Visual Frets for a Free-Gesture Musical Interface

by

Leila Hasan

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degrees of

Bachelor of Science in Electrical Science and Engineering

and

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2001

Author .....
Department of Electrical Engineering and Computer Science
June 3, 2001

Certified by .....
Joseph Paradiso
Principal Research Scientist
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Visual Frets for a Free-Gesture Musical Interface

by

Leila Hasan

Submitted to the Department of Electrical Engineering and Computer Science
on June 3, 2001, in partial fulfillment of the
requirements for the degrees of
Bachelor of Science in Electrical Science and Engineering
and
Master of Engineering in Electrical Engineering and Computer Science

Abstract

A new electronic musical instrument has been created, referred to as a 'termenova' (Russian for daughter of Theremin), that combines a free-gesture capacitive-sensing device with an optical sensing system that detects the reflection of a hand when it intersects a beam of a visible laser array. The laser beams, which are made visible by a thin layer of theatrical mist, provide visual feedback and guidance to the performer to alleviate the difficulties of using a non-contact interface as well as adding an interesting optical component for the audience. The system uses several capacitive sensing channels that detect the proximity of the player's hands; this distance is mapped to pitch or volume. The laser guide positions are calibrated before play with position-controlled servo motors interfaced to a main controller board; the location of each beam corresponds to the position where the performer should move his or her hand to achieve a prespecified pitch. The optical system senses the distance of the player's hands from the source of each laser beam, providing an additional dimension of musical control.

Thesis Supervisor: Joseph Paradiso
Title: Principal Research Scientist

Acknowledgments

To **Joe Paradiso**, my thesis advisor, for his encouragement, for his imaginative solutions, and for his infectious enthusiasm for engineering and music.

To **Matthew Hancher**, who's limitless advice helped save my project on multiple occasions and who never let me give up; **Max Davis**, for mechanical design sessions with tea; **Dan Stiehl**, for lending me a lathe when I really needed one; **Laurel Smith**, for late night smart dog sessions; and **Farhad Ebrahimi**, for driving me home from work through the ice and snow.

To all those who helped me get this far: **Holly Gates**, who taught me how to solder 6 years ago and a million things after that; **Francisco Delatorre**, for making me laugh when I'm sure I don't want to; and **Rebecca Zook**, for her poetry for electrons and crazy girls.

To **Anne Hunter**, without whom many MIT students would never make it through.

And finally to **my family**, who remain some of the most interesting and wonderful people I have ever met.

Contents

List of Figures

Chapter 1

Introduction

The theremin, invented by Leon Theremin in 1919[2, p.24], is widely regarded as the first successful electronic musical instrument. It uses a heterodyned variable frequency oscillator controlled by the proximity of the player's hand or body to a capacitive sensing element, such as an antenna. In addition to being considered one of the first electronic musical instruments, it is the first known example of a *free-gesture musical interface*, where there is no physical contact between the player's body and the instrument. Use of this type of interface has maintained a steady interest over time, with an explosion in popularity in recent years, as work with electric field sensing, infrared proximity detection, and computer vision have produced variations on the theremin and evolved entirely new free-gesture interfaces that use computers to interpret the gestural cues at a higher musical level.

The appeal of a free gesture musical interface such as the theremin lies in its elegance and expressive potential. When playing the theremin, the slight tensing of the fingertips translates to a perceptible change in pitch; delicate vibratos and other lyrical qualities normally confined to the world of non-electronic musical instruments become part of the musical language. And while the theremin, the first of the free gesture musical interfaces, was created by Leon Theremin because he envisioned an instrument that could, in his words, "create sound without using any mechanical energy"[1, p. 8], for most performers (and the audiences) of non-contact music playing, the enthralling part of such an instrument doesn't lie in the effortlessness of the

playing; rather there is a mesmerizing effect found in the apparent extraction of music from "ether"[2], from nothing but the surrounding empty space.

In recent times, there has been a sudden upwelling of performing music with non-contact interfaces. Many of these devices use infra-red in various ways to achieve a distinct kind of gesture capture. For example, the $500 Dimension Beam from Interactive Light creates an infra-red imaging field that tracks hand motions in a 30"x18" area, with gestures triggering MIDI events [3]. The Twin Towers, developed by Leonello Taraballa and Graziano Bertini at CNUCE in Pisa, like the theremin measures proximity, but instead of antennae uses a pair of IR-based proximity detectors which can also sense two axes of tilt[12]. [1] Other non-contact interfaces include the Videoharp, developed by Dean Rubine and Paul McAvinney of Carnegie Mellon University, which senses finger gestures by examining the pattern of shadows projected by the player's hand onto a photosensitive strip sensor[10, p.49], and the Laser Koto, created by Miya Masaoka, which mixes up two different non-contact instruments; one hand plays an ultrasound-based interface, while the other interacts with a set of proximity-sensing laser beams [9, p.116].

Though there are a variety of non-contact instruments, and there is a definite audience for performances on such devices, there is a striking lack of virtuosic players. In the case of the theremin (and with non-contact instruments in general), to adeptly play the instrument, not just as a special effect but as a complex lyrical instrument, one requires precise skill, an exceptional sense of pitch, and a steady hand. The performer gets only audio feedback, and waving one's hands in the air with precision is a difficult skill to master; the playing space often has no visible structure, and there is, by definition, no physical contact, which would provide any kind of absolute reference.

This thesis documents the design and implementation of a new type of electronic musical instrument, given the name *termenova* (Russian for "daughter of Theremin"), which combines the non-contact sensing methods of the Theremin with an array of

[1] Note that these interfaces infer proximity from reflected intensity, and therefore can't make reflectance-independent measurements like a true range-finder[6].

dynamic laser guides. These guides, or visual *frets*, define the play space to the performer by marking the locations of known pitches or events. The system can calibrate the positions of the laser guides for each performer and for any set of desired frequencies, as well as dynamically re-tune or reposition during play. Additionally, each guide can sense the position of the player's hand along the laser beam, adding another dimension of continuous control.

This document is arranged as follows:

Chapter 2 is an overview of the high level attributes of the system and the approach to be taken by a user of the system.

Chapter 3 follows the design process and implementation.

Chapter 4 contains the results and conclusions.

Chapter 5 proposes future work with the system.

Chapter 2

System Overview

The termenova retains the unique and elegant non-contact properties of the theremin while providing a perceivable structure to the performer. The playing area is defined by an array of *visual frets*; these frets are realized by a set of visual wavelength lasers arranged so that their beams are parallel and made visible to the human eye by filling the play area with a glycol based fog. In its simplest mode of operation, The laser beams serve as a guideline to the performer for the location of pitches. The theremin's capacitive sensor acts as a continuous proximity sensor, with the lasers marking out known distances from the antennae. Moving one's fingertips to a laser beam's location will produce the pitch that is specified by that fret.

The response of the theremin varies widely; the oscillators used to generate the heterodyned frequencies in the classic theremin tend to naturally drift, and environmental factors such as humidity and temperature further change the reaction of the instrument. Even with a modern, more robust method employed for capacitive sensing, characteristics of the performer, such as their capacitance coupling and their play method and body posture, will also all affect the relationship between the physical laser beam positions and the corresponding pitches. To allow for calibration for such effects, each laser is equipped with a reflectance sensor tuned to the modulation frequency of its associated laser that can unambiguously detect the presence of a performer's hand. As the performer moves his or her hand (in play position) through the laser beams, the capacitive pickup voltage is recorded as each fret position is passed.

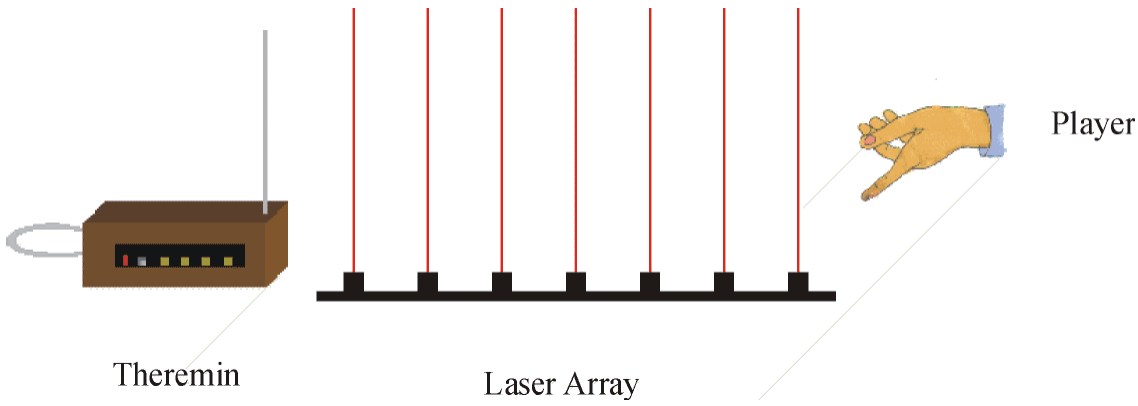
To adjust to this information, the physical positions of the frets are adjusted so the beams represent hand locations for a known set of pitches.

There is also the potential to play the termenova like a harp, much like Jean Michel Jarre's Laser Harp [4]; here an interrupted laser beam corresponds to 'plucking' a visual string, with the sound being modulated by both the capacitively-detected hand position and the position of the hand along the laser axis, inferred through the amplitude of reflected illumination. Thus the termenova can be played as a discrete, continuous, or mixed discrete and continuous pitch instrument.

2.1 System Modes

The system consists of an electric field sensing device interfaced to a laser positioning system. The player stands with the laser array between them and the sensing device, using the laser array as a guide for placement of their hands during play . A overview diagram of the system can be seen in Figure 2-1.

Figure 2-1: Overview of System



The instrument has several main modes of use, calibration mode, continuous play mode, and discrete play mode.

2.1.1 Calibration

Before using the instrument, the player must calibrate it for their body as well as the room conditions. The signal seen by the electric field sensing device will vary with a variety of factors, including capacitance of the player, playing position and style, temperature, humidity, and proximity to other objects. Therefore, for any kind of change in player or environment, the instrument must be calibrated, or "tuned".

The calibration procedure first requires the player to specify a desired set of frequencies they would like the laser beams to represent, i.e. when their fingertips are touching a beam, they would like that body position to map to a desired frequency. Next, the player moves their hands through the laser array area as if they were playing the instrument. A mapping of the readings from the sensing device to the current positions of the lasers will be created as the player moves their hands through the beams, and, based on this data, a new set of positions for the lasers will be created to approximately match the user specified mapping. The lasers will move to these positions and the player will again move their hand in playing position through the beams. With this second reading, the positions of the lasers can be accurately determined.

2.1.2 Play Mode

In play mode, the termenova can be used as a continuous or discrete pitch instrument, or as a combination of the two.

Continuous Pitch

While playing the termenova as a continuous pitch instrument and maintaining the same playing position they did for calibration, the laser guides will correspond to the places for the player to place their hand to achieve the pre-specified pitches from the instrument. Moving between guides will produce a continuous change in frequency, and the player can play any pitch they desire, with the lasers serving as guidelines to the play area. However, the player should attempt to maintain the same playing

position consistently, or the locations of the pitches may shift.

Discrete Pitch

While playing in discrete mode, the instrument resembles a harp, with the beams corresponding to discrete pitches. The sensing element in this mode is the optical sensor of the laser guides themselves instead of the electric field sensing device.

Hybrid Mode

The instrument can also be played as a hybrid of the continuous and discrete modes. The position of the performer's hand along the laser beam can be used as another element of continuous control. The instrument is played as in the continuous pitch mode, with distance from the capacitance sensor continuously mapped to frequency, but when the hand is positioned above a laser, the pitch locks, and subsequent movement will result in a continuous variation of pitch, with the pitch returning to the reference value if the hand stops moving and is still in the path of the laser. In this mode, one can set up the system such that other effects (e.g. timbre control) will be controlled by moving vertically along the the beam.

Chapter 3

Design and Implementation

3.1 Design Considerations

While designing the system, several goals and guidelines had to be kept in mind:

Light Source A source must be chosen for the visible beams.

Beam Positioning The visible beams need to be capable of changing position.

Object Sense The system must be capable of sensing when there is an obstruction in a beam, and for what beam there is an obstruction.

Sensing Device A proximity-sensing device needs to be chosen that delivers position data.

System Cost There are on the order of 10 visual guides, so for a system that requires separate lasers, electronics, and mechanics for each visual guide, the individual module costs should be kept at a minimum, since the total cost will be multiplied by a factor of approximately 10.

3.1.1 Light Source

To create a beam of light, the light must be collected and collimated. Lasers were the natural choice for this application, though in the future there is also the possibility of using bright LED's if the light can be properly collimated.

3.1.2 Beam Positioning

If laser light is used for the visual guides, there are two distinct ways of positioning the beams:

- With one laser and an oscillating mirror, or
- with many lasers, each mounted on a position controllable platform.

While there is a certain elegance to the former (and indeed, this is the method of choice employed by Jarre's *Laser Harp*[4]), it requires a laser of considerable power, as the illumination must be distributed among 10 different beams. Such a laser system is generally quite expensive (in the many thousands of dollars range), unwieldy, and difficult to control.

The second system requires many lasers, as well as a distinct movable platform for each laser. However, a 5mW laser diode is in the several dollars range and requires only simple drive circuitry to be modulated in the kHz range.

This system calls for a set of laser platforms, or modules (as they will henceforth be referred), and requires a mechanical system capable of moving each module to a distinct position. There are several ways to do this; The initial design used a cable drive system, which required a total of one motor for the entire system as well as one solenoid per module. The specifics of this design can be found in Appendix A.

The implemented design uses a rack and pinion system, with each module containing one motor; with this design, each module can be moved independently to an exact position.

3.1.3 Object Sense

In order to "see" the player's hand and calibrate the system properly, there must be a mechanism that can sense when an object is obstructing a beam and differentiate which beam it is.

One way to do this would be with an array of break-beam sensors. In such a scheme, a photo-sensor is aligned with the light of a corresponding laser, and when

an object passes through the beam, the sensor will register the missing light source. There are several problems with such a scheme. A light sensor would need to be mounted a minimum of several feet directly above each laser, so some sort of platform or hanging device would need to be installed above the play area. Additionally, since the laser beam locations are dynamic, each light sensor would need to move in a precise way to remain aligned with its corresponding laser (alternatively, instead of having one sensor per laser, the upper platform could be filled with a dense array of sensors).

Another, less bulky way of solving the problem is to use reflectance sensors. In such a setup, a photo-sensor is located near the source, aimed in the same direction as the source, and picks up the light that is reflected back. This system requires the sensors to be able to differentiate the relatively small amount of desired laser light from all other received light, i.e. from other lasers or any kind of ambient light. However, well established solutions exist to this problem as described in Section 3.2.1, and reflectance sensors were chosen for the system. Using reflectance sensors instead of break-beam sensors also allows for the determination of the approximate distance of the object from the sensor, instead of simply determining whether the object is there. This added sensing element can be used to add more levels of control to the instrument.

Another way to monitor the beams would be with video; a video camera could be observing each beam. However, this would require a significant number of cameras (10-20), and seems like an overly complex and somewhat unwieldy solution.

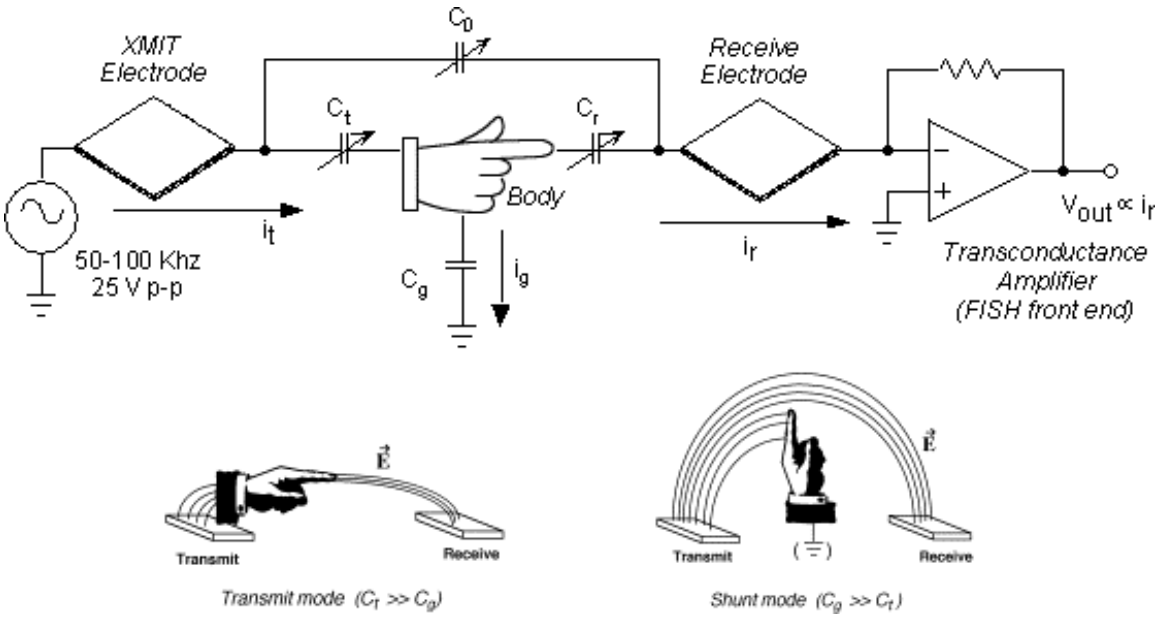
3.1.4 Sensing Device

The sensing device must have a range that is extensive enough for the player to take advantage of the visual guides. For example, the system is less advantageous, though still somewhat helpful, with a standard theremin, as the range is approximately 3-4 octaves within 1 foot of the antenna [2, p.23].

A device called the *LazyFish* was chosen for this application. This device was developed by Joshua Smith at the MIT Media Lab and has multiple channels of

electric field sensing [11]. It is capable of employing two methods of capacitive sensing: *transmit* and *shunt* modes [7]. In *shunt* mode, the performer moves between a receive antenna and a transmit antenna. The signal decreases as the player blocks more of the transmitted signal from the receiver, as shown in Figure 3-1. In *transmit* mode, the body of the performer is connected to a broadcasting electrode, either by standing on a driven plate or by some other unobtrusive method of attachment (e.g. a tethered electrode in a wallet pocket), and a signal is transmitted into the performer through the electrode, as also illustrated in Figure 3-1. The player's body, attached to the electrode, becomes an antenna for the transmit signal; the received signal becomes stronger as the body of the player gets closer to the receiver. Several feet of ranging capability can be achieved by using this mode, and it is extremely simple to implement, as only the approach of the performer's body will create a response at the receiver electrodes.

Figure 3-1: Lazy Fish: Shunt and Transmit Modes



3.2 Implemented System Overview

The termenova contains a set of laser modules that are capable of moving independently on a shared track and communicating with a main controller. This separate main controller regulates the entire system, using data from the modules and from the field sensing device to determine the desired state for each module.

Every module contains a modulated laser with matched synchronous receiver; the receiver detects the light reflected from an obstruction of the beam. Each laser has a distinct frequency, and the receivers detect only the light reflected back from its own module. The sensors are sensitive enough to detect bare-handed reflections up to several feet away.

To enable independent movement, each module is equipped with its own miniature servo motor. The motor shaft is coupled to a pinion that traverses a rack; the shaft is also coupled to a multi-turn potentiometer, which provides absolute position data.

On board every module is a microcontroller which transmits and receives data from the laser system and motor system. Additionally, it communicates with a main controller board via a serial bus, with a single receive line and single transmit line shared by all the modules. A system block diagram is shown in Figure 3-2.

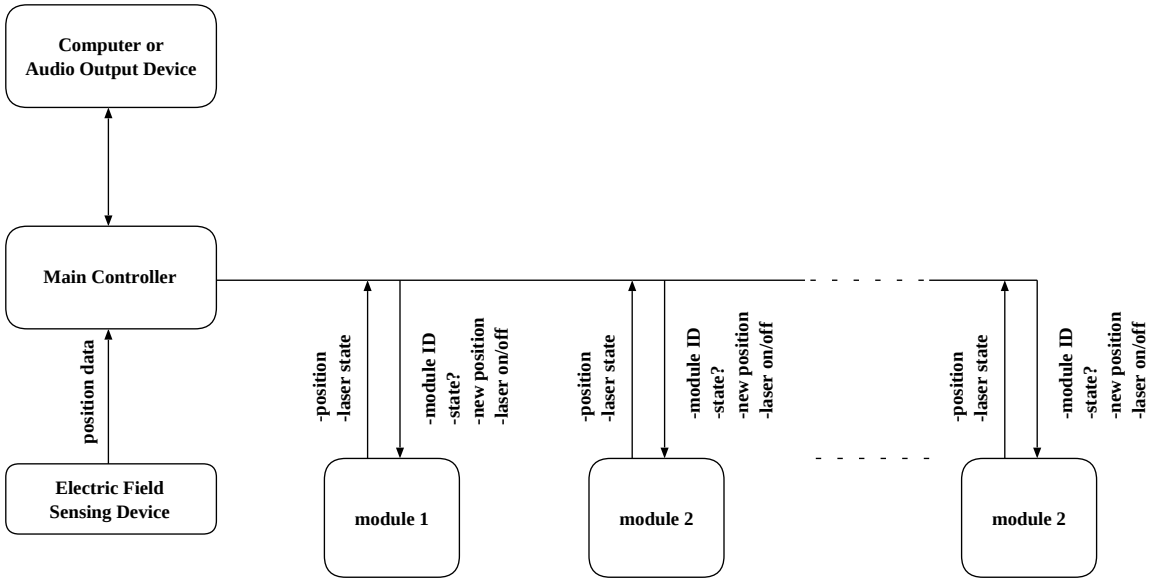
3.2.1 Laser System

The laser system consists of a visible wavelength ($\sim 635nm$) red laser unit and a synchronous detector, both driven by a single signal line from the microcontroller. The laser system provides the visual guides; it also can sense the presence of an object blocking a beam.

Laser Diodes

For the visible array of frets, a set of laser diodes are used. Laser diodes are readily available, inexpensive, and relatively simple to control and drive; all these were important considerations in the design, as it calls for 10-20 modules and each contains a full set of parts, including a laser. A laser diode/driver package was found whose

Figure 3-2: System Block Diagram



electronics were easily modified to accomodate being driven at the desired carrier frequencies, in the vicinity of 5kHz.

Photo-detector

A combined photodiode and transimpedance amplifier was used to sense the reflected laser. A 15mm diameter focusing lens with a focal length of 22.5mm was used to collect the light and focus it onto the 2.3mmx2.3mm photodiode surface. A more detailed description of the photo-detector can be found in Appendix B.

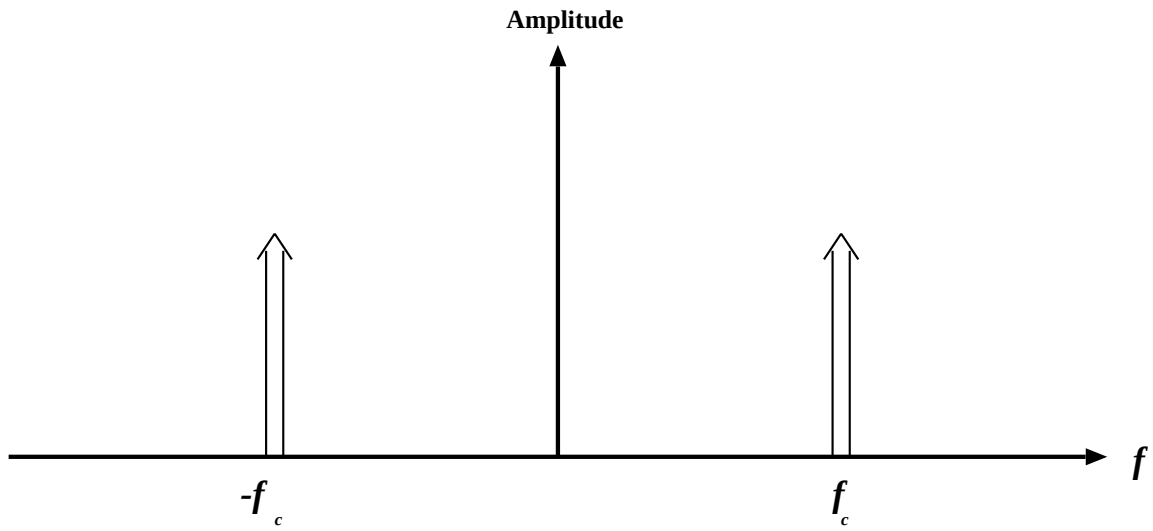
Synchronous Demodulation

Synchronous demodulation refers to a process by which a received signal is multiplied by the modulation of the excitation source (which beats it down to the baseband), then low-pass filtered to recover the original signal. It is used in many applications, such as AM radio and spectroscopy, to find a desired signal buried in a sea of unwanted signals and noise.

In this project, the original signal is a constant (the laser driving voltage) which is modulated by a carrier frequency of approximately 5KHz. In the frequency domain,

this modulated signal looks like a simple sinusoid, as in Figure 3-3 [1].

Figure 3-3: Modulated Laser Signal (Frequency Domain)



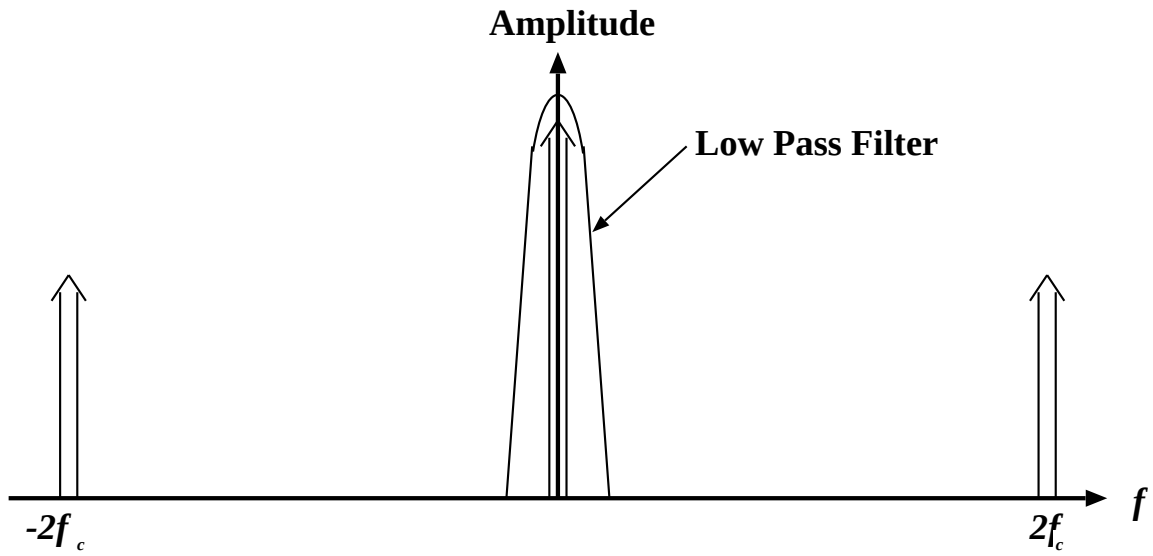
The modulated signal and carrier signal are then multiplied, which corresponds to convolution in the frequency domain and which results in a signal consisting of two components: one at twice the carrier frequency, the other at DC. The result of this can be seen in Figure 3-4. This signal is then low-pass filtered, to isolate the component at the DC baseband, which corresponds to the original signal.

When an object reflects the modulated laser beam back at the module, the signal seen by the photo-detector at the receiver end will contain a great deal of noise with a small amount (relative to the noise) of the modulated signal. Any undesired noise or signal that is farther in frequency from the carrier signal than the bandwidth of the low pass filter will not appear in the output of the demodulator, and the result predominantly corresponds to the amount of reflected laser light being seen by the photodiode.

The demodulated output is amplified and fed to the analog port of the micro-

[1] This figure actually corresponds to the signal as modified by a sinusoid. In actuality, it is being modified by a square wave, which results in a much more complicated representation. For ease of explanation, and because the results in this particular application are nearly identical for a sinusoid and for a square wave, a sinusoidal modulation signal will be assumed.

Figure 3-4: Demodulated Laser Signal (Frequency Domain)



controller where the threshold for a reflected beam can be set in software. A block diagram for the system can be found in Figure 3-5. For the circuit diagram and detailed circuit description, refer to Appendix B.1.

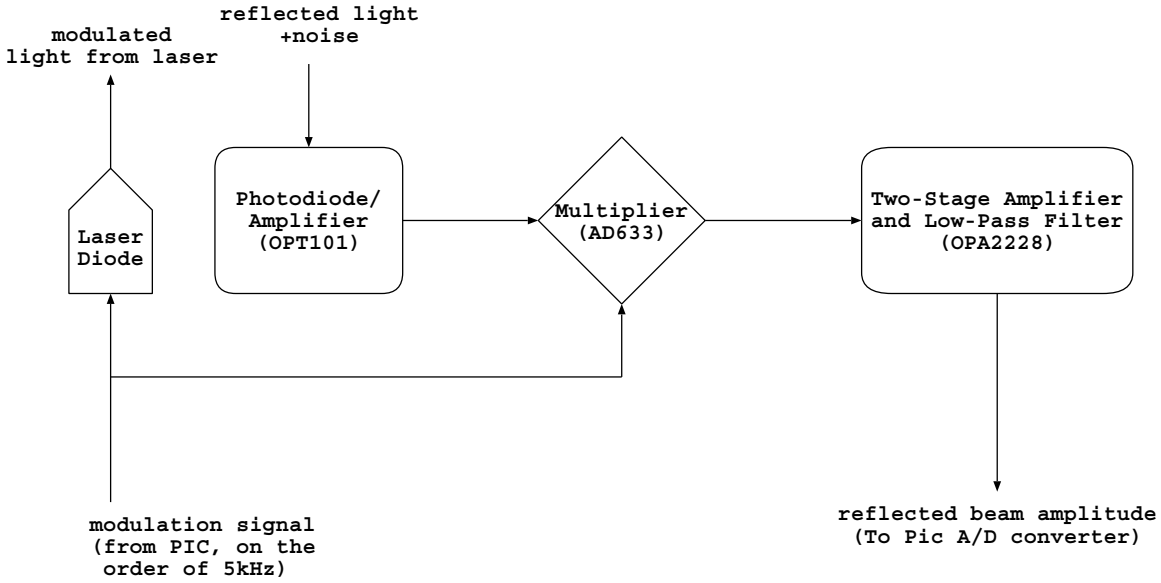
3.2.2 Motor System

For the motor system, an inexpensive, absolute positioning system for the entirety of the track was desired. To achieve this, the method employed by hobbyist RC servos was considered; hobby servos incorporate robust position control using only a potentiometer as feedback, and are extremely inexpensive ($10-$30). However, they tend to be capable of moving a maximum of only one half turn of the servo.

RC Servo

An RC servo was modified to retain a powerful motor and useful gear train, but with the potentiometer and electronics removed. An inexpensive 20-turn vertical trimpot was coupled to the shaft, and its output fed to a 16-bit analog to digital converter.

Figure 3-5: Laser System Block Diagram



Position Data

Ideally, a 16-bit A/D corresponds to over 65,000 positions. With a gear diameter of .6" and a 20 turn pot, that is 65,000 over a 37" long area of travel, which translates to .0006"/bit. This is a gross overestimation of the capabilities of the system, as it is far from ideal. However, even assuming a worst case scenario of only 12 good bits from the 16-bit A/D, there are still about 4000 positions, which for 37" of travel corresponds to .009"/bit, which is still amply accurate for the needs of the system, as the average human being can not visually discern a hand movement of 1/100".

Motor Control

For motor control, the only input to the controller is absolute position, acquired from the potentiometer. The microcontroller applies pulse-width modulation (PWM) to the motor drive voltage, which controls the torque of the motor and approximately controls the speed, since the track is relatively uniform.

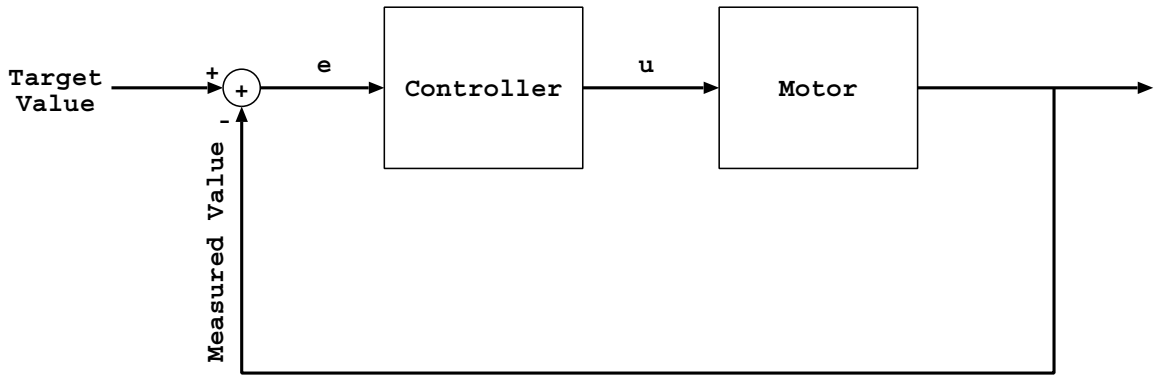
Position control using only position data has the potential to result in an unstable system, but with the motors moving relatively slow and the gain kept below a rea-

sonable threshold, robust position control is possible. This was realized by a digital PD (proportional+derivative) controller [5]. A sketch of the control system is shown in Figure 3-6, and the equation for a PD controller based on this system is:

$$u = K_p * e + K_d * \frac{de}{dt} \tag{3.1}$$

- where e is the error between the target and the measured position
- u is the control signal from the controller, here a PWM voltage.
- K_p is the proportional gain
- and K_d is the derivative gain

Figure 3-6: Motor System Block Diagram



A circuit diagram and analysis of the circuit can be found in Appendix B.2, and the complete control code can be found in Appendix D.2.

3.2.3 Microcontroller

A PIC16F876 was used as the microcontroller in each of the modules; it was chosen as it has the following useful features:

- a 10 bit A/D converter[2]

[2]The reason the PIC A/D was not used for the motor positioning is that the 10 bits of the A/D actually translate into around 8 bits of robust data, which is not enough for the motor position data

- Universal Asynchronous Receiver Transmitter (UART)
- 10-bit PWM module
- 2 timer/counters
- external interrupt capability
- lots of I/O

Module Controller Tasks

The PIC communicates via the UART to the main control board on a serial bus that is shared between all the modules. The main board polls all the boards in succession, logging data values and sending new positions or states when appropriate. A description of the serial protocol can be found in AppendixD.1.

To control the motor, the PIC gets a reading from the 16-bit A/D and drives the motor with the PWM module and an output pin to specify direction, using the control algorithm described in 3.2.2.

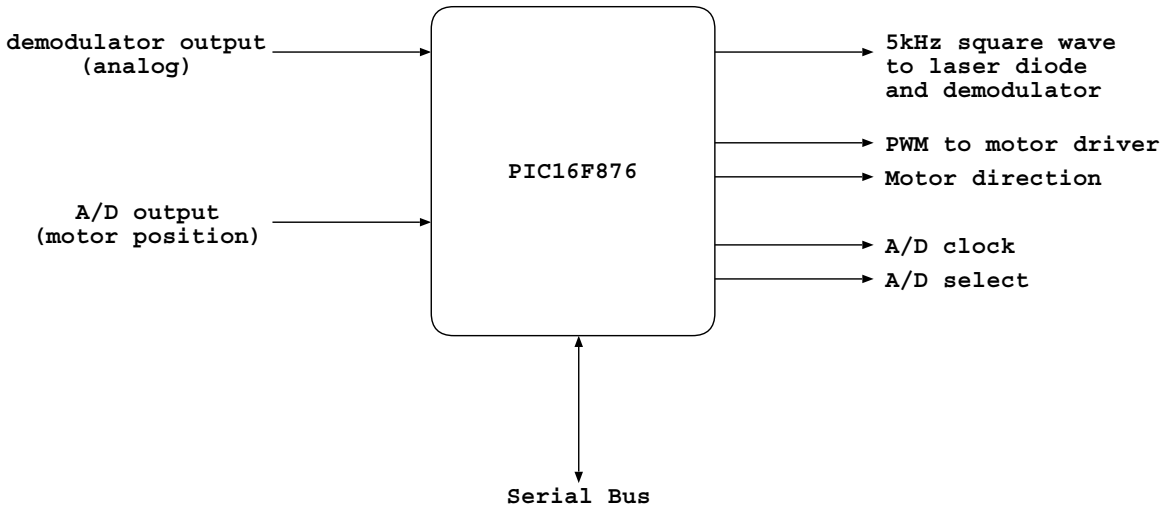
Each PIC uses an internal timer to generate a distinct modulation frequency, which is used by the laser diode and synchronous demodulator. The output of the synchronous demodulator is read by an analog pin of the PIC, and compared to the pre-determined threshold value to determine if an object is in the pathway of the beam. The analog intensity data is also used as a continuous controller when the instrument is in hybrid mode, as previously described in Section 2.1.2.

A listing of the PIC tasks is shown in Table 3-7, and the full assembly code for the modules can be found in Appendix D.2.

3.2.4 The Lazyfish

The LazyFish is an electric field sensing device created at the MIT Media Lab which uses the same fundamental principles as the theremin, but provides a more robust, longer ranging proximity sensor. The theremin detects the the change in capacitance seen by the antenna when it capacitively couples to the player's body. The LazyFish,

Figure 3-7: Controller Tasks



in transmit mode (the mode that is used in the implementation of this system) drives the body of the player with a transmit signal; the player's entire body then acts a transmitting antenna for this signal, and the receive antenna examines the amplitude of the received signal to determine proximity [7].

When using the LazyFish in transmit mode, a sensing distance of several feet can be achieved. Additionally, this system is far less prone to drift, as it does not contain the highly sensitive, drift-prone analog electronics embedded in the theremin.

3.2.5 Main Controller

The main controller board interfaces between the LazyFish, module controllers, and output device. The controller has vastly different functions during calibration and play.

Calibration: Main Controller

During calibration, the main controller gets readings from the LazyFish and modules, and creates a current mapping of frequencies to module positions. From this mapping and the list of desired pitches, the controller determines a new set of positions for the modules based on a piecewise linear fit algorithm.

The algorithm works as follows:

- The modules shall be referred to as mod1, mod2, ...modn in order of distance from the capacitance sensor.
- The modules will be placed so that mod1 and the modn are at the extreme ends of the track, and the others are dispersed between.
- A current frequency will be mapped to each of the modules based on information taken from the LazyFish during the first pass of the player through the playing area during calibration.
- Suppose mod1's desired pitch lies somewhere between the pitches measured at the positions of mod1 and mod2. A simple linear relation based on their current positions and pitches will be determined as follows as shown in Figure 3-8 and Equation 3.2.

$$p_{new} = (f_{desired} - f_1)\frac{p_2 - p_1}{f_2 - f_1} + p_1 \tag{3.2}$$

where

f_1 is the current frequency at mod1.

f_2 is the current frequency at mod2.

p_1 is the current position of mod1.

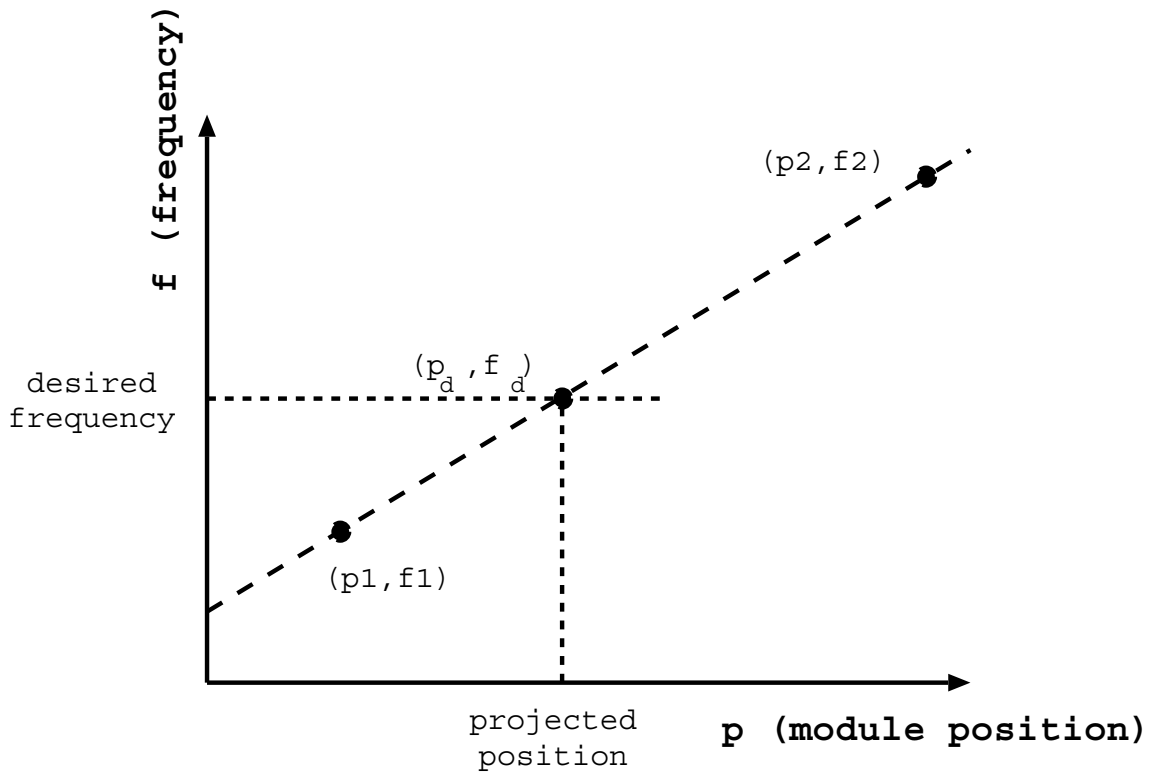
p_2 is the current position of mod2.

$f_{desired}$ is the desired frequency for that module.

p_{new} is the new position.

After determining this for all the modules, the controller instructs the modules to move to the new positions. The player passes their hands through the play area for calibration for the second time, and a new set of frequencies are recorded. The piecewise linear fit is repeated, which results in a close enough approximation to the actual positions which define the desired pitches.

Figure 3-8: Linear Mapping



Play: Main Controller

During play mode, the main controller provides the output device with continuous updates on the pitch data received either from the LazyFish or from the laser guides, depending on if the system is in continuous, discrete, or hybrid mode. The lasers on the modules may be commanded to be on or off, for the possibility of having the system teach the player a melody by selectively lighting the lasers and having the user "play" the notes as they are highlighted. The output of the main controller can be processed by an external program to produce interesting mappings and additional effects that go well beyond the simple theremin mode that this thesis concentrates on.

Chapter 4

Results

In testing the feasibility and functionality of the system, two main aspects needed to be investigated:

- the mapping of the optical detector's outputs to the corresponding capacitance-sensing outputs
- the proximity sensing capabilities of the optical detectors.

4.1 Experiment: Fish to Detector Mapping

In the first experiment, the output of a *Classic Fish* [7] was compared to the detector output. This data gives a picture of both what the detectors are seeing as the hand moves through the play field as well as the consistencies of the mapping of the detector positions to the sensed capacitance.

4.1.1 Experimental Setup I

The system was setup with several functional modules and a *Classic Fish*. The *Classic Fish*, built at the MIT Media Lab by Joe Paradiso, Tom Zimmerman, and Josh Smith, is the precursor to the *LazyFish*, and is similar in fundamental behavior, although all demodulation is done with analog electronics. Both boards use synchronous detection of a transmitted signal to determine proximity, though the *LazyFish* uses much higher

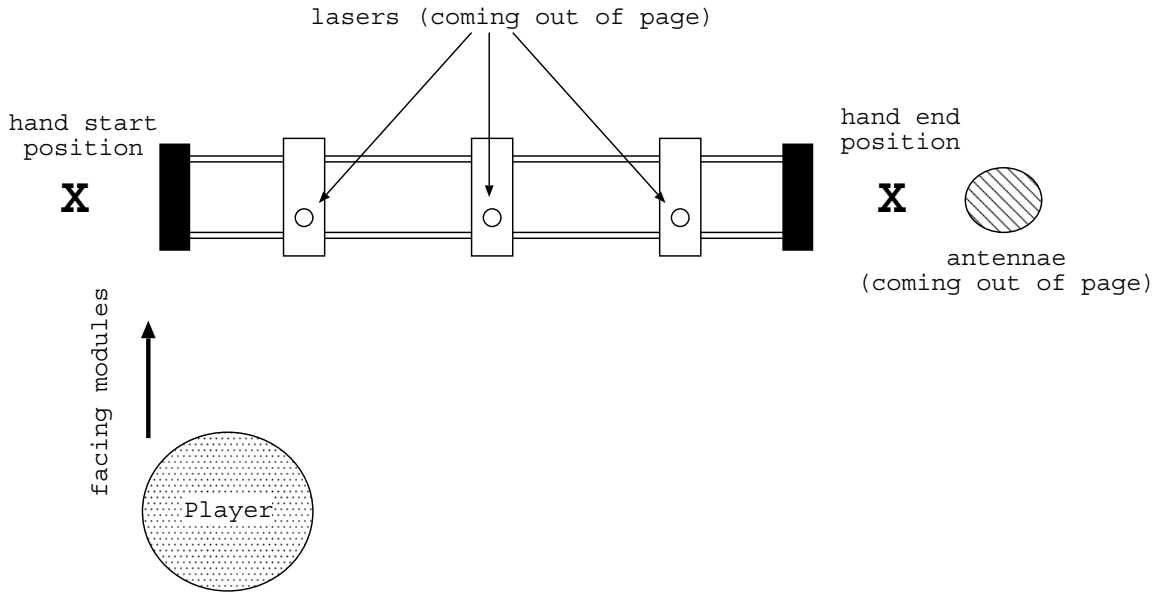
frequencies for its carrier signal. The *Classic Fish*, however, is much easier to adjust (all parameters are controlled by on-board trimpots), and hence was used for initial testing.

The *Classic Fish* was set up to produce an analog value in the range 0-5V, connected to the analog port of the main board PIC. The main PIC then converted this signal into the digital domain using its on-board A/D converter.

Three functional modules were set up on the positioning base, and set at arbitrarily spaced positions along the rails. (The position data for this experiment is irrelevant, as positions during play are set by an arbitrary mapping, so any positions are acceptable). An antenna, created from a 1" diameter by 2' long copper tube, was placed at the far end of the positioning base and connected to the receiver of the *Classic Fish*. Next, a subject stood in front of the module furthest from the antennae (module 1). The subject, holding the Fish's transmit electrode in their left hand, then moved their right hand, located approximately 1.5' above the modules, steadily through the beams toward the antenna and then back to the starting position.

This experimental setup is shown in Figure 4-1. Fish sensor data plus the optical detector data for each module was continuously logged during each run.

Figure 4-1: Experimental Setup 1: Top View



4.1.2 Experiment I Results

The experiment was tried with the hand in two positions: the first was with the palm of the hand facing the surface of the modules, so the laser was reflecting off the palm-side surface of all the fingers.. The second was with the palm held perpendicular to the surface of the modules, so the laser was reflecting off the edges of the fingertips. The latter hand position is akin to the one the system was designed for.

In Figure 4-2, we see the plot of the outputs of the Fish and the modules for a continuous set of samples. Module 1 is passed over first, then module 2, then module 3, then all the way back past module 1 again. From the figure, we can see that when the hand passes each module, there are several peaks in the output level; these peaks correspond to the reflection off each finger. The crevices between each pair of fingers reflect far less light than the surface of the fingers, and a smaller received signal results. Note the extra peak on the first instances of module 2 and module 3, corresponding to the thumb.

Figure 4-3 shows the results when the hand is held parallel to the beams, so the laser unit sees the edge of the fingers. For each module, there is a sharp peak corresponding to where the hand was aligned with the laser, and light reflecting off a surface normal to the beam was presented to the detector.

The mapping of the module data to the Fish data is shown in Figure 4-4. The graph shows that for the module closest to the antennae(module 3), the mapping of the measured capacitance to the optical detector output is consistent for both directions, while for the module furthest from the antennae (module 1), the mapping in one direction is quite off from the mapping in the other.

Figure 4-2: Experiment I (palm down)

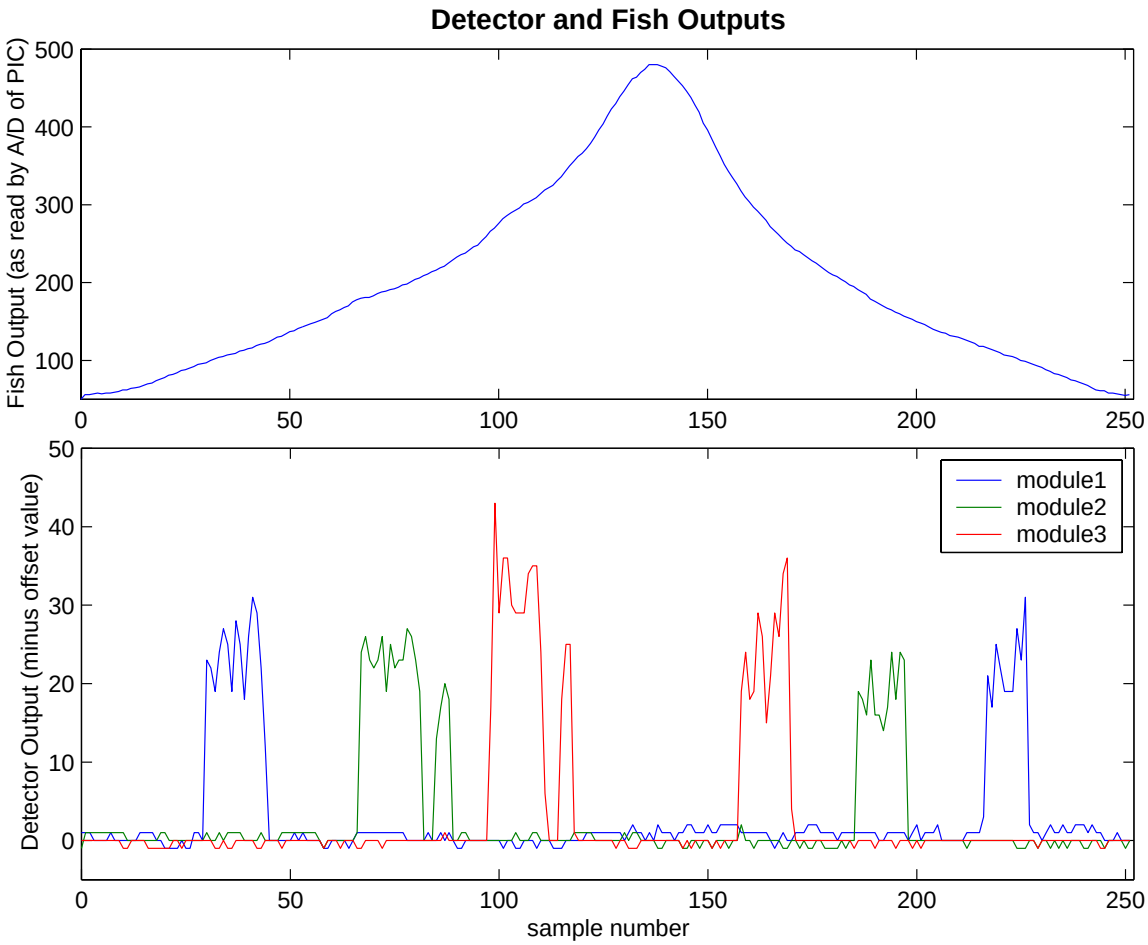


Figure 4-3: Experiment I (edge of hand)

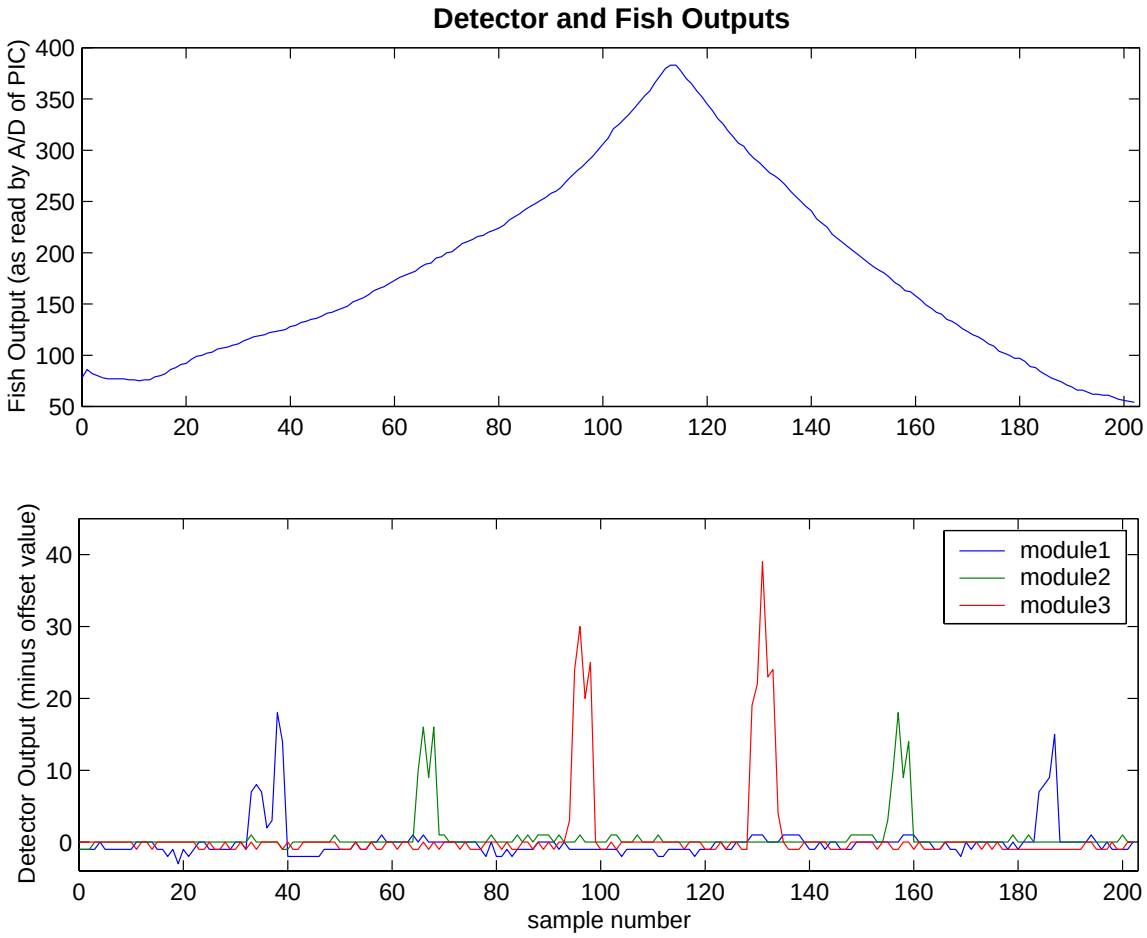
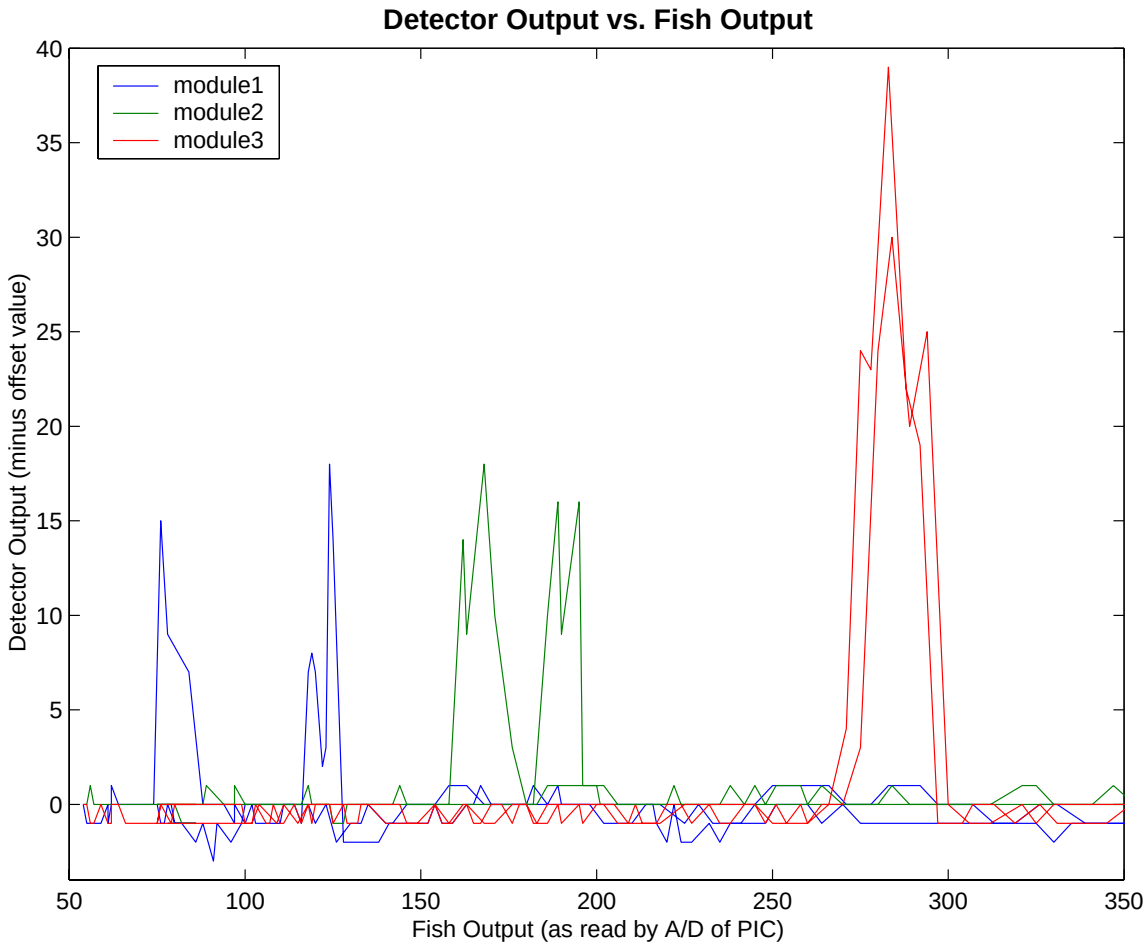


Figure 4-4: Mapping of Outputs (edge of hand)



4.2 Experiment I: Discussion

The discrepancy in the mapping is easily attributed partially to the fact that the player's body moves differently if they are moving toward the antenna than if they are moving away from the antenna. Also, as the player's hand comes closer to the antenna, the effect of the body position on the signal is much less pronounced, as most of the signal seen at the receiver is coming from the hand. When the body and hand are equidistant from the antenna, the receiver is getting most of the signal from the body. The effects of this were especially pronounced since the player was standing such that their body and playing hand were nearly equidistant from the antenna when located at module 1. Further experimenting with body and module positioning must be done to determine the most effective placement of the player with respect to the system. It is likely that this error can be minimized, either by constraining the player so that their playing hand is always significantly closer to the antenna than their body, or by having software that can do error correction, based on which direction the player is moving (additional hardware would need to be implemented to ascertain the direction). Alternatively, an intelligent, interactive mapping could be implemented that treats the capacitive sensor results as relative, dynamically interpolating between the absolute frequencies set by the laser beacons.

From all three figures, we see that the output of module 3 seems to be almost twice as large as that of module 1 and module 2. One of the more difficult and unforgiving aspects of the system is the alignment of the laser, lens, and photodiode. As the active area of the photodiode (.23mmx.23mm) is much less than the focal length of the lens (22.5mm), the unit is in the telephoto region, so the alignment of the lens to the detector is very sensitive. The current system has only two set-screws for placement of the laser, and a small amount of play in the positioning of the photodiode unit. A more precise lens assembly allowing for a more controlled motion of the lens would greatly improve the alignment, and the received signal would be much larger.

Another problem that can be discerned from the graphs is that the signal coming out of each module is relatively small; when an object is seen, there is a response

(for the most sensitive module) of about 40, where the full possible range is 0-255 (corresponding to 8-bits of A/D information from the PIC). Alignment will partially help with this, but additional gain could be put in the amplifying stage, so that the output of the synchronous demodulator spans more of the range of the A/D of the PIC.

One reason the gain was not initially set to a higher value was that the output offset of the multiplier is significant, and it drifts quite a bit, especially for the first few minutes of use, mostly due to temperature changes in the circuit on startup. After reaching equilibrium, it still drifts, but to a much lesser extent. Study of the data implies that a larger gain stage would not make the offset error significant enough to saturate the signal, so the gain may be increased by a factor of 4 or higher to achieve a greater dynamic range in the detector values. A low-Q bandpass filter placed between the photo-detector and the multiplier would also allow for more AC gain. However, a more robust modulation scheme could be implemented by eliminating the multiplier entirely and implementing a synchronous switch in its stead. In the current system, the multiplier is multiplying the received signal with a square wave; this is analogous to merely turning the received signal on and off at a fixed frequency, which could be done with a simple switch instead of a four-quadrant multiplier. This would eliminate the DC offset from this part of the circuit.

4.3 Proximity Sensing with Optical Detector

In this setup, the proximity sensing capabilities of the optical detectors were tested.

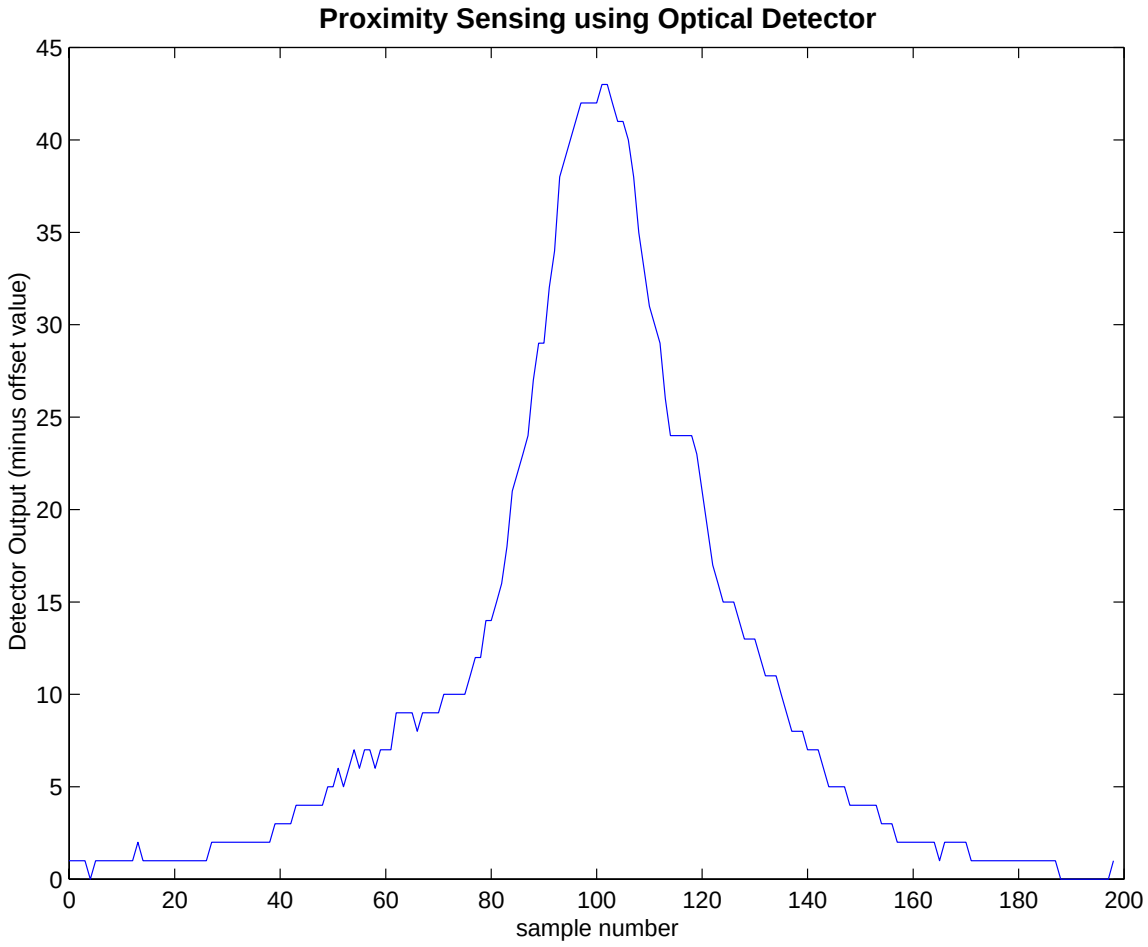
4.3.1 Experimental Setup II

For this experiment, only one of the modules was used, without the capacitive-sensing Fish. The subject stood in front of the module, starting with their hand held as high as they could (corresponding to about 4 feet above the module) in the path of the beam. Then they moved their hand in the path of the beam at a steady rate down to about 1 foot above the module, then back to the start position.

4.3.2 Experiment II: Results

The results are shown in Figure 4-5. As one might expect, the signal approximately drops off in proportion to the square of the distance from the module to the hand.

Figure 4-5: Experiment II



4.3.3 Experiment II: Discussion

The focus of this experiment was to ascertain whether the output of the optical detector is sufficient to use it as a proximity detector. According to the graph, one can infer that there is enough information for this application. Furthermore, with a higher gain stage for the post-multiplier amplifier and with the implementation of

other modifications suggested in Section 4.2, the proximity data (currently about 4 bits worth) can probably be improved to 6-8 bits worth.

Further experiments on using the optical detector for proximity include studying the effects of tilt on the perceived distance or examining the output of the Fish as the vertical distance is changed.

Chapter 5

Conclusions and Applications

In this thesis, the design, implementation, and testing of a new kind of free gesture musical interface were explored. From the results, we can conclude that such a system is both feasible and reasonably economical. There remain many issues pertaining to robustness and precision, but the fundamentals of the system have been created, and with further experimentation and design revision, a device that opens up new modes of musical control and expression can be created.

5.1 Difficulties

5.1.1 Mechanical Issues

Multi-carriage positioning systems are, in general, extremely expensive and specialized, usually for extreme precision applications. For this thesis, a modular mechanical absolute-positioning system was built and tested. The system is not nearly as precise as available commercial positioning systems, but it's also only a tiny fraction of the cost.

Though an industry-level of precision was not required for this application, there are still improvements that can be made to the positioning system. The potentiometer assembly that is used for the feedback has a fair amount of backlash, plus there is backlash in the motor itself. The total backlash ends up being about $20 - 25^o$, which

corresponds to up to about $\frac{1}{10}$th of an inch along the track. The backlash can be corrected for in software, but it makes motor control more difficult and results in a less elegant system. A more precise shaft coupling could be created, or the current one could be modified to be more rigid.

An additional mechanical issue is that of noise. The motors themselves are noisy RC servos. When they are inside the modules, the module housing acts as a amplifier to the noise. Additionally, the current bushings are made of acrylic (a material which has poor sliding characteristics), and the movement of them upon the rails contributes a fair amount of high-pitched noise.

The acrylic can easily be replaced with delrin, which is an excellent bushing material, but the servos are inherently noisy, mostly on account of the nylon geartrain. A possible solution would be to fill the gear-box housing with a suitable lubricant, which would cut down the gear noise quite a bit. An additional solution would be to use the motor noise as yet another part of the instrument. The motors could be controlled to move rhythmically, to add another possible dimension, one that is both aural and visual.

5.1.2 Hardware Issues

Though both the motor board and laser board are functional, there are some improvements and/or additions that could be made; some are for functionality and some are for adding new application potential.

The laser board alignment method and receiver gain need to be improved, as discussed in Section 4.2. However, the most problematic aspect of the laser board hardware was the multiplier (AD633), which had an drifting offset of up to 50 mV. A trimming pot was used to mostly correct for DC offset, but the drifting offset problem still remained. One solution would be to use a more expensive multiplier, as the one that is being used is at the lowest end (and is appropriately priced). However, good multipliers are somewhat costly and are overkill for this application. There exist other solutions for this kind of simple synchronous demodulation, such as the synchronous switching mentioned in Section 4.2 or the synchronous undersampling employed by

the *LazyFish* [11], which performs the same function as the multiplier and low-pass-filter by using a microcontroller and ADC (the *LazyFish* uses a PIC16C711 which has a built-in ADC).

5.1.3 Additional Issues

There are many aspects of the design that need extensive testing; the system needs to be used by different performers to achieve a sense of what a player would want to do with the instrument. Additionally, the system is intended to be adaptable to any player, so it must be relatively robust to slight differences in playing method. Further calibration and/or redesigning may have to be done to account for issues such as tilting of the hands, playing at a slant, or the probable event that players will have differing ideas about where the laser should be with respect to their hands; the system assumes the hands should be placed so the end of the fingertip overlaps the beam, but it would be desirable to have a system where the player could choose which part of their hand they wanted the the laser to correspond to.

5.2 Software Mappings and Applications

Software mappings can completely redefine the nature of an electronic instrument. Using information from the current system or from a slightly modified version, a variety of mappings can be produced that go well beyond the simple musical world of the theremin. In addition, there are aspects of this system that have potential for performance beyond the musical, as it is also a visual device. Most of these applications are not within the scope of this thesis, but may be explored in the future.

5.2.1 Pitch Lock

One problem with playing a theremin is that without an excellent sense of relative pitch, it is very difficult to play a note without being a slight bit sharp or flat; this

is one reason why very few people in the world are capable of producing an accurate melody from a theremin. A useful element to add to the system would be a *lock to pitch* feature, where, if so desired, the output would slide to the closest frequency that is defined as part of the current melodic language (which could include frequencies beyond the 10 or so marked by the laser guides), while still allowing for modulation control, to retain the capability for vibrato and other desirable effects.

5.2.2 Fixed Position Mapping

If a computer module is added to the system, then a software mapping of the system would be possible. Here, the player chooses the position of the lasers as well as the pitches they represent, and the software generates a smooth continuous mapping of pitches between laser guides. Additionally, with the computer module, there are an infinite number of possible mappings, not limited to a fixed pitch mapping.

5.2.3 Multiple Mappings In-Performance

During a piece, the performer may desire to use a different scale, or have the visual frets represent a different set of pitches, which would imply a different set of positions for the laser guides. In calibration, multiple readings of the players hands in the play area are taken, so it would be relatively simple to make use of this information to generate a new set of laser positions for the new set of pitches.

5.2.4 Dynamic Tuning

A common problem with using electric field sensing systems (especially analog ones like the theremin) is that the characteristics of the sensing device or of the objects being sensed may change over time. If this change is significant over the course of a performance, the system should have a way to dynamically re-tune the system while the performer is still playing. In order to do this, accurate position, with respect to the laser guides, and frequency data must be gathered and compared to the original mapped values. For example, if the system could recognize the direction of the

movement of the players hand, then when a beam break occurred, a relatively accurate mapping of the pitch associated with that laser guide could be generated, and screened for errors. Replacement of the current photodiode with a dual or quadrant array photodiode could provide the system with the capability of discerning the direction of the hand motion.

5.2.5 Z-axis Measurement

As shown by the work in this thesis, an approximate Z-axis measurement may be made by observing the amplitude of the reflected laser light. This amplitude increases when the hand is closer to the source, though it is also largely dependent on the angle of the reflected light off the hand as well as the reflectivity of the object doing the reflecting. The intensity information from this mapping changes the instrument from a 1-dimensional device to a multi-dimensional one. For example, a mapping could be done whereby the instrument is in continuous mode until the hand is in line with a laser; at this point, the system could lock to the pitch, and the player could use the Z-axis measurement to do pitch modulation, or introduce any other effect they desired.

5.2.6 Multi-Colored Guides

It would be desirable to have a non-monochrome set of visual frets, to aid with play. Though only red laser diodes were used for cost reasons, a system could be built which also took advantage of green laser diodes (about 100 times as expensive as red, and not nearly as long-lasting). For a cheaper multi-colored method, an LED based system may be possible, if bright enough LEDs were used and the light could be properly collimated.

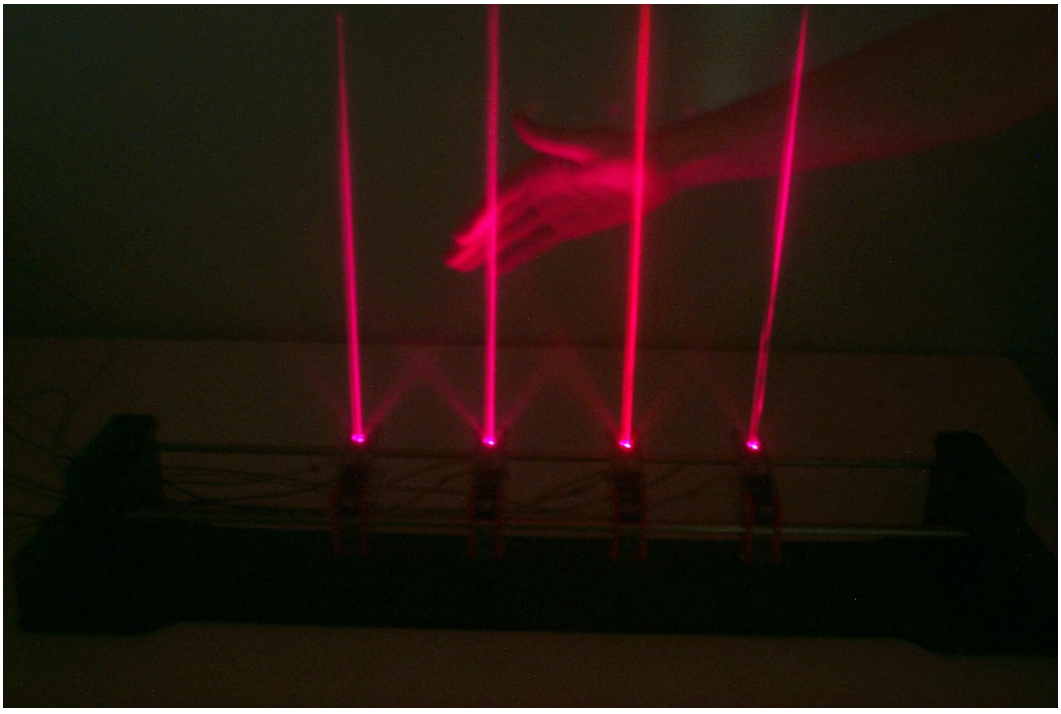
5.2.7 Visual Effects

The lasers may be turned on and off by the external computer; this has potential for interesting visual effects, such as synchronizing laser effects to musical events.

Also, a person could be taught how to play a piece of music by a computer, as the computer could turn on only the laser that corresponds to the pitch that is next to be played, making the lasers highlights for the next note. If there are continuous pitch changes in the music, the computer could control a motor and make the laser move appropriately; the player would then follow the laser with their hand, which, if tuned correctly, would create the desired sound. In this case, it would be the performer that is being used as the interface, existing between the computer and the music creating device.

These are but a few of the many possible applications that this hybrid sensing instrument would allow for. A picture of the termenova being played is shown in Figure 5-1

Figure 5-1: Termenova being played



Appendix A

Initial Design

The first design had a cable that ran the entirety of the track and was actuated by a single powerful servo motor at an extreme end of the track. Each module would be equipped with a solenoid valve, which would grip to the cable when active. Modules would move into position by selectively gripping or disengaging from the cable. During calibration, the modules would use the reflectance sensors to follow the hand, and when a module reached the position that corresponded to a specific frequency, it would disengage from the cable. The design was chosen to minimize the amount of moving parts, especially motors.

This design had several major flaws. The first and most difficult to accommodate was that the exact position of the modules could not be known, as there was no position data of any kind coming from the modules. All the modules would be moved by the same motor, sometimes many at once, which would make positioning even more difficult since the dynamics of the cable drive would be difficult to predict. Also, following the fingers of the hand would be extremely difficult; following an edge with only one reflectance sensor is nearly impossible, and even with two reflectance sensors would be difficult.

Therefore, a system was designed that encompassed an absolute positioning system for each individual module.

Appendix B

Board Design

A set of circuit boards was designed and fabricated for the modules; because the space in the modules was quite limited, two separate boards needed to be built. Additionally, because of the tight spaces and alignment issues, both boards were modeled in Solidworks to check for clearance of all parts.

There is a laser board, which contains the laser driver and synchronous demodulation circuitry, and a control/motor board, which contains the microcontroller as well as the motor module.

B.1 Laser Board

A circuit diagram of he laser board can be found in Figure B-2, and a photo can be found in Figure B-1.

The OPT101 is a linear response photodiode with incorporated transimpedance amplifier. It features an adjustable response, which was taken advantage of for this implementation. A capacitor in parallel with a resistor (with desired values obtained from the datasheet) connected between pins 2 (In-) and 5 (Out) provides external feedback to adjust for the desired DC gain. In the designed system, the DC gain was adjusted to be $\frac{1}{10}$ of the default gain of the chip, since the responsivity at the original gain resulted in the photodiode saturating in ambient light conditions.

The AD633 is a four-quadrant analog multiplier. The signal from the OPT101 is

Figure B-1: Laser Board

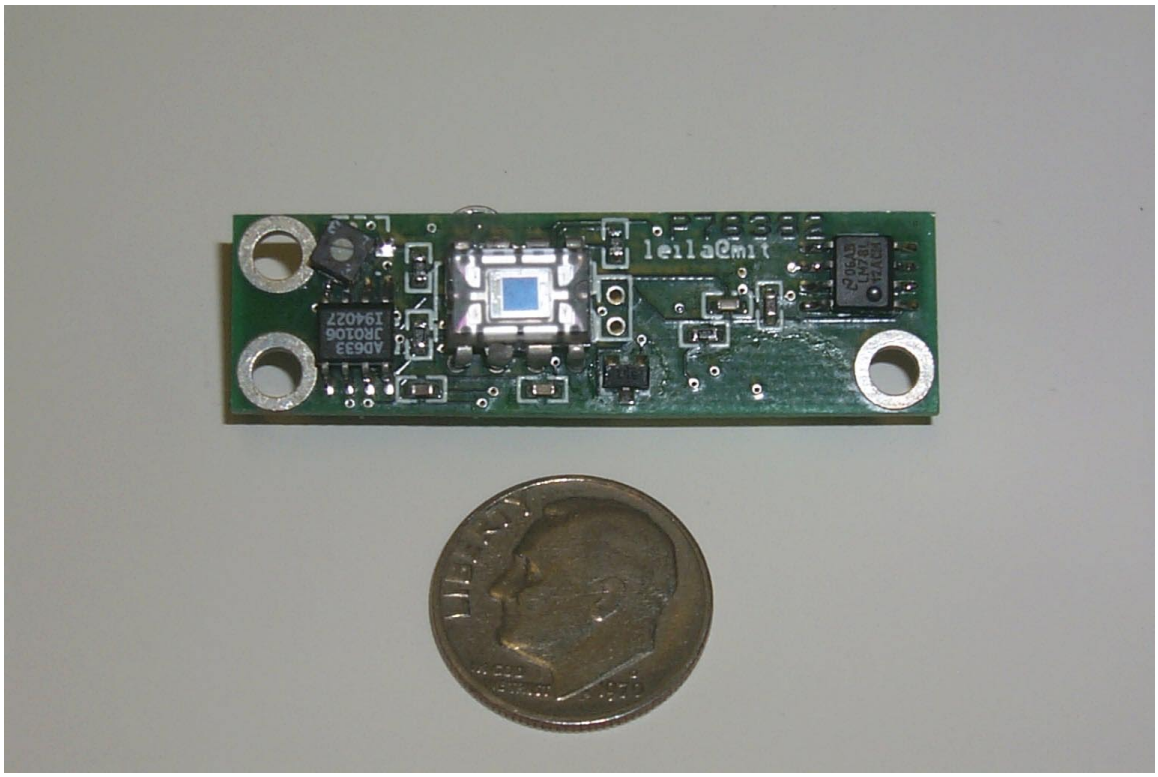
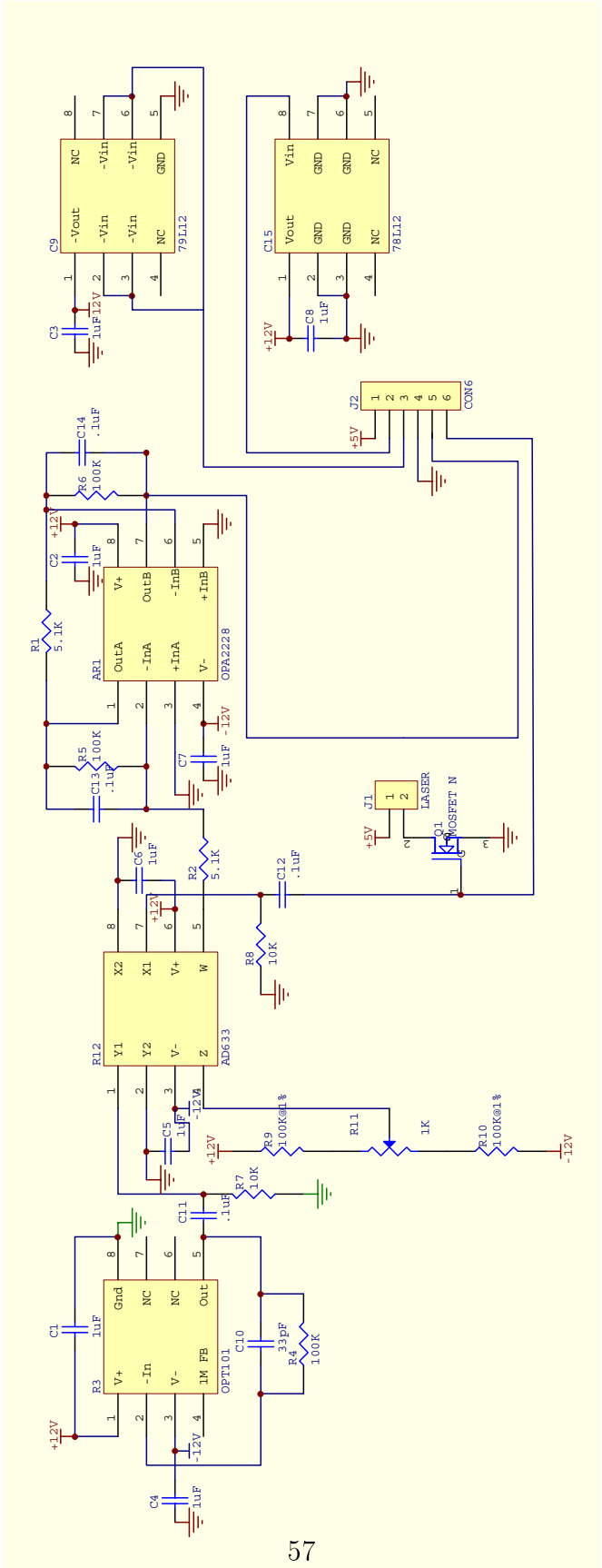


Figure B-2: Laser Board Circuit Diagram



multiplied here by the modulation signal, which was used to drive the laser diode. Both these inputs are AC-coupled with a simple high-pass filter before the multiplier stage. The output of the AD633 is:

$$W = \frac{X_1 * X_2 Y_1 * Y_2}{10V} + Z \tag{B.1}$$

where Z is another possible input to the multiplier.

The AD633 has a maximum output offset voltage of 50mV, which became significant in this design, as the output of the multiplier is amplified by a factor of 400 [1]. This means there is a possible offset of $\pm 2V$. This offset is specifically undesirable since the output of the amplifier is fed into an analog port of the PIC, which takes only positive values. A $-2V$ offset in output of the demodulator will most definitely result in a negative value being seen by the PIC. To prevent this from occurring, an adjustable trimming offset was added to the Z-input of the multiplier, to ensure that the output is positive.

The OPA2228 is a dual high precision, low offset voltage op-amp chip; it's used for both the low-pass filtering and amplification of the multiplier output. The output of the multiplier needs to scaled by a factor of about 400 to have it span a reasonable range of the PIC A/D, so two 20x stages, each low pass filtering with a 50 Hz cutoff, were implemented.

B.2 Control/Motor Board

The motor board circuit diagram is shown in Figure B-4, and a photo is shown in Figure B-3.

The PIC16F876 is described in Section 3.2.3

The L293 is a bidirectional motor driver, capable of providing up to 1A continuous current. It consists of four half-H bridges, which were paired up to form 2 full-H

[1] Part of the reason this amplification factor is so high is because of the divide-by-ten incorporated by the multiplier

Figure B-3: Motor Board

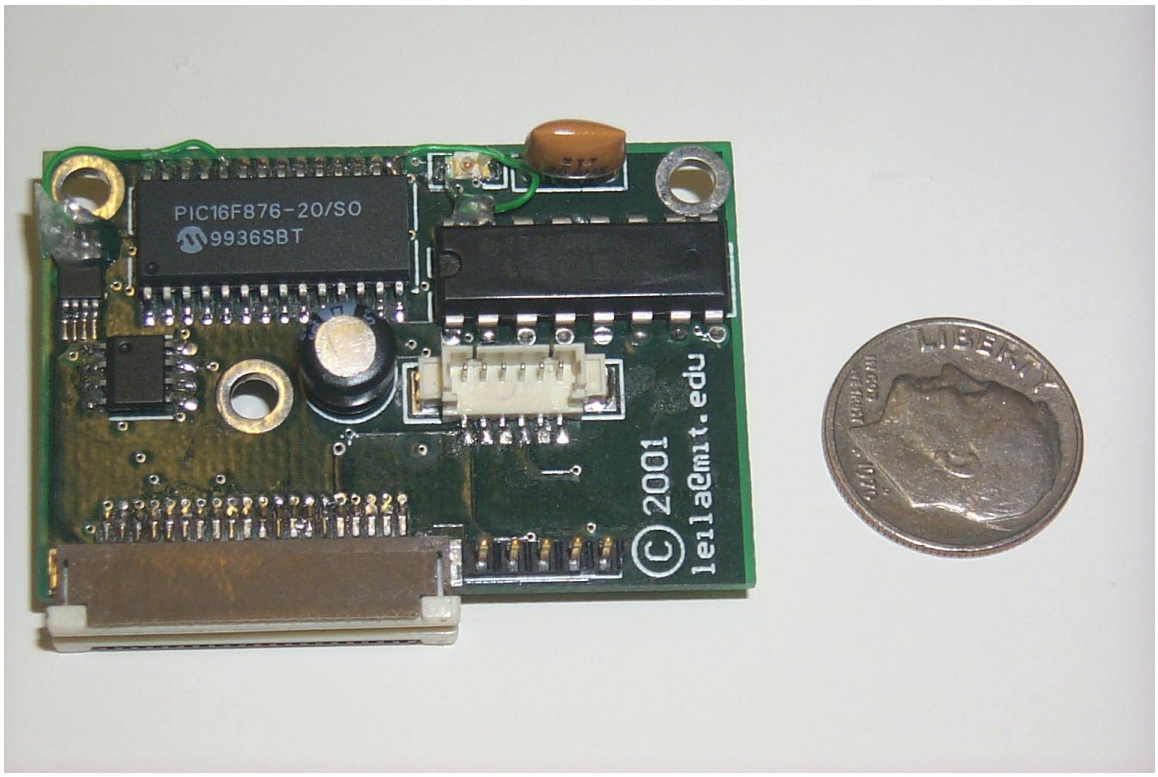
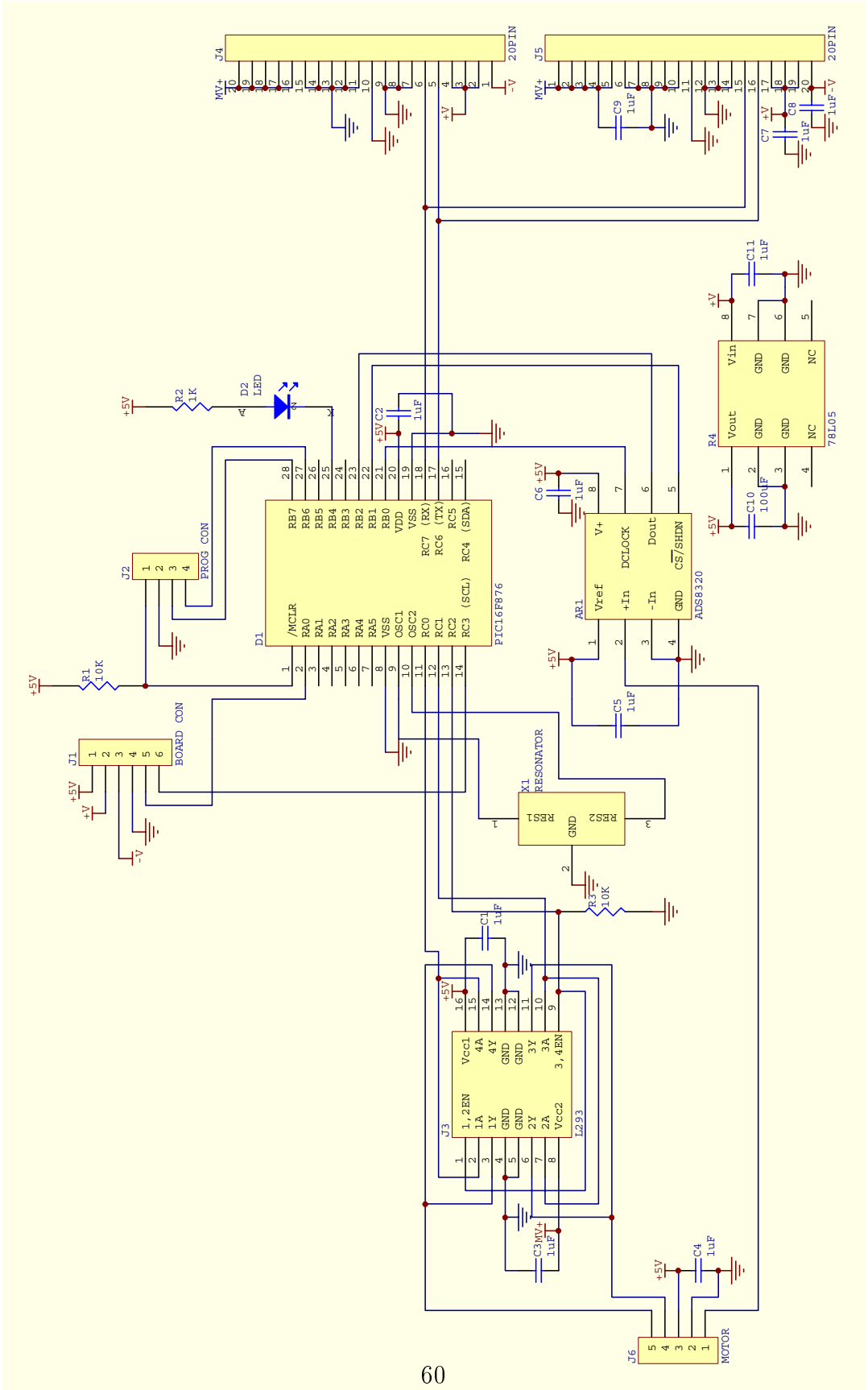


Figure B-4: Motor Board Circuit Diagram

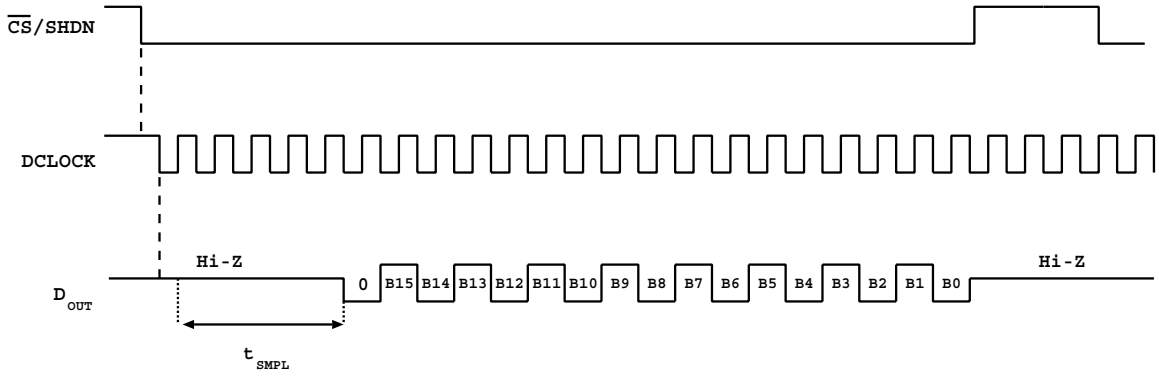


reversible drive bridges and put in parallel to maximize output current capability. PWM is performed on the enable pins of the chip.

The position data from the motor was obtained with a 20 turn 5K potentiometer, with the output set up as a voltage divider and fed to a analog to digital converter.

The ADS8320 is a 16 bit A/D with serial output. The PIC drives the A/D with a clock signal and conversion enable, and receives the data in serial format, MSB first. The timing is shown in Figure B-5.

Figure B-5: ADS8320 Timing Diagram

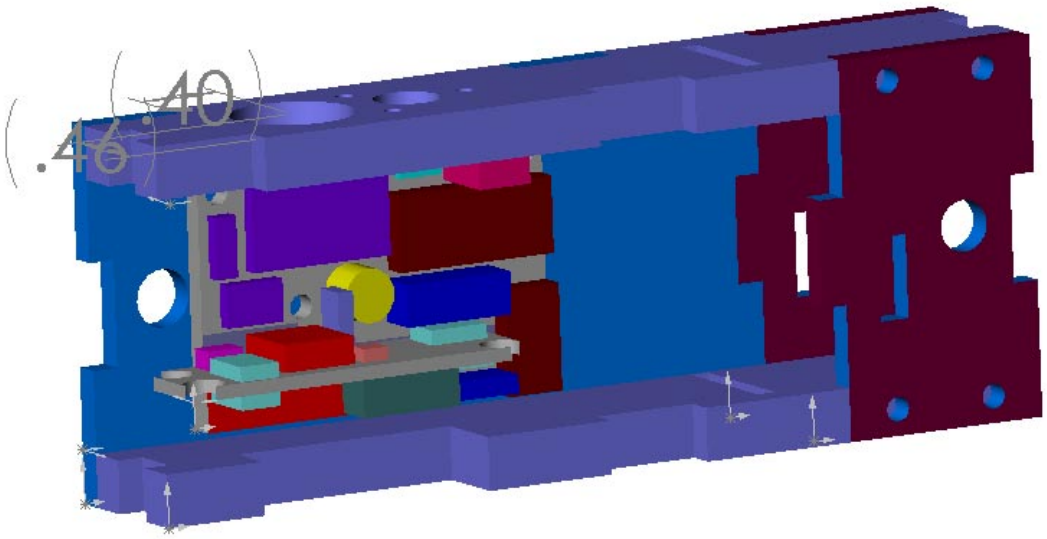


The control/motor board also contains a pair of connectors for 20 pin flex cable, which provides power to both boards and the shared serial bus. All modules are connected in parallel.

B.3 Solidworks Model

A Solidworks model of the boards was done to ensure that they maintained the tight tolerances required. The laser board needs to be properly placed so the lens and photodiode are aligned, and this would have been difficult without creating a 3-D model of the system. A 3-D view of the Solidworks model is shown in Figure B-6.

Figure B-6: Solidworks Model of Boards



Appendix C

Base and Module Design

The base and modules were both designed in Solidworks. The .dxf files used for both the lasercutting and waterjetting of all the pieces for both the modules and the base were generated from the Solidworks models.

- A snapshot of the full Solidworks assembly is shown in Figure C-1.
- A fully built module, containing boards, motor, and pot is shown in Figure C-2.
- Figure C-3 shows the module connected to the rod assembly.

Figure C-1: Solidworks Model of Full Assembly

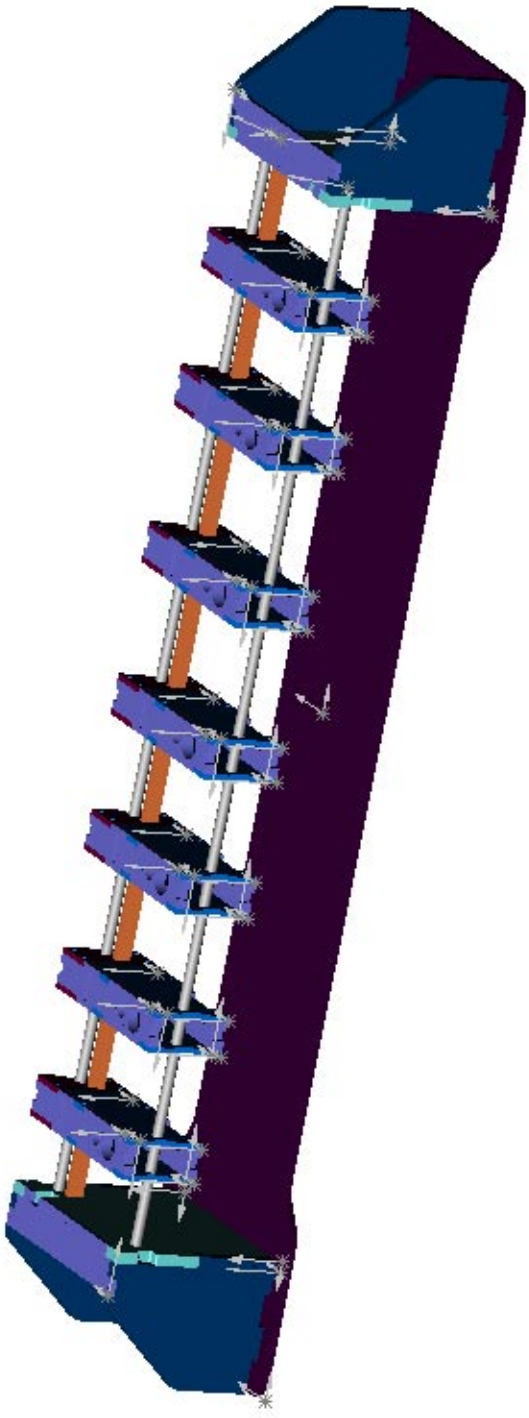


Figure C-2: Module with boards

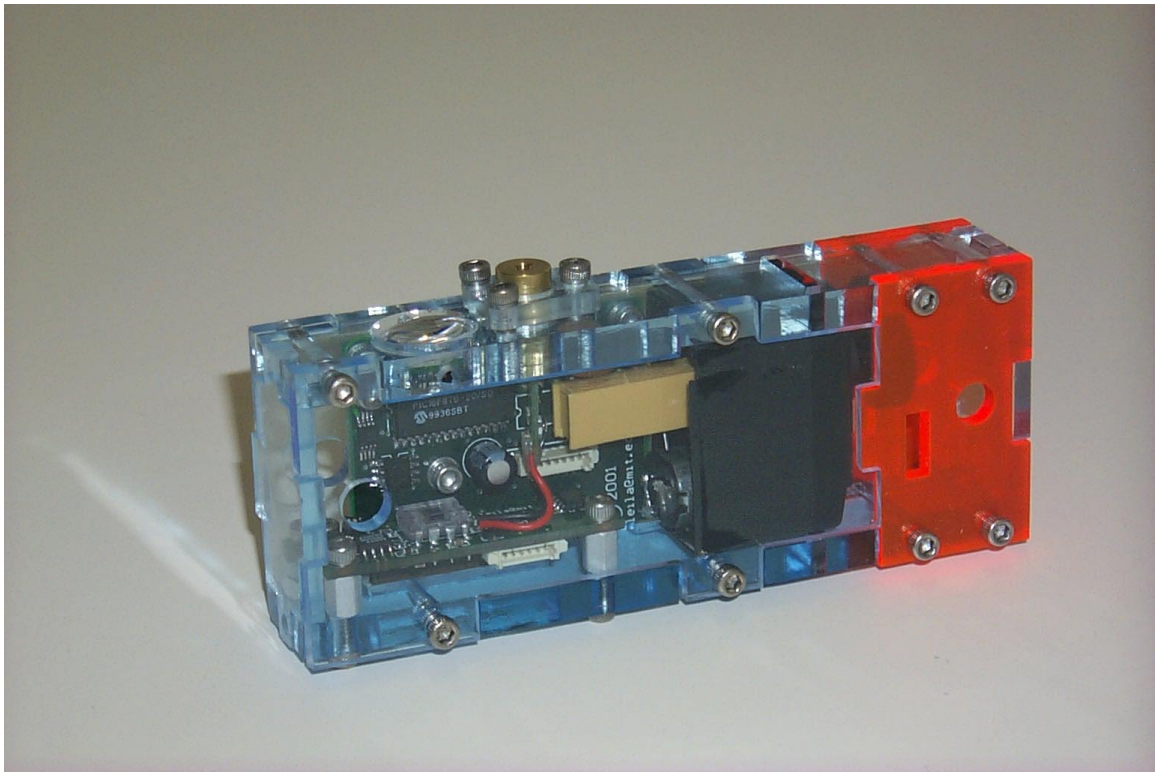
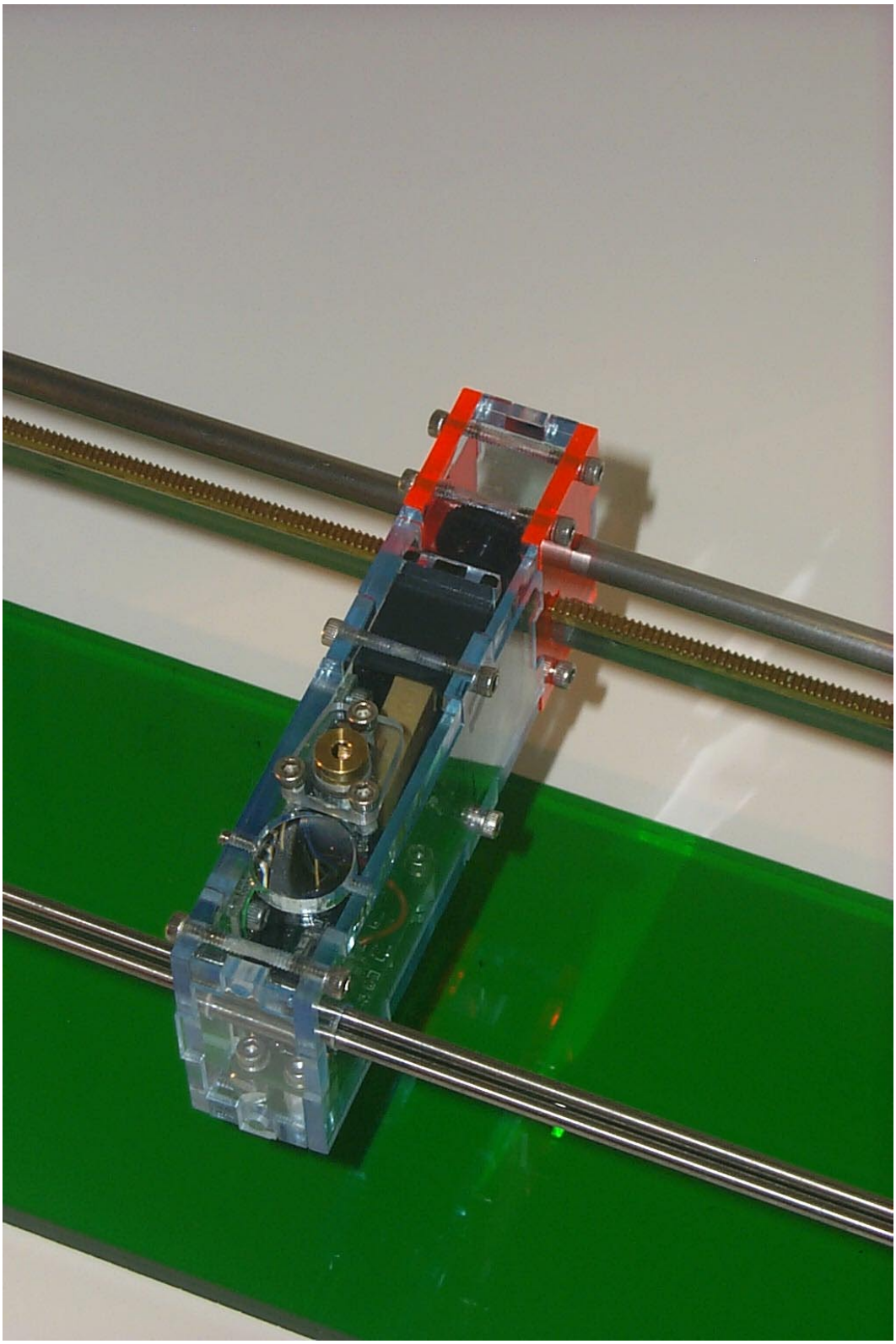


Figure C-3: Module on rails



Appendix D

Protocols and Code

D.1 Serial Interface

The serial interface consists of a common transmit line and a common receive line between the main board and each of the module boards. The main board talks to a module by sending a recognizable sequence, containing the ID of the module it wants to talk to. Since only one device should drive a line at once, the module board transmit lines are disabled until the main board sends an individual module a command that requires them to send back data; when this occurs, the modules briefly enable transmission.

In the current implementation, only the main board can initiate communication. During play, the main board polls all the modules in succession, checking to see if the modules detect an object breaking the laser beam. With only a few modules, this method is inelegant but sufficient; however, as more modules get added, the polling process will become too slow to accomodate getting detector information. In this case, an *open collector* serial bus can be implemented.

An *open collector* serial bus is employed so that a module can initiate communication without being polled by the main controller. In this scheme, instead of driving the transmit pin high and low (as in standard RS-232 communications) the module drives the pin low and high-impedance. The main board receive line is connected to a pullup resistor, so the received signal looks like normal RS-232. With this setup,

the modules can simply send the main board data when an event occurs, instead of waiting to be polled, as a collision with another module is not catastrophic; the modules poll the line as they send, and if they notice that the line is getting pulled low when it shouldn't be, they know that another module is trying to communicate and can act appropriately. This protocol also frees up the main board from polling the modules constantly.

D.2 Code

All the module code was written in assembly, with the files as follows:

nova.asm main file

mdh.inc math functions

mama.inc register and pin assignments

motor.asm servo control code

adc.asm external ADC code

uart.asm uart initialization code

timing.asm delay routines

ser.asm serial testing code

serial.asm serial protocol for reception

baby.asm serial transmission functions

laser.asm laser board control code

The main board code nwas written in C, to take advantage of the built-in software UART code. All the code for the main board is located in mom.c

```
#include p16f876.inc
#include mdh.inc
#include mama.inc

; reset vector and the interrupt vector.

        ORG     0000
        GOTO    main
        ORG     0004
        GOTO    isr

main:
        CALL    init_hardware           ; Initialize the hardware
CALL init_state
CALL  uart_init

        BSF     INTCON, GIE             ; Switch everything on
BSF INTCON, PEIE
  CALL flash

loop: ;waiting for serial command
CALL rwait
CALL serial_rx
GOTO loop


init_hardware:
        CLRF    STATUS
CLRF PORTA
CLRF PORTB
CLRF PORTC
        BSF     STATUS, RP0

; default input/output
; configuratons for the pins and initialize output pins.
MOVLW 0x0E
MOVWF ADCON1 ;set A/D for channel 0 analog
MOVLW BIT(0)
MOVWF TRISA ;set PORT A as outputs except A0

MOVLW 0x05 ;all port b outputs except RB2
MOVWF TRISB
        MOVLW   BIT(UART_RX) | BIT(UART_TX) | BIT(0)
        MOVWF   TRISC   ; Make PORTC output except
; serial pins + motorboard bad pin
```

```
;interrupts
MOVLW 0xFF
MOVWF PR2 ;set pwm period to 1.2kHz
BSF PIE1,TMR2IE ;enable T2
BSF PIE1,TMR1IE ;enable T1 interrupt
        BCF     STATUS, RP0     ; Switch back to first page
;timer2
MOVLW 0x0F
MOVWF CCP1CON ;set CCP1CON for PWM mode
MOVLW 0x07
MOVWF T2CON ;set T2 prescale to 16
CLRF CCPR1L ;set duty cycle to 0
;timer1
MOVLW 0xFE
MOVWF TMR1H
CLRF TMR1L ;set timer1 freq = 9.6 kHz
MOVLW BIT(TMR1ON) ;enable timer 1
MOVWF T1CON ;T1 prescale 1, internal clock
;adc
MOVLW BIT(ADCS1)
MOVWF ADCON0 ;A/D for Fosc/32, channel0
BSF ADCON0,0x00 ;turn on A/D module

        CLRF    STATUS
      RETURN          ; And we're done

init_state: ;initialize serial registers
CLRF STATE
CLRF SERSTATE
CLRF FLAGS
CLRF ALARMS
RETURN

#include motor.asm ;PD control
#include adc.asm ;external ADC functions
#include uart.asm ;uart initialization
#include timing.asm ;delay code
#include isr.asm ;interrupts
#include ser.asm ;debugging code
#include serial.asm ;serial protocol
#include baby.asm ;module responses to mainboard
#include laser.asm ;demodulator code

END
```

```
; mdh.inc - Personal handy routines and such
; Copyright 2000 Matthew D. Hancher

; note to self: sub R, W : { Z=(R==W); C=(R>=W); }

 ;; Convert from bit number to bitmask (i.e. 3 -> 0x80)
#define BIT(i) (1<<(i))

DDUMPL MACRO c
#ifdef DEBUG
MOVLW c
CALL swait
MOVWF TXREG
#endif
ENDM

DDUMPF MACRO r
#ifdef DEBUG
MOVF r, W
CALL bindump
#endif
ENDM

DSCRLF MACRO
#ifdef DEBUG
CALL scrlf
#endif
ENDM

 ;; Move a 16-bit literal into two registers
MOVLF16 MACRO r, l
MOVLW (l>>8)
MOVWF r
MOVLW (l&0xFF)
MOVWF r+1
ENDM

 ;; Compare two 16-bit numbers, setting C and Z appropriately
 ;; (big-endian) { Z=(a==b); C=(a>=b); }
CMP16 MACRO a,b
LOCAL done
MOVF b, W
SUBWF a, W
BTFSS STATUS, Z
GOTO done
```

```
MOVF b+1, W
SUBWF a+1, W
done:
ENDM

 ;; Subtract a 16-bit literal from a 16-bit number, railing at zero
 ;; { r=(r<l)?0:r-l; }
SUBL16Z MACRO   r,l
        LOCAL   nobrw, zero, done
        MOVLW   (l&0xFF)
        SUBWF   r+1, F
        BTFSC   STATUS, C
        GOTO    nobrw
        MOVLW   1
        SUBWF   r, F
        BTFSS   STATUS, C
        GOTO    zero
nobrw:  MOVLW   (l>>8)
        SUBWF   r, F
        BTFSC   STATUS, C
        GOTO    done
zero:   CLRF    r
        CLRF    r+1
done:
        ENDM

 ;; Subtract one 16-bit number from another, railing at zero
 ;; { a=(a<b)?0:a-b; }
SUB16Z  MACRO   a,b
        LOCAL   nobrw, zero, done
MOVF b+1, W
        SUBWF   a+1, F
        BTFSC   STATUS, C
        GOTO    nobrw
        MOVLW   1
        SUBWF   a, F
        BTFSS   STATUS, C
        GOTO    zero
nobrw:  MOVF b, W
        SUBWF   a, F
        BTFSC   STATUS, C
        GOTO    done
zero:   CLRF    a
        CLRF    a+1
done:
```

```
        ENDM

 ;; Add a 16-bit literal to a 16-bit number, railing at wrap (0xFFFF)
ADDL16Z MACRO   r,l
        LOCAL   nocry, wrap, done
        MOVLW   (l&0xFF)
        ADDWF   r+1, F
        BTFSS   STATUS, C
        GOTO    nocry
        MOVLW   1
        ADDWF   r, F
        BTFSC   STATUS, C
        GOTO    wrap
nocry:  MOVLW   (l>>8)
        ADDWF   r, F
        BTFSS   STATUS, C
        GOTO    done
wrap:   MOVLW 0xFF
MOVWF r
MOVWF r+1
done:
        ENDM

 ;; Add one 16-bit number to another, railing at wrap (0xFFFF)
ADD16Z MACRO a, b
LOCAL nocry, wrap, done
MOVF b+1, W
ADDWF a+1, F
BTFSS STATUS, C
GOTO nocry
MOVLW 1
ADDWF a, F
BTFSC STATUS, C
GOTO wrap
nocry: MOVF b, W
ADDWF a, F
BTFSS STATUS, C
GOTO done
wrap: MOVLW 0xFF
MOVWF a
MOVWF a+1
done:
ENDM

 ;; Multipy two 16-bit numbers, generating a 32-bit result
```

```
 ;; (big-endian) *** NB: Smashes SYSREG0
MUL16 MACRO r, a, b
LOCAL mloop
LOCAL mnoadd
CLRF r
CLRF r+1
CLRF r+2
CLRF r+3
MOVLW 0x10
MOVWF SYSREG0
mloop:
BCF STATUS, C
BTFSC b+1, 0
BSF STATUS, C
RRF b, F
RRF b+1, F
BTFSS STATUS, C
GOTO mnoadd
MOVF a, W
ADDWF r, F
BTFSC STATUS, C
INCF r+3, F
MOVF a+1, W
ADDWF r+1, F
MOVLW 0x01
BTFSC STATUS, C
ADDWF r, F
BTFSC STATUS, C
ADDWF r+3, F
mnoadd:
BCF STATUS, C
BTFSC r+3, 0
BSF STATUS, C
RRF r, F
RRF r+1, F
RRF r+2, F
RRF r+3, F
DECF SYSREG0, F
BTFSS STATUS, Z
GOTO mloop
ENDM

 ;; Multipy two 16-bit numbers, generating a 32-bit result
 ;; (big-endian) *** NB: Smashes SYSREG0
MUL16L MACRO r, b, l
```

```
LOCAL mloop
LOCAL mnoadd
CLRF r
CLRF r+1
CLRF r+2
CLRF r+3
MOVLW 0x10
MOVWF SYSREG0
mloop:
BCF STATUS, C
BTFSC b+1, 0
BSF STATUS, C
RRF b, F
RRF b+1, F
BTFSS STATUS, C
GOTO mnoadd
MOVLW (l>>8)
ADDWF r, F
BTFSC STATUS, C
INCF r+3, F
MOVLW (l&0xFF)
ADDWF r+1, F
MOVLW 0x01
BTFSC STATUS, C
ADDWF r, F
BTFSC STATUS, C
ADDWF r+3, F
mnoadd:
BCF STATUS, C
BTFSC r+3, 0
BSF STATUS, C
RRF r, F
RRF r+1, F
RRF r+2, F
RRF r+3, F
DECF SYSREG0, F
BTFSS STATUS, Z
GOTO mloop
ENDM
```

```
;mama.inc
;register and pin assignments

MOD_ADD EQU 0x04; module address
BLOCK EQU 0x00; detect flag - high if beam being blocked

;timing registers
deltron0 EQU 0x20
deltron1 EQU 0x21
deltron2 EQU 0x25

temp EQU 0x22
;TEMP0 EQU 0x23
;TEMP1 EQU 0x24

conv_reg EQU 0x26
conv_nib EQU 0x27

;motor control
DESHI EQU 0x28
DESLO EQU 0x29
POSHI EQU 0x30
POSLO EQU 0x31
ERR0 EQU 0x32
ERRHI EQU 0x33
ERRLO EQU 0x34
IERRHI EQU 0x35
IERRLO EQU 0x36
MTEMP EQU 0x37
DETECT_AMP EQU 0x38; amplitude of reflected value
THRESH EQU 0x39;  threshhold value


;; SPI & A/D registers

SPIHI EQU 0x40
SPILO EQU 0x41
SPIREG EQU 0x42
ANAVHI EQU 0x43
ANAHI EQU 0x44
ANALO EQU 0x45
ANAREG EQU 0x46

 ;; Generic local registers
REG0 EQU 0x50
```

```
REG1 EQU 0x51
REG2 EQU 0x52
REG3 EQU 0x53
REG4 EQU 0x54
REG5 EQU 0x55
REG6 EQU 0x56
REG7 EQU 0x57
REG8 EQU 0x58
REG9 EQU 0x59
REGA EQU 0x5A
REGB EQU 0x5B
REGC EQU 0x5C
REGD EQU 0x5D
REGE EQU 0x5E
REGF EQU 0x5F

 ;; Serial port registers
SERSTATE EQU 0x60
STXCKSUM EQU 0x61
STXPTR EQU 0x62
STXLEN EQU 0x63
SRXLEN EQU 0x64
SREG0 EQU 0x65
SREG1 EQU 0x66
SREG2 EQU 0x67
SREG3 EQU 0x68
SREG4 EQU 0x69
SRX_BUFFER EQU 0x110
STX_BUFFER EQU 0x190
SRX_BUFSIZE EQU 0x60
STX_BUFSIZE EQU 0x60

 ;; Global variables
STATE EQU 0x70
FLAGS EQU 0x71
ALARMS EQU 0x72
SYSREG0 EQU 0x78
SYSREG1 EQU 0x79
SYSREG2 EQU 0x7A
SYSREG3 EQU 0x7B
SYSREG4 EQU 0x7C
STACK_FSR EQU 0x7D
STACK_STATUS EQU 0x7E
STACK_W EQU 0x7F
```

```
 ;; Serial port constants
SS_RX EQU 0 ; Currently receiving a packet
SS_TX EQU 1 ; Currently transmitting a packet
SS_ESC EQU 2 ; Last character was ESC
SS_TXSTARTED EQU 3 ; Packet header has been transmitted
SS_TXFINISH EQU 4 ; time to transmit packet terminator
SER_ESC EQU 0xAA
SER_START EQU 0x00
SER_END EQU 0x01




;PORT A assignments
DEMOD EQU 0x00

;PORT B assignments
PIN_SHDN EQU 0x01
PIN_DATA EQU 0x02
PIN_DCLK EQU 0x03
LED EQU 0x04

;PORT_C assignments
MOTA EQU 0x04
MOTB EQU 0x01
MOTEN EQU 0x02
LASER EQU 0x03
;serial pins
UART_TX EQU 0x06
UART_RX EQU 0x07
```

```
;;motor.asm
;;contains motor control code

motorf:
BCF PORTC,MOTA
BSF PORTC,MOTB
RETURN

motorb:
BSF PORTC,MOTA
BCF PORTC,MOTB
RETURN

stopmotor:
CLRF CCPR1L
RETURN

gocontrol:
CALL propint
MOVF CCPR1L,F
BTFSS STATUS,Z
GOTO gocontrol
RETURN

;proportional control
prop:
CALL analog_read
MOVF POSHI,W
MOVWF DESHI
MOVF POSLO,W
MOVWF DESLO
CMP16 ANAHI,DESHI
BTFSC STATUS,C ;if no borrow
goto propback ;desired position is behind us (lower)
CALL motorf
SUB16Z DESHI,ANAHI ;else desired position is in front of us
MOVF DESHI,W
MOVWF ERRHI
MOVF DESLO,W
goto propforw
propback: CALL motorb
SUB16Z ANAHI,DESHI
MOVF ANAHI,W
MOVWF ERRHI
MOVF ANALO,W
```

```
propforw: MOVWF ERRLO
MOVF ERRLO,W
RETURN

;proportional + integral control
;Kp = 16, Ki = 1/256
propint:
CALL prop
BCF STATUS,C
RLF ERRLO,F
RLF ERRHI,F
RLF ERRLO,F
RLF ERRHI,F
RLF ERRLO,F
RLF ERRHI,F
RLF ERRLO,F
RLF ERRHI,F ;add Kp of 16
MOVF ERRHI,F
BTFSC STATUS,Z ;if the error is <FF
goto smallpi ;add int
railit: MOVLW 0xFF ;else rail it
MOVWF CCPR1L
RETURN
smallpi: ADD16Z IERRHI,ERRHI ;add to int error
ADD16Z ERRHI,IERRHI
MOVF ERRLO,W
MOVWF CCPR1L ;else use it as the PWM value
MOVF CCPR1L,F
BTFSS STATUS,Z
goto notzero
DECFSZ MTEMP,F
RETURN
CLRF IERRHI
CLRF IERRLO
MOVLW 0x0F
MOVWF MTEMP
RETURN
notzero: SUBLW 0x3F
BTFSC STATUS,C
RETURN
MOVLW 0x3F
MOVWF CCPR1L ;minimum PWM value
piend: RETURN

encoder:
```

```
RRF ANAHI,F ;convert 16 bit analog read to
RRF ANALO,F ;12 bit motor position format 0x0HHH
RRF ANAHI,F
RRF ANALO,F
RRF ANAHI,F
RRF ANALO,F
RRF ANAHI,F
RRF ANALO,F
MOVLW 0x0F
ANDWF ANAHI,F
RETURN
```

```
;;adc.asm
;;functions for the external ADC

;dump ADC value onto serial
show_ADC:
CALL analog_read
MOVF SPIHI,W
CALL conv_byte
MOVF SPILO,W
CALL conv_byte
RETURN

 ; SPI_WORD
 ; Read a value from the A/D converter (this also initiates an A/D
 ; conversion with this chip) and leave the result in SPIHI:SPILO.
spi_word:
        MOVLW   0x16
        MOVWF   SPIREG
        BCF     PORTB, PIN_DCLK
        BCF     PORTB, PIN_SHDN
spi_loop:
        BCF     PORTB, PIN_DCLK
        BCF     STATUS, C
        BTFSC   PORTB, PIN_DATA
        BSF     STATUS, C
        RLF     SPILO, F
        RLF     SPIHI, F
        BSF     PORTB, PIN_DCLK
        DECFSZ  SPIREG, F
        GOTO    spi_loop
        BSF     PORTB, PIN_SHDN
        RETURN


 ; ANALOG_READ
 ; Perform an A/D conversion with 4x oversampling,
 ; leaving the result in ANAHI:ANALO
analog_read:
CLRF ANALO
CLRF ANAHI
CLRF ANAVHI
MOVLW 0x10
MOVWF ANAREG
ana_loop:
CALL spi_word
```

```
MOVF SPIHI, W
ADDWF ANAHI, F
BTFSC STATUS, C
INCF ANAVHI, F
MOVF SPILO, W
ADDWF ANALO, F
MOVLW 1
BTFSC STATUS, C
ADDWF ANAHI, F
BTFSC STATUS, C
ADDWF ANAVHI, F
DECFSZ ANAREG, F
GOTO ana_loop
RRF ANAVHI, F
RRF ANAHI, F
RRF ANALO, F
RRF ANAVHI, F
RRF ANAHI, F
RRF ANALO, F
RRF ANAVHI, F
RRF ANAHI, F
RRF ANALO, F
RRF ANAVHI, F
RRF ANAHI, F
RRF ANALO, F
RETURN
```

```
;; uart.asm
;; asynchronous serial stuff

; initialize SPBRG register for the appropriate baud rate
; set bit BRGH for high speed

; 1200 (NO BRGH):
;#define SPBRG_VAL 0xFF

; 9600 (BRGH):
;#define SPBRG_VAL 0x81

; 19.2K (BRGH):
;#define SPBRG_VAL 0x40

; 38.4K (BRGH):
#define SPBRG_VAL 0x20

; 56K (BRGH):
;#define SPBRG_VAL 0x15

; 115K (BRGH):
;#define SPBRG_VAL 0x0A

#define USE_BRGH

 ;; Initialize the UART
uart_init:
        MOVLW   BIT(RP0)
        MOVWF   STATUS
        MOVLW   SPBRG_VAL
        MOVWF   SPBRG           ; Set baud rate

BCF PIE1, RCIE ;disable interrupt on receive
BCF PIE1,TXIE ;disable interrupt on transmit

        MOVLW   0x04

; enable the serial port by clearing SYNC bit
; set pin SPEN

        MOVWF   TXSTA           ; set BRGH,don't enable transmission

        BCF     STATUS, RP0
        MOVLW   0x90
```

```
        MOVWF   RCSTA           ; Enable reception
        RETURN

rwait:
BTFSS RCSTA,OERR ; check for error
GOTO no_rerr
BCF RCSTA,CREN
BSF RCSTA,CREN ;clear error
no_rerr: BTFSS PIR1, RCIF ; check for received flag
GOTO rwait

RETURN


swait:
        BTFSS   PIR1, TXIF ; make sure transmit register is empty
        GOTO    swait
        RETURN

send: CALL swait
MOVWF TXREG
RETURN
```

```
;;timing.asm
;;routines for delays

delay_ms:
MOVLW  deltron2
MOVWF FSR
MOVF 00,W
BTFSC STATUS,Z
GOTO aaa
eee: MOVLW 06
MOVWF deltron1
ccc: CLRF deltron0
bbb: DECFSZ deltron0,F
GOTO bbb
DECFSZ deltron1,F
GOTO ccc
MOVLW 77
MOVWF deltron0
ddd: DECFSZ deltron0,F
GOTO ddd
DECFSZ 00,F
GOTO eee
aaa: RETLW 00

delay_s:
MOVLW 0xFA
MOVWF 0x25
CALL delay_ms
MOVLW 0xFA
MOVWF 0x25
CALL delay_ms
MOVLW 0xFA
MOVWF 0x25
CALL delay_ms
MOVLW 0xFA
MOVWF 0x25
CALL delay_ms
RETURN

delayt:
MOVLW 0x7F
MOVWF 25
CALL delay_ms
RETURN
```

```
;;ser.asm
;;random serial routines and test code

flash:
BCF  PORTB,LED ;led on
CALL  delay_s
BSF PORTB,LED ;led off
CALL  delay_s
RETURN

check:
BSF STATUS,RP0
BSF TXSTA,TXEN ;enable transmission
BCF STATUS,RP0
CALL delay_ms
MOVLW  'l'  ;l led
SUBWF  RCREG,W
BTFSC  STATUS, Z
CALL led
MOVLW  'm'  ;m led
SUBWF  RCREG,W
BTFSC  STATUS, Z
CALL flash
MOVLW  0x68 ;h hi
SUBWF  RCREG,W
BTFSC  STATUS,Z
CALL hello
MOVLW  0x65 ;e error
SUBWF  RCREG,W
BTFSC  STATUS, Z
CALL rerror
MOVLW '?';?  display commands
SUBWF RCREG,W
BTFSC STATUS,Z
CALL help
MOVLW  'p';m motor
SUBWF  RCREG,W
BTFSC  STATUS, Z
CALL setmotor
MOVLW  'a';a ADC
SUBWF  RCREG,W
BTFSC  STATUS, Z
CALL show_ADC
MOVF RCREG,W
CALL swait
```

```
BSF STATUS,RP0
BCF TXSTA,TXEN ;disable transmission
BCF STATUS,RP0
RETURN

rerror:
MOVLW ASC_e ;e
CALL send
MOVLW 0x72 ;r
CALL send
MOVLW 0x72 ;r
CALL send
MOVLW 0x6F ;o
CALL send
MOVLW 0x72 ;r
CALL send
MOVLW ASC_CR
CALL send
MOVLW ASC_LF
CALL send
RETURN

led:
BSF PORTB,LED
MOVLW 0x7F ; 127 ms
MOVWF 25
CALL delay_ms
BCF PORTB,LED
MOVLW 0x7F
MOVWF 25
CALL delay_ms
BSF PORTB,LED
MOVLW 0xFF ;255 ms
MOVWF 25
CALL delay_ms
BCF PORTB,LED
MOVLW 0xFF
MOVWF 25
CALL delay_ms
BSF PORTB,LED
return

help:
MOVLW 'h'
CALL  send
```

```
MOVLW ' '
CALL  send
MOVLW 'h'
CALL  send
MOVLW 'i'
CALL  send
CALL  next_line
MOVLW 'l'
CALL  send
MOVLW ' '
CALL  send
MOVLW 'l'
CALL  send
MOVLW 'e'
CALL  send
MOVLW 'd'
CALL  send
CALL  next_line
RETURN


hello:
MOVLW 'h'
CALL send
MOVLW 'i'
CALL send
MOVLW 0x0A ;line feed
CALL send
MOVLW ASC_CR
CALL send
RETURN


setmotor:
MOVLW 'p'
CALL send
MOVLW 'o'
CALL send
MOVLW 's'
CALL send
CALL rwait
MOVF RCREG,W
MOVWF TXREG ;echo
CALL  convert ;get first nibble, convert from ascii
MOVWF POSHI ;put into low nibble of POSHI
```

```
CALL rwait
MOVF RCREG,W
MOVWF TXREG
CALL convert
MOVWF POSLO ;put into high nibble of POSLO
SWAPF POSLO,F
CALL rwait
MOVF RCREG,W
MOVWF TXREG
CALL convert ;put into low nibble of POSLO
ADDWF POSLO,F ;now POSHI:POSLO from 0x000-0xFFF
CALL next_line
RETURN

next_line:
MOVLW ASC_CR
CALL  send
MOVLW ASC_LF
CALL  send
RETURN

convert:
MOVWF conv_reg
MOVLW 0x3A
SUBWF conv_reg, W
BTFSS STATUS,C
goto conv1
ADDLW 0x03
goto conv2
conv1: MOVLW 0x30
SUBWF conv_reg,W
conv2: ANDLW 0x0F
RETURN

conv_to:
MOVWF conv_reg
MOVLW 0x0A
SUBWF conv_reg,W
BTFSC STATUS,C
ADDLW 0x07
ADDLW 0x3A
CALL send
RETURN

conv_byte:
```

```
MOVWF conv_nib
SWAPF conv_nib,F
MOVF conv_nib,W ;got top byte
ANDLW 0x0F
CALL conv_to
SWAPF conv_nib,F
MOVF conv_nib,W ;got bottom byte
ANDLW 0x0F
CALL conv_to
RETURN
```

```
 ;;
 ;;  SERIAL PROTOCOL CODE
 ;;
 ;;

 ;;
 ;;  Serial receive routine called by master ISR
 ;;  (figure out what do do with each incoming byte)
 ;;

serial_rx:
MOVF RCREG, W ; Let's see what we've got
MOVWF SREG0
BTFSC RCSTA, FERR ; If there was an error or we are
GOTO srx_punt ; transmitting, punt this byte
BTFSC RCSTA, OERR
GOTO srx_punt
BTFSC SERSTATE, SS_TX
GOTO srx_punt
BTFSC SERSTATE, SS_ESC ; Is this character escaped?
GOTO srx_ctrl
MOVLW SER_ESC ; Is this an escape character?
SUBWF SREG0, W
BTFSC STATUS, Z
GOTO srx_esc
MOVLW 'm'; testing
SUBWF SREG0, W
BTFSC STATUS, Z
CALL flash

srx_char: ; Toss a character in the buffer
BTFSS SERSTATE, SS_RX ; Are we in fact receiving a packet?
RETURN
; MOVLW SRX_BUFSIZE
; SUBWF SRXLEN, W
; BTFSC STATUS, Z
; GOTO srxc_overrun ;if buffer is too huge, get out
MOVLW SRX_BUFFER % 0x100
ADDWF SRXLEN, W
BSF STATUS, IRP
MOVWF FSR
MOVF SREG0, W
MOVWF INDF ;move byte into buffer
INCF SRXLEN, F ;increment length of packet
BCF STATUS,IRP
```

```
RETURN

srxc_overrun: ; This packet is too large; best just to give up
BCF SERSTATE, SS_RX
RETURN

srx_start: ; Reset for a new packet
CLRF SRXLEN
BSF SERSTATE, SS_RX
RETURN

srx_end: ; Process a received packet
BCF SERSTATE, SS_RX
MOVF SRXLEN, W ; Ignore null packets
BTFSC STATUS, Z
RETURN
MOVWF SREG1
MOVLW SRX_BUFFER % 0x100
MOVWF FSR
BSF STATUS, IRP
MOVLW 0

srx_cksum:
XORWF INDF, W
INCF FSR, F
DECFSZ SREG1, F
GOTO srx_cksum
ADDLW 0 ; Ignore invalid packets
BTFSS STATUS, Z
RETURN
DECF SRXLEN, F ; Strip off checksum
GOTO process_packet

srx_esc: ; This is an escape character
BSF SERSTATE, SS_ESC
RETURN

srx_ctrl: ; This character is escaped
BCF SERSTATE, SS_ESC
MOVLW SER_START ; Is this a start sequence?
SUBWF SREG0, W
BTFSC STATUS, Z
GOTO srx_start
;; include check for ESC char
MOVLW SER_ESC ; Is this an escape character?
```

```
SUBWF SREG0, W
BTFSC STATUS, Z
GOTO srx_char ; if so, treat as literal value
MOVLW SER_END ; Is this an end sequence?
SUBWF SREG0, W
BTFSC STATUS, Z
GOTO srx_end
; BTFSS SERSTATE, SS_RX ; Are we mid-packet?
; GOTO srx_punt
GOTO srx_char ; Escaping other characters does nothing
RETURN

srx_punt:
BCF RCSTA, CREN
BSF RCSTA, CREN ; Tidy up and get the hell out
BCF SERSTATE, SS_ESC
CLRF SERSTATE
RETURN

process_packet:
MOVLW SRX_BUFFER % 0x100
MOVWF FSR
BSF STATUS, IRP
MOVLW MOD_ADD ;module address
SUBWF INDF, W
BTFSS STATUS, Z ;check to see if match
RETURN ;if not, then it's not for me!
INCF FSR, F ;next byte
MOVLW 0x00 ;state update
SUBWF INDF, W
BTFSC STATUS, Z
GOTO mod_state
MOVLW 0x01 ;calibration position
SUBWF INDF, W
BTFSC STATUS, Z
GOTO calib_pos
MOVLW 0x02 ;check for break, then send pos
SUBWF INDF, W
BTFSC STATUS, Z
GOTO break_pos
MOVLW 0x03 ;send detector data
SUBWF INDF, W
BTFSC STATUS, Z
GOTO mamatalk
MOVLW 0x04 ;flash led
```

```
SUBWF INDF, W
BTFSC STATUS, Z
GOTO test_led1
RETURN
```

```
;; baby.asm
;; module board responses to main board commands

mamatalk:
CLRF  STATUS
BSF STATUS,RP0
BSF TXSTA,TXEN ;enable transmission
BCF STATUS,RP0
MOVLW 'x';garbage byte
CALL send
MOVLW 0xFF ;garbage byte
CALL send
MOVLW 's';start byte
CALL send
CALL get_laser ;get detector state
CALL send
MOVLW 'X';garbage byte
CALL send
CALL swait
BSF STATUS,RP0
BCF TXSTA,TXEN ;disable transmission
BCF STATUS,RP0
return

test_led1:
CLRF STATUS
BCF  PORTB,LED ;led on
CALL  delayt
BSF PORTB,LED ;led off
CALL delayt
RETURN

;move to position specified by main board
calib_pos:
BCF STATUS,RP0 ; make sure on first page
INCF FSR,F ; check next byte
MOVF INDF,W
MOVWF POSHI ; chuck into hi motor position byte
INCF FSR,F ; check next byte
MOVF INDF,W
MOVWF POSLO ; chuck into lo motor position byte
CALL dis_laser
CALL gocontrol
CALL en_laser
RETURN
```

```
;transmit to main board the current motor position and detector value
mod_state:
CLRF STATUS
CALL analog_read ; get current motor position
MOVF ANAHI,W
BSF STATUS,RP0
BSF TXSTA,TXEN ;enable transmission
BCF STATUS,RP0
CALL send
CALL detect
; MOVF DETECT_AMP,W ;check to see if past threshhold
; CALL send
BSF STATUS,RP0
BCF TXSTA,TXEN ;disable transmission
BCF STATUS,RP0
RETURN

break_pos:
BCF STATUS,RP0
bploop: CALL detect
BTFSS FLAGS, BLOCK
GOTO bploop ; wait til see a hand
bploop1: CALL detect
BTFSC FLAGS, BLOCK
GOTO bploop1 ; wait til hand is gone
CALL analog_read ; get current motor position
MOVF ANAHI,W
BSF STATUS,RP0
BSF TXSTA,TXEN ;enable transmission
BCF STATUS,RP0
CALL send
MOVF ANALO,W ; send motor position
CALL send
BSF STATUS,RP0
BCF TXSTA,TXEN ;disable transmission
BCF STATUS,RP0
RETURN
```

```
;;laser.asm
;;laser board control code

ltimer:
BCF PIR1,TMR1IF ;clear flag
MOVLW 0xFE
MOVWF TMR1H
CLRF TMR1L
MOVLW BIT(LASER)
XORWF PORTC,F ;toggle laser state
RETURN


en_laser:
BSF T1CON,TMR1ON ;enable timer1
RETURN

dis_laser:
BCF T1CON,TMR1ON
BCF PORTC,LASER ;turn off laser
RETURN

get_laser:
BSF ADCON0,0x02 ;start a/d conversion
glloop: BTFSC ADCON0,0x02 ;check for end of conversion
GOTO glloop
MOVF ADRESH,W
MOVWF DETECT_AMP
RETURN

show_laser
CALL get_laser
CALL conv_byte

RETURN

detect:
CALL get_laser
MOVLW THRESH
SUBWF DETECT_AMP,W
BCF FLAGS,BLOCK
BTFSC STATUS,C ;if no borrow
BSF FLAGS,BLOCK ;then laser blocked
RETURN
```

```
//mom.c
//main board code

#include <16f876.h>
#fuses nowdt,hs,noprotect,put,brownout,nolvp
#use delay(clock=20000000)
#define LED PIN_B7

#use rs232 (baud=57600, xmit=PIN_C2,rcv=PIN_C3)

int hibyte;
int lobyte;
int detect;
int baby=0;


void printout(int what){
printf("%c",what);
return;
}

#use rs232 (baud=38400, xmit=PIN_C6,rcv=PIN_C7)

void babytalk(){
printf("l");
return;
}

void babytalk1(){
printf("m");
return;
}

void swait(){
delay_us(10);
return;
}


void talk(int module, int command){
putc(0xAA);
swait();
putc(0x00);
swait();
putc(module);
```

```
swait();
putc(command);
swait();
putc(module^command);
swait();
putc(0xAA);
swait();
putc(0x01);
swait();
return;
}

void start(){
putc(0xAA);
swait();
putc(0x00);
swait();
return;
}
void finish(){
putc(0xAA);
swait();
putc(0x01);
swait();
return;
}


void getstate(){
start();
putc(0x02); /*mod address*/
swait();
putc(0x00); /*mod state*/
swait();
putc(0x00^0x02); /*checksum*/
swait();
finish();
hibyte=getc();
/* lobyte=getc();
detect=getc();*/
return;
}

void say_hi(){
while(baby!='s'){
```

```
baby=getc();
}
baby=getc();
return;
}
void flash(){
output_high(LED);
delay_ms(500);
output_low(LED);
delay_ms(1000);
return;
}


#use rs232 (baud=57600, xmit=PIN_C2,rcv=PIN_C3)

void main(){
int aaa;
long lazy;
SETUP_ADC(ADC_CLOCK_DIV_32);
setup_adc_ports(ALL_ANALOG);
set_adc_channel(0);
output_low(LED);
delay_ms(500);
output_high(LED);
printf("yo\r\n");
while(1){
aaa=getc();
if(aaa=='1'){
talk(0x02,0x03);
say_hi();
printf("%X ",baby);}
if(aaa=='2'){
talk(0x03,0x03);
say_hi();
printf("%X ",baby);}
if(aaa=='3'){
talk(0x04,0x03);
say_hi();
printf("%X ",baby);}
if(aaa=='4')
talk(0x02,0x04);
if(aaa=='5')
talk(0x03,0x04);
if(aaa=='6')
```

```
talk(0x04,0x04);
if(aaa==' '){
lazy=READ_ADC();
printf("%lu ",lazy);
talk(0x02,0x03);
say_hi();
printf("%U ",baby);
talk(0x03,0x03);
say_hi();
printf("%U ",baby);
talk(0x04,0x03);
say_hi();
printf("%U\r\n",baby);
}
}
}
```

Bibliography

[1] Joel Chadabe. *Electric Sound: The Past and Promise of Electronic Music.* Prentice-Hall, Inc, 1997.

[2] Albert Glinsky. *Theremin: Ether Music and Espionage.* University of Illinois Press, 2000.

[3] Cynthia Jacobs. Investigating non-tactile midi controller for severely disabled children, 1997. http://www.templetap.com/ntmidi.html.

[4] Jean Michel Jarre. Laserharp. http://www.jeanmicheljarre.com.

[5] Norman S. Nise. *Control Systems Engineering.* Wiley, 2000.

[6] J. Paradiso, K. Hsiao, J. Strickon, J. Lifton, and A. Adler. Sensor systems for interactive surface. *IBM Systems Journal*, October 2000.

[7] Joseph Paradiso. Electronic music interfaces. *IEEE Spectrum*, December 1997.

[8] Dr. Godfried-Willem Raes. Gesture-controlled virtual musical instruments, 1999. http://www.logosfoundation.org/ii/gesture-instrument.html.

[9] Gino Robair and Bean. Electric ladyland. *Electronic Musician*, 2001.

[10] Dean Rubine and Paul McAvinney. The videoharp. In *International Computer Music Conference Proceedings*, Cologne, September 1988.

[11] Joshua Reynolds Smith. *Electric Field Imaging.* PhD thesis, MIT Media Lab, 1988.

[12] Leonello Tarabella, Graziano Bertini, and Tommaso Sabbatini. The twin towers: a remote sensing device for controlling live-interactive computer music. In *2nd International Workshop on Mechatronical Computer Systems for Perception and Action*, Pisa, Italy, February 1997.