Design and Implementation of a Preload Electronics Architecture for a MEMS Accelerometer

by

Alvin Lai Lin

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of Master of Engineering in Electrical Engineering and Computer Science at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2006

© Alvin Lai Lin, MMVI. All rights reserved.

The author hereby grants to MIT permission to reproduce and distribute publicly paper and electronic copies of this thesis document in whole or in part.
Design and Implementation of a Preload Electronics Architecture for a MEMS Accelerometer

by

Alvin Lai Lin

Submitted to the Department of Electrical Engineering and Computer Science on February 3, 2006, in partial fulfillment of the requirements for the degree of Master of Engineering in Electrical Engineering and Computer Science

Abstract

This thesis describes the design and implementation of an electronics system to provide rebalancing and readout for a force-rebalanced microelectromechanical accelerometer. A feedback control loop is devised using a novel preload architecture, compensating the proof mass of the sensor and providing an accurate acceleration measurement. This architecture is compared to alternative methods of linearizing the control loop. The electronics system is divided into analog and digital subsystems. The design is analyzed at several abstraction levels. The system is implemented for prototype testing with discrete components on a printed circuit board with a MEMS sensor attached. A computer program is implemented to receive and process the readout data using the serial port.

The design methodology consists of a top-down design flow based on simulation. At each iteration in the design process, the lower level abstraction is verified with the previous model. Eventually, the design reaches the level of synthesizable digital logic and discrete analog components. This thesis describes the design process and implementation details for creating an accelerometer system prototype ready for lab testing. Detailed simulations indicate that the implemented design is likely to meet the design goals for a personal navigation system suitable for a human or land vehicle. Conclusions on design methodology and verification techniques are also presented.

Technical Supervisor: David J. McGorty
Title: Principal Member Technical Staff, C.S. Draper Laboratory

Thesis Advisor: Jeffrey H. Lang
Title: Professor
Acknowledgments

I would like to thank my thesis supervisors, Dave McGorty and Professor Jeff Lang for their guidance and support during this research. In addition, this work would not have been possible without the tremendous contributions of my friends and colleagues at Draper Laboratory, especially Richard Elliott and John LaChapelle. I am extremely grateful for their assistance throughout the course of my research.

This thesis was prepared at The Charles Stark Draper Laboratory, Inc., under Contract B54530612 / 4200047939, sponsored by Honeywell International Inc., Minneapolis, Minnesota.

Publication of this thesis does not constitute approval by Draper or the sponsoring agency of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.

Permission is hereby granted to the Massachusetts Institute of Technology to reproduce any or all of this thesis.

Alvin L. Lin
# Contents

## 1 Introduction
1.1 Overview ....................................................... 15  
1.2 Design Methodology ........................................ 16  
1.3 Research Objective .......................................... 18  
1.4 Thesis Outline ................................................... 18

## 2 System Overview
2.1 Sensor Characteristics .......................................... 20  
2.1.1 Ideal Sensor Model & Behavior ............................. 20  
2.1.2 Physical Sensor Details & Consequences ................. 26  
2.1.3 Modeling the MEMS Sensor ................................. 27  
2.2 Electronics Requirements .................................... 28  
2.3 Architecture Design ........................................... 28  
2.3.1 Preload Architecture Concept .............................. 29  
2.3.2 Alternative Architecture ................................... 31  
2.4 Analog & Digital Subsystems ................................. 32  
2.5 Frequency and Sample Rate Selection ....................... 34  
2.6 Functional Simulation ........................................... 36

## 3 Digital Design
3.1 Digital Overview ............................................... 39  
3.2 Demodulator ..................................................... 41  
3.3 CIC Decimation Filter .......................................... 43
5.5.2 Sensor Placement ...................................... 90
5.5.3 Bypass/Decoupling Capacitors ..................... 90
5.5.4 Symmetry ............................................... 91

6 Conclusions and Recommendations for Future Work 93
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Top-down Design Methodology</td>
<td>17</td>
</tr>
<tr>
<td>2-1</td>
<td>High Level Block Diagram of Accelerometer System</td>
<td>20</td>
</tr>
<tr>
<td>2-2</td>
<td>Physical Diagram of MEMS Accelerometer</td>
<td>22</td>
</tr>
<tr>
<td>2-3</td>
<td>Ideal Characterization of Sensor Scale Factor</td>
<td>23</td>
</tr>
<tr>
<td>2-4</td>
<td>Ideal Characterization of Sensor Torque Constant</td>
<td>24</td>
</tr>
<tr>
<td>2-5</td>
<td>Example Scheme to Rebalance Sensor Using DC Torquing</td>
<td>25</td>
</tr>
<tr>
<td>2-6</td>
<td>Finite Slew Rate Effects on Torque Voltage</td>
<td>27</td>
</tr>
<tr>
<td>2-7</td>
<td>Preload Scheme to Rebalance Sensor</td>
<td>30</td>
</tr>
<tr>
<td>2-8</td>
<td>Preload Architecture Electronics Overview</td>
<td>33</td>
</tr>
<tr>
<td>2-9</td>
<td>Frequency Spectrum of Ideal 20KHz Square Wave</td>
<td>34</td>
</tr>
<tr>
<td>2-10</td>
<td>Functional Model Setup for Preload Architecture</td>
<td>37</td>
</tr>
<tr>
<td>2-11</td>
<td>Functional Model Simulation Results</td>
<td>38</td>
</tr>
<tr>
<td>3-1</td>
<td>Digital Subsystem Block Diagram</td>
<td>40</td>
</tr>
<tr>
<td>3-2</td>
<td>Optimized Sine Storage for Demodulator ROM</td>
<td>42</td>
</tr>
<tr>
<td>3-3</td>
<td>Demodulator Multiplexer to Eliminate Multiplications</td>
<td>43</td>
</tr>
<tr>
<td>3-4</td>
<td>Integrator Stage for CIC Filter</td>
<td>44</td>
</tr>
<tr>
<td>3-5</td>
<td>Comb Stage for CIC Filter</td>
<td>44</td>
</tr>
<tr>
<td>3-6</td>
<td>Frequency Response for CIC Decimation Filter</td>
<td>45</td>
</tr>
<tr>
<td>3-7</td>
<td>Block Diagram for CIC Decimation Filter</td>
<td>47</td>
</tr>
<tr>
<td>3-8</td>
<td>Simulation Model for PI Controller</td>
<td>48</td>
</tr>
<tr>
<td>3-9</td>
<td>System Step Response for Properly Tuned $K_p$ and $K_i$</td>
<td>49</td>
</tr>
<tr>
<td>3-10</td>
<td>System Response to Input Sine Wave at 100Hz</td>
<td>50</td>
</tr>
</tbody>
</table>
3-11 Block Diagram for CIC Interpolation Filter ...
3-12 Noise Transfer Function Magnitude Response for D/A Sigma-Delta Modulator ...
3-13 Standard Form for Sigma-Delta Modulator ...
3-14 Component of Output Transmitter ...
3-15 Example UART Packet ...
3-16 Order of Transmitted Output Data Packs ...
3-17 Output Transmitter Finite State Machine ...

4-1 Abstract Analog Subsystem Block Diagram ...
4-2 Detailed Analog Subsystem Block Diagram ...
4-3 Power Spectrum for Sigma-Delta Modulator Output ...
4-4 Zoomed Power Spectrum for Sigma-Delta Modulator Output ...
4-5 Analog Subsystem Active 4th Order Sallen-Key Butterworth Low Pass Filter Circuit ...
4-6 Analog Subsystem Low Pass Filter Magnitude Response ...
4-7 Power Spectrum of Analog Subsystem Low Pass Filtered Output ...
4-8 Inverter Circuit ...
4-9 Slew-Rate Limiter Circuit: First Order Sallen-Key Butterworth Filter ...
4-10 Slew-Rate Limited Torque Signal ...

5-1 Digital Torque Magnitudes in Response to a +10g Step Input ...
5-2 Screenshot of Data Capturing Java Application ...
5-3 Layer Stackup for PCB ...
5-4 Power Plane in PCB ...
5-5 Sensor Placement and Signal Integrity Protection ...
5-6 Bypass Capacitor Placement for the MAX3221 ...
5-7 Bypass Capacitor Placement for the FPGA ...
List of Tables

1.1 Design Goals for Accelerometer Electronics ........................................... 16
2.1 Typical Sensor Characteristics ................................................................. 23
2.2 Nominal Voltages for Preload Architecture .............................................. 30
2.3 Nominal Frequencies and Sample Rates for Preload Architecture ................. 34
3.1 CIC Decimation Filter Parameters ............................................................. 43
3.2 CIC Decimation Filter Attenuation at Selected Frequencies ......................... 46
3.3 Register Sizes for CIC Decimation Filter .................................................. 47
3.4 Tuned $K_i$ and $K_p$ From Step Response of Detailed Simulation ................ 49
3.5 CIC Interpolation Filter Parameters ......................................................... 51
3.6 Register Sizes for CIC Interpolation Filter ............................................... 51
3.7 Density of Ones for Sigma-Delta Modulated Outputs ................................... 53
3.8 Parameters for D/A Sigma-Delta Modulator .............................................. 54
3.9 State Space Coefficients for Sigma-Delta Modulator ................................... 56
3.10 Feedback Values for Sigma-Delta Modulator ............................................ 57
3.11 Acceleration Output CIC Decimation Filter Parameters ............................. 59
3.12 Register Sizes for Acceleration Output CIC Decimation Filter ................... 60
3.13 Compensation Variable CIC Decimation Filter Parameters ........................ 60
3.14 Register Sizes for Compensation Variable CIC Decimation Filter ................ 61
3.15 Synthesis Sizes for Digital Blocks .......................................................... 64
3.16 Percent of Total Synthesis Size for Digital Blocks ................................... 65
4.1 Level Shifter Voltage Levels ..................................................................... 69
4.2 Level Shifter Signal Representations ............................................. 70
4.3 Example Scheme to Create Out-of-Phase Commutated Torque Signals 75

5.1 Components Chosen for the Prototype Board .............................. 84
5.2 Percentage Utilization of the FPGA ............................................ 85
5.3 Voltage Levels, Functions, and Presence on Power Plane of Voltage
 Signals in the Preload System ...................................................... 88
Chapter 1

Introduction

1.1 Overview

The development of microelectromechanical systems (MEMS) has unlocked a spectrum of applications which are uniquely leveraged by these systems. MEMS systems contain electronic components as well as micromachined mechanical components and are able to provide sensing, mechanical action and computation capabilities on one integrated circuit [10]. Such a MEMS system is capable of better performance and longer operational lifetime than the corresponding macroscopic system. For these reasons, the Charles Stark Draper Laboratory has been developing high performance inertial sensors using MEMS technology. These sensors typically consist of single axis accelerometers or plane rotation gyroscopes.

Single axis accelerometers are useful for many applications involving navigation or movement. The acceleration measurement can be used to prevent a vehicle or person from being operated in a manner that exceeds its safety boundaries or structural integrity limits. The acceleration can also be integrated over time to provide a velocity estimate. This velocity estimate is useful for navigation purposes; it can be integrated over time to provide a position estimate. Using MEMS technology to implement accelerometers allows for a small package size and a durable and reliable system. In addition, several accelerometers can be packaged together to allow for an integrated three axis solution.
<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Target Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Range</td>
<td>±10g</td>
</tr>
<tr>
<td>Input Bandwidth</td>
<td>0Hz to 100Hz</td>
</tr>
<tr>
<td>Input Attenuation</td>
<td>&gt;-3dB</td>
</tr>
</tbody>
</table>

Table 1.1: Design Goals for Accelerometer Electronics

The MEMS accelerometer is a passive sensor that consists of micromachined mechanical parts that can be stimulated with electrical signals [10]. An electronics system is required to read, filter, and process the signals coming out of the sensor as well as provide the necessary signals going into the sensor [28]. This thesis describes the design and implementation of such an electronics system. The electronics system uses a novel approach that has not been used before in the design of accelerometer electronics. Thus, there are no stringent specifications placed on the performance of the final system. Rather, there are several design goals set with the intention of providing a framework from which performance can then be measured and compared to alternative systems. These design goals are outlined in Table 1.1. They are the specifications necessary (without precision requirements) for a personal navigation accelerometer.

1.2 Design Methodology

The design methodology consists of a hierarchical approach with a heavy emphasis on simulation. Recent advances in computer simulation technology allow the system to be represented from an ideal mathematical model all the way down to register transfer level (RTL) logic for the digital subsystem. Confirmation of a working system at all levels of abstraction ensure that the design works as intended and will have minimal unexpected effects after implementation and fabrication.

The first modeling and simulations are run at the highest possible level, with subsequent simulations adding detail by replacing high level blocks with more detailed blocks. This is commonly referred to as a top-down design process [23]. As the blocks gain detail, they more closely resemble the real world components which they
represent. The design is complete when the blocks are as detailed as possible can no longer be replaced with more exact models. At this point, the design is ready to translate into real world electronic components for final testing.

Figure 1-1 illustrates the top-down design methodology for the entire electronics system. In the first step, the entire system is simulated at a functional level. As each simulation at a given abstraction level runs satisfactorily, the design process can continue by following the arrows down the chain. After the functional simulation, the first digital simulation is a detailed version of the digital components. In this detailed model, all digital signal processing structures are intact in their architecture, but they are not detailed to the point where registers and transistors are defined. Next, VHDL is used to describe the digital hardware and the simulation results of this hardware are compared to the detailed simulation results. The only difference between the detailed simulation and the VHDL simulation should be the differing levels of quantization found in the two simulations. The final testing phase for the digital electronics system is to synthesize the digital logic onto an FPGA for lab testing. This is the most convenient way to run an test with an actual MEMS sensor rather than a model.

The analog design methodology is also shown in Figure 1-1 and is different than the digital design methodology. The functional simulation can lead directly into an analog simulation of the design, bypassing the detailed simulation found in the digital design flow. A detailed simulation is not necessary because the functional simulation
can translate well into the analog design and simulation without an intermediate step. After the analog simulation is verified, lab testing can be performed to verify that the analog components match the analog simulation.

1.3 Research Objective

The objective of this thesis is to design and implement an electronics design for a force rebalanced MEMS accelerometer using the preload architecture. Since the preload architecture has never been implemented, the design requires a development of the entire electronic system. The design process is clearly described and documented, with all significant design decisions explained. The final product is a working MEMS accelerometer system on a test board, capable of accurately measuring acceleration within the limits described in Table 1.1.

1.4 Thesis Outline

The remainder of this thesis is organized as follows. Chapter 2 provides a high level system overview of the accelerometer electronics system, as well as a detailed description of the MEMS sensor. Chapter 3 presents the design of the digital logic. Chapter 4 presents the design of the analog circuitry. Chapter 5 describes the software design as well as other implementation details such as part selection and printed circuit board (PCB) layout. Chapter 6 details some final conclusions and makes suggestions for future work.
Chapter 2

System Overview

In this chapter, the overall system is described. The MEMS sensor is explained and characterized in detail. Next, the electronics requirements for the system are given. After this, the preload architecture is presented, which provides insight on how the system is organized. Alternative architectures are also discussed along with several key advantages and disadvantages of each alternative. Finally, the overviews of the analog and digital subsystems are presented with an explanation on the separation of the two domains.

A high level block diagram of the accelerometer system is shown in Figure 2-1. In this figure, the entire electronics system is abstracted into one block. The MEMS sensor receives input signals from the electronics system, and it responds to these with an output signal. This sensor output signal is then taken into the electronics system where processing and analysis is done. The electronics system produces an acceleration estimate from the sensor output, which it packages and sends to a computer so that readout can be performed. In addition, the electronics system takes this acceleration estimate and uses it to provide the proper input signals into the MEMS sensor.
2.1 Sensor Characteristics

The MEMS accelerometer used for this thesis belongs to a class of MEMS accelerometers known as force-rebalanced accelerometers [28]. This name refers to the fact that a force is applied to the MEMS device in order to eliminate unwanted non-linear effects. To understand how these accelerometers are used in a closed loop system, the sensor must be understood in more detail. This section describes the workings of the MEMS accelerometer.

2.1.1 Ideal Sensor Model & Behavior

The sensor around which the preload architecture is designed can be modeled as an asymmetrical proof mass rotating around a pivot, as shown in Figure 2-2. The input axis is perpendicular to the plane of the proof mass. Because the proof mass is asymmetrical, it has more mass on one side of the pivot and applied acceleration on the input axis causes a rotation around the pivot. This angle of rotation can then be detected by capacitive pickoff.

Referring to Figure 2-2, the distance between the sense plates and the proof mass changes as the mass rotates around the pivot. Since the capacitance is inversely
proportional to the distance between the parallel plates, the capacitance thus changes depending on the rotation angle of the proof mass. When a voltage is applied to the sense plate, charge is accumulated on the proof mass according to the capacitor law $Q = CV$. Thus, as the capacitance changes, the amount of charge accumulated on the proof mass changes as well. An external charge amplifier can convert this charge into a voltage level, and thus the rotation of the proof mass can be detected.

Movement of the proof mass can be modeled as dynamics of a mass-spring-damper system. The amount of rotation of the proof mass has a non-linear relationship with acceleration, dependent on the spring constant. Detection of the rotation is also non-linear, of the form $\frac{1}{g}$ where $g$ is the distance gap between the sense capacitors and the proof mass. This is due to the inverse relationship between capacitance and distance. These non-linearities pose a problem for the electronics system. While the acceleration could be computed using non-linear equations, this method would rely on models of the sensor which may not be fully accurate due to real world non-idealities. In addition, the range of inputs is limited because the proof mass will eventually reach its physical limit of rotation. At this point, the proof mass will not be able to rotate further and stronger accelerations will not be measurable.

A better method for obtaining an accurate acceleration measurement is to ensure that the proof mass never moves significantly. By running the sensor in a closed loop system, the proof mass can be restored to its null position during operation, thus eliminating the non-linear factors. In the closed loop system, the position of the proof mass remains relatively constant and the rebalancing force will have a linear relationship with the input acceleration. This rebalancing force is equal and opposite to the external acceleration torque, and thus the acceleration can be measured.

Four torque plates are used to provide the rebalancing force to the proof mass. Two plates are above and below the proof mass left of the pivot point and the other two plates are on the right side of the pivot point. The left and right torquers provide force around the pivot point in opposite directions. Figure 2-2 shows a side view representation of the MEMS accelerometer, illustrating the torque and sense plates.

If the sensor is stabilized by feedback, the proof mass remains in the null position
and the spring and \( \frac{1}{g} \) non-linearities can be ignored. In order to design this feedback loop, though, the characteristics of the sensor must still be understood. The sensor characteristics must also be known in order to construct an accurate model of the sensor. Such a model is required in order to design a feedback system that can properly rebalance the sensor.

Although there are non-linear effects regarding the relationship between input torque voltage, input acceleration, and output voltage, the system can be simplified into small signal inputs and the corresponding small signal output. This simplification is possible if we assume that in the final system, the proof mass is restored to the null position. As long as the null position is maintained, simplifications can be made in the ideal model of the sensor. The rotational stiffness and the capacitor non-linear dynamics can both be ignored. In this case, the capacitive pickoff voltage coming out of the charge amplifier is linearly dependent on the input acceleration. This gain term is referred to as the scale factor of the sensor. The scale factor thus has units of \( \frac{V}{g} \). In other words, applying a certain amount of acceleration to the sensor will result in a linearly dependent amount of voltage on the output where the gain is given by the scale factor. The scale factor is measured in real world \( g \)’s where \( g = \frac{9.8 \text{m}}{s} \) to simplify design and testing.
Again assuming that the input acceleration as well as the input torques are relatively small, the input torque voltages have a square relationship with the position of the proof mass. The proportionality constant relating rebalanced $g$'s to input voltage squared is referred to as the torque constant. Rebalanced $g$'s refers to the input acceleration that would be necessary to produce a certain output voltage. The torque constant could have been given in terms of output voltage divided by input voltage squared, but the definition in terms of rebalanced $g$'s is more convenient for design and testing. The torque constant has units of $\frac{mV}{g}$. Typical values for the scale factor and torque constant are shown in Table 2.1.

To illustrate the effect of acceleration input on the accelerometer output voltage,
we must assume that the proof mass is at the null position. Of course, this depends on the system being properly rebalanced, but the initial assumption can be that the mass does not move. At the null position, the non-linearities mentioned above can be ignored. If the input torquers are held at 0V, the output voltage varies as shown in Figure 2-3. The x-axis represents the input acceleration in g's, where $g = \frac{9.8m}{s}$. The y-axis represents the output voltage. Thus, the slope of this graph is the scale factor of the accelerometer.

The accelerometer has two sets of torque plates, the left set and the right set. Applying a certain voltage $V$ to the left set of torquers results in a rotational force on the proof mass around the pivot. Applying that same voltage $V$ to the right set of torquers results in an equal and opposite rotational force on the proof mass. Since the force applied is proportional to the square of the voltage, it doesn’t even matter if $+V$ or $-V$ is applied, since the force will be proportional to $V^2$ anyways. Assuming there are no other non-linearities in the system we can create a graph of the acceleration.
that would be balanced by a certain voltage applied to each torque plate. This graph is shown in Figure 2-4.

In Figure 2-4, one curve corresponds to the left torque plate active (with the right torque voltage at 0V). The other curve is the opposite situation with the right torque plate active (with the left torque voltage at 0V). This graph implies one rebalancing methodology for a closed loop accelerometer system. Since we know that both Figures 2-3 and 2-4 hold true when the proof mass is at its null position, we can match the graphs to each other and figure out what voltage to apply to the torque plates. For example, if the input acceleration was 10g, 5.9V should be applied to the left torquer. If the input voltage was -10g, then 5.9V should be applied to the right torquer. This scheme was used for previous accelerometer designs and works adequately, although it suffers from a few drawbacks which are mentioned in Section 2.3.2.

Since the relationship between input acceleration and necessary rebalancing voltage is known, a scheme can be developed that rebalances the sensor based on the
current acceleration. An example graph of transfer characteristics is shown in Figure 2-5. This is not the scheme used in the preload architecture, but it does help demonstrate the concept behind force rebalancing the proof mass. In this scheme, when there is a positive acceleration, a DC voltage is placed on the right torquer. When there is a negative acceleration, a DC voltage is placed on the left torquer. Obeying the torque constant scaling factor, the DC voltage shown in Figure 2-5 perfectly rebalances the sensor and keeps the proof mass in the null position. Although this is not the rebalancing scheme used in the preload architecture, this scheme provides a conceptual foundation on which the preload architecture can be built.

2.1.2 Physical Sensor Details & Consequences

As the sensor is being torqued, the torquers cannot simply use a DC voltage to rebalance the proof mass. A DC voltage would build charge on the proof mass, creating another non-linear system as the charged proof mass experiences electrostatic forces with the charged torque plates. In addition, the charge built on the proof mass would affect the charge on the sense plates, thus corrupting the output signal. To counteract this effect, a square wave is applied to the torquer plates so that no net charge builds up on the proof mass. In actuality, charge inevitably does build up on the proof mass, but the torque square wave can operate at a frequency distinct from the sense square wave so that the two do not interfere with each other. An ideal square wave has the same effect as a DC voltage due to the square law. For each pair of torquer plates, the upper plate and the lower plate have opposite polarity input square waves so that at any given moment, they both provide the same direction of force on the proof mass.

Although an ideal square wave would have the same net torque as a DC voltage, it is impossible for a real electronic system to generate an ideal square wave. Instead, the system generates a square wave with a finite slew rate. An example wave is shown in Figure 2-6. This figure shows the equivalent torquing voltage that could be applied if charge buildup on the proof mass was not a concern. This equivalent voltage is simply the absolute value of the actual voltage placed on the torquer plates. This
effective voltage is not a clean DC voltage and is marred whenever the square wave switches polarity. Thus, the actual torque placed on the sensor is neither constant nor at the desired voltage level. A non-constant torque is acceptable, as long as we ensure that the frequency spectrum does not affect the sensor operation or the readout of the sensor. However, differing slew rates on the left and right torquers (due to mismatched analog components) will result in differing equivalent DC voltages. The solution is to purposely reduce the slew rate to a known value so that both left and right torquers match exactly in their slew characteristics.

2.1.3 Modeling the MEMS Sensor

The MEMS sensor is modeled with the rotational stiffness and capacitive pickoff non-linearity that occur when the proof mass is rotated. Simulations using the MEMS sensor model cannot assume that these non-linearities are eliminated; instead, the simulations must show that the acceleration readout from the electronics is accurate
Despite the non-linearities. However, as long as the feedback loop is implemented properly and the bandwidth constraints on the input are properly designed, the position of the proof mass will stay near 0 and the assumptions on the scale factor and torque constant will hold true. Detailed models of various MEMS sensors have been developed by Draper Laboratory engineers and the design of such models is outside the scope of this thesis.

2.2 Electronics Requirements

There are several design requirements that every electronics architecture for the accelerometer must obey. For the control loop to close and rebalance the proof mass, the system must be linear time-invariant. The square relationship between the torque input is not a linear relationship; this must be linearized in some way. The preload architecture is able to linearize the system, which will be discussed further in Section 2.3.1.

The sensor output is amplitude modulated to the square wave placed on the sense plates. The electronics must demodulate this signal in order to obtain the proof mass position. The digital electronics must also be capable of reading out the acceleration estimate to an external microprocessor or computer.

While there are no strict requirements on hardware size or power consumption, the minimization of both these characteristics is preferred. An efficient design in terms of hardware and power can be more easily adapted for use in an ASIC. A lower power design will extend the operational lifetime of the electronics system when running on battery power. In addition, the lack of wasteful electronics makes the system simpler and easier to debug.

2.3 Architecture Design

Any architecture that provides feedback for a force rebalanced MEMS accelerometer must be capable of linearizing the feedback loop. In the control loop, the only non-
linear relationship is the square law between the torque voltage and the sensor output voltage. There are a variety of ways to linearize this relationship. The preload architecture has been developed as one way to create a linear feedback loop.

2.3.1 Preload Architecture Concept

With zero input acceleration, the preload architecture uses a bias voltage $V_B$ as the amplitude of each torquer square wave. With a non zero input acceleration, a voltage $v_a$ proportional to the readout acceleration is added to one torquer and subtracted from the other torquer. If the left torquer amplitude is $V_L$ and the right torquer amplitude is $V_R$, they can be defined as follows.

\[ V_L = V_B + v_a \] (2.1)

\[ V_R = V_B - v_a \] (2.2)

Thus, the total rebalanced acceleration can be computed as a function of these voltages and the torque constant $\tau$.

\[ a = \tau_L V_L^2 - \tau_R V_R^2 = \tau_L (V_B^2 + 2V_B v_a + v_a^2) - \tau_R (V_B^2 - 2V_B v_a + v_a^2) \] (2.3)

If $\tau_R$ and $\tau_L$ are equal, Equation 2.3 can be simplified as follows.

\[ a = 4\tau V_B v_a \] (2.4)

Even if $\tau_R$ and $\tau_L$ are not exactly matched, this can be compensated by sending different $V_B$ values to each torquer as well as having different gains on the two $v_a$ signals coming out of the processing. Thus, the control loop has been linearized and a feedback controller can be used to restore the proof mass to its null position. In addition, the $v_a$ value that is used to restore the proof mass is proportional to the input acceleration and can be read out to an external processor or computer.

Recalling the scheme depicted in Figure 2-5, a similar analysis can be performed.
Variable | Nominal Value
--- | ---
$V_B$ | 2.936V
$v_a$ | 0.2936V

Table 2.2: Nominal Voltages for Preload Architecture

Figure 2-7: Preload Scheme to Rebalance Sensor

for the preload architecture. In the preload scheme, both torquers are simultaneously active so there is more power consumed in the torquers. To minimize this power consumption, $V_B$ should be minimized while satisfying the design constraints listed in 1.1. Thus, at a +10g or a -10g input acceleration, one torquer should be completely off. Using this realization allows us to solve for nominal values of $V_B$ and $v_a$. These values are listed in Table 2.2.

Using the nominal design settings for the preload architecture, we can construct a graph similar to Figure 2-5 that shows the needed DC torque voltages to rebalance each possible acceleration in the input range. This graph for the preload architecture is shown in Figure 2-7. In fact, the nominal voltage for $V_B$ is a minimum value. If the DC torque voltage were to somehow go negative, this would destroy the linearity
of the preload architecture. The square law of the sensor guarantees that only the absolute value of the torque voltage matters. In addition, the torque plates actually receive a square wave modulated voltage instead of a DC voltage. So the supposed negative DC voltage torque would actually end up being an unintended positive value.

Figure 2-7 shows the input and output characteristics for a perfectly rebalanced sensor. The major difference between this graph and Figure 2-5 is that in the preload scheme, the torque magnitudes are always a linear function of acceleration. By contrast, the scheme where only one torquer is active has a square root relationship with the input acceleration.

### 2.3.2 Alternative Architecture

An alternative design to the preload architecture is an electronics system designed to rebalance the sensor using the scheme shown in Figure 2-5. In this scheme, either the right or left torquer is activated depending on whether the input acceleration is positive or negative. The inactive torquer receives a 0V signal.

A control loop can be designed to rebalance the sensor so that the proof mass remains in the null position. Since there is an inherent square law in the sensor, the control loop must contain a square root function in order to linearize the feedback path. In addition, there must be a sign selector bit that determines whether the left or right torquer is active.

The requirements of the alternative architecture’s control loop present several disadvantages. Performing a digital square root computation is relatively wasteful and impractical compared to the strictly linear operations needed for the preload architecture’s control loop. An algorithm such as Dijkstra’s square root algorithm in digital logic requires multiple clock cycles to converge to a solution and requires a relatively large area of hardware [27]. Multiple clock cycles of processing time is disadvantageous since this increases the latency of the control loop. A control loop with more latency has a higher time constant and is slower to respond to perturbations, thus leading to decreased accuracy in the system. The feedback system relies on movements in the proof mass quickly being compensated; a slow feedback time constant
means that the non-linearities cannot be ignored since the proof mass will be slow to move back to the null position. A large area of logic is also a problem. The system will eventually be migrated to an ASIC and extra space results in added cost and added power consumption.

As mentioned above, a sign selector bit is necessary in the alternative architecture to indicate which torque plates are active. This sign bit causes problems, especially around 0g. Around 0g, it is critical that the sign selector bit is perfectly synchronized with the torque magnitude; otherwise, the system will be attempting to rebalance in the wrong direction. While the feedback loop will ensure that the system remains stable, this will still lead to inaccurate measurements around 0g since it is impossible to perfectly synchronize these signals independent of temperature. In addition, at exactly 0g, the sign bit will show up as a white noise source since external noise will cause the indicated acceleration to flicker between positive and negative values. This white noise sign bit causes capacitive coupling problems with other signals, further reducing the system’s accuracy at 0g.

The preload architecture is able to eliminate the problems of both the digital square root and the sign selector bit. While it suffers from its own drawbacks (notably, higher torque power consumption), it does not suffer from accuracy problems around 0g. It is capable of a lower latency, since it does not have any iterative algorithms in the control loop. Finally, the preload architecture should be able to have a smaller physical size due to a reduced amount of logic.

2.4 Analog & Digital Subsystems

Figure 2-8 shows a representation of the electronics design needed for the preload architecture. Coming out of the sensor, the sense gain and anti-aliasing filter are built into the accelerometer interfacing ASIC, so they are outside the scope of this thesis. Every other component shown in this diagram (with the exception of the sensor) is designed and implemented.

Coming out of the sensor, the amplitude modulated signal is gain boosted and then
undergoes an anti-aliasing filter in preparation for the digital sampling. A sigma-delta modulator turns the analog signal into a 1-bit digital signal that can be processed by the digital signal processing. Next, a digital demodulator turns the amplitude modulated signal into a baseband value. A 4th order CIC decimation filter reduces the sample rate as well as performs low-pass and notch filtering, which will be further explained in Section 2.5. This signal feeds into a PI controller, which is responsible for providing feedback signals to keep the proof mass in the null position. The output of the PI controller is taken and decimated further for readout. Since the control loop has been linearized, the readout could actually have been taken from several places in the loop where the signal is proportional to $v_a$.

After the controller, the signal also goes into a 2nd order CIC interpolation filter that increases the sample rate. The next block consists of math operations to perform the preload architecture operations described in Equations 2.2 and 2.1. After the two torque voltages are calculated, they go into two separate sigma-delta modulators that oversample the digital signal to push the quantization noise out of band.

After the torque signal sigma-delta modulators, analog circuitry is needed to create
the proper torque voltages. An analog low pass filter eliminates the modulation of the sigma-delta modulator and creates a baseband signal. Next, a regulator adjusts the digital voltage level to the proper analog levels. The signals must then be amplitude modulated to the torque commutation frequency to avoid charge buildup on the sensor. The slew rate of the commutated torque signals must be limited with another low pass filter to match slew rates on all torque signals. A final inversion is necessary on some torque signals before they enter the sensor.

2.5 Frequency and Sample Rate Selection

<table>
<thead>
<tr>
<th>Signal</th>
<th>Nominal Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sense Generator</td>
<td>20KHz</td>
</tr>
<tr>
<td>Digital Input Sample Rate</td>
<td>5.12MHz</td>
</tr>
<tr>
<td>Digital Processing Rate</td>
<td>20KHz</td>
</tr>
<tr>
<td>Output Sample Rate</td>
<td>625Hz</td>
</tr>
<tr>
<td>Digital Output Sample Rate</td>
<td>5.12MHz</td>
</tr>
<tr>
<td>Torque Commutator</td>
<td>2.5KHz</td>
</tr>
</tbody>
</table>

Table 2.3: Nominal Frequencies and Sample Rates for Preload Architecture

There are various analog and digital signals in the preload architecture that require frequency and/or sample rate selection. They must be carefully picked to avoid interfering with each other while insuring proper operation of the sensor. Nominal values for these frequencies and sample rates are shown in Table 2.3.
A sense generator frequency of 20KHz is used for this specific accelerometer due to the physical characteristics of the sensor. With this sense generator frequency, the proof mass position is amplitude modulated to 20KHz. Assuming the only spectral content of the sensor output is at 20KHz, then a digital input sample rate of 40KHz would be necessary to sample the signal. However, the sense generator is a square wave and thus the input signal has many harmonics that will cause aliasing in the spectrum of interest if the input sample rate is set too low. The anti-aliasing filter built into the accelerometer interface ASIC has its cutoff frequency at 2MHz, so it cannot be counted on to eliminate these harmonics.

The spectrum of an ideal 20KHz square wave is depicted in Figure 2-9. Harmonics are present on all odd multiples of the carrier frequencies. The magnitude of the harmonic goes down inversely with the multiple. For example, the magnitude of the harmonic at the 7th multiple of the carrier frequency is $\frac{1}{7}$ the magnitude of the carrier frequency. Thus, if we choose 5.12MHz to be the sample rate, the harmonic at the Nyquist frequency of 2.56MHz has a magnitude of $\frac{20\text{KHz}}{2.56\text{MHz}} = 7.81 \times 10^{-3}$ assuming the carrier has a magnitude of 1. This represents at least 42.14dB of attenuation (referenced to the carrier) on the harmonics that will be folded down into the spectrum of interest. However, the output signal of the sensor is not an ideal square wave so the attenuation at higher frequencies is even higher than this. In addition, the interface ASIC contains an anti-aliasing filter with a cutoff at 2MHz, so the aliasing is negligible.

The bulk of the digital processing occurs at a lower sample rate than the input sample rate to the digital subsystem. Once the signal comes in, digital signal processing can be performed to avoid aliasing and corruption effects from analog noise, capacitive coupling, and the expected harmonics from the square wave. The signal does not need to be processed at 5.12MHz, especially with an input bandwidth limit of 100Hz. Although it could be processed at this speed, such processing would be very wasteful on power, since many more operations than necessary would be performed. Instead, the main signal processing is performed at 20KHz. This frequency is much more than necessary to fully represent the 100Hz range of data. However, the signal processing should not be set too low; otherwise, the latency of the loop would increase
and have negative effects on the feedback response. In addition, the ratio of 5.12MHz to 20KHz has convenient aspects in regard to the CIC decimation filter, which will be further explained in Section 3.3.

The output sample rate refers to the signal that is transmitted to a computer. 625Hz was chosen for this sample rate because it is capable of fully representing the 100Hz input bandwidth. This signal is not in the feedback path, so latency is not a concern with this relatively slow sample rate. 625Hz also divides into 20KHz by a power of 2, allowing efficient decimation. Finally, 625Hz is sufficiently slow that 64-bit data could be transmitted into a PC’s serial port at that rate.

The output was interpolated back up to 5.12MHz in preparation of entering the D/A sigma-delta modulator. 5.12MHz was chosen as a sufficiently high oversampling ratio for the sigma-delta modulators, pushing the noise out of band.

The torque commutator frequency had to be chosen to not interfere with the sense generator frequency. Due to the proximity of the sense generator plates and torque plates in the MEMS accelerometer, there is substantial coupling between the torque plates and the charge pickoff on the sense plates. Thus, a 2.5KHz square wave is used for the torque commutation frequency. This square wave only has odd harmonics, so it does not contain a harmonic at 20KHz. The harmonics that appear at 17.5KHz and 22.5KHz are a concern, but can be filtered out if they pose a problem. In addition, 2.5KHz for the torque commutators is fast enough to avoid the charge buildup problem where charge accumulates on the proof mass.

### 2.6 Functional Simulation

The preload architecture was simulated at varying levels of complexity. Before detailed models of the digital components were constructed, a system level simulation was first constructed. In this system level simulation, the details regarding the inner workings were abstracted into functional blocks. A Simulink model for this functional simulation is shown in Figure 2-10. The purpose of this simulation model is to verify that the preload architecture can properly rebalance the accelerometer and properly
read out the input acceleration.

In this model, many of the implementation details have been eliminated in an attempt to create a working closed loop simulation to prove that the preload architecture works in concept. There is no discrete time processing in this model. Instead, everything is processed in continuous time. The CIC decimation filter has been replaced with a continuous time low pass filter, since there are no sample rates to convert. The digital PI controller is replaced with a gain and an integrator. The interpolation filter and sigma-delta modulators are completely eliminated since they have no meaning in a continuous time simulation.

The simulation results from this functional model are shown in Figure 2-11. The input waveform is a 10g amplitude sine wave at 100Hz. The output readout has some noticeable latency and attenuation; both of these factors are expected due to the transfer characteristics of the control loop.
Figure 2-11: Functional Model Simulation Results
Chapter 3

Digital Design

The majority of the signal processing for the preload architecture is performed in the digital domain. Generally, performing computations in the digital domain is beneficial when there is a large amount of data that needs to be processed quickly and accurately [2]. Digital circuitry is ideal for this application because it is resistant to electrical noise and temperature effects as well as being robust to process variations during manufacturing [9].

This chapter first provides an overview of the digital system and explains the required function of each component block. Next, it goes into detail on the design and implementation of each block. Finally, a synthesis analysis is presented, comparing the implementation size of the various components.

3.1 Digital Overview

An overview of the digital subsystem is shown in Figure 3-1. In this diagram, the analog blocks have been compressed into abstract subsystems while the digital blocks are shown in relative detail. The signal coming in from the analog input stage is a sigma-delta modulated signal at 5.12MHz. This is a 1-bit signal with a sample rate of 5.12MHz, and it represents the sensor signal amplitude amplitude modulated at 20KHz. Thus, the signal needs to be demodulated to move the signal to baseband.

The compensation stage runs at a sample rate of 20KHz, and a CIC decimation
filter is used to reduce the sample rate. Separate from the feedback path, another CIC decimation filter is used to further decrease the sample rate to 625Hz so that it can be transmitted to a computer. Before being transmitted, this data is packaged using a UART transmitter so that it can be received by the software running on the computer. In the feedback path after the digital compensator, the signal is interpolated back up to 5.12MHz. Several math operations are performed on the signal in order to compute the proper rebalancing voltages for the torque plates. These math operations comprise the core operation of the preload architecture and are represented in Equations 2.1 and 2.2. Finally, the signal goes through a D/A sigma-delta modulator to transmit the torque amplitudes back into the analog domain.

All signals in the DSP path are represented as two’s complement fixed point values between -1 and +1. Thus, the most significant bit (MSB) is the sign bit and all other bits represent the fractional value of the signal. Fixed point representations in the feedback loop have the advantage that they result in smaller and simpler hardware than the analogous floating point logic [4]. Using two’s complement fixed point representations, the precision of each register can easily be controlled by simply adding more least significant bits to the register.
3.2 Demodulator

The first digital stage in the preload architecture is the demodulator. The demodulator is necessary because the output of the sensor is amplitude modulated to the sense generator signal, which is nominally 20KHz. The demodulation can be performed by multiplying the incoming signal by a 20KHz sine wave perfectly in phase with the sense generator square wave. In actuality, the signal coming in from the sigma-delta modulator is slightly out of phase with the demodulation reference due to analog delays. This discrepancy manifests itself as a constant phase delay and thus a constant gain error (with absolute value less than 1) on the demodulated output. The phase difference was experimentally measured to be small, so the gain error was negligible.

There are several ways to generate the digital sine values necessary for the demodulation. One of the simplest and most effective methods is by table lookup [16]. In this method, the values of the sine are precomputed and stored in a ROM and referenced by an index. The drawbacks to this method is that the precomputed values must be rounded and are only available for quantized time instances. In the preload architecture, this is not a problem because the sample rate is fixed and the precision of the sine values can be arbitrarily increased as necessary by increasing the bits for each sample.

As an optimization of ROM size at the expense of more logic, only the first quadrant of the sine wave is stored in the ROM. Using this information, the value of the entire sine wave can be determined at any phase value. Another optimization is that the sign bit of the sine wave is not stored in the ROM. Since only the first quadrant of the sine wave is stored, the sign bit is known to be always 0, indicating a non-negative number. Simple logic can be used to determine the proper sign bit and values in the other quadrants of the sine wave. The demodulator functions as a digital multiplexer. When the sigma-delta modulated input is high, the output is the current value of the demodulator reference. When the sigma-delta modulated input is low, the output is the inverted value of the demodulator reference.

A 20KHz sine wave sampled at 5.12MHz has 256 samples. As mentioned above,
only 64 of these samples need to be stored in a ROM. The demodulator needs only one state element: a counter that goes from 0 to 256. This counter refers to the current sample of the sine wave. When the counter is below 64, it is a pointer directly into the sine ROM. However, when the counter increments above 64, additional logic is needed to compute the proper demodulator output.

Figure 3-2 illustrates the 4 different modes that the demodulator operates in. Only the values in region A are stored in the sine ROM. When the sense generator (which is also controlled by the digital subsystem) makes a low to high transition, a signal is sent to the demodulator that the counter needs to be reset to 0. While the demodulator is in region A, the counter functions as the index for the sine ROM. In region B, the sine ROM must be traversed in reverse order. In region C, the sine ROM is traversed in forward order but the value must be negated. In region D, the sine ROM is traversed backwards and the value is also negated. These optimizations save on register space at the cost of additional logic.

The demodulator actually performs no multiplies, even though a multiplication is theoretically necessary on every input sample. This is because the input to the
system is a sigma-delta modulated 1-bit input that only has values of +1 and -1. So, the corresponding sine value is computed as well as the negative of this value. If the sigma-delta modulated input is +1, then the sine value is connected directly to the demodulator output. If the sigma-delta modulated input is -1, then the negative of the sine value is connected to the demodulator output. The inverse of a two’s complement number can be obtained by inverting all the bits and adding one [5]. A diagram of this setup is shown in Figure 3-3.

### 3.3 CIC Decimation Filter

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downsampling Ratio</td>
<td>R</td>
<td>256</td>
</tr>
<tr>
<td>Delays per Comb Stage</td>
<td>M</td>
<td>1</td>
</tr>
<tr>
<td>Stages</td>
<td>N</td>
<td>4</td>
</tr>
<tr>
<td>High Sample Rate</td>
<td>$f_s$</td>
<td>5.12MHz</td>
</tr>
<tr>
<td>Low Sample Rate</td>
<td>$f_r$</td>
<td>20KHz</td>
</tr>
<tr>
<td>Input Bitwidth</td>
<td>$B_{in}$</td>
<td>24</td>
</tr>
<tr>
<td>Output Bitwidth</td>
<td>$B_{out}$</td>
<td>32</td>
</tr>
</tbody>
</table>

Table 3.1: CIC Decimation Filter Parameters

The CIC decimation filter functions as an integrated low pass filter, notch filter, and downsampler. A detailed description of CIC filter design and theory is given in [12]. The CIC decimation filter in the preload system has a decimation ratio of 256.
going from a high sample rate of 5.12MHz to a low sample rate of 20KHz. These values and other parameters are given in Table 3.1.

The CIC decimation filter consists of \( N \) cascaded integrator stages followed by \( N \) cascaded comb stages. Each comb stage has a delay of \( M \) samples. An example of an integrator stage can be seen in Figure 3-4 and a comb stage with \( M=1 \) can be seen in Figure 3-5. Between the integrator and comb sections, there is a sample and hold block that runs at the lower sample rate.

The system functions for an integrator stage and a comb stage are given in Equations 3.1 and 3.2, respectively. Note that these system functions are both referenced to the high sample rate \( f_s \), which in this case is 5.12MHz. Using these stage transfer functions, an overall system function can be derived, since the CIC decimation filter consists only of these linear time-invariant blocks cascaded together. Thus, the overall system function is simply the product of all the block system functions together [20]. The CIC decimator filter’s system function in terms of parameters is given in Equation 3.3.

\[
H_I(z) = \frac{1}{1 - z^{-1}} \quad (3.1)
\]
Figure 3-6: Frequency Response for CIC Decimation Filter

\[ H_C(z) = 1 - z^{-RM} \]  \hspace{2cm} (3.2)

\[ H(z) = \frac{(1 - z^{-RM})^N}{(1 - z^{-1})^N} \]  \hspace{2cm} (3.3)

Plugging in the values from 3.1, a transfer function for the preload CIC decimation filter can be calculated. The frequency response for this transfer function is shown in Figure 3-6. A 4th order filter was chosen to provide enough roll-off to sufficiently attenuate certain frequencies. A lower order filter does not roll off as steeply, and thus will allow more high frequency noise to pass through.

At this stage in the preload architecture, the bandwidth of interest is <100Hz, so we would like to efficiently eliminate any frequencies above this value. However, the signal is already bandlimited and a few specific noise sources can be explicitly eliminated by the CIC decimation filter. The drawback of a higher order CIC filter is that the latency of the filter increases. As seen in Figure 3-6, the filter has linear phase for all frequencies, no matter the parameters of the filter. However, the group
delay of the filter increases with an increasing filter order. Ideally, we would like as short a delay as possible to decrease the loop latency and improve the feedback response.

Due to the parameters chosen for the CIC decimation filter, there is a notch in the frequency response exactly at 20KHz. Recalling the frequency selection from Table 2.3, 20KHz is the sense generator frequency, which is thus the modulation frequency of the sensor output. This is important because the analog electronics after the sensor add in some DC offsets due to imperfections in the electronics. The demodulator then modulates these DC components to 20KHz in the demodulation process. The CIC decimation filter can then filter out these undesired noise sources.

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Noise Source</th>
<th>Attenuation</th>
</tr>
</thead>
<tbody>
<tr>
<td>17.5KHz</td>
<td>Torque Feedthrough</td>
<td>-68.49dB</td>
</tr>
<tr>
<td>20KHz</td>
<td>Analog DC Offset</td>
<td>-344.1dB</td>
</tr>
<tr>
<td>22.5KHz</td>
<td>Torque Feedthrough</td>
<td>-77.4dB</td>
</tr>
</tbody>
</table>

Table 3.2: CIC Decimation Filter Attenuation at Selected Frequencies

In the MEMS sensor, capacitive coupling between the sense plates and the torque plates can cause a 2.5KHz spur to appear on the output of the sensor. This is because 2.5KHz is the modulation frequency of the torque commutator, as shown in Table 2.3. This effect is referred to as torque feedthrough. After the digital demodulation, the torque feedthrough frequency components are up-modulated to 17.5KHz and 22.5KHz. Although the square wave also produces harmonics at odd multiples of the 2.5KHz spur, these are ignored since their magnitude is negligible in comparison to other noise sources. At 17.5KHz and 22.5KHz, the attenuation of the CIC decimation filter is still high enough to eliminate the effect of the torque feedthrough. Table 3.2 shows the attenuation at these frequencies.

There are several alternate implementations that can perform the function of the CIC decimation filter. For example, the low pass filtering, anti-alias filtering, and notch filtering functions can be performed by a traditional FIR or IIR filter. The decimation can then be performed by sampling the filtered signal. The CIC implementation, however, has the advantage that it requires no multiplications.
<table>
<thead>
<tr>
<th>Register</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>24</td>
</tr>
<tr>
<td>Integrator 1</td>
<td>56</td>
</tr>
<tr>
<td>Integrator 2</td>
<td>54</td>
</tr>
<tr>
<td>Integrator 3</td>
<td>47</td>
</tr>
<tr>
<td>Integrator 4</td>
<td>40</td>
</tr>
<tr>
<td>Comb 1</td>
<td>37</td>
</tr>
<tr>
<td>Comb 2</td>
<td>36</td>
</tr>
<tr>
<td>Comb 3</td>
<td>35</td>
</tr>
<tr>
<td>Comb 4</td>
<td>34</td>
</tr>
<tr>
<td>Output</td>
<td>32</td>
</tr>
</tbody>
</table>

Table 3.3: Register Sizes for CIC Decimation Filter

The CIC decimator is designed to use two’s complement arithmetic and allows for designed “wrap-around” of adders instead of overflowing or saturating registers [12]. There are \(4N = 8\) registers, and they must all be sized correctly so that the filter remains stable and with an error less than 1 LSB. The formulas for determining register sizes are given in [12] and result in the sizes shown in Table 3.3. Truncation can be used at every step except for the last comb’s output going into the output register, which must be rounded to ensure a zero mean error on the solution [12].

A block diagram for the CIC decimation filter is shown in Figure 3-7. There are 4 integration blocks, followed by a downsampler, followed by 4 comb blocks. The integrators run at the fast sample rate while the combs run at the low sample rate. The latency of the loop is essentially determined by the slow sample rate, since each fast delay is 256 times longer than a slow sample rate delay. Thus, the total latency of the CIC decimation filter is 4 slow samples, since each comb contributes 1 sample delay at the low sample rate.

The CIC decimation filter is able to efficiently filter and decimate the signal in a hardware-efficient step with relatively low latency. Reducing hardware size is impor-
tant to save on power consumption and chip area. The low latency provided by the CIC decimation filter is necessary to provide for a fast feedback loop response. For these reasons, the CIC decimation filter is well suited for the preload architecture.

3.4 PI Controller

The PI controller is necessary to provide the feedback compensation to close the feedback loop. In order to determine the type of feedback controller necessary, the preload architecture was simulated at a functional level, as shown in Figure 2-10. First, the simulation was run with just a proportional term. However, this resulted in an unstable system. After adding an integral term, the feedback system became stable. In addition, the integral term ensures that the system had zero steady-state error [13]. A zero steady-state error is important because a non-zero error implies that the proof mass is not properly restored to the null position. If the proof mass is not restored perfectly to the null position, this results in non-linear effects which decrease the accuracy of the accelerometer system. In the functional simulation, the proportional term of the compensator is simply a gain and the integrator is an analog integrator.

The functional simulation indicated that a PI controller was necessary to close the loop. The next step was to test to ensure that a PI controller could properly fulfill the bandwidth requirements outlined in Table 1.1. To do this, the detailed
simulation model was run with a PI controller as shown in Figure 3-8. In this figure, \( K_i \) represents the integral coefficient and \( K_p \) represents the proportional coefficient.

As expected, the detailed simulation model produces in a stable feedback system. An improperly tuned system will exhibit poor bandwidth and transient characteristics and may even be unstable. Adjusting the values of \( K_i \) and \( K_p \) results in varying responses to the system input. These constants can be tuned by applying a step input to the system until the response looks relatively fast and smooth with no oscillations. Due to the presence of the integrator, the steady state error on the feedback system is zero. A pair of properly tuned \( K_p \) and \( K_i \) results in a step response as shown in Figure 3-9. Optimizing the step response is not extremely crucial in this case, because the implemented system is likely going to differ from the simulation and require retuning. Nevertheless, it is useful to check the system response to ensure that the design requirements are still reasonable.

<table>
<thead>
<tr>
<th>Coefficient</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( K_p )</td>
<td>0.2</td>
</tr>
<tr>
<td>( K_i )</td>
<td>0.15</td>
</tr>
</tbody>
</table>

Table 3.4: Tuned \( K_i \) and \( K_p \) From Step Response of Detailed Simulation

Using the coefficients calculated before and summarized in Table 3.4, the bandwidth and attenuation requirements listed in Table 1.1 can be checked. As the input
frequency increases, the attenuation of the system also increases. So, the worst case attenuation occurs with an input of 100Hz. The detailed simulation was run with an input of a 10g amplitude sine wave running at 100Hz. The results from this simulation are shown in Figure 3-10. Here, the attenuation is only -0.2646dB, which is well within the 3dB specification. Of course, this simulation might not exactly match the implemented system, but still indicates that the specification will likely be met.

### 3.5 CIC Interpolation Filter

The CIC interpolation filter is a hardware efficient method for performing interpolation [12]. The simplest traditional method for performing interpolation is to up-sample, stuff zeros in between samples, low pass filter the result, and pass the final signal through a gain [19]. The problem with this method is that it require a low pass filter, traditionally implemented with an FIR or IIR filter. Such a filter requires numerous multiplications in order to get a rolloff steep enough to be suitable for this application.

The CIC interpolation filter is able to perform the same function as the traditional interpolation technique without any multiplications [12]. The structure of the CIC interpolation filter is similar to that of the CIC decimation filter except that the comb
stages and the integrator stages are reversed. This structure is shown in Figure 3-11. A second order filter is sufficient to properly interpolate the signal. A higher order interpolator is not necessary because the data is already known to be bandlimited within 100Hz.

![Block Diagram for CIC Interpolation Filter](image)

**Figure 3-11: Block Diagram for CIC Interpolation Filter**

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upsampling Ratio</td>
<td>R</td>
<td>256</td>
</tr>
<tr>
<td>Delays per Comb Stage</td>
<td>M</td>
<td>1</td>
</tr>
<tr>
<td>Stages</td>
<td>N</td>
<td>2</td>
</tr>
<tr>
<td>High Sample Rate</td>
<td>$f_s \cdot R$</td>
<td>5.12MHz</td>
</tr>
<tr>
<td>Low Sample Rate</td>
<td>$f_s$</td>
<td>20KHz</td>
</tr>
<tr>
<td>Input Bitwidth</td>
<td>$B_{in}$</td>
<td>32</td>
</tr>
<tr>
<td>Output Bitwidth</td>
<td>$B_{out}$</td>
<td>32</td>
</tr>
</tbody>
</table>

**Table 3.5: CIC Interpolation Filter Parameters**

The parameters for the CIC interpolation filter are shown in Table 3.5. The signal is interpolated up to 5.12MHz so that the subsequent sigma-delta modulator can run at this frequency. This high sample rate was chosen so that a relatively non-steep analog filter can low pass filter the digital output. While a lower sample rate could be used for the output, this would require a much steeper filter.

<table>
<thead>
<tr>
<th>Register</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>32</td>
</tr>
<tr>
<td>Comb 1</td>
<td>34</td>
</tr>
<tr>
<td>Comb 2</td>
<td>34</td>
</tr>
<tr>
<td>Integrator 1</td>
<td>34</td>
</tr>
<tr>
<td>Integrator 2</td>
<td>48</td>
</tr>
<tr>
<td>Output</td>
<td>32</td>
</tr>
</tbody>
</table>

**Table 3.6: Register Sizes for CIC Interpolation Filter**

The register sizes were computed using the techniques shown in [12] and are shown
in Table 3.6. The CIC interpolator has no rounding or truncation due to the sizing of the registers except for at the output stage. The output of the CIC interpolator rounds the final integrator output in order to keep the error zero mean.

### 3.6 Torque Adjustment Module

The torque adjustment module converts the acceleration reading into digital signals proportional to the torque necessary to rebalance the sensor. Up to this point, the signal being processed in the control loop is proportional to $v_a$, the acceleration. At this point, we need to compute $V_L$ and $V_R$. These signals are produced using the following equations:

\[ V_L = V_B + \kappa_L v_a \]  
\[ V_R = V_B - \kappa_R v_a \]  

These equations are identical to Equations 2.1 and 2.2 except that these have the $\kappa$ terms, which represent individual gains for the two channels. The $\kappa_L$ and $\kappa_R$ terms allow for scaling the acceleration estimate to take full range of the digital electronics. Utilizing the full range of the digital representation maximizes the precision of the digital representations.

An alternate design would have been to place the torque adjustment before the CIC interpolation filter. This design would have had power savings by virtue of the fact that the multiplications and additions would be occurring at the 20KHz sample rate rather than at the fast 5.12MHz sample rate. However, this design has the drawback that now two CIC interpolation filters are needed, doubling the amount of registers required. So, the torque adjustment module is placed at the output of the single CIC interpolation filter in order to save on hardware size by eliminating unnecessary storage registers.

The output of the torque adjustment module saturates at 0 and +1. These outputs
represent $V_L$ and $V_R$, and it would not make sense for these voltages to be negative. A negative voltage does not make sense because the analog torque commutator will amplitude modulate these torque voltages onto a square wave, so a negative sign would be meaningless. In addition, the inherent square law in the MEMS accelerometer prohibits a negative torque from being applied to the proof mass. Thus, an explicit saturation is applied to the output of the torque adjustment module to prevent overflow and underflow in the digital output. Scaling is performed throughout the system to prevent these cases from occurring, but this extra overflow and underflow protection keeps the system stable in case of input overflow or system error.

### 3.7 D/A Sigma-Delta Modulator

A sigma-delta modulator with a 1-bit DAC was chosen to convert the digital torque readings into an analog output. The sigma-delta modulator takes the digital input\(^1\) and converts it into a serial digital stream of zeros and ones. This stream can then be low pass filtered in order to obtain the analog equivalent of the original digital signal. The sigma-delta modulator has several advantages for this application. An accurate one-bit DAC is cheap and accurate, while a multi-bit DAC is generally slower, less accurate, and more expensive. In addition, the sigma-delta modulator is able to shape the error noise spectrum so that the total noise power can be reduced using an analog lowpass filter [3].

<table>
<thead>
<tr>
<th>Digital Number</th>
<th>Density of Ones</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>25%</td>
</tr>
<tr>
<td>0.5</td>
<td>50%</td>
</tr>
<tr>
<td>1</td>
<td>75%</td>
</tr>
</tbody>
</table>

Table 3.7: Density of Ones for Sigma-Delta Modulated Outputs

Table 3.7 shows the density of ones for various digital inputs. The sigma-delta modulator is designed to reach its limits with inputs of 0 and +1, since these are the smallest and largest inputs the modulator will ever see. However, instead of the 0%\(^1\)This input is between 0 and +1, since it has been limited by the torque adjustment module
and 100% density of ones for these input values, the densities 25% and 75% are used. This is because at densities of ones beyond the range of 25% to 75%, the sigma-delta modulator tends to create a periodic signal rather than a random signal, which is preferred. A periodic signal on the output of the sigma-delta modulator produces unwanted harmonics.

<table>
<thead>
<tr>
<th>SDM Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order</td>
<td>3rd</td>
</tr>
<tr>
<td>Structure</td>
<td>Cascaded Resonator</td>
</tr>
<tr>
<td>Form</td>
<td>Feedback</td>
</tr>
<tr>
<td>Output Range</td>
<td>25% to 75%</td>
</tr>
<tr>
<td>Oversampling Ratio</td>
<td>25,600</td>
</tr>
<tr>
<td>Quantization Levels on Output</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 3.8: Parameters for D/A Sigma-Delta Modulator

The parameters selected for the D/A sigma-delta modulator are listed in Table 3.8. A 3rd order sigma-delta modulator is chosen to push the quantization noise sufficiently out of the band of interest. The cascaded resonator feedback structure makes no difference in the noise transfer function of the sigma-delta modulator versus another type of structure. However, the feedback structure makes it slightly easier to do the scaling necessary to properly set the output density range of the sigma-delta modulator.

3.7.1 Coefficient Generation

The Delta-Sigma Toolbox [25] was used to generate a noise transfer function and a signal transfer function of the sigma-delta modulator, using the parameters shown in Table 3.8. The noise transfer function’s magnitude response is shown in Figure 3-12. This represents the additive quantization noise from the sigma-delta modulator, and the graph shows that this added noise is negligible and is shaped to maximize the signal-to-noise ratio of the signal [22].

With the noise transfer function and signal transfer functions calculated, an actual design can be realized, again using the parameters in Table 3.8 and the Delta-Sigma toolbox. The third-order cascaded resonator feedback form is shown in Figure 3-13.
Figure 3-12: Noise Transfer Function Magnitude Response for D/A Sigma-Delta Modulator

Figure 3-13: Standard Form for Sigma-Delta Modulator
This is the standard form for an odd order cascaded resonator feedback sigma-delta modulator [29]. The coefficients $A$, $B$, $C$, and $D$ in this figure represent the transition vectors in a state space system. These coefficients can be calculated using the Delta-Sigma Toolbox using the given noise transfer function and signal transfer function. In the preload architecture, the signal transfer function is unity for all frequencies; this is possible since filtering has already been done in earlier stages of the digital subsystem to eliminate unwanted harmonics and noise.

<table>
<thead>
<tr>
<th>SDM Coefficient</th>
<th>Computed Value</th>
<th>Rounded Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>$3.326668594709286 \times 10^{-1}$</td>
<td>$0.25$</td>
</tr>
<tr>
<td>$a_2$</td>
<td>$2.209493489729413 \times 10^{-1}$</td>
<td>$0.25$</td>
</tr>
<tr>
<td>$a_3$</td>
<td>$1.765587264838553 \times 10^{-1}$</td>
<td>$0.125$</td>
</tr>
<tr>
<td>$g$</td>
<td>$3.139264230431509 \times 10^{-4}$</td>
<td>$2^{-12}$</td>
</tr>
<tr>
<td>$b_1$</td>
<td>$3.326668594709286 \times 10^{-1}$</td>
<td>$0.25$</td>
</tr>
<tr>
<td>$b_2$</td>
<td>$2.209493489729413 \times 10^{-1}$</td>
<td>$0.25$</td>
</tr>
<tr>
<td>$b_3$</td>
<td>$1.765587264838553 \times 10^{-1}$</td>
<td>$0.125$</td>
</tr>
<tr>
<td>$b_4$</td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td>$c_1$</td>
<td>$1.014194322335891 \times 10^{-1}$</td>
<td>$0.125$</td>
</tr>
<tr>
<td>$c_2$</td>
<td>$2.878260599737154 \times 10^{-1}$</td>
<td>$0.25$</td>
</tr>
<tr>
<td>$c_3$</td>
<td>$4.529836508919259$</td>
<td>$4$</td>
</tr>
</tbody>
</table>

Table 3.9: State Space Coefficients for Sigma-Delta Modulator

Table 3.9 shows the generated coefficients for producing a unity signal transfer function and the noise transfer function shown in Figure 3-12. However, the signal to noise ratio in the region of interest is unnecessarily high and can be sacrificed in order to reduce on the hardware size. Using the computed values in Table 3.9 directly in the standard form of the sigma-delta modulator would require 10 multiplications per output sample.

The method described in [11] shows how a sigma-delta modulator can be designed with zero multiplications. Specifically, the coefficients are rounded to the nearest power of two so that they can be implemented with shifters rather than multipliers. This is much more efficient in hardware and can save in both hardware area and power consumption. The problem is that the poles and zeroes of the sigma-delta modulator move from the rounding and the noise transfer function is no longer as ideal as the one shown in Figure 3-12. However, in this system, there is a sufficiently
high signal to noise ratio that the noise transfer function can be disturbed slightly from the rounded coefficients while not significantly disturbing the noise spectrum in the bandwidth of interest.

3.7.2 Output Density of Ones Scaling

At this point, the D/A sigma-delta modulator has been designed to output a full scale output, using the standard form shown in Figure 3-13. This means that with a digital input of 0, the modulator output will have a density of ones of 0%. Conversely, with a digital input of +1, the modulator output will have a density of ones of 100%. This range of outputs is the traditional range used by sigma-delta modulators. This does not satisfy the design goals stated in Table 3.7 where the output density of ones limits are 25% and 75%.

<table>
<thead>
<tr>
<th>SDM Output</th>
<th>Typical Feedback Value</th>
<th>Preload SDM Feedback Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>-0.5</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1.5</td>
</tr>
</tbody>
</table>

Table 3.10: Feedback Values for Sigma-Delta Modulator

Under typical operation, the sigma-delta modulator feeds back a term depending on what the output of the modulator is. In the standard form, this term is either 0 or +1. However, if the sigma-delta modulator feeds back -0.5 instead of 0 and +1.5 instead of +1, the output range will be effectively changed. Table 3.10 summarizes the changes needed in the feedback values in order to constrain the output range of the sigma-delta modulator. In this case, the output is constrained to 25% and 75% density of ones; exactly the output range that is desired for the D/A sigma-delta modulator. Although the feedback values are now -0.5 and 1.5, the outputs of the sigma-delta modulator are still 0 and +1, and can be outputted on a single bit line and connected to a 1-bit DAC. The feedback values, on the other hand, must still use the fixed point arithmetic as they will be shifted a certain amount depending on the values of a1, a2, and a3.

Widening the range of feedback values effectively constrains the output of the
To PC

Temperature Estimate

Figure 3-14: Component of Output Transmitter

The output transmitter block is not part of the feedback path. Rather, it is a method from which the acceleration estimate can be extracted from the embedded hardware using a computer. The components of the output transmitter are shown in Figure 3-14. A downsampled acceleration estimate is transmitted to the computer as well as a temperature compensation variable. Although the acceleration estimate could have been taken directly from before or after the PI controller, an additional decimation

3.8 Output Transmitter

The output transmitter block is not part of the feedback path. Rather, it is a method from which the acceleration estimate can be extracted from the embedded hardware using a computer. The components of the output transmitter are shown in Figure 3-14. A downsampled acceleration estimate is transmitted to the computer as well as a temperature compensation variable. Although the acceleration estimate could have been taken directly from before or after the PI controller, an additional decimation
filter was added to reduce the rate of data exported from the hardware. The output does not need to be received at a 20KHz sample rate since the bandwidth of interest is <100Hz.

The temperature compensation variable shown in Figure 3-14 is a 1-bit sigma-delta modulated input that feeds directly into the CV CIC decimation filter. Unlike the acceleration input, the compensation variable does not need to be demodulated or low pass filtered, since it is not a modulated signal nor does it have unwanted high frequency noise. The CV CIC decimation filter effectively low pass filters the 5.12MHz signal and produces the proper DC value of the temperature estimate. This temperature estimate is needed in order to experimentally determine how the system performs at different temperatures.

### 3.8.1 Acceleration Output CIC Decimation Filter

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downsampling Ratio</td>
<td>R</td>
<td>32</td>
</tr>
<tr>
<td>Delays per Comb Stage</td>
<td>M</td>
<td>1</td>
</tr>
<tr>
<td>Stages</td>
<td>N</td>
<td>2</td>
</tr>
<tr>
<td>High Sample Rate</td>
<td>$f_s$</td>
<td>20KHz</td>
</tr>
<tr>
<td>Low Sample Rate</td>
<td>$f_L$</td>
<td>625Hz</td>
</tr>
<tr>
<td>Input Bitwidth</td>
<td>$B_{in}$</td>
<td>32</td>
</tr>
<tr>
<td>Output Bitwidth</td>
<td>$B_{out}$</td>
<td>32</td>
</tr>
</tbody>
</table>

Table 3.11: Acceleration Output CIC Decimation Filter Parameters

The acceleration output CIC decimation filter is implemented using the same structure described in [12]. Parameters for this filter are shown in Table 3.11. This filter is merely reducing the sample rate so that the output is at a convenient rate to send to a computer. A simple downsampler could have been used, but the CIC decimation filter provides a little protection in case there are unexpected high frequencies present in the signal. The CIC decimation filter in this case will low pass filter the signal, preventing the higher frequency harmonics from folding down into the <100Hz frequency range.

Table 3.12 shows the register sizes for the acceleration output CIC decimation
The input comes in on 32 parallel lines and also leaves on 32 parallel lines. The data is not serialized until it reaches the UART packager.

### 3.8.2 Compensation Variable CIC Decimation Filter

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downsampling Ratio</td>
<td>R</td>
<td>8192</td>
</tr>
<tr>
<td>Delays per Comb Stage</td>
<td>M</td>
<td>1</td>
</tr>
<tr>
<td>Stages</td>
<td>N</td>
<td>2</td>
</tr>
<tr>
<td>High Sample Rate</td>
<td>$f_s$</td>
<td>5.12MHz</td>
</tr>
<tr>
<td>Low Sample Rate</td>
<td>$f_R$</td>
<td>625Hz</td>
</tr>
<tr>
<td>Input Bitwidth</td>
<td>$B_{in}$</td>
<td>1</td>
</tr>
<tr>
<td>Output Bitwidth</td>
<td>$B_{out}$</td>
<td>24</td>
</tr>
</tbody>
</table>

Table 3.13: Compensation Variable CIC Decimation Filter Parameters

The compensation variable has its own CIC decimation filter, illustrated in Figure 3-14. This is necessary since the compensation variable is stochastically uncorrelated with the acceleration estimate and must be treated as its own signal. The compensation variable CIC decimation filter has different parameters than the acceleration output CIC decimation filter. This is because it is an incoming 1-bit stream sampled at 5.12MHz. The full parameters for the compensation variable CIC decimation filter are shown in Table 3.13.

The register sizes for the CIC decimation filter were generated using the method described in [12] and are shown in Table 3.14. A 24-bit output was chosen because it is a multiple of 8, which will prove convenient for the UART transmission packager, described in Section 3.8.3. This CIC decimation filter is able to take the 1-bit
<table>
<thead>
<tr>
<th>Register</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>1</td>
</tr>
<tr>
<td>Integrator 1</td>
<td>27</td>
</tr>
<tr>
<td>Integrator 2</td>
<td>27</td>
</tr>
<tr>
<td>Comb 1</td>
<td>27</td>
</tr>
<tr>
<td>Comb 2</td>
<td>26</td>
</tr>
<tr>
<td>Output</td>
<td>24</td>
</tr>
</tbody>
</table>

Table 3.14: Register Sizes for Compensation Variable CIC Decimation Filter

Figure 3-15: Example UART Packet

Sigma-delta modulated compensation variable input sampled at $5.12MHz$ and turn it into a 24-bit value sampled at $625Hz$. The outputs of the acceleration estimate CIC decimation filter and the compensation variable CIC decimation filter are synchronized so that the outputs arrive simultaneously. This will prove useful for the UART transmitter.

### 3.8.3 UART

The UART standard is a serial communications protocol that many embedded hardware devices and computer software packages use to communicate with each other [18]. There are several options in defining the specific communications protocol used
in the UART and in the preload system, the communications packet was chosen to use ten total bits: one start bit, 8 data bits, and one stop bit. A packet such as one used in the preload system is shown in Figure 3-15. A parity bit, used for error correction, was omitted because it was not expected to be necessary for the short cable lengths used in the prototype test system.

Between transmitted UART packets, the communications line is held at '1'. In this manner, sequential transmitted packets do not need to be synchronized with each other since the receiver is asynchronously alerted to an incoming packet by a high to low transition. After the stop bit is sent on a byte, the next byte can come immediately or after any amount of delay. This delay does not need to be an integer multiple of the baud rate. The baud rate does need to be agreed upon by the receiver and the embedded electronics acting as the transmitter. This is because there is no clock sent, and the computer samples the transmission line when it estimates by its internal clock that the data should be valid. For the preload system, the baud rate was chosen to be 57600 which would allow data coming in at 56 bits of data to be transmitted to the computer at 625Hz. The total number of transmissible bits if outputs are arriving at 625Hz is shown in the Equation 3.6. 73.7 bits per sample is sufficient to transmit the 56 bits of data that the UART transmitter receives at the 625Hz sample rate.

\[
\frac{57,600 \text{ bits sent}}{\text{sec}} \times \frac{8 \text{ data bits}}{10 \text{ bits sent}} \times \frac{1}{625 \text{Hz}} = \frac{73.7 \text{ bits}}{\text{sample}} \quad (3.6)
\]

A 32-bit acceleration estimate and a 24-bit compensation variable needed to be
transmitted to the computer. These data arrived at the UART transmitter at 625Hz. All these bytes were split into seven separate 8-bit data packets and one 8-bit synchronization packet was added to the beginning, as shown in Figure 3-16. The synch packet is necessary so that the computer knows how to process the incoming data. Without the synch packet, the computer receiver would receive a packet of data and not know which of the 7 possible bytes it had received.

For this synch packet, ‘10101010’ was chosen, which is a value that should be relatively rare to be broadcast as part of the word. Once the synchronization is set, the computer will maintain synchronization and the synch packet will merely be a check to make sure that no packets have been dropped. With the synch packet, this brings the total number of transmitted bits per sample up to 64, which is still under the 73.7 bits per sample limit that was established in Equation 3.6.

A timing generator produces a strobe that is triggered when the data is ready. The transmitter is implemented using a large shift register that can hold all 80 bits that need to be transmitted. These 80 bits consist of the 64 data bits and 16 start and stop bits. When the data ready strobe arrives, the shift register is filled with all the data, the synchronization word, as well as the start bits and stop bits.

To complete the module, there is a separate strobe running at the output baud rate. When the data ready strobe arrives, the baud strobe triggers a process to begin transmitting the shift register LSB at the baud rate. Every time a bit is transmitted, the shift register rotates all of the bits down. A counter is necessary to keep track of when the entire shift register is done sending.

This entire process can be summarized pictorially in Figure 3-17 as a finite state machine. There are 3 states: the idle state, the ‘fill shift register’ state, and the ‘send bit and shift’ state. The output strobe corresponds to data arriving at the UART transmitter, which then must be packaged by filling the shift register, which takes one clock cycle. Next, each bit is sent sequentially until the shift register is empty, in which case the machine re-enters the idle state and waits for the next piece of data. This is a Mealy machine because the shift register is filled with incoming data that eventually becomes an output of the machine [17].
3.9 Synthesis Results

A synthesis was run on all of the digital blocks separately to compare their relative logic sizes as well as their relative usage of registers. The FPGA synthesis tool divides combinational logic into ‘Slices’ and registers into ‘Slice Flip-Flops’. So, the number of slices provides a rough idea of the amount of logic for a certain block, while the slice flip-flops gives a rough indication of the number of registers consumed by that block.

<table>
<thead>
<tr>
<th>Block</th>
<th>Slices</th>
<th>Slice Flip-Flops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Demodulator</td>
<td>324</td>
<td>61</td>
</tr>
<tr>
<td>CIC Decimation Filter</td>
<td>209</td>
<td>371</td>
</tr>
<tr>
<td>PI Controller</td>
<td>930</td>
<td>88</td>
</tr>
<tr>
<td>CIC Interpolation Filter</td>
<td>109</td>
<td>182</td>
</tr>
<tr>
<td>Torque Adjustment Module</td>
<td>871</td>
<td>1,640</td>
</tr>
<tr>
<td>D/A Sigma-Delta Modulator</td>
<td>183</td>
<td>103</td>
</tr>
<tr>
<td>Output CIC Decimation Filter</td>
<td>110</td>
<td>180</td>
</tr>
<tr>
<td>Compensation Variable CIC Decimation Filter</td>
<td>70</td>
<td>131</td>
</tr>
<tr>
<td>UART Transmitter</td>
<td>56</td>
<td>99</td>
</tr>
</tbody>
</table>

Table 3.15: Synthesis Sizes for Digital Blocks

The summary of the synthesis data is shown in Tables 3.15 and 3.16. The PI
controller takes up the most combinational logic, yet only consumes a small number of registers. This is because it contains two combinational multipliers yet few registers to store the data. The sheer amount of logic required by a multiplier helps explain the design decision to use CIC filters instead of FIR or IIR filters. The CIC filters contain no multiplies, and thus save on a significant amount of logic.

The torque adjustment module is large, both in terms of slices and slice flip-flops. The module contains a significant amount of logic, since it has the two κ multipliers as well as the adder and subtractor for the $V_R$ and $V_L$ calculation. This explains the high number of slices, but not the slice flip-flops. The torque adjustment module requires a large number of slice flip-flops because it stores the multiplied value while checking for saturation. Hardware optimizations could be performed to reduce the precision of the values stored in the temporary registers. However, this was not necessary since there was sufficient space in the FPGA to fit the entire system.

<table>
<thead>
<tr>
<th>Block</th>
<th>Slices</th>
<th>Slice Flip-Flops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Demodulator</td>
<td>11%</td>
<td>2%</td>
</tr>
<tr>
<td>CIC Decimation Filter</td>
<td>7%</td>
<td>13%</td>
</tr>
<tr>
<td>PI Controller</td>
<td>33%</td>
<td>3%</td>
</tr>
<tr>
<td>CIC Interpolation Filter</td>
<td>4%</td>
<td>6%</td>
</tr>
<tr>
<td>Torque Adjustment Module</td>
<td>31%</td>
<td>58%</td>
</tr>
<tr>
<td>D/A Sigma-Delta Modulator</td>
<td>6%</td>
<td>4%</td>
</tr>
<tr>
<td>Output CIC Decimation Filter</td>
<td>4%</td>
<td>6%</td>
</tr>
<tr>
<td>Compensation Variable CIC Decimation Filter</td>
<td>2%</td>
<td>5%</td>
</tr>
<tr>
<td>UART Transmitter</td>
<td>2%</td>
<td>3%</td>
</tr>
</tbody>
</table>

Table 3.16: Percent of Total Synthesis Size for Digital Blocks
Chapter 4

Analog Design

This chapter presents the analog design for signals in the feedback path of the preload architecture. These components are labelled as the analog output stage in Figure 4-1. The analog input stage is embedded in the accelerometer interface ASIC and does not need to be designed as part of the preload architecture. The analog components presented in this chapter represent designs in the feedback path between the digital logic and the sensor. Implementation details, part selection, and miscellaneous analog components are discussed in Chapter 5.

Figure 4-1 shows a simplified version of the preload system. All the digital components have been compressed into one block with three signals feeding into the analog subsystem. These three signals are the left and right torque magnitudes, and the torque commutator. The torque magnitudes are sigma-delta modulated signals with a 5.12MHz sample rate. The torque commutator is a 2.5KHz square wave; commutation is necessary to prevent charge buildup on the sensor proof mass.

4.1 Analog Overview

The analog output stage is responsible for preparing the torque signals that go into the accelerometer. Specifically, there are four torque signals, as shown in Figure 4-2. These correspond to the upper and lower plates on the left and right side of the MEMS sensor. Figure 4-2 also shows the analog components in a little more detail.
Figure 4-1: Abstract Analog Subsystem Block Diagram

Figure 4-2: Detailed Analog Subsystem Block Diagram
The torque magnitudes from the digital subsystem must first pass through a level shifter to establish the proper analog voltage levels. Next, a low pass filter is necessary to convert the sigma-delta modulated signal into a baseband analog voltage. After the low pass filter, the signal will be naturally bandlimited to 100Hz, since this will represent the $V_L$ and $V_R$ values presented in Equations 2.1 and 2.2.

The torque commutator takes the baseband signals and amplitude modulates them onto a 2.5KHz square wave. The purpose of this square wave is to prevent charge buildup on the sensor. Note in Figure 4-2 that the left and right torque magnitudes are commutated with opposite phase from each other. In the block diagram, this is represented with an inverter on the left torque magnitude although in reality, it is not implemented with an inverter. The actual implementation is described in Section 4.4. After the torque commutators, the slew-rate limiters filter the two commutated torque signals so that the slew-rate on both channels matches perfectly. This is done with another low pass filter. This signal is ready to feed into the sensor, and each channel is also inverted to feed into the opposite plate. For example, the slew-rate limited left channel torquer can feed into the upper left torque plate, while its inverse can feed into the lower left torque plate.

4.2 Level Shifter

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Digital Output</th>
<th>Level Shifter Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital ‘0’</td>
<td>0V</td>
<td>-4V</td>
</tr>
<tr>
<td>Digital ‘1’</td>
<td>3.3V</td>
<td>12V</td>
</tr>
</tbody>
</table>

Table 4.1: Level Shifter Voltage Levels

The level shifter is responsible for converting between the voltage levels used in the digital subsystem into known precise analog voltage levels. For the preload system prototype, an FPGA with LVTTL outputs was used with outputs of either 0V or 3.3V. The level shifter was implemented using a comparator that compared the digital output to a 1.65V reference. The output of the comparator would either be -4V or +12V, as shown in Table 4.1.
<table>
<thead>
<tr>
<th>Characteristic</th>
<th>SDM Output</th>
<th>Filtered Level Shifter Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Value</td>
<td>25% Density of Ones</td>
<td>0V</td>
</tr>
<tr>
<td>Maximum Value</td>
<td>75% Density of Ones</td>
<td>8V</td>
</tr>
</tbody>
</table>

Table 4.2: Level Shifter Signal Representations

These voltage levels were chosen because the system has power rails of -15V and +15V and these two values result in convenient analog levels after the low pass filter. Specifically, a 25% density of ones results in a low pass filtered analog value of 0V and a 75% density of ones results in a low pass filtered analog value of +8V. Recalling the sigma-delta modulator design, 25% and 75% densities of ones were the minimum and maximum represented values on the output. In addition, +12V is a convenient voltage level because this is the maximum supply voltage that many electrical components run on. These representations are summarized in Table 4.2.

Based on the voltage levels shown in Table 4.1, the value of $V_B$ has been chosen to be +4V. This is because if $v_a$ is 0 (the situation with 0g input), both torque plates will receive square waves with a 4V amplitude, assuming the rebalance bias is set at the digital value of 0.5. The rebalance bias does not need to be set at the digital value of 0.5, but this allows for the maximum amplitude input since it allows for the widest swing of $v_a$ before saturating the torque adjustment module.

With a rebalance bias of 0.5, the highest acceleration the system can rebalance occurs when one torquer is on its minimum value and the other torquer is on its maximum value. In other words, one torque plate has an amplitude of 0V and the other has an amplitude of 8V. The torque constant of $\frac{0.29g}{V^2}$ allows us to compute the maximum acceleration that this system can rebalance:

$$(8V)^2 \cdot \frac{0.29g}{V^2} = 18.6g$$ (4.1)

Thus, the system is capable of rebalancing the 10g input range desired from the system design goals. In fact, the rebalance bias can be reduced to save power, a technique which will be discussed in more detail in Section 5.1.
4.3 Low Pass Filter

The low pass filter turns the sigma-delta modulated signal into a baseband signal. An example power spectrum of the sigma-delta modulated output is shown in Figure 4-3. This graph shows the spectrum from 0Hz to 2.56KHz. The signal power is all contained in the region <100Hz, and all the power in this graph is normalized to the peak signal, set at 0dB. As expected, the sigma-delta modulator has shaped the noise floor so that there is less noise in the band where the signal occurs.

Figure 4-4 shows a zoomed graph of the same power spectrum. In this graph, it is more evident that the signal spectrum is contained within the 100Hz bandwidth. Ideally, a low pass filter would be a box filter and cut off precisely at 100Hz, eliminating all noise above this frequency. However, such a filter is only possible if it has an infinite filter order, which means that it has an infinite number of components.

An analog filter was designed with $f_{3dB} = 2KHz$. For the filter topology, an active 4th order Sallen-Key Butterworth filter was chosen and synthesized using the FilterLab filter design software [26]. This generated the circuit shown in Figure 4-5.
Figure 4-4: Zoomed Power Spectrum for Sigma-Delta Modulator Output

Figure 4-5: Analog Subsystem Active 4th Order Sallen-Key Butterworth Low Pass Filter Circuit
The resistors were chosen so that they satisfied 1% standard values and could thus be easily built on a test board. The magnitude response for this low pass filter is shown in Figure 4-6, and satisfies the maximally flat shape expected for a Butterworth filter [19]. There is no attenuation up to 100Hz, so the signal will be passed through completely with no loss.

Figure 4-7 shows the power spectrum of the output of the analog low pass filter. Compared to the unfiltered power spectrum shown in Figure 4-3, the filter has eliminated the high frequency noise while preserving the bandwidth of interest. Thus, the sigma-delta modulator is being utilized properly, as the noise has been pushed to high frequencies, and then eliminated by the analog low pass filter.

4.4 Torque Commutator

The torque commutator for each channel is implemented with a single pole double throw (SPDT) switch. The SPDT switch is an analog multiplexer with two inputs and one output; the connection between input and output is determined by a binary
Figure 4-7: Power Spectrum of Analog Subsystem Low Pass Filtered Output

Figure 4-8: Inverter Circuit
control signal [24]. In this case, the control signal is the torque commutator signal coming out of the digital subsystem. The inputs need to be a low pass filtered torque signal and its inverse. The inverse can be generated using a circuit like that shown in Figure 4-8.

<table>
<thead>
<tr>
<th>Commutator Value</th>
<th>Left Output Sign</th>
<th>Right Output Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>'0'</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>'1'</td>
<td>-</td>
<td>+</td>
</tr>
</tbody>
</table>

Table 4.3: Example Scheme to Create Out-of-Phase Commutated Torque Signals

Equipped with the torque magnitude and its inverse, the commutator can be fully completed using an SPDT switch with the digital commutator signal as the control bit. The left and right commutated torque signals must have opposite phase from each other in order to torque the sensor properly. To accomplish this, a scheme can be used where the inputs to the left channel commutator are reversed in comparison to the right channel commutator. An example of such a scheme is demonstrated in Table 4.3. In this table, the sign indicates whether or not the torque magnitude or its inverse should be output from the SPDT switch. For example, if the torque commutator is at '0', the left commutator should be outputting the left torque magnitude while the right commutator should be outputting the inverse of the right torque magnitude.

4.5 Slew-Rate Limiter

The slew-rate limiter makes the slew-rate of each commutated torquer fixed and constant rather than being determined by the hardware capabilities of the analog multiplexer used for the commutator. This is important to prevent the two channels from having different slew-rates, which would unbalance the equivalent DC torques imparted by each channel. While this problem of differing slew-rates could be compensated digitally by varying $\kappa_L$ and $\kappa_R$, it is much simpler to have a well matched system where this type of experimental digital compensation does not need to be performed.

An analog multiplexer capable of handling voltages above 5V, such as the MAXIM
Figure 4-9: Slew-Rate Limiter Circuit: First Order Sallen-Key Butterworth Filter

MAX4583, has a switching time in the worst case of 200ns [15]. The switching time for these circuits is defined as the time it takes to reach 90% of the way to the final value. We would like to slow the transient much more than 200ns so that the slew-rate that we set dominates the transient response during a switching event. If the slew-rate is only reduced to slightly more than 200ns, the slew-rate characteristics of the analog multiplexer could still distort the output waveform. In addition, the torque commutator frequency is set at 2.5KHz, so the slew-rate limiter must allow this frequency and its first few harmonics to pass through.

The slew-rate limiter can be implemented as a low pass filter with $f_{3dB} = 20KHz$. This frequency is sufficiently high enough to allow the 2.5KHz square wave and its first 3 harmonics to pass through. All higher harmonics are attenuated or eliminated, making the slew rate much slower. The filter was designed using the FilterLab filter design software [26] as a first order active filter using the Sallen-Key Butterworth topology. This filter is shown in Figure 4-9, again using 1% resistors. The output of this filter can be fed directly into the upper torque plate, while its inverse is taken and fed into the bottom torque plate.

A time domain representation of the slew-rate limited torque signal is shown in Figure 4-10. In this case, the commutated signal has passed through the slew-rate limiter and now exhibits the slow transient response that we were designing for. To achieve 90% of the transition to its next value, this slew-rate limited signal requires 19.47μs. This is nearly 10 times longer than the 200ns transient delay of the analog multiplexer, so the slew-rate should be safely equal for both the left and right channels.
Figure 4-10: Slew-Rate Limited Torque Signal
Chapter 5

Implementation Details

This chapter discusses implementation details that were not covered in Chapters 3 and 4. Various coefficient scaling is necessary to conserve power and improve accuracy. This coefficient scaling is discussed in Section 5.1. A computer software program was designed to receive data from the preload prototype board, and the design of this program is presented in Section 5.3. In order to communicate properly with the sensor interface ASIC as well as the PC computer, some voltage conversion was necessary, which is described in Section 5.2. Section 5.4 lists the actual parts chosen for the prototype board and some reasons why certain parts were chosen over others. Finally, Section 5.5 describes the layout of a printed circuit board prototype for testing of the preload system.

5.1 Coefficient Scaling

Proper coefficient scaling maximizes the precision of the electronic subsystem and also reduces the power consumed by the system. The preload architecture torques the sensor on both channels simultaneously. In a 0g situation, both sides are torqued with $V_B$ and the power consumed on each channel is represented by the following equation [21]:

$$P_L = P_R = 2C_LV_B^2$$  \hspace{1cm} (5.1)
The value of $C_L$ is not known, but the important point of this equation is to illustrate that the power consumption increases with the square of $V_B$. Thus, reducing $V_B$ to the minimum value possible while maintaining the design goals in Table 1.1 is a priority.

The torque voltages must never have a negative magnitude. The digital subsystem will prevent this from happening; specifically, the torque adjustment module will saturate the torque magnitude at 0 or +1, which will saturate the analog torque magnitudes at 0V or +8V. However, we would like to scale $V_B$ so that it is the minimum value which causes this saturation to occur precisely at the maximum allowable input.

The worst case scenario in terms of channel saturation occurs when there is a +10g or a -10g input. In these cases, we would like one torque magnitude to be 0 and the other one to be +8V. This is most easily done experimentally, by putting in a +10g or a -10g acceleration input and seeing what the rebalance bias should be in order to satisfy the design goal. Specifically, the torque adjustment module’s minimum output should be near 0 in this case.

The constants $\kappa_L$ and $\kappa_R$ are the digital channel gains in the torque adjustment module prior to the addition to or subtraction from the rebalance bias. They were arbitrarily set at 0.8, which gives a little bit of freedom in both the positive and negative directions to compensate for channel inequalities in the analog realm.

Figure 5-1 shows the left and right magnitudes after the torque adjustment module. This simulation was run with a step input of +10g. The rebalance bias is set at 0.4. Before the step occurs, the input acceleration is at 0g and both torquer magnitudes are 0.4. When the step occurs, the left channel magnitude dips down almost to 0 and the right channel magnitude rises. Thus, 0.4 is a good rebalance bias in this case because it is low enough that the channel magnitude almost reaches 0 with a maximum or minimum input acceleration.

A design trade-off could be to lower $V_B$ further by reducing the rebalance bias. In this case, $\kappa_L$ and $\kappa_R$ would need to be reduced in order to avoid hitting 0 with a maximum and minimum input acceleration. This alternative design would have lower power as a result of the lower $V_B$ value. However, it would have less accuracy since
the range of possible digital values would be reduced. In the fixed point representation scheme, this would mean fewer bits per sample. One advantage of the preload architecture is that the rebalance bias and the $\kappa_L$ and $\kappa_R$ values can all be adjusted depending on the specific application requirements.

5.2 ASIC & PC Interfacing

The input and output characteristics of the digital logic use the LVTTL standard, where 0V represents a low signal and 3.3V represents a high signal. However, this was not the same standard used by either the accelerometer interface ASIC or the PC computer receiving data from the UART connection. The ASIC ran on standard TTL levels, where 0V is a low signal and 5V is a high signal. Thus, a simple level shifter was needed to convert the LVTTL signals into TTL signals that the ASIC could understand.

The serial port on the PC uses the RS-232 standard\(^1\), which describes valid voltage

\(^1\)The RS-232 standard is also commonly referred to as the EIA232 standard.
levels and what each pin is reserved for on the serial port connector. The RS-232 voltage standard has a signal over +6V as a low value and a signal less than -6V as a high value. Luckily, there are commercial ICs, that can perform the conversion from LVTTL into RS-232 levels. For this project, the Maxim MAX3221 was chosen as an RS-232 converter [14].

5.3 Software Design

A Java software application was created to receive data incoming from the UART described in Section 3.8.3. This simple application was responsible for capturing the acceleration data and the compensation variable being exported from the preload system. To use the serial port, the Java application must include the `javax.comm` package API.

When the Java application is running, incoming data on the serial port will trigger an interrupt that can be caught by using the following code:

```java
public void serialEvent(SerialPortEvent event) {
    switch (event.getEventType()) {
    case SerialPortEvent.DATAAVAILABLE:
        // PROCESS DATA HERE
        break;
    }
}
```

The first task that the Java application must accomplish is to obtain synchronization with the incoming data. As described in Section 3.8.3, a special synchronization packet is sent so that the computer can distinguish between each packet. The Java application will read in bytes and throw them away until it finds a synchronization byte.

At this point, the application knows that the next 4 bytes will be the acceleration estimate and the 3 bytes after that will represent the compensation variable. So, the application reads in 7 bytes and converts them into integers representing the sent digital values. The next byte that the Java application should receive is a synchronization
byte, so it checks to make sure that the communications is still synchronized. If not, then the application throws away bytes again until it re-achieves synchronization.

The Java application was also programmed with the ability to average a certain number of incoming samples together in order to act as a rudimentary low pass filter. This was done just to reduce the amount of data if it were to prove unnecessary. With an input bandwidth of <100Hz, the 625 acceleration estimates arriving every second were more than necessary to represent the bandwidth of interest.

Figure 5-2 shows a screenshot of the Java application, which was able to successfully capture data with no dropped packets. The only problems encountered with the application occurred when the computer running the application ran out of memory, which could occur on an overnight testing run. However, this could be mitigated by increasing the amount of memory allocated for the Java application.
<table>
<thead>
<tr>
<th>Function</th>
<th>Manufacturer</th>
<th>Part Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analog Multiplexer</td>
<td>MAXIM</td>
<td>MAX4583</td>
</tr>
<tr>
<td>ASIC Interface Level Shifters</td>
<td>Analog Devices</td>
<td>CMP401</td>
</tr>
<tr>
<td>FPGA</td>
<td>Xilinx</td>
<td>XCV400E</td>
</tr>
<tr>
<td>Input Sigma-Delta Modulator</td>
<td>Analog Devices</td>
<td>AD7724</td>
</tr>
<tr>
<td>Op-Amps</td>
<td>Analog Devices</td>
<td>OP285</td>
</tr>
<tr>
<td>RS-232 Level Converter</td>
<td>MAXIM</td>
<td>MAX3221</td>
</tr>
<tr>
<td>Torque Channel Level Shifter</td>
<td>Analog Devices</td>
<td>AD790</td>
</tr>
</tbody>
</table>

Table 5.1: Components Chosen for the Prototype Board

5.4 Component Selection

This section describes the actual commercially available parts that were selected for various pieces of the preload system and the rationale for choosing each part. In most cases, there are numerous possible parts that would have been suitable, and the specific part was chosen based on availability and/or cost. This section does not discuss such parts as resistors, capacitors, and linear regulators, where any manufacturer would have been suitable; rather, it covers those components where the decision on a specific part was important to the correct and accurate function of the preload architecture. The selected components are summarized in Table 5.1 and are discussed in alphabetical order by function.

5.4.1 Analog Multiplexer

The analog multiplexer functions as the torque commutator in the analog output stage. As described in Section 4.4, a high speed multiplexer is necessary to avoid distorting the transient behavior when the commutator signal switches polarity. The MAX4583 has a rise time of 200ns, which was shown in Section 4.4 to be sufficiently fast to avoid significantly distorting the commutated torque signal after the slew-rate filter [15].

This multiplexer can be configured as a SPDT switch, which is the configuration that is used in the preload system. It was also chosen for its ability to handle voltages up to 13V. Most commercially available analog multiplexers either handle supply
voltages of 5V or 12V; a 5V supply would not be sufficient in this case since the low pass filtered torque magnitudes are expected to range from 0V to +8V (and the inverse ranges from -8V to 0V).

5.4.2 ASIC Interface Level Shifters

The FPGA outputs use the LVTTL standard while the ASIC only accepts input that follow the TTL standard, so comparators were used to shift the voltage levels between domains. A 1.65V reference (which is halfway between a high and low value of LVTTL) was used as a comparison to decide whether the output should be 0V or 5V. The CMP401 was chosen to perform this function because of its low cost and ease of use [7]. The values that the FPGA sends to the ASIC are all DC values, so speed is not an issue for these comparators. In cases where speed is an issue, such as the torque channel level shifter, a different comparator was used.

5.4.3 FPGA

The Xilinx XCV400E was chosen as the FPGA [30]. This is one of the largest available FPGAs that comes in a package other than a Ball Grid Array (BGA) package. While BGA packages allow a higher number of connections and reduce the amount of crosstalk between input and output pins, a flat package allows pins to be more easily checked with an oscilloscope on a prototype board.

<table>
<thead>
<tr>
<th>Logic Element</th>
<th>Used</th>
<th>Available</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slice Flip Flops</td>
<td>1,414</td>
<td>9,600</td>
<td>14%</td>
</tr>
<tr>
<td>4 input LUTs</td>
<td>2,705</td>
<td>9,600</td>
<td>28%</td>
</tr>
<tr>
<td>Slices</td>
<td>1,713</td>
<td>4,800</td>
<td>35%</td>
</tr>
</tbody>
</table>

Table 5.2: Percentage Utilization of the FPGA

Table 5.2 shows the percentage of the available space within the FPGA that was used by a synthesis of the digital subsystem. There is sufficient space to experiment with additional digital hardware, since only 35% of the available slices are being used. In addition, the other components of the FPGA, such as slice flip flops and

85
input LUTs also have ample available resources in case design exploration is desired.

5.4.4 Op-Amps

The op-amps in the torque channel are critical to properly signalling the intended torque magnitude. Op-amps typically have an DC offset voltage so that even when they are in a stable negative feedback configuration, the output voltage differs from the input voltage by a constant value [24]. The OP285 is a precision op-amp and has a maximum of 600μV offset voltage, which will result in a relatively accurate analog output stage [6]. In the analog low pass filter that filters the sigma-delta modulated digital signal, the input into the first stage of the Sallen-Key topology filter has a maximum frequency of 5.12MHz. Thus, the 9MHz bandwidth of the OP285 is more than sufficient to process the torque signals properly [6].

5.4.5 Torque Channel Level Shifter

The torque channel level shifter is used to perform the level shift described in Section 4.2. In this case, the level shift has to be performed at 5.12MHz and the precision is very important, unlike the level shift performed for the ASIC interfacing. The AD790 is a good fit for this application because it is a fast, precision comparator. Specifically, it has a 45ns maximum propagation delay and a 250μV maximum offset voltage [8]. In addition, it can handle voltages up to ±15V, which is necessary since it will be converting the FPGA’s LVTTL signals into -4V and +12V.

5.5 Printed Circuit Board Layout

This section covers some of the key design decisions and techniques used in designing the printed circuit board (PCB) layout. The PCB represents a test platform for experimentation and verification of the preload architecture and overall system design. As such, it must be designed to be as noise-free as possible yet still allowing for a wide range of verification procedures as well as maximum reprogramming and
reconfiguration opportunities.

5.5.1 Layer Stackup

The layer stackup was created with two design principles in mind. First, costs increase greatly with number of layers, so the total number of layers on the PCB should be as small as possible. Secondly, there are many ‘sensitive’ signals in the preload system that are susceptible to corruption through capacitive coupling. To reduce this crosstalk, they should be sandwiched in between ground planes.

The final layer stackup is shown in Figure 5-3. Signals are routed on the following layers: Primary Component Side, Analog Signals, Digital Signals, and Secondary Component Side. Each of these layers is separated from the next closest signal carrying layer by a ground plane. Analog signals are less resistant to noise than digital signals and are thus confined to their own layer. If analog and digital signals were mixed onto the same layer, the wide swinging digital signals would tend to corrupt the sensitive analog signals.

Both the Primary Component Side and Secondary Component Side are capable of accepting surface mount components. This feature costs more, but is very necessary for the sensor placement, as will be explained in Section 5.5.2. Only standard vias are used in this system, since partial and hidden vias are difficult to debug. In addition, no vias under pads were used because it can be difficult to test connectivity between
the pad and the via. Instead, in the situation where a via under pad would be typically considered, a short trace on the component side is extended to a standard via.

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Needed For</th>
<th>On Power Plane?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-15V</td>
<td>High Voltage Component Supply</td>
<td>Yes</td>
</tr>
<tr>
<td>-5V</td>
<td>Compensation Variable Amplifier</td>
<td>No</td>
</tr>
<tr>
<td>-4V</td>
<td>Analog Output Stage</td>
<td>No</td>
</tr>
<tr>
<td>1.65V</td>
<td>LVTTL Level Shifter Reference</td>
<td>No</td>
</tr>
<tr>
<td>1.8V</td>
<td>Secondary FPGA Power Supply</td>
<td>No</td>
</tr>
<tr>
<td>2.5V</td>
<td>TTL Level Shifter Reference</td>
<td>No</td>
</tr>
<tr>
<td>3.3V</td>
<td>Main FPGA Power Supply</td>
<td>Yes</td>
</tr>
<tr>
<td>+5V</td>
<td>Powering TTL Components</td>
<td>Yes</td>
</tr>
<tr>
<td>+12V</td>
<td>Analog Output Stage</td>
<td>No</td>
</tr>
<tr>
<td>+15V</td>
<td>High Voltage Component Supply</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 5.3: Voltage Levels, Functions, and Presence on Power Plane of Voltage Signals in the Preload System

Table 5.3 shows the voltage levels present in the system. Of all of these voltage levels, only -15V, 3.3V, +5V, and +15V are placed on the power plane. All the other voltages are routed through one or more of the signal layers. The presence of only one power plane reduces the total number of layers and thus reduces the cost. The four voltages on the power plane were chosen because they are needed in the most places around the board. There are many components that run on these voltages, while the other voltage levels are used for less components.

The power plane is shown in Figure 5-4. This image shows the twists and bends that the each power plane must take to reach the vias that are connected to the corresponding voltage. The level of complexity of this power plane prohibits more DC voltage levels from being added to this plane, which explains why only 4 total voltages are on this plane. The 0V voltage level is connected to the most components in the system, but does not need to be included on the power plane since it is already present on all the ground planes.

88
Figure 5-4: Power Plane in PCB

Figure 5-5: Sensor Placement and Signal Integrity Protection
5.5.2 Sensor Placement

The placement of the sensor is one of the most critical elements of the entire PCB design. This is because the sense pickoff of the sensor is reading the low swing voltage on the proof mass and amplifying this value to feed into the digital domain. Slight noise errors will destroy this reading and ruin the accuracy of the entire system. To protect the signal between the MEMS accelerometer and the interface ASIC, the accelerometer is mounted directly underneath the ASIC. In addition, every layer (including the power plane) has a small grounded copper fill around the sensor signal's via. This setup is shown in Figure 5-5. The surface traces are present to prevent the use of a via under a pad.

5.5.3 Bypass/Decoupling Capacitors

Nearly all of the components used in the preload system require the use of bypass/decoupling capacitors to low pass filter out noise on the power supplies. In general, the technique used for placing these bypass capacitors was to place them on the same layer as close to the component needing the bypass capacitor. For example, Figure 5-6 shows the MAX3221 RS-232 level shifter with its bypass capacitors around it.

This technique worked for all of the components except for the FPGA. The FPGA had so many power pins that needed bypass capacitors that it was physically impossible to array the capacitors around the device without blocking all the signals that
needed to go in and out of the FPGA [1]. So, with the FPGA mounted on the Primary Component Side layer, the bypass capacitors were added to the underside of the board on the Secondary Component Side. Figure 5-7 shows the placement of these bypass capacitors in relation to the FPGA. In this diagram, the hatched markings on the via indicate a connection with the copper plane. Note that in this figure, only one ground plane is indicated, when in actuality there are 3 ground planes that are all connected to the ground via.

The technique shown in Figure 5-7 could be used for all the bypass capacitors in the system. However, this would greatly increase the number of vias in the system since a via would be required for every bypass capacitor. This is very wasteful and would increase the cost of the PCB significantly. Instead, the trace to a bypass capacitor on the same surface as the component is much cheaper and simpler.

### 5.5.4 Symmetry

Many of the signals in the analog output stage are extremely sensitive to noise. For this reason, the op-amps and comparators for the left and right channel analog output stages are assembled in a perfectly symmetrical manner around the accelerometer. This is done in an effort to equalize the noise that each one experiences so that the left and right channels behave in a similar fashion.

In addition, the op-amps and comparators for the analog output stage are mounted
on the Secondary Component Side, which is where the accelerometer itself is mounted. In this fashion, they can be placed very close to the accelerometer itself in an effort to minimize trace lengths and reduce noise effects on the signals.
Chapter 6

Conclusions and Recommendations for Future Work

This thesis provided a detailed design and implementation for a novel electronics system for a force-rebalanced MEMS accelerometer. The system was designed with the flaws of previous electronics systems in mind. Specifically, previous systems used techniques that resulted in poor accuracy around 0g. The preload architecture was designed to have uniform accuracy at all input accelerations with a minimum of digital hardware. The design goal of minimizing digital hardware is evident in the design choices to use CIC filters and linearizing the feedback loop without performing a square root operation.

In Chapter 2, the MEMS sensor was carefully characterized and modeled so that both functional and detailed simulation models could be built around it. The electronics requirements for the system were outlined and refined to match the design goals and the sensor characteristics. In this chapter, the preload concept was also introduced and compared to an alternative architecture that used a square root to linearize the control loop. The preload architecture was split into digital and analog subsystems based on how well each domain suited various computations. The frequencies and sample rates used in the preload system were also chosen in this chapter. Finally, a functional simulation was presented that showed how the system worked on a very basic level.
Chapter 3 presented the digital design in detail. The demodulation process was first presented, with hardware optimizations to the demodulator emphasized. Next, the advantages of the CIC decimation filter were explained and the filter was characterized in detail. The PI controller was presented with an explanation of tuning the dynamics of the feedback system. After the controller, the CIC interpolation filter was presented with an explanation of the necessity of multiple sample rates within the digital system. The torque adjustment module was the next block; this block was simple, but the design decision to place the torque adjustment at the fast sample rate was explained. The D/A sigma-delta modulator was next explained, with a presentation of various related issues including coefficient generation and output scaling. The output transmitter block, which is not part of the feedback loop, was explained with details regarding the two different CIC decimation filters and the UART packager. Finally, synthesis results for the various blocks was examined, comparing the relative sizes of the various digital subsystems.

The next chapter dealt with the components of the analog output stage. First, the level shifter was presented; this block converts the signals coming out of the digital subsystem to the proper voltage levels. Next, the analog low pass filter that eliminates the high frequency noise and converts the sigma-delta modulated signal into a baseband signal was described. After this, the torque commutator and its design were explained. Finally, the slew-rate limiter, the necessity for such a filter, and the design of the filter were presented.

Chapter 5 dealt with implementation details that did not really fit into either the digital subsystem or the analog subsystem. Some coefficient scaling was necessary to minimize power consumption and maximize the precision of the digital subsystem. Some voltage level shifting was necessary to properly interface with the ASIC and the computer. The Java application written to receive data from the test board was discussed in detail. In chapter 5, the critical component selection design choices were also discussed. Finally, the printed circuit board was described, with attention to certain details including the layer stackup, sensor placement, decoupling capacitor placement, and symmetric design techniques.
The top-down design methodology was appropriate and worked well for this project. The functional simulation allowed us to quickly determine whether the preload architecture was a viable method for rebalancing the MEMS sensor. This was critical because the preload architecture might have had unexpected issues that prevented the proper operation of the control loop. Working down through the abstraction layers in the design methodology proved helpful since incremental changes could be made that increased the level of detail of abstract blocks until a working system was designed and implemented.

One mistake made during the design process was the implementation of a closed loop VHDL simulation of the entire system. While in some systems this could be a valuable tool, it turned out to be unnecessary for this particular system. The main problem with a closed loop VHDL simulation is that it requires a detailed discrete time approximation of the MEMS sensor. Since the sensor is a higher-order system with complex dynamics, creating this model turns out to be a non-trivial task and results in a very complicated VHDL model of the sensor. Another annoyance is the need for discrete time approximations for all the analog components. This is less of a problem because there are automated design tools that can generate discrete time filters approximating the behavior of continuous time analog filters. Instead of constructing the entire closed loop VHDL simulation, a better design approach would be to implement only the digital components in VHDL. This model would have to be simulated in a different way since a closed loop simulation would not be possible without VHDL models of the continuous time components. The detailed digital simulation (from the higher abstraction level) could be run and the inputs and outputs to the digital blocks could be recorded. The recorded inputs could be fed into the VHDL simulation and the outputs of the VHDL simulation could be compared to the outputs from the detailed digital simulation. This simulation method is much faster and simpler to implement than a full-fledged VHDL simulation and yields results that are equally accurate.

An extremely effective design technique developed during this thesis consisted of cosimulation between a detailed Simulink model and a VHDL model representing
actual logic. This technique is similar to the digital verification mentioned in the previous paragraph but can be applied to individual digital blocks to verify their accuracy. The method consists of running an entire detailed Simulink model in a closed loop fashion and monitoring the inputs and outputs to a single digital block. Then, these inputs can be used to probe the VHDL model. The outputs from the VHDL simulation should match perfectly to the detailed digital simulation outputs except for differing quantization noise. The detailed simulation can be set to maximum floating point precision to estimate the digital quantization noise by examining the difference between the simulations.

The design is likely to satisfy the design goals described in Table 1.1, although a prototype was not fully tested. However, in detailed simulations, the input range was set using the technique described in Section 5.1 to fully cover the ±10g range. That section also describes how to change the input range if a different range is desired or if the implementation does not match the model. In a detailed simulation, the attenuation at the 100Hz frequency is -0.2646dB, which is better than the -3dB specification. If the model differs significantly from the implementation, the PI controller can be tuned or replaced with a different controller in an effort to get better attenuation results at the high frequency limit.

Future work should consist of circuit board testing and experimentation. This system was presented as a proof-of-concept idea with no firm goals on accuracy, drift, or temperature reliability. Further exploration should attempt to quantify and improve the accuracy and reliability of the system. Also, further investigation is necessary to verify that the system performs accurately at all input accelerations and does not experience any dead band or discontinuity problems that plague other accelerometer systems of this type.

Further research can also be done to optimize much of the hardware inside the digital subsystem. For example, the adders and multipliers in the preload system are implemented as combinatorial arithmetic units, and more efficient units could be used. In addition, experimentation can be done with changing the system frequencies and sample rates in an attempt to reduce capacitive coupling and improve the accuracy.
of the system. Also, varying precisions of digital representations can be tried, also to improve accuracy.

Finally, the ultimate goal of this research is to develop a mixed-signal ASIC that combines all the functionality of the sensor interface ASIC, the digital subsystem, and the analog output stage into one compact device. Such a device would have major advantages in terms of noise coupling, since all the signals would be internal to the ASIC. However, fabrication of such a mixed-signal ASIC is expensive, so the system must be fully characterized and understood before a design can be attempted.
Bibliography


