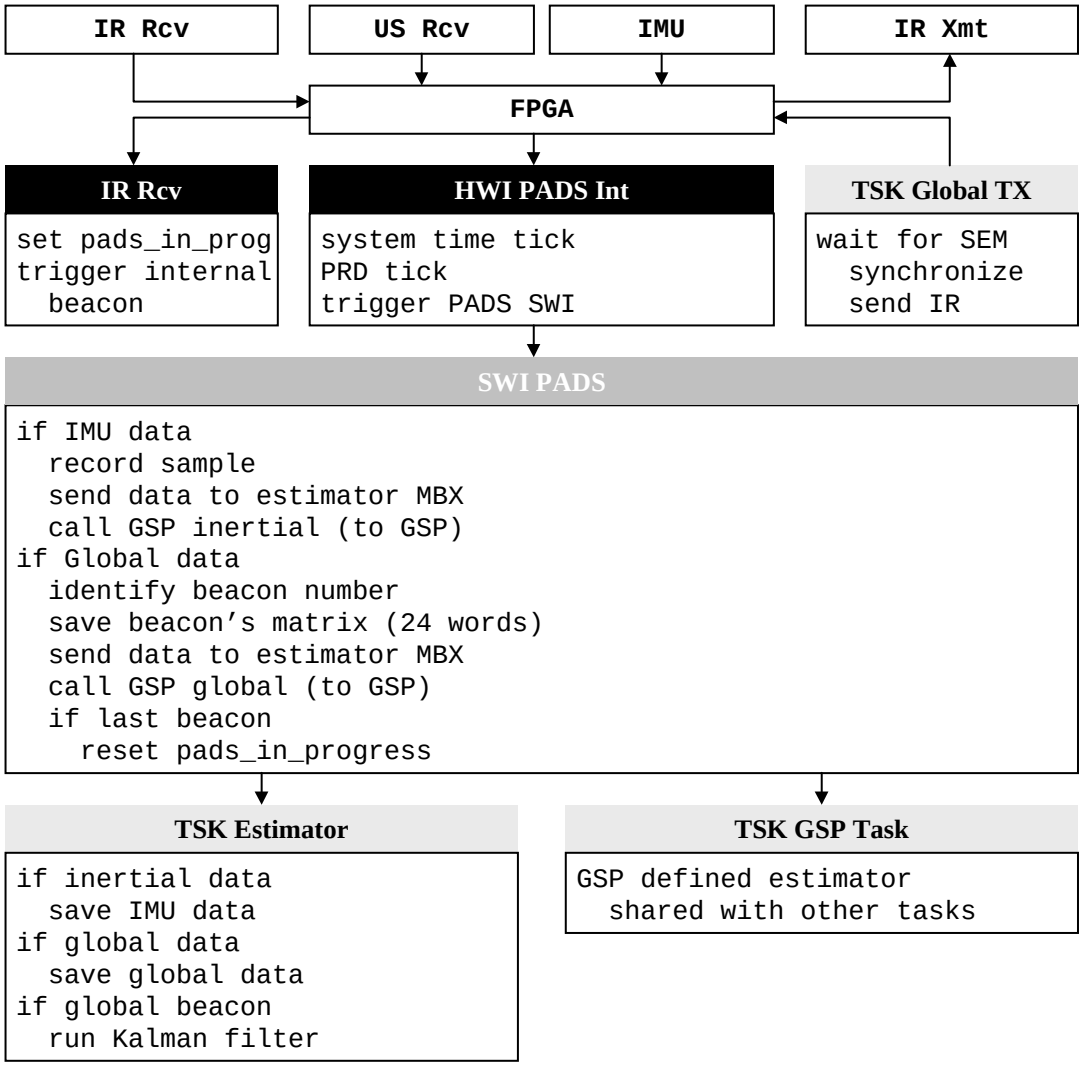


Figure G.23 SCS metrology module general algorithms

The data for the standard SPHERES estimator is placed in DSP/BIOS mailboxes. The estimator task pends on those mailboxes and executes when data is available. This allows the estimator task to remain asleep in the background when no data exists; the use of mailboxes allows the estimator task to run for extended periods of time without loosing data. Therefore, the estimator can process global metrology data over extended periods of time and then use all the inertial data collected throughout the global metrology processing period.

The interface with the GSP uses a semaphore. The semaphore is posted consecutively, but it is the responsibility of the scientist to save the data through the inertial and global data collection functions, since the SCS will not save the data for the scientist otherwise. Figure G.24 presents a sample timing diagram of the scheduling of the metrology treads during both inertial and global data cycles.

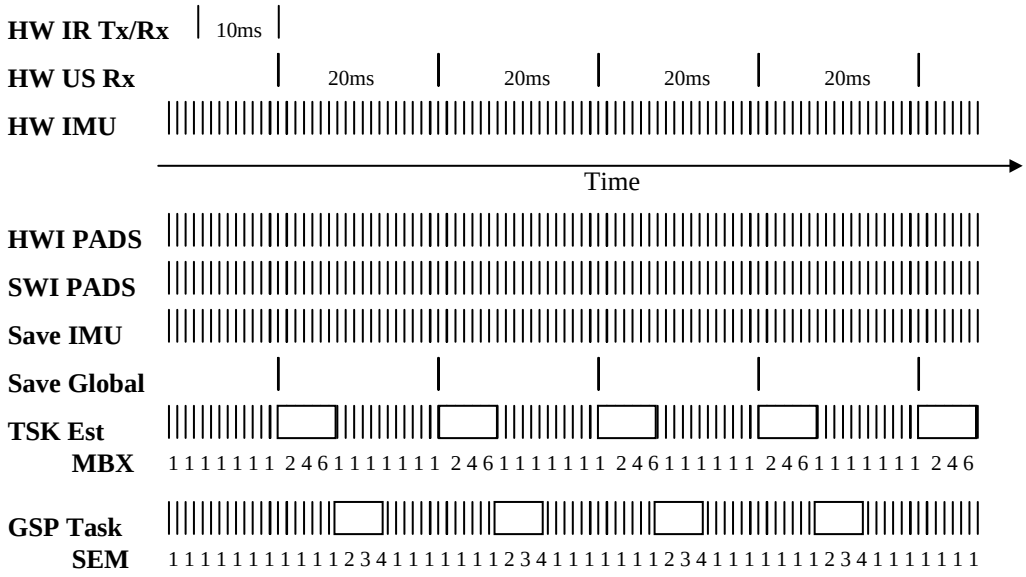


Figure G.24 SCS metrology treads scheduling

Source Files

- pads.c
- pads_correct.c
- pads_estimator.c
- pads_request.c

Internal Header Files

- pads_internal.h
- pads_correct.h
- est_USsubfunc.h

Public Header Files

- pads.h

G.2.6 Housekeeping

The primary function of the housekeeping module is to maintain the ground station informed on the state of health (SOH) of the satellite and save information to the FLASH loader variables. The housekeeping module also checks the status of the commports during boot time and then periodically throughout operations. As shown in Figure G.25, three threads are used to perform these functions:

- **Fast Housekeeping PRD** - The main housekeeping thread, it collects the information periodically and sends out the state of health packets. Because the SOH packets are used to acknowledge commands from the ground station, the housekeeping task is also used to acknowledge commands.
- **FLASH TSK** - Because the boot loader program resides in the FLASH, it is essential to protect that space in memory. Therefore, the SCS provides a single-point interface to the FLASH via this task. The task filters the write addresses to ensure that programs do not overwrite the boot loader (although the could overwrite themselves). This task implements the time delays necessary between FLASH write cycles.
- **CP Monitor TSK** - This task is started upon boot to check the interrupt flags of the communications ports, which exhibits a race condition after a hard reset. The task resets the commport interrupt flags. During operations the task checks that the status of the commports corresponds to the indicated interrupt flags approximately once a second.

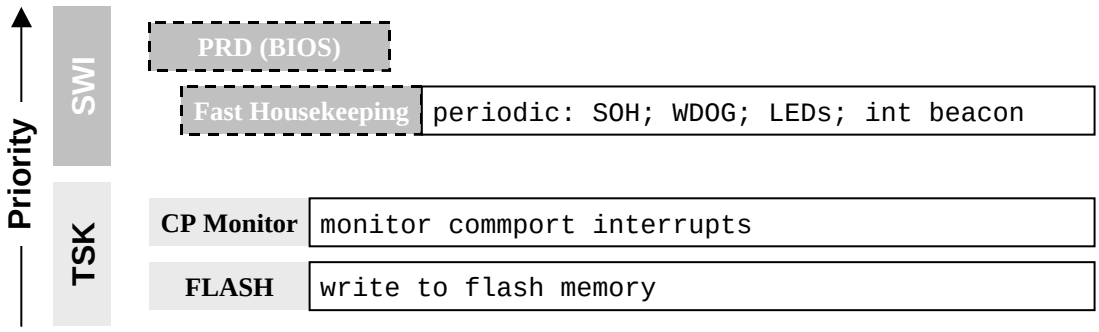


Figure G.25 SCS housekeeping module threads

State of health packets are created and transmitted at 1Hz by default. To create the SOH packets, the housekeeping module collects information from the other modules. The state of health of the satellite is collected as follows:

- System time (from system)
- Program ID (from system)
- Tank usage (from propulsion)
- Test time (from system)
- Maneuver time (from control)
- Last test result (local, set by control)
- Number of received IR pulses (from metrology)
- Communications status (from communications)
- Beacon information (from system)
- Controller state (from control)
- Acknowledgement (local, commanded by communications)

The housekeeping periodic interrupt provides the software interface to the metrology FPGA digital outputs which consist of the Enable LED, the internal watchdog, and the internal beacon. The housekeeping interrupt controls the state of the Enable LED on the SPHERES control panel based on the status of the controller. It provides a software interface to the beacon control which can be used in any other module.

The interface of the watchdog is of special importance, since the watchdog control signal must be continuously flipped to prevent the watchdog circuitry from forcing a hardware reset. If the housekeeping task does not respond (any task with higher priority does not return control) the system will reset. This function is not performed at the lowest priority because it is possible that metrology or autonomy algorithms which run in background tasks take several seconds, which would cause a hardware reset.

The fact that the housekeeping interrupt controls both the watchdog and the FLASH loader variables is used to provide two functions to the satellites: the ability to force a hardware reset and to enter boot loader mode automatically. To force a hardware reset the

housekeeping thread is commanded to no longer flip the watchdog control line. To enter boot loader mode, the housekeeping thread first writes the boot loader variables with `Enter_boot` set to `0x01`, and then forces a hard reset.

By default the housekeeping task will save the following information once a second to the FLASH:

- Tank time
- Satellite ID (if changed)
- Beacon locations (if changed)
- IMU calibration data (if changed)
- Enter boot (if commanded)

This information is shared by any program since the physical configuration of the satellite (Satellite ID, IMU calibration data) does not change between programs. Further, since the beacon locations are expected to be constant each test session, they only need to be uploaded when the first program is used. By saving the tank time once a second, the housekeeping module maintains a reasonable estimate of tank usage even if the unit is reset.

Source Files

- housekeeping.c
- util_FlashLib.c
- util_BranchTo.c

Internal Header Files

- housekeeping_internal.h
- util_FlashLib.h
- util_timing.h

Public Header Files

- housekeeping.h

G.2.7 GSP Interface

The scientist code interfaces with the other modules through the GSP interface. The other modules include calls to the functions of the GSP module, so that scientist can concentrate all their work in one module. The GSP interfaces with the control and metrology modules, operating as part of the PADS SWI and the Controller SWI. All the modules can post a semaphore to trigger the GSP Task, although the scientist can select which events trigger the task. These interfaces are illustrated in Figure G.26.

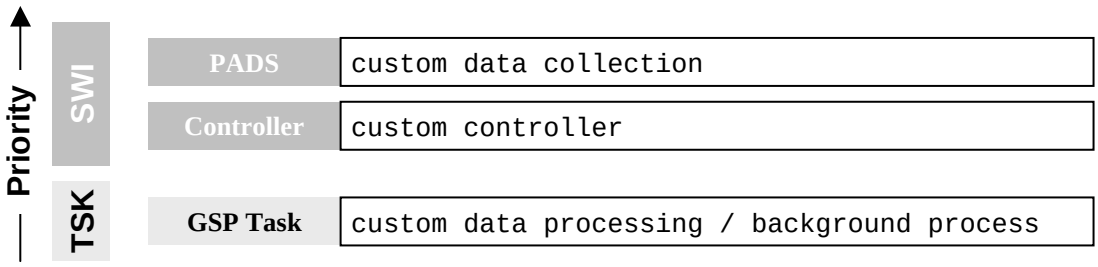


Figure G.26 SCS GSP module threads

The GSP functions available to scientists are:

- **Program Initialization**
  - **gspIdentitySet** - Sets the *logical* identity of the satellite.
  - **gspInitProgram** - Run *once* during boot time, before the DSP/BIOS real-time environment is set, to initialize global variables and perform other functions required before the real-time system is started. These include:
    - Initializing the TDMA windows
    - Allocating the buffers for inertial metrology data
    - Configuring the metrology FPGA for the global metrology setup
    - Setting metrology (inertial and global) rates
    - Setting the background telemetry period
- **Control**
  - **gspInitTest** - This function is run once at the start of each test. The function should initialize any variables used in the GSP control algorithms and set the desired control period for that test.

- **gspControl** - This function implements the control algorithm for the test. It can be programmed directly or it may call other functions developed by the scientist.

- **Metrology**
  - **gspPadsInertial** - This function should collect the data and store it for later processing, since the function is run within the 1kHz PADS SWI. If the scientist can perform quick calculations with the inertial data, this function can also update the system state using the inertial data. If the GSP task takes extended periods of time, this function should provide structures to save data even if the task is not available.
  - **gspPadsGlobal** - This function should only collect the global metrology data and store it for later processing because it is run within the 1kHz PADS SWI. If the GSP task takes extended periods of time, this function should provide structures to save data even if the task is not available.

- **Task**
  - **gspInitTask** - This function executes once when DSP/BIOS sets up the real-time environment, including the GSP Task thread (i.e., it runs shortly after `main()` terminates). The function should initialize any variables needed by the task, as well as setup the masks for events that will trigger the task.
  - **gspTaskRun** - This function executes in the GSP Task thread when an event occurs that posts a semaphore. The possible events which can start this task are:
    - `CTRL_DONE_TRIG` - a controller period is done
    - `DATA_TX_DONE_TRIG` - datacomm has finished transmitting data
    - `DATA_RX_DONE_TRIG` - datacomm has finished assembling data
    - `PADS_GLOBAL_START_TRIG` - an IR pulse was received
    - `PADS_GLOBAL_BEACON_TRIG` - global metrology data received
    - `PADS_INERTIAL_TRIG` - inertial metrology data received
    - `PADS_ESTIMATOR_DONE_TRIG` - the default estimator is done
    - `TASK_TIME_TRIG` - the task sleep period is over
    - `TEST_START_TRIG` - a test was started
    - `GSP_USER_TRIG` - user trigger

    Scientists can select which events will trigger the task by setting a mask at during the task initialization.

> Since all of the events share the same task, the scientists must carefully use this resource and realize that the task may not respond to an event in real-time. For example, if the task is used with a custom estimator, and it is also used to perform post-control data processing, the post-control functions may not be carried in real time. Further, the task has a maximum semaphore depth of 16, therefore no more than 16 events will be accounted for and new events are lost.

Scientists can use any of the functions defined in the *public* header files of the other modules. These provide scientists with full knowledge of the state of the satellite and interfaces to all the hardware.

G.3 Standard Libraries

The following section describe the available standard functions at the time of print of this thesis. These functions can be used by scientists to implement simple algorithms outside their area of interest so that they may concentrate on development of their program. They also serve as baseline algorithms for scientists to compare their results.

G.3.1 Controllers

- **Angular control** - 3D controllers to apply proportional/derivative (PD) control of angular position and control with respect to the global metrology frame with the option to specify the gains.
- **Position control** - 3D controllers to apply proportional/derivative (PD) and proportional/integral/derivative (PID) control laws to the position and velocity of the satellites with respect to the global metrology frame with the option to specify the gains.

  Also provides a function to transform delta-V commands into x-y-z body forces, which can then be transformed to thruster on/off times by the mixers.
- **Switchline control** - 3D switchline controllers for position and attitude control. Two versions are available: one with coupled position and attitude and one with decouple position and attitude.

G.3.2 Estimators

- **Extended Kalman Filter** - An extended Kalman filter which utilizes both inertial and global data to estimate the position and attitude of the satellite in the global frame.
- **Range and bearing** - An extended Kalman filter to obtain the range and bearing between two satellites which use their internal beacons for global metrology. This provides relative state information in multi-satellite systems.

G.3.3 Maneuvers

- **Regulation** - Regulates a constant rotation about one axis while maintaining the satellite in the same position.
- **Open Loop Translation and Rotation** - returns the necessary thruster on/off times for an open loop translation or rotation with respect to the body frame.

G.3.4 Mixers

- **Standard Mixer** - Creates a set of thruster on/off times based on input force and torque parameters, duty cycle, and control period.
- **Mixer with Thruster Correction** - Enhances the standard mixer by correcting for differences in the actual measured thrust of each thruster.

G.3.5 Terminators

- **Timed terminators** - Terminators to end a maneuver or test in a specified period of time after they start.

G.3.6 Math

- **Matrix and Vectors Manipulation Methods**
  - Square of a matrix (B = A * A)
  - Matrix times vector (c = A * b)
  - Matrix times matrix (C = A * B)
  - Matrix time matrix transpose (C = A * B') and (C = A' * B)
  - Matrix add (C = A + B)
  - Vector add (c = a + b)
  - Vector outer product (c = a * b)
  - Vector inner product (c = a' * b)
  - Calculate skew symmetric matrix of A
  - Normalize a vector
  - Calculate the magnitude of a vector
  - Invert a 3x3 matrix
  - Determination of the body to global rotation matrix
  - Determination of the rotation matrix for use with quaternions
- **Matrix Inversion Methods** - provides several methods to invert matrices, including: Cholesky decomposition and LU decomposition.
- **Jacobi** - Computes the eigenvalues and eigenvectors of a matrix.
- **LTI Filter** - Implements a causal form II LTI filter.

G.3.7 Utilities

- **Data compression** - Provides utilities to collect large amounts of data from either the Global or the Inertial metrology system and compress the data before downloading it through the communications module.
- **Serial print** - Enables scientists to transmit serial data through the expansion port (in the satellites) or the simulation debug file (in the simulation) without having to use the standard ANSI C `printf` function which requires substantial memory space.

Appendix H

SPHERES COMMUNICATIONS

This appendix describes the communications system for the SPHERES experiment. It acts as a reference specification for communications between the laptop and the satellites. The interface specification has three sections. The first section describes the low-level interface with the DR200x modules. Next, the basic link-level interface (packet structure, protocols) is introduced. Then the interaction between the GUI and the satellites during testing is described.

H.1 DR2000 Configuration

The DR2000 firmware was customized for SPHERES to operate in *transparent* mode. Under this mode, the DR2000 will send data without checking any contents or formats, with only three important logical values: packet size and the to/from values. The DR2000 will consider as data any bytes sent to it and send them out 'as is' except for the escape sequence $$ at the start of a packet which indicates that the data after the escape sequence is a configuration word (described below).

H.1.1 DR2000 Packet Structure

The DR2000 packet structure consists of a pre-amble that excites the receiver crystal, a 5 byte header, the body of the packet, and a 2-byte CRC. Table H.1 lists the elements of the DR2000 packet. The pre-amble is hard-coded in the DR2000 firmware. The header is cre-

ated using the configuration of the DR2000 (which is stored in FLASH, but should be updated every time a project starts for data safety). The body are the bytes received by the transmitting DR2000. The CRC is created automatically from the data. Note that the body is the only thing that must be sent to the DR2000; the header and CRC's are completely transparent to the transmitter or receiver and are never seen by the receiver. Figure H.1 depicts the packet structure.

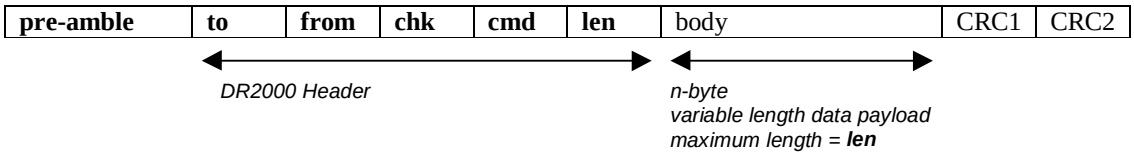


Figure H.1 DR2000 packet structure

TABLE H.1 DR2000 packet structure

Byte	Width	Name	Function
0 – 2	3	pre-amble	A hard-coded element of the firmware which sends a sequence of 0xAA and 0x55 bytes to excite the receiving satellite's crystal.
3	1	to	The intended recipient. Use 0x00 to indicate broadcast mode. Because setting the TO address takes approximately 400ms, the TO address should always be set to 0x0 (broadcast) except in special circumstances.
4	1	from	This satellite's ID. Only packets sent with the to equal to from or broadcast mode will be transferred out of the DR2000. All other packets will be discarded.
5	1	chk	8-bit checksum of the packet "body".
6	1	len	The length of the body in the packet. The maximum length is stored in the configuration, but this length may be variable in transparent mode.

TABLE H.1 DR2000 packet structure

Byte	Width	Name	Function
7 – (n+7)	n (0<n<256)	body	The body of the packet is transmitted ‘as is’ to the intended satellites. The body of a packet cannot start with the escape sequence $$, but it may otherwise contain any bytes.
(n+8) – (n+9)	2	CRC	A two byte CRC created out of the of the packet header and body (excludes pre-amble).

The *start of a packet* must be well understood, since an escape sequence in the middle of a packet is ignored. A packet is transmitted out of a DR2000 whenever:

- `"len"` bytes are received by the DR2000, in which case a packet is sent out immediately after calculating the CRC. Byte number "`len`+1" is considered the start of the next packet.
- There is a pause larger than 2ms between bytes. After the pause the DR2000 calculates the CRC for the small packet and updates the len byte in its actual transmission. The receiver will expect len bytes and extracts the body. Only the number of transmitted bytes are sent out of the receiver, the packet is not padded to complete the `len` specified in the receiver.

Therefore, it is important to ensure that a packet is sent continuously out of a serial port, without pauses of more than 2ms. While the receiving satellite may have timeouts that would allow it to reconstruct a packet that is divided among multiple DR2000 transmissions, there will be twice the DR2000 overhead.

H.1.2 DR2000 Commands

The DR2000 must also be configured to operate in the right mode and frequency prior to operation with the SPHERES system. This initialization occurs automatically in the SPHERES satellites, but must be performed individually in every external element (such as the communications laptop). Table H.2 lists the DR2000 commands that are pertinent to the SPHERES program; Table H.3 lists the valid hardware ID’s used in SPHERES.

TABLE H.2 DR2000 configuration commands

Command	Name	Description	SPHERES
$$TOADxx	To address	Set the TO address configuration in FLASH.	`xx = 00`
$$FRADxx	From address	This satellite's address. Table 3-3 lists the valid values in detail.	Ground: `xx = 30` Satellites: `xx = 31 – 39`
$$SIZExx	Maximum packet size	The maximum length of the body before the DR2000 immediately sends out a packet	`xx = 25`
$$RFMDx	DR2000 Mode	Changes the mode of the DR2000 between transparent (x=1) and structured (x=0).	`x=1`
$$RDSPx	RF baud rate	Changes the baud rate of the RF transmissions. Always use x=0 for 57.6kbps	`x=0`

TABLE H.3 Valid satellite IDs for the `to` and `from` fields

Satellite Name	HW Address (hex)
Broadcast	0x00
Laptop/Ground	0x30
SPHERE s/n 1	0x31
SPHERE s/n 2	0x32
SPHERE s/n 3	0x33
SPHERE s/n 4	0x34
SPHERE s/n 5	0x35
SPHERE Default[a]	0x39

a. A SPHERE satellite with corrupted memory will reset its satellite ID to 0x39. The satellite must then be reconfigured with a valid ID between 0x31-0x35.

These commands set the satellites to the default configuration every time they boot. All programs should do the same. After this is done once, there is no need to use these values again. As an example, the sequence used by SPHERE satellite with serial number 1 is:

```
$$RFMD1
$$TOAD00
$$RDSP0
$$SIZE25
$$FRAD31
```

H.2 Link-layer Interface

This section describes the low-level interface to the SPHERES communications system. SPHERES utilizes two communications channels (868MHz and 916MHz) operating at 56.7kbps. Nominally the 868MHz channel is used for satellites-to-laptop (STL) communication, while the 916MHz channel is reserved for satellite-to-satellite (STS) communication. Since the interface to both radios is identical, this channel assignment is strictly a matter of convention and may be reconfigured if necessary (e.g. in the event of a hardware failure). Channel bandwidth is divided between transmitting stations (e.g. satellite and laptop) using a time division multiple access (TDMA) protocol. Relative bandwidth assignments are user definable. Data packets are fixed-length and consist of a structured header and a user-defined payload.

This section first examines the packet format and then examines how this format relates to the TDMA scheme.

H.2.1 SPHERES Packet Structure

The SPHERES communication system is based around a fixed-size packet scheme. The packet structure consists of a 5-byte header and a 32-byte data section. Recent revisions to the DR2000 firmware allow variable-length packets, but most 'standard' packets to date use the common packet length. The packet structure is depicted in Figure H.2. The DR2000 adds its own header (5-byte) and CRC (2-byte) to each packet transmitted by the radio. These components are stripped off of received packets before RS-232 transmission.

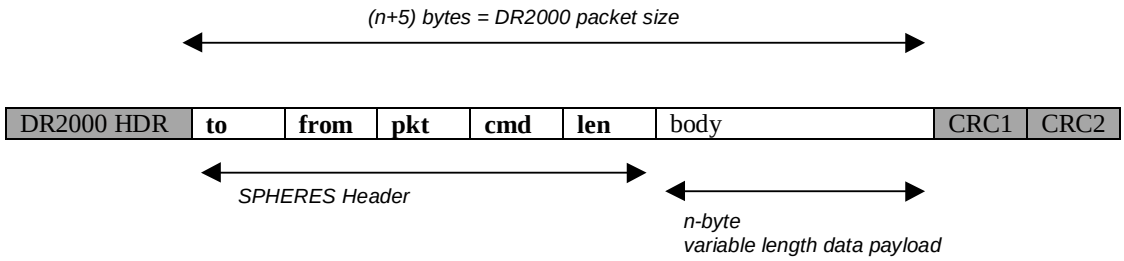


Figure H.2 SPHERES packet structure (n=32)

Important sizes are depicted by arrows in the picture. The base packet length is defined by the body of the packet. We have selected a packet-body size of 32bytes. This gives a reasonable compromise between unused capacity and header overhead. The hardware packet size, set on the DR2000, is five bytes longer than the body to account for the SPHERES packet. header. The header carries information about the packet contents and routing. The header is described in Table H.4.

TABLE H.4 SPHERES header structure

Byte	Length	Field	Description
0	1	to	This contains the receiver hardware ID as seen by each individual SPHERE satellite. Each physical satellite is assigned a unique hardware identifier in the range `0x31-0x35` (hex). Acceptable values are listed in Table 3-3. These addresses must be associated with a 'logical' identity (e.g. SPHERE1, SPHERE2, SPHERE3) in the users program. This allows us to map between logical and hardware addresses, so that any satellite can take on any task.
1	1	from	This bit contains the station ID of the transmitting satellite. The number must meet the same characteristics as described in the to field.

2	1	chk	The checksum field is an 8-bit checksum of the body of the packet, used by the SPHERES system for error detection (but not error correction). The checksum is a simple unsigned sum of the unsigned bytes of the body, truncated to 8 bits. It is calculated as follows:
			`chk = 0;`
			`for (i=0; i<len; i++)`
			`  chk += body[i];`
			`chk = chk & 0xFF;`

TABLE H.4 SPHERES header structure

Byte	Length	Field	Description

3	1	cmd	This is the command field that describes the type of packet. The command field also indicates which channel is being used (868 or 916). Each command byte is structured as follows:
			Bit Description
			0-5 command (range 0-63 decimal, 0x0-0x3F)
			6 indicates whether the packet must be acknowledged. This bit is only used for communications between the Ground station laptop and the SPHERES satellites; no other acknowledgement structure has been implemented.
			7 channel: 0 = 868, 1 = 916
			To build a command byte, one must use the following formula:
			`cmd = CHANNEL + ACK + COMMAND`
			Where:
			`CHANNEL = 0x00 for 868`
			`CHANNEL = 0x80 for 916`
			`ACK = 0x40 if ACK is required`
			`ACK = 0x00 for no ACK`
			`COMMAND = 0x00 – 0x3F`
			Currently defined command assignments are detailed in Section 6 and in the source file `commands.h`.

4	1	len	The length is checked by the receiving satellite to ensure a full packet is processed, and that a new packet is not overwritten if a short packet is transmitted. The length must be set to a value between 0-32, as a 32 byte data size is the longest the DR2000 will allow with the default settings. The default value for len in SPHERES is 32 (0x20).

To send a packet, one first generates a header according to the above structure. This is followed by the n-byte body of the packet. At the link-layer, the format of this section is arbitrary. The header and body must be sent through the serial port to the DR2000. For any standard packet type, unused bytes should still be sent using a filler character. Sending 0x0 or perhaps 0xAA for better bit balancing are good choices.

Inter-packet time must be carefully controlled since the DR2000 does not provide any flow-control information. If packets are sent to the DR2000 too rapidly, data loss will result. There are four stages to each packet transmission (Figure H.3) First, the packet is sent serially from the TX-computer (DSP or CPU) to the TX-DR2000. The packet is then copied to an internal transmit buffer, where the DR2000 header and CRC are added. Next

the packet is transmitted, via wireless, from the TX-DR2000 to the RX-DR2000. The last step involves the serial transfer from the RX-DR2000 to the RX-Computer.

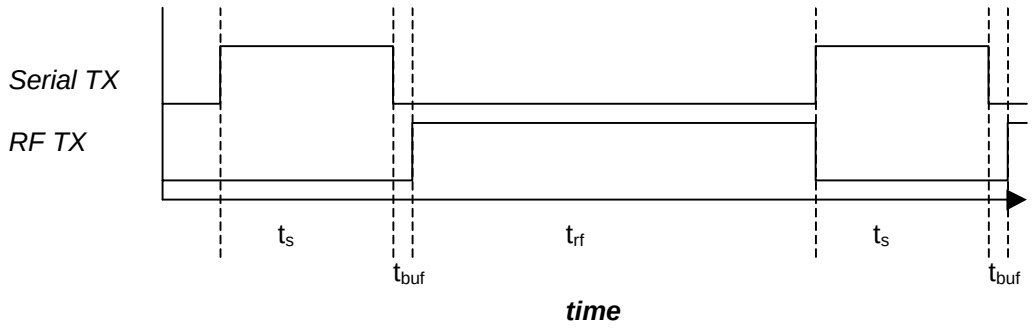


Figure H.3 Packet transmission sequence

To ensure that transmissions do not overlap there must be a minimum time-separation between packets, to ensure the DR2000 buffers are not overwritten. The minimum separation between the start of two packets is:

$$t_{min} = t_s + t_{buf} + t_{rf} \tag{H.1}$$

where t_s is the transmission time between the SPHERES avionics and the DR2000 over the standard UART line, t_{buf} is the buffering time of the DR2000 and t_{rf} is the RF transmit time.

The UART transmission time of an n byte packet is:

$$t_s = \frac{10(n+5)}{115200} \tag{H.2}$$

Each byte sent over a standard UART line is added one stop bit and one start bit, for a total of 10-bits per transmitted byte. The SPHERES header is five bytes long, which must be transmitted over the RS232 line to send the packet. The RS232 line is operated at 115.2kbps. For a packet where n=32 t_s=3.21ms.

The measured buffering time is t_{buf} = 600μs.

For a packet with n bytes in the body, the RF transmission time can be calculated as follows:

$$t_{rf} = \frac{14(n + 15)}{57600} \quad \text{(H.3)}$$

There are a total of 15 header bytes (5 SPHERES header bytes, 3 pre-amble bytes of the DR2000, 5 DR2000 header bytes, and the 2-byte CRC), therefore the (n+15). Further, each byte is converted to a 12-bit word for bit balancing purposes, and added a stop and a start bit, making each word 14-bits long. The RF transmission frequency is 57.6kbps = 57600 bits per second. For the standard SPHERE package where n=32 t_{rf} = 11.42ms.

Therefore, for the SPHERES standard packet, to total time between packet starts must be at least:

$$t_{min} = 3.2ms + 0.6ms + 11.42ms = 15.22ms \quad \text{(H.4)}$$

H.2.2 Time-Division Protocol

Access to the two RF channels must be managed to ensure that transmitting stations do not inadvertently transmit at the same time. Simultaneous transmissions would cause loss of data and degrade the overall productivity of the SPHERES experiment. We have chosen a time-division multiple access scheme for channel management. In such a scheme each station has an assigned time-interval (or window) in which it is permit-ted to transmit. At other times, the station remains in receive mode and must store outgoing packets until its next transmit window. Since most of the communications traffic in our experiments are expected to be predictable and periodic, TDMA is a good access scheme that ensures fair and efficient access to the radio channels.

The SPHERES TDMA scheme is depicted graphically in Figure H.4. The basic unit of time is called a frame. During each frame, all stations are given a chance to transmit. A frame begins with the receipt of a synchronization packet from the laptop. Within a frame,

stations are assigned a transmit window by specifying a start time and a stop time. The frame timers on each satellite will not reset until the next synchronization packet is received.

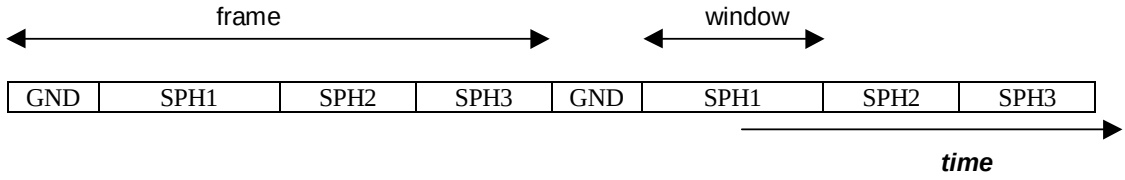


Figure H.4 Time Division Multiple Access scheme

We adopt this synchronization scheme because the we have limited control over the timed behavior of the laptop. Moreover, in tests of the standard windows timer routines, we have observed temporal jitter of up to about 20ms. Making the laptop the master time reference rather than a slave, eliminates the need to worry about synchronizing the laptop to the internal satellites time. Because the laptop dictates the frame size by sending the synchronization packets, and because the laptop software cannot change due to NASA regulations, the frame size has been fixed to 200 ms. Table H.5 lists the standard 200ms frame specification for the STL channel; Table H.6 lists the STS frame.

TABLE H.5 STL standard frame specification (200ms frame)

Number of Stations	Station	Length (ms)	Start (ms)	Stop (ms)
4	SPH1	53	0	53
	SPH2	53	53	106
	SPH3	54	106	160
	GND	40	160	200
3	SPH1	80	0	80
	SPH2	80	80	160
	GND	40	160	200
2	SPH1	160	0	160
	GND	40	160	200

TABLE H.6 STS standard frame specification (200ms frame)

Number of Stations	Station	Length (ms)	Start (ms)	Stop (ms)
3	SPH1	66	0	66
	SPH2	66	66	132
	SPH3	68	132	200
2	SPH1	100	0	100
	SPH2	100	100	200
1[a]	SPH1	200	0	200

a. STS use during single satellite operation is useful if there is a DR2000 that is listening on that channel in order to download telemetry data.

We make the following notes about the frame specification process:

- The satellite ID's are the logical identifications of the SPHERES satellites, and not the hardware ID (i.e., SPHERE1, SPHERE2, etc., rather than hardware ID 0x31, 0x32, etc.).
- To completely describe the frame structure we need to specify the frame length, and start and stop times for each station. These times are offsets from the start of a frame and are measured in units of ms.
- After a frame expires, the next frame will not begin until the next synchronizing packet is received. This satisfies the safety related requirements that will disable the satellites if it loses communication with the laptop.
- Users and guest scientists can specify the frame structure for their experiments. This can change from test to test and is specified in the GUI-ini file provided by the user.
- Stations need not receive equal windows.
- The ground (laptop) station must be the last window on the STL channel. This is to allow for frame synchronization (see below). This window must be at least 40 ms long. If the laptop has additional data to transmit, it should first transmit the extra packets, and then send the synchronization.
- Users are solely responsible for ensuring that their window divisions are consistent. As a matter of convention, users may freely configure the STS channel but should employ the standard STL assignments in most circumstances.

H.2.3 Packet Acknowledgement

The SPHERES core operating system implement a simple packet acknowledgement scheme dedicated to ensure commands from the laptop are received and executed syn-

chronously between multiple satellites. This packet acknowledgement is implemented by using the `cmd` field of the SPHERES header and a byte in the state-of-health (SOH) packet (described below), which is sent to the laptop. While other satellites also see the SOH, and could conceivably use the same system to require inter-satellite acknowledgements this is not recommended. First, the state-of-health packet is only transmitted via STL, therefore only STL acknowledgements are possible. Second, the SPHERES core software does not contain functions that process an acknowledgement, and therefore the user would still need to implement special procedures. Therefore, any other type of packet acknowledgement is the responsibility of the guest scientist.

Figure H.5 illustrates the packet acknowledgement sequence. In order to enable packet acknowledgements bit 7 of the `from` field in the header must be set. When a SPHERE satellite receives a packet that requires an acknowledgement, it will set its bit field (given its logical satellite ID) to 1 in the SOH, and request that the SOH be sent immediately. The SOH will be sent to the laptop in the satellite's next TDMA window. Because the laptop is the only unit expected to send packets that require acknowledgement and because that packet causes the TDMA synchronization, the implemented scheme requires that a satellite reply with the acknowledgement in the same frame. If the laptop does not receive an acknowledgement in the same frame, it will retransmit the command. The laptop tracks the number of acknowledgements expected, and will retransmit the command to all satellites until it receives the required number of replies - all satellites always retransmit the acknowledgements. Note that in this process, there is no memory of the original packet, but rather a new one is created until the satellites respond correctly.

Because a packet acknowledgement is intended to synchronize the SPHERES test times, every time that a SPHERE gets a command to start a test it should request for an acknowledgement. The satellite, in turn, will always reset the test time when a command to start a test is received, even if a test is already in progress. In the case where the satellite is already running a test, the satellite will terminate that test immediately, and restart a test with the new test command received. This behavior ensures that all satellites will be run-

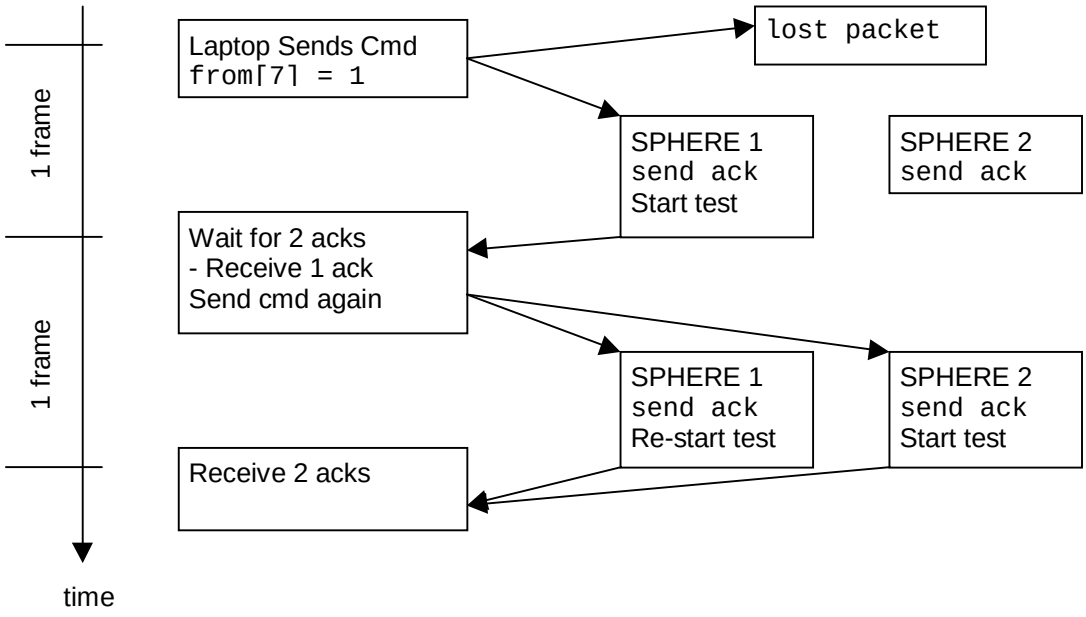


Figure H.5 Packet acknowledgement sequence example (1 lost packet)

ning the same test with the same time (within 1ms of each other) once the acknowledgement process is complete.

H.3 Application Layer Interface

This section describes the different messages that are exchanged between the SPHERES satellites and the laptop base station. All other packages are not interpreted by either the satellites or the laptop in a standard configuration (the user may create message receiver procedures for the satellites, but that is at a higher level interface and is part of the Guest Scientist Program interface, not of the basic communications interfaces).

H.3.1 Transmitted Messages

The laptop must generate two types of packets: General-Purpose Commands (GPC), and Initialization Packets.

H.3.1.1 General Purpose Commands

Figure H.6 presents the contents of a general purpose command packet. These packet act as frame synchronization for the satellites, control test execution, and resets. The laptop broadcasts this packet every 200ms, creating the 200ms TDMA frame size specified above. Because the laptop transmission frequency is fixed, the frame size is restricted to 200ms.

Header:

```
to   = 0x00
from = 0x30
chk  = <checksum>
cmd  = COMM_CMD_GENERALCMD (base + 0x21 = 0x41)
len  = 0x20
```

0	1	2	3	4	5	6	7
Run Time cmd	Run Time unit	unused		Test Number SPHERE 1		Test Control SPHERE 1	

8	9	10	11	12	13	14	15
Test Number SPHERE 2		Test Control SPHERE 2		Test Number SPHERE 3		Test Control SPHERE 3	

16	17	18	19	20	21	22	23
Test Number SPHERE 4		Test Control SPHERE 4		Test Number SPHERE 5		Test Control SPHERE 5	

24	25	26	27	28	29	30	31
Reset SPH s/n 1	Reset SPH s/n 2	Reset SPH s/n 3	Reset SPH s/n 4	Reset SPH s/n 5	STL Sync	STS Sync	Enter Boot Load

Figure H.6 General purpose command structure

The bit fields for the different command types are described in Table H.7.

Starting & Stopping a Test

Sending a start or stop test are mutually exclusive; the SPHERES will check that if a test start command is sent the stop is not present, or vice versa; if both commands are present the packet will be ignored.

TABLE H.7 General purpose command structure details

Byte	Size (bits)	Field	Type	ID[a]	Description
0	1 (8)	run time cmd	unsigned char	LOG	This field is used to send the satellites commands during tests for manual operations of the satellites. This field includes a one-byte `unsigned char` command that will be passed on to the SPHERE controller software even while a test is already in progress.

1	1 (8)	run time satellite	unsigned char	LOG	This field indicates the logical ID of the commanded satellite. Any combination of bits is allowed. The bit fields are:
					`Bit Description`
					`0 SPHERE 1`
					`1 SPHERE 2`
					`2 SPHERE 3`
					`3 SPHERE 4`
					`4 SPHERE 5`
					`5-7 unused`

2-3	2 (16)	unused	-	-	-
4-5	2(16)	test number SPH1	unsigned short	LOG	This two-byte, unsigned integer specifies the test number to start. This field is ignored unless the Start Test bit is set.

6-7	2(16)	test control SPH1	unsigned short	LOG	The test control field is used to start and stop tests as well as to command synchronization of the satellites. All unused bits are reserved for future use.
					`Bit Description`
					`0 reserved`
					`1 Start Test`
					`2 Stop Test`
					`3 Synchronize SPH time`
					`4-15 unused`

8-9	2(16)	test number SPH2	unsigned short	LOG	See above.
10-11	2(16)	test control SPH2	unsigned short	LOG	
12-13	2(16)	test number SPH3	unsigned short	LOG	
14-15	2(16)	test control SPH3	unsigned short	LOG	
16-17	2(16)	test number SPH4	unsigned short	LOG	
18-19	2(16)	test control SPH4	unsigned short	LOG	

TABLE H.7 General purpose command structure details

Byte	Size (bits)	Field	Type	ID[a]	Description
20-21	2(16)	test number SPH5	unsigned short	LOG	
22-23	2(16)	test control SPH5	unsigned short	LOG	

24
 	
25
 	
26
 	
27
 	
28
	1(8)
	
	1(8)
	
	1(8)
	
	1(8)
	
	1(8)
		reset SPH 0x31
		
		reset SPH 0x32
		
		reset SPH 0x33
		
		reset SPH 0x34
		
		reset SPH 0x35
			unsigned char
			
			unsigned char
			
			unsigned char
			
			unsigned char
				HW
				
				HW
				
				HW
				
				HW
					This is a byte field to initiate reset of the satellite. Because resets can cause loss of data or resources, they must be repeated several times in a row before a satellite will process them. The following commands are supported:
					`Bit Description Repeat`
					`0 Vent Tank 2`
					`1 Soft Reset 2`
					`2 Hard Reset 2`
					`3 916 Reset 2`
					`4 868 Reset 2`
					`5 Tank count 2`
					`6-7 unused`

29
 	
30
	1(8)
	
	1(8)
		STL Sync
		
		STS Sync
			unsigned char
			
			unsigned char
				LOG	Synchronize / Enable the corresponding communications channel. This byte must be received to synchronize every frame and in turn enable communications on this channel. The communications channel will not transmit after their initial window until a new Sync bit is received. Each bit is used for a logical SPHERE satellite:
					`Bit Description`
					`0 SPHERE 1`
					`1 SPHERE 2`
					`2 SPHERE 3`
					`3 SPHERE 4`
					`4 SPHERE 5`
					`5-7 unused`

TABLE H.7 General purpose command structure details

Byte	Size (bits)	Field	Type	ID[a]	Description

31	1(8)	Enter Boot Load	unsigned char	HW	This command forces the satellite with corresponding hardware ID to enter boot loader mode. Once the command is received the satellite will not exit boot loader mode until it has been re-programmed with a new program. One bit is assigned to each satellite; any combination of bits may be used, since the boot loader identifies which satellite is being programmed. Note that due to its potential effect to erase the current program, the command must be received 3 times in a row take effect:
					`Bit Description Repeat`
					`0 SPH HW ID 0x31 3`
					`1 SPH HW ID 0x32 3`
					`2 SPH HW ID 0x33 3`
					`3 SPH HW ID 0x34 3`
					`4 SPH HW ID 0x35 3`
					`5-7 unused`

a. The ID field specifies whether the command is destined for the HW - hardware ID or the LOG - logical ID.

All packets that include a test start or test stop must request an acknowledgement as described in Section H.2.3. Therefore, a start/stop test should be re-sent until the satellite acknowledges its receipt. Note that the satellite will acknowledge the packet as soon as its received, regardless of whether the start/stop test is valid at the time the command was received. The satellite sending out the start/stop test is responsible to check the SOH packet to determine if a test actually started or not. For example, if the satellite is currently in "idle" mode and a "test start" command is sent, the satellite will acknowledge the packet as received, but a test will not start because the "enable" button has not been activated.

H.3.1.2 Initialization Packets

Initialization packets are sent to the satellites so that they know the locations of the beacons and the current temperature. Each satellite stores this information in the onboard flash memory. When a satellites boots, it reads the flash and processes the beacon locations saved there. Since the satellites has no way of knowing the age of this data, it sets a

bit in its state-of-health packets to signify that it is using old beacon data and has not received new data since power-on. If the GUI sees that this bit is set, it can generate extra initialization packets for the satellite. Such initialization is optional from the satellite's point of view - it will operate correctly without beacon initialization.

For simplicity, we have adopted the simple but somewhat redundant packet structure presented in Figure H.7 and described in Table H.8.

Header:

```
to   = <any valid address>
from = 0x30
chk  = <checksum>
cmd  = COMM_CMD_BEACON_INFO (base + 0x26 = 0x46)
len  = 0x20
```

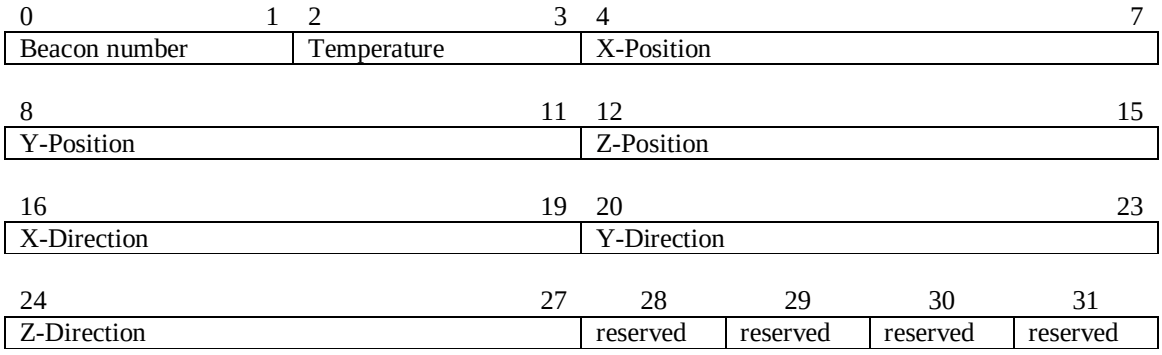


Figure H.7 Initialization packet structure

TABLE H.8 Initialization packet structure details

Byte	Size (bits)	Field	Type	Description
0-1	2(16)	beacon number	unsigned short	This value selects the beacon that the packet applies to. Valid numbers are from 1-6.
2-3	2(16)	temperature	unsigned short	The current temperature specified in tenths of a degree C. (i.e. 22.0°C = 220).

TABLE H.8 Initialization packet structure details

Byte	Size (bits)	Field	Type	Description
4-7	4(32)	X-position	float	4-byte IEEE floating point that specifies the 3D position of the beacon in meters. The value should be cast into an integer in order to transmit it correctly; e.g.: `float xPos = 3.187; int xVal; xVal = *((unsigned int *) &xPos);`
8-11	4(32)	Y-position	float	
12-15	4(32)	Z-position	float	
16-19	4(32)	X-direction	float	4-byte IEEE floating point that specifies the three element unit vector direction of the beacon.
20-23	4(32)	Y-direction	float	
24-27	4(32)	Z-direction	float	
28-31	4(32)	unused	-	-

By convention, unused beacons should specify zeros for the direction vector. The direction vector of active beacons should be normalized (i.e. have unity magnitude).

Determination of Beacon Position and Direction

Definitions

- *D* = distance of beacon to seat track
- *P(i)* = (x_i, y_i, z_i) = position of each used seat track hole location with respect to the ISS
- *B(i)* = (x_i, y_i, z_i) = corrected position of each beacon transmitter element
- *q(i)* = (x_{dir-i}, y_{dir-i}, z_{dir-i}) = unit vector direction of each beacon

The following data must be known in order to solve the equations:

- Number of beacons in use
- *P(i)* for each beacon in use
- Gold and black angles for each beacon, from here on referred to as $gold_i$ and $black_i$

Steps to determine position & direction

1. Location Correction

Correct the location of the beacon transmitter itself by adding the offset of the transmitter from the seat track. The correction depends on the location of the seat tracks as per Table H.9

TABLE H.9 Corrections and seat track locations

Track	Correction
Overhead	$Bx_i = Px_i$ $By_i = Py_i$ $Bz_i = Pz_i + D$
Starboard	$Bx_i = Px_i$ $By_i = Py_i + D$ $Bz_i = Pz_i$
Deck	$Bx_i = Px_i$ $By_i = Py_i$ $Bz_i = Pz_i - D$
Port	$Bx_i = Px_i$ $By_i = Py_i - D$ $Bz_i = Pz_i$

This calculates all the *B*(i)'s.

2. Move the frame origin to center of beacon area

   First determine the total expansion of the beacon area; for all beacons find:

$$x_{max} = \max(Bx_i) \tag{H.5}$$

$$x_{min} = \min(Bx_i) \tag{H.6}$$

$$y_{max} = \max(By_i) \tag{H.7}$$

$$y_{min} = \min(By_i) \tag{H.8}$$

$$z_{max} = \max(Bz_i) \tag{H.9}$$

$$z_{min} = \min(Bz_i) \tag{H.10}$$

   Now determine the 'center' of the beacon area:

$$x_{gc} = (x_{max} + x_{min}) / 2 \quad \text{(H.11)}$$

$$y_{gc} = (y_{max} + y_{min}) / 2 \quad \text{(H.12)}$$

$$z_{gc} = (z_{max} + z_{min}) / 2 \quad \text{(H.13)}$$

Now 'center' the beacon locations using the general center locations; for all beacons:

$$B'(i) = (x'_i, y'_i, z'_i) = \quad \text{(H.14)}$$

$$x'_i = x_i - x_{gc} \quad \text{(H.15)}$$

$$y'_i = y_i - y_{gc} \quad \text{(H.16)}$$

$$z'_i = z_i - z_{gc} \quad \text{(H.17)}$$

3. Determine direction with unit vectors

   The vector overhead depends on the beacon location as specified in Table H.10.

TABLE H.10 Unit vector determination

Track	Correction
Overhead Deck	x_{dir-i} = -sin($gold_i$) * sign(x'_i) y_{dir-i} = -cos($gold_i$) * sin($black_i$) * sign(y'_i) z_{dir-i} = -cos($gold_i$) * cos($black_i$) * sign(z'_i)
Starboard Port	x_{dir-i} = -sin($gold_i$) * sign(x'_i) y_{dir-i} = -cos($gold_i$) * cos($black_i$) * sign(y'_i) z_{dir-i} = -cos($gold_i$) * sin($black_i$) * sign(z'_i)

where sign(#) is defined as:

$$\text{if } (\# < 0) = (-1) \quad \text{(H.18)}$$

$$\text{if } (\# >= 0) = (1) \quad \text{(H.19)}$$

4. Transmit Packets

   A total of "i" packets will be transmitted, one for each beacon, the packet should be composed of the bytes described in Table H.11.

TABLE H.11 Beacon location packet structure

Byte	Size (bits)	Field	Type	Value
0-1	2(16)	beacon number	unsigned short	i
2-3	2(16)	temperature	unsigned short	(temp * 10)
4-7	4(32)	X-position	float	B'(i) = x'_i
8-11	4(32)	Y-position	float	y'_i
12-15	4(32)	Z-position	float	z'_i
16-19	4(32)	X-direction	float	$\theta(i) = x_{dir-i}$
20-23	4(32)	Y-direction	float	y_{dir-i}
24-27	4(32)	Z-direction	float	z_{dir-i}

To store the `float` variables into a binary `char` packet the following cast can be used:

```
float x[5], y[5], z[5];
int x_int[5];
unsigned char packet[32];

x[i] = <x final pos of beacon i>;          // float
x_int[i] = *((unsigned int *) &x[i]);      // int
memcopy (&packet[i*4], y[i], sizeof(int)); // unsigned char
```

H.3.2 Received Messages

The laptop must archive all incoming data from the radio to a binary file. Post processing will be used to extract telemetry and experiment results. Although the laptop is free to ignore the majority of this traffic, it must perform some preliminary packet parsing to decode the state-of-health of each active satellite.

H.3.2.1 Packet Parsing

Packets are nominally 37 bytes long (including header), but loss of individual bytes has been observed. Therefore, the laptop should use simple pattern matching on the incoming

data to look for the header, which presents the best pattern to match. Specifically, look for the following byte sequence presented in Table H.12.

TABLE H.12 Packet parsing matching sequence

Header	Value Match
TO	0 0x30-0x39
FROM	0x30-0x39 0xB0-0xB9
PKT	<any>
CMD	<specific cmd>
LEN	0x25

Using only the `TO`, `FROM`, and `LEN`, fields the probability of a false positive match in a random data stream of data is about $2x10^{-6}$. If we consider that we only need to parse the state-of-health packets, we can also match by `CMD` field, and the `TO` field will always be broadcast. This will drop the probability of false matching to about $4x10^{-9}$. This low figure indicates that direct, localized pattern matching (i.e. without memory of past PKT fields, or when the last header was found), should provide sufficient performance.

H.3.2.2 State-of-Health Packets

State of health (SOH) packets are generated by each active satellite at 1Hz. These are added to the onboard message transmit queues, so actual reception time of these messages may vary depending on the current traffic levels. The packets are formatted as presented in Table H.13.

TABLE H.13 State of Health packet structure details

Byte	Size (bits)	Field	Type	Description
0-3	4(32)	Time Stamp	unsigned int	The time is measured in ms. The time is referenced to either the power-on time, or the last time a time-synchronization command was received.
4-7	4(32)	Program ID	unsigned int	This is a unique identifier that is associated with each program file.

TABLE H.13 State of Health packet structure details

Byte	Size (bits)	Field	Type	Description
8-11	4(32)	Tank Usage	unsigned int	This is the aggregate firing-time of the thrusters on the satellite (thruster-milliseconds). This value is divided by the expected total gas availability to provide a percentage estimate of the amount of remaining propellant.
12-15	4(32)	Test Time	unsigned int	Time in ms since the start of the current test.
16-19	4(32)	Maneuver Time	unsigned int	Time in ms since the start of the current maneuver.
20	1(8)	Last Test Result	unsigned char	Indicates the manner in which the last test completed. The following codes are defined: Bit Description 0 No data / test in progress 1 Normal / OK 2 Stop via hardware Enable button 3 Stop via communications Stop Test 4 Stopped due to communications failure 5 Unknown Test 6 Test Timeout 9 Satellite Not Enabled 10-255 User defined
21	1 (8)	Temperature	unsigned char	The temperature set in the SPHERE metrology section. The temperature is specified in tenths of a degree C. (i.e. 22.0°C = 220).
22-23	2(16)	IR Rcv Counter	unsigned short	The number of IR pulses observed since the last reset.
24-25	2(16)	Test Number	unsigned short	Currently running test number. A value of 0 indicates that no test is running
26-27	2(16)	Maneuver Number	unsigned short	Current maneuver number. This value is zero if no test is running.
28	1(8)	Status	unsigned char	This is a bit-field description of the SPHERES status: Bit Description 0 Battery status (1 = OK, 0 = Low) 1 STS Enabled 2 STL Enabled 3 Using old beacon data 4-7 Unused

TABLE H.13 State of Health packet structure details

Byte	Size (bits)	Field	Type	Description

29	1(8)	Operating Mode	unsigned char	This describes the current operating mode of the satellite. The whole byte is used to identify the mode:
				`Value Description`
				`0 Idle`
				`1 Transition (enable button pressed)`
				`2 Position Hold`
				`3 Running a test`
				`4 Suspend (due to lack of RF Sync)`
				`5-255 Unused`

30	1(8)	Satellite Role	unsigned char	Indicates the current logical role performed by the satellite identified in the header by its hardware ID. The SPHERES interpret this value to establish the connection between logical address or role (i.e. SPHEREn) and the hardware address (i.e. the FROM field in the header).
				Must be one of the following values:
				`Value Description`
				`1 SPHERE 1`
				`2 SPHERE 2`
				`3 SPHERE 3`

31	1(8)	Acknowledge-ment	unsigned char	This byte is set to 1 if the satellite is acknowledging a command received in the previous General Command Packet received. It is zero if this is not an acknowledgement.

H.3.2.3 Background Telemetry

Active satellites automatically and periodically broadcast their onboard state estimates. It is unlikely that the GUI will have to parse these packets, but the packet definition is included in Figure H.8 and Table H.14 for completeness.

The telemetry packet use 16-bit short integers to transmit all the data, therefore these numbers are scaled using the factors presented in Table H.15. Multiply the received short integer by its factor to obtain the original values.

Header:

```
to   = 0 (Broadcast)
from = <unit hardware ID>
chk  = <checksum>
cmd  = COMM_CMD_TELEMETRY (base + 0x1B = 0x3B)
len  = 0x20
```

0 – 2	3	4 – 5	6 – 7
Time Stamp	Unit role	X-Position	Y-Position
8 – 9	**10 – 11**	**12 – 13**	**14 – 15**
Z-Position	X-Velocity	Y-Velocity	Z-Velocity
16 – 17	**18 – 19**	**20 – 21**	**22 – 23**
ε_1 quaternion	ε_2 quaternion	ε_3 quaternion	η quaternion
24 – 25	**26 – 27**	**28 – 29**	**30 – 31**
ω_x angular velocity	ω_y angular velocity	ω_z angular velocity	unused

Figure H.8 Telemetry packet structure

TABLE H.14 Telemetry packet structure details

Byte	Size (bits)	Field	Type	Description
0-2	3(24)	time stamp	unsigned int	Current satellite time in ms. Note the most significant byte of the unsigned integer must be masked off to allow the role to be stored there.
3	1(8)	satellite role	unsigned char	The current role of the satellite as described in Table 6-4.
4-5	2(16)	X-position	short	The position of the satellite with respect to the global frame. A 16bit signed short indicating the value. The signed short is created by dividing the original float using the factors presented in Table 6-5.
6-7	2(16)	Y-position	short	
8-9	2(16)	Z-position	short	
10-11	2(16)	X-velocity	short	The velocity of the satellite with respect to the global frame, scaled as above.
12-13	2(16)	Y-velocity	short	
14-15	2(16)	Z-velocity	short	

TABLE H.14 Telemetry packet structure details

Byte	Size (bits)	Field	Type	Description
16-17	2(16)	ε_1 quaternion	short	The quaternion components that describe the attitude of the satellite with respect to the global frame, scaled as above.
18-19	2(16)	ε_1 quaternion	short	
20-21	2(16)	ε_1 quaternion	short	
22-23	2(16)	η quaternion	short	
24-25	2(16)	ω_x rate	short	The angular velocities of the satellite with respect to the global frame of reference, scaled as above.
26-27	2(16)	ω_x rate	short	
28-29	2(16)	ω_x rate	short	
30-31	2(16)	unused	-	-

TABLE H.15 Telemetry data conversion factors

Element	Units	Maximum (`0x7FFF`)	Function	Factor
Position	m	3.5m	3.5[m] / 32767[b] =	.00010681 [m]/bit
Velocity	m/s	1.0m/s	1.0[m] / 32767[b] =	.000030519 [m/s]/bit
Quaternion	-	1.0	1.0 / 32767[b] =	.000030519 [ ]/bit
Angular Velocity	rad/s	1.5rad/s	1.5[rad/s] / 32767[b] =	.000045778 [rad/s]/bit

Appendix I

SPHERES EXPERIMENTAL RESULTS

I.1 Results at the MIT SSL

The SPHERES facilities have operated at the MIT SSL since the Spring of 2000 utilizing the original prototypes. During this time research was done on the effects of communications on formation flight systems (2000), formation flight algorithms (2000+), docking algorithms (2002+), system identification and fault detection (2003+), tether prototypes (2003+), and began research to support the Mars Orbiting Sample Return mission (2004+). Table I.1 summarizes the research conducted at the MIT SSL facilities in Cambridge, MA since 2000.

TABLE I.1 Research conducted at the MIT SSL

Research	Year	Application	Guest Scientist
F.F. Communications	2000	DSS	
F.F. Control	2000	TPF	JPL
Docking Control	2002 +	Orbital Express (DARPA)	
Mass ID / FDIR	2003 +	Modeling	Ames
Tethers	2003 +	SPECS	Goddard
MOSR	2004 +	Mars Sample Return	

The tests on communications during 2000 provided an initial understanding of simple communications issues regarding formation flight. The goal of the tests was to determine the effects between high frequency transfer or state information and the size of the state being transferred. To this purpose five experiments were setup; a number of tests for each setup was performed. The final results evaluated the difference in performance between the different experimental setups. [Saenz-Otero, 2000]

Development of formation flight algorithms began in the Spring of 2000 in preparation for tests aboard the KC-135 reduced gravity airplane (RGA) and later on for tests at the MSFC Flat Floor. These tests included control of one, two, or three degrees of freedom between two spacecraft (2000+, for the RGA) and formation flight of three and five spacecraft (2003+, for the MSFC Flat Floor). [Hilstad, 2003], [Kong, 2004a]

The formal development of docking algorithms began during 2002 [Nolet, 2004]. While previous tests conducted sets of maneuvers that resulting in docking, these did not follow the full scientific process, but were rather a demonstration of the facility's capabilities.

Work with NASA Ames Research Center provided the first complete cycle of remote investigation using SPHERES. Through 2003 and 2004 researchers at NASA Ames paired with a member of the SPHERES team to conduct experiments at the MIT SSL while they worked from their home facility. The Ames investigators developed their own algorithms in house, and incorporated them into the SPHERES Core using the provided API. For several months the algorithms were sent electronically to MIT, debugged over a few days, and then experiments conducted. Once the algorithms were close to mature, the Ames investigators visited the MIT SSL facilities to finalize their design. After the final design was complete, the researchers returned to NASA Ames, and the SPHERES team conducted further tests to obtain more data. The Ames algorithms were also tested aboard the RGA. [Berkovitz, 2003]

Development of tether mechanism that utilize the SPHERES satellites' expansion port began at the MIT SSL during 2003. Prototype designs were developed and used both at

the MIT SSL facilities and at the MSFC Flat Floor. These prototypes utilize power and signals from the SPHERES expansion port to incorporate the tether mechanism as an integral part of the satellite. Development at the MIT SSL has provided the knowledge necessary to create medium fidelity models of tethered spacecraft systems; these models were utilized for tests in the MSFC Flat Floor.

Research to support the Mars Orbiting Sample Return mission (MOSR) began in 2004 by the creation of an initial design of a capture mechanism and the development of algorithms to simulate impact forces. Tests at the MIT SSL include the development of control algorithms that ensure a SPHERE satellite will impact a target in a fixed location at a prespecified velocity and angular rate, representative of what is expected in the actual space capture. At this point one substantial thread of iterations has occurred to present the MOSR team with a proof-of-concept on the use SPHERES to support their research.

I.2 Results aboard the KC-135

Both the prototype and the flight units have been used for experiments aboard the KC-135 reduced gravity airplane over the course of six weeks. Table I.2 presents the month and year of each flight week, as well as the type of satellites (prototype or flight equipment) used in each flight. A short description of each week's research is presented next.

TABLE I.2 KC-135 flight weeks and satellites operated

Dates	Prototype	Flight
Feb. 2000	2	-
Mar. 2000	2	-
Oct. 2001	2	-
July 2002	2	1
Feb. 2003	-	3
Nov. 2003	-	2

February 2000

The February 2000 flights were performed as a proof-of-concept flight for the SPHERES satellites. This week of flights did not concentrate on the development of control algorithms, but rather on the thorough testing of all the SPHERES systems. These tests were the first time that the SPHERES hardware was exposed to a 6DOF environment, and therefore the first time that a controller had to operate more than two thrusters at a time. It was also the first time that the operators were required to use the interfaces and control software in a stressful environment. Table I.3 summarizes the tests conducted with the prototype units during the February 2000 flights.

TABLE I.3 February 2000 flight results

Feb. 2000	Test Topics	Experiment Repetitions
Flight 1	1 DOF Open Loop Rotations	20
	1 DOF Open Loop Translations	20
Flight 2	1 DOF Open Loop Rotations	40
Flight 3	1 DOF Closed Loop Rotations	20
	3 DOF Rotation Damp	15
	2 SPHERES Rotation Comparison	5
Flight 4	3 DOF Closed Loop Rotations	20
	2 SPHERES Rotation Comparison	20

The resulting information led to the first major iteration in the design of the SPHERES facilities. The satellites were upgraded prior to the next flight to increase the reliability of the propulsion, communications, and power systems, which presented failures during the February 2000 flights. The graphical user interface went through its first iteration: the type of data presented changed (presented more processed data, instead of raw information), although the type of interface remained basically unchanged.

March 2000

The March 2000 tests were the first tests where science was conducted aboard the KC-135. These flights concentrated on two main topics: to test the global frame metrology system in a 3D environment and to perform angular rotation formation flight maneuvers. The tests of the global metrology system included tests that consisted solely of data collection as well as tests that attempted to control the satellites with respect to the global frame. The formation flight maneuvers began with initial development of 3DOF angular rotation controllers and the collection of data from the KC-135 frame of reference. The formation flight tests used the knowledge obtained from the angular rotation and frame measurements tests to perform relative maneuvers between two satellites. Table I.4 summarizes the experiments conducted during March 2000.

TABLE I.4 March 2000 flight results

Mar 2000	Test Topics	Experiment Repetitions
Flight 1	Global System ID	4
	Global Frame Control	5
	Angular regulation (Euler)	15
	Angular regulation (Quadrenniums)	10
	KC Frame ID	20
Flight 2	Global System ID	5
	Global Frame Control	25
	KC Frame ID	30
Flight 3	Master/Slave FF Tests	20
	Global Frame Control	20
Flight 4	Angular regulation	10
	Global Frame Control	20
	Minimum Gas Turn	10

October 2001

The February and March 2000 flights both demonstrated that one of the most challenging sub-systems in the SPHERES testbed was global metrology. While several tests during 2000 demonstrated the ability for inertial control using the gyroscopes, SPHERES had not been able to control consistently with respect to the global metrology system. Therefore, in what can be considered a long-term iteration, the SPHERES team updated the global metrology system to experiment once more in the KC-135.

Table I.5 summarizes the experiments conducted on October 2001. The majority of the October 2001 tests concentrated on testing the global metrology system. Having learned that the SPHERES satellites lacked enough authority to perform complex maneuvers in the KC-135 frame (mostly due to airplane turbulence), the new tests concentrated on 1DOF maneuvers and station keeping. Throughout the week tests were run and data analyzed. The data included both free-float data to determine if the global metrology system had enough bandwidth to capture the motion of the satellites during turbulence, as well as data when control was attempted. Data analysis indicated further need to refine the metrology system before being able to determine if the units had enough authority. During the last day of flights the global frame control was mostly limited to attitude control (with respect to the global frame), which was deemed possible through analysis of data earlier in the week.

The second set of tests was to calculate the inertia of the satellites. This information was obtained over two days, with data analysis in between to ensure the right data was obtained. This analysis time was essential, since it led to new tests that collected data not considered necessary on the first day. The data helped develop the attitude control tests of the last day of experiments.

July/August 2002

These flights carried out the first 6DOF tests of the flight units. The first flight unit was tested in this environment for full operations, especially for repeatability and reliability of

TABLE I.5 October 2001 flight results

Oct. 2001	Test Topics	Experiment Repetitions
Flight 1	Inertia Measurement	6
	Closed Loop Inertial Control	12
	Global Frame Control	20
Flight 2	Hardware Tests	10
	Global Frame Control	30
Flight 3	Inertia Measurements	10
	Global Frame Control	30
Flight 4	Global Frame Control	40

its sub-systems. While not a direct part of the science, it demonstrated the success of a major design iteration from the prototype units to the flight units. The flight unit was used in the first flight training videos for astronauts. These videos were improved over future flights.

The science conducted in these flights, using the prototype units, once again concentrated on the global metrology system, but also included a substantial amount of docking control. Lessons were learned from past experiences with the global metrology system, and the experiments were modified to better fit the environment of the KC-135. The global metrology tests during these flights were conducted solely as relative flight between the units, attempting to maintain formation as a system. The total system was not controlled to maintain a certain position or attitude with respect to the global frame; rather, the units were required to maintain formation. The tests were more successful than in the past, but the limited test time of 20 second prevented substantial data from being collected. A summary of these tests is presented in Table I.6.

The success of the relative motions in the initial days led to the development of docking controllers which, as with the past global system controllers, performed docking maneuvers controlling only the distance between the units, and not the absolute position of the system. These tests completed successfully over a few parabolas. The relevant part of

TABLE I.6 July/August 2002 flight results

Jul./Aug. 2002	Test Topics	Experiment Repetitions
Flight 1	Global Frame Control	40
Flight 2	Global Frame Control	40
Flight 3	Global Frame Control	20
	Docking	20
Flight 4	Global Frame Control	20
	Docking	20

these tests is that they utilized the iterative nature of SPHERES over a wide range of time frames: the success of the algorithms was due to the better understanding of the KC-135 environment over multiple flights, to the development of the relative motion controllers, and to the ability to iterate over one day on the docking algorithms.

February 2003

The February 2003 experiments were conducted solely with the flight units, and the tests were geared mostly to conducting small amounts of science each day. There were three main areas of study during these flights: high level system identification via one-degree-of-freedom maneuvers, to later be used in FDIR algorithms; high-speed collection of inertial data to fully identify the thrusters with the goal to create a B matrix in the standard state-space equations, while also helping fully identify the dynamics of each satellite and the behavior of the inertial sensors; and control within the global frame was once again attempted in two manners, one setup using the KC-135 frame, and another setup using the ultrasound system only between units, without referencing the KC-135 frame at all. Table I.7 summarizes the tests conducted during February 2003.

The tests suffered a setback during the second flight, when the communications system failed for the majority of the flight and no data collection was possible. The SPHERES designed hardware did not fail; it was the COTS laptop computer that served as a control station which failed. While this was a setback of the individual flight, it provided direct

TABLE I.7 February 2003 flight results

Feb. 2003	Test Topics	Experiment Repetitions
Flight 1	1DOF System ID	20
	Thruster ID	10
	Global Frame Control	10
Flight 2	1DOF System ID	3
	3 SPHERE Handling Ops.	33
Flight 3	1DOF System ID	16
	Thruster ID	15
	Relative Frame Control	6
Flight 4	1DOF System ID	7
	Thruster ID	14
	Global & Relative Frame Control	15

feedback of the one-fault tolerant parts of the SPHERES facilities and resulted in a change of the manifest of flight equipment that will be sent to the ISS.

The system identification tests were ran multiple times the first day of tests. Analysis of this data resulted in changes of the type of data captured during the last day of tests. The thruster identification tests were conducted all week to obtain as much data as possible and create a good model of the SPHERES actuators/sensors transfer functions. The collection of this data was not necessarily iterative, since the type of data collected and the methods to collect them did not change throughout the week; rather, these sets of tests ran into the time limitations of the KC-135 to the point where tests had to be repeated multiple days. Lastly, the global frame control did go through real iterations, leading to the execution of several formation flight algorithms that used the on-board beacons of the satellites to maintain both pointing and separation relative to each other.

November 2003

The November 2003 flights were dedicated to conducting formation flight maneuvers and FDIR data collection. Tests on the best methods to send critical data for a distributed sys-

tem were also conducted. Table I.8 summarizes the tests conducted during this week of experiments. Since the formation flight maneuvers were only relative, the global metrology system was not used during this week of flights.

TABLE I.8 November 2003 flight results

Nov. 2003	Test Topics	Experiment Repetitions
Flight 1	Beacon Track	9
	Docking	6
	Lost in Space	5
	Inertia ID	7
	Distributed Control Architecture	10
Flight 2	Beacon Track	20
	Distributed Control Architecture	10
	Inertia ID	10
Flight 3	Docking	11
	Lost in Space	8
	Distributed Control Architecture	10
	Inertia ID	10
Flight 4	Docking	12
	Lost in Space	17
	Inertia ID	10

A formation flight system would follow each of these steps in sequence: lost in space maneuver, beacon tracking to capture the other unit, and then perform formation flight or docking. Due to the limited 20 seconds experiment test time, the different maneuvers of a formation flight deployment sequence were separated and tests conducted individually. The data of each individual test were collected to demonstrate how the different algorithms create a formation flight system. The tests began with the demonstration over two days of the ability of two units to track their angular position with respect to each other (beacon track). Docking algorithms were also tested, to demonstrate control of separation between units. The "Lost in Space" algorithms demonstrated different ways in which two

satellites, deployed in space with a random attitude, can find each other by following a prescribed 3DOF rotation; the goal was to balance fuel consumption and duration of the maneuver. The ability of the SPHERES software to separate the tests into these steps allowed significant tests to be conducted in the short test-time of the KC-135.

I.3 Results at the MSFC Flat Floor

The Marshall Space Flight Center Flat Floor provides an environment similar to that of the MIT SSL: 3DOF tests under 1g conditions utilizing air carriages. But the MSFC facility provides a substantially expanded operational area of over 10 meter square (30 feet square). This extended area allows scientists to perform tests with up to five SPHERES, which is not physically possible at the MIT SSL; it enables more realistic lost-in-space demonstrations and docking maneuvers; and it allows the use of tethers several meters in length. The SPHERES team took advantage of these three options during two campaigns: three days in June 2004 and five days during December 2004. Tests included demonstration of formation flight maneuvers with two, three, and five spacecraft relevant to the TPF mission, docking algorithms over large separations, and tethered formations with two and three satellites. Table I.9 summarizes the tests conducted at the MSFC Flat Floor.

TABLE I.9 MSFC flat floor experiments

	Week 1			Week 2				
Algorithm	**Day 1**	**Day 2**	**Day 3**	**Day 1**	**Day 2**	**Day 3**	**Day 4**	**Day 5**
TPF Rotations	✓	✓	✓	✓	✓	✓	✓	✓
Docking		✓	✓					
Tether				✓	✓	✓	✓	✓

The formation flight maneuvers consisted of satellite deployment and tracking of a circular path. Tests with two units involved constant and expanding separations of 0.5m, 1m, and 1.5m to demonstrate both deployment and image capture maneuvers. The three and five satellite tests used constant separations to demonstrate formation flight algorithms.

The three unit tests utilized a separation of 1m between units, for a total array separation of 2m; five units formations maintained constant separations of 1m, for an array size of 4m.

The docking experiments demonstrated docking maneuvers with a free rotating target; this was achieved incrementally through four steps over two days. The first experiment consisted of 2 units performing a cooperative docking maneuver starting about 1.5m apart, with the chaser unit pointing in a random direction. The second configuration was close proximity tracking of a free rotating target to tune the control algorithm which controls movement in the tangential direction. Next, docking maneuvers with uncooperative targets were performed. For the third set of tests, the target was slowly drifting but not rotating; this allowed tests of the control algorithms in the presence of perturbations caused by plume impingement on the target. Finally, multiple docking maneuvers with a slowly rotating target were successfully performed.