

MEASURING ANGLES IN AN
ENVIRONMENTALLY EXPOSED GIMBAL
ASSEMBLY

FINAL TECHNICAL REPORT

Submitted by: Team # 5353

Jonathan A Cox

Martin M Lopez

Craig Warren McNabb (Project Leader)

Siddharth Narang

Matthew Ryan Schellenberg

Conor Staples

Submitted to:

Mentor: Gerald Pine

Sponsor: Raytheon Missile Systems

Jacob A Lilevjen

Christopher E Pentland

Kristy Pearson

Brian S Scott

Raytheon Missile Systems

Submission Date: May 1st, 2013

Interdisciplinary Engineering Design Program

azengineering



The University of Arizona Electronic Theses and Dissertations Reproduction and Distribution Rights Form

The UA Campus Repository supports the dissemination and preservation of scholarship produced by University of Arizona faculty, researchers, and students. The University Library, in collaboration with the Honors College, has established a collection in the UA Campus Repository to share, archive, and preserve undergraduate Honors theses.

Theses that are submitted to the UA Campus Repository are available for public view. Submission of your thesis to the Repository provides an opportunity for you to showcase your work to graduate schools and future employers. It also allows for your work to be accessed by others in your discipline, enabling you to contribute to the knowledge base in your field. Your signature on this consent form will determine whether your thesis is included in the repository.

Name (Last, First, Middle) <i>Steples Connor William</i>
Degree title (eg BA, BS, BSE, BSB, BFA): <i>BS</i>
Honors area (eg Molecular and Cellular Biology, English, Studio Art): <i>Optical Science and Engineering</i>
Date thesis submitted to Honors College: <i>5/1/13</i>
Title of Honors thesis: <i>Measuring Angles in an Environmentally Exposed Gimbal Assembly</i>
The University of Arizona Library Release Agreement <p>I hereby grant to the University of Arizona Library the nonexclusive worldwide right to reproduce and distribute my dissertation or thesis and abstract (herein, the "licensed materials"), in whole or in part, in any and all media of distribution and in any format in existence now or developed in the future. I represent and warrant to the University of Arizona that the licensed materials are my original work, that I am the sole owner of all rights in and to the licensed materials, and that none of the licensed materials infringe or violate the rights of others. I further represent that I have obtained all necessary rights to permit the University of Arizona Library to reproduce and distribute any nonpublic third party software necessary to access, display, run or print my dissertation or thesis. I acknowledge that University of Arizona Library may elect not to distribute my dissertation or thesis in digital format if, in its reasonable judgment, it believes all such rights have not been secured.</p>
<input checked="" type="checkbox"/> Yes, make my thesis available in the UA Campus Repository! Student signature: <i>Connor Steples</i> Date: <i>5/1/13</i> Thesis advisor signature: <i>David D. Pine</i> Date: <i>5/1/13</i>
<input type="checkbox"/> No, do not release my thesis to the UA Campus Repository. Student signature: _____ Date: _____

MEASURING ANGLES IN AN
ENVIRONMENTALLY EXPOSED GIMBAL
ASSEMBLY
FINAL TECHNICAL REPORT

Submitted by: Team # 5353
Jonathan A Cox
Martin M Lopez
Craig Warren McNabb (Project Leader)
Siddharth Narang
Matthew Ryan Schellenberg
Conor Staples

Submitted to:

Mentor: Gerald Pine
Sponsor: Raytheon Missile Systems
Jacob A Lilevjen
Christopher E Pentland
Kristy Pearson
Brian S Scott
Raytheon Missile Systems

Submission Date: May 1st, 2013

Interdisciplinary Engineering Design Program



ABSTRACT

Raytheon Missile Systems has a need for an angular rate measurement system to be implemented within the allocated space at the front of an existing weapon system known as Paveway™. Ultimately, the armament needs to be more accurate and thus the need for this measurement system, which measures the angular rate of change between two bodies rotating about each other on a two axis gimbal assembly, was born. The system must be environmentally sealed, operate on a fixed voltage supplied by the weapon, have low friction, provide low error measurements, and most importantly be low cost. The end design involved the usage of two COTS (commercial off the shelf) gyroscopes that would not only be able to measure the angular rate of change of both pitch and yaw, but be able to easily satisfy all of the other requirements posed by the sponsor as well. The final technical report discusses the high level design process, hardware and software implementation, as well as testing results and project conclusion. Additionally, the project's mitigation plan, which was a large part of the project, will also be discussed further in Appendix C.

Table of Contents

1	Introduction.....	5
1.1	Scope of the Document	5
1.2	Changes Made Since Critical Design Review (CDR).....	5
1.3	Problem Statement and Background Information	5
1.4	Scope of Project	5
1.5	Product Expectations.....	5
1.6	Customer Description.....	6
2	System Requirements.....	6
2.1	Functional Requirements	6
2.2	Performance Requirements	7
2.3	Utilization Requirements	7
3	Summary of Preliminary Design Results (PDR).....	8
3.1	Concepts Considered.....	8
3.2	Preferred Concept	8
3.3	Changes to the Concept After PDR.....	8
4	Top-Level Design of Final Design Concept	8
4.1	Gyroscopes.....	8
5	Hardware Subsystem/Sub-assembly and Interface Design	9
5.1	Micro-electro-mechanical Systems (MEMS) Gyroscopes	9
5.1.1	Operation.....	9
6	Software Algorithm Description and Interface Document	12
6.1	Modules.....	12
6.1.1	MPLAB IDE Modules.....	13
6.1.2	Header Files Modules.....	15
6.1.3	Teraterm	15
6.2	SPI Interface.....	15
6.3	Hardware and Software Interfacing	16
6.3.1	Cerebot MX3cK Microprocessor	16
6.3.2	ADXRS453 Gyroscope	18
7	Analysis	19
7.1	Requirement 101	19
7.2	Requirement 102.....	20
7.3	Requirement 103.....	20
7.4	Requirement 104.....	20
7.5	Requirement 105.....	20
7.6	Requirement 106.....	21
7.7	Requirement 202.....	21

8	Development plan and implementation	22
8.1	Electronic Gyroscope:	22
9	Requirements review / Acceptance test / performance	24
9.1	Functional Requirements	24
9.2	Performance Requirements	25
10	Closure.....	28
10.1	Summary:	28
10.2	Challenges:	28
10.3	Improvements:	28
11	Appendix A Gyroscope Software.....	30
11.1	Overview:.....	30
11.2	Written Software: Main.C	30
11.3	ADXRS453.h	34
11.4	Communication.h.....	35
11.5	Console.C.....	36
11.6	Console.h	38
11.7	Delays.C.....	38
11.8	Delays.h	39
11.9	Working Code	39
12	Appendix B: Mechanical Drawings.....	40
13	Appendix C: Report for the laser spot imaging system	42
1	Overview:	42
2	Top-Level Design of Final Alternative Design Concept	42
2.1	Laser Spot Imaging System (LSIS).....	42
3	Hardware Subsystem/Sub-assembly and Interface Design.....	44
3.1	Position Sensitive Detector	44
3.2	S5991-01 Position Sensitive Detector Specifics	45
3.2.1	Dimensional Diagrams	45
3.2.2	Specifications	46
3.2.3	Electrical Properties	48
3.2.4	Detector Error and Effective Area.....	50
3.2.5	Important Handling Notes	50
3.3	Laser Diode	51
3.4	System Mounts.....	53
3.5	Testing Mounts	54
4	Software Algorithm Description and Interface Document	56
4.1	LSIS Angular Calculation Engine.....	56
4.1.1	Cartesian Coordinates Calculation	56

4.1.2	Cartesian to Spherical Coordinate Conversion.....	57
4.1.3	Cartesian to Spherical Coordinate Conversion Derivation.....	58
4.1.4	Details for and Utilization of the 16-Bit 28-Pin Starter Board by MICROCHIP	61
4.1.5	UART Hardware Interface	63
4.1.6	Code Used for Lab Testing of the LSIS Design	64
5	Changes to the alternative lsis design	67
5.1	Laser diode change.....	67
6	Initial testing.....	68
7	Elimination From Project	69

1 INTRODUCTION

1.1 Scope of the Document

The scope of this document includes the detailed analysis of our primary design that uses gyroscopes to measure angular rate of change. This document describes how gyroscopes work, calculations regarding measurements, and how they will be utilized in our project to satisfy our system requirements. The implementation of the electronic gyroscope design includes testing through the given military standards so that in fact it will satisfy the system requirements. This document includes our risk analysis and the pursuit of a secondary design or the Laser Spot Imaging System (LSIS), which was our risk mitigation plan (seen in Appendix C). This document will explain our risk analysis and implementation of our design that led us to satisfy our requirements.

1.2 Changes Made Since Critical Design Review (CDR)

The major changes that we have made since the Preliminary Design Review were narrowing our number of ideas from three ideas to two. At the CDR stage we have identified two different designs that at our sponsor's discretion wished to be pursued. The two design concepts include the electronic gyroscope design and the Laser Spot Imaging System (LSIS). The implementation of both designs was done simultaneously and in parallel to each other. The goal was to find and eliminate the design that would not uphold our requirements. Due to time constraints and the precision of our hardware components we have eliminated the LSIS system, though a formal report for the system is included in Appendix C. Given our design and analysis we believe that our sponsor will have the time and capability to explore this option. The implementation of the electronic gyroscope idea was pursued and tested in the different environments complying the military standards that our sponsor requested.

1.3 Problem Statement and Background Information

Our team must design a system that measures angular rate of change of the birdie with respect to the GEDA in the Paveway™ weapon system. This system will be implemented in the current weapon to enable the tracking of moving targets. Any system we design must meet all of the military specifications for temperature, weather, and vibration as described in our system requirements.

1.4 Scope of Project

The system must output angular rate of change in two axes: pitch and yaw. The system shall meet all of the accuracy, bias, and linearity requirements as specified by our sponsor. The system shall also meet all of the military weather requirements as specified by our sponsor. This project does not require angular position measurements between the two bodies, but this data could be used towards finding the angular rate of change if needed.

1.5 Product Expectations

The product is expected to meet all of the system requirements set forth by the customer. The product must also fit within the existing weapon design. Most importantly the design must be cost effective for the customer end use. The customer made cost his most important requirement for the final design. If both systems work as expected and fulfill all requirements the final design submitted to the customer will be the one that can be produced at a lower cost. The total cost will also take into account the labor required to implement the final design.

1.6 Customer Description

Our customer, Raytheon, is a technology and innovation driven leader specializing in defense, homeland security, and other government markets throughout the world. Our sponsor, Brian Scott, wants us to design a system that measures the angular rate of change in the Paveway™ weapon system. The current weapon system consists of a laser-guided bomb that guides towards a ground target with the assistance of a laser-illuminated target. The new angular rate of change design will be incorporated to aid with the need to hit moving targets.

2 SYSTEM REQUIREMENTS

The requirements were derived from the customers need for us to design, build, and test an accurate system to measure the angular rate of change between two moving bodies. The process began with the interpretation of the problem statement and the initial meeting with the sponsor. All ideas were then written down and given a measurable quantity. Once these were agreed upon they were then established as requirements. The entire list of requirements were broken down into three sub-sections: functional, performance, and utilization requirements.

2.1 Functional Requirements

The functional requirements are based upon the needs set forth by the sponsor Raytheon. The new angular measurement system needs to measure the angular rate of change with known certainty to help guide the weapon system towards a moving target. The functional requirements are shown in Table 1.

Table 1: Functional Requirements

Number	Type	Name	Description	Test Method		Test Description
				A = Analyze	T = Test	
				A	T	
101	Functional	Angular Rate Scale Factor Error	The system shall measure the angular rate of the two free axes of a gimbal assembly within 1% (3 sigma)		X	Must
102	Functional	Angular Rate Non-linearity	The nonlinearity of the angular rate measurement shall be less than 1% (3 sigma)		X	Must
103	Functional	Angular Rate Bias	The angular rate bias shall be less than 490 deg/hr 3 (sigma)		X	Must
104	Functional	Angle Range	The system shall measure angular rates of change within the limits of constrained motion of the gimbal assembly, defined as ± 30 degrees in any free axis		X	Must
105	Functional	Input Power	The system shall be powered with +5V and pull no more than 100 milliamps		X	Must
106	Functional	Frequency Response	The frequency response at -3dB gain shall be greater than 65 Hz		X	Must
107	Functional	Output	The system shall output all measured values to MATLAB or equivalent program via serial protocol		X	Desired

2.2 Performance Requirements

The performance requirements are based upon the typical environments seen by the Paveway™ weapon system and the methods set forth in MIL-STD-810F. MIL-STD-810F is the military standard requirement test document used by the customer to verify weapon systems are capable of withstanding military grade environments. The Paveway™ weapon system will be tested for high and low temperatures and a vibration environment. The additional requirements of sand, rain, and electromagnetic interference (EMI) exposure will be analyzed as part of the overall Paveway™ weapon system. The performance requirements are shown in Table 2.

Table 2: Performance Requirements

Number	Type	Name	Description	Test Method		Test Description														
				A = Analyze	T = Test															
				A	T															
201	Performance	Operating Temperature	The system shall have the capability to withstand a temperature range of -55C to +85C (MIL-STD 810)			X														
202	Performance	Vibration	<p>The system shall have the capability to withstand vibration requirements in accordance with MIL-STD-810 and the Paveway™ weapon system as described in the following vibration profile.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="2">Paveway™ Vibration Breakpoint Table</th> </tr> <tr> <th>Freq</th> <th>PSD Level</th> </tr> <tr> <th>(Hz)</th> <th>(g²/Hz)</th> </tr> </thead> <tbody> <tr> <td>20</td> <td>0.00162</td> </tr> <tr> <td>100</td> <td>0.04</td> </tr> <tr> <td>1000</td> <td>0.04</td> </tr> <tr> <td>2000</td> <td>0.01005</td> </tr> </tbody> </table>	Paveway™ Vibration Breakpoint Table		Freq	PSD Level	(Hz)	(g ² /Hz)	20	0.00162	100	0.04	1000	0.04	2000	0.01005			X
Paveway™ Vibration Breakpoint Table																				
Freq	PSD Level																			
(Hz)	(g ² /Hz)																			
20	0.00162																			
100	0.04																			
1000	0.04																			
2000	0.01005																			
203	Performance	Rain	The system shall have the capability to withstand exposure to rain (need amount of rain, duration) (MIL-STD 810)	X																
204	Performance	Sand	The system shall have the capability to withstand exposure to sand (need amount of sand, duration) (MIL-STD 810)	X																
205	Performance	EMI	The system shall not produce an electric or magnetic field in its immediate vicinity in accordance with MIL-STD-810.	X																

2.3 Utilization Requirements

The last set of requirements are designed to meet the life expectancy and financial goals of the sponsor. The current system used by the customer is relatively expensive in comparison to the total cost of the Paveway™ weapon system. The main goal of our project is to create a inexpensive and accurate way to meet these requirements. The utilization requirements are shown in Table 3.

Table 3: Utilization Requirements

Number	Type	Name	Description	Test Method		Test Description
				A = Analyze	T = Test	
				A	T	
301	Utilization	Manufacturing Cost	The cost of the system shall not exceed \$300.00 at a manufacturing rate of 1000 units per year.	X		
302	Utilization	Shelf Life	The system shall have a shelf life of 20 years.	X		

3 SUMMARY OF PRELIMINARY DESIGN RESULTS (PDR)

3.1 Concepts Considered

The four initial concepts we considered to meet the customer's requirements were: gyroscopes, the LSIS design, optical encoders, and potentiometers. Potentiometers were eliminated early in the design process due to measurement errors associated with them over the temperature ranges required by the weapon system. Optical encoders were eliminated because the accuracy required exceeds the cost requirement set forth.

3.2 Preferred Concept

Our preferred concept is the gyroscope idea. This concept offers the most established technology and greatest chance to meet our system requirements. The main issue with this idea is a decrease in performance over large temperature ranges. The LSIS is a less mature technology that involves higher risk. The novelty of this system however offers tremendous upside if it proves feasible. This design will potentially be less expensive and more accurate if the components perform as expected during the testing phase of this design.

3.3 Changes to the Concept After PDR

Since PDR we have given thought as to how to mitigate the risks involved in each of our concepts. The idea of using a thermoelectric heating/cooling element to keep the temperature within a desired range to maximize system performance has been discussed. This would alleviate the need to consider operation of either system at extreme temperatures. This mitigation option will be researched and decided upon after the testing phase has begun.

4 TOP-LEVEL DESIGN OF FINAL DESIGN CONCEPT

4.1 Gyroscopes

This section describes the top level design of the concept of pitch and yaw angular rate measurement by gyroscopes located in the birdie. For the purpose of this experiment, the gyroscopes will send the required values to a microcontroller that will send the values via serial protocol that would be processed. In practical application, the microcontroller in the birdie will interface with the microcontroller receiving output from the GEDA inertial measurement unit to calculate the relative rate of change of orientation between the

components. This attitude rate of change will then be used for dynamic feedback. A practical out value may be MATLAB or an equivalent software program. A block diagram describing the interaction of these components is shown in Figure 1.

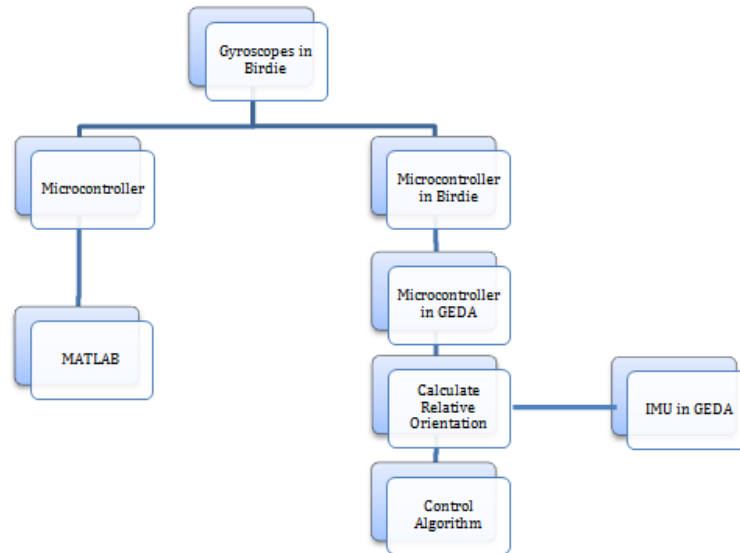


Figure 1: High Level Block Diagram of the Gyroscope System

The gyroscopes will be enclosed in the birdie capsule of the Raytheon Paveway™ weapon system. A modeled representation of this setup is included below in Figure 2.

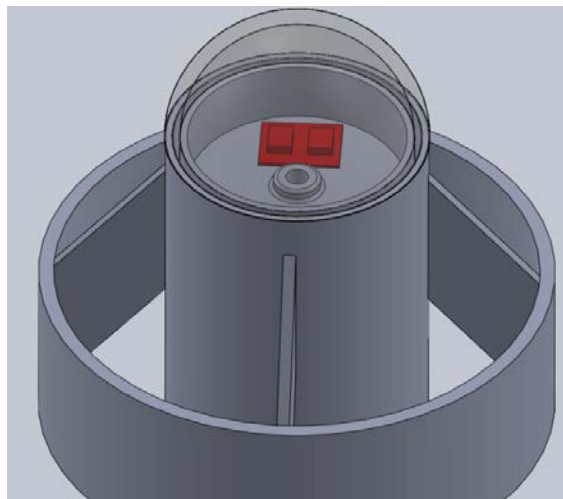


Figure 2: Gyroscope Assembly

5 HARDWARE SUBSYSTEM/SUB-ASSEMBLY AND INTERFACE DESIGN

5.1 Micro-electro-mechanical Systems (MEMS) Gyroscopes

5.1.1 Operation

This section describes the operation of MEMS gyroscopes and the mechanical interface of the ADXRS453 with the system. The operation of a MEMS gyroscope is based on the principle that a vibrating object tends to continue vibrating in the same plane as it rotates. This vibration is induced by the Coriolis Effect described by Equation (1).

$$a_c = -2(v \times \Omega) \tag{1}$$

where v is the velocity in plane and Ω is the out of plane angular rotation as shown in Figure 3.

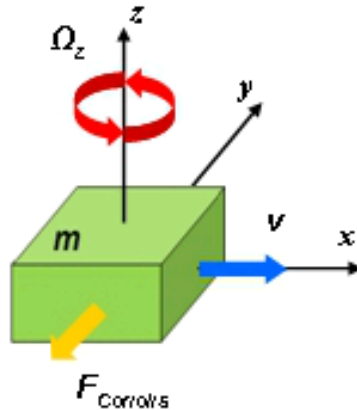


Figure 3: Gyroscope and the Coriolis Effect

When the gyroscope moves in the x direction with velocity v and an angular rate Ω is applied, the Coriolis Effect applies a force in the y direction that in turn causes a vibration due to the attempted displacement of the object. This vibration is converted to an electrical signal via capacitors and is assumed to be proportional to the magnitude of the angular rate of rotation Ω . Figure 4 shows a typical MEMS gyro and angular rate measurement setup by driving capacitive plates.

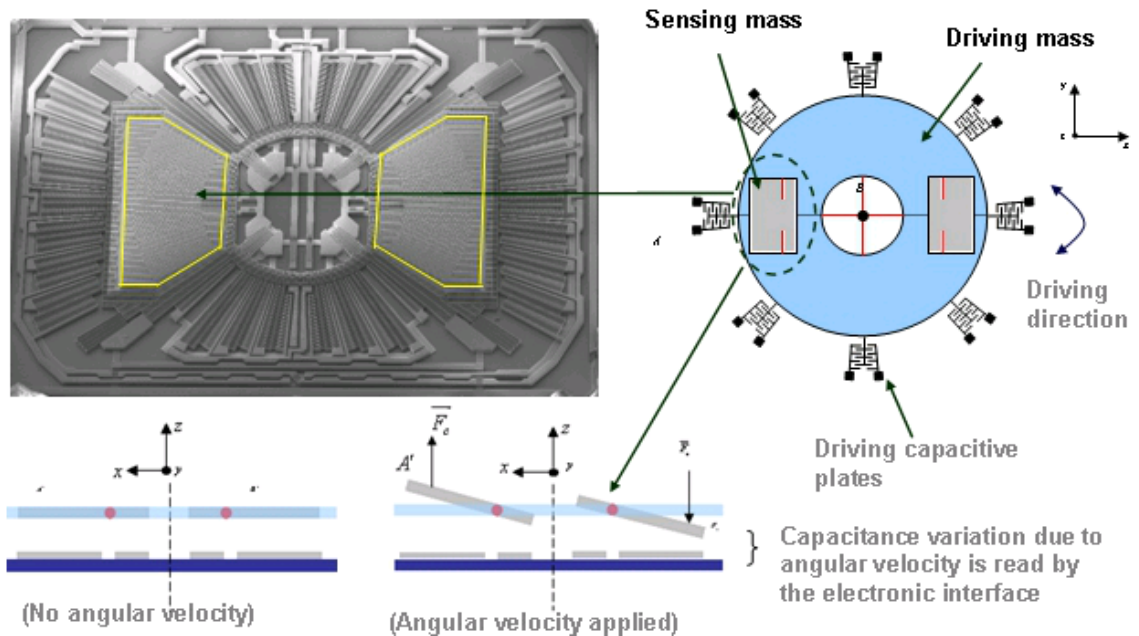


Figure 4: MEMS Gyroscope Capacitance Measurement

The gyroscope is fabricated on a silicon wafer and vibrates at a characteristic frequency due to a quartz oscillator. The conversion of mechanical to electrical energy is often accomplished by piezoelectric elements. The axis of rotation of a MEMS gyroscope is specified along with the null output (zero rate level) and sensitivity as described in Figure 5.

- In plane axis (pitch / roll, x / y)**
 - Is the axis along the sensor package surface (roll /pitch)
- Out of plane axis (yaw, z)**
 - Is the axis perpendicular to the sensor package surface (yaw)
- Zero rate level**
 - Sensor output (digital/analog) when no angular rate is applied
- Sensitivity (mV/dps<sup>(*)
 - Is the ratio between sensor output and angular rate applied (gain of the sensor)</sup>**

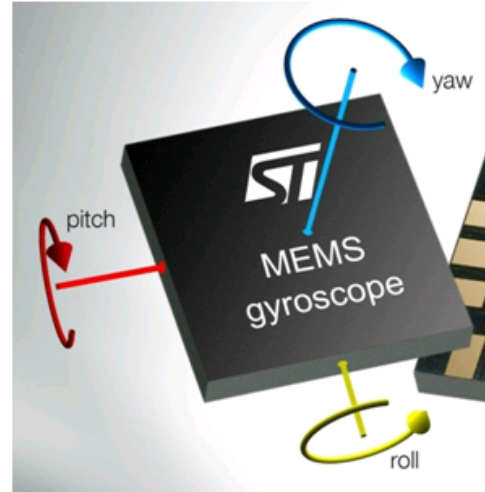


Figure 5: MEMS Gyroscope Characteristics

The ADXRS453 measures angular rate in one axis. Two gyroscopes were therefore used to measure the angular rate of change in both axes. Figure 6 shows a single ADXRS453 evaluation board with gyroscope. Complete mechanical drawings are included in the Appendix.

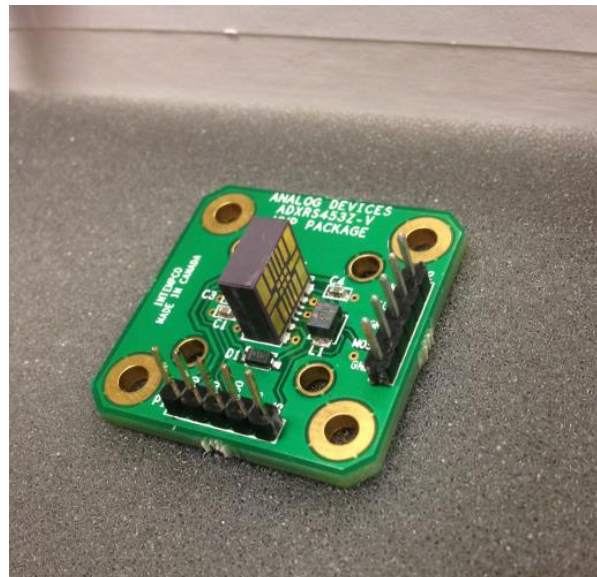


Figure 6: ADXRS453 Evaluation Board

A presentation assembly was produced to demonstrate the measurement capabilities of the gyroscope system. The assembly consists of the Manfrotto MH054M0 Magnesium Ball Head gimbal (manufactured

for high performance cameras and tripods) and a mounting plate, shown in Figure 7 and Figure 8, respectively. Complete drawings of the mount plate, which was fabricated by 3D printing, are included in the Appendix.



Figure 7: Manfrotto MH054M0 Ball Head

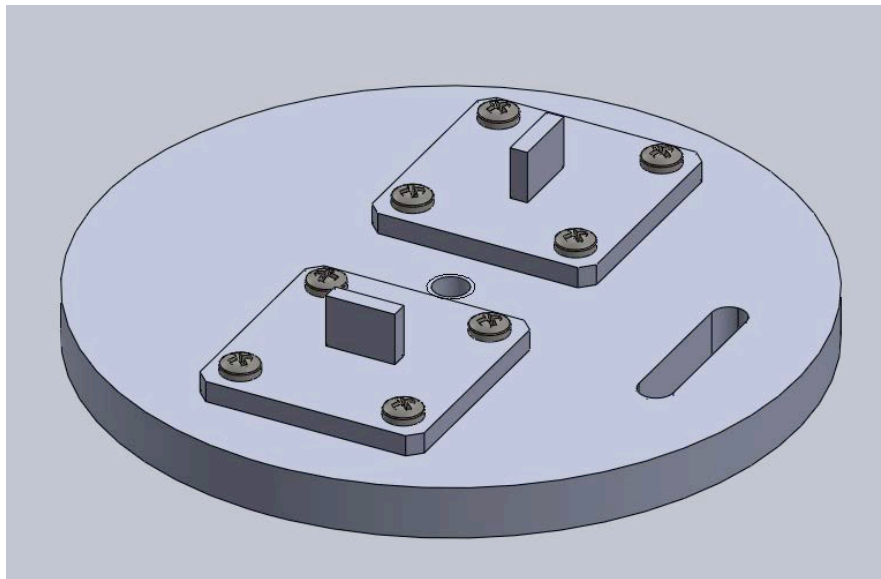


Figure 8: Mount Plate for Presentation Display

6 SOFTWARE ALGORITHM DESCRIPTION AND INTERFACE DOCUMENT

6.1 Modules

Software modules are necessary to provide an input/output interface between the gyroscopes and the Paveway™ weapon system. The testing of this interface will be completed using the modules listed in Figure 9. The modules are explained in detail in Sections 6.1 through 6.3.

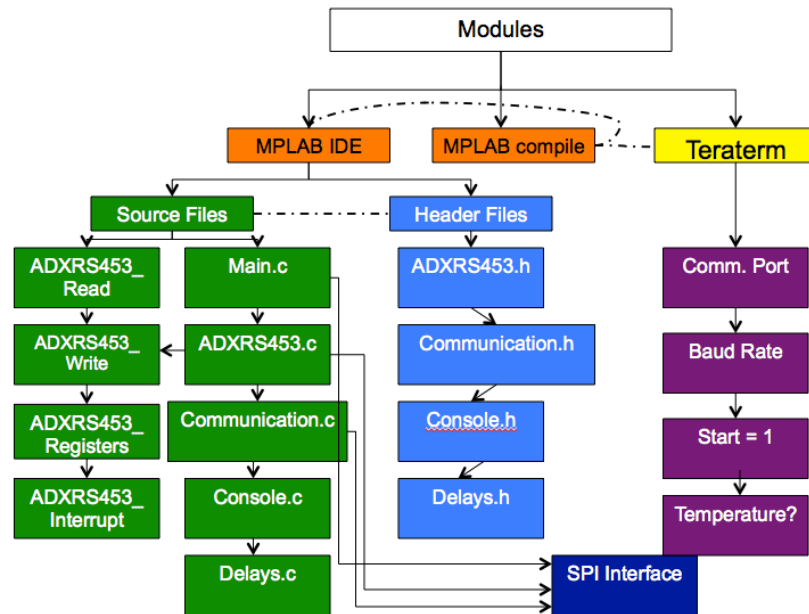


Figure 9: Module Block Diagram

6.1.1 MPLAB IDE Modules

6.1.1.1 ADXRS453_Read ():

This function will begin transmission using an address from the gyroscope and send it to a temporary register. After this particular transmission ends, the slave (gyroscope pin) will request the appropriate data required and receive the transmission from the microprocessor. After this functionality is initialized, the transmission will end again and move to ADXRS453_Write () function.

6.1.1.2 ADXRS453_Write ():

This function will receive the transmitted data from the gyroscope in the MISO (master in slave out) fashion and write the sent values from gyroscope in the write register, which will further be initialized in the Registers () module

6.1.1.3 ADXRS453_Registers ():

This functionality will receive the values from the Read () and Write () registers in form of channels and store the data in a vector, which will further be utilized to perform calculations based on requirements.

6.1.1.4 ADXRS453_Interrupt ():

An interrupt function is an asynchronous signal indicating the need for attention or a synchronous event (in software) indicating the need for change in execution. This functionality can be implemented on both hardware and software level. On the hardware level, the interrupt function saves its current state of execution and begins execution of an interrupt handler. On software level, these functions are implemented as instructions in an instruction set, which cause a context switch to an interrupt handler. We use interrupts to handle multitasking when we are performing real-time computing.

6.1.1.5 Main.c ():

The Main function in the code will set the configuration of the PIC32 chip on the Cerebot microprocessor and set the oscillator in a phase locked loop configuration. It will set the multiplier to 20x and divider to 2x for proper functioning of time delays function in the program with respect to PIC 32 chip. This module not only shows the operation of commands used for getting gyro output but also establishes a link between these commands and Teraterm. The module starts with a floating value and converts it to character array with 3 digits of accuracy and returns the converted value of the gyro data transmitted via SPI.

6.1.1.6 ADXRS453.C ():

This function/module in our program will perform the initialization, read/write and clears the parity bit in order ensure that the overall parity of data word is odd. The module will first check for parity bit and then initialize the ADXRS453 gyro to check if its active. '1' meaning if initialization was successful and ID starts is 0x52, '0' meaning if initialization was unsuccessful. Then the 'ADXRS453_Data' function in the program starts reading from the gyro using MISO (Master In Slave Out) pins and reads the register corresponding to our angular rate, which is 0x00 and 0x01. Using MOSI (Master Out Slave In) configuration pins, the Cerebot microprocessor will extract the data in a 'Rx/Tx' (Receive/Transmit) fashion.

6.1.1.7 Communication.c ():

This function will initialize the SPI interface by setting the MSB (most significant bits) and LSB (least significant bits) using a transfer format of 0x0 and 0x1 respectively. Using constraints like clock frequency, clock polarity and clock edge, it will synchronize the frequency of Cerebot microprocessor with the adxrs453 gyroscope.

6.1.1.8 Console.c ():

This module initializes the UART communication peripheral. Universal Asynchronous Receiver/Transmitter (UART) is used to communicate between components namely PIC, desktop and modem. The connection established via UART is initialized on a mutually known baud rate and requires a minimum of two connections namely TX (transmitting) and RX (receiving). These two connections will be established in Read () and Write () functions where transmission of data between two hardware components is taking place.

6.1.1.9 Delays.c ():

This function creates a time delay in milliseconds when moving from Read, Write and Transmission operations.

6.1.2 Header Files Modules

6.1.2.1 ADXRS453.h:

This header file provides libraries that will provide global definitions and macros for source file ADXRS453.c described above. This file will include all register mappings like sensor data; write data, read data, rate, temperature, faulty registers, check bits etc.

6.1.2.2 Communication.h

This file provides high-level functions needed to use the SPI and I2C interface supported by the MICROCHIP microprocessor. This simple serial protocol will help us to connect multiple devices in a master-slave relationship. Multiple master devices may share single bus. The same device may function as both master and a slave in different executions when it comes to calculations. Finally, it will initialize the clock frequency, clock polarity and clock edge of the SPI and I2C interface.

6.1.2.3 Console.h:

This header file will help in simulating and programming the microprocessor. It will establish a link between UART and the commands user inputs in Teraterm to display the rate data and temperature.

6.1.2.4 Delays.h:

This file will perform the calculations of clock timing in the microprocessor and will compliment the Delays.c () module described above.

6.1.3 Teraterm

In order to display output from the microprocessor to Teraterm, we will be establishing an interface between both MPLAB and communication ports respective to the microprocessors. We will set the baud rate to 9600 and setup the communication ports to initialize the “*start = 1*” command to output gyro data. In order to display temperature we will input the command: “*temperature?*”

6.2 SPI Interface

In order to read the data transmitted from the gyroscope, we need to program it using Serial Peripheral Interface (SPI). Using this interface, the device will communicate in master/slave mode where master is the microprocessor and slave is the gyroscope. Microprocessor will initiate the data frame and two slave devices namely two sides of the gyroscope chip will respond accordingly.

The program will go through a hierarchy of commands to implement this serial connection according to specified sensitivity. Initially, this module will set the clock to required frequency with appropriate phase and polarity. Next, it will select the pin according to required reading from the chip and set it high. Finally it will begin transmission of data from gyroscope and output will be refreshed every 200 ms using shift registers.

6.3 Hardware and Software Interfacing

6.3.1 Cerebot MX3cK Microprocessor

The Cerebot MX3cK is a microcontroller development board, which was programmed using the PIC32 family of microprocessors. JE pin was utilized for programming in Serial Peripheral Interface (SPI) for the implementation of the Gyroscope design. This microprocessor is seen below in Figure 10 and the evaluation board schematic is shown below in Figure 11.

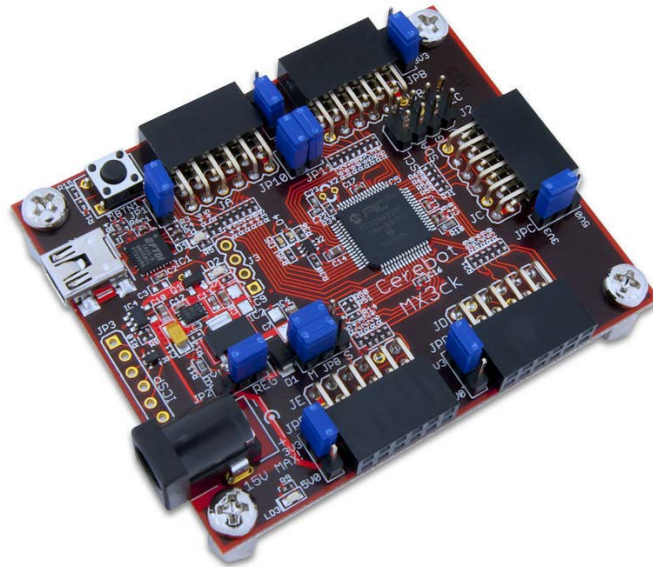


Figure 10: 32-Bit 42-Pin Starter Board by DIGILENT

Cerebot MX3cK Specifications:

- Microcontroller: PIC32MX320F128H
- Flash Memory: 128K
- RAM Memory: 16K
- Operating Voltage: 3.3V
- Max Operating Frequency: 80Mhz
- Typical operating current: 75mA
- Input Voltage (recommended): 7V to 15V
- Input Voltage (maximum): 20V
- I/O Pins: 42 total
- Analog Inputs: 12
- Analog input voltage range: 0V to 3.3V

EVALUATION BOARD SCHEMATIC

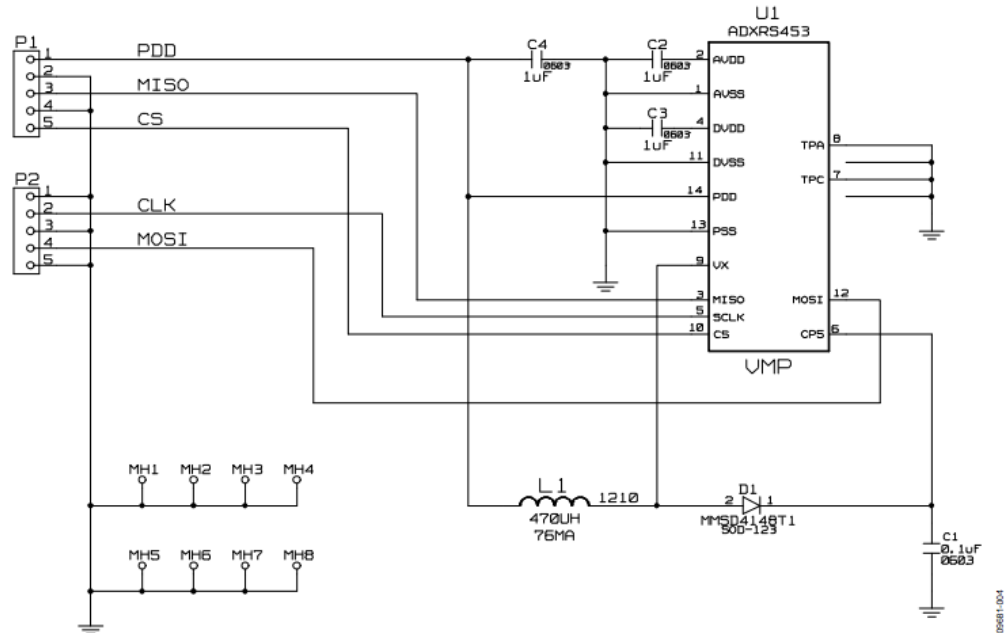


Figure 11: Cerebot Evaluation Board Schematic

DC Current per pin: +/-18mA

6.3.2 ADXR5453 Gyroscope

The ADXR5453 is an angular rate sensor (gyroscope) intended for industrial, instrumentation, and stabilization applications in high vibration environments. An advanced, differential, quad sensor design rejects the influence of linear acceleration, enabling the ADXR5453 to offer high accuracy rate sensing in harsh environments where shock and vibration are present. The device is shown below in Figure 123.

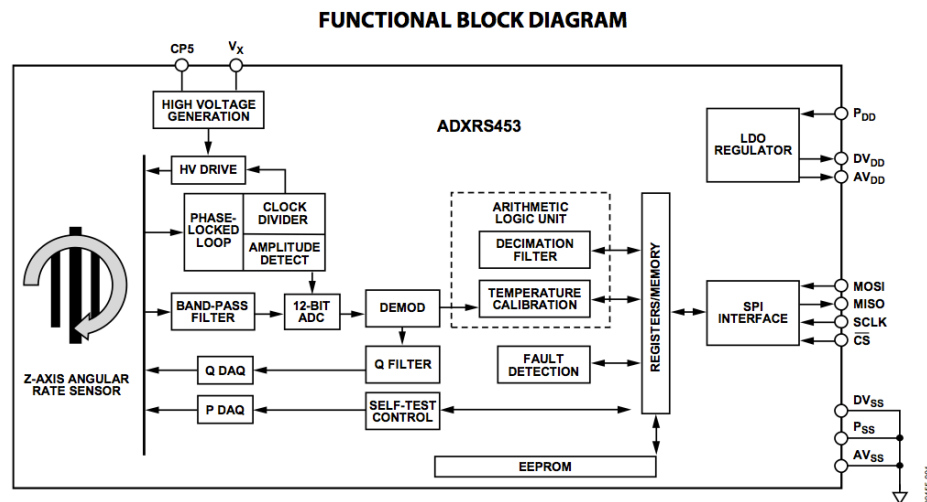


Figure 12: ADXR5453 Internal Structure

Features of this gyroscope include:

- Complete rate gyroscope on a single chip
- $\pm 300^\circ/\text{sec}$ angular rate sensing
- Ultrahigh vibration rejection: $0.01^\circ/\text{sec}/g$
- Excellent $16^\circ/\text{hour}$ null bias stability
- Internal temperature compensation
- 2000 g powered shock survivability
- SPI Digital output with 16-bit data word
- Low noise and low power
- 3.3 V to 5 V operation
- -40°C to $+105^\circ\text{C}$ operation
- Ultra small, light, and RoHS compliant

For more details on interfacing and schematics see Appendix B.

7 ANALYSIS

This section will cover the validation of requirements that could not be verified through testing (primarily those that require the testing of many sensors and environmental factors that are given as specifications in the data sheet).

7.1 Requirement 101

The system shall measure the angular rate of change of the two free axes of the gimbal assembly with less than 1 percent error (3 sigma).

Figure 13 shows the percent error of 16 gyroscopes from -50 to 130 degrees C. After software compensation for the characteristic bias of the sensors toward negative errors, all gyroscopes should meet this requirement over the operating temperature -40 to 72 degrees C.

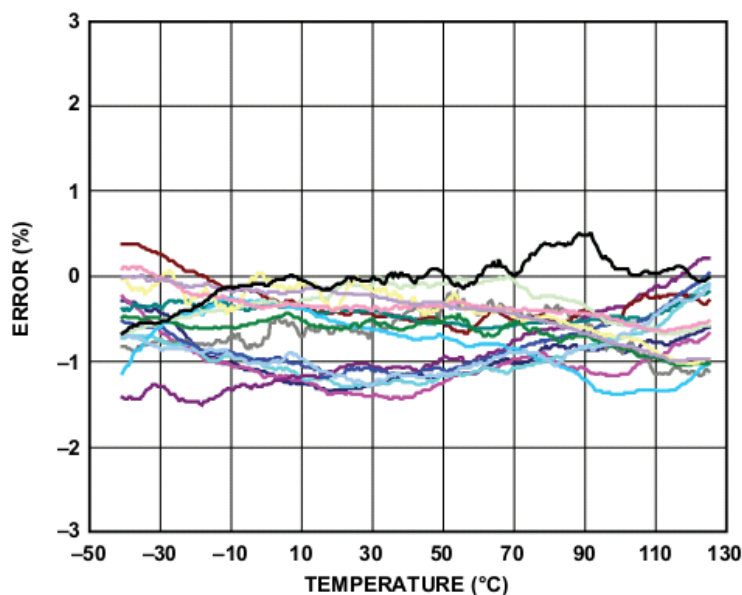


Figure 13: Accuracy over Temperature for 16 Devices Soldered on PCB

7.2 Requirement 102

The nonlinearity of the angular rate measurement shall be less than 1 percent (3 sigma).

This requirement was verified via testing (Section 9).

7.3 Requirement 103

The angular rate bias shall be less than 490 degrees/hour (3 sigma).

Figure 14 shows the null drift over temperature for 16 gyroscopes from -50 to 130 degrees C and verifies that each gyroscope should meet this requirement.

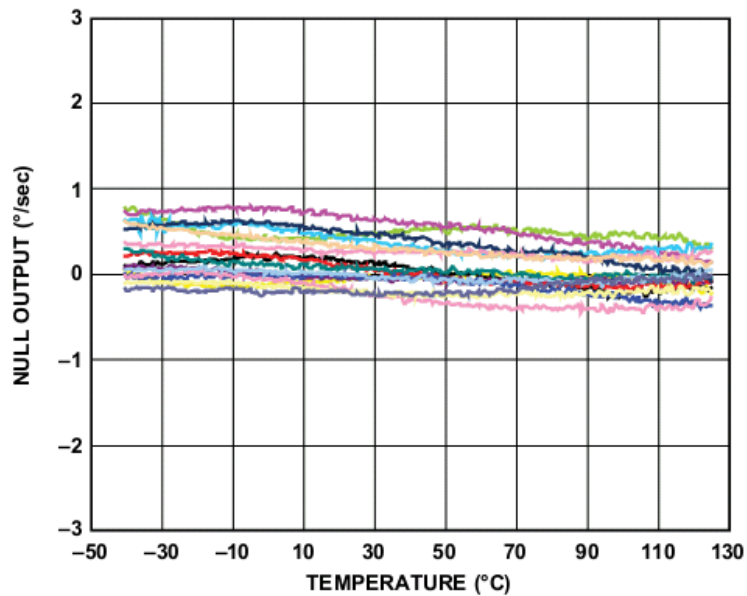


Figure 14: Null Drift over Temperature with 16 Devices Soldered on PCB

7.4 Requirement 104

The system shall measure angular rates of change within the limits of constrained motion of the gimbal assembly, defined as $\pm 30^\circ$ in any free axis.

The ADXRS453 has a measurement range of 300 degrees/second-400 degrees/second, as shown in Table 4.

Table 4: ADXRS453 Measurement Range

MEASUREMENT RANGE	Full-scale range	FSR	± 300	± 400	$^\circ/\text{sec}$
-------------------	------------------	-----	-----------	-----------	---------------------

7.5 Requirement 105

The system shall be powered with +5V and pull no more than 100 milliamps.

The supply voltage of the gyroscope is given in Table 5.

Table 5: Gyroscope Supply Voltage

Supply Voltage (P_{DD})	$-0.3 \bar{V}$ to $+6.0V$
-----------------------------	---------------------------

7.6 Requirement 106

The frequency response at -3dB gain shall be greater than 65 Hz.

Figure 15 shows that the system will meet this requirement.

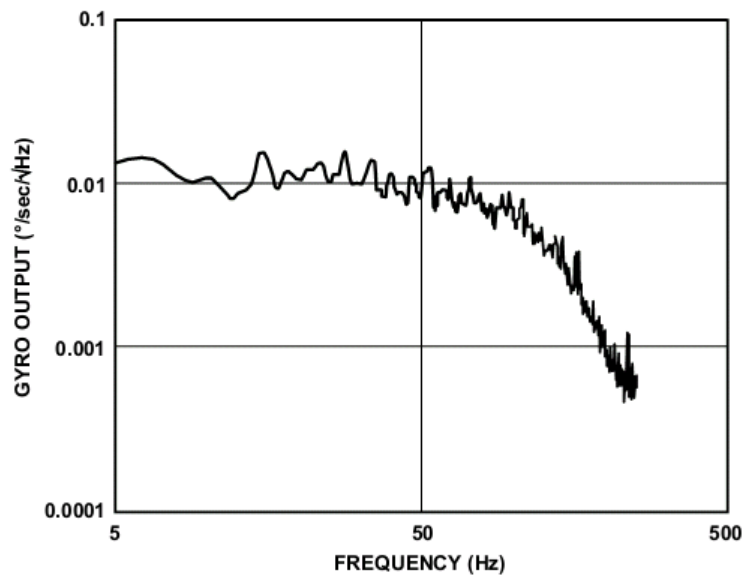


Figure 15: Frequency Response of ADXRS453

7.7 Requirement 202

The system shall have the capability to withstand a 100 G acceleration in accordance with MIL-STD-810.

Figure 16 shows the system response to a 99 G shock and that the effect on accuracy is negligible.

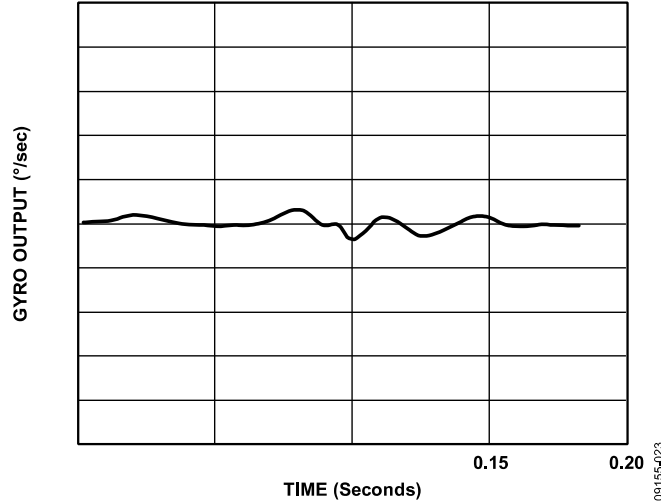


Figure 16: Response to 100 G Shock

8 DEVELOPMENT PLAN AND IMPLEMENTATION

8.1 Electronic Gyroscope:

The goal to seek a cost effective solution to accurately measure the angular rate of change between the Birdie and the GEDA consisted of two different implementations the LSIS and the electronic gyroscope. Developing both designs simultaneously allowed two different solutions until one design could not be implemented. Due to the discretion of our sponsor we implemented both designs until time could not permit us to continue developing the LSIS system.

The implementation of the electronic gyroscope design consisted of finding electronic gyroscopes that must measure the angular rate of change under the temperature requirements. The ADXRS453 performed at -55 Celsius and +85C. The programming was developed in C in bus known as Serial Peripheral Interface (SPI). The goal in programming the gyroscopes consisted of the communication between master and slave. The microprocessor will establish the data communication and establish the data frame as the master. The gyroscopes will perform as slaves and will be selected by the master or the microprocessor.

The time allocated for the development of the gyroscope design was delayed by testing the system in the different environments proposed in the in the requirements. The major concern of the gyroscope design was if the system will function under the different types of temperature as well as testing the rate bias. Our delay with testing was due to seeking a temperature chamber that will allow us to test the system in a specific temperature profile.

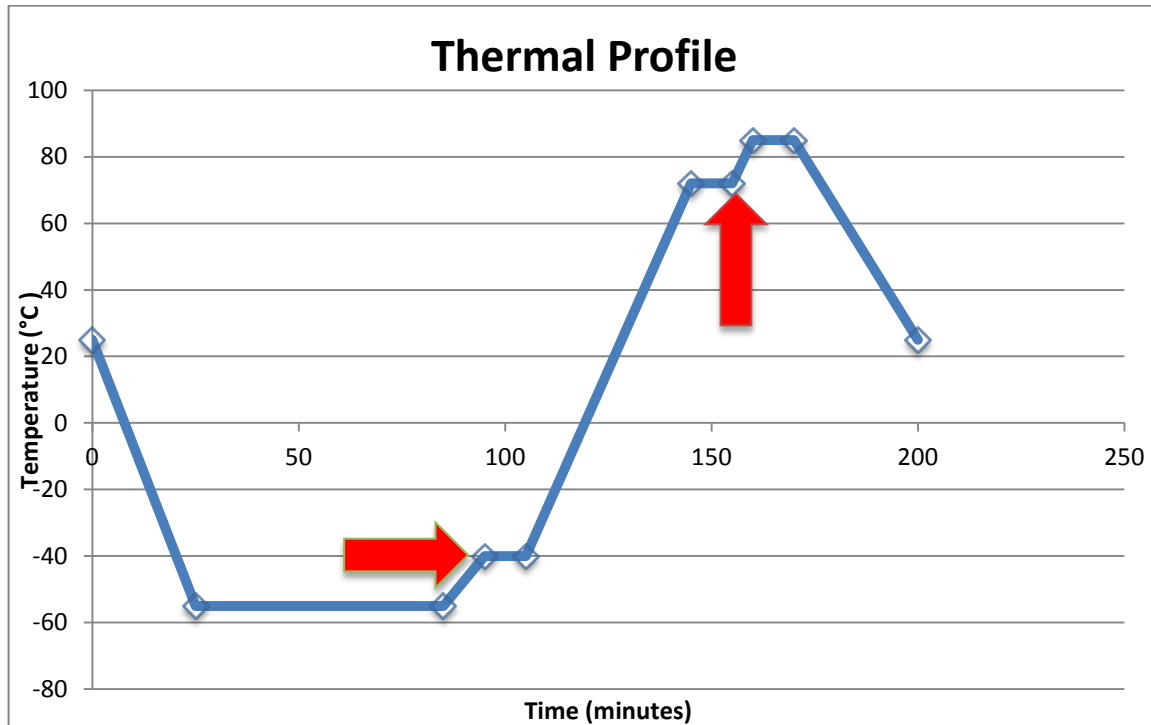


Figure 17: Testing temperature profile

The red arrows in Figure 17 in the testing temperature profile are temperature frequencies refer to the operational temperature that the gyroscope shall perform and accurately measure the angular rate. Mounts were fabricated to facilitate testing of the gyroscope on the rate table. An image of this mount is shown below in Figure 18 and Figure 19. A mechanical drawing is provided in the Appendix.

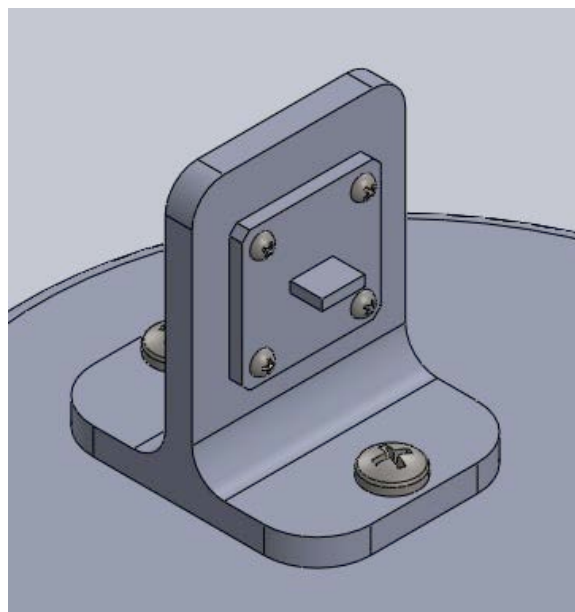


Figure 18: Mount for Testing

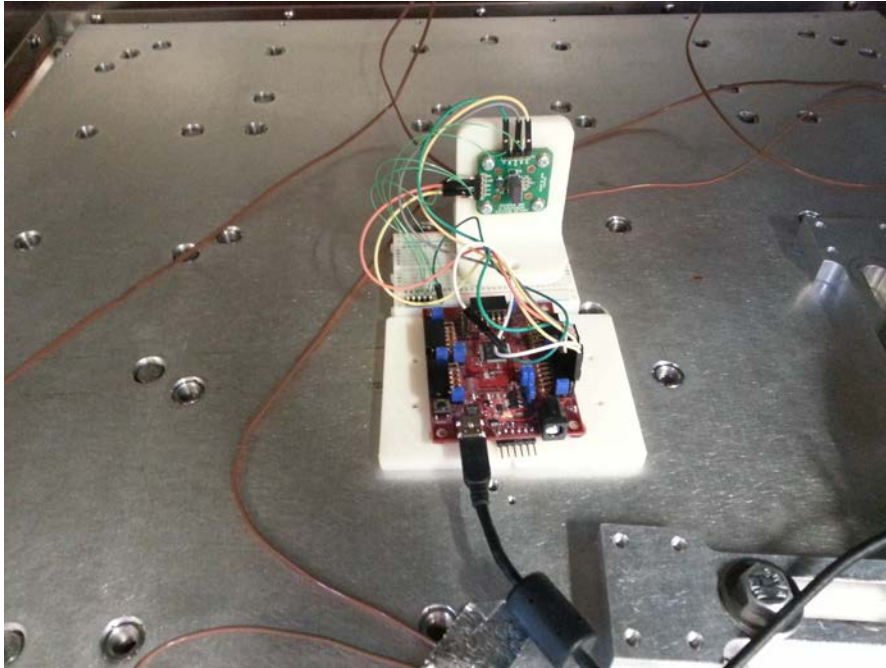


Figure 19: Gyroscope on Rate Table for Testing

9 REQUIREMENTS REVIEW / ACCEPTANCE TEST / PERFORMANCE

9.1 Functional Requirements

The development of the gyroscope design required a test acceptance test plan as well as a method to verify that all of the requirements were met. The functional requirements were tested first since the major concern was whether the gyroscopes would function in the different temperature extremes. This information can be seen below in Table 6.

Table 6: Functional Requirements Acceptance Test Plan

Number	Type	Name	Description	Test Method		Test Description	Test Result
				A = Analyze	T = Test		
				A	T		
101	Functional	Angular Rate Scale Factor Error	The system shall measure the angular rate of the two free axes of a gimbal assembly within 1% (3 sigma)		X	Must	The test angular rate was tested in a rate table and accurately measures the rate of change between 1 degree and from 5 - 40 degrees in 5 degree intervals.
102	Functional	Angular Rate Non-linearity	The nonlinearity of the angular rate measurement shall be less than 1% (3 sigma)		X	Must	Requirement was met. The static non-linearity came to be 0.54 %.
103	Functional	Angular Rate Bias	The angular rate bias shall be less than 490 deg/hr 3 (sigma)		X	Must	Requirement was met

104	Functional	Angle Range	The system shall measure angular rates of change within the limits of constrained motion of the gimbal assembly, defined as ± 30 degrees in any free axis		X	Must	Requirement verified through the
105	Functional	Input Power	The system shall be powered with +5V and pull no more than 100 milliamps		X	Must	Requirement verified with the use of the processor's 5 volt input to the gyroscope
106	Functional	Frequency Response	The frequency response at -3dB gain shall be greater than 65 Hz		X	Must	Requirement was met
107	Functional	Output	The system shall output all measured values to MATLAB or equivalent program via serial protocol		X	Desired	Requirement was met

9.2 Performance Requirements

The performance requirements were verified through testing and analysis. The temperature requirement has been met after testing the gyroscopes and the microprocessor in a temperature chamber. Our results can be illustrated in Table 7 below.

Table 7: Performance Requirements Acceptance Test Plan

Number	Type	Name	Description	Test Method A = Analyze T = Test		Test Description	Test Result	
				A	T			
201	Performance	Operating Temperature	The system shall have the capability to withstand a temperature range of -55C to +85C (MIL-STD 810)		X		Testing was verified in the environmental chamber. The system performed during the required temperatures.	
202	Performance	Vibration	The system shall have the capability to withstand vibration requirements in accordance with MIL-STD-810 and the Paveway™ weapon system as described in the following vibration profile.	X			Once the system has been mounted inside the Birdie and the components have been soldered and connected to the remaining IMU component of the Paveway weapon system.	
			Paveway™ Vibration					
			Breakpoint Table					
			Freq					PSD Level
			(Hz)					(g ² /Hz)
			20					0.00162
			100					0.04
			1000					0.04
2000	0.01005							

203	Performance	Rain	The system shall have the capability to withstand exposure to rain (need amount of rain, duration) (MIL-STD 810)	X			
204	Performance	Sand	The system shall have the capability to withstand exposure to sand (need amount of sand, duration) (MIL-STD 810)	X			
205	Performance	EMI	The system shall not produce an electric or magnetic field in its immediate vicinity in accordance with MIL-STD-810.	X			

The gyroscopes were found to measure within 1% accuracy over the operating temperature range (-40 to 72 degrees C) as well as over angular rates (0 to 40 degrees per second). Figure 20 shows this behavior.

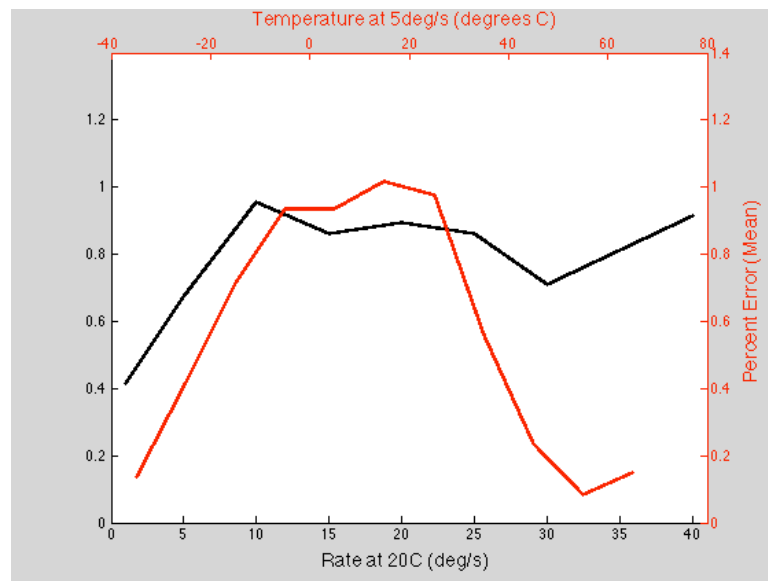


Figure 20: Percent Error Over Rate and Temperature

The random error was found to be less than 0.2 degrees over the operating temperature and various rates, which is negligible for most measurements. Figure 21 shows these results.

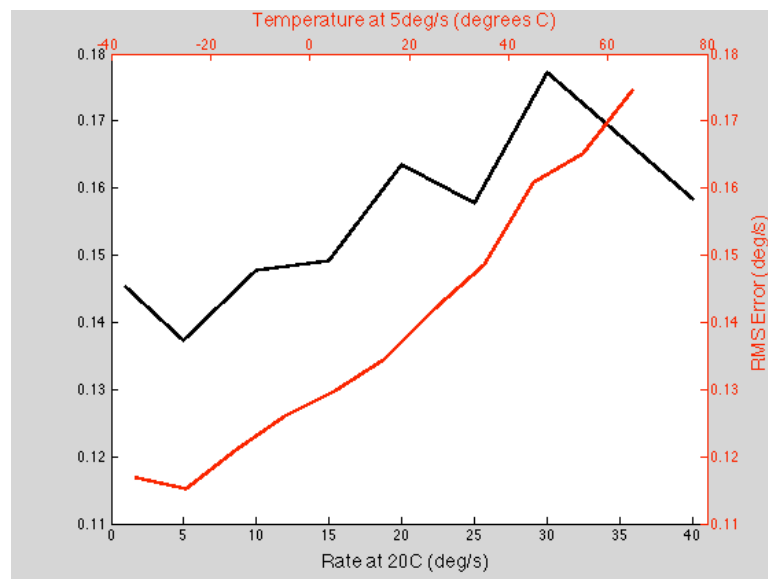


Figure 21: RMS Error Over Rate and Temperature

The nonlinearity of the sensor was found to be 0.54%, less than the 1% requirement. Figure 22. shows the data points (green), average reading (black) and best-fit curve (red). The greatest difference between the red and black curves determines the nonlinearity.

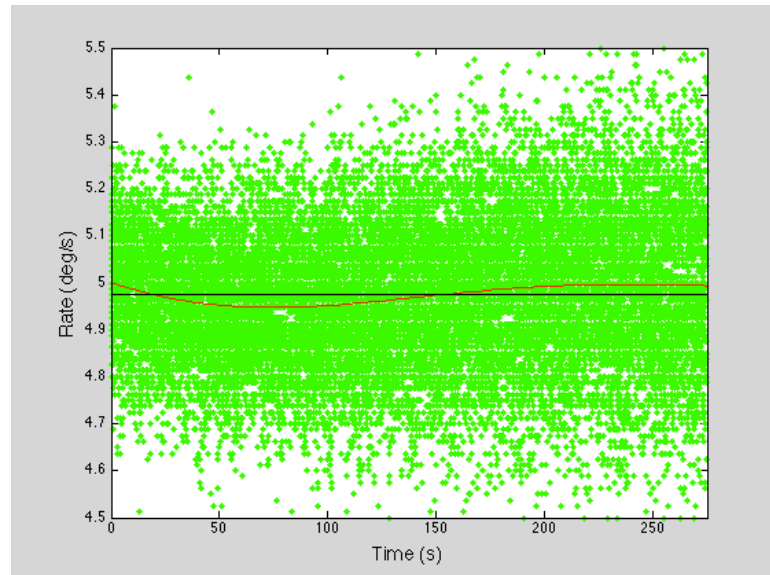


Figure 22: Static Nonlinearity

The angular rate bias for our system was found to be 8.2 degrees per hour from testing, much less than the requirement of 490 degrees per hour.

10 CLOSURE

10.1 Summary:

The goal to develop a system that will accurately measure the angular rate of change between the Birdie and the GEDA. There were two different design concepts that were worked simultaneously and both had different difficulties. Both designs were required to function under the temperature requirements and the rates in which are highlighted in our functional and performance requirements.

10.2 Challenges:

Some of the challenges that we encountered during the design and development of our project dealt with balancing functional requirements with environmental requirements with utilization requirements. Trying to find commercial off the shelf products that met both the functional and environmental requirements was the most difficult. The temperature requirement of -55 to 85 degrees Celsius was one of our most difficult ones to meet because most of the commercial off the shelf products are not specified for those temperatures, let alone all of the military standards.

10.3 Improvements:

Overall, because the system met and exceeded all requirements, it is difficult to note what improvements could have been made to the overall design. One major improvement that can be considered, however, is potentially the failure of the ability to also implement a working mitigation plan. The LSIS system (see Appendix C) was never fully created and tested due to technical difficulties. With more time and resources, this plan may have been realized as well.

11 APPENDIX A GYROSCOPE SOFTWARE

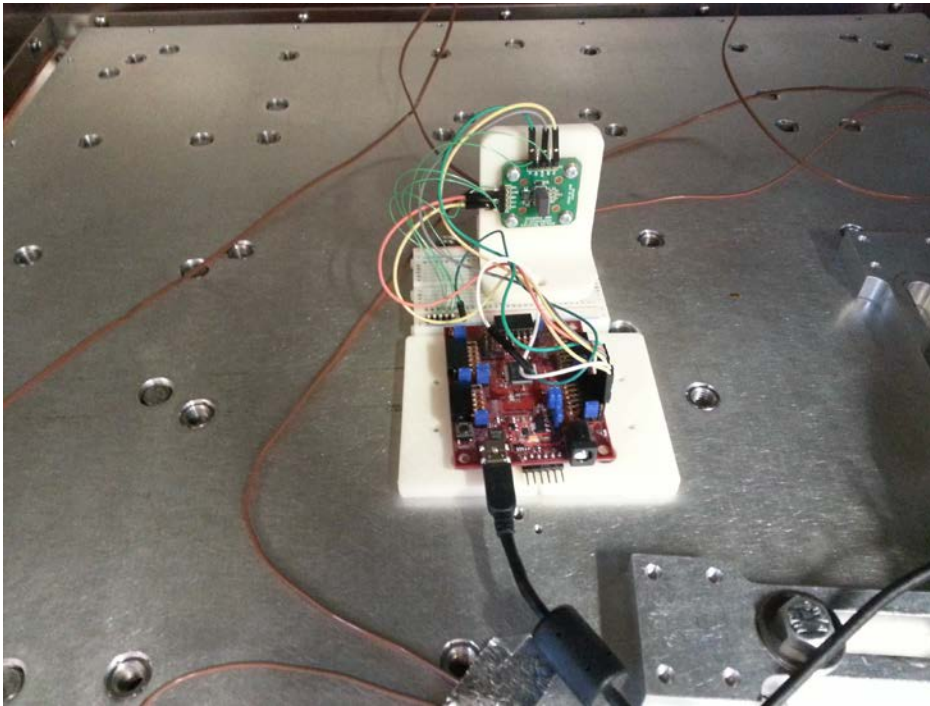


Figure 23: Gyroscope and Cerebot Complete System

11.1 Overview:

This section covers the implementation of the electronic gyroscope design. In this section the implementation of programming the gyroscopes in Serial Peripheral Interface (SPI). The software component of the gyroscope was written in C and utilizing PIC 32. The SPI was programmed so that the microprocessor behaved as the master with the gyroscopes behaving as the selected slaves. The data frame is established by the microprocessor and the gyroscopes perform record the rate of change as the gyroscopes move.

11.2 Written Software: Main.C

The following is the Main of the software component in which the gyroscopes and the microprocessor are programmed.

```
#include "Communication.h"  
#include "ADXRS453.h"  
#include "Delays.h"  
#include "math.h"  
  
#pragma config POSCMOD = XT    /*!< XT oscillator mode selected */  
#pragma config FNOSC = PRIPLL /*!< Primary Oscillator with PLL Module*/  
#pragma config FPLLMUL = MUL_20 /*!< 20x multiplier */  
#pragma config FPLLIDIV = DIV_2 /*!< 2x divider */
```

```
#pragma config FPLLLODIV = DIV_1    /*!< PLL output divided by 1 */
#pragma config FPBDIV   = DIV_1    /*!< PBCLK is SYSCLK divided by 1 */
#pragma config FWDTEN   = OFF      /*!< The WDT is not enabled */

/*Available Commands */
const char* commandsList[] = {"help?",
                              "temperature?",
                              "start="};
const char* commandsDescription[] = {
    " - Displays all available commands.",
    " - Displays the ambient device temperature.",
    " - Starts measurement. Accepted value: 1."};

unsigned char commandsNumber = (sizeof(commandsList) / sizeof(const char*));
unsigned char receivedCommand[20];
unsigned char invalidCommand = 0;
unsigned char commandType = 0;
unsigned char command = 0;
unsigned char displayCommand = 0;
double      commandParam = 0;

void FloatToString(char * buf, double val)
{
    long intPart = 0;
    short fracPart = 0;
    char charPos = 0;
    char localBuf[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
    char i = sizeof(localBuf) - 1;

    intPart = (long)val;
    fracPart = (short)((val - intPart) * 1000 + 0.5);
    while(i > (sizeof(localBuf) - 4))
    {
        localBuf[i] = (fracPart % 10) + 0x30;
        fracPart /= 10;
        i--;
    }
    localBuf[i] = '.';
    if(intPart == 0)
    {
        i--;
        localBuf[i] = '0';
    }
    while(intPart)
    {
        i--;
        localBuf[i] = (intPart % 10) + 0x30;
        intPart /= 10;
    }
    for(charPos = i; charPos < sizeof(localBuf); charPos++)
    {
        *buf = localBuf[charPos];
        buf++;
    }
}
```

```
*buf = 0;
}

void main(void)
{
    /*! Variables holding information about the device */
    unsigned char temperature = 0; /*!< Last read value. */
    unsigned long rxData = 0; /*!< Angular velocity(raw data)*/
    double angularVelocity = 0; /*!< Angular velocity(degrees/second) */
    /*! Temporary variables */
    unsigned char tempString[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
    unsigned long samplesNr = 0; /*!< Number of samples read from the device */
    unsigned char gyroOn = 0; /*!< Status of measurement process */

    /*! Initialize the UART communication peripheral. */
    UART_Init(9600);
    /*! Initialize the ADXRS453 device. */
    if(ADXRS453_Init())
    {
        CONSOLE_WriteString("ADXRS453 OK");
        UART_Write(0x0D);
    }
    else
    {
        CONSOLE_WriteString("ADXRS453 Error");
        UART_Write(0x0D);
    }
    while(1)
    {
        CONSOLE_GetCommand(receivedCommand);
        invalidCommand = 0;
        for(command = 0; command < commandsNumber; command++)
        {
            commandType = CONSOLE_CheckCommands(receivedCommand,
                                                commandsList[command],
                                                (double*)&commandParam);
            if(commandType == 0)
            {
                invalidCommand++;
            }
            if((command == 0) && (commandType != 0)) /*!< "help?" command*/
            {
                CONSOLE_WriteString("Available commands:");
                UART_Write(0x0D);
                for(displayCommand = 0; displayCommand < commandsNumber;
                    displayCommand++)
                {
                    CONSOLE_WriteString(commandsList[displayCommand]);
                    CONSOLE_WriteString(commandsDescription[displayCommand]);
                    UART_Write(0x0D);
                }
            }
        }
        if((command == 1) && (commandType != 0)) /*!< "temperature?" command*/
        {
            /*! Read temperature from the device */

```

```
temperature = ADXRS453_GetTemperature();
/*! Send the requested value to user */
CONSOLE_WriteString("temperature=");
itoa(tempString, temperature, 10);
CONSOLE_WriteString(tempString);
CONSOLE_WriteString(" degrees Celsius");
UART_Write(0xD);
}
if((command == 2) && (commandType != 0)) /*!< "start=" command*/
{
    if(commandParam == 1)
    {
        gyroOn = 1;
    }
    /*! Send feedback to user */
    CONSOLE_WriteString("start=");
    itoa(tempString, (unsigned long)commandParam, 10);
    CONSOLE_WriteString(tempString);
    UART_Write(0x0D);
}
if(gyroOn == 1)
{
    samplesNr = 10000;
    while(samplesNr)
    {
        rxData = ADXRS453_Data(); /*!< read data twice because next SPI word will be temp */
        DelayMs(1);
        rxData = ADXRS453_Data();
        // If data received is in positive degree range
        if(rxData < 0x8000)
        {
            UART_Write('+');
            angularVelocity = floorf(rxData / 80.0);
        }
        // If data received is in negative degree range
        else
        {
            UART_Write('-');
            angularVelocity = floorf((0xFFFF - rxData + 1) / 80.0);
        }
        /*! Send the requested value to user */
        FloatToString(tempString, angularVelocity);
        CONSOLE_WriteString(tempString);
        CONSOLE_WriteString(" deg/s\n");
        UART_Write(0xD);
        DelayMs(100);
        samplesNr --;
    }
    gyroOn = 0;
    /*! Measurement process has finished */
    CONSOLE_WriteString("start=0");
    UART_Write(0xD);
}
if(invalidCommand == commandsNumber)
{
```

```

    /*! Send feedback to user */
    CONSOLE_WriteString("Invalid command");
    UART_Write(0x0D);
}
}
}

```

11.3 ADXRS453.h

```

#ifndef __ADXRS453_H__
#define __ADXRS453_H__

/*****
***** Macros and Constants Definitions *****/
/*****

#define ADXRS453_STARTUP_DELAY 0.05 /*!< sec */

/*! The MSB for the spi commands */
#define ADXRS453_SENSOR_DATA 0x20
#define ADXRS453_WRITE_DATA 0x40
#define ADXRS453_READ_DATA 0x80
/*! Memory register map */
#define ADXRS453_RATE1 0x00 /*!< Rate Registers */
#define ADXRS453_TEMP1 0x02 /*!< Temperature Registers */
#define ADXRS453_LO CST1 0x04 /*!< Low CST Memory Registers */
#define ADXRS453_HI CST1 0x06 /*!< High CST Memory Registers */
#define ADXRS453_QUAD1 0x08 /*!< Quad Memory Registers */
#define ADXRS453_FAULT1 0x0A /*!< Fault Registers */
#define ADXRS453_PID1 0x0C /*!< Part ID Register 1 */
#define ADXRS453_SNH 0x0E /*!< Serial Number Registers, 4 bytes */
#define ADXRS453_SNL 0x10

/*! Check bits */
#define ADXRS453_P 0x01 /*!< Parity bit */
#define ADXRS453_CHK 0x02
#define ADXRS453_CST 0x04
#define ADXRS453_PWR 0x08
#define ADXRS453_POR 0x10
#define ADXRS453_NVM 0x20
#define ADXRS453_Q 0x40
#define ADXRS453_PLL 0x80
#define ADXRS453_UV 0x100
#define ADXRS453_OV 0x200
#define ADXRS453_AMP 0x400
#define ADXRS453_FAIL 0x800

#define ADXRS453_WRERR_MASK (0x7 << 29)
#define ADXRS453_GET_ST(a) ((a >> 26) & 0x3) /*!< Status bits */

unsigned char ParityBit(unsigned long data);
/*! Initializes the ADXRS453 and checks if the device is present. */
unsigned char ADXRS453_Init(void);
/*! Reads data from ADXRS453. */

```

```
unsigned long   ADXRS453_Data(void);  
/*! Reads register from ADXRS453. */  
unsigned short ADXRS453_GetRegisterValue(unsigned char regAddress);  
/*! Writes register to ADXRS453. */  
void ADXRS453_SetRegisterValue(unsigned char regAddress, unsigned short regData);  
/*! Reads temperature from ADXRS453 and converts it to degrees Celsius. */  
unsigned char ADXRS453_GetTemperature(void);  
  
#endif /* __ADXRS453_H__ */
```

11.4 Communication.h

```
#ifndef __COMMUNICATION_H__  
#define __COMMUNICATION_H__  
#include <xc.h>  
  
#define PMOD1_CS_PIN    (1 << 9)  
#define PMOD1_CS_PIN_OUT TRISG &= ~(PMOD1_CS_PIN);  
#define PMOD1_CS_LOW    PORTG &= ~(PMOD1_CS_PIN);  
#define PMOD1_CS_HIGH   PORTG |= PMOD1_CS_PIN;  
  
unsigned char SPI_Init(unsigned char lsbFirst,  
                       unsigned long clockFreq,  
                       unsigned char clockPol,  
                       unsigned char clockEdg);  
  
/*! Writes data to SPI. */  
unsigned char SPI_Write(unsigned char slaveDeviceId,  
                        unsigned char* data,  
                        unsigned char bytesNumber);  
  
/*! Reads data from SPI. */  
unsigned char SPI_Read(unsigned char slaveDeviceId,  
                       unsigned char* data,  
                       unsigned char bytesNumber);  
  
/*!Initializes the I2C communication peripheral. */  
unsigned char I2C_Init(unsigned long clockFreq);  
  
/*! Writes data to a slave device. */  
unsigned char I2C_Write(unsigned char slaveAddress,  
                        unsigned char* dataBuffer,  
                        unsigned char bytesNumber,  
                        unsigned char stopBit);  
  
/*! Reads data from a slave device. */  
unsigned char I2C_Read(unsigned char slaveAddress,  
                       unsigned char* dataBuffer,  
                       unsigned char bytesNumber,  
                       unsigned char stopBit);  
  
#endif /* __COMMUNICATION_H__ */
```

11.5 Console.C

```
#include "Console.h"

unsigned char UART_Init(unsigned long baudRate)
{
    unsigned long pbFrequency = 8000000;
    unsigned short brgValue = 0;

    /* BaudRate = Fpb / (16 * (UxBRG + 1)) */
    brgValue = pbFrequency / (16 * baudRate) - 1;
    U1MODE = 0; // Clear the content of U1MODE register
    U1BRG = brgValue;
    U1MODEbits.PDSEL1 = 0;
    U1MODEbits.PDSEL0 = 0; // 8-bit data, no parity
    U1MODEbits.STSEL = 0; // 1 Stop bit
    U1STAbits.URXEN = 1; // Enable UART1 receiver
    U1STAbits.UTXEN = 1; // Enable UART1 transmitter
    U1MODEbits.ON = 1; // Enable UART1 peripheral
}

void UART_Write(unsigned char data)
{
    while(U1STAbits.TRMT == 0);
    U1TXREG = data;
}

unsigned char UART_Read(void)
{
    unsigned char receivedChar = 0;

    while(U1STAbits.URXDA == 0);
    receivedChar = U1RXREG;

    return receivedChar;
}

void CONSOLE_WriteString(const char* string)
{
    while(*string)
    {
        UART_Write(*string++);
    }
}

void CONSOLE_GetCommand(unsigned char* command)
{
    unsigned char receivedChar = 0;
    unsigned char charNumber = 0;

    while(receivedChar != 0x0D)
    {
        receivedChar = UART_Read();
    }
}
```

```
    command[charNumber++] = receivedChar;
}
}

unsigned char CONSOLE_CheckCommands(unsigned char* receivedCommand,
    const char* expectedCommand,
    double* commandParameter)
{
    unsigned char commandType = 1;
    unsigned char charIndex = 0;

    unsigned char parameterString[5] = {0, 0, 0, 0, 0};
    unsigned char parameterIndex = 0;

    while((expectedCommand[charIndex] != '?') &
        (expectedCommand[charIndex] != '=') &
        (commandType != 0))
    {
        if(expectedCommand[charIndex] != receivedCommand[charIndex])
        {
            commandType = 0;
        }
        charIndex++;
    }
    if(commandType != 0)
    {
        if(expectedCommand[charIndex] == '=')
        {
            if(receivedCommand[charIndex] == '=')
            {
                charIndex++;
                while(receivedCommand[charIndex] != 0x0D)
                {
                    parameterString[parameterIndex] = receivedCommand[charIndex];
                    charIndex++;
                    parameterIndex++;
                }
                *commandParameter = atof(parameterString);
            }
            else
            {
                commandType = 0;
            }
        }
        if(expectedCommand[charIndex] == '?')
        {
            if(receivedCommand[charIndex] == '?')
            {
                commandType = 2;
            }
            else
            {
                commandType = 0;
            }
        }
    }
}
```



```
    }  
  
    return commandType;  
}
```

11.6 Console.h

```
#ifndef __CONSOLE_H__  
#define __CONSOLE_H__  
  
#include <xc.h>  
#include <stdlib.h>  
  
unsigned char UART_Init(unsigned long baudRate);  
  
/*! Writes one character to UART. */  
void UART_Write(unsigned char data);  
  
/*! Reads one character from UART. */  
unsigned char UART_Read(void);  
  
/*! Writes one string to UART. */  
void CONSOLE_WriteString(const char* string);  
  
/*! Reads one command from UART. */  
void CONSOLE_GetCommand(unsigned char* command);  
  
/*! Compares two commands and returns the type of the command. */  
unsigned char CONSOLE_CheckCommands(unsigned char* receivedCommand,  
                                     const char* expectedCommand,  
                                     double* commandParameter);  
  
#endif /*__CONSOLE_H__*/
```

11.7 Delays.C

```
#include "Delays.h"  
  
void DelayMs(unsigned short milliseconds)  
{  
    unsigned long ticks = 0;  
  
    _CP0_SET_COUNT(0);  
    while(ticks < TICKS_FOR_1MS * milliseconds)  
    {  
        ticks = _CP0_GET_COUNT();  
    }  
}
```

11.8 Delays.h

```
#ifndef __DELAYS_H__
#define __DELAYS_H__

#include <xc.h>

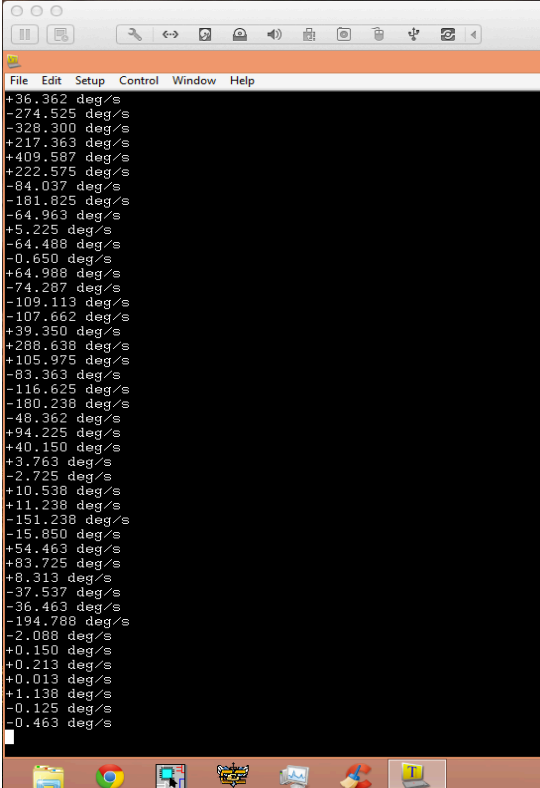
#define SYSTEM_CLOCK      80000000 /*!< System clock frequency is 80MHz */
#define CORE_TIMER_CLOCK  SYSTEM_CLOCK / 2.0 /*!< Core timer runs at 40 MHz */
#define MILLIS_IN_A_SECOND 1000 /*! Number of milliseconds in a second */
static const unsigned long TICKS_FOR_1MS = CORE_TIMER_CLOCK /
MILLIS_IN_A_SECOND;

/*****
***** Functions Declarations *****/
/*****
/*! Creates a delay of milliseconds. */
void DelayMs(unsigned short milliseconds);

#endif /* __DELAYS_H__ */
```

11.9 Working Code

The main function has permitted the gyroscope to record the angular rate of the gyroscopes. An example of this working code is shown below in Figure 24.



```
File Edit Setup Control Window Help
+36.362 deg/s
-274.525 deg/s
-328.300 deg/s
+217.363 deg/s
+409.587 deg/s
+222.575 deg/s
-84.037 deg/s
-181.825 deg/s
-64.963 deg/s
+5.225 deg/s
-64.488 deg/s
-0.650 deg/s
+64.988 deg/s
+74.287 deg/s
-109.113 deg/s
-107.662 deg/s
+39.350 deg/s
+288.638 deg/s
+105.975 deg/s
-83.363 deg/s
-116.625 deg/s
-180.238 deg/s
-48.362 deg/s
+94.225 deg/s
+40.150 deg/s
+3.763 deg/s
-2.725 deg/s
+10.538 deg/s
+11.238 deg/s
-151.238 deg/s
-15.850 deg/s
+54.463 deg/s
+83.725 deg/s
+8.313 deg/s
-37.537 deg/s
-36.463 deg/s
-194.788 deg/s
-2.088 deg/s
+0.150 deg/s
+0.213 deg/s
+0.013 deg/s
+1.138 deg/s
-0.125 deg/s
-0.463 deg/s
```

Figure 24: Example of Fully Functioning Code

12 APPENDIX B: MECHANICAL DRAWINGS

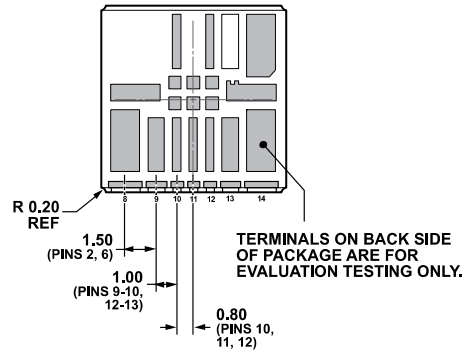


Figure 25: ADXRS453 Mechanical Drawing

EVAL-ADXRS453Z-V BOARD LAYOUT

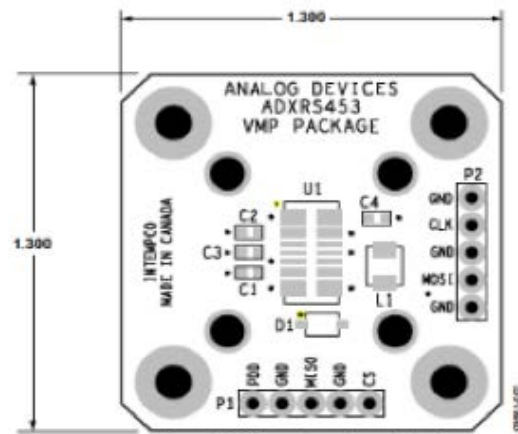


Figure 26: ADXRS453 Evaluation Board Mechanical Drawing

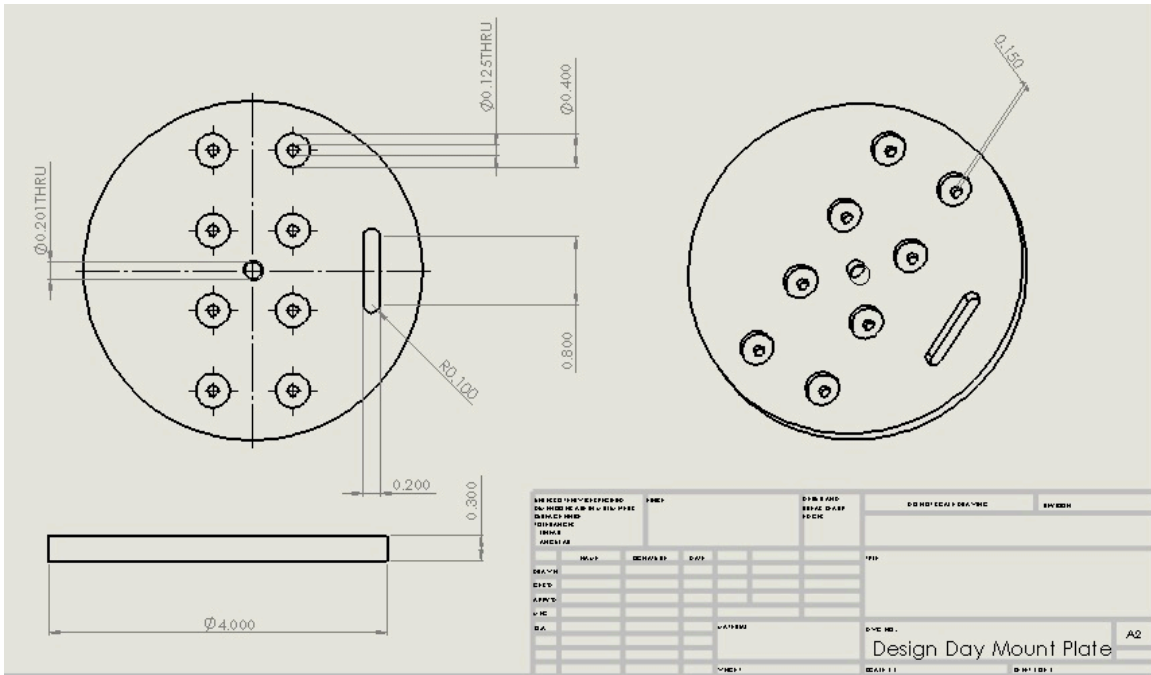


Figure 27: Mounting Plate for Display Mechanical Drawing

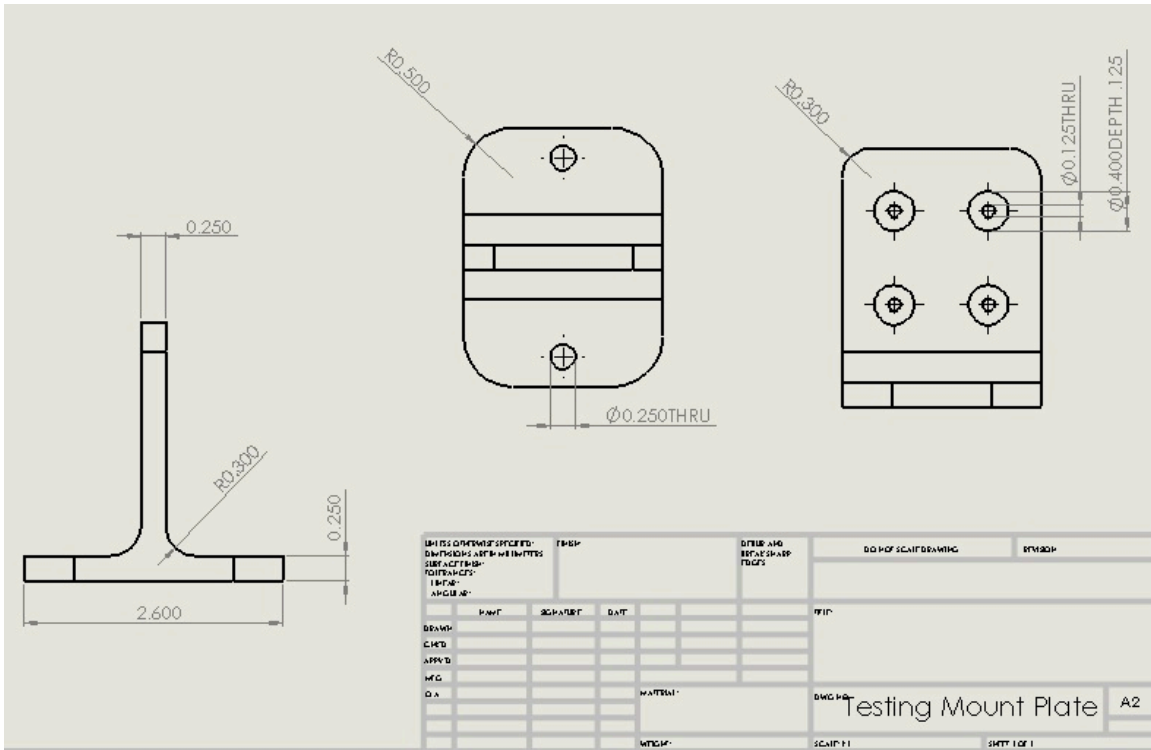


Figure 28: Mounting Plate for Testing Mechanical Drawing

13 APPENDIX C: REPORT FOR THE LASER SPOT IMAGING SYSTEM

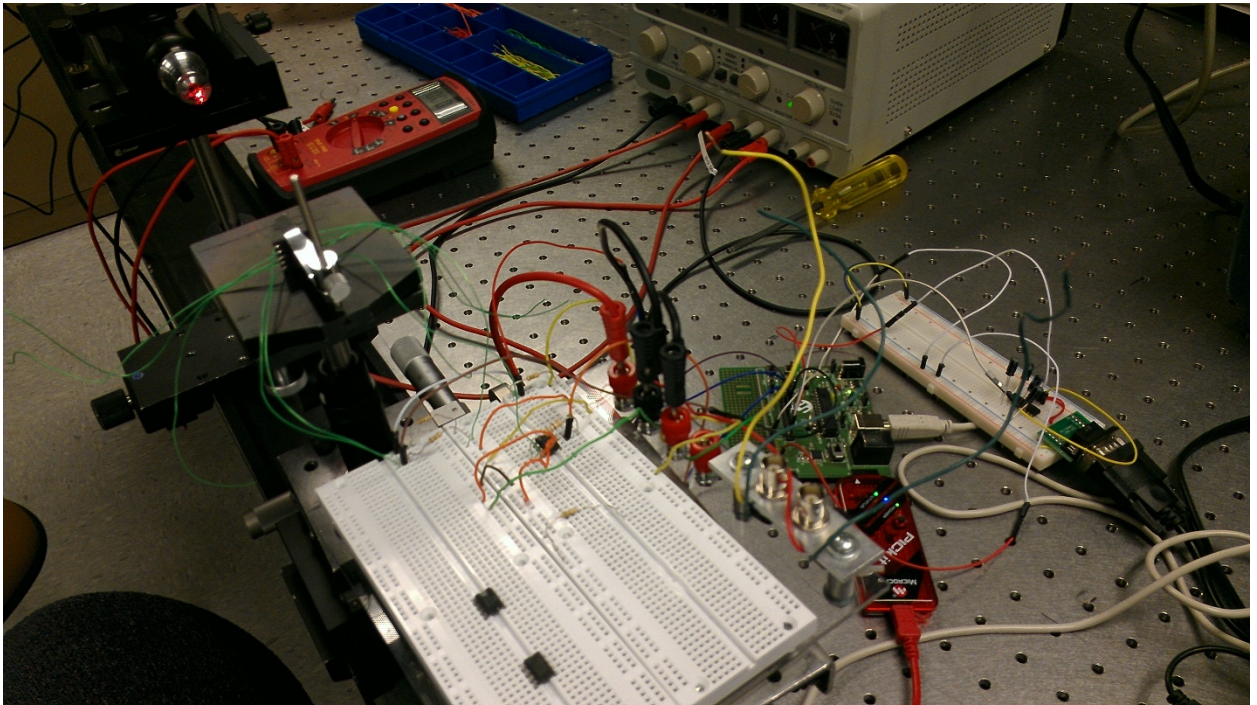


Figure 29: LSIS Design Creation and Laboratory Testing

1 OVERVIEW:

This section covers the alternative LSIS design (Laser Spot Imaging System) which was an optical approach to measuring angles in an exposed two axis gimbal assembly. It was very simple and involved only two primary components: A PSD (Position Sensitive Detector) which would be placed on one of the two moving bodies and a laser source to drive the PSD connected to the other body. The resulting output from the PSD would be captured and evaluated by means of a microprocessor and then transmitted to a computer screen so that the resulting output data could be viewed. While the LSIS concept seemed to work on paper, when implementing the design in the lab, the system itself was poorly configured due to a lack of resources and technical data. After numerous failed attempts at getting the system to work in a controlled lab environment, the idea was finally dismissed.

2 TOP-LEVEL DESIGN OF FINAL ALTERNATIVE DESIGN CONCEPT

2.1 Laser Spot Imaging System (LSIS)

A secondary design for the project takes a different approach. The Laser Spot Imaging System (LSIS) utilizes optical components to measure absolute angles instead of angular rates. This absolute method produces multiple angles at various points in time and the angular rate of change between the two bodies may then be found by differentiating the positions with respect to time. The high level organization of this system is shown below in Figure 30.

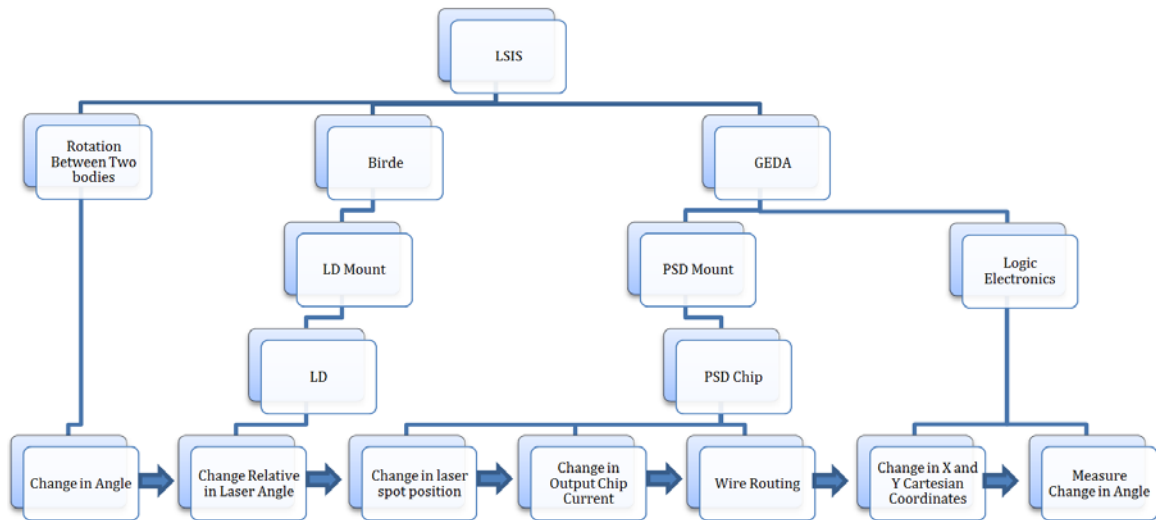


Figure 30: High Level Block Diagram of the LSIS

To get a more visual understanding of the system and its layout, drawings that illustrate this version of the design are included in Figure 3132, Figure 3233, and Figure 33. The four major components that comprise the system have been colored so that they are easily identifiable. The two metal mounts (designed just for an intuitive understanding of how the system would be mounted) for the Position Sensitive Detector (PSD) and the Laser Diode (LD) are shown in black while the PSD is shown in blue and the LD in red.

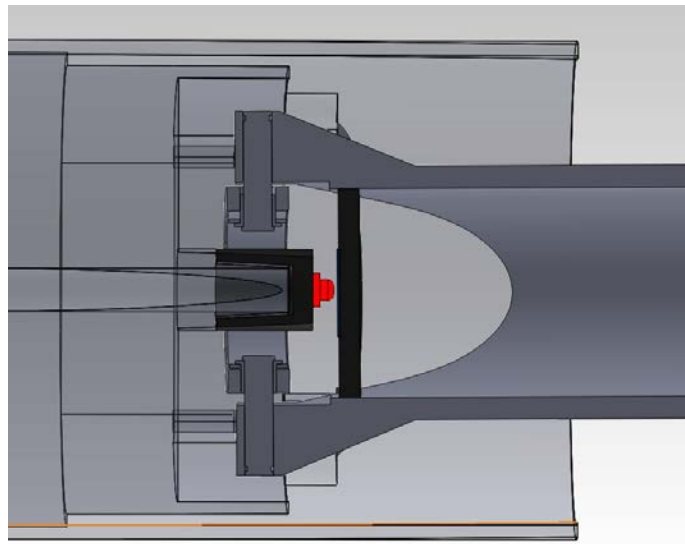


Figure 31: Side Horizontal View of LSIS Assembly

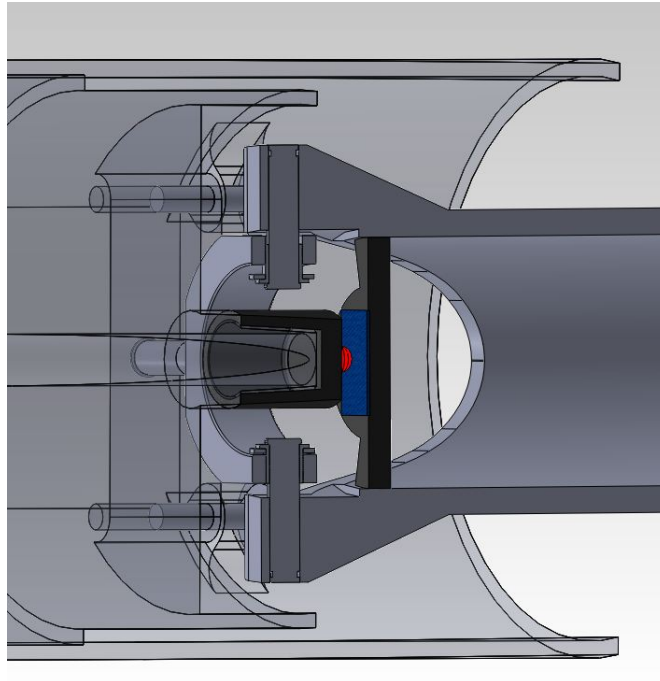


Figure 32: Isotropic View of LSIS Assembly

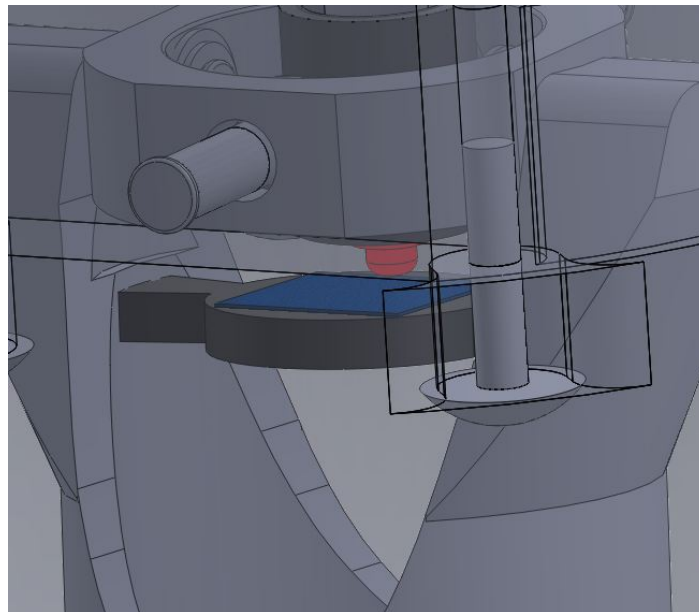


Figure 33: Side Vertical View of LSIS Assembly

3 HARDWARE SUBSYSTEM/SUB-ASSEMBLY AND INTERFACE DESIGN

3.1 Position Sensitive Detector

The detector for this particular design drives nearly all of the requirements for the project and needed to be chosen carefully. The Position Sensitive Detectors (PSD) sold by Hamamatsu Corporation come very close to fulfilling this great demand. The S5990-01 and S5991-01 are the two most appropriate PSDs for detecting an optical intensity and thus determining the angle of rotation between the two bodies concerned. The two detectors are shown below courtesy of Hamamatsu Corporation in Figure 34.

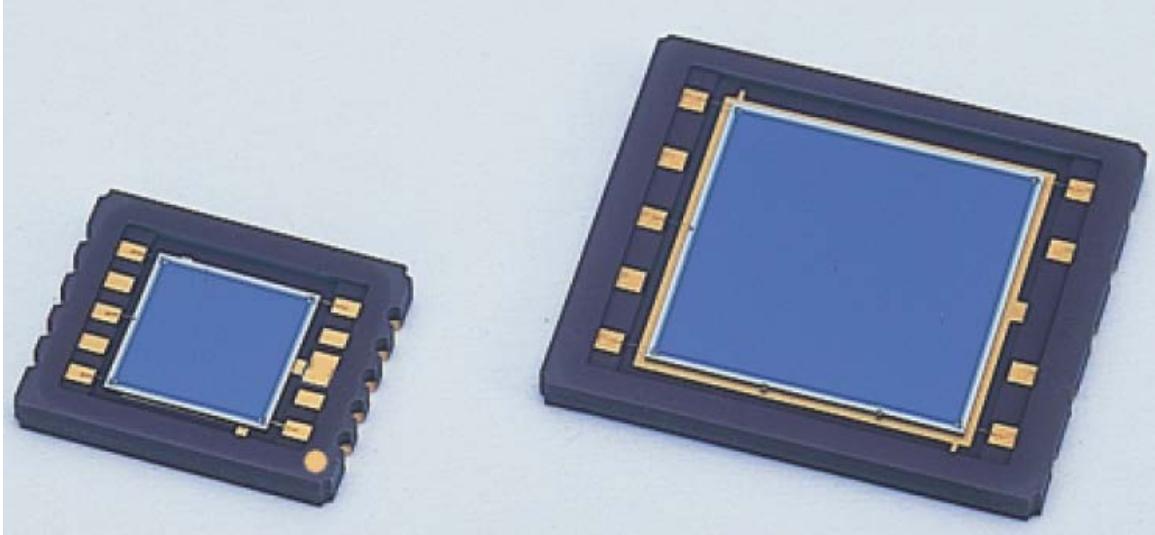


Figure 34: S5990-01 and S5991-01 PSDs (from left to right)

A few differences between the two detectors are their physical size but also spot detection resolution as well as position detection error. On top of that, the larger S5991-01 is \$76 per unit while the smaller S5990-01 is only \$30 per unit. Although the larger PSD (S5991-01) is more expensive, it has an effectively smaller resolution and fits the project's physical restraints with a bit more finesse than with the smaller detector. Otherwise, virtually all other specifications on this detector are the same for the S5990-01 as well.

3.2 S5991-01 Position Sensitive Detector Specifics

The larger PSD was the primary choice for the detector element in the LSIS design so the specifications for the device are included below.

3.2.1 Dimensional Diagrams

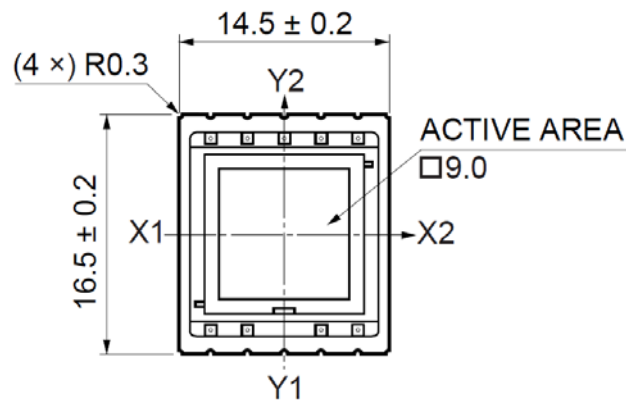


Figure 35: S5991-01 Front View

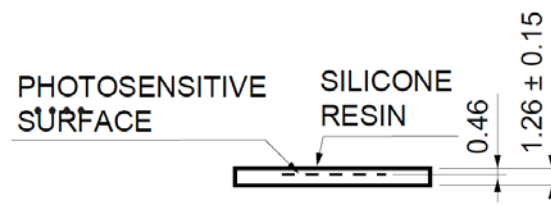


Figure 36: S5991-01 Top View

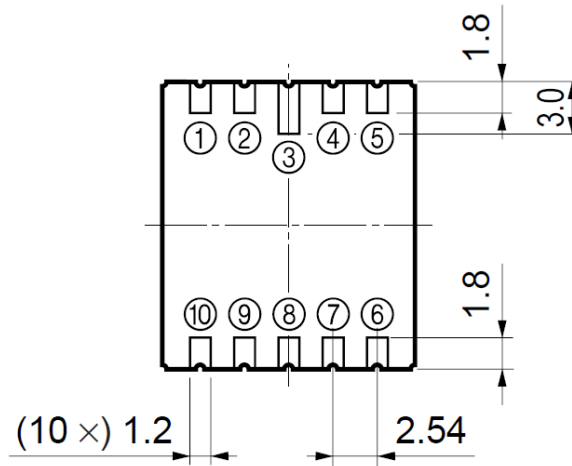


Figure 37: S5991-01 Rear View

Note: Burrs shall protrude no more than 0.3mm on any side of the package.

3.2.2 Specifications

Table 8: S5991-01 I/O

Pin Number	Description
1	Anode X1 (I_1)
2	NC
3	NC (pin should be open-circuited)
4	NC
5	Anode Y1 (I_3)
6	Anode X2 (I_2)
7	NC
8	Cathode
9	NC
10	Anode Y2 (I_4)

Table 9: S5991-01 Maximum Ratings and Electrical/Optical Characteristics

Parameter	Symbol	Condition	Min	Typical	Max	Unit
Reverse Voltage	V_R Max.			20		V
Operating Temperature	Topr		-20		60	°C

Storage Temperature	T _{stg}		-20		80	°C
Spectral Response Range	λ		320		110	nm
Peak Sensitivity Wavelength	λ_p			960		nm
Photo Sensitivity	S	$\lambda = \lambda_p$		0.6		A/W
Interelectrode Resistance	R _{ie}	V _b =0.1V	5	7	15	k Ω
Position Detection Error	E	$\lambda=900\text{nm}$ V _R =5V SLS: $\phi 0.2\text{mm}^*$		± 150	± 250	μm
Saturation Photocurrent	I _{st}	$\lambda=900\text{nm}$ V _R =5V R _L =1k Ω		500		μA
Dark Current	I _D	V _R =5V		1	50	nA
Rise Time	t _r	$\lambda=900\text{nm}$ V _R =5V R _L =1k Ω		2		μs
Thermal Capacitance	C _t	V _R =5V F=10kHz		500	1000	pF
Position Resolution	ΔR	I _o =1 μA B=1kHz		1.5		μm

*SLS (Spot Light Size) in the range that is 80% from the center to the edge. Recommended spot light size is larger than 0.2mm.

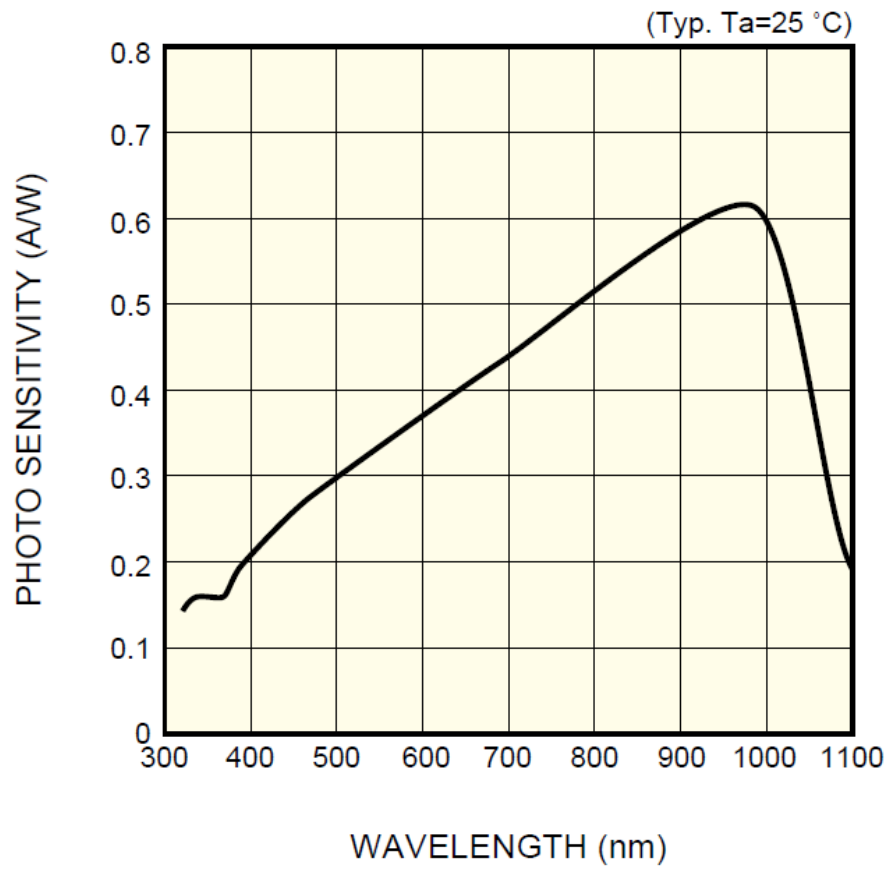


Figure 38: S5991-01 Spectral Response

3.2.3 Electrical Properties

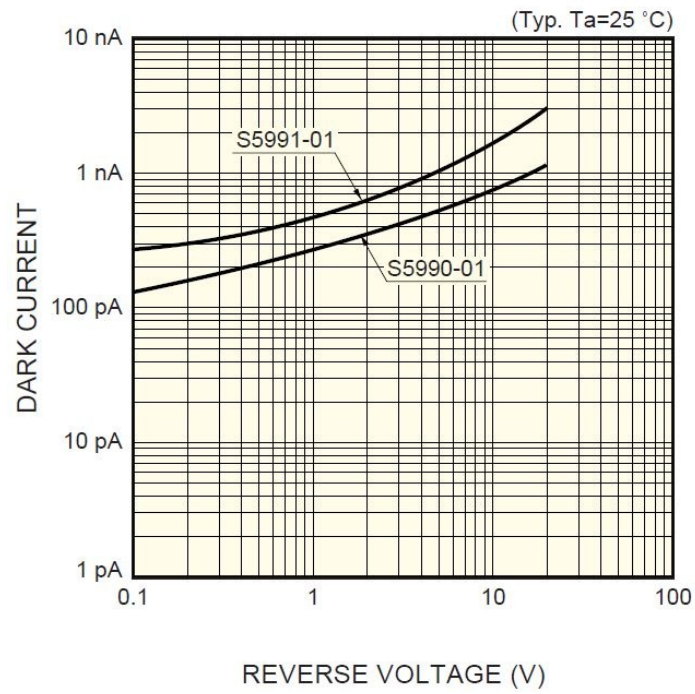


Figure 39: S5991-01 Dark Current vs. Reverse Voltage

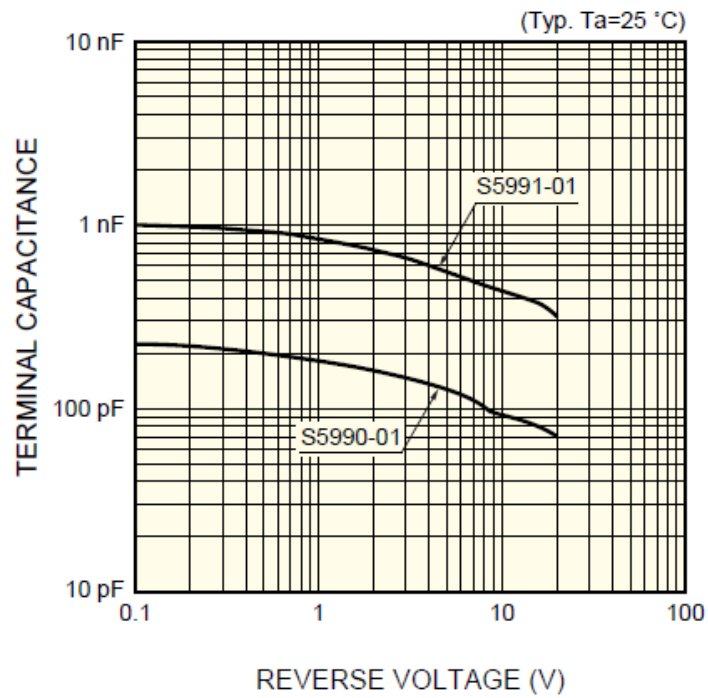
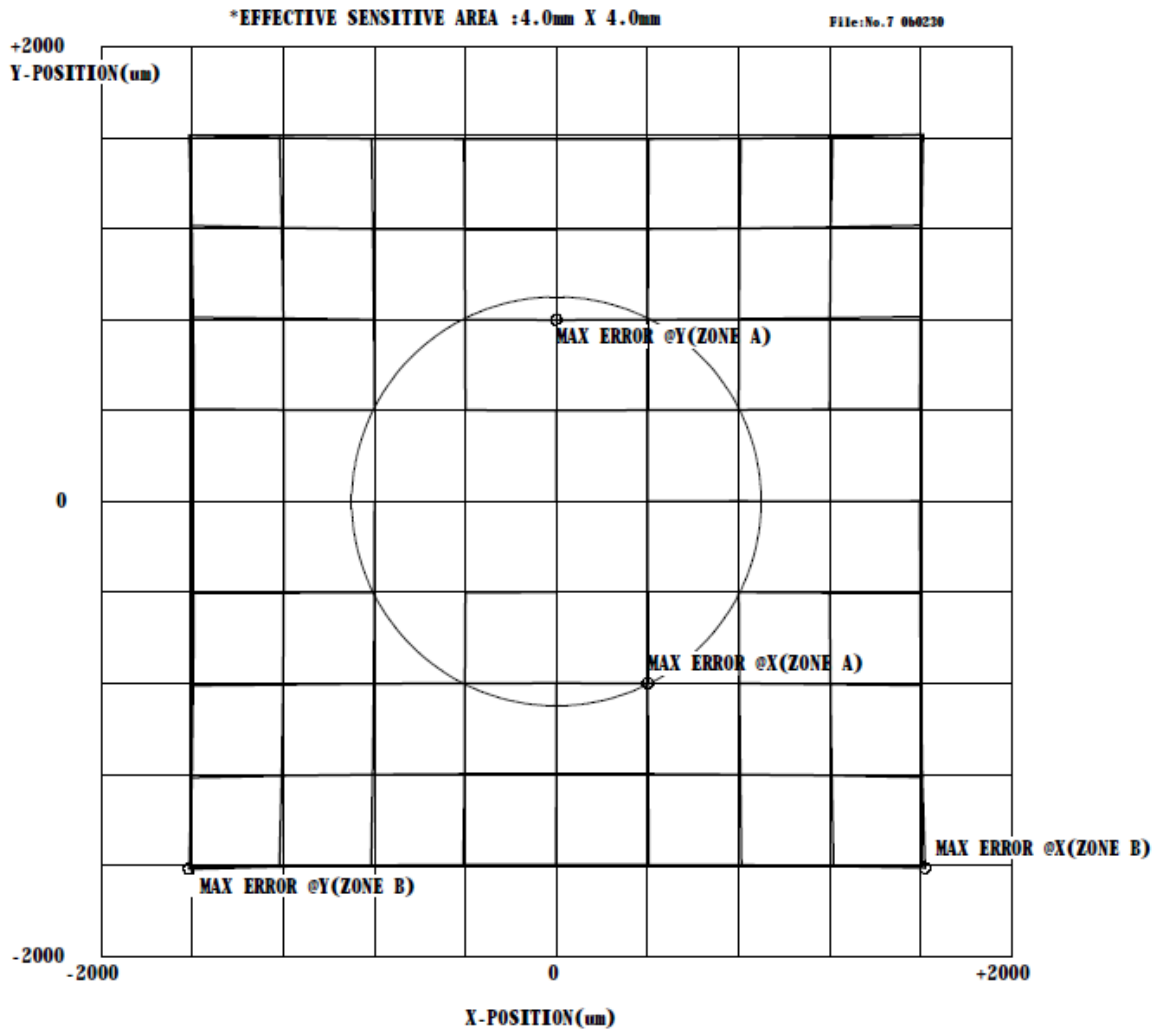


Figure 40: S5991-01 Terminal Capacitance vs. Reverse Voltage

3.2.4 Detector Error and Effective Area

TYPE No.	S5990-01	BIAS VOLTAGE	5 V
SERIAL No.	1	STEP WIDTH	400 μm
TESTED BY	NORIYASU	SPOT SIZE	200 μm
DATE	2010-02-16	WAVELENGTH	900 nm



ZONE A ($r=0.9\text{mm}$) : MAXIMUM ERROR ; -1.90 μm (X) 2.00 μm (Y)
 ZONE B ($S=3.2\text{mm} \times 3.2\text{mm}$) : MAXIMUM ERROR ; -18.80 μm (X) 17.10 μm (Y)
 COEFFICIENT = 1.030

Figure 41: S5991-01 Detector Efficiency Area and Standard Error as per Position

3.2.5 Important Handling Notes

- The light input window of this product uses soft silicone resin. Precautions must be made to avoid touching the window in order to keep it free from grime and damage (either to the surface or to the contacts) as this will decrease sensitivity.
- Use rosin flux when soldering, to prevent the terminal lead corrosion. Reflow oven temperature should be at a maximum temperature of 260°C for no longer than 5 seconds under the conditions that no moisture absorption occurs. Reflow soldering conditions differ depending on the type of PC board and reflow oven. Carefully check these conditions before use.
- Silicone resin swells when it absorbs organic solvent, so do not use any solvent other than alcohol.
- Avoid unpacking until you actually use this product to prevent the terminals from oxidation and dust deposits or the coated resin from absorbing moisture. When the product is stored for 3 months while not unpacked or 24 hours have elapsed after unpacking, perform baking in nitrogen atmosphere at 150°C for 3 to 5 hours or at 120°C for 12 to 15 hours before use.

3.3 Laser Diode

The detector for the LSIS design is the most crucial part and therefore the laser diode (LD) counterpart to the system was matched to the detector's specifications. Ultimately, the 904-5 Infrared Laser Diode, sold by US-Lasers, was chosen as its technical specifications seemed to meet our requirements. The diode is shown below in Figure 42 along with a schematic in Figure 43 and its technical specifications in Table 10.



Figure 42: US-Lasers 904-5 IR Laser Diode

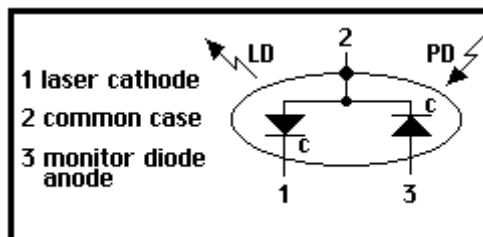


Figure 43: US-Lasers 904-5 LD Schematic

Table 10: US-Lasers 904-5 LD Technical Properties

Items	Symbols	Values	Unit
Optical output power	Po	5	mW
Laser diode reverse voltage	VLDR	2	V
Photo diode reverse voltage	VPDR	30	V
Operating temperature	Topr	-10 ~ +40	°C
Storage temperature	Tstg	-40 ~ +85	°C

OPTICAL and ELECTRICAL CHARACTERISTICS - (Tc=25 °C)

Items	Symbols	Min.	Typ.	Max.	Unit	Test Condition
Optical output power	Po	-	5	-	mW	-
Threshold current	Ith	10	20	35	mA	-
Operating current	Iop	15	25	45	mA	Po=5mW
Operating voltage	Vop	2.0	2.4	2.7	V	Po=5mW
Lasing wavelength	8 D	894	904	914	nm	Po=5mW
Beam divergence	θ F	8	10	11	deg	Po=5mW
Beam divergence	θ z	25	31	40	deg	Po=5mW
Slope Efficiency (mW/mA)	0	0.4	0.5	0.7		-
Monitor current	Im	10	100	200	μ A	Po=5mW, Vr=5V
Astigmatism	As	-	11	-	μ m	Po=5mW
MTTF			3000-5,000 hrs.			Po=5mW, NA=0.4
Emitter Size	1 x 4 Microns					
Emitter Distance to Cap Lens	0.3mm					
Structure	Index Guided					

When testing the laser diode upon receipt, the angular divergence of the diode was extremely high as can be seen below in Figure 44. What this essentially means is that even though the distance between the PSD and the LD is very small, the divergence is high enough to cause the output spot size to be very large. This would not allow for any decent accuracy within the system.

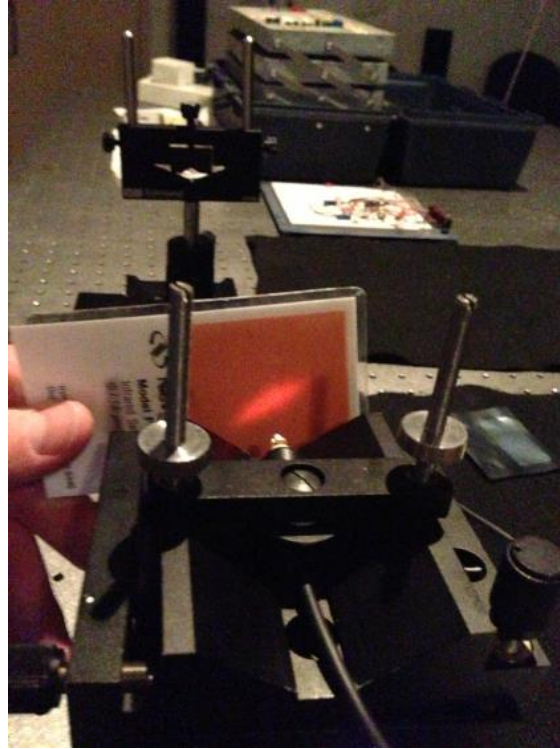


Figure 44: High Divergence Angle from US-Lasers 9904-5 LD.

3.4 System Mounts

The LSIS system is designed to fit in the hollow cavity of the GEDA in the Paveway™ weapons system. In order to set up all components correctly, mounts for both the PSD and the LD need to be manufactured such that they fit within the allocated space and provide for optimal component functionality. The following figure is an example of a generic GEDA mount that would work with the LSIS design and fit within the hollow GEDA tube near its end.

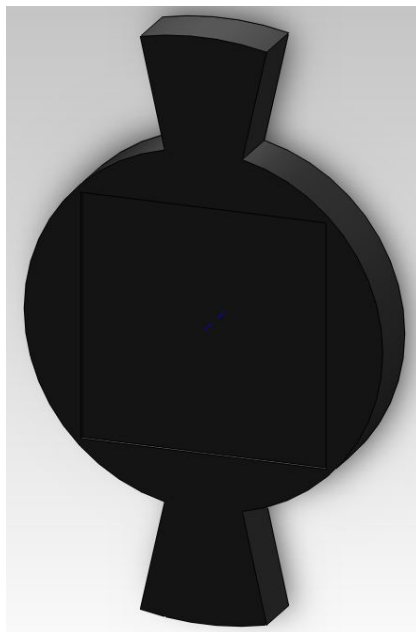


Figure 45: GEDA Mount

The next figure is an example of a generic Birdie mount that would work with the LSIS design. This mount currently fits directly over what is now a metal tube-like structure designed to hold wires coming from the birdie. As already discussed with Raytheon sponsors, the design does not have to take these wires into account (which would normally be included in the entire Paveway™ assembly).

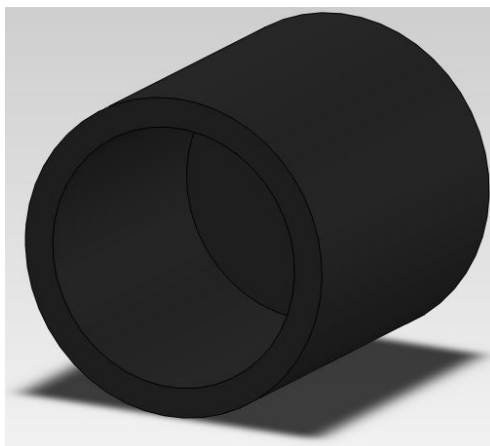


Figure 46: Birdie Mount

3.5 Testing Mounts

In order to adequately test the system, a few mounting devices were thought of. Both ideas were simple, but the first required the completion of working code and functioning hardware. Given the benefit of the doubt, the first testing mount was designed as seen below in Figure 47. It would have been utilized in a fully automatic fashion using a rate table with a precise optical encoder so that results were very accurate.

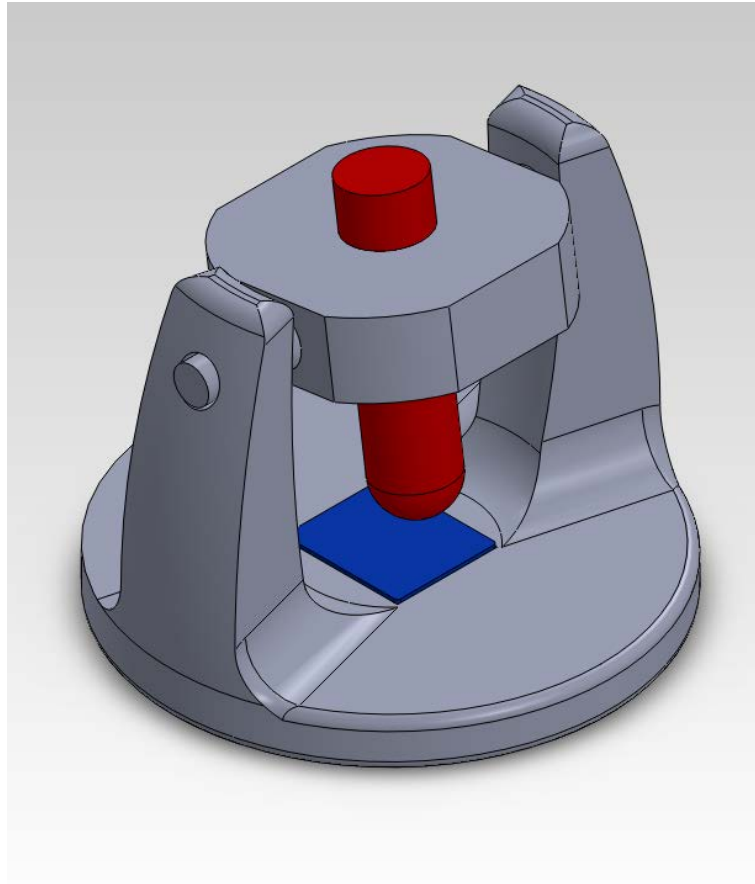


Figure 47: Initial LSIS Testing Mount

When, by the final presentation date, the LSIS design was still encountering problems (undetermined by this point as to whether the problems were being caused by hardware and/or software) the next design, seen below in Figure 48 was created and allowed for a less advanced but more manual way of obtaining test data. The mount allowed for the laser diode to be fixed at specific angles and the result could then be analyzed for various testing scenarios.

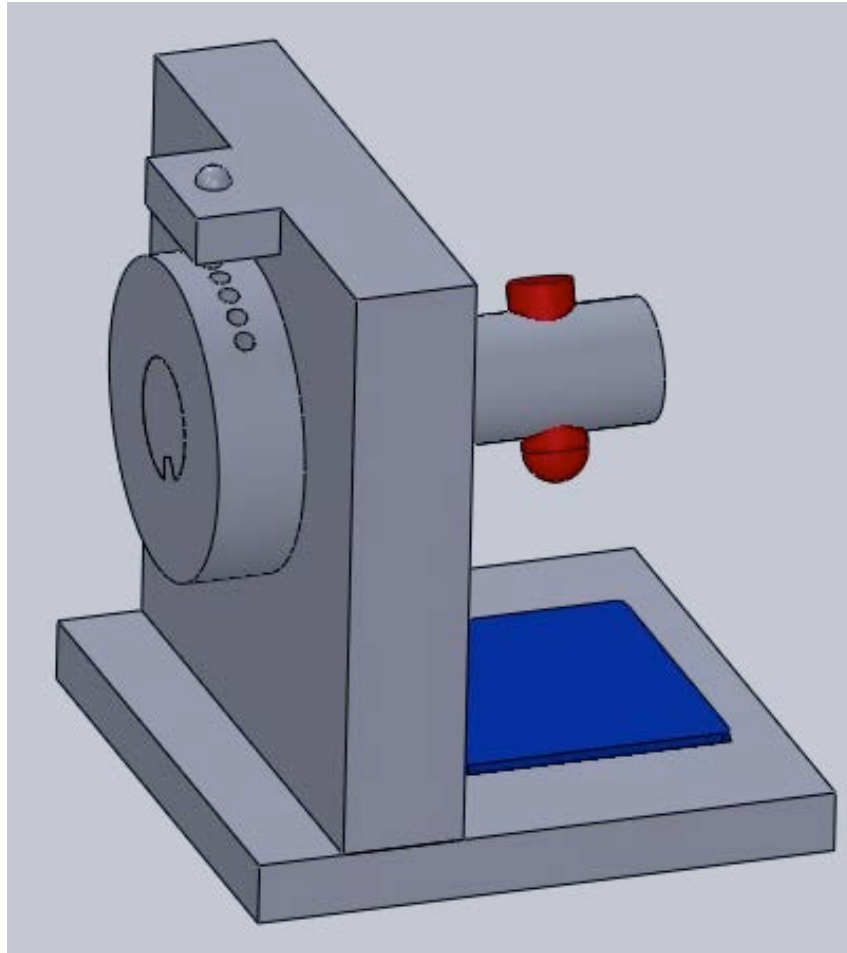


Figure 48: Secondary LSIS Testing Mount

4 SOFTWARE ALGORITHM DESCRIPTION AND INTERFACE DOCUMENT

4.1 LSIS Angular Calculation Engine

4.1.1 Cartesian Coordinates Calculation

The PSD chip outputs four currents depending on the position of the laser spot on the chip. Equations (2) and (3) were provided from Hamamatsu Corporation to determine Cartesian position on the detector:

$$x = \frac{L(I_2 + I_3) - (I_1 + I_4)}{2(I_1 + I_2 + I_3 + I_4)} \quad (2)$$

$$y = \frac{L(I_2 + I_4) - (I_1 + I_3)}{2(I_1 + I_2 + I_3 + I_4)} \quad (3)$$

Using these two equations, the planar x and y coordinates on the detector may be determined.

4.1.2 Cartesian to Spherical Coordinate Conversion

In a static scenario, the laser beam and detector are perpendicular to one another and thus the angle of rotation is trivially zero. However, as the two bodies connected to each component change relative angles, the incident laser beam will not be perfectly perpendicular anymore. Essentially, due to the nature of the rotation and radius of such rotation associated with the laser diode, the detector appears to be spherical in shape when in fact it is actually planar. What this means is that the greater the relative angle is between the two bodies, the “faster” the spot moves across the area of the detector. To accommodate for this, the mathematical relationship between the PSD and the LD needed to be explicit and simplified. The resulting angle of rotation θ and ϕ measured by the system as a function of X and Y respectively is defined in Equations (4) and (5).

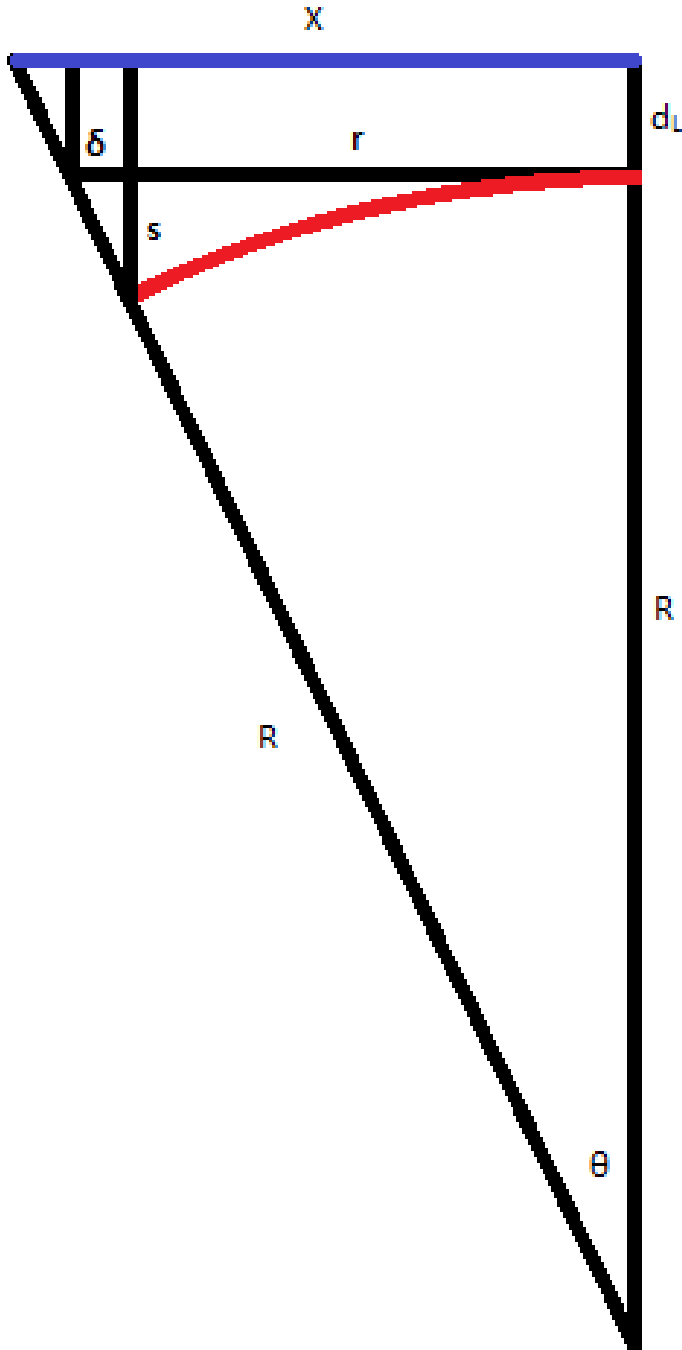
$$\theta = \tan^{-1} \left[\frac{x}{d_L} \left(1 - \frac{R}{R+d_L} \right) \right] \quad (4)$$

$$\phi = \tan^{-1} \left[\frac{y}{d_L} \left(1 - \frac{R}{R+d_L} \right) \right] \quad (5)$$

X and Y are the Cartesian coordinates of the laser spot on the PSD, θ and ϕ are the angles of rotation in the X and Y dimensions respectively, d_L is the distance from the LD to the PSD at $\theta = \phi = 0$, and R is the radius of rotation measured from the center of the gimbal assembly.

4.1.3 Cartesian to Spherical Coordinate Conversion Derivation

The following figure is a simplified, mathematical version of the profile of the LSIS design. The top, blue line represents the PSD detector, while the red arc represents the path that the LD travels along upon a rotation. This simplification considers only the x coordinate dimension and negative angles θ .



Trigonometric relationships:

$$\tan(\theta) = \frac{\delta}{s} = \frac{r + \delta}{R} = \frac{x}{R + d_L} = \frac{x - r - \delta}{d_L}$$

Derivation:

First:

$$\tan(\theta) = \frac{x - r - \delta}{d_L}$$

$$x = r + \delta + \tan(\theta) d_L$$

Second:

$$\frac{r + \delta}{R} = \frac{x}{R + d_L}$$

$$\delta = \frac{Rx}{R + d_L} - r$$

Substitution the second into the first:

$$x = r + \frac{Rx}{R + d_L} - r + \tan(\theta) d_L$$

$$x - \frac{Rx}{R + d_L} = \tan(\theta) d_L$$

$$\left(\frac{x}{d_L}\right) \left(1 - \frac{R}{R + d_L}\right) = \tan(\theta)$$

$$\therefore \theta = \tan^{-1} \left[\frac{x}{d_L} \left(1 - \frac{R}{R + d_L}\right) \right]$$

$$\therefore \phi = \tan^{-1} \left[\frac{y}{d_L} \left(1 - \frac{R}{R + d_L}\right) \right]$$

Figure 49: Top View Mathematical Break Down of LSIS Design

The equations were recreated in MATLAB using the following code:

```
1 %%CDR Report Cartesian to Spherical Test Code
2
3 [x,y] = meshgrid(-4.5:0.01:4.5,-4.5:0.01:4.5);
4 R=1;
5 dL=0.05;
6
7 x0=(x/dL)*(1-R/(R+dL));
8 y0=(y/dL)*(1-R/(R+dL));
9
10 theta=atan(x0);
11 phi=atan(y0);
12 gamma=theta+phi;
13
14 mesh(x, y, gamma);
15 title('Cartesian to Spherical Conversion');
16 xlabel('X Coordinate (mm)');
17 ylabel('Y Coordinate (mm)');
18 zlabel('Angle of Rotation (\circ)');
```

Figure 50: MATLAB Code For Verifying Mathematical Derivation

This code allows for solving for a combined variable γ , which adds θ and ϕ together with respect to both the x, and y coordinates. R and d_L were chosen arbitrarily for now. The 3D plot below in Figure 51 is the result and represents the conversion from a Cartesian to spherical representation of the design, which must be taken into account for all angles other than zero.

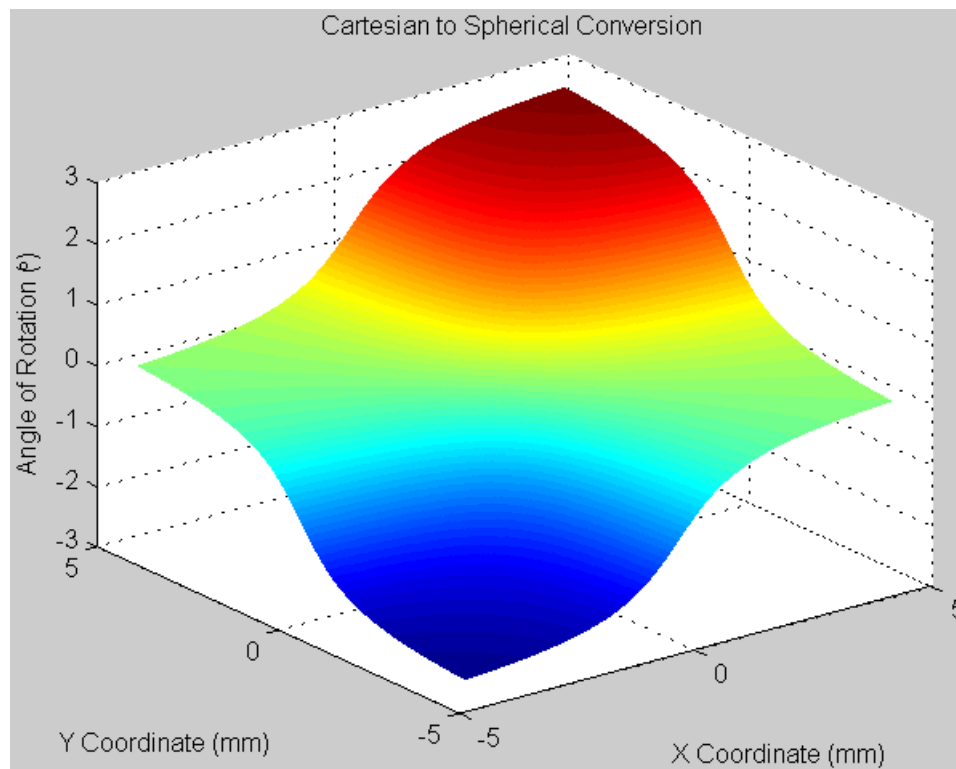


Figure 51: Full 3D Representation of Actual Cartesian Coordinate Compared to Physical Spot Location

It is easier to analyze this plot when considering only one dimension and values from -4.5mm to 0mm.

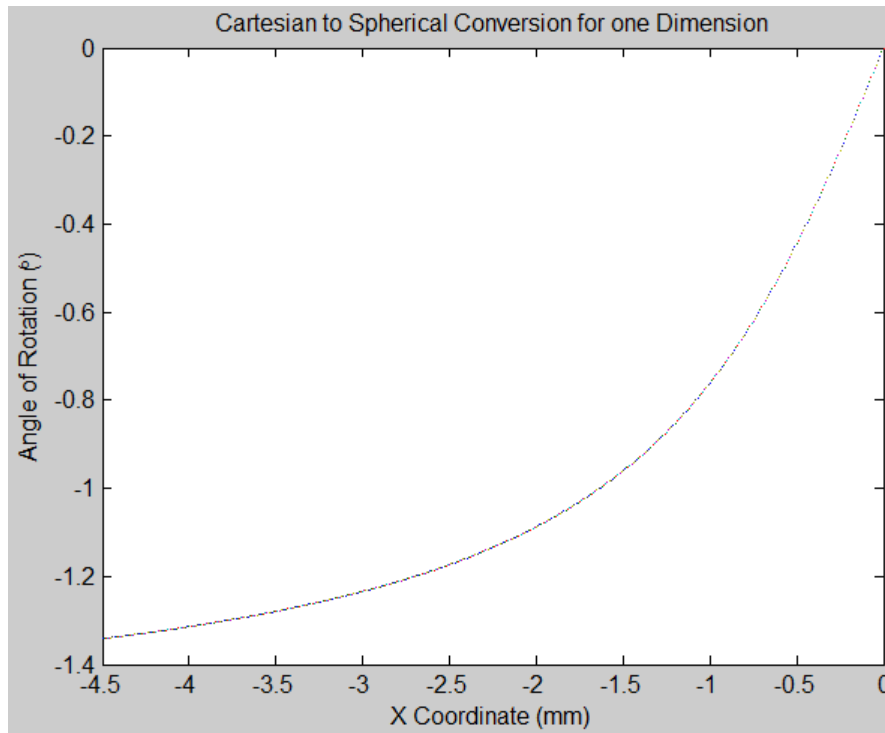


Figure 52: Half Spherical to Cartesian Representation in Two Dimensions

And then extending this into the 3 dimensional realm once again, but with the same boundaries produces the following plot:

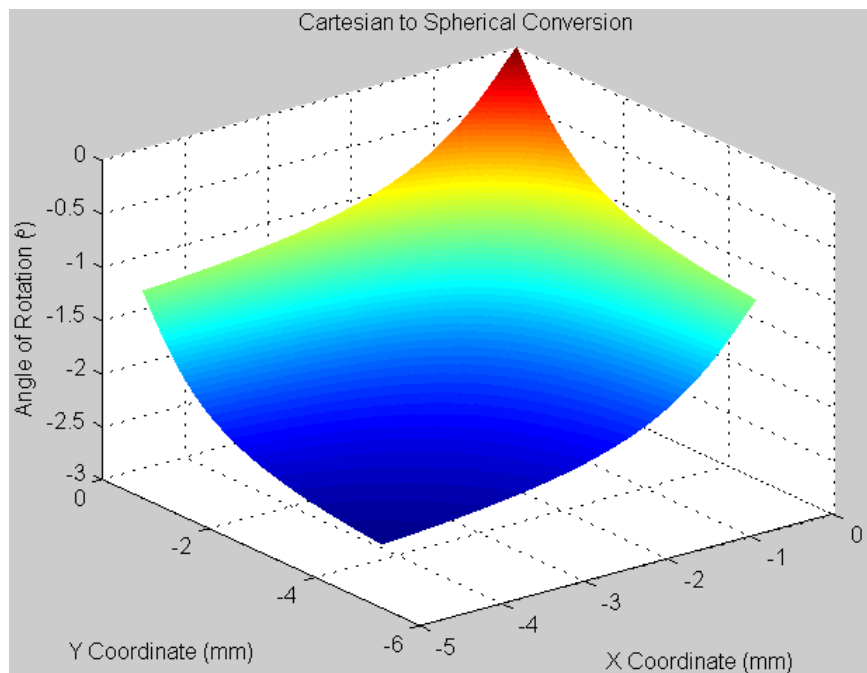


Figure 53: Half Spherical to Cartesian Representation in Three Dimensions

4.1.4 Details for and Utilization of the 16-Bit 28-Pin Starter Board by MICROCHIP

This low-cost, 16-bit, 28-pin Starter Development Board supports 28-pin PIC24 microcontrollers or Digital Signal Controller (DSC) devices. This board is an ideal prototyping tool to help validate key design requirements using these microcontrollers and DSCs and is shown below in Figure 54.

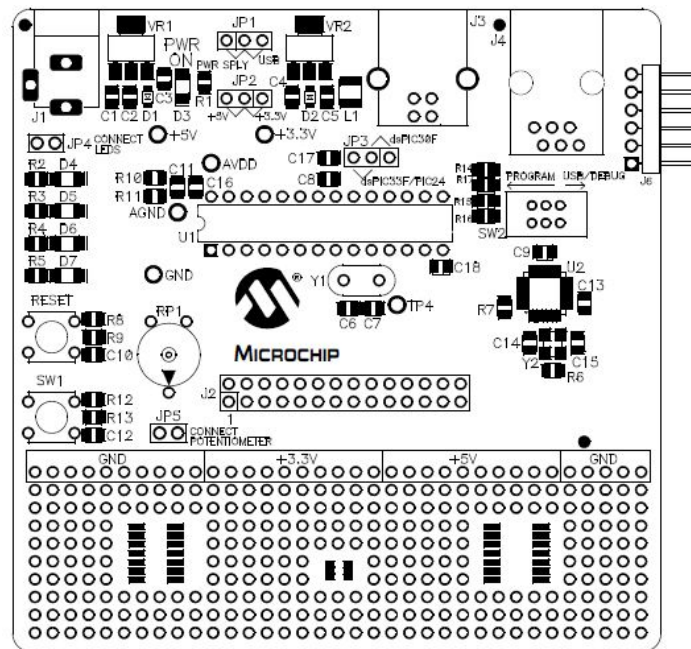


Figure 54: 16-Bit 28-Pin Starter Board by MICROCHIP

Features of this board include:

- Prototyping tool for all 28-pin, SDIP PIC24, dsPIC30F and dsPIC33F devices.
- Includes a 28-pin PIC24FJ64GA02 and dsPIC33FJ12GP202.
- On-board regulators for 3.3V or 5V operation.
- Power input from 9V power supply or USB power source.
- Single UART communication channel via USB bridge.
- Connectors for MPLAB® ICD 2 In-circuit Debugger/Programmer and PICKIT.
- Header for access to all device I/O pins.
- Circuit prototyping area including pads for SOIC and SOT-23 devices.

The board was used in conjunction with the PIC24F microprocessor. This chip allowed for the four outputs from the PSD to be read and analyzed in an internal C program (pins AN9-AN12). Additionally, the chip was configured such that two other pins would utilize Tx and Rx functionality via the chip's onboard UART (Universal Asynchronous Receiver/Transmitter). The whole setup is shown below in Figure 55.

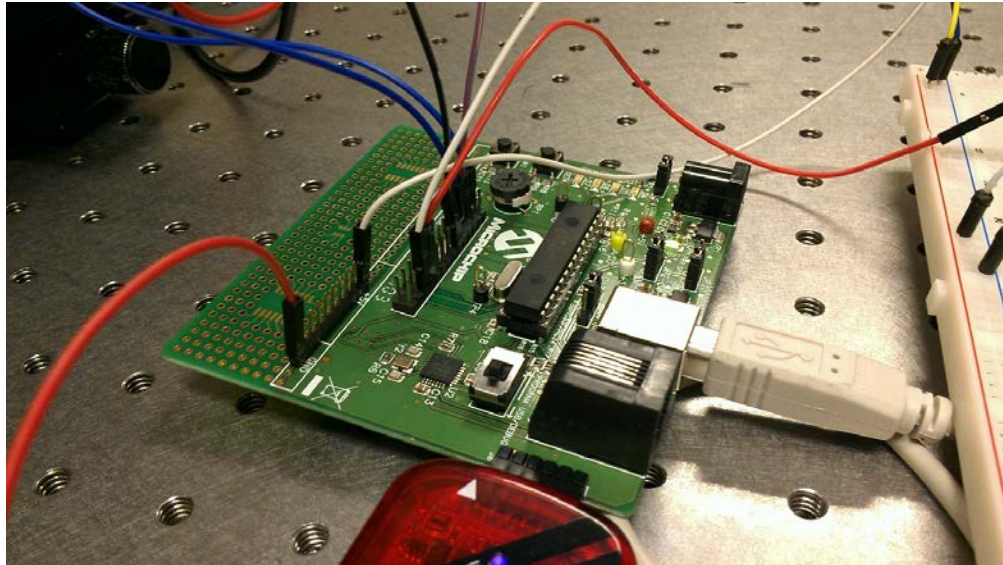


Figure 55: Microprocessor Setup for LSIS

Additionally, two LM747 operational Amplifiers were soldered to the board and used in order to amplify convert the output PSD signal into something that the PIC24F could readily utilize. The schematic for the LM747 op-amp is included below in Figure 56 and the basic circuit layout is also included below in Figure 57.

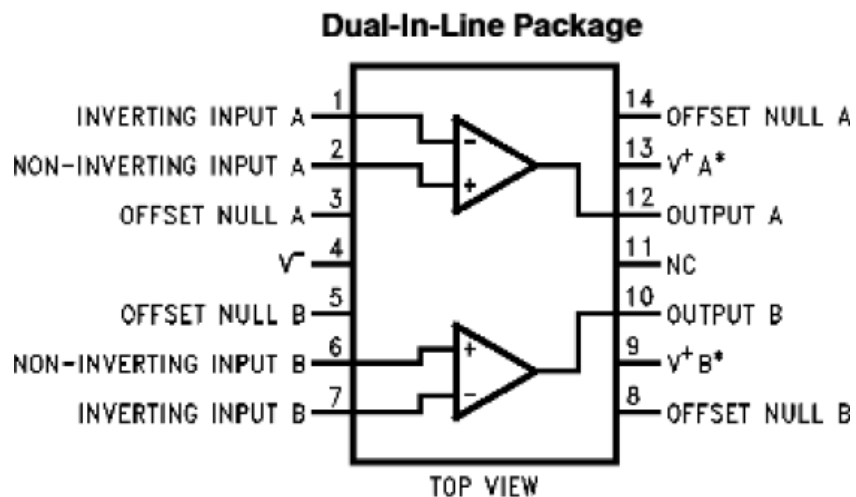


Figure 56: LM747 Dual Op-Amp

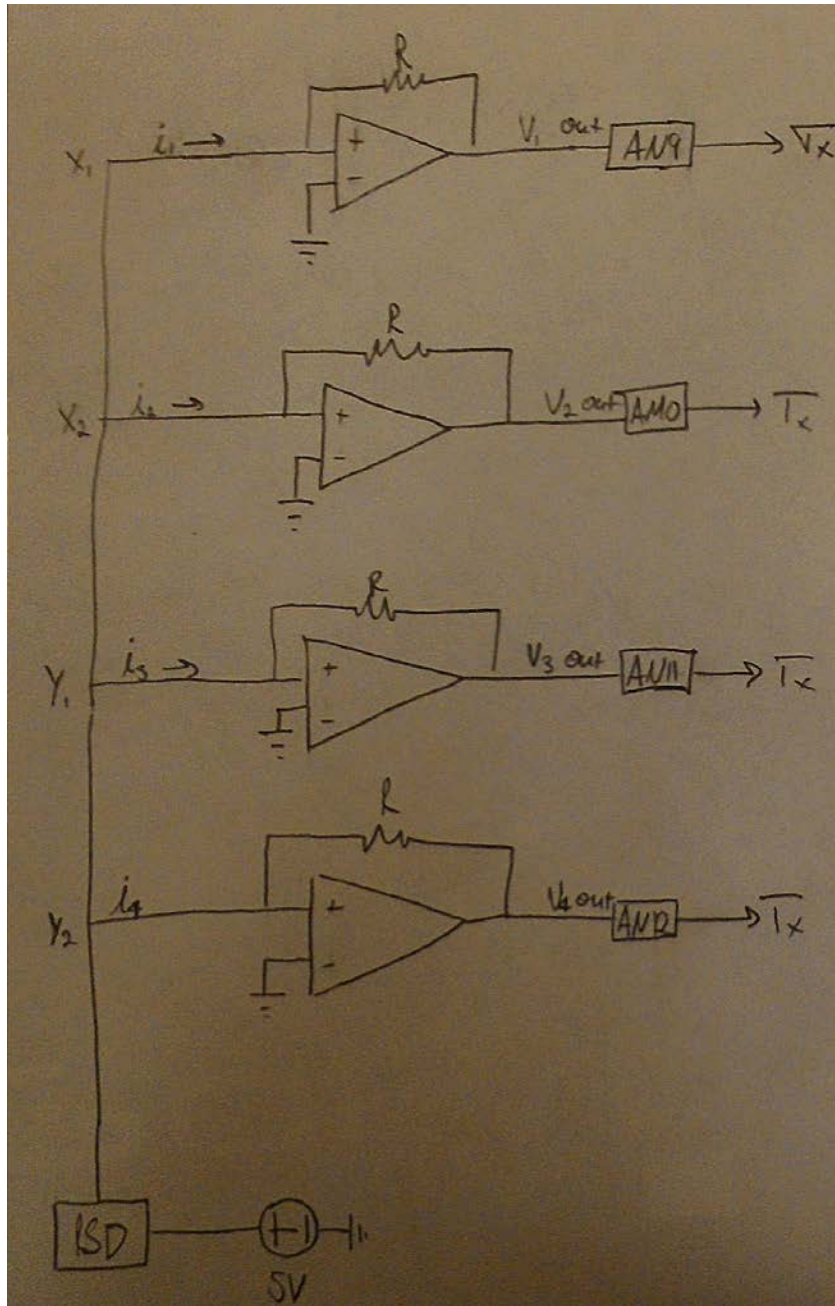


Figure 57: LSIS Rough Circuit Layout

4.1.5 UART Hardware Interface

In order to interface the LSIS with a terminal output viewing screen on a computer, a MAX232 dual driver/receiver, a R232 serial port to TTL converter and a serial to USB cable were used as the onboard transmission and receiving for the Microchip starter board was not functioning properly. The setup can be seen below in Figure 58. In short, this system was essentially the “middle man” between the PIC24F’s onboard UART and the computer to receive the output data.

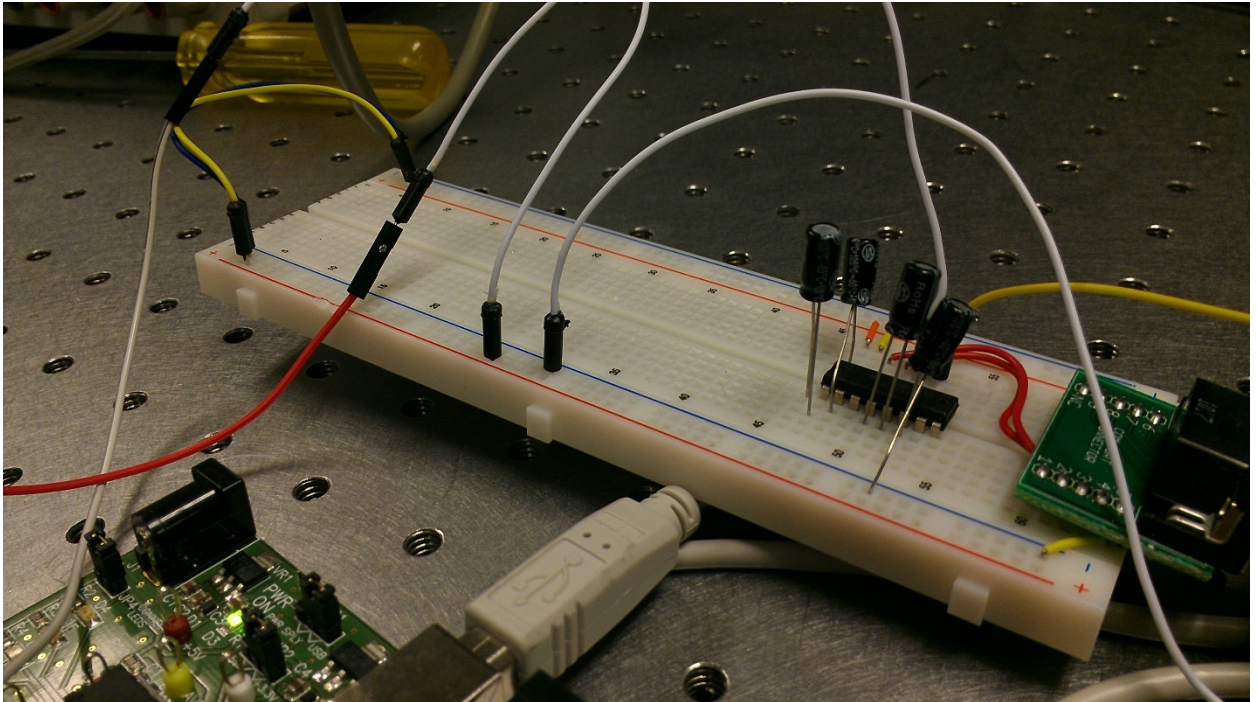


Figure 58: UART System Utilizing a 232Serial to USB interface

A schematic for the MAX232 from Ti is also shown below in Figure 59.

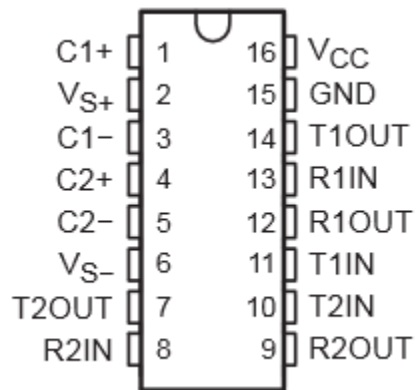


Figure 59: MAX232 Dual Driver/Receiver Chip

4.1.6 Code Used for Lab Testing of the LSIS Design

The following code was executed using the MicroChip MPLAB software. It is included below for the sponsor's reference and should be fully functioning assuming the setup is very similar to what was created in lab.

```
#include <stdio.h>
#include <stdlib.h>
#include <p24fj64ga002.h>
#include <math.h>
```

```
_CONFIG1(JTAGEN_OFF & GCP_OFF & GWRP_OFF & BKBUG_ON & COE_ON & ICS_PGx1 &
```

```
FWDTEN_OFF & WINDIS_OFF & FWPSA_PR128 & WDTPS_PS32768 )

_CONFIG2( IESO_OFF & SOSSEL_SOSC & WUTSEL_LEG & FNOSC_PRIPLL &
FCKSM_CSDCMD &
    OSCIOFNC_OFF & IOL1WAY_OFF & I2C1SEL_PRI & POSCMOD_XT )

#define XTFREQ      7372800      // On-board Crystal frequency
#define PLLMODE     4           // On-chip PLL setting (Fosc)
#define FCY         (XTFREQ*PLLMODE)/2 // Instruction Cycle Frequency (Fosc/2)

#define BAUDRATE    115200
#define BRGVAL      ((FCY/BAUDRATE)/16)-1

volatile float roc_theta=0, roc_phi=0;
volatile unsigned char a=0, b=0, c=0, d=0;
volatile double v1=0, v2=0, v3=0, v4=0;

void initTMR1()
{
    TMR1 = 0;
    PR1 = 5800*5;//500ms

    IFS0bits.T1IF = 0;
    IEC0bits.T1IE = 1;

    T1CON = 0x8030;
}

void initUART1()
{
    U1MODEbits.UARTEN = 1; // enable UART 1 pins

    RPINR18bits.U1RXR = 9;
    RPOR4bits.RP8R = 3;

    U1BRG = BRGVAL;
    U1MODE = 0x8000;
    U1STA = 0x0440;
    U1STAbits.UTXEN = 1;

    IFS0bits.U1RXIF = 0;
    IFS0bits.U1TXIF = 0;
}

void transmitUART1(char data)
{
    while(!U1STAbits.TRMT);//Check is UART1 is busy, if not then send
    U1TXREG = data;
}

void initADC()
{
    AD1PCFG = 0xE1FF; //AN9,10,11,12
    AD1CON1 = 0x00E0;
    AD1CSSL = 0;
}
```

```
AD1CON2 = 0;
AD1CON3 = 0x1F02;
AD1CON1bits.ADON = 1;
}

void msDelay(int a)
{
    //1 msDelay
    PR2 = (287/5)*a;
    TMR2 = 0;
    T2CONbits.TON = 1;
    while(TMR2 < PR2);
    T2CONbits.TON = 0;
    TMR2 = 0;
}

int readADC(int a)
{
    AD1CHS = a; // 1. select analog input channel
    AD1CON1bits.SAMP = 1; // 2. start sampling
    while (!AD1CON1bits.DONE); // 3. wait for the conversion to complete
    return ADC1BUF0;
}

int main(void)
{
    int vd1=0, vd2=0, vd3=0, vd4=0;
    double i1=0, i2=0, i3=0, i4=0;
    double x=0, y=0, L=10;
    double d_l=2.54, R=10, theta1=0, phi1=0, theta2=0, phi2=0;
    double T=1000,theta, phi;

    //TMR2 initialization
    IFS0bits.T2IF = 0;
    IEC0bits.T2IE = 1;

    initUART1();
    initTMR1();
    initADC();

    while(1)
    {
        //ADC register input current values from detector using a time sample T
        vd1 = readADC(9);
        msDelay(10);
        vd2 = readADC(10);
        msDelay(10);
        vd3 = readADC(11);
        msDelay(10);
        vd4 = readADC(12);
        msDelay(10);

        v1=(vd1*3.3)/1023;
        v2=(vd2*3.3)/1023;
    }
}
```

```
v3=(vd3*3.3)/1023;
v4=(vd4*3.3)/1023;

i1=v1/1000;
i2=v2/1000;
i3=v3/1000;
i4=v4/1000;

theta1=theta2;
phi1=phi2;

x=(L/2)*(((i2+i3)-(i1+i4))/(i1+i2+i3+i4));
y=(L/2)*(((i2+i4)-(i1+i3))/(i1+i2+i3+i4));

theta2=atan((x/d_l)*(1-R/(R+d_l)));
phi2=atan((y/d_l)*(1-R/(R+d_l)));

roc_theta=(theta1-theta2)/T;
roc_phi=(phi1-phi2)/T;

}
return (EXIT_SUCCESS);
}

void __attribute__((interrupt, no_auto_psv)) _T1Interrupt(void)
{
    _T1IF = 0;
    printf("Rate of Change Theta: %2.3f\r\n", roc_theta);
    printf("Rate of Change Phi: %2.3f\r\n", roc_phi);
    //
    printf("v1: %2.3f\r\n", v1);
    printf("v2: %2.3f\r\n", v2);
    printf("v3: %2.3f\r\n", v3);
    printf("v4: %2.3f\r\n\r\n", v4);
}

void __attribute__((interrupt, no_auto_psv)) _T2Interrupt(void)
{
    _T2IF = 0;
}
```

5 CHANGES TO THE ALTERNATIVE LSIS DESIGN

5.1 Laser diode change

The L7650 LD, also sold from Hamamatsu Corporation, was originally chosen because its output wavelength was closest to the peak sensitivity of the PSDs and its beam width was as small as possible while still within the required spot size resolution. The LD is shown below in Figure 60.



Figure 60: L7650 LD

This LD was eventually eliminated from the design because of the amount of time it would take to receive from the company. Additionally cheaper alternatives were found.

6 INITIAL TESTING

Before coding took place, we manually tested the LSIS design for very rudimentary and basic functionality. In place of accurate measurements, general and decently approximated ones were taken. Figure 61 and Figure 62 below attempt to show what this general testing looked like in lab. Essentially, a Keplerian beam expander was created using a microscope objective, spatial filter, collimating lens, and an exit lens. The system allowed for the use of a larger laboratory laser to be used in place of the rather lacking IR LD ordered from US-Lasers. Additionally, the laser spot size was more readily controlled if a larger or smaller spot size was desired and also filters could easily be added into the system to reduce the incident beam intensity upon the PSD. Ultimately the s%991-1 PSD seemed to work as promised, without delving into very accurate measurements.

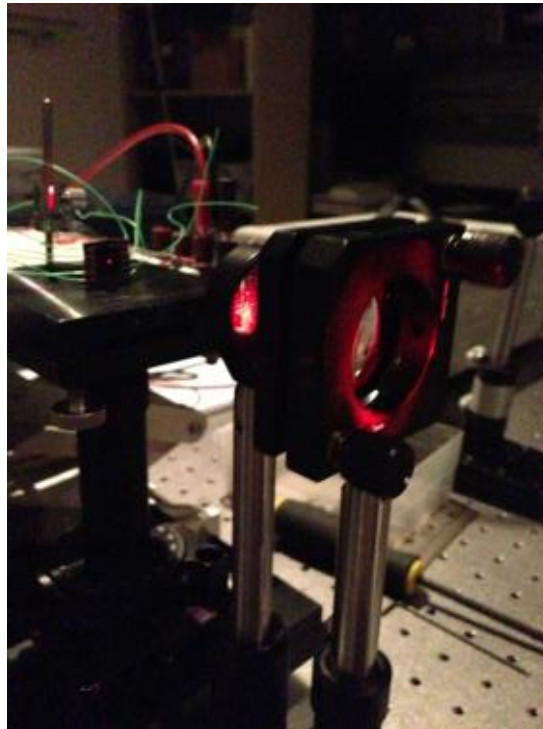


Figure 61: LSIS Close-up of Receiving End of Keplerian Beam Expander (Detector)

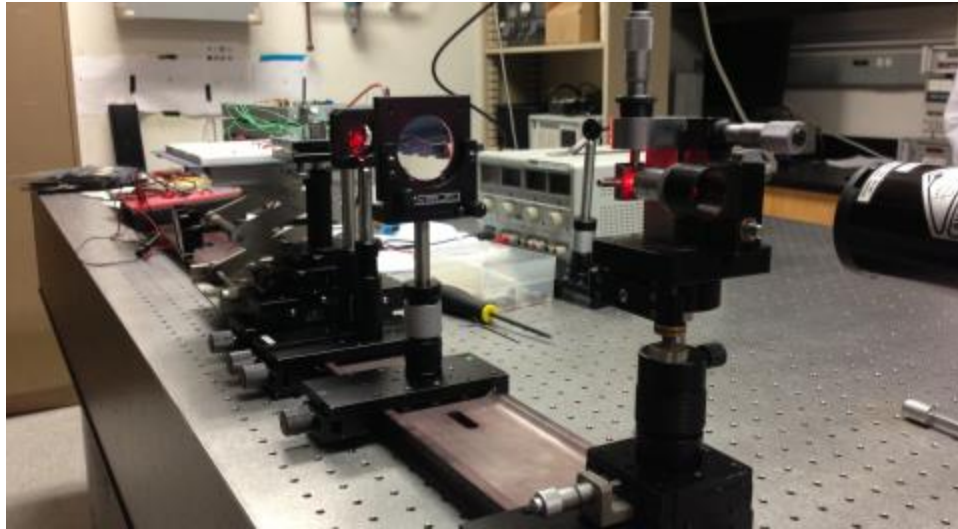


Figure 62: LSIS Close-up of Transmitting End of Keplerian Beam expander (Source)

7 ELIMINATION FROM PROJECT

On April 16th, 2013 the LSIS design was finally dismissed. The electronics that amplified the signal coming out of the PSD were giving inconsistent readings for unchanged circuit designs. The exact resistor values as well as amplifier configurations were theoretically difficult to obtain due to the lack of technical information provided regarding the S5991-1 from Hamamatsu. Not to mention the lack of a properly working infrared laser diode, the PSD also seemed to be malfunctioning, potentially from over use, or negligence. By this point, given the time crunch and lack of test data for both the gyroscope design as well as LSIS, the team decided to forgo the LSIS design in lieu of providing the main design with 100% support.

Roles and Responsibilities

Jonathan A Cox (Mechanical Engineer):

Member was responsible for all mechanical drawings for the design as well as researched and explained the theory behind the usage of gyroscopes to solve the initial problem. Member also performed a preliminary analysis on the gyroscopes with included data regarding the specifications of the commercial parts.

Martin M Lopez (Systems and Industrial Engineer):

Member was responsible for team time management in the form of a Gantt chart and largely aided the rest of the team where needed including programming, modeling, the development of requirements and the acceptance test plan as well as general research.

Craig Warren McNabb (Optical Engineer and Project Leader):

Team leader was the primary contact between sponsor and team. Member also utilized testing equipment at sponsor location to fully test the gyroscope design against all requirements.

Siddharth Narang (Electrical Engineer):

Member was responsible for all electrically related components for the gyroscope design. This includes all MPLAB program as well as hardware configuration and soldering regarding the evaluation boards as well as the microprocessors required for gyroscope functionality.

Matthew Ryan Schellenberg (Systems and Industrial Engineer):

Member was responsible for the development of the requirements and the acceptance test plan. Also managed the team budget, risk analysis and risk mitigation including the research of unused parts for the project such as thermo electric heaters.

Conor Staples (Optical Engineer):

Member was responsible for the research and development of the risk mitigation plan dubbed LSIS (Laser Spot Imaging System). Any parts, design, or documentation regarding this part of the project was largely managed by this team member.