DEVELOPMENT OF AN ACTIVE SUSPENSION SCALE VEHICLE PLATFORM

KEITH J. WAKEHAM

Development of an Active Suspension Scale Vehicle Platform

by

© Keith J. Wakeham

A thesis submitted to the
School of Graduate Studies
in partial fulfilment of the
requirements for the degree of
Master of Mechanical Engineering

Department of Engineering
Memorial University of Newfoundland

September 2011

St. John's Newfoundland

Abstract

Active suspension has shown an ability to selectively improve various aspects of suspension performance. These criteria can range from limiting vertical accelerations to more economically using suspension or tire deflections. This is used as a starting point in the development of a quarter car test apparatus.

This research develops optimal controllers for various car models and provides a comparison which is useful in determining required model complexity. An electrical model of an actuator is incorporated into the more basic quarter car model and again an optimal controller is developed and simulated. The results show that there is potential for real world applications using a slightly more complex optimal controller incorporating an actuator model. Using these simulations various parameters are generated to develop a design of a quarter car test apparatus. Initial work on a development of an actuator was conducted which led to the selection of an off-the-shelf voice coil along with other various parts. Being mindful of future developments, the test apparatus was designed and developed around a Field Programmable Gate Array (FPGA) based controller. The FPGA has potential for high speed parallel processing which makes it ideal for running multiple controllers simultaneously as well as implementing mathematically intensive operations but at a significant cost in development time. A basic optimal control implementation was developed and tested and proved to function very well, opening the possibilities of other development.

Acknowledgements

While it's unorthodox I'd like to acknowledge myself. I was born into a damaged family, a single parent who was disabled, dying and leaving me orphaned as an unwanted teenager. Subjected to the disorganization of a youth services program where I had to tell my brother to leave my life. I found myself living on my own and ashamed to be supported as a teenager by the government while finishing high school. I've learned the impermanence and fragility of life, how to do things in spite of poor instruction, and to be proud of my own accomplishments. Though I disagree with elements of my upbringing, I acknowledge that people along the way offered help and opportunities. As such I am not the average, or the normal, or the stick by which others should be measured. I view things differently because I've had to and adapt quickly as a result. I work in peculiar ways and operate at the extremes of ideas and emotions. I can't say I know precisely why I was afforded this opportunity. I saw and admired a professor and wanted to follow for at least a while; to learn and work together on a multitude of ideas. Without what I've just said acknowledging my supervisor would have been meaningless and common. He has provided me a stepping stone in life like so many others. I had but to reach out and accept, and in doing so I'm sure he has regretted offering it some days and rejoiced others. For most people a Master's degree is a stepping stone to be congratulated. For me it's part of a story of highs and lows, but never anything in the middle.

Contents

List of Tables

List of Figures

Nomenclature

α_{xx}	Weighting Matrix Terms
β	Magnetic Flux Density (T)
$\ddot{x}$	Acceleration (m/s^2)
$\dot{p}$	Derivative of momentum
$\dot{q}$	Derivative of flow
$\dot{x}$	LTI State Derivative Matrix
$\dot{x}$	Velocity (m/s)
$\omega_{I_{xx}}$	Pitch Mass Angular Velocity (rad/s)
$\omega_{I_{yy}}$	Roll Angular Velocity - Full Car (rad/s)
ω_{sus}	Natural Frequency of Suspension (Rad/s)
ρ_x	Performance Index Weighting Term
θ_s	Pitch Angle (rad)
ζ	Damping Ratio

A	LTI System Matrix
B	LTI Disturbance Matrix
b_s	Suspension Damping $(N \cdot s/m)$
b_t	Tire Damping $(N \cdot s/m)$
C	LTI Gravity Matrix
c	Damping of 1 DOF system
e	effort (or force)
f	flow (or velocity)
F_a	LTI Force Actuator Matrix
f_{sus}	Natural Frequency of Suspension (Hz)
$f_{wheelhop}$	Tire Natural Frequency (Hz)
g	Gravity, Acceleration Due To (m/s^2)
i	Current (Amp)
i_{coil}	Current In Voice Coil (A)
J	Performance Index
J_p	Pitch Inertia $(kg \cdot m^2)$
K	LQR Gain Matrix

k	Stiffness (N/m)
k_f	Front Stiffness - Simplified Half Car (N/m)
k_r	Rear Stiffness - Simplified Half Car (N/M)
k_s	Suspension Stiffness (N/m)
k_t	Tire Stiffness (N/m)
L	LTI Road Input Matrix
l	Conductor Length (m)
l_f	Distance from front axle to C.G. (m)
l_r	Distance from rear axle to C.G. (m)
L_{coil}	Voice Coil Inductance (H)
m	Mass (kg)
m_s	Sprung Mass (kg)
N	LQR Input-State Combination Weighting Matrix
P	LQR Solution To Matrix Riccati Equation
P_b	Power of Bond Graph Element (watt)
Q	LQR State Combination Weighting Matrix
q_{k_s}	Suspension Spring Deflection 2 DOF (m)

q_{k_t}	Tire Spring Deflection 2 DOF (m)
$q_{k_{fs}}$	Front Suspension Stiffness - Half Car(m)
$q_{k_{ft}}$	Front Tire Stiffness - Half Car (m)
$q_{k_{rs}}$	Rear Suspension Stiffness - Half Car (m)
$q_{k_{rt}}$	Rear Tire Stiffness - Half Car (m)
$q_{ks_{fl}}$	Deflection Front Left Suspension Spring - Full Car (m)
$q_{ks_{fr}}$	Deflection Front Right Suspension Spring - Full Car (m)
$q_{ks_{rl}}$	Deflection Rear Left Suspension Spring - Full Car (m)
$q_{ks_{rr}}$	Deflection Rear Right Suspension Spring - Full Car (m)
$q_{kt_{fl}}$	Deflection Front Left Tire Spring - Full Car (m)
$q_{kt_{fr}}$	Deflection Front Right Tire Spring - Full Car (m)
$q_{kt_{rl}}$	Deflection Rear Left Tire Spring - Full Car (m)
$q_{kt_{rr}}$	Deflection Rear Right Tire Spring - Full Car (m)
R	LQR Input Combination Weighting Matrix
r	Gyrator Constant
r_1	Weighting Term Sprung Acceleration (Krtolica-Hrovat)
r_2	Weighting Term Pitch Angular Acceleration(Krtolica-Hrovat)

r_3	Weighting Term Front Deflection (Krtolica-Hrovat)
r_4	Weighting Term Rear Deflection (Krtolica-Hrovat)
R_{coil}	Voice Coil Resistance (Ω)
u	LTI Input Matrix
v_{m_s}	Absolute Sprung Velocity 2 DOF (m/s)
v_{m_u}	Absolute Unsprung Velocity 2 DOF (m/s)
$v_{m_{fu}}$	Front Unsprung Velocity - Half Car (m/s)
$v_{m_{ru}}$	Rear Unsprung Velocity - Half Car (m/s)
v_{m_s}	Sprung Mass Velocity (m/s)
$v_{mu_{fl}}$	Velocity Front Left Unsprung Mass - Full Car (m/s)
$v_{mu_{fr}}$	Velocity Front Right Unsprung Mass - Full Car (m/s)
$v_{mu_{rr}}$	Velocity Rear Left Unsprung Mass - Full Car (m/s)
$v_{mu_{rr}}$	Velocity Rear Right Unsprung Mass - Full Car (m/s)
v_s	Sprung Mass Absolute Velocity - Half Car (m)
v_{uf}	Front Unsprung Mass Absolute Velocity - Half Car (m/s)
v_{ur}	Rear Unsprung Mass Absolute Velocity - Half Car (m/s)
x	Displacement (m)

x	LTI State Matrix
z_r	Road Absolute Position - Quarter Car (m)
z_s	Sprung Mass Absolute Position - Quarter Car (m)
z_u	Unsprung Mass Absolute Position - Quarter Car (m)
z_{sf}	Front Sprung Mass Absolute Position - Half Car (m)
z_{sr}	Rear Sprung Mass Absolute Position - Half Car (m)
z_{uf}	Front Road Absolute Position - Half Car (m)
z_{uf}	Front Unsprung Mass Absolute Position - Half Car (m)
z_{ur}	Rear Road Absolute Position - Half Car (m)
z_{ur}	Rear Unsprung Mass Absolute Position - Half Car (m)

Chapter 1

Introduction

1.1 Background and Motivation

The advent of anti-lock braking systems, electronic stability controllers, electronic power steering, semi-active and active suspensions, and other new technologies are creating safer vehicles. While each system on its own can be exhaustively tested, once it is integrated into a very complicated non-linear system and subjected to human control, the performance of the system becomes less predictable. Once a vehicle is near the limits of deflections or traction, the prediction of the system in a simulation environment becomes difficult to accurately reproduce. The current options used in industry are full scale testing or complex simulation environments.

Full scale testing at the limits of vehicle operation is extremely high risk in terms of cost of equipping a vehicle that may be irreparably damaged, and in the control of the vehicle in relation to human life. Instrumenting a vehicle absorbs the bulk of the project cost leaving little for the actual experimentation. This results in testing scenarios much less demanding and catering primarily to the day to day driving style but not the high risk extreme manoeuvres that could more strongly relate to increasing vehicle safety.

Vehicle simulation software such as CarSim [1] is reaching a high level of maturity providing a good simulation platform. In contrast, the design of the controllers happens on simple models, CarSim provides an environment to test and refine them before they ever reach a real vehicle. However good the simulation is, it is dependent on the underlying models which will have some foundation in empirical experimentation. These models may not reflect the real world perfectly, especially at the limits of hardware performance and actuator implementations.

Similitude is a method of translating one model to that of another. It is normally used in relationship to fluid dynamics for drag, lift, and dynamics of aircraft and ships. More recent work has helped develop similitude for scale ground vehicles to full sized ground vehicles with regards to lateral dynamics by testing a small scale vehicle on a rolling road[2, 3, 4, 5, 6]. This method utilizes Buckingham Pi analysis to design the vehicle and obtain appropriate dynamics from the steering actuator. Promising results are opening the doors to more scale vehicle dynamics experimentation.

Electric hybrid vehicles have an easily accessible energy storage device. Normally active suspension requires actuators which have commonly been of a hydraulic variety, requiring a hydraulic pump connected to the engine. A battery enables electro-magnetic actuators such as voice coils or linear motors to be utilized, giving potentially higher bandwidth. Under this pretense it is permissible to develop an electric scale test apparatus with active suspension for the testing of suspension controllers.

1.2 Research Outline

While active suspension work surged and receded to some extent in the early 1990s, there is an abundance of research on creating a better force actuator system. The low cost launch point that would allow testing new theories is that of a linear plant with a wide bandwidth actuator for which work by Gysen et. al [7] has shown great promise with the field of electric actuators. In order to facilitate the development of a scale vehicle for active suspension control, the first step is an examination of the background material in all the related fields in chapter 2. The various levels of model complexity are discussed and developed in chapter 3. This allows the suspension setup to be tuned to give the appropriate suspension and tire natural frequencies when compared to a full scale vehicle. Matching all the dynamics of the system may be impossible, but certain aspects such as the natural frequencies of the suspension and the tires are critical to the dynamics of a suspension system.

Once the dynamics are appropriate, an optimal control technique is needed. In this case the Linear Quadratic Regulator, or LQR, technique is chosen and explained in chapter 4. This controller is normally shown as an example using a quarter car model. Exploration into what is an appropriate level of controller complexity requires the comparison of quarter, half, and full car model based controllers. Due to issues resulting in an inability to solve the control scheme for a full car based controller, quarter and half car models are explored. The results of these simulations are explained in chapter 5.

This optimal control is entirely theoretical in regards to its output being a force. No simple actuator exists solely where a force is the direct output to a commanded input, instead a force actuator is of another type with its own feedback controller. Thus, the actuator dynamics play into the response of the system. Modelling the actuator and modifying the chosen control system gives a better understanding of how the dynamics of the actual system may work, and the results of this work is shown in chapter 6.

In order to validate the simulation work a 1/4 car test rig is designed based on the specifications from the previous simulations. The design is required to be robust but very simple and serviceable as it has to accommodate a wide array of controllers. An explanation of the design requirements and construction is detailed in chapter 7.

The controller needs to be implemented in hardware as well as having states recorded. This is implemented through a combination of Field Programmable Gate Array, or FPGA, and a microcontroller. Some aspects of the controllers to be implemented may require complex mathematical filtering, such as Kalman fitlers and variations thereof. In order to effectively control the system this would require significant microprocessing requirements. The FPGA allows for a hardware implementation that runs at a consistent speed and has an ability to be reconfigured easily. While mathematical operations can be implemented and run in parallel for processing speed, interface operations are very complicated. A relatively simple to program microcontroller is interfaced to the circuit in order to sample data and send it to a computer via USB. The details of this development is shown in chapter 8.

Finally the results of initial simulations are shown in chapter 9. A discussion of these results and of potential future work is presented in chapter 10.

Chapter 2

Literature Review

2.1 Vehicle Models

Studying active suspension involves comparing and utilizing several different types of vehicle models. A wide background of models is provided by Hrovat [8]. Several key models are identified by Hrovat's survey of the developments in active suspension. The linear time invariant models generally follow a simple progression with increasing degrees of freedom (DOF). The main models include:

- 1 DOF quarter car
- 2 DOF quarter car
- 2 DOF half car
- 4 DOF half car
- 3 DOF full car

- 7 DOF full car

There are three main models each with an associated variation; the quarter car model, the half car model and the full car model. The simple versions of these remove the effects of the tire stiffness (spring) and unsprung mass (weight of tire, wheel, uprights, and percentage of A-arms and suspension components). The 1 DOF quarter car is a simplification where the entire mass of the vehicle is assumed to only heave, and as a result all four springs are modelled as a single spring. When the tire is modelled as an unsprung mass (potentially with damper) then the associated model complexity rises with another DOF per wheel. Thus the 2 DOF quarter car model is the classic problem of two masses, one connected to the ground and the other mass via separate springs and dampers.

Figure 2.1 shows the reduced complexity models that have eliminated the unsprung weight and tire stiffness. The half car model assumes that the vehicle can pitch and heave, so the front and rear have springs that idealize the front left and right, and the rear left and right springs and dampers. The full car is allowed to pitch, heave and roll so all 4 springs and dampers have to be modelled. These models have been used to illustrate several key concepts related more to controls theory rather than actual performance. Tseng and Hrovat [9] examined the simplified 2 DOF quarter car as shown in figure 2.2 but without the spring and the damper which is the primary components of a passive suspension. This is the most ideal case of a 2 DOF model because it negates the effects of gravity and limitations of potentially available actuators.

As an extension to this, the half car pitch model, shown in figure 2.1b, is the next logical step in modelling the dynamics, is to develop a controller. Using this model, which again negates the effects of unsprung mass and the tire compliance, provided an examination of model complexity. Krtolica and Hrovat's study of the decoupling [10] effect, which is to say a method for determining when the front and rear can be treated separately for controller design, provides the basis of criteria to model the front and rear sections of the vehicle separately. Krotlica and Hrovat use the performance index:

$$J = \int_0^\infty r_1 \ddot{z_s}^2(t) + r_2 \ddot{\theta}_s^2(t) + r_3 {z_f}^2(t) + r_3 {z_r}^2(t) \tag{2.1}$$

Where r_1 through r_4 are weighting terms relating to the heave acceleration $\ddot{z}^2$, pitch acceleration θ_s^2, front deflection z_f, and rear deflection respectively z_r. Using this as the performance index they were able to determine that if pitch inertia J_p relates to sprung mass m_s as per equation 2.2 within 20% via the front and rear length (figure 2.1), that the front and rear of the vehicle are sufficiently decoupled. The second critera relates to the weighting factors for the front and rear deflection as per equation 2.3 however the sensitivies of this term have yet to be investigated.

$$m_s \cdot l_f \cdot l_r \approx J_p \tag{2.2}$$

$$r_1 \cdot l_f \cdot l_r \approx r_2 \tag{2.3}$$

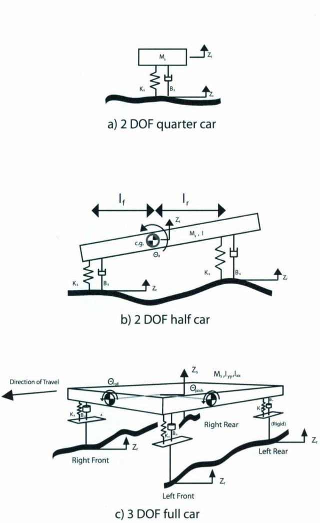
Wong [11] uses a different method of decoupling the front and the rear of the vehicle for a passive suspension. The theory uses a model similar to that shown in figure 2.1b, but without the damper. This allows one to determine two coupling terms that allows one to adjust the front stiffness and rear stiffness to satisfy equation 2.4. This causes the pitch motions to be decoupled from the bounce motions. He states that under this criterion that the passive suspension would result in poor ride quuality which is very different than that of Krtolica and Hrovat.

$$k_f \cdot l_f \approx k_r \cdot l_r \tag{2.4}$$

2.2 Active Suspension

Hrovat's comprehensive works cited in his survey [8] provide the basis for the development and implementation of active suspension within a theoretical framework. Original works citied to be some of the first attempts in active suspension used a Linear Quadratic Gaussian Regulator approach to a simple one DOF model. This model is the most basic implementation of optimal control where the only elements are the sprung mass, the actuator, and the road input shown in figure 2.3.

From this, a considerable amount of work was completed, including work surrounding the idea of a skyhook damper [12], a damper that is connected from the mass to a reference inertial frame. Continued work on the quarter car model eventually brings research to a 2 DOF model for passive suspension shown in figure 2.2a and the active equivalent having both the suspension stiffness and damper replaced

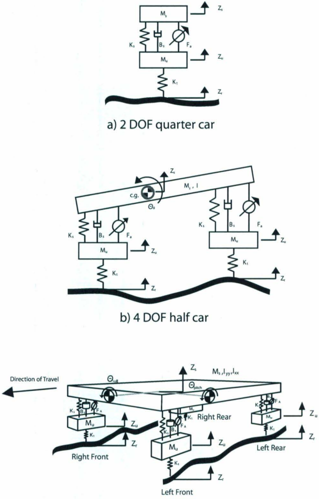


a) 2 DOF quarter car

b) 2 DOF half car

c) 3 DOF full car

Figure 2.1: Simple Models



a) 2 DOF quarter car

b) 4 DOF half car

c) 7 DOF full car

Figure 2.2: Improved Models

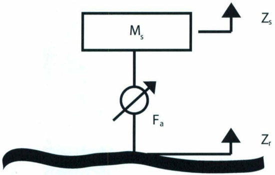


Figure 2.3: Simple 1 DOF Active Suspension

by a force actuator or in some cases the unsprung mass is set to zero and puts the force actuator in series with the tire stiffness.

2.3 Scale Vehicles

Hrovat speculates in his research [8] that the original attempts at active suspension were less successful for a number of reasons, inlcuding that of Coulomb friction causing non-linearities when elements are static and a friction force that is related to normal force. That is, in order to move the suspension, a higher force is required to start the movement than to keep it going. This becomes a primary concern when attempting to implement active suspension on a vehicle for real world application.

Several attempts have been made at full active suspension including that of the Lotus 99T and 100T [13] Formula One racing cars. While the technology was short-lived in Formula One, it has been continually looked at for benefits. Currently, semi-active suspensions have made significant ingress and can be found on several production cars using magneto-rheological dampers to adjust damping continuously [14].

Recent work conducted at the University of California at Berkeley relates to exploration of vehicle improvements using preview control suspension on a full-sized vehicle [15]. Lotus Engineering originally equipped the vehicle with hydraulic actuators, load cells, LVDTs, accelerometers, and gyros as well as a preview sensor.

There is a relatively high cost to this setup and large risks would be taken by testing near operational limits. The arrangement of sensors already requires significant cost on the order of tens of thousands of dollars. Brennan [2, 3] argues that there are four primary motivating factors relating to the use of scaled vehicles: cost, safety, convenience, and flexibility. Brennan's work primarily relates to lateral dynamics and similitude which is expanded in [4, 5, 6]. Eventually, the work with the Illinos Roadway Simulator reached a fairly substantial level of development where Brennan implemented and tested the limits of lateral steering control [16, 17, 18].

Under the guidance of Brennan, Perersheim continued work related to scale vehicles, specifically in the area of developing guidelines on the scaling of vehicle components [19] while working on the more recently designed Pennsylvania State University Rolling Roadway Simulator (PURRS).

It should be noted that Brennan's work required the use of a simple bicycle model which relies heavily on cornering stiffness and tire models. A bicycle model is a model that assumes the lateral dynamics of a vehicle can be idealized as a single front and rear tire. The front wheel can be steered, while the rear remains aligned with the vehicle body. The forces on the vehicle relate to the velocity and steering angles combined with a tire model. Generally the Magic Tire Formula [20] developed by Pacejka is used for this purpose. However, simple derivations utilize a cornering stiffness to determine these forces. In order to facilitate this, he initially derived cornering stiffness experimentally [2]. This work would later be expanded by Polley [21] and is the primary resource for scale tire models. Further work now expands this to solid rubber tires by Witaya et al. [22] who has also started development on the Scaled Vehicle for Interactive Dynamic Simulation, SIS. SIS is a 1/10th scale RC vehicle modified for lateral dynamics. While the vehicle appears to have a suspension it would be of minimal importance to the current research.

While it may be argued that rolling roadways are beginning to mature, they are entirely limited to the lateral dynamics of vehicles currently. As mentioned before, Witaya et al. have begun to expand the research by removing the vehicle from the roadway but little work has been done on scale vehicle suspension dynamics. Some basic work surrounding vehicle roll-over was used to characterize important factors by Mittal and Gulve utilizing a scale vehicle[23]. More recently, work has been conducted using a modified scale vehicle by Verma et al. for longitudinal dynamics research [24]. This work was primarily related to that of scaling theory for motor, transmission, and vehicle properties such that the full scale vehicle, a High Mobility Multipurpose Wheeled Vehicle (HMMWV), could be emulated.

However, scale dynamics incorporating vehicle suspensions is still a very new topic with very little research. The only available information on any developments was a publication by Annis and Southward [25] where a large scale RC vehicle was cut down the middle and mounted with the rear wheels held rigid. This is referred to as a quarter-car but appears to possibly incorporate some pitching dynamics. It was demonstrated with some basic sky-hook semi active controls.

2.4 Quarter Car Test Rigs

A two degree of freedom quarter car as mentioned previously is a major topic and is the first means of developing active suspensions in a laboratory environment. Due to the non-dimensionality already mentioned in terms of natural frequencies, the weights and size of a quarter car are not the most important aspects when designing an active suspension quarter car test rig. The intended goal has dictated the designs used.

Langdon highlights the initial development of a full scale non-linear 2 DOF suspension rig with interchangeable suspension attachments in [26]. It was intended as a passive non-linear suspension test rig and backed out a linear model that fit well but showed some limitations at the extents of operations.

Chantranuwathana and Peng provided many real world solutions to implementing active suspensions in [27]. They were expanding initial work with the assumption that LQR would be used and a force tracking controller would be required. They eliminated the need for a load cell by using an observer based on accelerations, velocities, and displacements to determine actuator force to feed back into an Adaptive Robust Control technique. The motivation for this was previous implementations on test rigs saw that hydraulic actuators were limited to under 2 Hz operations and thus could not provide significant improvements without controller instability.

Pneumatics was initially explored by Anakwa et al. in [28] however it was entirely targeted as an academic exercise. It was a simple 1 DOF active suspension with a roller-cam as a road input to simulate a sinusoid. It showed some promise but appears to have not been fully explored and is intended for a tool for students to learn industrial controls

Lauwerys et al. [29] have developed a semi-active / active suspension test rig with an electro hydraulic actuator, and use frequency identification methods to develop a transfer function and suspension model through experimental work. Limitations of hydraulic actuators are later outlined by Gysen et al. [7] highlighting the initial development of a 3 phase linear actuator to be packaged inside a BMW 540 Macpherson strut volumetric envelope. It used roll data from a lap of the Nurburgring and the target of stopping body roll as the design requirement. This work was expanded by Gysen et al. [30] to include more experimental work on a linear quarter car test rig using a random road input and an LQR controller producing a force target to be matched by the actuator through current sensing of the 3 coils. The force-current relationship proved linear to 2100N of force, where non-linearities were visible. No explanation was given as to why this non-linearities was introduced.

While similar to Gysen, Lee and Kim [31] developed a quarter car test rig and linear brushless permanent-magnet motor and combined it with a variety of controllers . The work showed good correlation with a basic test rig comprised of a sprung and unsprung mass, a linear motor, and suspension spring. The system used a solid wheel with a rigid interface to the unsprung mass. This meant that the tire dynamics were almost removed. The solid wheel was actuated with a spinning cam so there was no designed tire compliance.

2.5 Literature Motivation

The cumulative effort of all of this research is renewed exploration into active suspension. The new possibilities for implementation are prompted by the promise of higher bandwidth electrically powered actuators in the form of linear motors or similar devices. Combining this with the possibilities of dynamic similitude it should be possible to explore responses using a scaled test apparatus. In order to facilitate this, investigations into the models should be conducted first to provide the necessary design criteria to design and develop such a test setup.

Chapter 3

Vehicle Models

3.1 Overview

Generally there are three distinct vehicle models that are normally used to design and simulate vehicle controls. These are the quarter car, the half car, and the full car model. The quarter car is the classic two masses with springs and dampers between the two masses and between the lower mass and road input. The simplifying assumption is that tire damping is negligible and the tire stiffness is significantly higher than that of the suspension. The half car model uses two masses connected to road inputs through springs which are connected to a singular beam at either end. Chantranuwathana et al. [27] have experimental data that shows tire damping coefficients are on the same order of magnitude of suspension damping, but the stiffness of a tire is approximately 10 times that of the suspension. This indicates that tire damping should be included in model development.

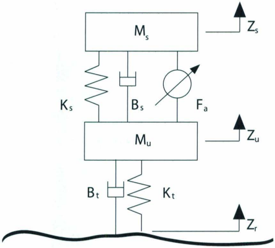


Figure 3.1: Quarter Car Model

3.2 Quarter Car

A quarter car model represents a corner of a vehicle as shown in figure 3.1. The sprung mass of the car body, m_s, is connected by a spring and damper to the unsprung mass of the suspension components, m_u, by the suspension spring, K_s, and the damper, B_s. The tire interfaces with the road via the tire which is modeled as a spring, K_t. Tire lift-off is permitted. It is generally assumed that the tire damping, b_t, is negligible; however it is included in the model derivation due to its importance described previously.

The general linear system is described by

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{3.1}$$

When gravitational and road inputs are separated out of $\mathbf{u}$, the equation becomes

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{F}_a + \mathbf{C}g + L\dot{\mathbf{z}}_r \tag{3.2}$$

Where the vehicle states, spring deflection (q_{k_s}), sprung velocity (v_{m_s}), tire deflection (q_{k_t}), and unsprung velocity (v_{m_u}), are:

$$\mathbf{x} = \begin{bmatrix} q_{k_s} & v_{m_s} & q_{k_t} & v_{m_u} \end{bmatrix}^T \tag{3.3}$$

Using Bond Graphs, the system is modelled in figure 3.2. The governing equations using bond graphs are formed from the derivatives by taking the differential of the power at an element with respect to itself (equation 3.4). This Bond Graph methodolgy was originally developed by Karnopp, Margolis, and Rosenberg [32]. The derivative of momentum ($\dot{p}$) is equal to the effort (e) in equation 3.5. The derivative of the displacement ($\dot{q}$) is equal to the flow (f) in equation 3.6. In order to keep things organized, the bonds are numbered for the quarter car as shown in figure 3.2.

$$P_b = e \cdot f \tag{3.4}$$

$$\dot{p}_x = e_x \tag{3.5}$$

$$\dot{q}_x = f_x \tag{3.6}$$

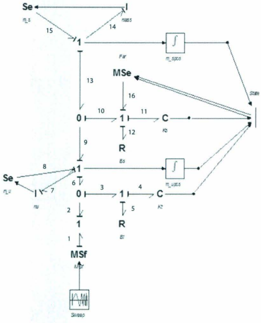


Figure 3.2: Quarter Car Bond Graph Model

Starting with spring deflection q_{ks} we have:

$$\dot{q}_{ks} = f_{11}$$

$$f_{11} = f_{12} = f_{13} - f_9$$

$$f_{13} = f_{14} = \frac{p_{14}}{m_s}$$

$$f_9 = \frac{p_7}{m_u}$$

Thus:

$$\dot{q}_{ks} = \frac{p_{14}}{m_s} - \frac{p_7}{m_u}$$

The momentum of the sprung mass can be calculated in the same way:

$$\dot{p}_{m_s} = e_{14}$$

$$e_{14} = e_{15} - e_{13} = -m_s \cdot g - e_{13}$$

$$e_{13} = e_{10} = e_{11} + e_{12} - f_a(t)$$

$$e_{11} = \frac{q_{11}}{k_s^{-1}} = q_{11} k_s$$

$$e_{12} = b_s \cdot f_{11} = b_s(f_{13} - f_9)$$

$$e_{12} = b_s \left(\frac{p_{14}}{m_s} - \frac{p_7}{m_u} \right)$$

Thus:

$$\dot{p}_{m_s} = -m_s \cdot g - q_{11} k_s - b_s \left(\frac{p_{14}}{m_s} - \frac{p_7}{m_u} \right) + f_a(t)$$

Following the same procedure for the tire deflection:

$$\dot{q}_{kt} = f_4 = f_3 = f_6 - f_2$$

Thus:

$$\dot{q}_{kt} = \frac{p_7}{m_u} - vt$$

And finally for the unsprung mass momentum:

$$\dot{p}_{m_u} = e_7 = e_8 + e_9 - e_6$$

$$e_6 = e_4 + e_5$$

$$e_4 = \frac{q_4}{k_t^{-1}} = q_4 k_t$$

$$e_5 = b_t \cdot f_5 = b_t (f_6 - f_2)$$

$$e_5 = b_t \cdot f_5 = b_t \left(\frac{p_7}{m_u} - v(t) \right)$$

$$e_8 = -m_u \cdot g$$

$$e_9 = e_{11} + e_{12} - fa(t)$$

$$e_9 = q_{11}k_s + b_s\left(\frac{p_{14}}{m_s} - \frac{p_7}{m_u}\right) - fa(t)$$

Thus:

$$\begin{aligned}\dot{p}_{m_u} = e_7 = -m_u \cdot g + q_{11}k_s + b_s\left(\frac{p_{14}}{m_s} - \frac{p_7}{m_u}\right)\\ -fa(t) - q_4k_t - b_t\left(\frac{p_7}{m_u} - v(t)\right)\end{aligned}$$

However some minor manipulation is required to convert the momentum terms, p_{14} and p_7, to m_sv_{14} and m_uv_7. With the state variables identified, matrices A and B for the linear system are developed and shown in equations 3.7 and 3.8, respectively.

$$\mathbf{A} = \begin{bmatrix} 0 & \frac{1}{m_s} & 0 & -\frac{1}{m_u} \\ -k_s & -\frac{b_s}{m_s} & 0 & \frac{b_s}{m_u} \\ 0 & 0 & 0 & \frac{1}{m_s} \\ k_s & \frac{b_s}{m_s} & -k_t & -\frac{(b_s+b_t)}{m_u} \end{bmatrix} \tag{3.7}$$

$$\mathbf{B} = \begin{bmatrix} -\frac{k_s}{{m_s}^2} \\ -\frac{b_s}{{m_s}^2} \\ 0 \\ \frac{b_s}{{m_s}^2} \end{bmatrix} \tag{3.8}$$

A non-linear tire is added, such that the tire only generates force in compression (negative displacement). If the tire spring deflection is positive, corresponding to tension, then the tire will produce no restoring force. Gravity then becomes the only restorative force. This is done through editing the code for the states as shown:

```
parameters
real c = 1.18343e-4; //Compliance, Stiffness(N/m)^-1
equations
    state = int(p.f); //pf = flow, or velocity of the deflection
if state < 0 then
p.e = 0;
else
p.e = state / c;
end;
```

3.3 Half Car

In many situations the pitching motion of the vehicle becomes important and as a result requires a model with the additional degrees of freedom shown in figure 3.3. The main body consists of a mass that is free to rotate and heave vertically. The angles of rotation are assumed to be small so that the end points will be considered to move vertically and the model will be linear. The vehicle states, sprung mass velocity (v_{m_s}), pitch velocity ($\omega_{I_{xx}}$), front and rear suspension deflection (q_{fs} and q_{rs}), unsprung velocity ($v_{m_{fu}}$ and $v_{m_{ru}}$), and tire deflections($q_{k_{ft}}$ and $q_{k_{rt}}$), are:

$$\mathbf{x} = \begin{bmatrix} v_{m_s} & \omega_{I_{xx}} & q_{k_{fs}} & q_{k_{rs}} & v_{m_{fu}} & v_{m_{ru}} & q_{k_{ft}} & q_{k_{rt}} \end{bmatrix}^T \tag{3.9}$$

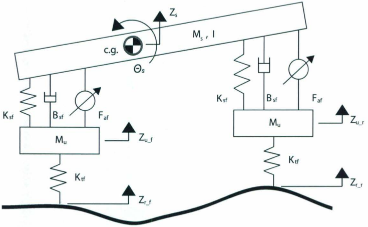


Figure 3.3: Half Car Model

As with the quarter car model, the linear system is developed from the bond graph model shown in figure 3.4. The states are adjusted such that momentum is converted to velocities. The matrices A and B, as describe earlier, for the linear system are shown in equations 3.10 and 3.11, respectively.

$$\mathbf{B} = \begin{bmatrix} -{m_s}^{-1} & -{m_s}^{-1} \\ \frac{l_r}{Ixx} & -\frac{l_f}{Ixx} \\ 0 & 0 \\ 0 & 0 \\ -{m_{uf}}^{-1} & 0 \\ 0 & -{m_{ur}}^{-1} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.11}$$

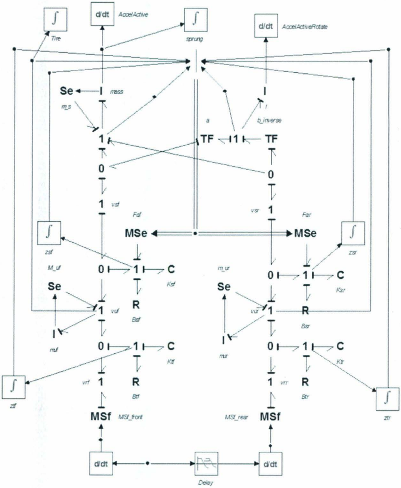


Figure 3.4: Half Car Bond Graph

$$\mathbf{A}x = \begin{bmatrix} -\frac{b_{sf}+b_{sr}}{m_s} & \frac{l_f b_{sf}-b_{sr}l_r}{I_{xx}} & -k_{sf} & -k_{sr} & \frac{b_{sf}}{m_{uf}} & \frac{b_{sr}}{m_{ur}} & 0 & 0 \\ \frac{l_r b_{sf}-l_f b_{sr}}{m_s l_f l_r} & -\frac{l_f l_r(b_{sf}+b_{sr})}{I_{xx} l_f l_r} & \frac{k_{sf}}{l_f} & -\frac{k_{sr}}{l_r} & -\frac{b_{sf}}{m_{uf} l_f} & \frac{b_{sr}}{m_{ur} l_r} & 0 & 0 \\ \frac{1}{m_s} & -\frac{1}{I_{xx} l_f} & 0 & 0 & -\frac{1}{m_{uf}} & 0 & 0 & 0 \\ \frac{1}{m_s} & \frac{l_r}{I_{xx}} & 0 & 0 & 0 & -\frac{1}{m_{ur}} & 0 & 0 \\ \frac{b_{sf}}{m_s} & -\frac{l_f b_{sf}}{I_{xx}} & k_{sf} & 0 & -\frac{b_{sf}+b_{tf}}{m_{uf}} & 0 & -k_{tf} & 0 \\ \frac{b_{sr}}{m_s} & \frac{b_{sr} l_r}{I_{xx}} & 0 & k_{sr} & 0 & -\frac{b_{sf}+b_{tf}}{m_{ur}} & 0 & -k_{tr} \\ 0 & 0 & 0 & 0 & \frac{1}{m_{uf}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{m_{ur}} & 0 & 0 \end{bmatrix} \tag{3.10}$$

3.4 Full Car

When both the pitch and roll motions in a single vehicle dynamics simulation becomes important a full car model is required. This is effectively two half-car models whose bodies are tied together such that the vehicle is allowed to roll. Figure 3.5 shows an illustration of a linear full car model. A similar development approach was conducted as in previous sections using the bond graph shown in figure 3.6 to develop the matrices A and B. Due to the size of these matrices they are included in Appendix A. All the states are listed in equation 3.12.

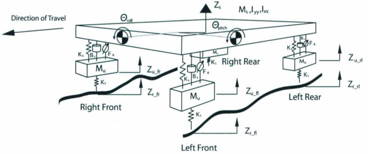


Figure 3.5: Full Car Model

$$\mathbf{x} = \begin{bmatrix} v_{m_s} \\ \omega_{I_{xx}} \\ \omega_{I_{yy}} \\ q_{ks_{fr}} \\ q_{ks_{fl}} \\ q_{ks_{rr}} \\ q_{ks_{rl}} \\ v_{mu_{fr}} \\ v_{mu_{fl}} \\ v_{mu_{rr}} \\ v_{mu_{rl}} \\ q_{kt_{fr}} \\ q_{kt_{fl}} \\ q_{kt_{rr}} \\ q_{kt_{rl}} \end{bmatrix} \tag{3.12}$$

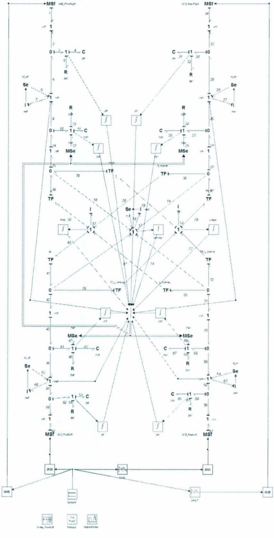


Figure 3.6: Full Car Bond Graph

3.5 Vehicle Parameters

The work by Darus et al. [33] studied a full car optimal controller for an average vehicle. It is one of the few asymmetric full car parameter sets available in the literature. In order to supplement this, the parameters of a compact car are utilized. The motivation for this relates to decoupling criteria outlined in chapter 2. When a compact car is laden with only one passenger the vehicle can satisfy the first coupling criteria. However, in the event that the vehicle is laden with a driver and five passengers it falls outside of the 20% requirement outlined Krotlica et al. [10] to be considered decoupled. This requirement would normally necessitate the use of a higher complexity controller. A combination of both these vehicle parameters are used in the development of the quarter and half car optimal controllers. All the vehicle parameters are presented in table 3.1.

Table 3.1: Vehicle Parameters

Item	Darus [33]	Small	Small w/pass	unit
Sprung Mass	1500	1054	1360	kg
Front Unsprung Mass	59	30	30	kg
Rear Unsprung Mass	59	30	30	kg
Pitch Inertia	2160	1326	1536	kg m^2
Front Suspension Stiffness	35	12.8	12.8	kN/m
Rear Suspension Stiffness	38	20.9	20.9	kN/n
Front Damping	1000	850	850	N$\cdot$ s/m
Rear Damping	1100	1565	1565	N$\cdot$ s/m
Dist Front axle to CG	1.4	0.953	1.138	m
Dist Rear axle to CG	1.7	1.422	1.237	m
Front Tire Stiffness	190	97.5	97.5	kN/m
Rear Tire Stiffness	190	97.5	97.5	kN/m

Chapter 4

Optimal Linear Quadratic Regulator Controller

4.1 Overview

The idea of a controlled system extends back to historical mechanical apparatus such as the flyball governor. The heart of a controller is the idea that there is an ideal state for the system to be in, and with some sort of controller and method of actuation, it will be in this state. If the system is not in this ideal state, then it is not performing at optimum efficiely.

From this has grown the idea of measuring these desired states and finding methods of actuation to achieve them. The goals of active suspension are broader but confined by some basic principles. The tire is the interface to the road and as such it is very stiff and difficult to adjust quickly. The vehicle body is a suspended system and as a result it cannot truly be actuated by anything in the "stationary" world.

The only viable place where an actuator can be placed is between the unsprung and sprung masses. If the goal was to control only the displacement of this, or potentially the velocity, then the control strategy would be very simple. A PID (proportional - integral - derivative) controller could be used to achieve this goal [34].

The actual goals of active suspension relate to the driver and occupants, the terrain, and the type and purpose of the vehicle. Loosely stated, in normal driving situations the sprung acceleration is the real goal as it relates to occupant comfort as people do not wish to be bounced around. If a vehicle is traversing over rough terrain at speed then handling might need to be improved and this can be done through holding tire displacement constant and minimizing the tire velocity. The tire effectively acts as a spring with the resulting traction related to this normal force. This available traction and relation is commonly calculated using The Magic Tire Formula [20] of Pacejka. If this force is changing, then in one instant when an operator believes the vehicle has traction but does not, the tire can break free and slide. Once the normal force is increased, if the tire is already sliding, it may not necessarily regain traction. From the operation perspective it is best to keep this as consistent as possible.

Vehicles such as "Rock Crawlers" utilize huge suspension travels but still reach their limits due to soft suspensions. Controlling the orientation of these vehicles to keep the body level would provide ability to transverse rougher terrain. Formula style race cars, on the other hand, may need to operate at a set distance from the ground for maximum down force and as a result need to control suspension travel as well. They may be on a rougher track, such as the road race held in Monaco and thus controlling suspension deflection may be a concern coupled with sprung mass velocity.

There are many other scenarios but the primary areas of investigation have been the areas of ride quality and road holding. Ride quality is easily defined as sprung mass acceleration, but road holding has seen many different definitions and weightings but generally is regarded as a minimizing of the tire (unsprung) velocity and deflection more so than the suspension deflection and sprung velocity. As Butsuen references in his thesis [35], there are many techniques available for control but Optimal Linear Quadratic Regulators suit the system goals.

4.2 Linear Quadratic Regulator

In order to determine optimal gains using the LQR method, a performance index is required. The method is described in many texts such as [36, 37, 38]. Initially, a performance index is the integral over time of several factors which is intended to be minimized. For a linear system, such as that described by equation 4.1 where matrix A describes the system and matrix B the inputs. A performance index can be described with positive semi-definite symmetric matrices as in equation 4.2 with terms for all the state variables multiplied together, as well as a similar combination for the inputs. An optimal gain matrix, K, can be assumed in equation 4.3 where K is the gain matrix and x is the state variable matrix. Under the idea of the LQR system, it has been found that the matrix K can be described as equation 4.4 referencing the solution, P, to the matrix Riccati equation shown in equation 4.5.

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{4.1}$$

$$J = \int_0^\infty (\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u})dt \tag{4.2}$$

$$\mathbf{u} = \textbf{-}\mathbf{K}\mathbf{x} \tag{4.3}$$

$$\mathbf{K} = \mathbf{R}^{-1}\mathbf{B}^T\mathbf{P} \tag{4.4}$$

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} - \mathbf{P}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{P} + \mathbf{Q} = 0 \tag{4.5}$$

In the case of active suspension it is the body acceleration and road holding that are competing objectives. We weight the importance of these factors for a desired result and various gains can be calculated using the described method to create a full state feedback controller for a force actuator. However, due to the complexity of the Multiple Input Multiple Output System (MIMO) it becomes more difficult to solve. Matlab [39] was unable to solve the half car controller scenario with either of its LQR or CARE commands. Octave (an open source Matlab competitor) was able to solve numerically for gains, but the gains for the half car model proved to be unstable. 3Sigma [40] provided a Maple [41] toolbox plugin that was able to solve numerically for stable gains for the half car model. It utilized additional techniques and fifty decimal place internal precision to which 3Sigma attributed its ability to solve the solution. The solving techniques used in 3Sigma and many other solvers has been described in the paper by Arnold and Laub in [42]. However, there was no numerical solution found to the full car model. This is generally found to be a limiting factor in the literature [8]. Hrovat presents that there have been some numerical solutions available but a reduced order model is required [8], reducing it from 7 DOF to a more manageable 6 DOF. This limits the ability to investigate a full controller comparison to half car and quarter car controllers.

4.3 Quarter Car

Butsuen [35] originally developed the performance index in equation 4.6 which has been used extensively to optimize quarter car controllers [43, 44, 45]. The performance index has weights ρ_1 through ρ_4 which are relative to the body acceleration $\ddot{z}_s$. As a result, heavily weighted body acceleration requires that the ρ terms are below unity, while penalizing for other factors requires weightings above unity.

$$J = \int_0^\infty \ddot{z}_s^2 + \rho_1(z_s - z_u)^2 + \rho_2\dot{z}_s^2 + \rho_3(z_u - z_r)^2 + \rho_4\dot{z}_u^2 dt \tag{4.6}$$

As this is a linear time invariant system, $\ddot{z}_s$ is a direct function of the state variables. This results in a more complicated weighting function. In order to solve the Riccati equation for optimal gains the expanded performance index is needed in the Matrix form:

$$J = \int_0^\infty (\mathbf{x}^T Q\mathbf{x} + 2\mathbf{x}^T N\mathbf{u} + \mathbf{u}^t R\mathbf{u})dt \tag{4.7}$$

The first step in developing the weighting matrix, Q, is to setup the performance index. For the quarter car, the original equation by Butsuen will be used. In terms of the performance index equation $z_s - z_u$ is the suspension deflection q_{ks}, and $z_u - z_r$ is the tire deflection q_{kt}. The sprung mass velocity, $\dot{z}_s$, is v_s and the unsprung mass velocity, $\dot{z}_u$, is v_u. Substituting the sprung mass acceleration from the state equation 3.2 into equation 4.6, the performance index becomes equation 4.8.

$$\begin{aligned} J = \int_0^\infty & -2\frac{b_s v_{m_s} f(t)}{{m_s}^2} - 2\frac{q_{ks} k_s f(t)}{{m_s}^2} - 2\frac{f(t)\,g}{m_s} \\ & -2\frac{v_{m_u} {b_s}^2 v_{m_s}}{{m_s}^2} + 2\frac{v_{m_u} b_s f(t)}{{m_s}^2} - 2\frac{q_{ks} k_s v_{m_u} b_s}{{m_s}^2} \\ & +2\frac{q_{ks} k_s b_s v_{m_s}}{{m_s}^2} + 2\frac{q_{ks} k_s g}{m_s} - 2\frac{v_{m_u} b_s g}{m_s} \\ & +2\frac{b_s v_{m_s} g}{m_s} + \frac{{q_{ks}}^2 {k_s}^2}{{m_s}^2} + \frac{{v_{m_u}}^2 {b_s}^2}{{m_s}^2} \\ & +\frac{{b_s}^2 {v_{m_s}}^2}{{m_s}^2} + \frac{f(t)^2}{{m_s}^2} + g^2 + \rho_1 {q_{ks}}^2 \\ & +\rho_2 {v_{m_s}}^2 + \rho_3 {q_{kt}}^2 + \rho_4 {v_{m_u}}^2 \end{aligned} \tag{4.8}$$

Writing J in the form of equation 4.7 requires matching coefficient terms. It should be noted that many of the terms are multiplied by two. This becomes beneficial as the matrix is symmetric. Consider the equation:

$$-2\frac{v_{m_u} {b_s}^2 v_{m_s}}{{m_s}^2}$$

This equation is made up of state variables v_{m_u} and v_{m_s}. The coefficients in the weighting matrix are:

$$\mathbf{x}^T \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} & \alpha_{14} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} & \alpha_{24} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} & \alpha_{34} \\ \alpha_{41} & \alpha_{42} & \alpha_{43} & \alpha_{44} \end{bmatrix} \mathbf{x} \tag{4.9}$$

When multiplied through the matrix $\mathbf{x}^T\mathbf{Q}\mathbf{x}$ becomes:

$$\begin{bmatrix} \alpha_{11}q_{ks}^2 & \alpha_{12}q_{ks}v_{ms} & \alpha_{13}q_{ks}q_{kt} & \alpha_{14}q_{ks}v_{mu} \\ \alpha_{21}q_{ks}v_{ms} & \alpha_{22}v_{ms}^2 & \alpha_{23}v_{ms}q_{kt} & \alpha_{24}v_{ms}v_{mu} \\ \alpha_{31}q_{ks}q_{kt} & \alpha_{32}v_{ms}q_{kt} & \alpha_{33}q_{kt}^2 & \alpha_{34}q_{kt}v_{mu} \\ \alpha_{41}q_{ks}v_{mu} & \alpha_{42}v_{ms}v_{mu} & \alpha_{43}q_{kt}v_{mu} & \alpha_{44}v_{mu}^2 \end{bmatrix} \tag{4.10}$$

From this it can be seen that there exists a $\alpha_{24}v_{ms}v_{mu}$ and a $\alpha_{42}v_{ms}v_{mu}$, but there only exists one term in the performance index equation. Since the weighting matrix is symmetric we divide the term equally over both weighting terms in the **Q** matrix. This develops the **Q**, **N**, and **R** matrices shown in equations 4.11, 4.12, and 4.13. Matching coefficients is suitable for simple matrices but for more complicated control systems a more efficient method to match the coefficients is proposed in the next section.

$$\mathbf{Q} = \begin{bmatrix} \frac{k_s{}^2}{m_s{}^2} + \rho_1 & \frac{k_s b_s}{m_s{}^2} & 0 & -\frac{k_s b_s}{m_s{}^2} \\ \frac{k_s b_s}{m_s{}^2} & \frac{b_s{}^2}{m_s{}^2} + \rho_2 & 0 & -\frac{b_s{}^2}{m_s{}^2} \\ 0 & 0 & \rho_3 & 0 \\ -\frac{k_s b_s}{m_s{}^2} & -\frac{b_s{}^2}{m_s{}^2} & 0 & \frac{b_s{}^2}{m_s{}^2} + \rho_4 \end{bmatrix} \tag{4.11}$$

$$\mathbf{N} = \begin{bmatrix} -\frac{k_s}{{m_s}^2} \\ -\frac{b_s}{{m_s}^2} \\ 0 \\ \frac{b_s}{{m_s}^2} \end{bmatrix} \tag{4.12}$$

$$\mathbf{R} = \begin{bmatrix} {m_s}^{-2} \end{bmatrix} \tag{4.13}$$

Two optimal quarter car controllers will be used on a half car model, one on the front and one on the rear. The control gains are based on the four quarter car states at each end with sprung mass pitch acceleration being resolved to vertical accelerations of the front and rear ends of the vehicle.

4.4 Half Car

The half car performance index is developed in an identical manner to that of the quarter car model. First a performance index is defined shown in equation 4.15 . The vertical acceleration is still left as the reference of all other weighting terms of ρ_1 through ρ_9. Both the sprung mass heave acceleration, $\ddot{z}_s^2$, and the sprung mass rotational acceleration, $\ddot{\theta}_s^2$, can be written in terms of the state variables and substituted into the performance index.

$$\begin{aligned} J &= \int_0^\infty \ddot{z}_s^2 + \rho_1 \ddot{\theta}_s^2 + \rho_2 (z_{sf} - z_{uf})^2 + \rho_3 \dot{\theta}_s^2 \\ &+\rho_4 (z_{uf} - z_{rf})^2 + \rho_5 v_{uf}^2 + \rho_6 (z_{sr} - z_{ur})^2 + \rho_7 v_s^2 \\ &+\rho_8 (z_{ur} - z_{rr})^2 + \rho_9 v_{ur}^2 dt \end{aligned} \tag{4.14}$$

The equation can again be expanded and put in the form of equation 4.7. Due to the size of the matricies, the individual terms are listed in Appendix B. As previously alluded to, partial derivatives can be used matching of coefficients more quickly. If the performance index has its derivative taken with respect to any two variables the remainder should be the term left in the corresponding matrix position. The matrix is symmetric along the diagonal and as a result the derivatives need to be divided by two. The terms in the diagonal are halved as well as there is a squared term in the initial equation. As a result all matrices can be developed in a similar fashion to that shown in equation 4.15. This operation was automated using the Maple software package and allows to match coefficients for much larger matrices. When other elements are introduced, such as an electrical model of the actuator, this becomes a very useful technique as there are more states, and these states are not generally associated with this type of system.

$$\mathbf{x}^T \begin{bmatrix} \frac{\partial^2 J}{2\partial x_1^2} & \frac{\partial^2 J}{2\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 J}{2\partial x_1 \partial x_8} \\ \frac{\partial^2 J}{2\partial x_2 \partial x_1} & \frac{\partial^2 J}{2\partial x_2^2} & \cdots & \frac{\partial^2 J}{2\partial x_2 \partial x_8} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 J}{2\partial x_8 \partial x_1} & \frac{\partial^2 J}{2\partial x_8 \partial x_2} & \cdots & \frac{\partial^2 J}{2\partial x_8^2} \end{bmatrix} \mathbf{x} \tag{4.15}$$

4.5 Parameter Weightings

The performance indices are set such that they relate to three cases. A ride quality weighted case has all relative weights (relative to vertical acceleration) less than one. There is a modified pitch case such that the pitch acceleration is weighted higher. By setting the pitch acceleration term higher it violates the second criteria of Krotilica and Hrovat for decoupling. This effectively allows for an investigation into the sensitivity of the original decoupling requirement outlined in chapter 2, equation 2.3. Finally a road holding case has all relative weights greater than one with pitch acceleration left as one. The weights used for simulation are presented in table 4.1.

Table 4.1: Performance Index Weights

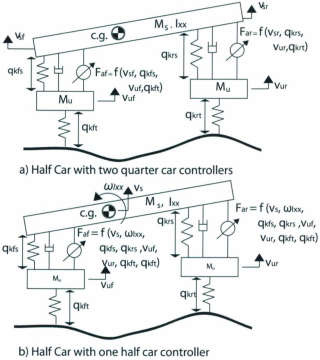
Parameter	1/2 Car	1/4 Car	Case 1	Case 2	Case 3
	Coefficient	Coefficient	Ride Quality	Ride Quality Pitch Weighted	Road Holding
$\ddot{z}_s$	unity		1	1	1
$\ddot{\theta}_{Ixx}$	ρ_1		1	20	1
v_{m_s}	ρ_7	ρ_2	0.16	0.16	1
$\dot{\theta}_{Ixx}$	ρ_3		0.16	0.16	1
$q_{k_{fs}}$	ρ_2	ρ_1	0.4	0.4	4.0E5
$q_{k_{rs}}$	ρ_6		0.4	0.4	4.0E5
$v_{m_{fu}}$	ρ_5	ρ_4	0.16	0.16	100
$v_{m_{ru}}$	ρ_9		0.16	0.16	100
$q_{k_{ft}}$	ρ_4	ρ_3	0.4	0.4	1.0E6
$q_{k_{rt}}$	ρ_8		0.4	0.4	1.0E6

Chapter 5

Simulation comparison of 1/4 car and 1/2 car

5.1 Controller Overview

This chapter details the comparison of two control techniques on a half car model. As previously describe the conventional idea of using a half car controller on a half car model is presented. As a comparison, the idea of using two quarter car controllers on the half car model is described. The method of implementing these two controllers is explained in section 5.2. From this, there is an overview in section 5.3 of the results which are expanded in section 5.4 discussing discreet bumps as an input, and in section 5.5 discussing the random road input representing a poor quality road.



a) Half Car with two quarter car controllers

b) Half Car with one half car controller

Figure 5.1: Quarter Car And Half Car Controllers On A Half Car Model

5.2 1/4 Car Controller on 1/2 Car Model

Normally it is appropriate for a quarter car optimal controller to be used with a quarter car model. By extension, a half car optimal controller is used with a half car model. There is another option, a half car pitch model with two quarter car controllers, one for the front suspension and one for the rear suspension, shown in figure 5.1a. The decoupling criteria mentioned previously from Krtolica et al. [10] relates the use of this type of controller scheme. The two quarter car controllers both act independently, and indirectly on the pitch mode, and neither has state feedback from the other. These are essentially two individual controllers developed using weight distribution to divide the sprung mass between the front and rear, and the technique for the quarter car described in chapter 4. A half car pitch model with an optimal half car controller is shown in figure 5.1b. This model has the output of the front and rear actuator based on all the vehicle states including pitch.

The two quarter car controllers have to be designed with a different performance index, as described in chapter 4, from that of the half car controller. However, both models can be tested on the same virtual roadway and the half car performance index can be used to evaluate both models. In order to utilize a reduced order complexity controller on a higher order model, the performance indices will be different. There is no direct way to translate weightings but it is possible to leave the component and acceleration terms weighted identically. A quarter car based performance index will not have a pitch or roll acceleration term as the vehicle mass is divided over each corner and used to solve the simple quarter car controller problem. For instance, two quarter car based controllers, one at the front and rear of a vehicle, will control pitch indirectly by minimizing vertical acceleration at each end.

The two models are tested with a variety of weights and on a combination of road surfaces for each weighting with several different vehicle properties. Repeated discrete bumps form one surface, while the second surface uses a rough random road model.

The results indicate that it takes a significantly high performance index weight on pitch acceleration to warrant the use of a half car controller. It will also be shown that even when the decoupling criteria in equation 2.2 are not met for certain vehicles, such as fully loaded compact cars, that quarter car controllers can still provide superior or equivalent performance for both ride quality and road holding.

5.3 Simulation Results and Analysis

The half car performance index described earlier is used to calculate a live performance index for simulations in the time domain. This is conducted in Bond Graphs utilizing a custom code block in 20-Sim [46] with the code shown below.

```
parameters
real rho = 1.0;
real rho1 = 1.0;
real rho2 = 400000.0;  //rho1 1/4 suspension deflection
real rho3 = 1.0;
real rho4 = 1.0e6; //rho3 1/4 tire deflection
real rho5 = 100.0; //rho4 1/4 unsprung velocity
real rho6 = 400000.0;
real rho7 = 1.0; //rho2 1/4 sprung velocity
real rho8 = 1.0e6;
real rho9 = 100.0;
equations
   output = int(rho * Accel_Heave^2) +
int(rho1 * Accel_Pitch^2) +
int(rho2 * Sus_Deflect_Front^2) +
int(rho3 * Vel_Pitch^2) +
int(rho4 * Tire_Deflect_Front^2) +
int(rho5 * Vel_Un_Front^2) +
int(rho6 * Sus_Deflect_Rear^2) +
int(rho7 * Vel_Heave^2) +
int(rho8 * Tire_Deflect_Rear^2) +
int(rho9 * Vel_Un_Rear^2);
```

The states are input into the performance index and integrated during two different scenarios; repeated discrete bumps and a random road profile over a 200 second. When the two quarter car controllers are used on the half car model, all the states are input into the half car performance index with the half car weightings. All the acceleration and component weightings are identical for the development of each LQR optimal controller. The code block has embedded comments such that they can easily be identified.

The results show an interesting finding that the half car controller is not superior in all cases. It was originally assumed that when the decoupling criteria of [10] was

Table 5.1: Superior Controller for Scenarios

Case	Ride	Ride w/Pitch	Road Holding
ISO8606 'poor'	Two 1/4	1/2 Car	1/4 or 1/2
Repeated Bumps	Two 1/4	1/2 Car	1/4 or 1/2

not met, the half car controller would prove consistently superior. A summary of the results are shown in table 5.1. The two cases to be discussed, an ISO8606 'poor' road case, which is a standard for rating road quality from very good to very poor, and a repeated bumps cases. With the exception of very significant pitch weighting, the quarter car controllers are superior or equivalent to the more complex half car controller.

5.4 Repeated Discrete Bumps

The discrete bumps are modeled with a continuous motion profile shown in figure 5.2. When the simulation was run for 15 seconds with zero intitial conditions the three suspensions produced the performance index values shown in table 5.2 for the parameters in [33]. The same profile was sent to the front and rear wheels with a delay corresponding to a forward velocity of 72 km/hr (20 m/s).

Table 5.2 shows significant differences between the performance of the passive, half car controller, and quarter car controllers. For the ride quality case the quarter car controllers are an order of magnitude lower, while the half car controller is a fifth of the magnitude of the passive suspension. When the pitch acceleration weighting is increased the quarter car responses remain the same due to lack of direct pitch input,

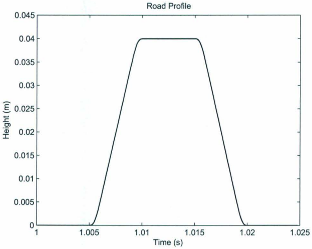


Figure 5.2: Road Bump Profile

Table 5.2: Performance Index Results for Repeated Bumps

Case	Passive	Half Car	Quarter Car
Ride Quality	6.80E6	1.23E6	2.43E5
Ride Quality w/Pitch	1.12E8	1.54E6	4.06E6
Road Holding	6.80E6	3.03E8	7.51E7

leading to the half car controller giving marginally superior performance.

The road holding case presents a challenge in determining the superior controller using the performance index. In order to increase the road holding ability, vertical and pitch acceleration performance is sacrificed. This is effectively stating that accelerations are not as important as the unsprung velocities and suspension and tire deflection. The absolute magnitudes of the acceleration terms are much higher than the other states such as tire deflection, and the terms are squared in the traditional performance index. The acceleration terms dominate unless other weighting factors are made extremely high, causing controller instability. The performance index has been modified by separating out the velocities and the deflections, thus focusing on the states most relevant to road holding and is shown in equation 5.2.

$$\begin{aligned} J &= \int_0^\infty \rho_2(z_{sf}-z_{uf})^2+\rho_3\dot{\theta}_s^2+\rho_4(z_{uf}-z_{rf})^2 \\ &\quad +\rho_5 v_{uf}^2+\rho_6(z_{sr}-z_{ur})^2+\rho_7 v_s^2+\rho_8(z_{ur}-z_{rr})^2 \\ &\quad +\rho_9 v_{ur}^2 dt \end{aligned} \tag{5.1}$$

The new performance index results for the same 15 second run are shown in table 5.3. Both the half and quarter car controllers reduce the performance index with the half car controller being only slightly better. The response for hitting a single bump is shown in figure 5.3. The responses of the quarter and half car controllers are very similar but it can be seen that as the rear wheel hits the bump that there is a ripple in the response for the half car controller, while the two quarter car controllers remain more level. The model with two quarter car controllers has a response preferable for road holding as it leads to more consistent tire forces. If the original performance

Table 5.3: Modified Performance Index Results for Repeated Bumps

Case	Passive	Half Car	Quarter Car
Road Holding	1354.1	937.9	956.1

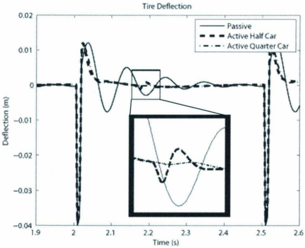


Figure 5.3: Tire Deflection(Road Holding With Repeated Bumps)

index results are referred to in table 5.2, this improvement in performance for the half car controller is at a cost of vertical and pitch accelerations. As a result the quarter car controller may be the preferable option.

5.5 Random Road Profile

Following Tyan et al., a random road can be generated using a first order filter model in conjunction with a random number generator [47]. This allowed a random road of a pre-defined roughness, based on the ISO 8606 classification, to be generated and

Table 5.4: Rand Road Performance Index Results

Case (DARUS)	Passive	Half Car	Quarter Car
Ride Quality	2.20E9	2.16E8	6.86E7
Ride Quality w/Pitch	2.75E10	3.95E8	9.44E8
Road Holding	2.20E9	6.64E9	4.62E9

stored. A rough road with a rating of "poor" to "very poor" was created for testing. As with the repeated bumps, a velocity of 72 km/hr (20 m/s) is used. This should be a sufficiently high velocity on a difficult road which should make apparent the differences in the controller performance. A sample of the road profile is shown in figure 5.4.

5.5.1 Ride Quality

The performance index results show the two quarter car controllers are superior to that of the half car controller for ride quality. Figure 5.5 shows a sample from the simulation that indicates that both controllers are significantly superior to the passive, but on average the quarter car response is of lower magnitude. The two quarter car controllers actually control pitch better than the half car controller as shown in figure 5.6. The normal trade-off for reduction in sprung mass acceleration (increased deflections and velocities) is observed to be conserved in figure 5.7.

When the controllers were tested for vehicles with coupled and decoupled suspensions, depending on loading, the results were very similar. The performance index results are summarized in tables 5.5 and 5.6 for a 40 second simulation time and again verify the effectiveness of the quarter car controllers.

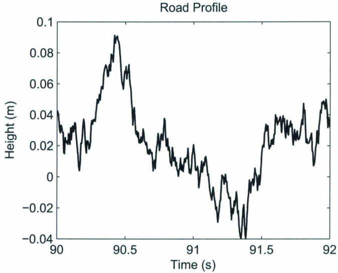


Figure 5.4: Sample of Random Road Profile

Table 5.5: Compact Car Performance Index Results(Decoupled)

Case	Passive	Half Car	Quarter Car
Ride Quality	8.82E7	6.60E7	4.20E6
Ride Quality w/Pitch	2.86E8	9.72E6	1.59E7
Road Holding (modified)	28325	13465	13672

Table 5.6: Compact Car w/Passengers Performance Index Results (Coupled)

Case	Passive	Half Car	Quarter Car
Ride Quality	8.90E7	8.60E6	6.39E6
Ride Quality w/Pitch	2.71E8	1.46E7	2.67E7
Road Holding (modified)	31275	14122	14189

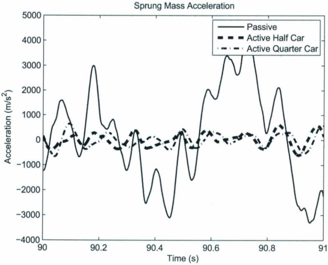


Figure 5.5: Sprung Mass Vertical Acceleration (Ride Quality)

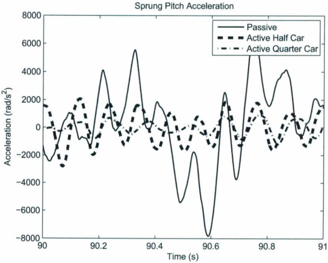


Figure 5.6: Sprung Mass Pitch Acceleration (Ride Quality)

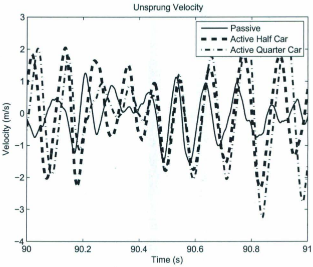


Figure 5.7: Unsprung Mass Velocity (Ride Quality)

5.5.2 Ride Quality With Increased Pitch Weighting

When the weight for sprung mass pitch acceleration is increased, the two quarter car controllers are inferior to that of the half car controller. While the vertical acceleration magnitudes for both controllers are similar, the half car controller appears to have reduced magnitudes for pitch accelerations as seen in figure 5.8.

When the compact car scenarios are taken into consideration along with a lower pitch weighting (ρ_1) of 5, the increased pitch weight performance index for the half car controller is 61% of the quarter car controller performance index for the decoupled vehicle. For the vehicle with coupled suspensions motions the fraction decreases to 55%, showing that the performance gains of the half car controller increase with coupling when pitch motion suppression is given higher weighting.

Table 5.7: Matched Performance Index Results

Case	Passive	Half Car	Quarter Car	ρ_1
Decoupled	1.67E8	8.47E7	8.59E7	2.6
Coupled	1.34E8	1.15E7	1.15E7	2.0

Through adjusting the pitch weighting term, ρ_1, the performance of a half car controller can be matched to that of the quarter car controller. From the results in table 5.7, an inverse relationship for quarter car controller performance can be seen. When the decoupled vehicle is considered, the pitch weight can be increased 92% relative to the heave acceleration through equation 2.3. When the coupled model is considered this drops to 42%. Thus, as the mass-inertia coupling increases the available range of allowable pitch weights for which the quarter car controller would be superior decreases. Referring to 2.3, this indicates that the pitch acceleration weighting term on the right hand side can be approximately 40% larger than the left hand side of the equation.

5.5.2.1 Road Holding

The responses of both controllers did show improvements for road holding through minimizing tire deflections and unsprung velocities of which a sample is shown in figure 5.9. However, it is again noted that the performance index had bias due to the magnitudes of the large accelerations. Separation of the important parameters provides a better view of the results. Table 5.8 shows the performance indices without the vertical or pitch acceleration terms.

Like the repeated bumps scenario, the quarter car controllers shows similar perfor-

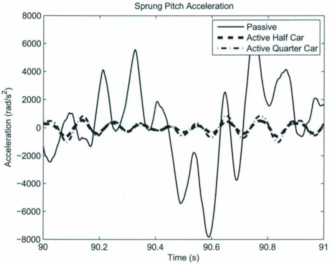


Figure 5.8: Sprung Mass Pitch Acceleration (Ride Quality With Increased Pitch Weight)

Table 5.8: Modified Performance Index Results

Case (DARUS)	Passive	Half Car	Quarter Car
Road Holding	2.36E5	6.87E4	7.15E4
Compact Decoupled	28325	13465	13672
Compact Coupled	31275	14122	14189

mance to that of the half car controller in all tested models. Similar to ride comfort, the quarter car controllers are providing very good performance so long as they are within certain limits for pitch acceleration weighting. Road holding ability is more concerned with minimizing other aspects and would not be expected to have higher magnitude pitch acceleration weightings.

5.6 Conclusions

The development of quarter car and half car models and controllers in the outlined scenarios highlighted situations where the model complexity required for a controller was not easy to predict intuitively. In order to determine the required complexity for a controller, a performance index was generated. Using optimal gains from the linear quadratic regulator technique for two half car models were tested.

When vertical accelerations were required to be minimized, a quarter car controller at both ends of a half car model performed better than a half car based controller for coupled and decoupled models. When pitch acceleration suppression was most important, the quarter car controllers could not differentiate the vertical heaving from the pitching and the half car controller was superior. It was found that the pitch

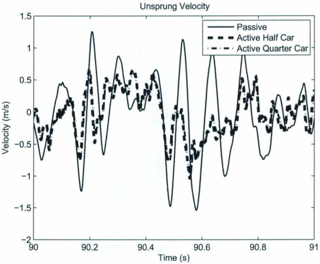


Figure 5.9: Unsprung Velocity (Road Holding)

acceleration weights could be approximately 40% higher, using equation 2.3, than heave acceleration and still maintain identical performance even when a model was coupled. This increases to 90% for decoupled models. For road holding, a modified performance index showed that the performance of the half car and quarter car controllers were nearly identical for all decoupled and coupled scenarios.

Generally speaking, the individual quarter car controllers provided similar response at a reduced complexity level for both ride quality and road holding cases with the exception of when strong pitch control was required. This expands the usable range for which individual quarter car controllers can be utilized on vehicle models especially for more realistic simulations.

Further, this provided the basis for motivation the development of a 1/4 car test rig as it showed how effective a quarter car suspension can be and that debugging and tuning it would have direct implications on higher DOF future test apparatus.

Chapter 6

Simulation of Five State 1/4 Car Controllers

6.1 Overview

As discussed by Hrovat [8], original attempts to develop test beds based on LQR control schemes were of poor quality especially when using hydraulic actuators. Gysen's work and background [48, 7, 30] extols the limitations of pneumatic and electro-hydraulic actuators citing a bandwidth of approximately 1 Hz, while his team's tubular permanent-magnet actuator, TPMA, or Lee's direct-drive tubular linear brushless permanent-magnet motor LBPMM [31], show much greater bandwidth with greater success for both in LQR and LQG control techniques.

The Bond Graph modelling technique as shown previously is a highly effective method of modelling any system. One of the more powerful features is its ability to model interconnecting systems. For instance an electro-hydraulic system is comprised of electrical, hydraulic, and mechanical systems that are all coupled together. Conventional theory for active suspensions dictates the idea that the mechanical system governs the force target, and a secondary controller is then utilized to match this force. With Bond Graphs it is easier to model the actuator and the system together as one linear system, and then, based on all state variables, develop a truly optimal controller.

6.2 Load Cell PID follower

The original 4 state LQR outlined in chapter 5, will produce a force as an output. As discussed earlier, force actuators are either motors or mechanisms that generally do not produce a force directly, but have a feedback mechanism. The conventional approach is outlined in the top loop of figure 6.1. The linearity between force and current shown by Gysen et al. [30] allows the loop to be altered for a low cost current sensor instead of the higher cost load cell shown in the middle loop. For illustrative purposes the 5 State LQR is shown at the bottom of the figure highlighting its shorter command structure.

6.3 Five State LQR

Similar to that of the original development of the 4 state quarter car model in chapter 3 using Bond Graphs, the 5 state LQR is derived in a similar manner, but with several main differences. According to Karnopp [32], an ideal motor is modelled as a gyrator with an inductance and a resistance for the winding. This relates the current to the

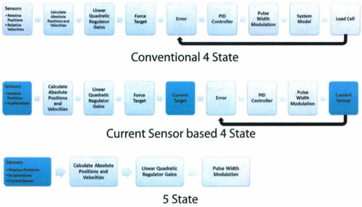


Figure 6.1: 4 State versus 5 State Flow Comparison

force, and the back electro motive force to the velocity and connects the mechanical and electrical components. The resulting new mechanical model is shown in figure 6.2. When incorporated into the bond graph this becomes the model shown in figure 6.3. As can be seen the inductor is providing the integral causality on the current, and the resistor is an energy dissipater representing heat lost proportional to the current.

As previously described, the linear system is given in equation 6.1. Using the bond graph method, and modifying the state variables to velocities and currents, give the system of equations 6.2. Of note is how the current is the only related state for the force actuator. As previously shown in the bond graph version of the system, the output from the MSE, or Modulated Source Effort, is not force but a voltage. This voltage can be easily targeted with any common PWM algorithm.

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{6.1}$$

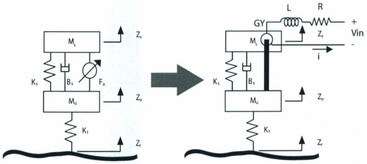


Figure 6.2: Four and Five State Mechanical Models

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 & -1 & 0 \\ -\frac{k_s}{m_s} & -\frac{b_s}{m_s} & 0 & \frac{b_s}{m_s} & \frac{r}{m_s} \\ 0 & 0 & 0 & 1 & 0 \\ \frac{k_s}{m_u} & \frac{b_s}{m_u} & -\frac{k_t}{m_u} & -\frac{b_s}{m_u} - \frac{b_t}{m_u} & -\frac{r}{m_u} \\ 0 & -\frac{r}{L_{coil}} & 0 & \frac{r}{L_{coil}} & -\frac{R_{coil}}{L_{coil}} \end{bmatrix} \begin{bmatrix} q_{k_s} \\ v_{m_s} \\ q_{k_t} \\ v_{m_u} \\ i_{coil} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ {L_{coil}}^{-1} \end{bmatrix} \begin{bmatrix} f(t) \end{bmatrix} \tag{6.2}$$

Initial attempts to reuse the original performance index shown in equation 6.3 were unsuccessful. The Riccati solvers could not reach a numerical conclusion as the current was not included in the performance index causing a zero value to occur in the performance index, Q. A requirement of the performance index in matrix form is that all values are positive semi-definite, which was not met. As a result the performance index was modified to include the current as a state and is penalized significantly less than all other states. In order for the Q matrix to be positive semi-definite and always increasing the performance index is modified for the 5 state to equation 6.4

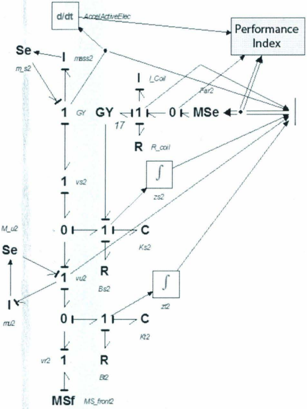


Figure 6.3: 5 State Bond Graph

Table 6.1: 5 State Simulation Parameters

Item	Value	Unit
Sprung Mass	4.8	kg
Unsprung Mass	2	kg
Suspension stiffness	172.8	N/m
Tire stiffness	8450	N/m
Suspension Damping	8.64	N·S/m

with the current term squared.

$$J = \int_0^\infty \ddot{z}_s^2 + \rho_1(z_s - z_u)^2 + \rho_2\dot{z}_s^2 + \rho_3(z_u - z_r)^2 + \rho_4\dot{z}_u^2 dt \tag{6.3}$$

$$J = \int_0^\infty \ddot{z}_s^2 + \rho_1(z_s - z_u)^2 + \rho_2\dot{z}_s^2 + \rho_3(z_u - z_r)^2 + \rho_4\dot{z}_u^2 + \rho_5 i_{coil}^2 dt \tag{6.4}$$

6.4 Ride Quality

Based on initial estimates of a scale vehicle, the parameters selected in order to simulate a 4 state controller versus a 5 state controller are shown in table 6.1. The simulations performance for two scenarios; a swept sine wave, and a random road with weightings shown in table 6.2. Using these weights, and the 3Sigma Riccati solver in Maple, the state feedback gains were calculated and are shown in table 6.3.

In terms of LQR based gains, the 4 state is the theoretical limit, thus it can be regarded as the ideal case.

Table 6.2: 5 State Performance Index Weights: Ride Quality

State	Parameter	value
$\ddot{z}_s$	unity	1
q_{k_s}	ρ_1	0.4
v_{m_s}	ρ_2	0.16
q_{k_t}	ρ_3	0.4
v_{m_u}	ρ_4	0.16
i_{coil}	ρ_5	1.0e-5

Table 6.3: Ride Quality Gains

Controller	q_{k_s}	v_{m_s}	q_{k_t}	v_{m_u}	i_{coil}	Feedback
4 State (Active)	-168.8	-2.2	1.3	6.7	-	-
4 State w/Proportional (Control)	-168.8	-2.2	1.3	6.7	-	50
5 State (Electric)	-11148.2	-173.5	52.3	451.8	658.6	-

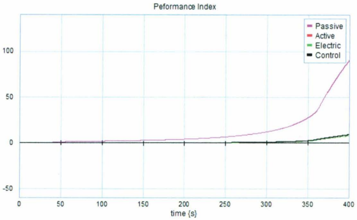


Figure 6.4: Performance Index Swept Sinewave Ride Quality

6.4.1 Swept Sine Wave

Using a swept sine wave from 0 to 62.83 rad/s (0 - 10 Hz) it is seen that the performance indices for a 3mm amplitude input relates to the graph in figure 6.4 where the performance of the 5 state and the 4 state are very closely matched. Even the 4 state with a feedback loop with a gain of 50 gives good performance. As before, the tires can lift off in this simulation, which is a unique feature, and once this wheel hop frequency is reached there is a marked decline in performance from the 4 state with a force feedback loop. From table 6.4, samples are chosen from the swept input. It is noticeable that there is very little difference between the active controllers and the 5 state.

Table 6.4: Performance Index Results: Swept Sinewave Ride Quality

Name	Controller	3 Hz	8 Hz	10 Hz
Passive	Passive	2.06421	16.19155	90.689455
Active	Theoretical Active	0.023565	1.13473	7.84353
Electric	5 State	0.023719	1.14640	7.92294
Control	Active with P controller = 50	0.027931	1.32560	9.14689

Table 6.5: Performance Index Results: Random Road Ride Quality

Name	Controller	Performance Index Value
Passive	Passive	5.6492
Active	Theoretical Active	1.0130
Electric	5 State	1.0208
Control	Active with P controller gain = 50	1.0359

6.4.2 Random Road

When the simulation is performed on a random road developed in a similar fashion to that used prior for the quarter car and half car comparison there was no noticeable difference in performance for a 200 second simulation. Examining the response of the sprung acceleration in figure 6.5 shows negligible differences. Even examining the response of the 5 state versus 4 state force actuator response, the 5 state and 4 state are nearly identical as seen in the sample in figure 6.6. When the performance index results are examined in table 6.5 or visually in figure 6.7, there is negligible difference between the 4 state and 5 state controllers, while the 4 state with the force feedback loop shows a slight decrease in performance.

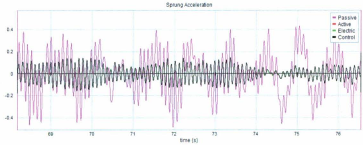


Figure 6.5: Sprung Acceleration Ride Quality

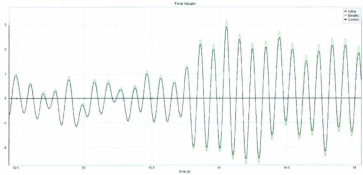


Figure 6.6: Ride Quality Force Requirement

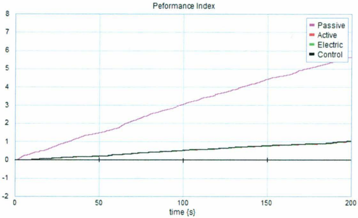


Figure 6.7: Performance Index Random Road

6.4.3 Control Delay

Control delays will result in force commands being implemented later than desired. Even with an FPGA there is a delay in processing and adjustment to the output to the H-bridge chip via pulse width modulation. PWM is further explained in chapter 8. For instance, utilizing a 512 Hz sample time would introduce a 0.00195 second delay. Once this delay is introduced, a feedback gain of 50 is unstable. It is found that the limit for the feedback gain is 1.1 for the proportional control as it is marginally stable at this value and the system becomes unstable very quickly as shown in figure 6.8 due to larger inputs. Using the maximum stable feedback gain of 1.05, the new results show how the 5 state is superior to the delayed feedback control in table 6.6. Having this delay for the control loop would be more representative of a real world implementation where the feedback loop would introduce even further delay in the system.

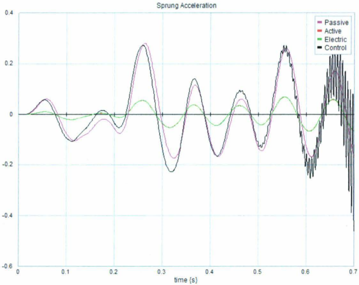


Figure 6.8: Random Road Unstable Proportional Gain K = 1.1

With a stable gain the impact to the acceleration is noticeable as seen in figure 6.9, but it is not near the theoretical limit. The 5 state in this regard shows greater promise for a real world implementation for ride quality.

6.5 Road Holding

6.5.1 Random Road

Using the same random road profile as the ride quality case, new LQR gains were calculated for both the 4 and 5 state controllers with values for the weights corresponding to road holding based on Rajamani and Butsuen's work [43, 35]. Table 6.7 shows the weightings, while table 6.8 shows the calculated gains based again on

Table 6.6: Performance Index Results: Random Road With Delay Ride Quality

Name	Controller	Performance Index Value
Passive	Passive	5.6492
Active	Theoretical Active	1.0130
Electric	5 State	1.0208
Control	Active with P controller gain = 1.05 with 0.00195s delay	3.6461

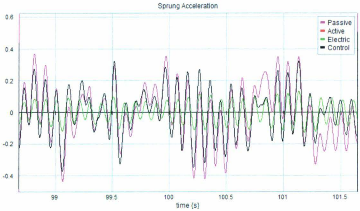


Figure 6.9: Acceleration Ride Quality With Delayed Controller

Table 6.7: 5 State Performance Index Weights: Road Holding

State	Parameter	value
$\ddot{z}_s$	unity	1
q_{k_s}	ρ_1	10000
v_{m_s}	ρ_2	100
q_{k_t}	ρ_3	10000
v_{m_u}	ρ_4	100
i_{coil}	ρ_5	1.0e-5
ft	ρ_6	1.0e-5
$(i_{coil} * ft)/2$	ρ_7	1.0e-5

Table 6.8: Road Holding Gains

Controller	q_{k_s}	v_{m_s}	q_{k_t}	v_{m_u}	i_{coil}	Feedback
4 State	307.2	76.2	-533.7	-42.9	-	-
4 State w/Proportional	307.2	76.2	-533.7	-42.9	-	50
5 State	20233.8	5012.0	-35044.0	-2819.2	658.9	-

3Sigma's Riccati solver in Maple.

It is interesting to note the results show the 5 state, the 4 state and the 4 state with a proportional feedback loop have near identical performance values which are all approximately half that of the passive suspension shown in table 6.9. Responses to suspension deflection and tire deflection examined in figures 6.10 and 6.11 show that there is virtually no difference between the control schemes. When the area is magnified as seen in figure 6.12, the differences in performance are well below

Table 6.9: Performance Index Results: Random Road Road Holding

Controller	Performance Index Value
Passive	124.993
Theoretical Active	61.354
Active with P controller = 50	61.260
5 State	61.354

experimental tolerances.

6.5.2 Control Delay

When a control delay is introduced as in the previous section of 1/512th of a second, the performance changes. A gain on the feedback controller of > 1 produced very unstable results, while a gain of 0.9 produced an even lower performance index value which is surprising. This can be seen in the response with a gain of 1.0 in the acceleration in figure 6.13 and in the performance index in figure 6.14.

An interesting side effect of introducing a delay is the 4 state with a force feedback loop actually achieved a lower performance index value as seen in the summary in table 6.10. This could be potentially due to a lag controller being more appropriate for the road holding case, and the artificial delay thus provides better real world performance. This should be investigated in the future.

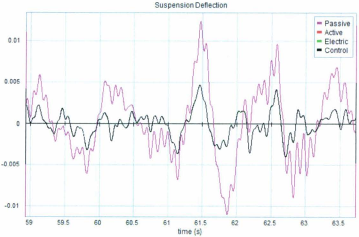


Figure 6.10: Suspension Deflection Road Holding

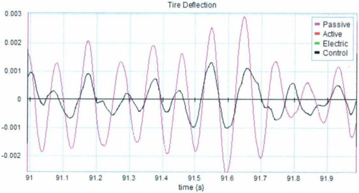


Figure 6.11: Tire Deflection Road Holding

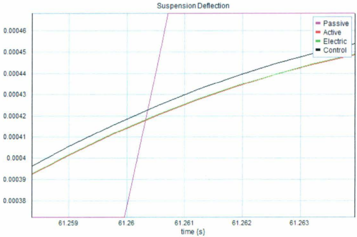


Figure 6.12: Suspension Deflection Detail Road Holding

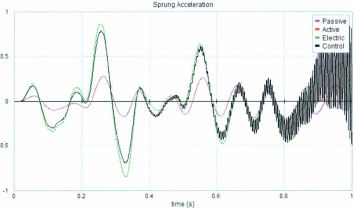


Figure 6.13: Road Holding Unstable Acceleration

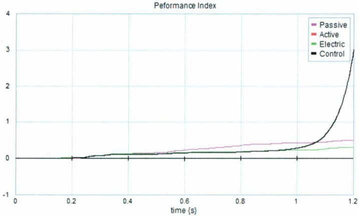


Figure 6.14: Performance Index Road Holding Unstable

Table 6.10: Performance Index Results With Delay: Random Road Road Holding

Controller	Performance Index Value
Passive	124.993
Theoretical Active	61.354
Active with P controller = 0.9	59.369
5 State	61.354

6.6 Conclusions

The two scenarios presented using a 5 state versus a 4 state LQR highlighted that there is potential improvement in the real world performance of a 5 state versus a 4 state LQR controller. This is due to there being only one controller instead of two for a real world implementation. Delays to the system were shown to cause instability very easily, and a major reduction in the maximum feedback gain values. Ride quality gains over 1.05 and with road holding gains larger than 0.9 caused huge instability. When the feedback gains were kept below these values for this set of model parameters the system maintained stability for long simulation times but could not achieve near the 4 state limits or the 5 state limits for ride quality. There is the slightly anomalous result that the delayed controller actually performed better in a road holding scenario and investigation into a lead or lag controller should be conducted to highlight if any other improvements are possible. It should be noted that operating near the outlined gains may prove to be difficult in a real world situation and could easily cause instability giving another potential advantage to the 5 state controllers.

Chapter 7

1/4 Car Test Apparatus Development

7.1 Overview

To test the real world benefits of active suspension, a physical testing apparatus is required. The theoretical implementations are valuable but are of unverified practical significance. The primary concern is the actuator, and that most theoretical implementations assume an ideal infinite force / bandwidth actuator with zero delay. Real implementations are less than ideal and Hrovat alludes to the difficulties surrounding implementation in his active suspension survey [8]. Concerns mainly surround the bandwidth of the actuators and Coulomb friction at key pivot points.

This chapter is broken down into several sections. Based on previous work there is a discussion on the requirements including the estimated weights and suspension parameters and how they relate to the general construction, in section 7.2. In section

7.3 an outline of provisions for further experiments is presented. The motivating factor is to not limit the design of a test apparatus and keep several possible future investigations open by requiring minimal modifications. In order to develop a working active suspension the main limitation has been the actuator itself, which is why section 7.4 discusses initial development of prototypes and available off the shelf actuators that were proposed. After the requirements, provisions and actuator are defined, the remainder of the required parts are specified and described in section 7.5. Finally the groundwork has been laid and a discussion of the actual design is found in section 7.6.

In order to give a better idea of what a quarter car test apparatus looks like, figure 7.1 of the finalized design has been included and can be referenced throughout for explanation.

7.2 Requirements

A quarter car linear test apparatus is to be built to simulate the simple quarter car shown in figure 7.2 which has been previously discussed and developed. It consists of a sprung mass m_s, an unsprung mass m_u, a suspension spring k_s, suspension damping b_s, and a tire stiffness k_t.

In order to keep the design as simple and expandable as possible a single square linear rail, THK 25 type, is used. The linear bearings have 4 threaded holes which provide a strong support for the apparatus. The guide blocks have recirculating ball bearings arranged such that the blocks can handle torques as well as normal forces and constrain all movement to one direction. This eliminates the need for multiple

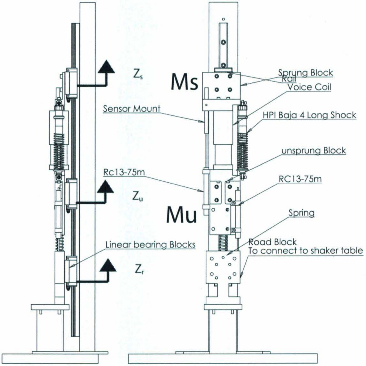


Figure 7.1: Quarter Car Test Apparatus Design

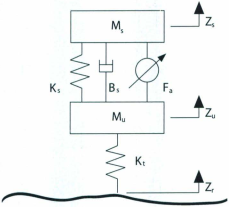


Figure 7.2: Quarter Car

rails and overall can reduce the bulkiness.

There are many constraints involving the design of such a test apparatus. Such constraints include:

1. Component Weights

2. Suspension Parameters

3. Ideal Tire Model

4. General Construction

7.2.1 Component Weights

In order to test a wide range of component parameters, it was necessary to be allow additional weight to be added to the sprung and unsprung components. This meant mounts were required for additional weights, or an easy way of connecting to the linear guide block mounts had to be designed. It also requires that the weights of the machined parts were to be kept as low as possible. At the initial conception it was apparent that the force actuator would be a significant mass, and depending on the arrangement, would significantly impact the design.

7.2.2 Suspension Parameters

The suspension parameters include both the spring stiffness and the damping term. It is permissible that the operator would wish to adjust the natural frequency of the suspension which impacts the suspension effectiveness for ride quality and road holding [43]. Assuming that the sprung mass, m_s, stays constant the stiffness must be adjusted to the appropriate term through equation 7.1. This gives a non-dimensional method of relating the suspension dynamics regardless of scale. The damping relationship shown in equation 7.2 is the non-dimensional form relating the actual damping value to that of the critical damping.

$$f_{sus} = \frac{\omega_{sus}}{2\pi} = \frac{1}{2\pi}\sqrt{\frac{k_s}{m_s}} \tag{7.1}$$

$$\zeta = \frac{c}{2\sqrt{m_s \cdot k_s}} \tag{7.2}$$

In order to adjust these parameters it is required to also have a variety of suspension springs as well as an adjustable damper. An available hydraulic actuator discussed later in section 7.5 gives a better idea of the size requirements. The hydraulic actuator uses a PID controller and a servo valve. The background literature revealed these are not effective at higher frequencies and have reduced strokes. This was factored into the sizing. They are, however, very powerful. The available unit could support a very large weight but with size comes additional cost and difficulty in setup, modifications and repairs. The apparatus needed to be able to be installed by hand without hoists or lifts and able to be debugged when connected to a controller on a desktop. This gave an idea of an approximate size but left a wide range available. The most effective size would be determined throughout the course of sourcing parts based on availability and costs.

There is a limited availability of dampers available for the required size with the most appropriate being from the scale RC manufacturers which are a coil-over variety. Their most basic adjustable damper is selected for which the only adjustment is to the internal orifice plate. More complicated shocks incorporate progressive spring stiffness as well as variable damping (variable rate; independently adjustable compression and rebound damping). While there are obvious performance benefits associated with variable dampers, they would require significant experimentation, measurement, and linearization in order to create the theoretical simulation baseline which would have to be integrated into the LQR model.

A linear model using bond graphs for a variable shock damping system was conducted for Sram Bicycle corporation by Redfield and Sutela [49]. The spring and damper totals 5 individual states, none of which are directly measurable without significant or bulky instrumentation. If such a shock was implemented it would require a sophisticated observer model in order to estimate these states.

7.2.3 Ideal Tire Model

7.2.3.1 Tire Stiffness

A tire is normally modelled as a linear spring of significant stiffness which would result in the tire wheel hop frequency through equation 7.3 which is an idealization that is widely accepted and shown by Rajamani [43]. This, like the suspension natural frequency, provides a non-dimensional term regardless of scale and is therefore easily adjustable by varying the stiffness to match the appropriate frequency.

$$f_{wheelhop} = \frac{\omega_2}{2\pi} = \frac{1}{2\pi}\sqrt{\frac{k_t}{m_u + m_s}} \tag{7.3}$$

7.2.3.2 Lift Off

Most active suspension simulations are conducted using a simple spring without gravity. In order to make the simulation more closely mimic that of the real world it has to be recognized that it is permissible that the tire can leave the ground. In order to facilitate this, a guide method for a tire spring is required and the spring must be free to disconnect. While this tends to represent a very difficult scenario it is a common occurrence that happens during speed bumps and pot holes. Thus, for the suspension dynamics for testing real world cases, it is a required part of the quarter car test apparatus.

7.2.4 General Construction

The test apparatus needs to be very robust and simple to use and modify. In order to reduce costs the simplest mounting blocks for the road, the unsprung mass, and the sprung mass are required. This will generally aid in robustness of the design as simpler mounts tend to be more effective.

Mountings need to be easily accessible which is challenging given the variety of shapes available for sensors, actuators, and springs. Compromises will be made attempting to keep a simplified structure to which all components may be mounted.

7.3 Future Work Provisions

There is a wide range of potential expansions to the initial basic active suspension implementation. Potential future work includes:

- Scale tire and platform
- Drive motor and roller road
- Non-linear suspension

Rather than use a spring to model the tire, an actual tire could be used. This requires the potential to mount to the unsprung mass as well as install a tire platform on the road profile.

A further extension of this would be to drive the tire and to have a cylinder shaped roller. This would enable testing traction control schemes with an accurate hardware implementation of the Magic Tire Formula model.

The most complicated extension is that of the non-linear suspension dynamics. This would require suspension geometry such as a Macpherson strut, double A-arms, or a multi-link suspension setup. It would likely require completely re-engineering the test apparatus but could potentially mount to existing blocks.

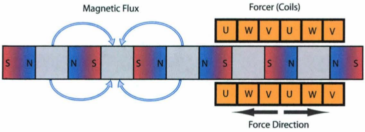


Figure 7.3: 3-Phase Linear Motor Magnetic Poles

7.4 Actuator Development

7.4.1 Linear Motor Prototype

Early in the development, a custom designed actuator was pursued. The initial type was that of a 3-phase brushless motor that was heavily focused on replicating the work of Lu et al. [50]. Lu had positive success with creating a small actuator utilizing a 3-phase design with magnets of alternating polarity and translator blocks to create radial magnetic poles. There was also much work related to linear motors for full scale suspensions [51, 48, 7, 30, 52]. As can be seen in figure 7.3, the translators are connected to the same poles at opposite ends causing magnetic flux from the permanent magnets to exit radially. According to Lenz law shown in equation 7.4, the force is proportional to the magnetic flux density crossed with the current and length of the conductor in that flux.

$$F = \beta \times i\dot{l} \tag{7.4}$$

Work was begun in the Ansys Magneto-Static simulation environment [53] shown in figure 7.4 is the simulation used to estimate the average flux. Based on available

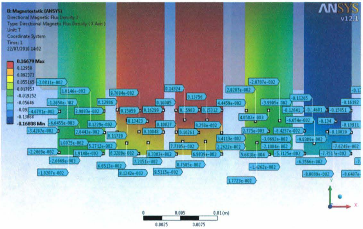


Figure 7.4: 3-Phase Brushless Motor Simulated Flux Field

N42 rated 1/2" x 1/2" cylinder magnets properties and average steel permability of 1.13851E-3 H/m, the averaged flux field was estimated to be 0.069T. This would have produced a force based on 32 gauge wire of 7.51N/Amp.

This fell within an allowable region to fully design and begin work on the construction of this 3-phase linear motor. The original design is shown in figure 7.5. The idea was that more coils could be added to produce more force if needed, expanding the design for the future. A rapid prototyper, such as a fused deposition machine, could produce the stator assembly while magnets and translators could be inserted into a 1/2" ID rod and have its ends capped as shown in the sectioned view in figure 7.6. Since a 3 phase design was going to be used, a method of detecting the correct field direction was required. Low cost Hall effect sensors were chosen and can be seen in both figures 7.5 and 7.6.

The initial prototype produced a measurable force but significantly below that

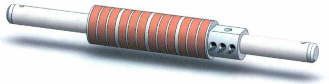


Figure 7.5: Linear Motor Design

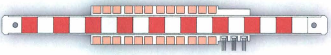


Figure 7.6: Linear Motor Design - Sectioned View

estimated. Attempting to use the prototyped structure as a guiding surface provided a significant amount of friction. As a result the actuator produced only negligible forces and it was apparent that the design was insufficient for the application.

The limiting factors proved to be frictional losses combined with a weak field flux, which dropped off significantly with distance. It was apparent that the magnet rod could not be used with conventional steel bearings and sliding friction was very significant. The original limitations such as the complicated control scheme and potential for force ripple, where not addressed. It was determined the design would require significant work to be functional. An alternative design was sought which would provide the necessary force, size, and simplicity.

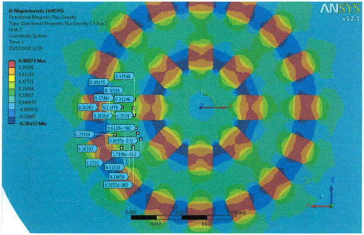


Figure 7.7: Voice Coil Prototype Magneto-Static Analysis

7.4.2 Voice Coil Prototype 1

McBean and Breazeal [54] investigated the development and use of a voice coil with a moving magnet assembly. Its main parts were very similar to that of the previously designed linear motor and an investigation into the voice coils as an alternative was started. Initially, reuse of magnetic components from the linear motor was chosen to speed development and determine feasibility of voice coils. Gysen et al. [30] had already shown good linearity for a 3 phase linear motor utilizing current sensors which permitted the idea that voice coils could provide a superior linear relationship due to the simpler design. Magneto-static simulations were conducted to determine the β term under this linear assumption and are shown in figure 7.7.

Using the material properties provided for the same 1/2" x 1/2" N42 cylinder magnets the magneto-static analysis revealed that the flux in this prototype design was between 0.2T to 0.4T. This simulation allowed for the radial layout of the magnets

Table 7.1: Current Tests

Force (N)	Current (A)
2.982	0.200
6.062	0.392

in a design to be built on the rapid prototyper. The design is shown in figure 7.8 and the final prototype is shown in figure 7.9.

Based on the 12.7mm diameter of the magnets, it was inferred that the primary active area of the coil was only 12.7mm tall. From the samples collected shown in table 7.1 there is a good linear relationship between force and current. The first sample point provides 14.91 N/A while the second point provided 15.46 N/A. The temperature of the coils elevated at 1 A with 26 gauge wire but was not excessive after several minutes of operation. Simulations conducted previously indicated that < 10 N of force could provide measurable improvements to ride quality for the scale test apparatus.

The stroke of this design was only 20mm which was not sufficient for use, nor was the design capable of being mounted to any test fixture. Instead, it provided the foundation of a redesign. Utilizing the relationship previously seen in equation 7.4, the flux term, β, could be calculated and is seen in equation 7.7. Using this as an estimate for a new design, magnet lengths were calculated for a new active area and the shape was modified for packaging.

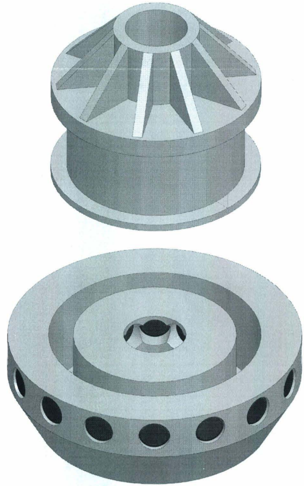


Figure 7.8: Voice Coil One Prototype Design

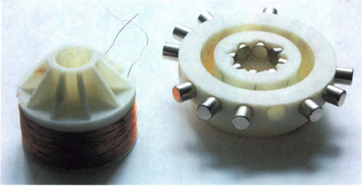


Figure 7.9: Voice Coil One Prototype Constructed Prototype

$$
\begin{aligned}
F &= \beta \times \cdot l && (7.5)\\
6.062 &= \beta \times 0.392 \frac{311m}{12.7mm/33mm} && (7.6)\\
\beta &= 0.13T && (7.7)
\end{aligned}
$$

7.4.3 Voice Coil Prototype 2

A long term goal for actuator development beyond the scope of this thesis was that the actuator could be used in a standalone scale full car model. As a result there had been some preliminary work designing a scale vehicle. The promising nature of the original prototype meant that the second prototype might be able to be designed for use in a quarter car apparatus in development while gaining provisions for a full car apparatus. The decision to impose these requirements was deemed acceptable.

At this point development started on a second prototype with several provisions for the usage in a quarter car and full car model. Design work had been proceeding

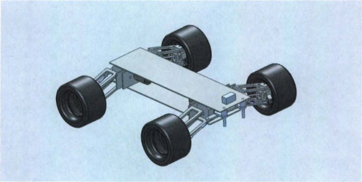


Figure 7.10: Mark IV full car design

already on the full car model shown in figure 7.10. The overall design included four independent motors with front and rear differentials that could be connected together if needed, shown in figure 7.11. The idea was that push rods with a rocker arm similar to that of Traxxas Revo RC car setup would be used shown in figure 7.12.

When this ideology was combined with the voice coil prototype a full redesign was conducted of the full car model in order to determine positioning and approximate weights and is shown in figure 7.13. The design underwent major alterations to accommodate direct actuation. This was motivated mainly by the β constant which was determined from the original voice coil prototype. The direct actuation method was chosen as the alternative, a higher force and lowered displacement actuator, meant additional packaging problems related to the coil diameter.

This meant that a significant displacement and lower force actuator was required. The model also provided a weight estimate for unsprung mass so as to conduct simulation into the required force in various scenarios. From this design, and combined with suspension and tire natural frequencies of 1.12 Hz and 8.7Hz respectively, with a

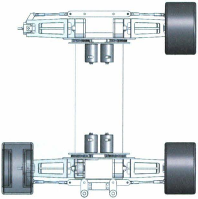


Figure 7.11: Mark IV full car design - Transmission Overview

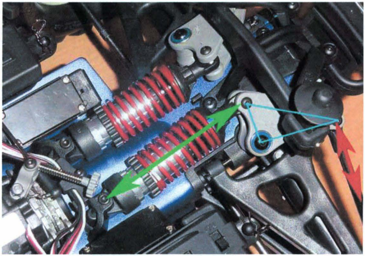


Figure 7.12: Traxxas Revo Pushrod Suspension

Table 7.2: Simulation properties

Item	Value	Unit
Sprung Mass	8	kg
Unsprung Mass	2	kg
Suspension stiffness	400	N/m
Tire stiffness	30000	N/m
Suspension Damping	33.94	N·S/m

damping ratio of 0.3, table 7.2 was derived. Previously developed Bond Graph models for the full scale quarter car active suspension were utilized to determine the amount of force required through a 6mm amplitude swept sine wave curve input ranging from 0 to 94 rad/s. The graph shown in figure 7.14 shows the force output of the actuator for the aforementioned input, based on a ride quality scenario with weighting factors shown in table 7.3. The graph shows that as the input frequency increases from zero to the wheel hop frequency the force increases from approximately +/- 1 N to approximately +/- 5 N. At the wheel hop frequency the unique feature of tire lift off with gravity as the restoring force changes the graph. Normally this would just increase to a peak and then reduce as the frequencies increase. In this case the wheel is forced off the ground at each peak and the force requirements increase. It is not expected to have road profiles above the wheel hop frequency but operation near it will be permitted. From this requirement it was found a force window of approximately 10N was required for the actuator as well as approximately 60mm of travel.

The finalized actuator design utilized 2" x 1/4" wide x 1/8" thick N42 magnets arranged in two circles with an overall exterior diameter of 65mm. The actuator had

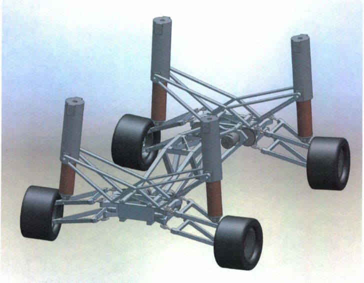


Figure 7.13: Mark V Prototype with Actuators

Table 7.3: Simulation Active Suspension Weights

Weight	Value	Description
ρ_1	0.4	Suspension Deflection
ρ_2	0.16	Sprung Velocity
ρ_3	0.4	Tire Deflection
ρ_4	0.16	Unsprung Velocity

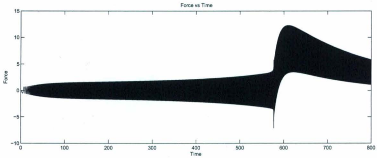


Figure 7.14: Force Plot For 6mm Swept Sine Wave Input Ride Quality

a stroke length of 100mm with an approximately 50mm active coil area making the winding area 150mm long. The design is shown in figure 7.15 while the near finalized version is shown in figure 7.16. It incorporated a rod placed internally and insertable bearings in the stator assembly. This would allow the whole system to be fully guided without need for external bearings, similar to a coil over shock.

With the assumed β constant, a spreadsheet was created to determine the expected force shown in appendix C. It was estimated that at a maximum of 3A the force producible would be 56N. Assuming all energy was to be dissipated as heat in a stationary setup, basic thermal relationship allowed an estimated operating time. Using copper wire with an enamel coating that can sustain 180 degrees Celsius, it was permissible to assume a temperature several degrees below this for operation. Assuming no heat dissipation and 150 degrees maximum with a room temperature of 30 degrees the operating period at 3A would be 43 seconds maximum. This would provide intermittent operation at higher forces if needed.

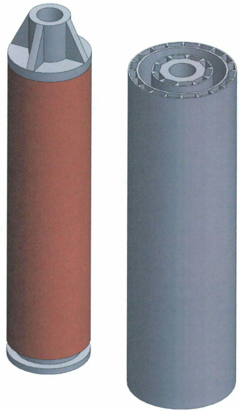


Figure 7.15: Voice Coil Two Design

$$Q = mc\Delta T \tag{7.8}$$

Due to an inability to drive the voice coil with enough current to experimentally lift the coil weight, a flux meter was used to measure the field. This was determined to be approximately 0.03T, or 1/5th the original estimate. This was speculated to be caused by thinner magnets which significantly reduced the flux much more than expected. This reduced available force by 1/5th as well causing the force constant to drop to 3.72 N/A. It also meant that the heating period of the coil of 43 seconds could occur for normal duty cycles as 10N was a continuous forcing requirement.

It was apparent at this stage that the magnet thickness and continuity of the field had a larger effect than anticipated on the field strength. An off the shelf solution was sought at this point.

7.4.4 Moticont Voice Coil

Moticont was able to supply a voice coil similar in design to the dimensions of the second prototype. In order to keep cost down a shorter stroke was selected with a similar diameter. LVCM-051-089-01 was selected with specifications shown in table 7.4, and is shown in figure 7.17. It has a similar force output as the original predictions for the second voice coil design but at a slightly reduced stroke. It was determined that this could be compensated for by adjusting the sprung weight to adjust the natural frequency combined with not using as aggressive road inputs. While the overall dimensions are similar to that of the second prototype there are several key differences.

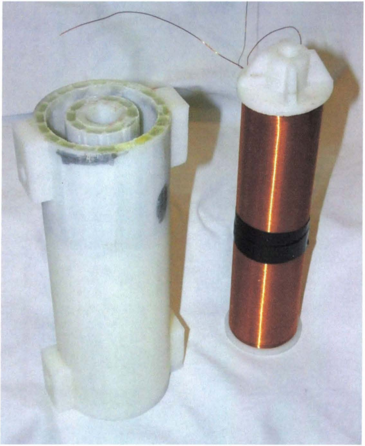


Figure 7.16: Voice Coil Two Constructed Prototype

Table 7.4: Moticont LVCM-051-089-01 Properties

Item	Value	Unit
Intermittent Force @ 10% Duty Cycle	0.4	N
Continuous Force	0.16	N
Force Constant	10.1	N/A
Back EMF Constant	10.1	V·s/m
Stroke	57.2	mm
Coil Clearance Per Side	0.38	mm
Coil Assembly Mass	195	g
Body Mass	1155	g
Coil Resistance	6.8	ohm
Coil Inductance @ 120	6.8	mH
Max Continous Power	40	Watt

The voice coil is unguided and thus requires external linear bearing tracks. This is a major design hurdle as using an internal support shaft makes it very easy to package and deal with handling moments caused by minor misalignments at the attachment. This requires a minimum of two round linear bearings or a square linear rail system that can support the entire mechanism with 2 degrees of constraint for forces and 2 degrees of constraint for torques.

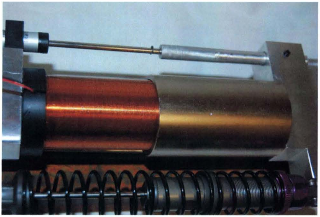


Figure 7.17: Moticont LVCM-051-089-01

7.5 Parts

7.5.1 Linear Potentiometer

In order to measure tire deflection and suspension deflection a linear sensor is required. Linear Variable Differential Transformer (LVDT) displacement transducers are normally used to measure displacement, however, due to the design of the transformer assembly, these are approximately three to four times longer when collapsed compared to the distance they can measure. This would inhibit their incorporation into a small test apparatus and would also add unnecessary weight.

A linear potentiometer presented itself as a smaller and more cost effective way of measuring displacement. Initially an OMEGA LP804-3 was found to meet required specifications, but OMEGA advised that they are in end of life. P3 America provided an alternative with the RC13M-75 shown in figure 7.18.

The plastic pinch clamps, shown in figure 7.19, allow for very versatile installation

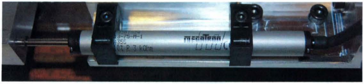


Figure 7.18: RC13 Linear Potentiometer

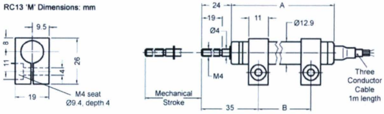


Figure 7.19: Linear Pot Schematic

and adjustments as the two clamps can be positioned anywhere along the shaft.

7.5.2 Coil Over Shock

The quarter car test apparatus corresponds approximately to a 1/5th scale RC vehicle. Due to the size of application, there are few dampers or coil over shocks that exist between the 1/10th and 1/8th scale RC and a full scale vehicle. The HPI Baja 5B shock, shown in figure 7.20, is the largest available RC vehicle coil over but suits the application very well. Its original design utilizes aluminum shocks with adjustable damping by a rotatable orifice plate shown in figure 7.21. Various springs are available shown in figure 7.22 providing a wide range of stiffness.

The orifice plate has 5 openings of sizes 1.46 mm, 1.32 mm, 1.20 mm, 1.14 mm, and 1.00 mm, to provide various damping constants. Should this prove not to be

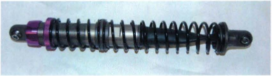


Figure 7.20: HPI Coil Over Shock

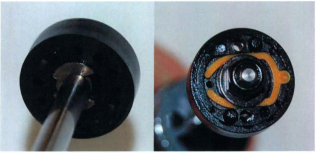


Figure 7.21: Adjustable Orifice Plate

sufficient to match the appropriately required damping there are various weights of oil from 5 wt to 60wt. The largest opening of 1.46 mm was set and the shock was filled with 20 wt oil.

Determining the damping coefficient is necessary in order to design the LQR controller. In order to facilitate this, the spring of the coil over shock was removed as shown in the setup in figure 7.23. The governing equation for the motion of any linear system is shown in equation 7.9. However, with the spring removed this equation becomes equation 7.10. In a scenario where the sprung mass is raised and allowed to free fall, with no external forces except gravity, the block accelerates, but begins to slow related to the damping, 'b' term. Eventually the damping term will balance with the gravitational term giving equation 7.11. Assuming in this scenario

Figure 7.22: Alternative Springs

that the velocity remains very "flat" over this region, we can calculate the damping coefficient, b.

$$\sum F = m\ddot{x} + b\dot{x} + kx \tag{7.9}$$

$$\sum F = m\ddot{x} + b\dot{x} \tag{7.10}$$

$$m \cdot g = b\dot{x} \tag{7.11}$$

Eleven drop tests were conducted. The data logger based on an Arduino, which is further described in chapter 8, to store position from the linear potentiometer. Velocity was calculated on the Arduino by simple differentiation shown in equation

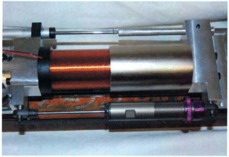


Figure 7.23: Baja 5B Coil Over Shock - Spring Removed

7.12. The previous displacement sample is subtracted from the current displacement sample, divided by the time between samples. This basic differentiation at a high enough rate provides a good approximation. There are limitations to this scheme which will be described in chapter 8.

After removing all the air from the shock several times the drop test provided very good linear results as shown in tests 1 and 2 in figure 7.24. The averaged results for each drop are shown in table 7.5. The averaged velocity over 11 drops was taken. The motivation for this was the thermal considerations for longer operation. The idea was to average the performance over the expected operating period of a few minutes. The dropping mass was measured at 2.038 kg and gravity was assumed to be 9.81 m/s^2 providing a force of 20.0N. From the previous relation in equation 7.11, with an averaged velocity from table 7.5 of 0.356 m/s, this gives a damping constant of 56.1 $(N \cdot s)/m$. More data is available in appendix D.

$$v = \frac{\Delta d}{\Delta t} \tag{7.12}$$

The suspension springs were tested using a drill press and a weight scale to measure

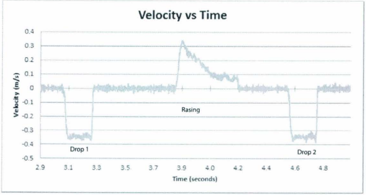


Figure 7.24: Drop Test Results Sample

Table 7.5: Damper Drop Test

Trial	Velocity (m/s)
1	-0.384
2	-0.370
3	-0.350
4	-0.355
5	-0.357
6	-0.373
7	-0.354
8	-0.345
9	-0.348
10	-0.341
11	-0.344
Average	-0.356

the force as shown in figure 7.25. The spring is depressed at approximately 10N intervals and the deflection measured. A linear regression is taken as well as an R^2 value to ensure that the curve is linear. The original 2 long black springs were tested and their deflection force curves are shown in figures 7.26 and 7.27, while other regression data is available in appendix E. As can be seen from the figures both springs are very linear and both are very similar. That is not the case for all available springs as shown in table 7.6 where matched springs coefficients can differ by 0.1% (Stock Long Black Springs) to 10% (Stock Short Black Springs). In most arrangements there are two springs placed in series with a small plastic slider. This reduces the spring stiffness similar to putting resistors in parallel by the relationship shown in equation 7.13.

$$\frac{1}{k_{eq}} = \frac{1}{k_1} + \frac{1}{k_2} \tag{7.13}$$

7.5.3 Tire Springs

Two tire springs are available for use as shown in figure 7.28. They have stiffnesses of 15.7 N/mm (short) and 10.5 N/mm (long) which allows a level of tuning. It is noticeable that each spring is of different length. The design takes into account this issue, as a maximum stroke of 75mm is permissible. As a result the full length of each spring fits near mid stroke in the design and even in a fully collapsed state both springs remain within the travel. They are both permitted to extend approximately 25 mm before the travel of the linear potentiometer is exhausted.

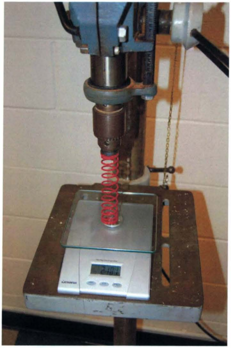


Figure 7.25: Spring Test Setup

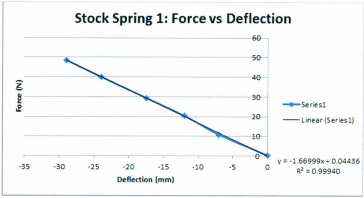


Figure 7.26: Spring Test 1 Regression Curve

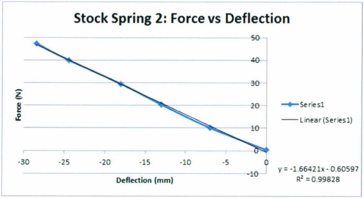


Figure 7.27: Spring Test 2 Regression Curve

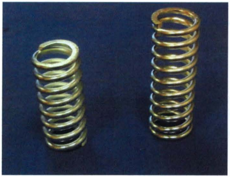


Figure 7.28: Tire Springs

Table 7.6: Spring Test Results

Spring	k (N/mm)		k (N/mm)	Spring
Description	R^2		R^2	Description
Stock Black Long	1.67		1.66	Stock Black Long
Damped Shock	0.999		1.000	Undamped Shock
Black Short	3.51		3.87	Black Short
Damped Shock	0.997		0.995	Undamped Shock
Red Long	1.76		1.72	Red Long
Label A	0.999		0.998	Label B
Red Med	2.64		2.78	Red Med
Label A	0.998		0.997	Label B
Red Short	4.62		4.43	Red Short
Label A	0.999		0.995	Label B
Orange Long	1.50		1.57	Orange Long
Label A	0.999		0.998	Label B
Orange Med	2.02		2.05	Orange Med
Label A	0.999		0.999	Label B
Black Long	1.83		1.89	Black Long
Label A	0.997		1.000	Label B
Black Med	2.56		2.45	Black Med
Label A	0.999		0.997	Label B

7.5.4 Instron 8800 / MTS Actuator

An Instron 8800 Controller combined with a hydraulic actuator as shown in figures 7.29 and 7.30 were used to create a road input. The Instron 8800 is controlled by a software package that utilizes a PID control with a high accuracy LVDT and a servo valve to provide positional control. It is able to follow a variety of waveforms as well as follow an analog reference input which is useful for future simulations.

7.6 Design

Based on the selected parts, provisions, and requirements previously outlined combined with a standard 25mm square linear rail, a prototype linear plant was designed as shown in figure 7.31. It utilized three aluminum blocks bolted to linear guide bearings. These linear bearings act as a support for the sprung and unsprung masses, as well as a road input, but also provided the required alignment between the sprung and unsprung masses for the Moticont voice coil.

Each block was designed with a specific purpose and provisions for future redesign. There are threaded bolt holes on both the road input block and the unsprung block that could facilitate an armature to hold a tire and a platform for it to rest upon. This would allow true tire dynamics to come into effect which would include tire damping. The full plans for the blocks are shown in appendix F.

Once constructed the test apparatus had all potentiometers aligned along with the voice coil on a flat surface and was tested for functionality as shown in figure 7.32. This facilitated testing of the LQR controller.

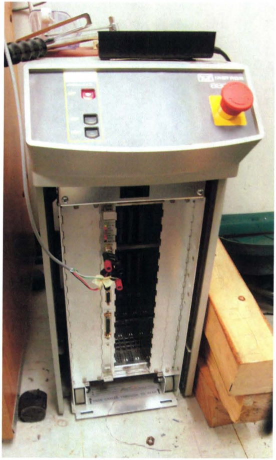


Figure 7.29: Instron 8800 Controller

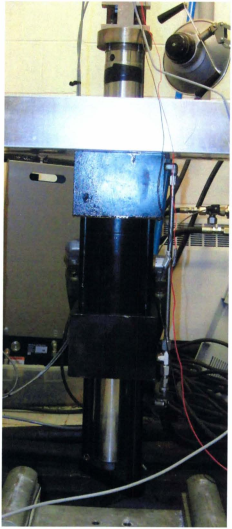


Figure 7.30: MTS Hydraulic Actuator

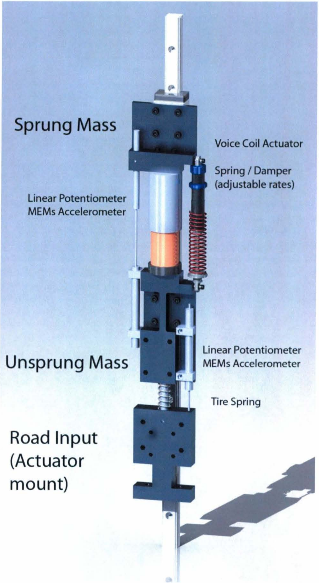


Figure 7.31: Quarter Car Design

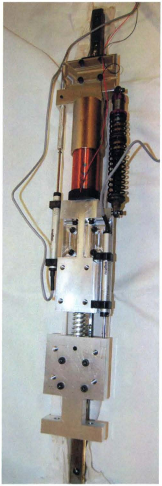


Figure 7.32: Quarter Car Constructed Prototype

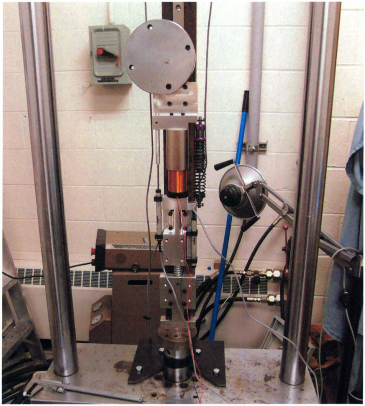


Figure 7.33: Quarter Car Test Apparatus Mounted On MTS Actuator

Chapter 8

Hardware Controller

8.1 Overview

As mentioned earlier a radically different approach to the controller is being taken by utilizing a Field Programmable Gate Array. An FPGA consists of many reconfigurable logic blocks that connect to input and output blocks (IOB). These are set up in a grid array as shown in figure 8.1. Each of these logic blocks contain basic reconfigurable elements that can create logical blocks such as AND, OR, NAND, XOR, etc. gates. These can then be arranged in such a way to create adders, multipliers, conditioning algorithms, or even general purpose processors.

The complexity of these systems necessitates a much higher level approach to programming. An idea similar to how programming languages are built on top of libraries, which themselves have been based on a lower level code, a descriptive languages approach has developed to accomplish tasks in hardware with an FPGA. Two primary languages have evolved: Verilog and VHDL (V[ery High Speed Inte-

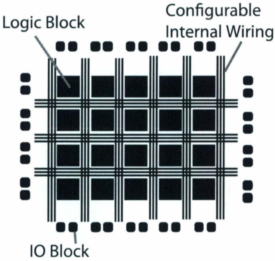


Figure 8.1: FPGA blocks

grated Circuit] Hardware Description Language). While both present challenges and strengths VHDL was chosen to implement the controller due to availability of fixed point mathematical libraries.

This chapter is divided into several sections starting with a description of the electronics and continuing into a simple explanation of how a 1 bit adder is created in theory and in VHDL. From there it goes into the major component blocks of the controller followed by individual explanations on what each block does and finally a description of the overall interconnecting of the blocks. Due to time constraints a secondary device was used for logging. Originally logging directly from the FPGA into a computer via the serial port was the planned method; however implementation of this did not fit into the time frame. Most the basics of VHDL programming are derived from several texts by Brown, Perry, and Kilts [55, 56, 57] respectively.

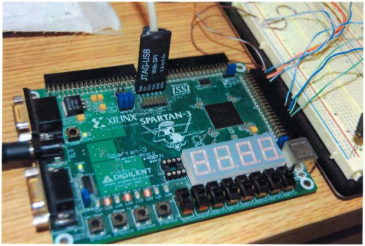


Figure 8.2: Spartan 3 Development Board

8.2 Electronics

The FPGA chosen was originally a Digilent S3 development board based on a Spartan 3 XC3S200 shown in figure 8.2. However, the code was eventually ported to a Nexsys 2 with a Spartan 3E chip shown in figure 8.3. This was easily done by changing the user constraint file to point to the correct pins, and the setup which is included in appendix G.

Referring to figure 8.5, the FPGA is directly wired to 4 analog to digital convertor (ADC) integrated circuit (IC) from Microchip, model number MCP3201. There is an H-Bridge Chip with its control lines connected to the FPGA which drives the linear motor. However, the current is run through a Hall effect based sensor which provides an analog output. This is pulled up to + 5 volt through two resistors so that rather than a +/- voltage, it is always positive for the expected current as shown in

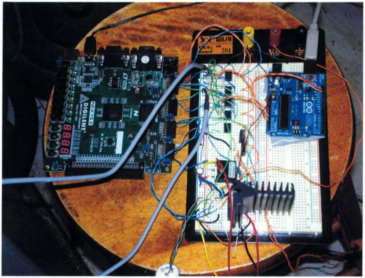


Figure 8.3: Nexsys 2 Development Board And Circuit

figure 8.4. The output is tied to one of the analog to digital converters. The current sensor requires +/- 15 volts in order to operate correctly which is supplied through two regulators. The two linear potentiometers connect to two ADC inputs and the analog output of the MTS machine for position connects to a fourth ADC.

8.3 Mathematical Adder

A 1-bit adder has two inputs, each with two states, and requires a 2-bit output. Input A can be 1 or 0 and Input B can be 1 or 0. Utilizing an XOR and an AND gate it can produce a summation. To create an adder that can calculate 1+1, or in binary $1_2 + 1_2 = 10_2$, the ones place needs a result of 0 through 1_2 XOR $1_2 = 0_2$ while the tens place needs a result of 1 through 1_2 AND $1_2 = 1_2$. This is shown schematically

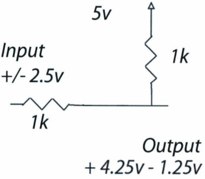


Figure 8.4: Pullup Resistor Divider

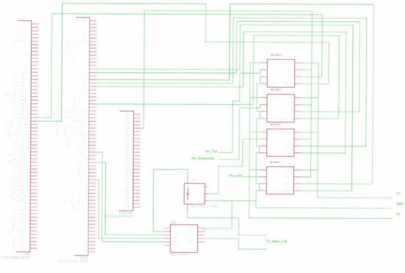


Figure 8.5: FPGA and Support Electronics Schematic

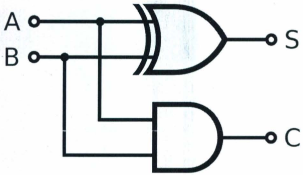


Figure 8.6: Basic Half Adder

in figure 8.6.

However, when multiple bits are used the AND gate becomes a 'carry out' gate for the ones place, but a 'carry in' gate becomes necessary for any bit greater than 1. The truth table for this is shown in table 8.1. The VHDL implementation is very simple and is shown below.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;

entity Adder is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           out : out  STD_LOGIC_VECTOR (1 downto 0));
end Adder;
```

Table 8.1: Full Adder Truth Table

Inputs			Outputs	
A	B	Cin	Cout	S
0	0	0	0	0
1	0	0	0	1
0	1	0	0	1
1	1	0	1	0
0	0	1	0	1
1	0	1	1	0
0	1	1	1	0
1	1	1	1	1

```
architecture Behavioral of Adder is

signal sum : std_logic_vector(1 downto 0) := "00";


begin

sum <= a + b;

out <= sum
```

This adder works very well, however, it may need to be controlled by a clock signal or have a reset or enable line. The ADC reader example shown in the next section will illuminate the differences.

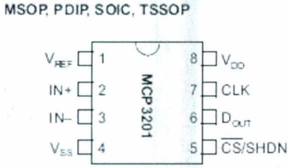


Figure 8.7: MCP 3201 Chip Pinout

8.4 ADC Reader

The basis of inputs in digital control is still usually an analog signal. In this case it is from linear potentiometers and a voltage output from the MTS machine that is appropriately scaled. The common ADC from Microchip, the MCP3201, was used shown in figure 8.7. This chip is a 12 bit serial output, and as such it requires a clock signal and a CS signal. When the CS signal is pulled low and a clock is sent to the ADC, the data output starts sending individual bits descending from most significant to least significant as shown in figure 8.8. This is a basic unidirectional form of and SPI, or Serial Peripheral Interface Bus without any form of commands or flow control.

In order to facilitate this, a counter was implemented running on the FPGA clock cycle in VHDL with enable, and reset lines. As can be seen in figure 8.9, the ADC reader has 4 inputs *datain* through *datain4*, a *clock*, an *enable*, and *reset* lines. The clock input is a reduced clock signal from the 50 MHz onboard oscillator providing a clock signal at a rate of (50E6 / 113) Hz, or 442477.876 Hz. It takes the ADC reader

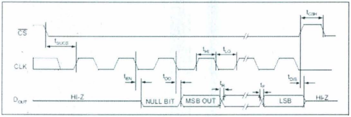


Figure 8.8: MCP 3201 Timing

27 clock signals to sample the data from the ADC. A minimum of 19 is required but 27 is utilized to provide a final sampling rate of 16388.0695, which has a 0.02% error to the target of 2^{14}, 16384.

In order to get a more accurately differentiated signal a running average is used. Implementing a mathematical routine to average samples in floating point math presented a problem of using a large amount of FPGA logic blocks for a relatively simple operation. The reasoning behind this is that a microcontroller will take many clock cycles to do floating point math in order to average a number. However, division by 2 is effectively a shift register in binary. This means that adding n bits together yields an n+1 bit storage. If L is the number of bits, or the length, then an average technique over 2 samples can be conducted by adding n bits together and taking the nth bit down to (n+L)th bit.

The VHDL code for the following explanation is found in appendix G. Initially there are included libraries that define 'common constructs'. These explain how to convert basic mathematical operations into hardware. The entity definition defines ports into and out of the block. These can be connected to exterior I/O pins or using a schematic, shown later, to other blocks. The '*architecture behavioral*' of the

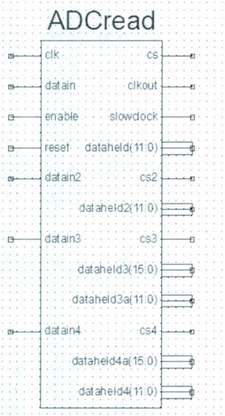


Figure 8.9: ADC Read Schematic Block

'ADCread' block is essentially the variables. These act as storage units and can be manipulated depending on definition.

The actual code is driven by the clock and is allowed to start counting if *enable* is high, and *reset* is low, at which point it lowers *csstate* causing the ADC to sample and hold the value. The *clkout* drives the clock line on the adc and after 4 pulses it begins to read out the data storing it in a temp vector. At a count of 12 (13th count), it changes the sign of the *slowclk*, which is the clock signal that will drive the rest of the code such as the running average. Its state is changed again near the end when the temp value is moved to the *output*. At the very bottom of the code the variables are pointed towards the I/O blocks. These pins do not have states and as such need to be tied to another value. These can happen continuously, as done here, or as conducted with *datain* inside the main code block, triggered by clock signals.

All the code is provided in appendix G.

8.5 Running Average

In order to reduce noise, 16 samples are taken and averaged out over a period of 32 samples. This reduces the approximately 16384 samples a second to 512 samples per second. It increases the accuracy of the differentiated signal along with smoothing the position signals. Noise is effectively reduced by the square root of the number of samples, and as a result reduced by a factor of 4; however this does introduce a one clock cycle latency in the controller, or 1.95ms.

According to the code below, if the count is below 17 counts it keeps adding the input, din, to the position. Once it is at the 17th count (0 counts as a sample time,

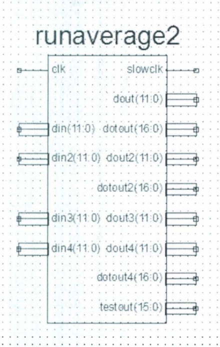


Figure 8.10: Running Average 2 Block

effectively 0 - 15 is 16 samples), it performs the calculation as outlined previously. The top 12 most significant bits are selected and output. This is effectively dividing by 16. However, it can be seen that the calculation for the velocity is conducted with the full 16 accumulated bits, even though the input is 12bit. Since these will be scaled later, and it is based on a time sample, it would be best to keep all the bits. It will add noise, but it would get truncated later on. The code can be found in appendix G.

8.6 Fixed Point Controller

Earlier it was explained how a basic adder works. The procedure is more complicated for multiplication, but in its simplest form a multiplier several shift registers and adders, one for each bit. This can be very consuming for the FPGA in terms cf logical blocks if over utilized which is a major problem. The limitation is that this only works with integer math directly with current libraries. As seen in the code, all $STD\_LOGIC\_VECTORS$, or signed values, are integers. This presents a problem when the LQR calculated gains that are decimal numbers based on multiplication to meters or meters/second. Floating point math is extensively utilized in the microcontroller and microprocessor world, but implementation of such schemes is very costly for transistor resources and requires many clock cycles in a processor to be conducted. An alternative is fixed point math. The idea is similar to integer math. For example:

```
    1 0 1 1
x   1 0 1 1
```

```
-----------
        1 0 1 1
      1 0 1 1 -
    0 0 0 0 - -
+ 1 0 1 1 - - -
---------------
  1 1 1 1 0 0 1 => 64 + 32 + 16 + 8 + 1 = 121


  11
x 11
----
  11
+11-
----
 121
```

With fixed point math, a point is chosen to represent where the decimal place is. This is always related to the integer least significant bit 0. The positions after the decimal place count negative and represent the value given in equation 8.1. Thus the -1 place is the value of 1/2 if represented by a binary 1, while the -2 place is 1/4, and so on. As can be seen below the storage of bits for a 4b.4b multiplied by a 2b.2b is 6b.6b. This is critical to remember in utilizing the IEEE proposed fixed point libraries. As these are incomplete, they do not return errors when assigned to incorrect bit sizes. Instead they compile and will be capable of programming the FPGA but will give an unpredictable result.

$$V_{-n} = \frac{1}{2^n} \tag{8.1}$$

```
    1 1 0 1 . 1 1 0 1

  x     1 1 . 0 1

--------------------------

        1 1 . 0 1 1 1 0 1

      0 0 0 . 0 0 0 0 0 -

    1 1 0 1 . 1 1 0 1 - -

+ 1 1 0 1 1 . 1 0 1 - - -

------------------------

1 0 1 1 0 0 . 1 1 1 0 0 1 =>

   32 + 8 + 4 + 1/2 + 1/4 + 1/8 + 1/64 = 44.890625
```

```
  13.8125

x  3.25

-----------

   0.690625

   2.7625--

+ 41.4375--

-----------
```

```
44.890625
```

Utilizing this fixed point math it is possible to scale the 0 - 4095 value from the ADC to a numerical value related to the 0.00 - 75.00mm output. However, a more effective method is to zero the ADC output, creating a 'signed' value. This zeroing is activated by a button input on the development board. This event stores the current ADC value, e.g. 2154, and then outputs a value of *In* minus the stored value. This is then scaled using a scaling factor in fixed point math.

$$position = scale * (sfixed(zeroedpos) * sfixed('0'\&scale)) \tag{8.2}$$

The matter of differentiating the velocity signal has been discussed earlier, and utilizing all the bits from the 16 samples relates specifically to the issue of scaling. In a 1/512th time period if there is a change of 1 bit, and each bit scales to 0.018310546mm than the example shown in equation 8.3 highlights the problem. A very high resolution ADC is required to calculate the velocity accurately or a larger time sample is required. If 16 samples are summed together however it relates to a 16 bit value. If during half the second 16 samples are 1 bit larger than the sensitivity gets increased and example becomes equation 8.4 and shows a potential lower value. This theoretically increases the accuracy of the 12 bit ADC for use in differentiation. The effectiveness of this algorithm has not been numerically verified.

$$v = 1 * 0.018310546mm/(1/512)s = 9.374999552mm/s \tag{8.3}$$

$$v = (8 - 0)1 * 0.018310546mm/((1/512)s * 16samples) = 4.68749824mm/s \tag{8.4}$$

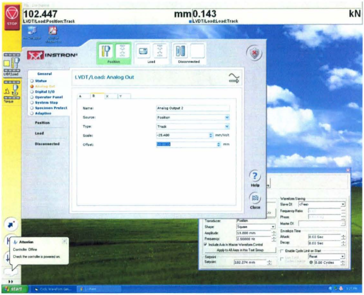
In the case of the LQR for the quarter car model as described in chapter 4, the states are the spring deflections and the absolute velocities. A fixed point gain can be multiplied directly to the zeroed deflections, while the velocities of the sprung and unsprung masses have to be calculated by simple addition as shown in equations 8.5 and 8.6. The MTS machine has a high resolution high speed analog output and can be scaled as needed as shown in figure 8.11. This signal originates with an LVDT that is sampled at 16 bits, and then scaled internally to a 16 bit output with a full scale voltage of +/- 10V. Since the analog digital converters are calibrated to 0 - 5V, it was scaled such that 20mm = 1V. Due to some calibration error in the setup, the actual value needed to be input is approximately 25mm to get this scaling. The zero point of the system is raised until the voltage reading is 2.5V. This allows for the maximum vertical travel in both the upward and downward direction with the desired accuracy.

$$v_{Unsprung} = v_{Road} + v_{TireDeflection} \tag{8.5}$$

$$v_{Sprung} = v_{Road} + v_{TireDeflection} + v_{SuspensionDeflection} \tag{8.6}$$

8.7 PWM

Pulse-width modulation, or PWM uses the idea that if an energy source is switched on and off very quickly that the capacitance and the inductance of a circuit will keep the current flowing causing a current ripple throughout depending on the duty cycle (how long the system is on versus off) and the type of load. For instance an inductive



102.447 mm
LVDT/Load:Position:Track

0.143 kN
LVDT/Load:Load:Track

LVDT/Load: Analog Out

Name: Analog Output 2

Source: Position

Type: Track

Scale: -25.400 mm/Volt

Offset: [illegible] mm

Transducer: Position

Shape: Square

Amplitude: 18.000 mm

Frequency: 2.50000 Hz

☑ Include Axis in Master Waveform Control

Setpoint: 102.274 mm

Waveform Slaving

Slave Of: <Free>

Envelope Time

Attack: 0.00 Sec

Decay: 0.00 Sec

□ Enable Cycle Limit on Start

Reset @ 0.00 Cycles

Attention

Controller Offline

Check the controller is powered on.

Figure 8.11: MTS Machine Analog Out Scaling

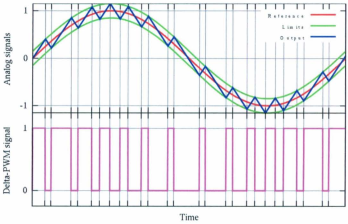


Figure 8.12: Pulse Width Modulation

load will produce a saw wave like current ripple while a capacitive load will be more logarithmic ripple. This can be seen in the sine wave shown in figure 8.12[1].

The saturated output from the fixed point controller is fed into the PWM controller. The PWM is an 11 bit counter using the first bit for direction. It continually counts up and compares the incoming value with the count. If the count is less than the incoming value then the output is switched on, and if the count is greater than the incoming value then the output is switched off. This allows for a linear PWM output for the values. With a 11 bit range this means a maximum value of 2048. Given an input of 1024, the PWM duty cycle is 50%. If the input is greater than 1024 then the PWM duty cycle is greater than 50% and vice versa. In VHDL this relies again on the clk signal and the schematic representation with the clock, enable, reset and speed lines as inputs, and a PWM output, Short Brake, and Direction as

[1] (Cyril BUTTAY, derivative work: Krishnavedala, 2011-05-29 03:52 (UTC))

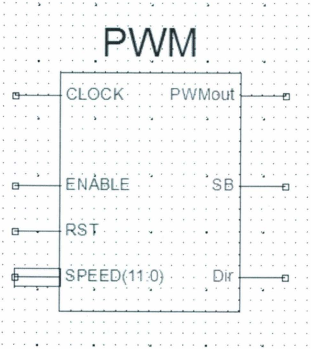


Figure 8.13: PWM Block

outputs.

These outputs depend on the various H-Bridge chips used, as some may utilize two direction lines, or opposite values on the SB or direction lines. The LMD18200 H-bridge chip is utilized due to its high voltage with suits the Moticont voice coil well. It is rated at 55v and 3A continuous giving a potential continuous power of 165 watts minus losses. The chip can survive 6A peak loading at short duty cycles. In order to do this would require an operating voltage of approximately 40V.

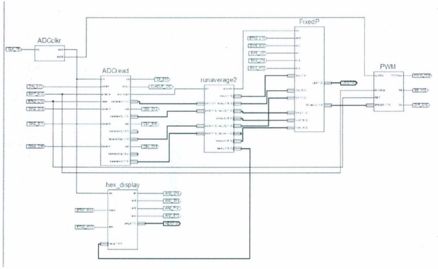


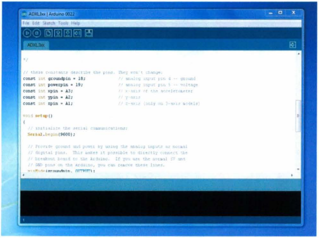
Figure 8.14: Controller Schematic

8.8 Interconnection

All of the previously outlined VHDL code can be compiled into schematics symbols shown throughout. These can be wired together as an analog to designing a circuit. The *ADCread* block connects to the RunAverage2 block. This in turn connects to the fixed point controller which also contains the current feedback control. Finally this gives a 12 bit output used to drive the PWM controller completing the FPGA based controller. The schematic representation is shown in figure 8.14.

8.9 Logging

The Arduino [58] is a simple 8 bit micro controller that is based on an Atmega 328 from Atmel Corporation with an open source bootloader. The Atmega 328 is



```
*/

// these constants describe the pins. They won't change:
const int groundpin = 18;             // analog input pin 4 -- ground
const int powerpin = 19;              // analog input pin 5 -- voltage
const int xpin = A3;                  // x-axis of the accelerometer
const int ypin = A2;                  // y-axis
const int zpin = A1;                  // z-axis (only on 3-axis models)

void setup()
{
  // initialize the serial communications:
  Serial.begin(9600);

  // Provide ground and power by using the analog inputs as normal
  // digital pins.  This makes it possible to directly connect the
  // breakout board to the Arduino.  If you use the normal 5V and
  // GND pins on the Arduino, you can remove these lines.
  pinMode(groundpin, OUTPUT);
```

Figure 8.15: Arduino Development Environment

a versatile micro controller that has become a very important and common part of the hobbyist electronics community. The heart of the Arduino is the development environment based on 'Java' and the 'Processing' environment. Its syntax is nearly identical to that of the programming language C. The environment is shown in figure 8.15 while the microcontroller development board is shown in figure 8.16.

A 'sketch' is written that samples four of the six internal ADCs which are tied in parallel to the analog inputs of the ADC that the FPGA are controlling. There are several issues with this that were deemed acceptable in order to speed development, they are:

1. Sample time is variable

2. Onboard processing would reduce the number of samples/second

3. ADCs are only 10 bit compared to the MCP3201 12 bits

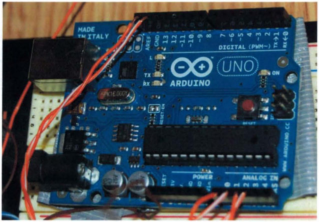


Figure 8.16: Arduino Microcontroller

4. ADCs are internal and may be more susceptible to noise from other circuits

The Arduino is programmed to sample as fast as possible and average 10 samples on 4 ADCs, then output the raw data over the serial bus at 115200bps. Part of the issue with a microcontroller is that when it is sampling data the controller is not doing anything else. The workaround is called ‘interrupts’ which interrupt the current task for a more important one. This is a stark contrast to that of an FPGA where there is no processor; rather it is all based in hardwired logic. Because everything happens sequentially any change in cycle time of the program by any one element means that the elapsed time is no longer constant. As such more processing time must be wasted to log the time between samples.

Within these limitations the Arduino is still chosen as a temporary logging device for two reasons: (1) It is very fast to develop for because it is possible to reuse pre-written libraries, and (2) onboard logging with an FPGA directly is difficult to interface with common flash memory such as an SD card. A transitional effort to send the samples and calculated values to the Arduino digitally would enable it to deal the control structure for a file system such as FAT32 and communication to an SD card. The long term goal would be to store data onboard which could be accomplished by embedding a microcontroller in the FPGA and tying it to the data digitally. This would further reduce communications delays but could take several months of development time.

The code for the Arduino logger can be found in appendix G.

Chapter 9

Results

9.1 Overview

Experimental work with the quarter car test apparatus was minimal due to long development time. This work is to serve as an initial baseline by providing a basic LQR implementation utilizing an FPGA and a voice coil actuator. Some background is provided about the setup and values used in the LQR solution in section 9.2. As described previously, a separate logging device was used. In order to speed up communications only the raw values are sent over the serial port. Section 9.3 describes this as well as how this raw data was processed. The experimental results are presented and examined in section 9.4 for both the passive and the ride quality cases. An expansion of this work is discussed in section 9.5 where the road input is used as the input to the Bond Graph model and simulated in 20-Sim. This allows a direct comparison of the ideal active suspension, and that of the actual setup. The road input is also used to simulate a passive suspension to give a better indication of the performance gains. The conclusion in section 9.6 talks about how a voice coil can successfully be used as high bandwidth actuator and provide good results for ride quality.

9.2 Setup

The components of the quart car model were weighed, and experimentally obtained values for spring stiffness and damping established a complete representation of the linear plant parametrically. The quarter car test apparatus parameters are shown in table 9.1. LQR gains were calculated for the ride quality scenario similar to that of the original quarter car design with weights shown in table 9.2. The current sensor and force proportionality constants allow an error to be calculated between actual and target forces allowing for a proportional controller to be implemented as described previously. A gain for this error to be multiplied by was experimentally derived. Several gains were tested and a gain of 100 provided very good results without saturating when used with the PWM controller. The PWM controller provide 0-100% voltage output over a 0-2047 integer range. All gains are shown in table 9.3.

This is the setup for the 4 state with a feedback loop which was shown to have potential limitations when compared with the 5 state. The 4 state LQR hardware development was already being developed when the 5 state was being investigated. Due to the required amount of programming and debugging the implementation of a 5 state LQR is beyond the scope of this thesis but is to be investigated in the future.

A sine wave was utilized for the road with a frequency range of 0.1 - 6.00 Hz with target amplitude of approximately 11.8mm. Due to calibration errors in the

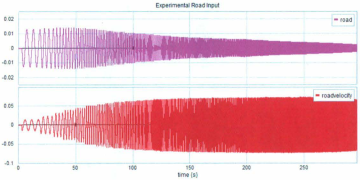


Figure 9.1: Experimental Road Input

software, this meant that 15mm was the input value in the software but 11.8mm was the actual stroke on the actuator. Initially the MTS software was set to produced a 1V / 25.4mm output but when this was calibrated for the FPGA scaling factor over the expected movement range, it was found that 1V corresponded to 20mm. The testing for the frequency range from 0.1 Hz - 3 Hz was conducted at 0.1 Hz intervals while 3 Hz to 6 Hz was conducted at 0.2 Hz intervals. The MTS setup uses its own PID control for position, and as a result of the mass and hydraulic system, the actual amplitude achieved is smaller at higher frequencies than requested. This phenomena is seen by the position and velocity graphs for the active suspension test in figure 9.1.

9.3 Processing

The Arduino outputs very basic data over the serial COM port formatted as shown below:

```
2343, 6373, 5925, 5176, 4700
```

Table 9.1: Quarter Car Test Apparatus Simulation Parameters

Item	Value	Unit
Sprung Mass	4.7	kg
Unsprung Mass	1.5	kg
Suspension stiffness	1005	N/m
Tire stiffness	10458	N/m
Suspension Damping	56.096	N·s/m

Table 9.2: Quarter Car Test Apparatus LQR Weights

Weight	Value	Description
ρ_1	0.4	Suspension Deflection
ρ_2	0.16	Sprung Velocity
ρ_3	0.4	Tire Deflection
ρ_4	0.16	Unsprung Velocity

Table 9.3: Quarter Car Test Apparatus Ride Quality Gains

Controller	q_{k_s}	v_{m_s}	q_{k_t}	v_{m_u}	Feedback
4 State	-1.002	-0.0503	0.003	0.0561	100

```
2340, 6372, 5925, 5179, 4512
2340, 6374, 5924, 5180, 4524
```

As described in the previous chapter, this corresponds to the various sensors. The comma separated values represent 10 integer summations of raw sensor data for the suspension deflection, tire deflection, current sensor, and road position (from Instron 8800 analog output). The last column is the time in microseconds between samples. This is required to more accurately differentiate the signal to produce a velocity since no low cost velocity sensors were available. It also clearly highlights the issue using a microcontroller. Between the first and second samples an identical amount of data was processed, but none of their times match perfectly. This would lead to differentiation error and inaccurate velocity samples if it was not accounted for.

Using Excel, an average was taken over the stationary period before the experiment of the raw samples and used to zero the input. The data was then scaled with the appropriate constants to millimeters, or force. Using the sample times, the sprung and unsprung velocities were calculated. This data was imported into 20-Sim and a 10 Hz low-pass filter was placed on the positional outputs as the natural frequency was less than 10 Hz, and the road input was less than 10 Hz. It is assumed that since all natural frequencies are < 10 Hz (Wheel hop is approximately 6.54 Hz) and all actuations signals are also < 10 Hz that this should reduce noise while keeping the dynamics needed to be examined. Due to cumulative sensor noise, from up to three sensors, it was also necessary to reduce the noise on the velocities, again using a simple 2nd order low-pass filter.

9.4 Results

While the experiments are performed for the passive and active cases nearly identically, the exact time the apparatus runs at a specified frequency is controlled by the operator. As such when comparing the two experiments the time scales should not be weighed heavily, rather the shapes and magnitudes should be scrutinized. Figure 9.2 shows how there is reduced sprung velocity with the active suspension when compared to that of the passive suspension. For the ride quality case the goal is to decrease sprung mass accelerations, and when actuated with a sinewave the sprung velocity is decreased which would generally related to reduce sprung accelerations. This shows the basic implementation is performing correctly with regards to the sprung velocity. A clearer indication is given if the data is transformed into the frequency domain shown in figure 9.3. This shows decreased sprung velocities throughout the entire range. For the active suspension the maximum amplitude and frequency occurs at approximately 46 mm/s, 0.785 Hz. For the passive suspension this is approximately 69mm/s, 1.321 Hz. According to the works of Rajamani and Butsuen [43, 35], this is the expected result. The suspension natural frequency shifts to a lower frequency and the amplitude is reduced.

In order to decrease the sprung velocity there are normal trade-offs; with ride quality the main trade off is the utilization of suspension deflection in order to absorb road inputs. Referring to figure 9.4, the suspension deflections stay relatively constant with the passive suspension, while the active suspension utilizes larger suspension deflections at lower frequencies in order to mitigate accelerations. When this is transformed into the frequency domain in figure 9.5, the passive suspension is rel-

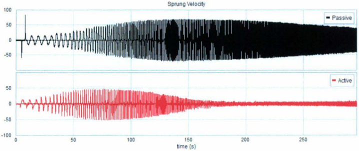


Figure 9.2: Quarter Car Experiment - Sprung Velocity

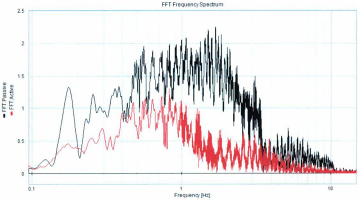


Figure 9.3: Quarter Car Experiment - Sprung Velocity FFT

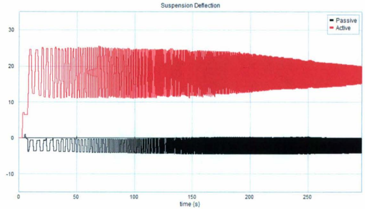


Figure 9.4: Quarter Car Experiment - Suspension Deflection

atively flat, while the active suspension has a shape that indicating high utilization at low frequencies. At approximately 0.1 Hz there is very large suspension deflection that decreases until it is operating at sufficiently high enough frequency road input that this tapers. This seems to again indicate functioning implementation of the LQR scheme.

The unsprung velocities time series and FFT transformed data reveal very little information but are shown in figures 9.6 and 9.7. Previous works indicate very little changes in the unsprung velocity between the active and passive suspensions. Due to the high stiffness of the tire, the velocities and deflections will generally be much lower. Differentiating the positional signal gives a noisy velocity signal which makes a strong inferences difficult. However, looking at maximums and minimums reveal that with active suspension the velocities range from 76.76 mm/s to -75.68 mm/s while the passive range from 73.10 mm/s to -72.69 mm/s. This indicates that the normal trade off of increasing unsprung velocity is likely happening.

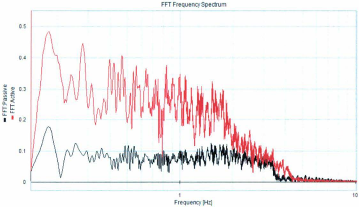


Figure 9.5: Quarter Car Experiment - Suspension Deflection FFT

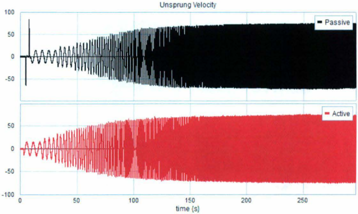


Figure 9.6: Quarter Car Experiment - Unsprung Velocity

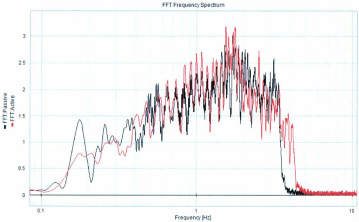


Figure 9.7: Quarter Car Experiment - Unsprung Velocity FFT

Like suspension deflection, tire deflection is generally increased to reduce sprung accelerations. This is visibly apparent in the time and frequency series data seen in figures 9.8 and 9.9. Throughout the entire time series there is more tire deflection than the passive range, and this is emphasized most at lower frequencies. In figure 9.8 it is noticeable that there is an offset between the simulations. This is the result of several cumulative effects brought about by the Coulomb friction. In order to run a test the code has to be zeroed first. This can only be done with the test apparatus stationary. It is possible that when the apparatus is stationary that its positions are not completely representative of the zero point when actuated with a sine wave. The Coulomb friction from the linear bearing can hold the test rig a small amount higher or lower causing an offset in the result.

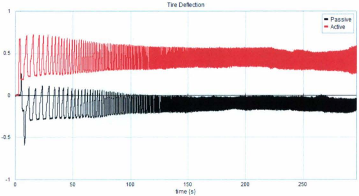


Figure 9.8: Quarter Car Experiment - Tire Deflection

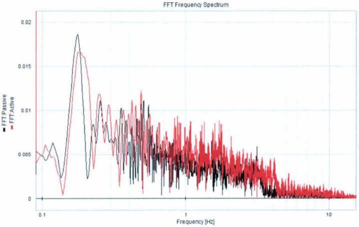


Figure 9.9: Quarter Car Experiment - Tire Deflection FFT

9.5 Simulation With Experimental Data

Recording all states means that the road input is stored, and has the potential to be utilized for simulation at a later time. A comparison of the experimental data and the ideal passive and active simulations are possible. Shown in figure 9.12 is the sprung acceleration results for the experimental and simulation based on the road input for the recorded time series. This has been a double differentiated position signal leading to a significant amount of noise, however it does show that at frequencies above 0.6 Hz the sprung acceleration is reduced. At low frequencies, less than 0.4 Hz approximately, the sprung acceleration is greater for the experimental setup while the theoretical is significantly less than both. It should be noted that the Coulomb friction was not included in the theoretical model used for comparison or controller development. When the frequency approaches 1 Hz the experimental data approximates the theoretical response. A more detailed view of figure 9.12 is shown in fiure 9.13.

If the passive experimental data is used for simulation in a similar fashion the results show good correlations. Figure 9.10 shows the experimental sprung acceleration when compared with that of the simulated using the road input results. At lower frequencies they are nearly identical except for some impulses and flats where the Coulomb friction causes discontinuties, but at higher frequencies there appears to be a higher damping constant than determined from the experimental data. Similar results are seen in the sprung velocity comparison shown in figure 9.11 where the shift in peak to a lower frequency and the reduction in velocity is characteristic of higher damping ratio. Unfortunately, Coulomb friction is difficult to measure in an exper-

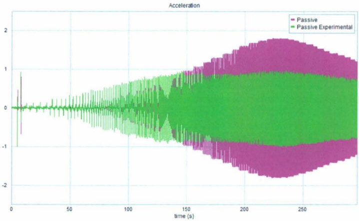


Figure 9.10: Sprung Passive Acceleration Comparison

imental sense, and due to issues with creating a linear time invariant system it has not been included. Work in the future should attempt to incorporate and measure this for more accurate simulations.

In order to reduce the sprung acceleration, the sprung velocity must be reduced as well. This is a less noisy signal as the position has only been differentiated once, as opposed to twice for acceleration. Referring to figure 9.14 it is apparent that at low frequencies the response of the experimental setup is not as effective as the theoretical but does perform better than passive at frequencies above 0.6 Hz. This is again showing successful LQR implementation for ride quality.

When the time series is enlarged in figure 9.15 it can be seen that while the experimental is not as effective as the theoretical, there is a significant reduction in sprung velocities. As the system reaches higher frequencies of actuation the experimental comes much closer to reaching that of the theoretical. This is likely attributable to

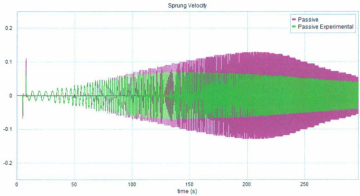


Figure 9.11: Sprung Passive Velocity Comparison

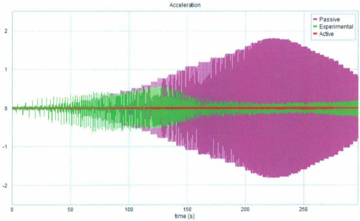


Figure 9.12: Sprung Acceleration Comparison

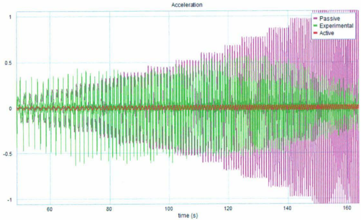


Figure 9.13: Sprung Acceleration Comparison - Detail

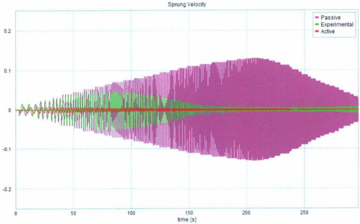


Figure 9.14: Sprung Velocity Comparison

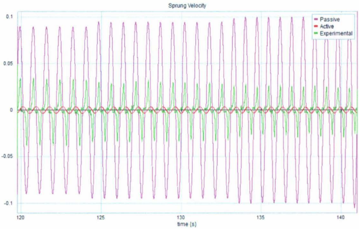


Figure 9.15: Sprung Velocity Comparison Zoomed

Coulomb friction which is much more of an issue at lower velocities. At an actuation frequency of 0.60 Hz the experimental active suspension begin to show noticeable ride quality improvements. Once the actuation frequency reaches 1.56 Hz the experimental reaches a limit in performance. The performance at higher frequencies is maintained until the end of the experiment at approximately 4.03 Hz. There is a mild decrease in performance but it is not significant. Actuation velocities over 5.00 Hz were difficult to achieve with MTS hydraulic actuator. While the data from 4.00 - 5.00 Hz should be available, 20-Sim presented issues, potentially due to a higher required sampling frequency.

When the suspension deflection is compared between theoretical active, passive, and the experimental, several things become apparent when referring to figure 9.16. The major problem of note is that there is considerable issue with zeroing the suspension deflection due to the Coulomb friction. The point at which the suspension moves

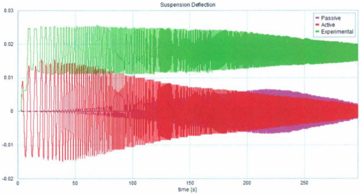


Figure 9.16: Suspension Deflection Comparison

around which normally pertains to the zero is not that of the original set before the experiment is run. This may have positively or negatively influenced the results. The other major point of note is that at lower frequencies the experimental implementation did not use as much suspension deflection. This could potentially be a positive attribute as theoretical implementations require very significant suspension travel at low frequencies from the resulting Coulomb friction.

The experimental tire deflection however did not correlate well with that of the theoretical as seen in figure 9.17. At low frequencies the amount of tire deflection was significantly greater than that of the theoretical, but at higher frequencies of 3.00 to 4.00 Hz, the actual utilized was lower than expected. This relates to not having as high a performance as the theoretical but could be very acceptable in a practical implementation.

As conducted previously, a performance index can be solved for the experimental and theoretical setups and is shown in figure 9.18. The performance of the experi-

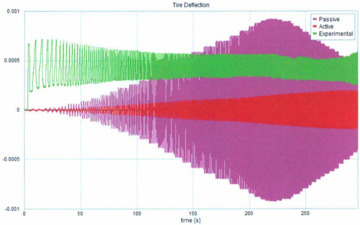


Figure 9.17: Tire Deflection Comparison

mental setup is very close to that of the theoretical active and well below that of the passive suspension. This indicates that for a range of 0.6 Hz to 4.00 Hz that the voice coil active suspension can successfully mitigate road inputs.

At frequencies lower than 0.6 Hz, it is interesting to note that the performance index results are slightly peculiar for the experimental setup. According to figure 9.19, the active suspension in the frequency sweep from 0.1 Hz to 1.40 Hz. Shows that the passive is better at very low frequencies. This can likely be attributed again to the Coulomb friction. While the actuator is trying to apply a force, it is insufficient to overcome the Coulomb friction of the system. The system stops accumulating error and increasing the performance index score until the movements are fast enough that the Coulomb friction is no longer a significant factor. While the theoretical active is least favourable for ride quality at these very low frequencies the experimental implementation is not as harsh. The experimental implementation does as a result

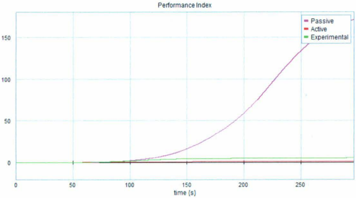


Figure 9.18: Performance Index Comparison

take a little longer to show its performance benefits.

Finally the force actuator is examined in figure 9.20 where it can be seen that the positional offset encountered earlier is leading to a constant amount of force being added. At low frequencies the forces appear truncated as the motion stops due to the Coulomb friction, but at higher frequencies where this is not as apparent the experimental and theoretical force ranges are similar.

9.6 Conclusion

According to the results, the implementation with the FPGA and a voice coil showed very positive results and unlike that of hydraulic or pneumatic implementations referenced previously, the voice coil provided high frequency response of over 4.00 Hz. The limitations of the test equipment meant that benefits at frequencies above 4.00 Hz could not be analyzed, but it appears that it could potentially continue for frequencies potentially as high as the wheel hop frequency. The limitation on this is

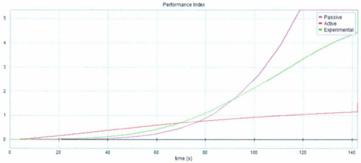


Figure 9.19: Performance Index Comparison Zoomed

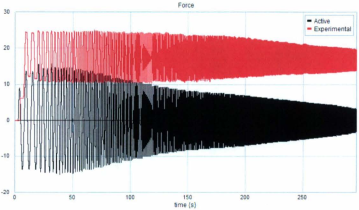


Figure 9.20: Experimental Force Comparison

related to the electrical slew rates of the current sensors, the ADC sampling, and speed and accuracy of the PWM implementation. It also highlights the potential for implementing multiple controllers with ease in an FPGA and opens the door for future developments.

Chapter 10

Conclusion

There are several conclusions that can be drawn over the course of this research each relating to a different area that contributed to the development and testing of the quarter car test apparatus. There has been previous work relating to the simulation of control techniques for a range of model complexity. When the LQR method was initially implemented in the real world, the results were poor. This was attributed to two main issues, Coulomb friction and poor actuator dynamics. The lack of an actuator whose output is a force has been a limitation for active suspension. Recent work has found promising results using linear electric motors which bridge this gap and have a significantly higher bandwidth than hydraulic and pneumatic actuators. It has been from that point which development has preceded on a test apparatus.

10.1 Simulation

Firstly, simulation work began with the most basic active suspension setup, a Linear Quadratic Regulator controlled quarter car active suspension. This work was conducted with Bond Graphs initially as an exercise in skills development. This ultimately led to questions on what is the most effective starting point for a real world implementation. An investigation was started into half and full car controllers. Applying multiple quarter car controllers to higher order models was suggested as a means of reducing controller complexity. Simulations were conducted of two quarter car controllers on a half car model and compared to simulations of a half car controller on a half car model. The performance index used in designing the half car controller was used during simulations to create an objective comparison. The overall finding was that the quarter car controllers were generally superior except in cases where the pitch acceleration was weighted very highly.

These simulations led to several design decisions in order to get approximate weights, stiffnesses, and damping coefficients which would be needed later for design. This allowed very crude work to begin on design and evaluation of potential components.

10.1.1 Future work

While the full car model is developed, the LQR control problem was unsolvable directly. Reducing model complexity through division of the problem, presented by Hrovat [8], could be used as the full car baseline for the full car controller. This could potentially give rise to comparing four quarter car controllers and two different variations of using half car controllers, either aligned for pitch or aligned for roll. However, the findings of the quarter car and half car comparison along with the works of Hrovat indicate that the most optimal controller may actually be the quarter car controller supplemented with the pitch and roll states. This adds only minor complexity but would require extensive simulations to test its performance.

10.2 Five State LQR

One major strength of Bond Graphs is their ability to easily interconnect various systems in the same model and create the governing equations without regard to the type of system. As Bond Graphs proved useful in modelling the quarter and half car controllers, and considering that electric motors are generally a very simple model, it was feasible to expand the normal idea of a four state quarter car (sprung mass velocity, unsprung mass velocity, and tire and suspension deflections) to include the state of the current in the motor coil. This was termed the five state LQR controller. It was generally assumed that the 4 state would be the theoretical limit as it represents the perfect response of the system but it was theorized that once more real world elements were added to the four state model that the five state could potentially be superior as the control loop was shorter, simpler, and closer to the ideal four state.

The simulations of the original systems performed as expected. The theoretical four state controller was superior to all other controllers. This model was augmented by adding a control loop. The model would provide a force target and a proportional error controller with a very high gain was used. Since there was no control delay there was no limit to this gain. As this gain was increased it proved to be superior to that of the five state LQR. When a delay was introduced to the proportional feedback controller that would be representative of the real world the results changed. The proportional gain was very unstable and required a reduction to a value at which point the five state LQR controller was now noticeably superior.

10.2.1 Future Work

The primary motivating factor of the development of a five state LQR controller was that it could easily rely on low cost current sensors rather than high cost load cells. Due to lack of development time the five state LQR never saw implementation on the test rig. A four state controller was implemented first as a baseline. Modifications to this controller should be of very high priority to compare the controllers in real test setup.

While it was already shown that most scenarios could use quarter car controllers with good results there is still room for a simulation exercise implementing a variation of the five state controller on a half car model. This could potentially be referred to as the ten state half car and compared with the 8 state half car (analog to 4 state quarter car) once control delays are introduced.

10.3 Quarter Car Apparatus

Throughout the development of simulations, several cases were chosen and refined to provide estimated parameters for the development of the quarter car test apparatus. From these values a significant amount of time and resources were utilized in searching for the most appropriate actuator. High cost and lower than expected suitability meant that some development of an actuator was conducted. Due to the previous success of linear motors it was determined to develop a three phase brushless linear motor, the primary idea being that the limiting factor would be the length and coil sets. This proved more difficult to develop and had less than impressive results relating to directional flux density.

New magnetostatic simulation work indicated that field strength for a voice coil arrangement would be significantly higher as well as provide more area for wire. Utilizing available parts from the linear motor design a prototype voice coil was developed. The intial prototype appeared sucessful but was never designed to actually be used in a test apparatus. A second version was designed, but due to using a different shape and thickness of magnet, the flux density again proved to be much lower than expected. While it was determined that this problem could be rectified through another redesign a significant amount of time had been utilized and was holding up the development of the test apparatus and it was inconclusive if another iteration would be usable or if more work would still be required. The approximate values for these actuators were then used to determine the specifications for an off the shelf unit. Once an available production voice coil was chosen a more finalized design could be developed of the quarter car apparatus.

One of the major issues when switching to a production voice coil was that it required linear bearings to support its alignment. Due to the size of the setup square linear rails were chosen for the design. The choice for square rails was motivated by reducing complexity and any potential alignment issues when compared to round rails. The size was chosen mainly based on the estimated sized of other components for mounting and construction. Insufficient attention was paid to the frictional aspects of the larger linear bearings which resulted in some non-linearities.

Other components were chosen relating to the size and weights such as the coil over shock, allowable block mass, and tire stiffness. Several provisions were made for future work to keep the apparatus as expandable as possible.

10.3.1 Future Work

The quarter car test apparatus has great potential for testing the real world performance benefits of the five state LQR, but also has potential for use in preview active control. With the built in provisions for mounting real tires an examination of tire damping can be investigated and aggressive simulations where the wheel separates. In order to mitigate the Coulomb frictional issues, a larger value for the unsprung mass should be utilized. The other options include replacing the rail for a lower frictional design. Damping at various suspension settings and spring settings should be investigated using Design of Experiments (DOE).

10.4 Controller and Logging

One of the expected developments to happen is elimination of some sensors, such as the one for tire deflection, and replacement with Kalman filters to estimate these states. This is a very processing intensive algorithm and can easily tax microcontrollers to the point of taking up to a second to process depending on the algorithm. It was decided that an FPGA might be a good alternative with the downside being longer development times. This meant the initial four state LQR controller was developed in the language VHDL with fixed point mathematics techniques. Fixed point math offered a compromise to floating point math that microcontrollers normally use for reduced memory requirements. Development was significantly slower than that of a microcontroller but with one specific benefit; the system could do things in parallel and at perfectly repeatable time intervals. When the state of a sensor is sampled the other sensors are sampled at the exact same time. This is not generally possible with a micro controller and considering that the analog signals were digitized and differentiated due to lack of a velocity sensor, this ability was especially desirable.

Since multiple 12 bit ADC chips were easily available, without requiring surface mount technique, it meant that at higher sample rates the differentiation of the signals could be noisy. In order to combat this a running average technique was adopted and implemented. This appeared to clean up the signal when the outputs were observed but no empirical measurements were taken.

The four state LQR is implemented with fixed point math with a proportional feedback control. The voice coil has a calibrated current sensor which means that the signal is scaled from a numerical value to that of a force by combining two scaling constants (Amps/bit * Newtons/Amp). This showed good results for the quarter car controller especially over 1.2 Hz. Previous works indicate a 2 Hz limit for hydraulic actuators but even with Coulomb frictional issues, the four state LQR showed good results up to 4 Hz.

10.4.1 Future Work

Since the controller is in its infancy, there are significant improvements to be made. As mentioned previously, the five state LQR should be implemented and tested against the four state LQR. Effective implementation of Kalman filtering algorithms will require very clever programming techniques and fixed point or floating point mathematics. Due to the availability of VHDL fixed point packages that will soon be standard for FPGA development it is strongly suggested that development be conducted in VHDL otherwise the significant amount of fixed point math that would be required and manually tracked would be overwhelming.

Bibliography

[1] Mechanical Simulation Corporation, "Carsim." www.carsim.com, 2011.

[2] S. Brennan and A. Alleyne, "A scaled testbed for vehicle control: the IRS," in *Control Applications, 1999. Proceedings of the 1999 IEEE International Conference on*, vol. 1, pp. 327 – 332, 1999.

[3] S. Brennan and A. Alleyne, "Using a scale testbed: Controller design and evaluation," *Control Systems, IEEE*, vol. 21, pp. 15 – 26, June 2001.

[4] S. N. Brennan, "Modeling and control issues associated with scale vehicles," Master's thesis, University of Illinois at Urbana-Champaign, 1999.

[5] S. Brennan and A. Alleyne, "The Illinois Roadway Simulator: A mechatronic testbed for vehicle dynamics and control," *Mechatronics, IEEE/ASME Transactions on*, vol. 5, pp. 349 – 359, December 2000.

[6] S. Brennan and A. Alleyne, "Robust scalable vehicle control via non-dimensional vehicle dynamics," in *The 5 th International Symposium on Advanced Vehicle Control: AVEC 2000, Ann Arbor, MI, Proceedings of*, pp. 713 – 720, 2001.

[7] B. Gysen, J. Janssen, J. Paulides, and E. Lomonova, "Design aspects of an active electromagnetic suspension system for automotive applications," *Industry Applications, IEEE Transactions on*, vol. 45, pp. 1589 – 1597, September - October 2009.

[8] D. Hrovat, "Survey of advanced suspension developments and related optimal control applications,," *Automatica*, vol. 33, no. 10, pp. 1781 – 1817, 1997.

[9] T. Tseng and D. Hrovat, "Some characteristics of optimal vehicle suspensions based on quarter-car models," in *Decision and Control, Proceedings of the 29th IEEE Conference on*, vol. 4, pp. 2232 – 2237, December 1990.

[10] R. Krtolica and D. Hrovat, "Optimal active suspension control based on a half-car model: an analytical solution," *Automatic Control, IEEE Transactions on*, vol. 37, pp. 528 – 532, April 1992.

[11] J. Wong, *Theory of ground vehicles.* Wiley, 2008.

[12] D. C. Karnopp, M. J. Crosby, and R. A. Harwood, "Vibration control using semi-active force generators," *Journal of Engineering for Industry*, vol. 5, no. 96, pp. 619 – 626, 1974.

[13] P. Wright and T. Matthews, *Formula 1 Technology.* Society of Automotive Engineers, 2001.

[14] S. M. Savaresi, C. Poussot-Vassal, C. Spelta, O. Sename, and L. Dugard, *Semi-Active Suspension Control Design for Vehicles.* Boston: Butterworth-Heinemann, 2010.

[15] M. D. Donahue, "Implementation of an active suspension, preview controller for improved ride comfort," Master's thesis, The University of California at Berkeley, 2001.

[16] S. N. Brennan, *On Size and Control: The Use of Dimensional Analysis in Controller Design.* PhD thesis, University of Illinois at Urbana-Champaign, 2002.

[17] S. Brennan and A. Alleyne, "Driver assisted yaw rate control," in *American Control Conference, 2001. Proceedings of the 2001*, vol. 1, pp. 1697 – 1701, June 1999.

[18] S. Brennan and A. Alleyne, "Integrated vehicle control via coordinated steering and wheel torque inputs," in *American Control Conference, 2001. Proceedings of the 2001*, vol. 1, pp. 7 – 12, 2001.

[19] M. Petersheim and S. Brennan, "Scaling of hybrid electric vehicle powertrain components for hardware-in-the-loop simulation," in *Control Applications, 2008. CCA 2008. IEEE International Conference on*, pp. 720 – 726, September 2008.

[20] H. Pacejka and S. of Automotive Engineers, *Tire and vehicle dynamics.* SAE-R, SAE International, 2006.

[21] M. Polley and A. Alleyne, "Dimensionless analysis of tire characteristics for vehicle dynamics studies," in *American Control Conference, Proceedings of the 2004*, vol. 4, pp. 3411 – 3416, July 2004.

[22] W. Witaya, W. Parinya, and C. Krissada, "Scaled vehicle for interactive dynamic simulation (sis)," in *Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on*, pp. 554 – 559, February 2009.

[23] A. D. Gulve, "Vehicle rollover anaylsis using 1/10th scale model and comparison with adams simulation software," Master's thesis, University of Detroit Mercy, 2005.

[24] R. Verma, D. Del Vecchio, and H. Fathy, "Development of a scaled vehicle with longitudinal dynamics of an hmmwv for an its testbed," *Mechatronics, IEEE/ASME Transactions on*, vol. 13, pp. 46 – 57, February 2008.

[25] N. D. Annis and S. C. Southward, "Scale-model test rig for passive and active suspension control comparison," *ASME Conference Proceedings*, vol. 2007, no. 48043, pp. 1247 – 1252, 2007.

[26] J. Langdon and S. C. Southward, "Development of a general use quarter-vehicle test rig," *ASME Conference Proceedings*, vol. 2007, no. 48043, pp. 1239 – 1245, 2007.

[27] S. Chantranuwathana and H. Peng, "Force tracking control for active suspensions-theory and experiments," in *Control Applications, 1999. Proceedings of the 1999 IEEE International Conference on*, vol. 1, pp. 442 – 447, 1999.

[28] W. Anakwa, D. Thomas, S. Jones, J. Bush, D. Green, G. Anglin, R. Rio, J. Sheng, S. Garrett, and L. Chen, "Development and control of a prototype pneumatic active suspension system," *Education, IEEE Transactions on*, vol. 45, pp. 43 – 49, February 2002.

[29] C. Lauwerys, J. Swevers, and P. Sas, "Design and experimental validation of a linear robust controller for an active suspension of a quarter car," in *American Control Conference, 2004. Proceedings of the 2004*, vol. 2, pp. 1481 – 1486, July 2004.

[30] B. Gysen, T. van der Sande, J. Paulides, and E. Lomonova, "Efficiency of a regenerative direct-drive electromagnetic active suspension," *Vehicular Technology, IEEE Transactions on*, vol. 60, pp. 1384 – 1393, May 2011.

[31] S. Lee and W. jong Kim, "Active suspension control with direct-drive tubular linear brushless permanent-magnet motor," *Control Systems Technology, IEEE Transactions on*, vol. 18, pp. 859 – 870, July 2010.

[32] D. C. Karnopp, D. L. Margolis, and R. C. Rosenberg, *System Dynamics Modeling and Simulation of Mechatronic Systems 4th edition.* John Wiley and Sons, Inc., second ed., 2006.

[33] R. Darus and Y. M. Sam, "Modeling and control of active suspension for a full car model," *5th International Colloquium on Signal Processing & Its Applications*, pp. 13 – 18, 2009.

[34] K. Ogata, *Modern Control Engineering (4th Edition).* Prentice Hall, November 2001.

[35] T. Butsuen, *The design of semi-active suspensions for automotive vehicles.* PhD thesis, Massachusetts Institute of Technology, 1989.

[36] P. Belanger, *Control Engineering A Modern Approach.* Oxford University Press, Inc., first ed., 1995.

[37] A. Tewari, *Modern control design with MATLAB and SIMULINK.* John Wiley, 2002.

[38] R. Dorf and R. Bishop, *Modern control systems.* Pearson Prentice Hall, 2011.

[39] I. The MathWorks, "Matlab v2009b." www.mathworks.com, 2009.

[40] 3 Sigma SARL, "Control toolbox." 3sigma.fr, 2010.

[41] Maplesoft, "Maple v13.0." www.maplesoft.com, 2009.

[42] I. Arnold, W.F. and A. Laub, "Generalized eigenproblem algorithms and software for algebraic Riccati equations," *Proceedings of the IEEE*, vol. 72, pp. 1746 – 1754, December 1984.

[43] R. Rajamani, *Vehicle dynamics and control.* Springer, second ed., 2006.

[44] H. Adibi-asl and G. Rideout, "Bond graph modeling and simulation of a full car model with active suspension," in *Proceedings of the 20th IASTED International Conference on Modelling and Simulation*, 2009.

[45] J. Maiorana, "Cosimulation of an active suspension," Master's thesis, University of Windsor, 2004.

[46] Controllab Products B.V., "20-sim v4.1.05." www.20sim.com, 2009.

[47] F. Tyan, Y.-F. Hong, S.-H. Tu, and W. S. Jent, "Generation of random road profiles," *Journal of Advanced Engineering*, pp. 1373 – 1378, 2009.

[48] B. L. J. Gysen, J. Paulides, J. L. G. Jannssen, and E. A. Lomonova, "Active electromagnetic suspension system for improved vehicle dynamics," *Vehicular Technology, IEEE Transactions on*, vol. 59, pp. 1156 – 1163, 2010.

[49] R. Redfield and C. Sutela, "A bond graph model of a full-suspension mountain bicycle rear shock," in *The Engineering of Sport 6* (E. F. Moritz and S. Haake, eds.), pp. 109 – 114, Springer New York, 2006.

[50] H. Lu, J. Zhu, Z. Lin, and Y. Guo, "A miniature short stroke linear actuator; design and analysis," *Magnetics, IEEE Transactions on*, vol. 44, pp. 497 – 504, April 2008.

[51] J. Paulides, L. Encica, E. Lomonova, and A. Vandenput, "Active roll compensation for automotive applications using a brushless direct-drive linear permanent magnet actuator," in *Power Electronics Specialists Conference, 2006. PESC '06. 37th IEEE*, pp. 1 – 6, June 2006.

[52] S.-M. Jang, J.-Y. Choi, H.-W. Cho, and S.-H. Lee, "Thrust analysis and measurements of tubular linear actuator with cylindrical halbach array," *Magnetics, IEEE Transactions on*, vol. 41, pp. 2028 – 2031, May 2005.

[53] ANSYS Inc., "Ansys Workbench 13.0.0." www.ansys.com, 2010.

[54] J. McBean and C. Breazeal, "Voice coil actuators for human-robot interaction," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 1, pp. 852 – 858, September - October 2004.

[55] S. Brown and Z. Vranesic, *Fundamentals of digital logic with VHDL design.* McGraw-Hill series in electrical and computer engineering, McGraw-Hill, 2009.

[56] D. Perry, *VHDL: Programming by example.* McGraw-Hill, 2002.

[57] S. Kilts, *Advanced FPGA design: architecture, implementation, and optimization.* Wiley, 2007.

[58] Arduino, "Arduino 0022." Open Source Software, www.arduino.cc, 2011.

Appendix A

Full Car Linear Model

$$\begin{bmatrix} a(1,1) & a(1,2) & \cdots & a(1,15) \\ a(2,1) & a(2,2) & \cdots & a(2,15) \\ \vdots & \vdots & \ddots & \vdots \\ a(15,1) & a(15,2) & \cdots & a(15,15) \end{bmatrix} \tag{A.1}$$

$$a(1,1) = -\frac{B_{sfl}}{m_s} - \frac{B_{srl}}{m_s} - \frac{B_{srr}}{m_s} - \frac{B_{sfr}}{m_s}$$

$$a(1,2) = -\frac{B_{srl}b}{m_s} + \frac{B_{sfl}a}{m_s} - \frac{B_{srr}b}{m_s} + \frac{B_{sfr}a}{m_s}$$

$$a(1,3) = -1/2\,\frac{B_{sfl}\,Tf}{m_s} + 1/2\,\frac{B_{srr}\,Tr}{m_s} - 1/2\,\frac{B_{srl}\,Tr}{m_s} + 1/2\,\frac{B_{sfr}\,Tf}{m_s}$$

$$a(1,4) = \frac{k_{sfr}}{m_s}$$

$$a(1,5) = \frac{k_{srr}}{m_s}$$

$$a(1,6) = \frac{k_{sfl}}{m_s}$$

$$a(1,7) = \frac{k_{srl}}{m_s}$$

$$a(1,8) = -\frac{B_{sfr}}{m_s}$$

$$a(1,9) = -\frac{B_{srr}}{m_s}$$

$$a(1,10) = -\frac{B_{sfl}}{m_s}$$

$$a(1,11) = -\frac{B_{srl}}{m_s}$$

$$a(1,12) = a(1,13) = a(1,14) = a(1,15) = 0$$

$$a(2,1) = -\frac{B_{srr}b}{Ixx} + \frac{B_{sfl}a}{Ixx} - \frac{B_{srl}b}{Ixx} + \frac{B_{sfr}a}{Ixx}$$

$$a(2,2) = -\frac{B_{srr}b^2}{Ixx} - \frac{B_{sfl}a^2}{Ixx} - \frac{B_{srl}b^2}{Ixx} - \frac{B_{sfr}a^2}{Ixx}$$

$$a(2,3) = 1/2\,\frac{B_{srr}b\,Tr}{Ixx} + 1/2\,\frac{B_{sfl}a\,Tf}{Ixx} - 1/2\,\frac{B_{srl}b\,Tr}{Ixx} - 1/2\,\frac{B_{sfr}a\,Tf}{Ixx}$$

$$a(2,4) = -\frac{ak_{sfr}}{Ixx}$$

$$a(2,5) = \frac{bk_{srr}}{Ixx}$$

$$a(2,6) = -\frac{ak_{sfl}}{Ixx}$$

$$a(2,7) = \frac{bk_{srl}}{Ixx}$$

$$a(2,8) = \frac{B_{sfr}a}{Ixx}$$

$$a(2,9) = -\frac{B_{srr}b}{Ixx}$$

$$a(2,10) = \frac{B_{sfl}a}{Ixx}$$

$$a(2,11) = -\frac{B_{srl}b}{Ixx}$$

$$a(2,12) = a(2,13) = a(2,14) = a(2,15) = 0$$

$$a(3,1) = 1/2\,\frac{B_{srr}\,Tr}{Iyy} + 1/2\,\frac{B_{sfr}\,Tf}{Iyy} - 1/2\,\frac{B_{srl}\,Tr}{Iyy} - 1/2\,\frac{B_{sfl}\,Tf}{Iyy}$$

$$a(3,2) = 1/2\,\frac{B_{srr}b\,Tr}{Iyy} - 1/2\,\frac{B_{sfr}a\,Tf}{Iyy} - 1/2\,\frac{B_{srl}b\,Tr}{Iyy} + 1/2\,\frac{B_{sfl}a\,Tf}{Iyy}$$

$$a(3,3) = -1/4\,\frac{B_{srr}\,Tr^2}{Iyy} - 1/4\,\frac{B_{sfr}\,Tf^2}{Iyy} - 1/4\,\frac{B_{srl}\,Tr^2}{Iyy} - 1/4\,\frac{B_{sfl}\,Tf^2}{Iyy}$$

$$a(3,4) = -1/2\,\frac{Tf\,k_{sfr}}{Iyy}$$

$$a(3,5) = -1/2\,\frac{Tr\,k_{srr}}{Iyy}$$

$$a(3,6) = 1/2\,\frac{Tf\,k_{sfl}}{Iyy}$$

$$a(3,7) = 1/2\,\frac{Tr\,k_{srl}}{Iyy}$$

$$a(3,8) = 1/2\,\frac{B_{sfr}\,Tf}{Iyy}$$

$$a(3,9) = 1/2\,\frac{B_{srr}\,Tr}{Iyy}$$

$$a(3,10) = -1/2\,\frac{B_{sfl}\,Tf}{Iyy}$$

$$a(3,11) = -1/2\,\frac{B_{srl}\,Tr}{Iyy}$$

$$a(3,12) = a(3,13) = a(3,14) = a(3,15) = 0$$

$$a(4,1) = -1$$

$$a(4,2) = 1/2\,itTf$$

$$a(4,3) = a(4,4) = a(4,5) = a(4,6) = 0$$

$$a(4,7) = -1$$

$$a(4,8) = a(4,9) = a(4,10) = a(4,11) = a(4,12) = a(4,13) = a(4,14) = a(4,15) = 0$$

$$a(5,1) = -1$$

$$a(5,2) = b$$

$$a(5,3) = 1/2\,Tr$$

$$a(5,4) = a(5,5) = a(5,6) = a(5,7) = a(5,8) = 0$$

$$a(5,9) = -1$$

$$a(5,10) = a(5,11) = a(5,12) = a(5,13) = a(5,14) = a(5,15) = 0$$

$$a(6,1) = -1$$

$$a(6,2) = a$$

$$a(6,3) = -1/2\ Tf$$

$$a(6,4) = a(6,5) = a(6,6) = a(6,7) = a(6,8) = a(6,9) = 0$$

$$a(6,10) = -1$$

$$a(6,11) = a(6,12) = a(6,13) = a(6,14) = a(6,15) = 0$$

$$a(7,1) = -1$$

$$a(7,2) = -b$$

$$a(7,3) = -1/2\ Tr$$

$$a(7,4) = a(7,5) = a(7,6) = a(7,7) = a(7,8) = a(7,9) = a(7,10) = 0$$

$$a(7,11) = -1$$

$$a(7,12) = a(7,13) = a(7,14) = a(7,15) = 0$$

$$a(8,1) = -\frac{B_{sfr}}{m_{ufr}}$$

$$a(8,2) = \frac{B_{sfr}a}{m_{ufr}}$$

$$a(8,3) = 1/2\,\frac{B_{sfr}\,Tf}{m_{ufr}}$$

$$a(8,4) = \frac{k_{sfr}}{m_{ufr}}$$

$$a(8,5) = a(8,6) = a(8,7) = 0$$

$$a(8,8) = -\frac{B_{sfr}}{m_{ufr}} - \frac{B_{tfr}}{m_{ufr}}$$

$$a(8,9) = a(8,10) = a(8,11) = 0$$

$$a(8,12) = -\frac{k_{tfr}}{m_{ufr}}$$

$$a(8,13) = a(8,14) = a(8,15) = 0$$

$$a(9,1) = -\frac{B_{srr}}{m_{urr}}$$

$$a(9,2) = -\frac{B_{srr}b}{m_{urr}}$$

$$a(9,3) = 1/2\,\frac{B_{srr}\,Tr}{m_{urr}}$$

$$a(9,4) = 0$$

$$a(9,5) = \frac{k_{srr}}{m_{urr}}$$

$$a(9,6) = a(9,7) = a(9,8) = 0$$

$$a(9,9) = -\frac{B_{srr}}{m_{urr}} - \frac{B_{trr}}{m_{urr}}$$

$$a(9,10) = a(9,11) = a(9,12) = 0$$

$$a(9,13) = -\frac{k_{trr}}{m_{urr}}$$

$$a(9,14) = a(9,15) = 0$$

$$a(10,1) = -\frac{B_{sfl}}{m_{ufl}}$$

$$a(10,2) = \frac{B_{sfl}a}{m_{ufl}}$$

$$a(10,3) = -1/2\,\frac{B_{sfl}\,Tf}{m_{ufl}}$$

$$a(10,4) = a(10,5) = 0$$

$$a(10,6) = \frac{k_{sfl}}{m_{ufl}}$$

$$a(10,7) = a(10,8) = a(10,9) = 0$$

$$a(10,10) = -\frac{B_{sfl}}{m_{ufl}} - \frac{B_{tfl}}{m_{ufl}}$$

$$a(10,11) = a(10,12) = a(10,13) = 0$$

$$a(10,14) = -\frac{k_{tfl}}{m_{ufl}}$$

$$a(10,15) = 0$$

$$a(11,1) = -\frac{B_{srl}}{m_{url}}$$

$$a(11,2) = -\frac{B_{srl} b}{m_{url}}$$

$$a(11,3) = -1/2\,\frac{B_{srl}\,Tr}{m_{url}}$$

$$a(11,4) = a(11,5) = a(11,6) = 0$$

$$a(11,7) = \frac{k_{srl}}{m_{url}}$$

$$a(11,8) = a(11,9) = a(11,10) = 0$$

$$a(11,11) = -\frac{B_{srl}}{m_{url}} - \frac{B_{trl}}{m_{url}}$$

$$a(11,12) = a(11,13) = a(11,14) = 0$$

$$a(11,15) = -\frac{k_{trl}}{m_{url}}$$

$$a(12,1) = a(12,2) = a(12,3) = a(12,4) = a(12,5) = a(12,6) = a(12,7) = 0$$

$$a(12,8) = 1$$

$$a(12,9) = a(12,10) = a(12,11) = a(12,12) = a(12,13) = a(12,14) = a(12,15) = 0$$

$$a(13,1) = a(13,2) = a(13,3) = a(13,4) = a(13,5) = a(13,6) = a(13,7) = a(13,8) = 0$$

$$a(13,9) = 1$$

$$a(13,10) = a(13,11) = a(13,12) = a(13,13) = a(13,14) = a(13,15) = 0$$

$$a(14,1) = a(14,2) = a(14,3) = a(14,4) = a(14,5) = a(14,6) = a(14,7) = 0$$

$$a(14,8) = a(14,9) = 0$$

$$a(14,10) = 1$$

$$a(14,11) = a(14,12) = a(14,13) = a(14,14) = a(14,15) = 0$$

$$a(15,1) = a(15,2) = a(15,3) = a(15,4) = a(15,5) = a(15,6) = a(15,7) = 0$$

$$a(15,8) = a(15,9) = a(15,10) = 0$$

$$a(15,11) = 1$$

$$a(15,12) = a(15,13) = a(15,14) = a(15,15) = 0$$

$$\begin{bmatrix} b(1,1) & b(1,2) & b(1,3) & b(1,4) \\ b(2,1) & b(2,2) & b(1,3) & b(2,4) \\ \vdots & \vdots & \vdots & \vdots \\ b(15,1) & b(15,2) & \cdots & b(15,4) \end{bmatrix} \tag{A.2}$$

$$b(1,1) = -{m_s}^{-1}$$

$$b(1,2) = -{m_s}^{-1}$$

$$b(1,3) = -{m_s}^{-1}$$

$$b(1,4) = -{m_s}^{-1}$$

$$b(2,1) = \frac{a}{Ixx}$$

$$b(2,2) = -\frac{b}{Ixx}$$

$$b(2,3) = \frac{a}{Ixx}$$

$$b(2,4) = -\frac{b}{Ixx}$$

$$b(3,1) = 1/2\,\frac{Tf}{Iyy}$$

$$b(3,2) = 1/2\,\frac{Tr}{Iyy}$$

$$b(3,3) = -1/2\,\frac{Tf}{Iyy}$$

$$b(3,4) = -1/2\,\frac{Tr}{Iyy}$$

$$b(4,1) = b(4,2) = b(4,3) = b(4,4) = 0$$

$$b(5,1) = b(5,2) = b(5,3) = b(5,4) = 0$$

$$b(6,1) = b(6,2) = b(6,3) = b(6,4) = 0$$

$$b(7,1) = b(7,2) = b(7,3) = b(7,4) = 0$$

$$b(8,1) = -{m_{ufr}}^{-1}$$

$$b(8,2) = b(8,3) = b(8,4) = 0$$

$$b(9,1) = 0$$

$$b(9,2) = -{m_{urr}}^{-1}$$

$$b(9,3) = b(9,4) = 0$$

$$b(10,1) = b(10,2) = 0$$

$$b(10,3) = -{m_{ufl}}^{-1}$$

$$b(10,4) = 0$$

$$b(11,1) = b(11,2) = b(11,3) = 0$$

$$b(11,4) = -{m_{url}}^{-1}$$

$$b(12,1) = b(12,2) = b(12,3) = b(12,4) = 0$$

$$b(13,1) = b(13,2) = b(13,3) = b(13,4) = 0$$

$$b(14,1) = b(14,2) = b(14,3) = b(14,4) = 0$$

$$b(15,1) = b(15,2) = b(15,3) = b(15,4) = 0$$

Appendix B

Half Car Performance Index

$$\boldsymbol{Q} = \begin{bmatrix} wp_{11} & wp_{12} & wp_{13} & wp_{14} & wp_{15} & wp_{16} & wp_{17} & wp_{18} \\ wp_{21} & wp_{22} & wp_{23} & wp_{24} & wp_{25} & wp_{26} & wp_{27} & wp_{28} \\ wp_{31} & wp_{32} & wp_{33} & wp_{34} & wp_{35} & wp_{36} & wp_{37} & wp_{38} \\ wp_{41} & wp_{42} & wp_{43} & wp_{44} & wp_{45} & wp_{46} & wp_{47} & wp_{48} \\ wp_{51} & wp_{52} & wp_{53} & wp_{54} & wp_{55} & wp_{56} & wp_{57} & wp_{58} \\ wp_{61} & wp_{62} & wp_{63} & wp_{64} & wp_{65} & wp_{66} & wp_{67} & wp_{68} \\ wp_{71} & wp_{72} & wp_{73} & wp_{74} & wp_{75} & wp_{76} & wp_{77} & wp_{78} \\ wp_{81} & wp_{82} & wp_{83} & wp_{84} & wp_{85} & wp_{86} & wp_{87} & wp_{88} \end{bmatrix} \tag{B.1}$$

$$\begin{aligned} wp_{11} = & \frac{\rho_1 bl^2 {B_{sr}}^2}{Ixx^2} + \frac{\rho_1 al^2 {B_{sf}}^2}{Ixx^2} + \rho_7 + \frac{{B_{sf}}^2}{{m_s}^2} \\ & + \frac{{B_{sr}}^2}{{m_s}^2} - 2\,\frac{\rho_1 al\, B_{sf}\, bl\, B_{sr}}{Ixx^2} + 2\,\frac{B_{sf} B_{sr}}{{m_s}^2} \end{aligned}$$

$$\begin{aligned} wp_{12} = & -\frac{{B_{sf}}^2 al}{{m_s}^2} + \frac{{B_{sr}}^2 bl}{{m_s}^2} + \frac{\rho_1 B_{sf}\, al^2 bl\, B_{sr}}{Ixx^2} - \frac{\rho_1 al\, B_{sf} B_{sr} bl^2}{Ixx^2} \\ & + \frac{B_{sf} B_{sr} bl}{{m_s}^2} - \frac{B_{sf}\, al\, B_{sr}}{{m_s}^2} - \frac{\rho_1 {B_{sf}}^2 al^3}{Ixx^2} + \frac{\rho_1 {B_{sr}}^2 bl^3}{Ixx^2} \end{aligned}$$

$$wp_{13} = -\frac{k_{sf} B_{sf}}{{m_s}^2} - \frac{k_{sf} B_{sr}}{{m_s}^2} + \frac{\rho_1 al\, k_{sf}\, bl\, B_{sr}}{Ixx^2} - \frac{\rho_1 al^2 k_{sf} B_{sf}}{Ixx^2}$$

$$wp_{14} = -\frac{B_{sf} k_{sr}}{{m_s}^2} - \frac{k_{sr} B_{sr}}{{m_s}^2} + \frac{\rho_1 al\, B_{sf}\, bl\, k_{sr}}{Ixx^2} - \frac{\rho_1 bl^2 k_{sr} B_{sr}}{Ixx^2}$$

$$wp_{15} = \frac{B_{sf} B_{sr}}{{m_s}^2} - \frac{\rho_1 al\, B_{sf}\, bl\, B_{sr}}{Ixx^2} + \frac{\rho_1 al^2 {B_{sf}}^2}{Ixx^2} + \frac{{B_{sf}}^2}{{m_s}^2}$$

$$wp_{16} = \frac{B_{sf} B_{sr}}{{m_s}^2} - \frac{\rho_1 al\, B_{sf}\, bl\, B_{sr}}{Ixx^2} + \frac{\rho_1 bl^2 {B_{sr}}^2}{Ixx^2} + \frac{{B_{sr}}^2}{{m_s}^2}$$

$$wp_{17}, wp_{18} = 0$$

$$\begin{aligned} wp_{21} = & -\frac{{B_{sf}}^2 al}{{m_s}^2} + \frac{{B_{sr}}^2 bl}{{m_s}^2} + \frac{\rho_1 B_{sf}\, al^2 bl\, B_{sr}}{Ixx^2} - \frac{\rho_1 al\, B_{sf} B_{sr} bl^2}{Ixx^2} \\ & + \frac{B_{sf} B_{sr} bl}{{m_s}^2} - \frac{B_{sf}\, al\, B_{sr}}{{m_s}^2} - \frac{\rho_1 {B_{sf}}^2 al^3}{Ixx^2} + \frac{\rho_1 {B_{sr}}^2 bl^3}{Ixx^2} \end{aligned}$$

$$\begin{aligned} wp_{22} = \rho_3 - 2\,\frac{B_{sf}\, al\, B_{sr}\, bl}{{m_s}^2} + 2\,\frac{\rho_1 B_{sf}\, al^2 B_{sr}\, bl^2}{Ixx^2} + \frac{\rho_1 {B_{sf}}^2 al^4}{Ixx^2} + \frac{\rho_1 {B_{sr}}^2 bl^4}{Ixx^2} \\ + \frac{{B_{sf}}^2 al^2}{{m_s}^2} + \frac{{B_{sr}}^2 bl^2}{{m_s}^2} \end{aligned}$$

$$wp_{23} = -\frac{k_{sf} B_{sr}\, bl}{{m_s}^2} + \frac{k_{sf} B_{sf}\, al}{{m_s}^2} + \frac{\rho_1 al^3 k_{sf} B_{sf}}{Ixx^2} + \frac{\rho_1 al\, k_{sf} B_{sr}\, bl^2}{Ixx^2}$$

$$wp_{24} = -\frac{k_{sr} B_{sr}\, bl}{{m_s}^2} + \frac{B_{sf}\, al\, k_{sr}}{{m_s}^2} - \frac{\rho_1 bl^3 k_{sr} B_{sr}}{Ixx^2} - \frac{\rho_1 B_{sf}\, al^2 bl\, k_{sr}}{Ixx^2}$$

$$wp_{25} = \frac{B_{sf} B_{sr}\, bl}{{m_s}^2} - \frac{\rho_1 {B_{sf}}^2 al^3}{Ixx^2} - \frac{{B_{sf}}^2 al}{{m_s}^2} - \frac{\rho_1 al\, B_{sf} B_{sr}\, bl^2}{Ixx^2}$$

$$wp_{26} = -\frac{B_{sf}\, al\, B_{sr}}{{m_s}^2} + \frac{\rho_1 {B_{sr}}^2 bl^3}{Ixx^2} + \frac{{B_{sr}}^2 bl}{{m_s}^2} + \frac{\rho_1 B_{sf}\, al^2 bl\, B_{sr}}{Ixx^2}$$

$$wp_{27}, wp_{28} = 0$$

$$wp_{31} = -\frac{k_{sf} B_{sf}}{{m_s}^2} - \frac{k_{sf} B_{sr}}{{m_s}^2} + \frac{\rho_1 al\, k_{sf}\, bl\, B_{sr}}{Ixx^2} - \frac{\rho_1 al^2 k_{sf} B_{sf}}{Ixx^2}$$

$$wp_{32} = -\frac{k_{sf}\,B_{sr}\,bl}{{m_s}^2} + \frac{k_{sf}\,B_{sf}\,al}{{m_s}^2} + \frac{\rho_1\,al^3 k_{sf}\,B_{sf}}{Ixx^2} + \frac{\rho_1\,al\,k_{sf}\,B_{sr}\,bl^2}{Ixx^2}$$

$$wp_{33} = \frac{\rho_1\,al^2 {k_{sf}}^2}{Ixx^2} + \frac{{k_{sf}}^2}{{m_s}^2} + \rho_2$$

$$wp_{34} = \frac{k_{sf}\,k_{sr}}{{m_s}^2} - \frac{\rho_1\,al\,k_{sf}\,bl\,k_{sr}}{Ixx^2}$$

$$wp_{35} = -\frac{\rho_1\,al^2 k_{sf}\,B_{sf}}{Ixx^2} - \frac{k_{sf}\,B_{sf}}{{m_s}^2}$$

$$wp_{36} = -\frac{k_{sf}\,B_{sr}}{{m_s}^2} + \frac{\rho_1\,al\,k_{sf}\,bl\,B_{sr}}{Ixx^2}$$

$$wp_{37}, wp_{38} = 0$$

$$wp_{41} = -\frac{B_{sf}\,k_{sr}}{{m_s}^2} - \frac{k_{sr}\,B_{sr}}{{m_s}^2} + \frac{\rho_1\,al\,B_{sf}\,bl\,k_{sr}}{Ixx^2} - \frac{\rho_1\,bl^2 k_{sr}\,B_{sr}}{Ixx^2}$$

$$wp_{42} = -\frac{k_{sr}\,B_{sr}\,bl}{{m_s}^2} + \frac{B_{sf}\,al\,k_{sr}}{{m_s}^2} - \frac{\rho_1\,bl^3 k_{sr}\,B_{sr}}{Ixx^2} - \frac{\rho_1\,B_{sf}\,al^2\,bl\,k_{sr}}{Ixx^2}$$

$$wp_{43} = \frac{k_{sf}\,k_{sr}}{{m_s}^2} - \frac{\rho_1\,al\,k_{sf}\,bl\,k_{sr}}{Ixx^2}$$

$$wp_{44} = \frac{\rho_1\,bl^2 {k_{sr}}^2}{Ixx^2} + \frac{{k_{sr}}^2}{{m_s}^2} + \rho_6$$

$$wp_{45} = -\frac{B_{sf}\,k_{sr}}{{m_s}^2} + \frac{\rho_1\,al\,B_{sf}\,bl\,k_{sr}}{Ixx^2}$$

$$wp_{46} = -\frac{\rho_1\,bl^2 k_{sr}\,B_{sr}}{Ixx^2} - \frac{k_{sr}\,B_{sr}}{{m_s}^2}$$

$$wp_{47}, wp_{48} = 0$$

$$wp_{51} = \frac{B_{sf}\,B_{sr}}{{m_s}^2} - \frac{\rho_1\,al\,B_{sf}\,bl\,B_{sr}}{Ixx^2} + \frac{\rho_1\,al^2 {B_{sf}}^2}{Ixx^2} + \frac{{B_{sf}}^2}{{m_s}^2}$$

$$wp_{52} = \frac{B_{sf}\,B_{sr}\,bl}{{m_s}^2} - \frac{\rho_1\,{B_{sf}}^2 al^3}{Ixx^2} - \frac{{B_{sf}}^2 al}{{m_s}^2} - \frac{\rho_1\,al\,B_{sf}\,B_{sr}\,bl^2}{Ixx^2}$$

$$wp_{53} = -\frac{\rho_1\,al^2 k_{sf}\,B_{sf}}{Ixx^2} - \frac{k_{sf}\,B_{sf}}{{m_s}^2}$$

$$wp_{54} = -\frac{B_{sf}\,k_{sr}}{{m_s}^2} + \frac{\rho_1\,al\,B_{sf}\,bl\,k_{sr}}{Ixx^2}$$

$$wp_{55} = -\frac{B_{sf}\,k_{sr}}{{m_s}^2} + \frac{\rho_1\,al\,B_{sf}\,bl\,k_{sr}}{Ixx^2}$$

$$wp_{56} = \frac{B_{sf}\, B_{sr}}{{m_s}^2} - \frac{\rho_1 al\, B_{sf}\, bl\, B_{sr}}{Ixx^2}$$

$$wp_{57}, wp_{58} = 0$$

$$wp_{61} = \frac{B_{sf}\, B_{sr}}{{m_s}^2} - \frac{\rho_1 al\, B_{sf}\, bl\, B_{sr}}{Ixx^2} + \frac{\rho_1 bl^2 {B_{sr}}^2}{Ixx^2} + \frac{{B_{sr}}^2}{{m_s}^2}$$

$$wp_{62} = -\frac{B_{sf}\, al\, B_{sr}}{{m_s}^2} + \frac{\rho_1 {B_{sr}}^2 bl^3}{Ixx^2} + \frac{{B_{sr}}^2 bl}{{m_s}^2} + \frac{\rho_1 B_{sf}\, al^2 bl\, B_{sr}}{Ixx^2}$$

$$wp_{63} = -\frac{k_{sf}\, B_{sr}}{{m_s}^2} + \frac{\rho_1 al\, k_{sf}\, bl\, B_{sr}}{Ixx^2}$$

$$wp_{64} = -\frac{\rho_1 bl^2 k_{sr} B_{sr}}{Ixx^2} - \frac{k_{sr} B_{sr}}{{m_s}^2}$$

$$wp_{65} = \frac{B_{sf}\, B_{sr}}{{m_s}^2} - \frac{\rho_1 al\, B_{sf}\, bl\, B_{sr}}{Ixx^2}$$

$$wp_{66} = \frac{\rho_1 bl^2 {B_{sr}}^2}{Ixx^2} + \frac{{B_{sr}}^2}{{m_s}^2} + \rho_9$$

$$wp_{67}, wp_{68}, wp_{71}, wp_{72}, wp_{73}, wp_{74}, wp_{75}, wp_{76} = 0$$

$$wp_{77} = \rho_4$$

$$wp_{78}, wp_{81}, wp_{82}, wp_{83}, wp_{84}, wp_{85}, wp_{86}, wp_{87} = 0$$

$$wp_{88} = \rho_8$$

$$\boldsymbol{N}=\begin{bmatrix} -\frac{\rho_1 al\, bl\, B_{sr}}{Ixx^2}+\frac{\rho_1 B_{sf}\, al^2}{Ixx^2}+\frac{B_{sr}}{{m_s}^2}+\frac{B_{sf}}{{m_s}^2} & -\frac{\rho_1 al\, B_{sf}\, bl}{Ixx^2}+\frac{\rho_1 bl^2 B_{sr}}{Ixx^2}+\frac{B_{sf}}{{m_s}^2}+\frac{B_{sr}}{{m_s}^2} \\ -\frac{\rho_1 al\, B_{sr}\, bl^2}{Ixx^2}-\frac{\rho_1 B_{sf}\, al^3}{Ixx^2}-\frac{B_{sf}\, al}{{m_s}^2}+\frac{B_{sr}\, bl}{{m_s}^2} & \frac{\rho_1 B_{sf}\, al^2 bl}{Ixx^2}+\frac{\rho_1 B_{sr}\, bl^3}{Ixx^2}-\frac{B_{sf}\, al}{{m_s}^2}+\frac{B_{sr}\, bl}{{m_s}^2} \\ -\frac{\rho_1 al^2 k_{sf}}{Ixx^2}-\frac{k_{sf}}{{m_s}^2} & \frac{\rho_1 al\, k_{sf}\, bl}{Ixx^2}-\frac{k_{sf}}{{m_s}^2} \\ \frac{\rho_1 al\, bl\, k_{sr}}{Ixx^2}-\frac{k_{sr}}{{m_s}^2} & -\frac{\rho_1 bl^2 k_{sr}}{Ixx^2}-\frac{k_{sr}}{{m_s}^2} \\ \frac{\rho_1 B_{sf}\, al^2}{Ixx^2}+\frac{B_{sf}}{{m_s}^2} & -\frac{\rho_1 al\, B_{sf}\, bl}{Ixx^2}+\frac{B_{sf}}{{m_s}^2} \\ \frac{B_{sr}}{{m_s}^2}-\frac{\rho_1 al\, bl\, B_{sr}}{Ixx^2} & \frac{\rho_1 bl^2 B_{sr}}{Ixx^2}+\frac{B_{sr}}{{m_s}^2} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{B.2}$$

$$\boldsymbol{R}=\begin{bmatrix} {m_s}^{-2}+\frac{\rho_1 al^2}{Ixx^2} & -\frac{\rho_1 al\, bl}{Ixx^2}+{m_s}^{-2} \\ -\frac{\rho_1 al\, bl}{Ixx^2}+{m_s}^{-2} & {m_s}^{-2}+\frac{\rho_1 bl^2}{Ixx^2} \end{bmatrix} \tag{B.3}$$

Appendix C

Force Actuator

Force Actuator

gauge	dia (mm)	Wire Area	ohm/1000ft	ohm/meter
26	0.40386	1.28101E-07	41.02	0.134580052
28	0.32004	8.04449E-08	65.33	0.21433727
30	0.254	5.06707E-08	103.7	0.340223097
32	0.2032	3.24293E-08	162	0.531496063
36	0.127	1.26677E-08	414.8	1.360892388
40	0.07874	4.86946E-09	1079	3.540026247
10	2.58826	5.26145E-06	0.9989	0.003277231
12	2.05232	3.30811E-06	1.588	0.005209974
14	1.62814	2.08196E-06	2.525	0.008284121
16	1.29032	1.30763E-06	4.016	0.013175853
18	1.02362	8.22939E-07	6.385	0.020948163
20	0.8128	5.18868E-07	10.15	0.033300525
22	0.64516	3.26907E-07	16.14	0.052952756
24	0.51054	2.04715E-07	25.67	0.08421916

Force Actuator				
gauge	OD(mm)	ID(mm)	Mean Dia(m)	Circumference
26	0.0455	0.0375	0.0415	0.130376095
28	0.0455	0.0375	0.0415	0.130376095
30	0.0455	0.0375	0.0415	0.130376095
32	0.0455	0.0375	0.0415	0.130376095
36	0.0455	0.0375	0.0415	0.130376095
40	0.0455	0.0375	0.0415	0.130376095
10	0.0455	0.0375	0.0415	0.130376095
12	0.0455	0.0375	0.0415	0.130376095
14	0.0455	0.0375	0.0415	0.130376095
16	0.0455	0.0375	0.0415	0.130376095
18	0.0455	0.0375	0.0415	0.130376095
20	0.0455	0.0375	0.0415	0.130376095
22	0.0455	0.0375	0.0415	0.130376095
24	0.0455	0.0375	0.0415	0.130376095

Force Actuator				
gauge	Height(mm)	Area	Packing Factor	Turns
26	0.1564	0.0006256	0.6	2930.194354
28	0.1564	0.0006256	0.6	4666.05212
30	0.1564	0.0006256	0.6	7407.824346
32	0.1564	0.0006256	0.6	11574.72554
36	0.1564	0.0006256	0.6	29631.29738
40	0.1564	0.0006256	0.6	77084.54054
10	0.1564	0.0006256	0.6	71.34151173
12	0.1564	0.0006256	0.6	113.4665772
14	0.1564	0.0006256	0.6	180.2912363
16	0.1564	0.0006256	0.6	287.0537675
18	0.1564	0.0006256	0.6	456.1215417
20	0.1564	0.0006256	0.6	723.4203463
22	0.1564	0.0006256	0.6	1148.21507
24	0.1564	0.0006256	0.6	1833.574502

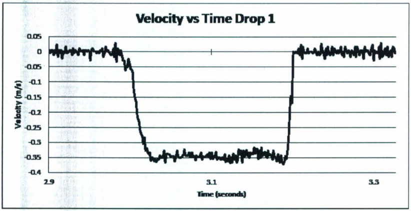
Force Actuator						
gauge	Length (m)	R (ohm)	B(T)	Current	F_t (N)	F_{active} (N)
26	382.0272978	51.41325	0.03	3	34.38245	11.167
28	608.3416551	130.3902	0.03	3	54.75074	17.783
30	965.8032116	328.5885	0.03	3	86.92228	28.233
32	1509.067518	802.0634	0.03	3	135.8160	44.114
36	3863.212846	5257.416	0.03	3	347.6891	112.93
40	10049.98139	35577.1	0.03	3	904.4983	293.79
10	9.30122772	0.030482	0.03	3	0.837110	0.2719
12	14.79332926	0.077072	0.03	3	1.331399	0.4324
14	23.50566737	0.194723	0.03	3	2.115510	0.6871
16	37.4249493	0.493105	0.03	3	3.368245	1.0940
18	59.4673455	1.245731	0.03	3	5.352061	1.7383
20	94.31671988	3.140796	0.03	3	8.48850	2.7571
22	149.6997972	7.92701	0.03	3	13.47298	4.3761
24	239.0542837	20.13295	0.03	3	21.51488	6.9882

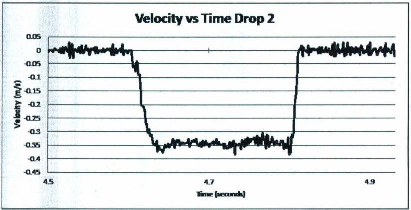
Force Actuator						
gauge	J/g	Kg/cm^3	Volume	Mass	Voltage	Power
26	0.385	8.94	4.8938E-05	437.505	154.239	462.7192
28	0.385	8.94	4.8938E-05	437.505	391.170	1173.512
30	0.385	8.94	4.8938E-05	437.505	985.765	2957.297
32	0.385	8.94	4.8938E-05	437.505	2406.19	7218.571
36	0.385	8.94	4.8938E-05	437.505	15772.2	47316.75
40	0.385	8.94	4.8938E-05	437.505	106731.5	320194.7
10	0.385	8.94	4.8938E-05	437.505	0.09144	0.274340
12	0.385	8.94	4.8938E-05	437.505	0.23121	0.693655
14	0.385	8.94	4.8938E-05	437.505	0.58417	1.752514
16	0.385	8.94	4.8938E-05	437.505	1.47931	4.437950
18	0.385	8.94	4.8938E-05	437.505	3.73719	11.21158
20	0.385	8.94	4.8938E-05	437.505	9.42238	28.26716
22	0.385	8.94	4.8938E-05	437.505	23.7810	71.34315
24	0.385	8.94	4.8938E-05	437.505	60.3988	181.1965

Force Actuator					
gauge	T_{max}	T_{room}	Delta	Joules	$time_{max}$
26	150	30	120	20212.75231	43.68253714
28	150	30	120	20212.75231	17.22414585
30	150	30	120	20212.75231	6.83487389
32	150	30	120	20212.75231	2.800104385
36	150	30	120	20212.75231	0.427179618
40	150	30	120	20212.75231	0.063126426
10	150	30	120	20212.75231	73677.62486
12	150	30	120	20212.75231	29139.45908
14	150	30	120	20212.75231	11533.57486
16	150	30	120	20212.75231	4554.523871
18	150	30	120	20212.75231	1802.845262
20	150	30	120	20212.75231	715.0611394
22	150	30	120	20212.75231	283.3173461
24	150	30	120	20212.75231	111.5515241

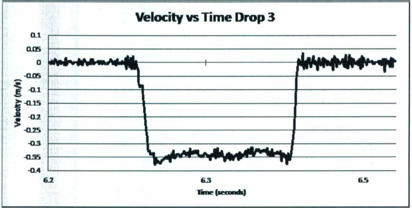
Appendix D

Drop Test

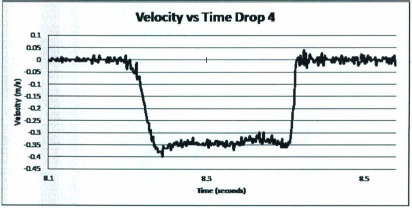




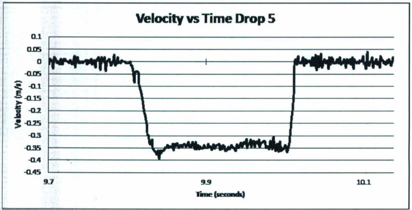
Velocity vs Time Drop 2



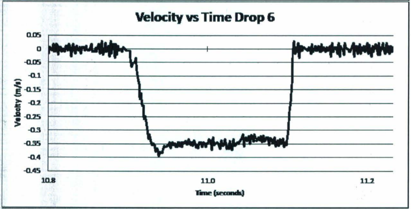
Velocity vs Time Drop 3



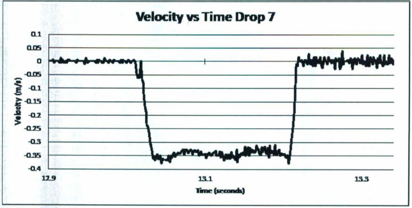
Velocity vs Time Drop 4



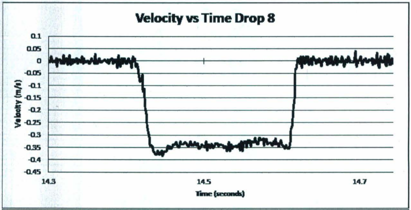
Velocity vs Time Drop 5
Velocity (m/s)
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
-0.35
-0.4
-0.45
9.7
9.9
10.1
Time (seconds)

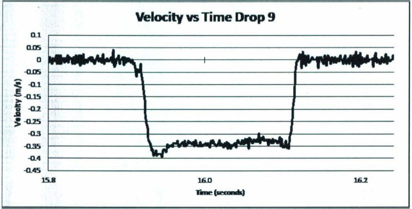


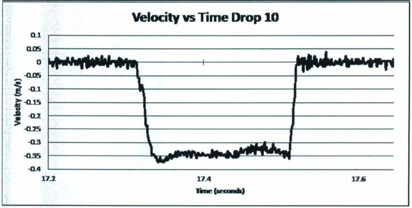
Velocity vs Time Drop 6
Velocity (m/s)
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
-0.35
-0.4
-0.45
10.8
11.0
11.2
Time (seconds)

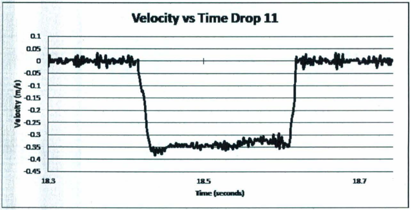


Velocity vs Time Drop 7
Velocity (m/s)
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
-0.35
-0.4
12.9
13.1
13.3
Time (seconds)


Velocity vs Time Drop 8
Velocity (m/s)
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
-0.35
-0.4
-0.45
14.3
14.5
14.7
Time (seconds)


Velocity vs Time Drop 9
Velocity (m/s)
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
-0.35
-0.4
-0.45
15.8
16.0
16.2
Time (seconds)

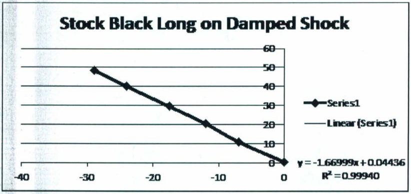

Velocity vs Time Drop 10
Velocity (m/s)
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
-0.35
-0.4
17.2
17.4
17.6
Time (seconds)


Velocity vs Time Drop 11
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
-0.35
-0.4
-0.45
Velocity (m/s)
18.3
18.5
18.7
Time (seconds)

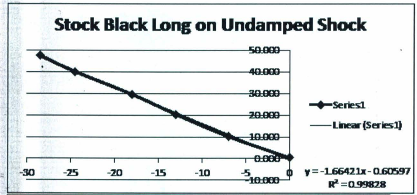
Appendix E

Spring Test

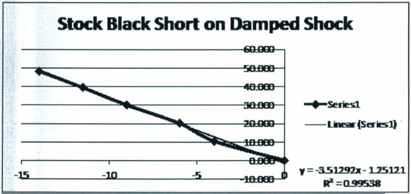
Stock Black Long on Damped Shock						
Distance (mm)	122	115	110	104.5	98	93
Scale (g)	38	1113	2087	3004	4080	4941
Delta (mm)	0	-7	-12	-17.5	-24	-29
Force (N)	0.373	10.919	20.473	29.469	40.025	48.471
Stiffness (N/mm)	-1.670		R^2	0.999		



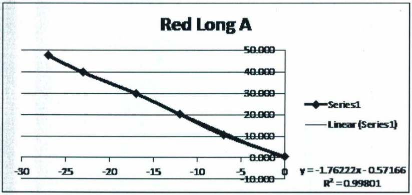
Stock Black Long on Undamped Shock						
Distance (mm)	123	116	110	105	98.5	94.5
Scale (g)	42	1035	2073	3008	4074	4835
Delta (mm)	0	-7	-13	-18	-24.5	-28.5
Force (N)	0.412	10.153	20.336	29.508	39.966	47.431
Stiffness (N/mm)	-1.664		R^2	1.000		



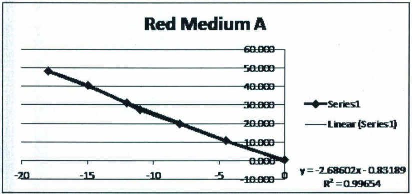
Stock Black Short on Damped Shock						
Distance (mm)	59	55	53	50	47.5	45
Scale (g)	19	1076	2056	3073	4018	4928
Delta (mm)	0	-4	-6	-9	-11.5	-14
Force (N)	0.186	10.556	20.169	30.146	39.417	48.344
Stiffness (N/mm)	-3.513		R^2	0.997		



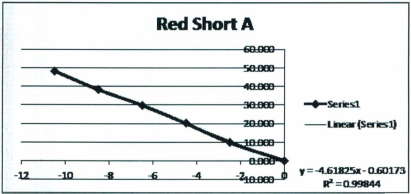
Red Long A						
Distance (mm)	58	55	52	49.5	47.5	45.5
Scale (g)	21	1035	2057	3146	4171	4857
Delta (mm)	0	-3	-6	-8.5	-10.5	-12.5
Force (N)	0.206	10.153	20.179	30.862	40.918	47.647
Stiffness (N/mm)	-3.867		R^2	0.995		



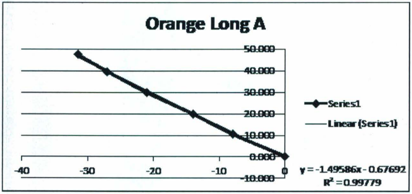
Red Medium A							
Distance (mm)	116	111.5	108	105	104	101	98
Scale (g)	41	1100	2038	2791	3151	4123	4918
Delta (mm)	0	-4.5	-8	-11	-12	-15	-18
Force (N)	0.402	10.791	19.993	27.380	30.911	40.447	48.245
Stiffness (N/mm)	-2.640		R^2	0.998			



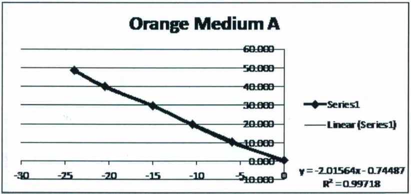
Red Short A						
Distance (mm)	49.5	47	45	43	41	39
Scale (g)	21	999	2051	3047	3896	4918
Delta (mm)	0	-2.5	-4.5	-6.5	-8.5	-10.5
Force (N)	0.206	9.800	20.120	29.891	38.220	48.246
Stiffness (N/mm)	-4.618		R^2	0.999		



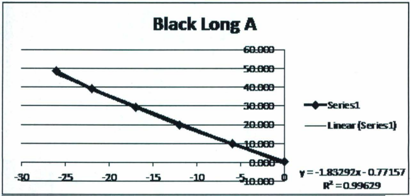
Orange Long A						
Distance (mm)	132	124	118	111	105	100.5
Scale (g)	36	1080	2029	3041	4041	4836
Delta (mm)	0	-8	-14	-21	-27	-31.5
Force (N)	0.353	10.595	19.904	29.832	39.642	47.441
Stiffness (N/mm)	-1.496		R^2	0.999		



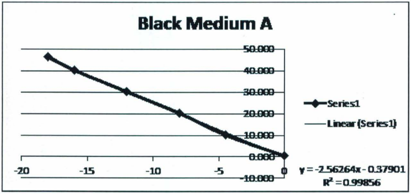
Orange Medium A						
Distance (mm)	116	110	105.5	101	95.5	92
Scale (g)	46	1047	2012	3033	4064	4958
Delta (mm)	0	-6	-10.5	-15	-20.5	-24
Force (N)	0.451	10.271	19.738	29.754	39.868	48.638
Stiffness (N/mm)	-2.016		R^2	0.999		



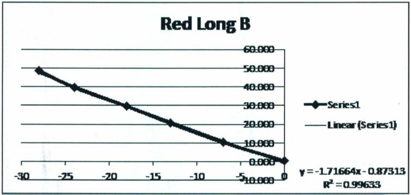
Black Long A						
Distance (mm)	134	128	122	117	112	108
Scale (g)	39	1022	2046	3004	3991	4934
Delta (mm)	0	-6	-12	-17	-22	-26
Force (N)	0.383	10.026	20.071	29.469	39.152	48.403
Stiffness (N/mm)	-1.833		R^2	0.997		



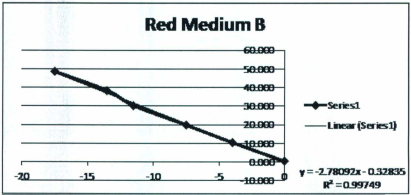
Black Medium A						
Distance (mm)	110	105.5	102	98	94	92
Scale (g)	44	1038	2058	3071	4106	4733
Delta (mm)	0	-4.5	-8	-12	-16	-18
Force (N)	0.432	10.183	20.189	30.127	40.280	46.431
Stiffness (N/mm)	-2.562		R^2	0.999		



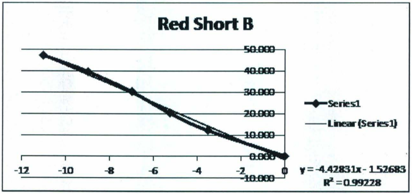
Red Long B						
Distance (mm)	132	125	119	114	108	104
Scale (g)	46	1052	2097	3021	4039	4960
Delta (mm)	0	-7	-13	-18	-24	-28
Force (N)	0.451	10.320	20.572	29.636	39.623	48.658
Stiffness (N/mm)	-1.717		R^2	0.998		



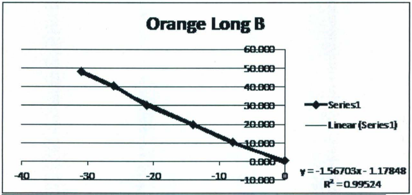
Red Medium B						
Distance (mm)	111.5	107.5	104	100	98	94
Scale (g)	42	1075	2028	3095	3908	4959
Delta (mm)	0	-4	-7.5	-11.5	-13.5	-17.5
Force (N)	0.412	10.546	19.895	30.362	38.337	48.648
Stiffness (N/mm)	-2.781		R^2	0.997		


Red Medium B

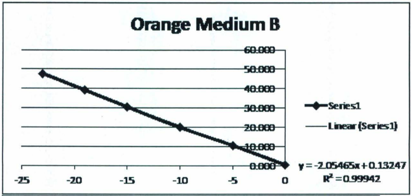
Red Short B						
Distance (mm)	45	41.5	39.75	38	36	34
Scale (g)	23	1222	2044	3072	4037	4806
Delta (mm)	0	-3.5	-5.25	-7	-9	-11
Force (N)	0.226	11.988	20.052	30.136	39.603	47.147
Stiffness (N/mm)	-4.428		R^2	0.995		


Red Short B

Orange Long B						
Distance (mm)	126	118	112	105	100	95
Scale (g)	38	1058	2026	3085	4117	4929
Delta (mm)	0	-8	-14	-21	-26	-31
Force (N)	0.373	10.379	19.875	30.264	40.388	48.353
Stiffness (N/mm)	-1.567		R^2	0.998		

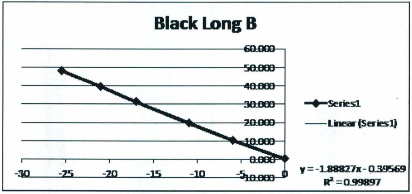


Orange Medium B						
Distance (mm)	110	105	100	95	91	87
Scale (g)	46	1084	2034	3125	4002	4870
Delta (mm)	0	-5	-10	-15	-19	-23
Force (N)	0.451	10.634	19.954	30.656	39.260	47.775
Stiffness (N/mm)	-2.055		R^2	0.999		

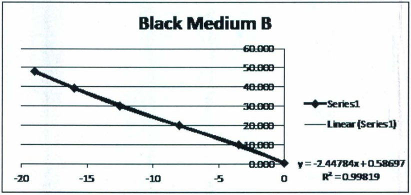


Black Long B

Distance (mm)	130	124	119	113	109	104.5
Scale (g)	40	1072	2007	3192	4032	4910
Delta (mm)	0	-6	-11	-17	-21	-25.5
Force (N)	0.392	0.516	19.689	31.314	39.554	48.167
Stiffness (N/mm)	-1.888		R^2	1.000		

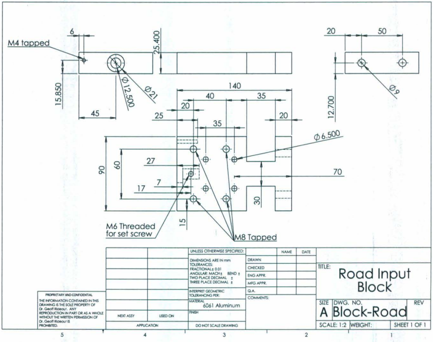


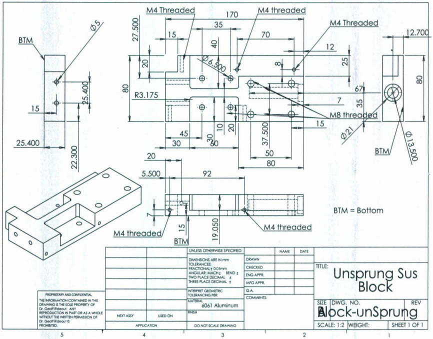
Black Medium B						
Distance (mm)	112	108.5	104	99.5	96	93
Scale (g)	42	1022	2036	3068	4020	4893
Delta (mm)	0	-3.5	-8	-12.5	-16	-19
Force (N)	0.412	10.026	19.973	30.097	39.436	48.000
Stiffness (N/mm)	-2.448		R^2	0.997		

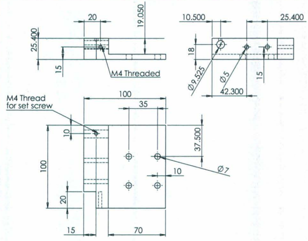


Appendix F

Quarter Car Plans




BTM
Ø5
15
25.400
22.300
25.400
M4 Threaded
170
35
15
M4 threaded
70
40
27.500
20
Ø6.500
8
M4 Threaded
12
25
12.700
80
80
R3.175
67
7
35
M8 threaded
15
Ø21
BTM
Ø13.500
45
30
30
60
10
20
37.500
50
80
20
5.500
92
7
15
19.050
M4 threaded
BTM
M4 threaded
BTM = Bottom
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN mm
TOLERANCES:
FRACTIONAL± 0.01mm
ANGULAR: MACH± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
6061 Aluminum
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
NEXT ASSY
USED ON
APPLICATION
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF Dr. Geoff Rideout. ANY REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF Dr. Geoff Rideout IS PROHIBITED.
TITLE:
Unsprung Sus Block
SIZE
B
DWG. NO.
Block-unSprung
REV
SCALE: 1:2
WEIGHT:
SHEET 1 OF 1
5
4
3
2
1



PROPRIETARY AND CONFIDENTIAL

THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF Dr. Geoff Rideout. ANY REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF Dr. Geoff Rideout IS PROHIBITED.

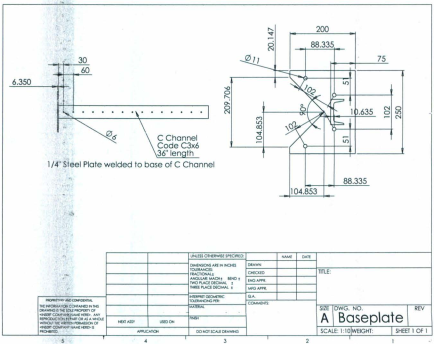
		UNLESS OTHERWISE SPECIFIED:		NAME	DATE
		DIMENSIONS ARE IN mm TOLERANCES: FRACTIONAL±0.01mm ANGULAR: MACH± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN		
			CHECKED		
			ENG APPR.		
			MFG APPR.		
		INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
		MATERIAL 6061 Aluminum	COMMENTS:		
NEXT ASSY	USED ON	FINISH			
APPLICATION		DO NOT SCALE DRAWING			

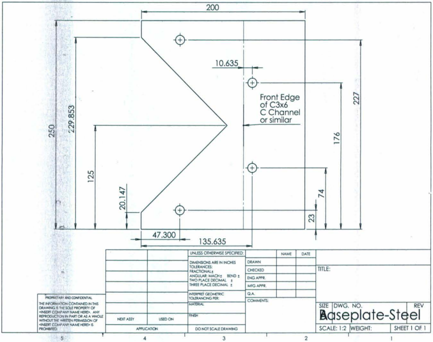
TITLE: Sprung Mass Block

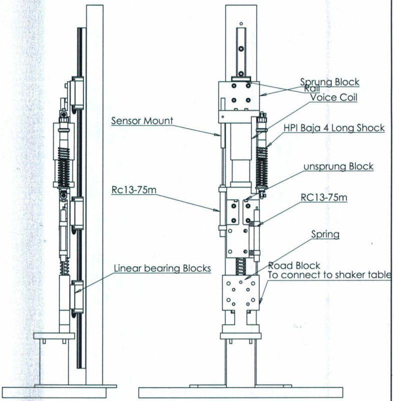
SIZE	DWG. NO.	REV
A	Block-Sprung	

SCALE: 1:2	WEIGHT:	SHEET 1 OF 1

5 4 3 2 1


30
60
6.350
Ø6
C Channel
Code C3x6
36" length
1/4" Steel Plate welded to base of C Channel
20.147
200
88.335
Ø11
75
51
102
90°
102
10.635
102
250
209.706
104.853
51
88.335
104.853
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL±
ANGULAR: MACH± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
TITLE:
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF <INSERT COMPANY NAME HERE>. ANY REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF <INSERT COMPANY NAME HERE> IS PROHIBITED.
NEXT ASSY
USED ON
APPLICATION
SIZE
A
DWG. NO.
Baseplate
REV
SCALE: 1:10
WEIGHT:
SHEET 1 OF 1
5
4
3
2
1


200
10.635
Front Edge of C3x6 C Channel or similar
250
229.853
125
20.147
227
176
74
23
47.300
135.635
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL±
ANGULAR: MACH± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
TITLE:
SIZE
A
DWG. NO.
Baseplate-Steel
REV
SCALE: 1:2
WEIGHT:
SHEET 1 OF 1
NEXT ASSY
USED ON
APPLICATION
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF <INSERT COMPANY NAME HERE>. ANY REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF <INSERT COMPANY NAME HERE> IS PROHIBITED.
5
4
3
2
1



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:

FINISH:

DEBUR AND BREAK SHARP EDGES

DO NOT SCALE DRAWING

REVISION

	NAME	SIGNATURE	DATE
DRAWN			
CHK'D			
APPV'D			
MFG			
Q.A			

MATERIAL:

WEIGHT:

TITLE:

DWG NO. Overview1

A4

SCALE:1:5

SHEET 1 OF 1

Appendix G

FPGA VHDL Code

G.1 ADCclker

```
— Company: Memorial University, Dr. Geoff Rideout
— Engineer: Keith J Wakeham
— Create Date:    15:00:26 05/14/2010
— Design Name: ADCclkr
— Module Name:    ADCclkr - Behavioral
— Project Name: Quarter Car LQR with P controller
— Target Devices: XC3SE1200
— Tool versions: Xilinx ISE 12.3
— Description: Reduces clock signals
—
— Dependencies: N/A
—
```

```
— Revision: 1.0
— Revision 0.01 - File Created
— Additional Comments:
— 34 counts on the adc and 1436 counts on the reduction clock
  gives an error of 0.008%
— Pretty good

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;


—— Uncomment the following library declaration if instantiating
—— any Xilinx primitives in this code.
—library UNISIM;
—use UNISIM.VComponents.all;


entity ADCclkr is
    Port ( clk : in  STD_LOGIC;
                                              sclk2 : out STD_LOGIC;
           sclk : out  STD_LOGIC);
end ADCclkr;


architecture Behavioral of ADCclkr is
```

```
signal slow : std_logic_vector(18 downto 0) := "
   0000000000000000000";
signal clkt : std_logic := '0';

signal count : std_logic_vector(6 downto 0) := "0000000";



begin
process (clk)
begin

If (clk = '1' and clk'event) then
        slow <= slow + 1;
        if slow = 56 then --56
                clkt <= '1';
        elsif slow = 112 then --112
                clkt <= '0';
                slow <= "0000000000000000000";
        end if;

        count <= count + 1;
end if;
sclk <= clkt;
sclk2 <= slow(1);
end process;
end Behavioral;
```

G.2 ADCreader

```
— Company: Memorial University, Dr. Geoff Rideout
— Engineer: Keith J Wakeham
—
— Create Date:    13:14:48 04/25/2011
— Design Name: ADCread
— Module Name:    ADCread - Behavioral
— Project Name: Quarter Car LQR with P controller
— Target Devices: XC3SE1200
— Tool versions: Xilinx ISE 12.3
— Description: Reads 4 ADC signals from MCP3201 or similar 12
  bit ADC
—
— Dependencies: N/A
—
— Revision: 1.0
— Revision 0.01 - File Created
— Additional Comments:
—
```

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

---- Uncomment the following library declaration if instantiating
---- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity ADCread is
    Port ( clk : in  STD_LOGIC;
           datain : in  STD_LOGIC;
           datain2 : in  STD_LOGIC;
                          datain3 : in  STD_LOGIC;
                          datain4 : in  STD_LOGIC;
           enable : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           cs : out  STD_LOGIC;
                          cs2: out STD_LOGIC;
                          cs3: out STD_LOGIC;
                          cs4: out STD_LOGIC;
           clkout : out  STD_LOGIC;
           slowclock : out  STD_LOGIC;
```

```
                    dataheld4a: out STD_LOGIC_VECTOR (15
                       downto 0);
                    dataheld4: out STD_LOGIC_VECTOR (11
                       downto 0);
                    dataheld3: out STD_LOGIC_VECTOR (15
                       downto 0);
                    dataheld3a: out STD_LOGIC_VECTOR (11
                       downto 0);
                    dataheld2: out STD_LOGIC_VECTOR (11
                       downto 0);
          dataheld  : out  STD_LOGIC_VECTOR (11 downto 0));
end ADCread;


architecture Behavioral of ADCread is
signal count : std_logic_vector(5 downto 0) := "000000";
signal temp : std_logic_vector(11 downto 0) := "000000000000";
signal temp2 : std_logic_vector(11 downto 0) := "000000000000";
signal temp3 : std_logic_vector(11 downto 0) := "000000000000";
signal temp4 : std_logic_vector(11 downto 0) := "000000000000";
signal output: std_logic_vector(11 downto 0) := "000000000000";
signal output2: std_logic_vector(11 downto 0) := "000000000000";
signal output3: std_logic_vector(15 downto 0) := "
   0000000000000000";
signal output4: std_logic_vector(11 downto 0) := "000000000000";
signal pwmf : std_logic := '0';
signal slow : std_logic := '0';
signal csstate : std_logic := '0';


begin
process(clk)
begin
if (clk = '0' and clk'event) then
        if reset = '0' then
                if enable = '1' then
                        count <= count + 1;
                        if count = 0 then
                                csstate <= '0';
                        elsif count = 4 then
                                temp(11) <= datain;
                                temp2(11) <= datain2;
                                temp3(11) <= datain3;
                                temp4(11) <= datain4;
                        elsif count = 5 then
                                temp(10) <= datain;
                                temp2(10) <= datain2;
                                temp3(10) <= datain3;
                                temp4(10) <= datain4;
                        elsif count = 6 then
                                temp(9) <= datain;
```

```
    temp2(9) <= datain2;
    temp3(9) <= datain3;
    temp4(9) <= datain4;
elsif count = 7 then
    temp(8) <= datain;
    temp2(8) <= datain2;
    temp3(8) <= datain3;
    temp4(8) <= datain4;
elsif count = 8 then
    temp(7) <= datain;
    temp2(7) <= datain2;
    temp3(7) <= datain3;
    temp4(7) <= datain4;
elsif count = 9 then
    temp(6) <= datain;
    temp2(6) <= datain2;
    temp3(6) <= datain3;
    temp4(6) <= datain4;
elsif count = 10 then
    temp(5) <= datain;
    temp2(5) <= datain2;
    temp3(5) <= datain3;
    temp4(5) <= datain4;
elsif count = 11 then
    temp(4) <= datain;
```

```
        temp2(4) <= datain2;
        temp3(4) <= datain3;
        temp4(4) <= datain4;
elsif count = 12 then
        temp(3) <= datain;
        temp2(3) <= datain2;
        temp3(3) <= datain3;
        temp4(3) <= datain4;
        slow <= not slow;
elsif count = 13 then
        temp(2) <= datain;
        temp2(2) <= datain2;
        temp3(2) <= datain3;
        temp4(2) <= datain4;
elsif count = 14 then
        temp(1) <= datain;
        temp2(1) <= datain2;
        temp3(1) <= datain3;
        temp4(1) <= datain4;
elsif count = 15 then
        temp(0) <= datain;
        temp2(0) <= datain2;
        temp3(0) <= datain3;
        temp4(0) <= datain4;
elsif count = 16 then
```

```
                csstate <= '1';

            elsif count = 26 then
                output <= temp;
                slow <= not slow;
                count <= "000000";
                output2 <= temp2;
                output3(15 downto 4) <= temp3;
                output4 <= temp4;

            end if;
        end if;
    elsif reset = '1' then
    temp <= "000000000000";
    output <= "000000000000";
    output2 <= "000000000000";
    output3 <= "0000000000000000";
    count <= "000000";
    end if;
end if;
end process;
cs <= csstate;
cs2 <= csstate;
cs3 <=csstate;
cs4 <=csstate;
```

```
dataheld <=output;
clkout <= clk;
slowclock <= slow;
dataheld2 <= output2;
—dataheld3 <= '0' & output & "0000";
—dataheld3 <= '1' & not(output) & "1111";
dataheld3 <= output3;
dataheld3a <= output3(15 downto 4);
dataheld4 <= output4;
dataheld4a <= output4 & "0000";
end Behavioral;
```

G.3 RunAverage2

```
————————————————————————————————————————
— Company: Memorial University, Dr. Geoff Rideout
— Engineer: Keith J Wakeham
—
— Create Date:    13:14:48 04/25/2011
— Design Name: RunAverage2
— Module Name:    RunAverage - Behavioral
— Project Name: Quarter Car LQR with P controller
— Target Devices: XC3SE1200
— Tool versions: Xilinx ISE 12.3
— Description: Averages 16 samples to reduce noise, provides
  differentiated signal
—
— Dependencies: N/A
—
— Revision: 1.0
— Revision 0.01 - File Created
— Additional Comments:
—
____________________________________________

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

—— Uncomment the following library declaration if instantiating
—— any Xilinx primitives in this code.
—library UNISIM;
—use UNISIM.VComponents.all;

entity runaverage2 is
    Port ( clk : in STD_LOGIC;
                              slowclk : out STD_LOGIC;
           din : in STD_LOGIC_VECTOR (11 downto 0);
```

```
                            din2: in  STD_LOGIC_VECTOR (11 downto
                               0);
                            din3: in STD_LOGIC_VECTOR (11 downto 0)
                               ;
                            din4  : in STD_LOGIC_VECTOR (11 downto
                               0);


            dout  : out  STD_LOGIC_VECTOR (11 downto 0);
                            dout2  : out STD_LOGIC_VECTOR (11 downto
                                0);
                            dout3  : out STD_LOGIC_VECTOR (11 downto
                                0);
                            dout4  : out STD_LOGIC_VECTOR (11 downto
                                0);


                            testout  : out STD_LOGIC_VECTOR(15
                               downto 0);
            dotout  : out signed (16 downto 0);
                            dotout2  : out signed  (16 downto 0);
                            dotout4  : out signed  (16 downto 0));
end  runaverage2;

architecture Behavioral of  runaverage2 is
signal count  : std_logic_vector(4 downto 0)  := "00000";
signal set  : std_logic:=  '0';
signal position : std_logic_vector(15 downto 0):= "
    0000000000000000";
signal position2 : std_logic_vector(15 downto 0):= "
    0000000000000000";

signal position4 : std_logic_vector(15 downto 0):= "
    0000000000000000";
signal position4a : std_logic_vector(15 downto 0):= "
    0000000000000000";



signal current : std_logic_vector(15 downto 0):= "
    0000000000000000";
signal currenta : std_logic_vector(15 downto 0):= "
    0000000000000000";

signal pout : std_logic_vector(11 downto 0) := "000000000000";
signal vout : signed(16 downto 0) := "00000000000000000";
signal cout : std_logic_vector(11 downto 0) := "000000000000";

signal positiona : std_logic_vector(15 downto 0):= "
    0000000000000000";
signal position2a : std_logic_vector(15 downto 0):= "
    0000000000000000";
```

```
signal pouta : std_logic_vector(11 downto 0) := "000000000000";
signal vouta : signed(16 downto 0) := "00000000000000000";

signal poutc : std_logic_vector(11 downto 0) := "000000000000";
signal voutc : signed(16 downto 0) := "00000000000000000";



begin


process(clk)
begin


if (clk = '1' and clk'event) then
        count <= count + 1;
        if set = '0' then
                if count < 16 then
                        position <= position + din;


                        positiona <= positiona + din2;


                        current <= current + din3;


                        position4 <= position4 + din4;
        elsif count = 16 then
                pout <= position(15 downto 4);
                vout <= signed('0' & position) - signed
                   ('0' & position2);
                --vout <= signed(position) - signed(
                   position2);

                pouta <= positiona(15 downto 4);
                vouta <= signed('0' & positiona) - signed
                   ('0' & position2a);

                cout <= current(15 downto 4);

                poutc <= position4(15 downto 4);
                voutc <= signed('0' & position4) - signed
                   ('0' & position4a);
        elsif count = 17 then
                position2 <= "0000000000000000";
                position2a <= "0000000000000000";
                currenta <= "0000000000000000";
                position4a <= "0000000000000000";
        elsif count = 18 then
                set <= '1';
        end if;
elsif set = '1' then
```

```
if count < 16 then
        position2 <= position2 + din;
        position2a <= position2a + din2;
        currenta <= currenta + din3;
        position4a <= position4a + din4;

elsif count = 16 then
        pout <= position2(15 downto 4);
        vout <= signed('0' & position2) - signed
           ('0' & position);
        --vout <= signed(position2) - signed(
           position);

        pouta <= position2a(15 downto 4);
        vouta <= signed('0' & position2a) -
           signed('0' & positiona);

        cout <= currenta(15 downto 4);

        poutc <= position4a(15 downto 4);
        voutc <= signed('0' & position4a) -
           signed('0' & position4);

elsif count = 17 then
        position <= "0000000000000000";
```

```
            positiona <= "0000000000000000";
            current <= "0000000000000000";
            position4 <= "0000000000000000";
        elsif count = 18 then
            set <= '0';
        end if;
    end if;


end if;

end process;

slowclk <= count(4);

dout <= pout;
dotout <= vout;

dout2 <= pouta;
dotout2 <= vouta;

dout3 <= cout;

dout4 <= poutc;
```

```
dotout4 <= voutc;


testout <= pout & "0000"; —STD_LOGIC_VECTOR(pout(15 downto 0))
   ;--poutc & "0000";
end Behavioral;
```

G.4 Fixed Point

```
————————————————————————————————————

— Company: Memorial University, Dr. Geoff Rideout
— Engineer: Keith J Wakeham
—
— Create Date:    13:52:25 07/16/2011
— Design Name: FixedP
— Module Name:    FixedP - Behavioral
— Project Name: Quarter Car LQR with P controller
— Target Devices: XC3SE1200
— Tool versions: Xilinx ISE 12.3
— Description: Fixed Point mathematical Controller
—
— Dependencies: N/A
—
— Revision: 1.0
— Revision 0.01 - File Created
— Additional Comments:
```

```
—
________________________________________________

library IEEE;

use IEEE.STD_LOGIC_1164.ALL;

use IEEE.STD_LOGIC_ARITH.ALL;

use IEEE.STD_LOGIC_UNSIGNED.ALL;

library ieee_proposed;

use IEEE_proposed.fixed_float_types.all;

use IEEE_proposed.fixed_pkg.ALL;


— Uncomment the following library declaration if using

— arithmetic functions with Signed or Unsigned values

—use IEEE.NUMERIC_STD.ALL;


— Uncomment the following library declaration if instantiating

— any Xilinx primitives in this code.

—library UNISIM;

—use UNISIM.VComponents.all;


entity FixedP is

    Port (

                                clk : in STD_LOGIC;

                                zero : in STD_LOGIC;

                                sw0 : in STD_LOGIC;

                                sw1 : in STD_LOGIC;
```

```
                                                  sw2 : in STD_LOGIC;
                                                  sw3 : in STD_LOGIC;
                                                  Din : in  STD_LOGIC_VECTOR (11
                                                     downto 0);
                                                  Din2 : in  STD_LOGIC_VECTOR (11
                                                     downto 0);
                                                  Vin : in signed (16 downto 0);
                                                  Vin2 : in signed (16 downto 0);
                                                  Vin4 : in signed (16 downto 0);
                                                  Cin : in STD_LOGIC_VECTOR (11
                                                     downto 0);
                                                  Driveout : out STD_LOGIC_VECTOR
                                                     (11 downto 0);
                                                  LED : out  STD_LOGIC_VECTOR (7
                                                     downto 0)

                                   );
end FixedP;

architecture Behavioral of FixedP is
--Postion 1
signal zeroedpos : signed (12 downto 0) := "0000000000000";
signal zeroheldpos : std_logic_vector(11 downto 0) := "
   000000000000";
```

```
signal scale : ufixed (−1 downto −16) := "0000010010110000"; —"
    about 0.018310546 mm per bit" http://www.bytecraft.com/
    Fixed_Point_Converter
signal position : sfixed (13 downto −16) := to_sfixed(0,13,16);
    —Position storage in mm from zero
signal gain1: sfixed (5 downto −2) := to_sfixed(0, 5, −2); —
    GAIN 3, (0.003, 5, −2) Ridequality, (0.192, 5, −2) roadholding
    , (−0.565, 5, −2) extreme tire
signal force : sfixed (19 downto −18) := to_sfixed(0,18,18); —
    Position storage in mm from zero
signal forcetest : std_logic_vector (7 downto 0) := "00000000";


—Postion 2
signal zeroedpos2 : signed (12 downto 0) := "0000000000000";
signal zeroheldpos2 : std_logic_vector(11 downto 0) := "
    000000000000";
signal scale2 : ufixed (−1 downto −16) := "0000010010110000"; —"
    about 0.018310546 mm per bit" http://www.bytecraft.com/
    Fixed_Point_Converter
signal position2 : sfixed (13 downto −16) := to_sfixed(0,13,16);
    —Position storage in mm from zero−0.535
signal gain2: sfixed (5 downto −2) := to_sfixed(0, 5, −2); —
    GAIN 1, (−1.002, 5, −2) Ridequality, (−0.524, 5, −2)
    roadholding, (−1.001, 5, −2) extreme tire
```

```
signal forcea : sfixed (19 downto -18) := to_sfixed(0,18,18); --
    Position storage in mm from zero
signal forcetesta : std_logic_vector (7 downto 0) := "00000000";


--Velocity 1
signal vscale : sfixed (0 downto -8) := to_sfixed(0.5859375
    ,0,-8); -- 0.018310546 * 512 samples/ sec /16 for added bits
    of accuracy for oversampling
signal vresult : sfixed (17 downto -8) :=to_sfixed(0,17,-8);
signal vgain: sfixed (5 downto -6) := to_sfixed(0.5, 5, -6); --
    GAIN 4 (0.0561, 5, -6) Ridequality, (0.044, 5, -6) roadholding
    , (-0.335, 5, -6) extreme tire
signal force2: sfixed (24 downto -14) :=to_sfixed(0,24,-14);
signal force2trunc: sfixed (19 downto -18) :=to_sfixed(0,19,-18);

--Velocity 2
signal vscale2: sfixed (0 downto -8) := to_sfixed(0.5859375
    ,0,-8); -- 0.018310546 * 512 samples/ sec /16 for added bits
    of accuracy for oversampling
signal vresult2 : sfixed (17 downto -8) :=to_sfixed(0,17,-8);
signal vgain2: sfixed (5 downto -6) := to_sfixed(0, 5, -6); --
    GAIN 2 (-0.0503, 5, -6) Ridequality, (0.0266, 5, -6)
    roadholding, (-0.050, 5, -6) extreme tire
signal force2a: sfixed (25 downto -14) :=to_sfixed(0, 25,-14);
signal force2trunca: sfixed (19 downto -18) :=to_sfixed(0,19,-18)
    ;


--Velocity 4
signal vscale4: sfixed (0 downto -8) := to_sfixed(0.79568 ,0,-8);
    -- left as old 0.024902 * 512 / 16
signal vroad : sfixed (17 downto -8) :=to_sfixed(0,17,-8);


signal vunsprung: sfixed (18 downto -8) :=to_sfixed(0,19,-8);
signal vsprung: sfixed (19 downto -8) :=to_sfixed(0,19,-8);



--Current
signal cscale : sfixed (0 downto -16) := to_sfixed(0.063171561
    ,0,-16); -- 0.00625461 AMP / Bit * 10.1 N/A = 0.063171561
signal zeroedcurrent : signed (12 downto 0) := "0000000000000";
signal realforce : sfixed (13 downto -16) := to_sfixed(0,13,16);



--Total Force P controller
signal forcetarget: sfixed (20 downto -18) :=to_sfixed(0,20,-18);
signal czeroheldcurrent : std_logic_vector(11 downto 0) := "
    000000000000";
```

```
signal forcetargettrunc: sfixed (13 downto -16) :=to_sfixed
   (0,13,-16);
signal error: sfixed (14 downto -16) :=to_sfixed(0,14,-16);
signal pgain: sfixed (8 downto -1) :=to_sfixed(-50,8,-1);
signal Driveheld : sfixed (23 downto -17) := to_sfixed(0,23,-17);
signal Driveheldtrunc : signed (11 downto 0) := "000000000000";



signal switches : std_logic_vector (3 downto 0);
signal seg : std_logic_vector (7 downto 0);



begin
process(clk)


begin


If (clk = '1' and clk'event) then


--Basic zeroing
        if zero = '1' then
                zeroheldpos <= Din;
                zeroheldpos2 <= Din2;
                czeroheldcurrent <= Cin;
        end if;
        zeroedpos <= signed('0' & zeroheldpos) - signed('0' & Din
          );
        zeroedpos2 <= signed('0' & zeroheldpos2) - signed('0' &
          Din2);


        zeroedcurrent <= signed('0' & czeroheldcurrent) - signed
          ('0' & Cin);


--LED output selection
                 case switches is
                when "0000" => seg <= to_SLV(position(11 downto
                    4)); -- 0
                when "0001" => seg <= to_SLV(position2(11 downto
                    4)); -- 1
                when "0010" => seg <= to_SLV(force(11 downto 4));
                     -- 2
                when "0011" => seg <= to_SLV(forcea(11 downto 4))
                    ; -- 3forcetarget
                when "0100" => seg <= to_SLV(forcetarget(11
                    downto 4));
                when "0101" => seg <= to_SLV(error(11 downto 4));
                when "0110" => seg <= STD_LOGIC_VECTOR(
                    Driveheldtrunc(11 downto 4));--to_SLV(
                    realforce(13 downto 5)); -- actual force
        when "0111" => seg <= to_SLV(Vsprung(7 downto 0))
           - to_SLV(Vunsprung(7 downto 0)) + 16;--to_SLV
           (realforce(7 downto 0)); -- actual force
        when "1000" => seg <= Din(7 downto 0);
        when "1001" => seg <= Din2(7 downto 0);
        when "1010" => seg <= Cin(7 downto 0);
        when "1011" => seg <= STD_LOGIC_VECTOR(Vin(7
           downto 0)); --to_slv(force2trunc(19 downto 12)
           ); -- b
        when "1100" => seg <= STD_LOGIC_VECTOR(Vin2(7
           downto 0)); -- c
        when "1101" => seg <= STD_LOGIC_VECTOR(Vin4(7
           downto 0)); -- d
        when "1110" => seg <= to_SLV(error(7 downto 0));
           -- E
        when others => seg <= "00000000"; -- F
         end case;


--Saturation Code
if force(19) = '0' and to_SLV(force(18 downto 7)) > 0
   then
        forcetest <= "01111111";
elsif force(19) = '1' and to_SLV(not force(18 downto 7))
   > 0 then
        forcetest <= "10000000";
```

```
else
        forcetest <= to_SLV(force(7 downto 0));
end if;


if force2(24) = '0' and to_SLV(force2(24 downto 19)) > 0
   then
        force2trunc(19 downto -18) <= "
           01111111111111111111111111111111111111";
elsif force2(24) = '1' and to_SLV(not force2(24 downto
   19)) > 0 then
        force2trunc(19 downto -18) <= "
           10000000000000000000000000000000000000";
else
        force2trunc(19 downto -18) <= force2(19 downto
           -14) & "0000";
end if;


if force2a(25) = '0' and to_SLV(force2a(25 downto 19)) >
   0 then
        force2trunca(19 downto -18) <= "
           01111111111111111111111111111111111111";
elsif force2a(25) = '1' and to_SLV(not force2a(25 downto
   19)) > 0 then
        force2trunca(19 downto -18) <= "
           10000000000000000000000000000000000000";
```

```
else
        force2trunca(19 downto -18) <= force2a(19 downto
           -14) & "0000";
end if;



if forcetarget(20) = '0' and to_SLV(forcetarget(19 downto
    13)) > 0 then
        forcetargettrunc(13 downto -16) <= "
           011111111111111111111111111111";
elsif forcetarget(20) = '1' and to_SLV(not forcetarget(19
    downto 13)) > 0 then
        forcetargettrunc(13 downto -16) <= "
           100000000000000000000000000000";
else
        forcetargettrunc(13 downto -16) <= forcetarget(13
            downto -16);
end if;


if Driveheld(23) = '0' and to_SLV(Driveheld(23 downto 11)
   ) > 0 then
        Driveheldtrunc(11 downto 0) <= "011111111111";
elsif Driveheld(23) = '1' and to_SLV(not Driveheld(23
   downto 11)) > 0 then
        Driveheldtrunc(11 downto 0) <= "100000000000";
        else
                Driveheldtrunc(11 downto 0) <= signed(to_SLV(
                    Driveheld(11 downto 0)));
        end if;



end if;

end process;

—Position 1
        position <= sfixed(zeroedpos) * sfixed('0' & scale);
        force <= position * gain1;

—Position 2
        position2 <= sfixed(zeroedpos2) * sfixed('0' & scale2);
        forcea <= position2 * gain2;

—Velocity 1 - relative velocity
        vresult <= sfixed(Vin) * vscale;
        force2 <= vunsprung * vgain;

—Velocity 2 - relative velocity
        vresult2 <= sfixed(Vin2) * vscale2;
        force2a <= vsprung * vgain2;


--Velocity 4 - ground velocity
        vroad <= sfixed(Vin4) * vscale4;


--All the absolute velocities
        vunsprung <= vroad + vresult;
        vsprung <=  vroad +  vresult +  vresult2;




--Current to real force
        realforce <= sfixed(zeroedcurrent) * cscale;
--Force
forcetarget <= force + force2trunc + forcea + force2trunca;


error <= realforce - forcetargettrunc;
Driveheld <= error * pgain;




switches(0) <= sw3;
switches(1) <= sw2;
switches(2) <= sw1;
switches(3) <= sw0;
```

```
LED <= seg;


Driveout <= STD_LOGIC_VECTOR(Driveheldtrunc);


end Behavioral;
```

G.5 PWM

```
— Company: Memorial University, Dr. Geoff Rideout

— Engineer: Keith J Wakeham

—

— Create Date:    13:15:26 11/30/2009

— Design Name: PWM

— Module Name:    PWM - Behavioral

— Project Name: Quarter Car LQR with P controller

— Target Devices: XC3SE1200

— Tool versions: Xilinx ISE 12.3

— Description: 11 bit Pulse Width Modulator for LMD18200

—

— Dependencies: N/A

—

— Revision: 1.0
— Revision 0.01 - File Created
— Additional Comments:
—
————————————————————————————————————————————

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

—— Uncomment the following library declaration if instantiating
—— any Xilinx primitives in this code.
—library UNISIM;
—use UNISIM.VComponents.all;

entity PWM is
    Port ( CLOCK : in  STD_LOGIC;
                       ENABLE : in  STD_LOGIC;
                       RST : in STD_LOGIC;
           SPEED : in  STD_LOGIC_VECTOR (11 downto 0);
                       PWMout : out STD_LOGIC;
                       —In1 : out STD_LOGIC;
                       —In2 : out STD_LOGIC;
                       SB : out STD_LOGIC;
                       —SB2: out STD_LOGIC;
           Dir : out  STD_LOGIC);
```

```
end PWM;

architecture Behavioral of PWM is

signal count : std_logic_vector(10 downto 0) := "00000000000";
signal pwmh : std_logic := '0';
signal dirstore: std_logic :='0';
—signal M2store: std_logic :='0';
signal SBstore:  std_logic :='0';
signal Direction: std_logic :=  '0';
signal speedheld : std_logic_vector(10 downto 0) := "00000000000"
    ;


begin

process(CLOCK)

begin

        if CLOCK = '1' and CLOCK'event then
                if RST = '0' then
                        if ENABLE = '1' then
                                SBstore <= '0';
                                count <= count + 1;
        if count = 0 then
                Direction <= speed(11);
                speedheld <= Speed(10
                   downto 0);
        end if;
        if Direction = '1' then
                Dirstore <= '1';
                if count > not(speedheld
                   (10 downto 0)) then
                        pwmh <= '0';
                else
                        pwmh <= '1';
                end if;
        elsif Direction = '0' then
                DirStore <= '0';
                if count > speedheld(10
                   downto 0) then
                        pwmh<= '0';
                else
                        pwmh <= '1';
                end if;
        end if;
else
pwmh <= '0';
```

```
                        SBstore <= '1';
                        DirStore <= '1';
                        count <= "00000000000";
                        end if;
        elsif RST = '1' then
                        pwmh <= '0';
                        SBstore <= '0';
                        DirStore <= '1';
                        count <= "00000000000";

        end if;
end if;
end process;


PWMout <= pwmh;
SB <= SBstore;
--SB2 <= Not(SBstore);
Dir <= DirStore;
--In1 <= DirStore;
--In2 <= Not(DirStore);



end Behavioral;
```

G.6 Hex Display

```
        -- HEX_DISPLAY                                        Chuck
           McManis 17-Feb-2001
--
-- This is a driver for a 7 segment LED display. It converts a
   4-bit nybble
-- into a hexadecimal character 0-9a-f. Some letters are upper
   case others are
-- lower case in an effort to distinguish them from numbers so b
    and 6 differ
-- by the presence of the top segment being lit or not. (for
   example)
--

--Modified by Keith Wakeham to correct the output for the wiring
   sequence on
--Digilent Spartan 3 development boards
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity hex_display is
    Port ( value : in std_logic_vector(15 downto 0);
                clk : in std_logic;
            blank : in std_logic;
```

```
          test : in std_logic;
               an : out std_logic;
               an2 : out std_logic;
               an3 : out std_logic;
               an4 : out std_logic;
               segs : out std_logic_vector(7 downto 0));
end hex_display;

— This is a good third project since it is simply combinatorial
   logic. When
— synthesized the selection statement (case) generates a decoder
    that takes
— four input lines and generates eight output lines. (the
   decimal point is
— always set to 'off.' If you want to get decimal point control,
    try adding
— another pin (dp) to the port description, and then you can
   assign it with
— a concurrent signal assignment
architecture behavioral of hex_display is

signal count : std_logic_vector(1 downto 0) := "00";
signal seg : std_logic_vector(7 downto 0) := "00000000";
```

```
begin
    process (value, blank, test, clk) is
    begin
         if clk = '1' and clk'event then
                count <= count + 1;
                 if count = 0 then
                               an4 <= '0';
                               an <= '1';
                               if (blank = '1') then
                                         seg <= "00000000";
                               elsif (test = '1') then
                                         seg <= "11111111";
                               else
                                         case value(15 downto 12)
                                             is
                                        when "0000" => seg <= "
                                           00000011"; — 0
                                        when "0001" => seg <= "
                                           10011111"; — 1
                                        when "0010" => seg <= "
                                           00100101"; — 2
                                        when "0011" => seg <= "
                                           00001101"; — 3
                                        when "0100" => seg <= "
                                           10011001"; — 4
```

```
          when "0101" => seg <= "
             01001001"; — 5
          when "0110" => seg <= "
             01000001"; — 6
          when "0111" => seg <= "
             00011111"; — 7
          when "1000" => seg <= "
             00000001"; — 8
          when "1001" => seg <= "
             00001001"; — 9
          when "1010" => seg <= "
             00010001"; — A
          when "1011" => seg <= "
             11000001"; — b
          when "1100" => seg <= "
             01100011"; — c
          when "1101" => seg <= "
             10000101"; — d
          when "1110" => seg <= "
             01100001"; — E
          when others => seg <= "
             01110001"; — F
           end case;
end if; —end case
```

```
elsif count = 1 then
        an4 <= '1';
        an3 <= '0';
        if (blank = '1') then
                seg <= "00000000";
        elsif (test = '1') then
                seg <= "11111111";
        else
                case value(11 downto 8)
                   is
                when "0000" => seg <= "
                   00000011"; -- 0
                when "0001" => seg <= "
                   10011111"; -- 1
                when "0010" => seg <= "
                   00100101"; -- 2
                when "0011" => seg <= "
                   00001101"; -- 3
                when "0100" => seg <= "
                   10011001"; -- 4
                when "0101" => seg <= "
                   01001001"; -- 5
                when "0110" => seg <= "
                   01000001"; -- 6
                when "0111" => seg <= "
                    00011111"; — 7
                when "1000" => seg <= "
                    00000001"; — 8
                when "1001" => seg <= "
                    00001001"; — 9
                when "1010" => seg <= "
                    00010001"; — A
                when "1011" => seg <= "
                    11000001"; — b
                when "1100" => seg <= "
                    01100011"; — c
                when "1101" => seg <= "
                    10000101"; — d
                when "1110" => seg <= "
                    01100001"; — E
                when others => seg <= "
                    01110001"; — F
                 end case;
        end if; —end case
elsif count = 2 then
        an3 <= '1';
        an2 <= '0';
        if (blank = '1') then
                 seg <= "00000000";
```

```
elsif (test = '1') then
        seg <= "11111111";
else
        case value(7 downto 4)
           is
       when "0000" => seg <= "
          00000011"; — 0
       when "0001" => seg <= "
          10011111"; — 1
       when "0010" => seg <= "
          00100101"; — 2
       when "0011" => seg <= "
          00001101"; — 3
       when "0100" => seg <= "
          10011001"; — 4
       when "0101" => seg <= "
          01001001"; — 5
       when "0110" => seg <= "
          01000001"; — 6
       when "0111" => seg <= "
          00011111"; — 7
       when "1000" => seg <= "
          00000001"; — 8
       when "1001" => seg <= "
          00001001"; — 9
```

```
                when "1010" => seg <= "
                    00010001"; — A
                when "1011" => seg <= "
                    11000001"; — b
                when "1100" => seg <= "
                    01100011"; — c
                when "1101" => seg <= "
                    10000101"; — d
                when "1110" => seg <= "
                    01100001"; — E
                when others => seg <= "
                    01110001"; — F
                 end case;
        end if; —end case
elsif count = 3 then
        an2 <= '1';
        an <= '0';
        if (blank = '1') then
                  seg <= "00000000";
        elsif (test = '1') then
                  seg <= "11111111";
        else
                  case value(3 downto 0)
                     is
```

```
when "0000" => seg <= "
   00000011"; — 0
when "0001" => seg <= "
   10011111"; — 1
when "0010" => seg <= "
   00100101"; — 2
when "0011" => seg <= "
   00001101"; — 3
when "0100" => seg <= "
   10011001"; — 4
when "0101" => seg <= "
   01001001"; — 5
when "0110" => seg <= "
   01000001"; — 6
when "0111" => seg <= "
   00011111"; — 7
when "1000" => seg <= "
   00000001"; — 8
when "1001" => seg <= "
   00001001"; — 9
when "1010" => seg <= "
   00010001"; — A
when "1011" => seg <= "
   11000001"; — b
```

```
                                        when "1100" => seg <= "
                                           01100011"; — c
                                        when "1101" => seg <= "
                                           10000101"; — d
                                        when "1110" => seg <= "
                                           01100001"; — E
                                        when others => seg <= "
                                           01110001"; — F
                                         end case;
                                  end if; —end case


                     end if; — end count
               end if; —end clk
            end process;
                segs <= seg;
end behavioral;
```

G.7 Logger2 - Arduino

```
/*
  Analog input, analog output, serial output


 Reads an analog input pin, maps the result to a range from 0 to
   255
 and uses the result to set the pulsewidth modulation (PWM) of an
    output pin.
 Also prints the results to the serial monitor.

 The circuit:
 * potentiometer connected to analog pin 0.
   Center pin of the potentiometer goes to the analog pin.
   side pins of the potentiometer go to +5V and ground
 * LED connected from digital pin 9 to ground

 created 29 Dec. 2008
 Modified 4 Sep 2010
 by Tom Igoe

 Modified 27 July, 2011
 by Keith Wakeham

 This example code is in the public domain.

 */

// These constants won't change.  They're used to give names
// to the pins used:
const int analogInPin1 = A0;  // Analog input pin that the
   potentiometer is attached to
```

```
const int analogInPin2 = A1;
const int analogInPin3 = A2;
const int analogInPin4 = A3;
//const int analogOutPin = 9; // Analog output pin that the LED
   is attached to

int sensorValue1 = 0;         // value read from the pot
int sensorValue2 = 0;
int sensorValue3 = 0;
int sensorValue4 = 0;
int sum1 = 0;
int sum2 = 0;
int sum3 = 0;
int sum4 = 0;
int count = 0;
unsigned long timer;
unsigned long pulsetime;

void setup() {
  // initialize serial communications at 9600 bps:
  Serial.begin(115200);

}

void loop() {
    // read the analog in value:

    if (count < 10){
      sensorValue1 = analogRead(analogInPin1);
      sensorValue2 = analogRead(analogInPin2);
      sensorValue3 = analogRead(analogInPin3);
      sensorValue4 = analogRead(analogInPin4);
      sum1 = sensorValue1 + sum1;
      sum2 = sensorValue2 + sum2;
      sum3 = sensorValue3 + sum3;
      sum4 = sensorValue4 + sum4;
      count++;

    } else {

//    output = sum * 0.007324218;//0.003662109;
    //output2 = sum2*0.004354581;
    pulsetime = micros() - timer;
//    dout = (sampleold - output)/(pulsetime*1e-3);
//    sampleold = output;
    Serial.print(sum1);
    Serial.print( ", ");
    Serial.print(sum2);
    Serial.print( ", ");
    Serial.print(sum3);
```

```
 Serial.print(", ");
 Serial.print(sum4);
 Serial.print(", ");
 Serial.println(pulsetime);
 timer = micros();
 count = 0;
 sum1 = 0;
 sum2 = 0;
 sum3 = 0;
 sum4 = 0;
 }
 //delay(10);
}
```