# A NOVEL CONTINUOUS PITCH ELECTRONIC WIND INSTRUMENT CONTROLLER

BY

ZUOFU CHENG

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Adviser:

Lecturer Lippold Haken

# ABSTRACT

We present a design for an electronic continuous pitch wind controller for musical performance. It uses a combination of linear position, magnetic reed, and air pressure sensors to generate three fully continuous control dimensions. Each control dimension is encoded and transmitted using the industry standard MIDI protocol to allow the instrument to interface with a large variety of synthesizers to control different parameters of the synthesis algorithm in real time, allowing for a high degree of expressiveness not possible with existing electronic wind instrument controllers. The first part of the thesis will provide a justification for the design of a novel instrument, and present some of the theory behind pitch representation, encoding, and transmission with respect to digital systems. The remainder of the thesis will present the particular design and explain the workings of its various subsystems.

# ACKNOWLEDGMENTS

I am grateful to my thesis adviser, Dr. Lippold Haken for his tireless support in the conception, design, and production of this instrument and his expertise in computer music. I would like to extend my thanks to the Department of Electrical and Computer Engineering at the University of Illinois for providing me with the financial support and scholarship that went into this work. I would like to thank Craig A. Zeilenga of the ECE Machine Shop for his work in realizing this instrument and Mark W. Smart of the ECE Electronics Services Shop for his assistance in fabricating the printed circuit board. I would also like to thank the ECE Publications Office for their work in editing this thesis. Finally, I extend my gratitude to faculty members Dr. N. Narayana Rao and Dr. James W. Beauchamp for their support of this project and my work at the University of Illinois.

# CONTENTS

# LIST OF FIGURES

# Chapter 1

# INTRODUCTION

## 1.1 A Brief History of Electronic Musical Instruments

The early history of electronic musical instruments had no particular focus on discrete pitches such as played by a piano keyboard or fretted guitar. One of the earliest successful electronic musical instruments was the theremin, designed by Leon Theremin in the 1920s. The theremin used the variable capacitance between its two antennae and the performer's hands to modulate the pitch and amplitude of the resulting sound in a continuous manner. A similar early continuous pitch electronic instrument is the ondes Martenot, invented by Maurice Martenot in 1928, which uses a sliding metal ring positioned across a wire to control pitch [1]. It is worth noting that both instruments took inspiration from continuous pitch acoustic instruments. The theremin attempted to emulate some aspects of a singing voice in timbre (though not in control), and the ondes Martenot was inspired by the playing methods of the cello [2].

This focus began to change with the increasing popularity of electronic organs. While the true timbre production methods of the tone-wheel organs (notably the Hammond B3) are unique in that the "harmonics" are in fact inharmonic intervals due to the limited number of tone wheels, pitch became quantized to those available on the piano keyboard. Although the first generation of programmable, affordable electronic musical instruments such as the Minimoog maintained the possibility of continuous pitch control methods via CV (control voltage), this was more a convenience due to the underlying analog tone generation methods, than an effort to allow for continuous pitch performance. The framework of a piano keyboard controlling an electronic tone generator was already set [3].

By the time digitally controlled (yet still with analog tone generation) synthesizers became popular in the late 1970s, there was already a desire to separate the performance interface or controller from the synthesizer. During this era, several manufacturers of electronic keyboards attempted to create a universal control protocol, which largely failed in the marketplace. It was not until the finalization of the MIDI specification in the early 1980s, that it became possible to control any synthesizer with any controller – provided that the controller, for the most part, produced discrete pitches. In fact, MIDI had been designed explicitly around the framework of a piano keyboard, such that the core method by which a note is played is through a series of sequential note-on and note-off messages which encode a single discrete "note number" value and a single velocity corresponding to the travel of the key [4].

While there have been commercial attempts to create a MIDI theremin, almost all attempts at novel music performance controllers have been discrete pitched, including the popular Yamaha wind controller series (WX) and Akai electronic wind instrument (EWI) [5]. While these controllers offer intuitive playability for acoustic wind instrument players (typically key switches are interpreted as Boehm fingerings to assist flute or saxophone players in learning the instrument), the use of discrete pitches limits the level of expressive performance. For example, on a fretless instrument such as a violin, the pitch is constantly being adjusted through acoustic feedback to play in tune as well as to create vibrato. An electronic instrument that can allow for this type of natural playing style would require not only a high degree of absolute pitch accuracy (to prevent audible artifacts due to quantization of pitch) but also an extremely fast response time to allow for the musician to tune and adjust.

## 1.2 The Continuum Fingerboard

One unique instrument that addresses these issues is the Continuum Fingerboard, designed and built by Dr. Lippold Haken of the University of Illinois [6]. It has an approximately piano keyboard-sized playing surface that tracks the x-, y-, and z-axes independently and continuously for up to 10 fingers. The Continuum has both very high pitch accuracy (greater than 200 cents per inch) and very fast response time (under 1 millisecond), which it achieves by using very precise Hall effect sensors to measure the displacement of a series of steel rods under the playing surface. This allows for both complex playing technique and sophisticated sound programming. For example, a subtle vibrato is typically played by a slight rolling of the finger at a certain x-position. The z-axis (finger pressure) can control loudness, and the y-axis (front to back) can control the morph between two timbres. The control data are transmitted via MIDI using "pitch bend" messages, which are typically used to support a ribbon controller or a pitch wheel on a synthesizer keyboard. The advantage of using MIDI is that it is inherently low latency and natively supported in most synthesizers, sequencers, and modular signal processing environments (such as Kyma or Max/MSP). One key importance in studying the design and midi encoding methods of the Continuum is that it allows one to port sophisticated synthesis algorithms directly to any new continuous pitch instrument, provided the resolution and number of control channels are similar.

## 1.3 Purpose and Organization of the Thesis

The purpose of the music performance controller as described is to allow for a highly expressive performance in a manner that uses techniques similar to those used for playing traditional instruments, but not necessarily identical nor instantly playable for instrumentalists fluent on any acoustic instrument. This is a subtle but important point, as we are in effect trying to reuse some of the expressive devices wind instrumentalists have learned through traditional musical training for playing certain expressive passages, such as a slow crescendo or a series of staccato notes. Therefore, we expect that skilled instrumentalists will be able to use some aspects of their training to play expressively on this instrument, unlike, for example, on a theremin, which makes no effort to capture the feel of any acoustic instrument. Unlike existing wind controllers, however, we are not designing the instrument to emulate any particular acoustic wind instrument. This is because for any acoustic instrument, certain usability parameters are constrained such that the instrument can play at the desired pitch range with sufficient dynamics and with a pleasant timbre.

Chapter 2 of this thesis will cover some basic background of musical instruments, such as pitch and tuning. We will also go over the basics of the MIDI protocol as well. In Chapter 3, we will cover the details of the design of our particular instrument, starting from the sensors that form the core technology of the instrument and encompassing the physical design as well as the embedded system design. Finally in Chapter 4, we will tackle the general issue of synthesis algorithms for an expressive instrument, and end with directions for future research in both improving the instrument and designing sounds for expressive playing.

# Chapter 2

# BACKGROUND

In order to explain the decisions made during the design of the instrument, it is necessary to explain some principles of music theory and human perception. Fundamentally, we are interested in an instrument that, like the Continuum, encodes several continuous channels of controller information and transmits this data to a synthesizer. While it is possible to arbitrarily map controller channels to sound generation parameters, we are interested in designing the instrument for a particular mapping that is natural to a wind musician playing tonal music. Throughout this thesis my concern is exclusively with tonal music. Therefore, any design of the instrument must keep in mind the basic elements of tonal music such as pitch, rhythm, and dynamics. Furthermore, we must keep in mind the limitations of the MIDI protocol, and devise efficient and compatible ways to communicate with a large variety of synthesizer units.

## 2.1 Introduction to Musical Pitch

All discrete or continuous pitched musical instruments used in Western music are capable of producing a series of pitches denominated into units of semitones of the twelve-tone equal-tempered scale within a certain playing range. For clarity, "pitch" for the rest of this thesis will refer to the perceived sensation of a sound having a certain fundamental frequency. This means that within an octave (frequency ratio of 2:1) there are 12 logarithmically spaced intervals having theoretical fundamental frequencies given by

$$P_n = P_{ref}\, \sqrt[12]{2}^{\,n-n_{ref}} \tag{2.1}$$

where n is the particular note in units of semitones relative to the reference pitch, typically denoted as middle C, the marked center key of the piano with a fundamental frequency of:

$$P_{ref} = 261.626\,\mathrm{Hz} \tag{2.2}$$

On a standard 88-key electronic keyboard, the above equation will yield the fundamental frequency for key number n counting all the black and white keys from the lowest key (key 1 being A0) given that $n_{ref} = 40$, the key number for middle C. Note that while discrete pitch instruments are quantized by design to the pitches denoted above, continuous pitch instruments must also be able to produce the pitches given by semitones. Musicians who play continuous pitch instruments therefore must use a combination of memory and feedback to reach the desired pitch quickly but still adjust to minute differences in tuning. These minute changes in pitch are generally measured in cents, that is, 1/100 of an equally tempered semitone or, equivalently, 1/1200 of an octave.

In order to design a digital instrument, we are interested in how finely the pitch must be quantized in order to mask any artifacts. The earliest psycho-acoustic research done by Fechner suggests that measuring the number of perceptual JNDs, or just noticeable differences, between two physical quantities should yield a useful measure of the perception of the physical quantity. This idea was based on an earlier experiment by Weber, who, working with weights, realized that a person holding 10 grams will notice a change in 5 grams, while a person holding one kilogram will not. Formally stated, this is written:

$$\Delta p = k\frac{\Delta S}{S} \tag{2.3}$$

where S is the nominal level of the stimulus, $\Delta S$ is the change in the stimulus, and $\Delta p$ is the change in perception. This formula is typically known as Weber's law. A naive interpretation of Weber's law implies that the JND and, therefore, the quantization unit of pitch can simply be a fixed interval, measured in some number of cents.

In practice several factors greatly increase the pitch resolution needed for a musical instrument. One phenomenon is that in fact for frequencies below 1

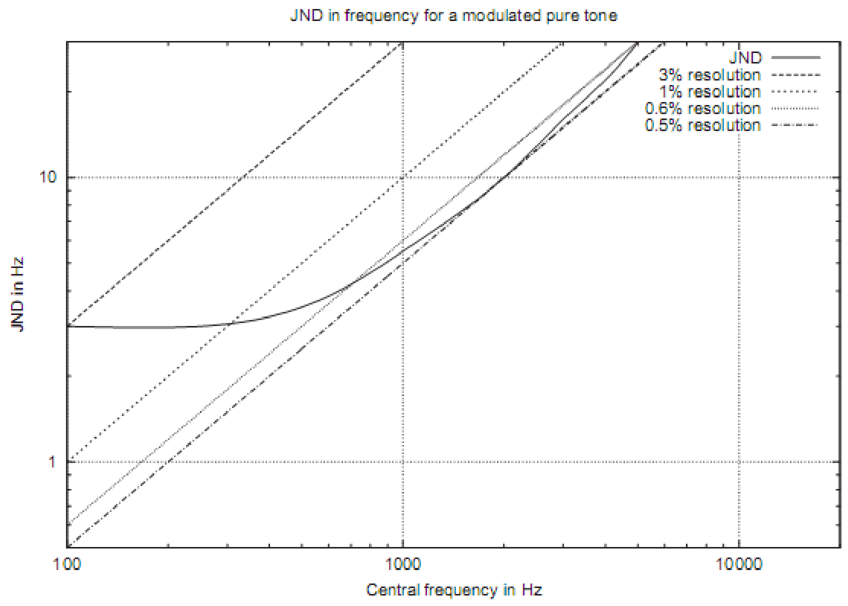

Figure 2.1: Pure tone JND and Weber's law with various k

kHz, the JND for pure tones is approximately fixed at 3 Hz, rather than being proportional to the nominal frequency, as shown by Figure 2.1. These results were derived by Shower and Biddolph by slowly modulating a sine wave at a center frequency by increasing amounts and noting when the subjects would perceive a change, thereby measuring the frequency JND [7]. This result by itself is not a problem, so long as the instrument works on a logarithmic scale, as 3 Hz is a rather large interval for low notes (for reference, the lowest key on a normal piano is A0, which has a nominal frequency of 27.5 Hz). However, in practice, it is unlikely that the desired timbre of any instrument is a pure sine wave. It has been shown that for pulse waveforms, the frequency JND is significantly smaller. Furthermore, we cannot simply be satisfied with pitch accuracy at a single frequency in the case of complex timbres; we have to consider the effects of the higher harmonics as well. Finally, the issue of beating arises in situations where ensemble playing may be desired. Even slight amounts of pitch inaccuracy can result in very clear beat patterns which may be heard provided the note is long enough. Typically, vibrato is used to mask this effect during legato phrases. The general consensus is that 0.2% or about 3.5 cents of pitch accuracy is required for an instrument to be in tune over a wide range.

## 2.2 Introduction to MIDI

As stated in the introduction, MIDI was designed in the early 1980s primarily by a consortium of manufacturers to standardize and replace the large number of mutually incompatible musical instrument interfaces [8, 9] and to provide a way for digitally controlled synthesizers to avoid having to use expensive A-D conversion in order to interface via control voltage (CV), which had been the previous standard for many years. As such, MIDI reflected the limited technology of digital systems at the time, running at a relatively slow 31.25 kilobit per second with a uni-directional connection and a complete lack of handshaking. However, MIDI has a significant number of advantages over later protocols that were designed to address its perceived shortcomings: particularly the low bitrate, large amount of cabling required for bi-directional communication, and inability to send audio channels in conjunction with control [10, 11]. One advantage of MIDI is that it is strictly point to point, allowing any controller to communicate to any synthesizer, one synthesizer to communicate to another, or a sequencer to communicate to both. Unlike USB (universal serial bus, commonly used for computer peripherals), there is no notion of a host required for communication, which greatly simplifies the link-level implementation and reduces latency. Furthermore, the lack of handshaking and protocol mandated packetization allows for a low-performance embedded processor to implement the protocol without significant hardware support. All that is required is an asynchronous serial port capable of approximately 31.25 kbaud with 8-bit packets, no parity bit, and a single stop bit, typically known as 8-N-1 mode.

At the very basic link level, MIDI is implemented as a uni-directional current loop. The most basic form of the line driver and receiver is shown in Figure 2.2.

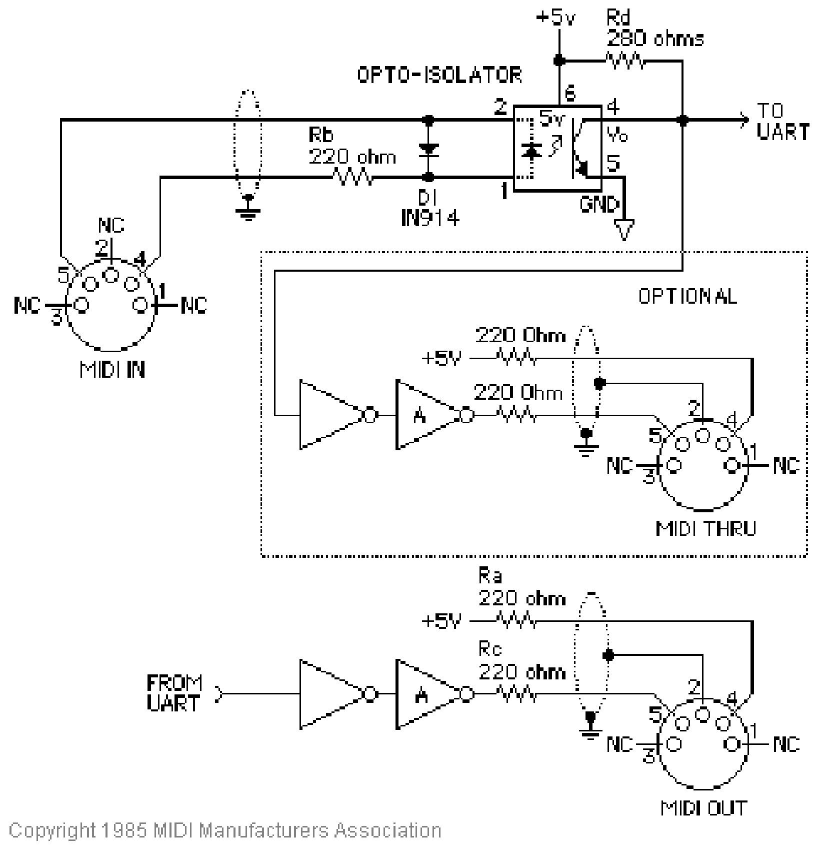

Figure 2.2: MIDI electrical specification diagram

One important feature of the electrical specification for MIDI is that the receiver must be optically isolated from the line driver. This is very important for musical instruments, as typically the sound generator may be located quite far from the performer's controller. This presents the natural risk of ground loops forming; that is, if the controller's power supply ground is not at the same potential as the synthesizer's, current will flow between any coupling that exists (for example, a data connection) so long as they are not isolated. This is usually manifested as a large 60 Hz hum for an analog audio connection or corrupted data for a digital connection. The mandated optical isolation reduces this problem and, therefore, is one of the reasons why MIDI has survived so long in industry.

We have already stated that MIDI operates as an 8-N-1 asynchronous serial communication bus with a rate of 31.25 kbaud. It should be noted that this number includes the required framing bits (start and stop), which together adds an overhead of 2 bits per byte of data. Therefore, the MIDI is only capable of transmitting a raw character rate of 25 kbit per second. More precisely, the transmission frame is shown below in Figure 2.3.

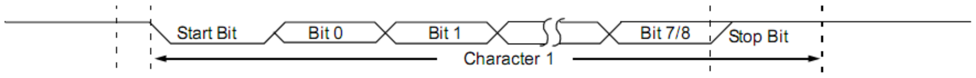

Figure 2.3: MIDI byte

Notice that the bit time in Figure 2.3 is defined as 32 $\mu$s. The transmitter may initialize a single character transfer at any time by driving the transmission line low for a single bit time. It should be noted (and easily seen from Figure 2.2) that this implies current begins to flow (therefore, the current flow is opposite the logical polarity). The transmitter then transmits the eight data bits in quick succession taking up a total of $32 \times 8$ or 256 $\mu$s before finally idling the line (returning to logical high) for at least 1 additional bit time. The receiver therefore relies on knowledge of the exact bit time to detect the start bit and lock in the eight data bits. Therefore, the stop bit is required to give the receiver time to resynchronize for every frame.

At the software level, MIDI messages operate as 16 independent channels. As we have seen, the MIDI link interface transmits a frame of a single byte. Therefore, almost all MIDI messages are multi-frame messages. The first frame (byte) of a message is typically called the “status byte”, which is followed by several “data bytes”. The status byte always follows the format 0b1cccnnnn, where the first bit signals a status byte, ‘ccc’ encodes the event, and ‘nnnn’ signals the channel number. Similarly, data bytes follow the format

0b0ddddddd, where 'ddddddd' encodes the seven bits of data. For example, the MIDI note-on message tells the synthesizer to start playing a note at a certain pitch. It requires the status code 001 followed by two bytes encoding the note number (pitch) and velocity. Therefore, to play middle C (note 60) at medium velocity (63) on the first channel, we can send 0b10010000 0b00111100 0b00111111. Notice that MIDI is completely asynchronous both at the link level as well as the software level. Messages encode events, which may happen at any time and require an instant response. It is also of note that MIDI assumes the primary mode of note input is a keyboard controller, as evident by the velocity being a required piece of data to start playing a note. Fortunately, there is also a way to encode both continuous pitch and control via MIDI. Continuous control is the more straightforward of the two and is handled by a dedicated control change message. This is because for each of the 16 channels, MIDI has 128 controllers, which each have a 7 bit value. This is intended as a way to control synthesis parameters, such as filter cutoff and oscillator tuning, from a keyboard that has a physical interface, such as knobs or buttons. This message has an event code '001' followed by a data byte that specifies the controller number and a data byte specifying the controller value. While all 128 controllers are transmitted in the same manner, in practice synthesizers interpret certain standardized controllers differently. For example, controller 64 is typically used for sustain, whereby holding down the sustain pedal sends a message that sets controller 64 to 127, and releasing the sustain pedal sets controller 64 to 0. Unfortunately, it is not possible to use continuous controllers for pitch modulation. This is because continuous controllers only transmit a 7-bit data value. Recall from the previous section that a pitch accuracy of 3.5 cents is considered the minimum for playing in tune. Suppose that the instrument is expected to cover four octaves, in which case each value would have to cover 37.5 cents, far coarser than what is required to play in tune. Fortunately, highly accurate pitch data may be transmitted by yet another mechanism: pitch-bend. Typically, this is used for the pitch bend wheels on MIDI keyboards, because the designers of MIDI realized that the standard 128 levels of controllers would cause a noticeable stepping in a glissando. Therefore, unlike with controllers, pitch bend is sent with 14 bits (two data bytes) of data. However, given that the original purpose of pitch bend is for pitch wheels on MIDI keyboards, pitch bend messages are sounded relative to currently playing notes. Encoding absolute pitch (as required for a continuous pitch musical instrument) is a difficult process, which will be discussed later in this thesis.

# Chapter 3

# DESIGN OF THE INSTRUMENT

## 3.1 Overview

As stated, the instrument is a continuous pitch MIDI wind controller. The player primarily controls dynamics by blowing into a saxophone-like mouthpiece located at the top of the instrument, similar to the z-axis on the Continuum, which measures finger pressure. Inside the mouthpiece is a reed, which the lip pressure (not to be confused with the air pressure of the breath) or displacement of the reed is measured. This mouthpiece is connected to the delrin main body of the instrument via an over-sized neck, which houses the main electronics. In addition to the electronics, the neck splits the airflow from the player's breath into two paths. The first path is connected to an integrated silicon pressure sensor. The second path carries air through a thin plastic tube to a vent at the bottom of the instrument. This path allows air to flow through the instrument with a certain resistance to make the overall feel of the instrument more consistent with that of an acoustic wind instrument. This resistance can be adjusted via a valve in the body of the instrument. Furthermore, this design causes a relatively static column of air to form at the input of the pressure sensor; this design allows for saliva and condensation to drain outside the instrument which prevents constant humidity from corroding the pressure sensor. This air flow tube is extremely important in maintaining the calibration and reliability of the pressure sensor.

In addition to the airflow tube, the body of the instrument houses a stainless steel slide. This is designed as the main method by which the player controls pitch; it is analogous to the x- (left to right) axis of the Continuum. Here the first and second prototypes of the instrument diverge significantly. The first prototypes use a capacitive caliper type mechanism, while later designs use an optical system. The optical mechanism will be discussed in detail later in the following section. Unlike typical linear digital readouts (DROs) used for computer-aided machining (CAM), the slide acts as the articulator, while the reader acts as the stator. This prevents issues where an internal ribbon cable may get caught in the mechanism of motion or where it may fold irrecoverably due to unpredictable extreme motions during performance. Furthermore, the body completely encloses the slide mechanism through its entire 10 inches of travel. This prevents the the delicate position sensing mechanism from being exposed and, therefore, fouled or damaged. In addition, this arrangement causes the amount of contact the slide makes with the body to be constant throughout its range of travel, which results in a near-constant friction coefficient and also minimizes any side-to-side play of the slide during performance. A necessary result of this arrangement is that the body of the instrument is quite long, because it has to completely enclose the length of the slide and the maximum length of its travel. A third prototype currently in development dispenses with this mechanism completely; it relies on an internal optical communication channel between the reader and the circuit board, and utilizes a sliding reader mechanism. This prototype will be discussed in the last section of the thesis.

The instrument's entire system is driven by a microcontroller (a digital signal controller) from the Microchip dsPIC family. This family was chosen primarily for its wide range of peripheral options, such as multiple serial ports and built-in high-speed analog-to-digital converters, as well as for the performance advantages of using 16-bit math over simpler microcontrollers, which are typically 8 bit. The controller directly drives all the hardware, foregoing an embedded operating system in order to increase system responsiveness and to maintain the simplicity of the software. Directly driving the hardware is possible because the link layer of MIDI is very simple compared to protocols such as USB and Firewire (IEEE-1394).

Figure 3.1 shows a top-level diagram of the first prototype main controller board. Subsequently. this will be referred to as the “X1” board. Figure 3.2 shows a revised schematic of the second prototype board, “X2.” The primary difference between the revisions is to support a change in the linear positional element; as a result, the main system controller and many support devices had to be changed. Each subsystem will be discussed in detail in the following sections of this thesis.

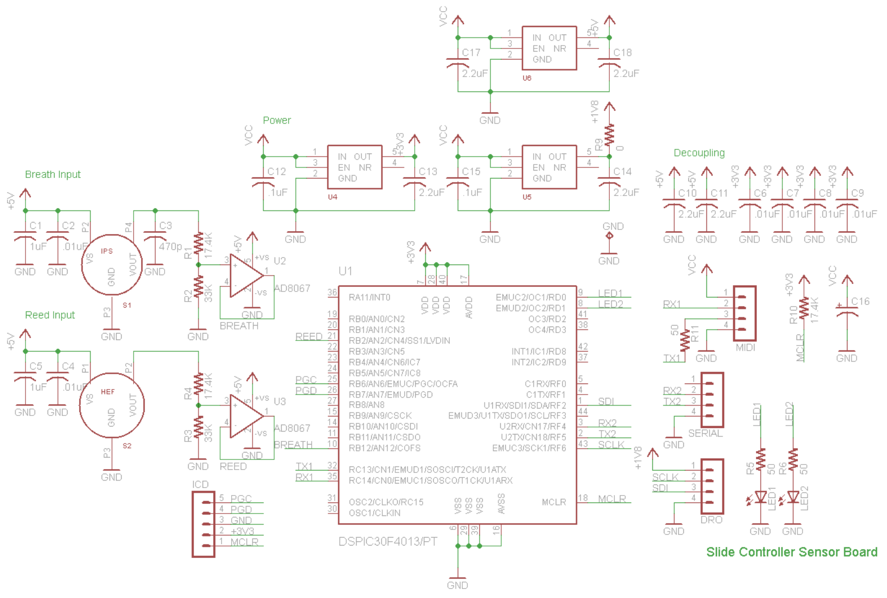

Figure 3.1: Top-level schematic of first revision (X1) board

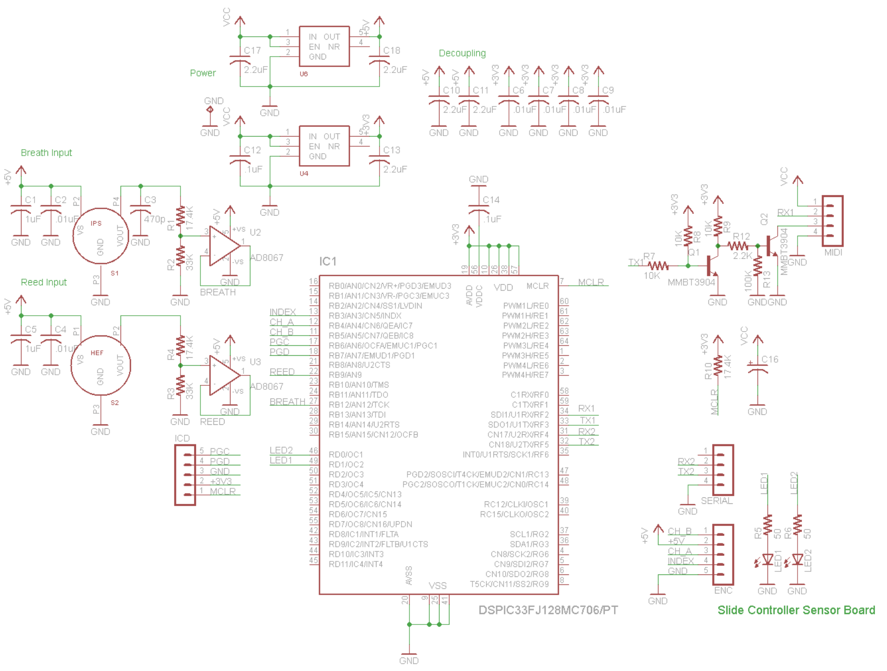

Figure 3.2: Top-level schematic of second revision (X2) board

## 3.2 Linear Position Sensing

The first design of the linear sensor utilizes digital readout similar to the kind used for positioning machine tools. This was chosen because of its very high (0.00005 inch or 0.05 mil) accuracy. The sensing element here is a passive plastic strip embedded into a steel slide acting as a ground. The sensing element consists of a fine grate of conductive material embedded in an insulator. The static reader is effectively a series of small antennae along with a known (small) trace inductor and resistor in series which are embedded into the circuit board driven by a series of tuned oscillators. The antennae are staggered to have a positioning pitch different than the pitch of the grate on the strip. Typically, while there is dielectric material underneath the antennae, the combined system forms an R-L-C circuit with known resonant frequency given approximately as

$$\omega_0 = \frac{1}{\sqrt{LC}} \tag{3.1}$$

As conducting material slides underneath the antennae, the capacitance of the circuit decreases; the resulting R-L-C resonant frequency changes towards the frequency of the oscillator, causing an attenuation of the oscillator's AC voltage due to conduction to the grounding slide. The onboard microprocessor (onboard to DRO itself; not to be confused with the dsPIC microcontroller in the instrument) then compares the relative amplitudes of the readback elements to derive a relative linear position to high degree and adds the relative amplitude to an internal absolute offset counter to derive the absolute position.

For the particular DRO used, the output given by the device's own controller is conveniently a 48-bit binary digital synchronous serial signal. This is illustrated in Figure 3.3, where the clock signal is on channel 1 and the data signal is on channel 2. The 48-bit signal is divided into two 24-bit "absolute" and "relative" twos complement values, sent once every 20 ms. Each unit here represents 1/20,480 of an inch, or approximately 0.05 mil. However, the instrument has no hardware origin; therefore there is no way to get a true absolute measurement. The "absolute" position is merely the current position of the slide relative to the scale at power up, whereas the "relative" origin can be reset by pulling up the clock to 1.5 V. The entire DRO is powered by a 1.5 V supply, which necessitates logic level translation in the X1 board to interface with the 5 V dsPIC30, which it communicates with via the dsPIC's SPI port (pins SDI and SCLK for data in and clock, respectively). The software onboard the dsPIC reads the data as two groups of three sequential 8-bit SPI transactions, triggered by the idling and then drop of the clock signal, as shown.

One significant advantage of this system versus a simpler encoding system is that it allows for higher accuracy than the granularity of the grate would suggest. This is because the amplitude of the sine wave read back by the microcontroller is an analog quantity; therefore, it is possible to tell not only if the conductor is under the reader at all, but also how much of the conductor is under the reader. The use of relative amplitude measurements allows for some leeway in how much contact the sensing element has with the reader. In fact,

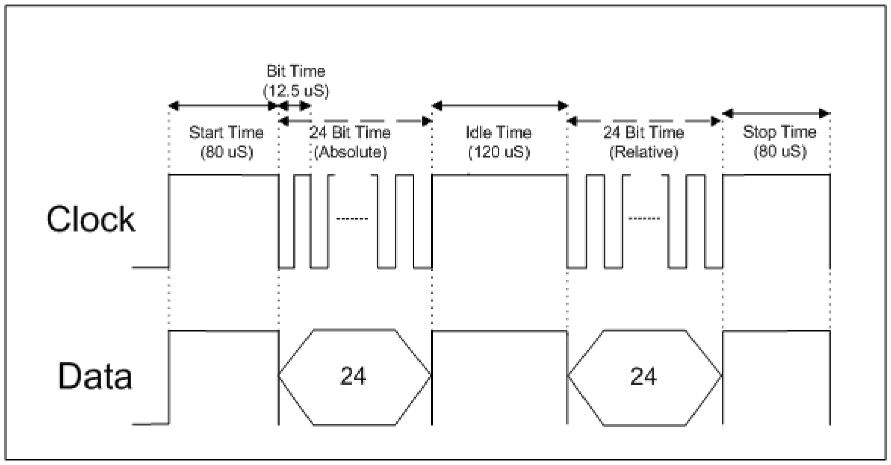

Figure 3.3: Data output of the DRO system

the system would still work to some degree without physical contact between the sensing element and the reader; however, it should be noted in this case that the system becomes extremely sensitive to any distance deviation (slant) from one end of the reader to the other. Furthermore, the accuracy of the measurement decreases due to the smaller overall capacitance, and this manifests itself in the onboard controller losing track of its absolute position. In this implementation, the slide is designed to lightly contact the reader.

A problem with this design is that, because of the light contact between the reader and the slide, very quick movement of the slide would cause the reader to lose contact with the surface of the membrane and, therefore, skip codes. Because the system lacks any absolute zeroing ability, any skipped codes would require a manual operator restart, which is clearly unacceptable. Furthermore, because of tolerances in machining the delrin instrument body, it was not possible to maintain a constant light contact between the slide and the reader throughout the entire range of travel. A slight warping of the instrument body, for example, would cause the slide to bind at one end of travel but not to make contact at the opposite end. This unequal friction is clearly unacceptable for expressive playing, as any attempt at subtle vibrato, for example, becomes exaggerated and distorted as the player attempts to overcome the effects of friction.

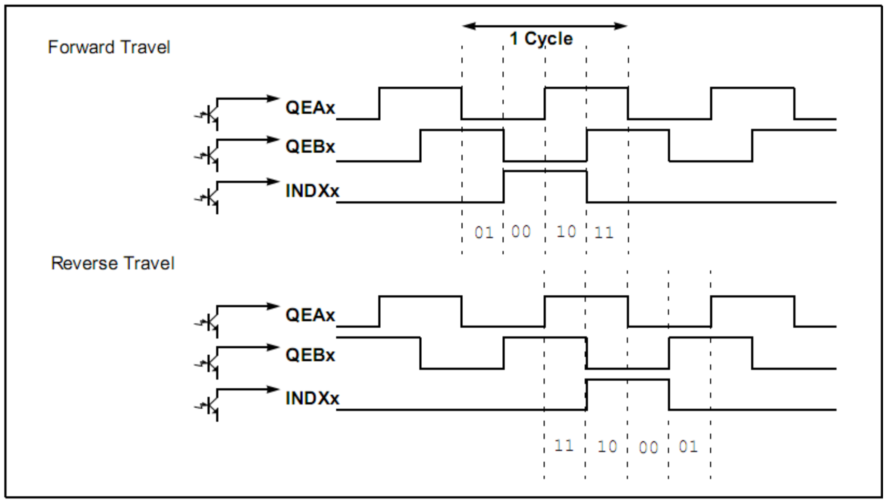

Figure 3.4: Data output of the optical encoder

The solution used in the X2 board involves a complete revamp of the linear sensing element. Instead of a capacitive DRO, the slide is attached to a transmissive linear optical strip, essentially a clear plastic strip demarcated by thin black lines at 250 lines per inch. This strip also incorporates an index value at the midway point (5 inches from either end), allowing for an absolute reference every time the reader passes through the index. The reader itself is a three-channel optical LED-based reader. Two of the channels are co-linear, with the A output ahead of the B output when the encoder is moving in the forward direction. The two channels are necessary so that forward travel may be distinguished from reverse travel. The third channel is the index channel, which is located on another track. All of the channels are active high (that is, they output a logic high when the track is marked) and are 5 V TTL compatible.

An output waveform of the reader is shown in Figure 3.4. Unlike the DRO system, the optical encoder does not keep track of the position internally. It is up to the system microcontroller to keep track of the relative position. An intermediate prototype used the microcontroller's GPIO (general purpose input/output) pins to read the encoder channels. However, because the encoder channels are asynchronous, this configuration caused glitches when the instrument slide was moved too fast. The solution required a change of the microcontroller from a dsPIC30 series to a dsPIC33 series controller, which incorporates a hardware module of asynchronous logic to read the the encoder channels. This change allows the slide to move as fast as reasonable possible (traveling the whole length in less than 500 ms) and still provide accurate measurements of position. Although the dsPIC operates at 3.3 V, the digital inputs are 5 V tolerant; therefore, the channels of the encoder are connected directly to the RB3/RB4/RB5 input ports.

One final point about the linear measurement is that the linear encoder is significantly less accurate than the DRO. While the DRO's absolute accuracy is about 0.05 mils, the encoder can only discriminate changes of 4 mils. Fortunately, for an instrument with a range of 2.5 octaves (3000 cents) over 10 inches of travel, this still translates to an absolute pitch accuracy of 1.2 cents, which is still well below the specifications set in Section 2.4.

## 3.3 Breath Pressure Sensing

Unlike the use of a linear encoder for pitch sensing, the use of pressure sensors for musical control has significant precedent, both in products such as the Yamaha WX series as well as in academic research. In general, pressure sensing is used to control amplitude and to determine the beginning of the note. At a minimum, therefore, the sensing mechanism must be fast enough to capture the fastest notes likely to be played by the performer. This rate is often given as approximately 100 Hz, given that several measurement points for each note event are likely needed to provide at least a basic ADSR (attack, decay, sustain, and release) envelope for each note. Analysis presented in [12] gives a maximum breath reporting rate of 140–160 Hz, as in the WX series of instruments; this rate is sufficient for controlling most synthesis methods commonly used with breath controllers (sampling, subtractive, digital waveguide). However, as several patches developed for the Continuum Fingerboard have shown, some more sophisticated synthesis methods can take advantage of higher-rate continuous control. In the Continuum, one method is to use the pressure envelope formed by the fingers as an input into some digital resonant system (either a digital waveguide or a series of harmonic resonators). This allows for a wide variety of sounds to be created using different finger gestures, creating a subtle level of control more akin to an acoustic instrument. This is analogous to the use in wind instruments of growl, a technique that essentially involves vocalizing into the mouthpiece to override the usually harmonic oscillation of the reed. In order to capture subtle effects such as vocalization through the breath sensor, a sensor with the fastest possible response must be used.

The sensor used is the Motorola MPX5010 (gauge configuration and surface mount packaging; full part number: MPXV5010GP). This device requires 5 V power and outputs an analog voltage from 0 to 5 V corresponding to to a pressure range of 0 to 1.5 psi. The pressure input is fed by a branch of the airflow tube as explained in the introduction of this chapter. A scaling circuit is used both as an input buffer and to scale the voltage of the sensor to 3.3 V (via an op-amp) before input to the microcontroller's built-in ADC. The input is sampled at a 12-bit depth at a rate of 250 ksample/s and is averaged inside the processor to a much lower reporting rate (which varies depending on the status of the instrument). This circuit is identical in both X1 and X2 versions and is read through microcontroller ADC port AN12.

## 3.4 Reed Displacement Sensing

The final control channel is the reed displacement, measured by a magnetic Hall effect sensor. Inside the mouthpiece is a cantilever, which rests against the reed. While the reed does not oscillate as in a real woodwind instrument, lip pressure on the reed will cause it to move the cantilever, which in turn moves a magnet inside the instrument neck. It does so through a ball joint, so as to maintain the air seal around the mouthpiece. This magnet is located right above the Hall effect sensor on the circuit board, and any slight change in position is picked up by the sensor. The sensor used is the Allegro A1301 (in SOT-23 packaging, part number: A1301KLHLT-T), a 5 V analog sensor. This is buffered through a circuit identical to the breath sensor. There is one difference between the revisions; for the X2 board, the ADC port was moved to AN9 from AN2 because of a conflict with the linear encoder inputs.

This arrangement works well because there is very low noise due to the high-performance Hall effect sensors. However, due to the very small displacement of the reed in normal playing (too much pressure on the reed will cause the mouthpiece to close completely), the total change in voltage is only approximately 40 mV. Read at 12 bit, this only yields about 32 values, which is less than what MIDI expects for controller channels (7-bit, 128 values). One obvious solution is simply to rescale the data up to a 7-bit range. While this may be acceptable for some applications, for many timbre controls (for example, a filter cutoff frequency) this will cause noticeable stepping. A better solution is to perform long term averaging of the data, as it is sampled at very high (250 ksample/s) speed. Similar to the breath control, simple averaging is used to derive a higher precision value for the reed displacement. This does present a problem in that the response time of the sensor is reduced. Possible improvements to the software design will be discussed in a section 3.7.

## 3.5 Microcontroller and Firmware

As stated above, both revisions use Microchip's dsPIC as the main controller. This is a 16-bit microcontroller with a large number of built-in peripherals. Of key importance are the built-in ADCs (used to read analog sensors), the built-in SPI port (used to read the DRO in revision X1), the quadrature encoder interface, and the UART (universal asychronous receiver transmitter) for MIDI transmission and debugging. The X1 design calls for a dsPIC30F4013 (in a 44-pin surface mount package), which provides all the above features minus the encoder module, while the X2 design uses the dsPIC33FJ128MC706 (a 64-pin surface mount chip). This chip was chosen for being the smallest chip (that is, fewest pins) that has both a 12-bit ADC as well as an encoder module. The dsPIC33 has essentially the same architecture as the dsPIC30, but runs at a higher clock rate while using lower power. For simplicity, both parts are run at 120 MHz using the internal oscillator and PLL.

The software is written in C and compiled using Microchip's MPLAB C30 compiler for dsPIC. The initialization procedure reads one second's worth of values for both breath and reed and stores the average (pre-multiplied by 512) for both sensors as a zero point. The main loop is driven by the state of the MIDI queue, which is also serviced in the main loop. The first task in the main loop is to check the UART transmission flag, which reports whether the UART is ready for more data. If so, a byte is de-queued and copied into the UART buffer for transmission. The next section of the main loop determines whether the instrument is currently considered playing or not playing (that is, whether the MIDI note is sounding). This is done by reading the current averaged breath value and comparing it to the previous one. If the instrument is not playing, exceeding a threshold will change the state of the instrument to playing. However, if the instrument is playing, a different (lower) threshold is used before the instrument is set to not playing. This hysteresis is realistic as a woodwind instrument takes more effort to start playing than to sustain a tone. If the instrument is determined to transition from off to on, an initial note number is determined and stored. This value is needed for pitch encoding as well as to send the note off message. Note that MIDI requires a note number for the note off (to allow for polyphony); the same note number must be given as the one used to turn on the note, even if the absolute pitch has changed, as during a glissando.

The next section of the main loop determines whether to fill the MIDI queue. If the queue has space for an entire MIDI message (3 bytes), a message is en-queued. If the instrument has been determined to change playing state from the previous section, the MIDI note-on or note-off message is en-queued. Otherwise, one of the three sensor messages (position, breath, or reed) is encoded as a parameter change (or pitch bend in the case of position) messages. The controller number for breath is 02h (typically used for breath) and 01h (modulation wheel) for reed displacement. The encoding is done in a round robin manner between the three axes so that on average the number of messages of each type sent is approximately equal. One extremely important point is that the parameter change messages are sent even if the MIDI note is off. This is required because many synthesizers take the point of MIDI note on as an initial point for all the parameters (for example, pitch bend). Without parameter change messages between the notes, the synthesizer would start playing the new note while maintaining the parameters of the old note, causing a glitch in the attack of the note.

Currently, the pitch encoding uses software emulated floating point (32-bit precision), which is possible due to the high performance of the microcontroller and the fact that only one channel must be computed. The detailed algorithm given here is only for the X2 revision as future revisions will use the linear encoder due to the reasons stated in section 3.2. The encoder module on the dsPIC reports an integer from 0 to 2500 for the full range of travel. Given that the total playing range is set to be 2.5 octaves, the fractional note number is given by

$$NN = 48 + .012n \tag{3.2}$$

The rounded integer is used as the MIDI note-on value. Notice that this number could be either larger or smaller than the fractional note number; therefore, the pitch bend may be either positive or negative. For correct operation, the synthesizer must have the pitch-bend range set to 24 semitones, which is typically the maximum allowed. This means that the entire 14-bit range encompasses 2400 cents. This at first seems coarse, but note that this is more accurate than the sensing mechanism allows (16,384 values versus 2500). However, this does limit the maximum glissando to somewhat less than the range of the instrument. Therefore, it is not possible to glide a single note for the entire length of the instrument.

The pitch-bend value can be thought of as a 14-bit value ranging from 0 to 16,383 but packed into two 7-bit values as opposed to bytes. The conversion from the fractional note number is therefore

$$PB = 8192(NN - rnd(NN)) + 8192 \tag{3.3}$$

As an example, playing halfway (50 cents) between Middle C and C-sharp corresponds to a note number of 60.5, of which the integer part is 61 (0x3D) and the fractional part is 4096 (0x20, 0x00). For the instrument to start playing this note, it would output 0xE0, 0x00, 0x20 followed by 0x90, 0x3D, 0xVV (where VV is some arbitrary velocity corresponding to the value of the breath when it first exceeds the note on threshold). The first message is the pitch bend on the first channel (note that MIDI transmits the least significant 7-bit value first). The second message is the actual note on for C-sharp.

The final section of the main loop performs the averaging on a block of samples, which is where most of the CPU time is spent. The low-level sensor reading is handled independently of the main processor execution. Here, the X2 revision also differs substantially from the X1, due to the fact that the dsPIC33 series incorporates a direct memory access (DMA) engine. The DMA engine allows a full 500 ksample/s to be sampled and processed efficiently. In order to read the two analog sensors, the DMA engine is given control of the A to D module. It is programmed to sample a block of 1024 samples (the maximum allowed, 512 samples for each channel) in a double-buffered manner. Once the 1024 sample block is completed, a pointer is set to the most recently sampled data. Because a large number of samples needs to be processed here within a short amount of time (approximately 2 ms), this averaging is all done in integer math. For the breath channel, the 512 samples are simply added, and a value corresponding to 512 times the zero point is subtracted from the sum. The sum is then shifted right by 14 bits to perform both the averaging divide and to scale the final value to the 7-bit MIDI controller range. The reed is a little more complicated; for unlike the breath sensor, the reed sensor does not read a full scale value throughout the full range of motion. The maximum change of the reed value is typically around 32 values (5 bits); therefore, the scaling factor used is a 9-bit right shift. Both the breath and the reed value are clipped to 0 and 127 to prevent accidental overflows. While the breath value essentially does not overflow, the reed value can overflow if the circuit board slips from the housing, thus changing the zero point. This software arrangement allows for the system to not use interrupts and at the same time respond approximately as fast as the sensors and MIDI allows. The latency here is approximately 2 ms, though this may be arbitrarily decreased by decreasing the DMA block size at the expense of noisier controller output.

## 3.6 Power and MIDI

One final aspect of the design that deserves explanation is the power and MIDI output circuit. The instrument uses the Yamaha WX jack in order to provide both MIDI signal and power over a single cable. This is mainly done to provide compatibility with the Yamaha VL70-m physical modeling synthesizer, which implements a form of digital waveguide synthesis. The connector used here is a five-pin mini-DIN, which provides 7 V power and accepts MIDI. Unlike the MIDI electrical specification shown in Figure 2.2 however, the synthesizer expects an open collector output instead of a current loop. The reason for this is that, unlike the case of connecting two synthesizers together (which may cause ground loops due to having multiple analog audio interconnects and independent power sources), a MIDI wind controller is simply connected at one point through a digital connection and powered by the synthesizer. The use of the open collector driver allows for the operating voltage of the controller to be different than that of the synthesizer. One problem encountered here in the early design is that the input voltage is unregulated and, in any case, quite high for most low power regulators. This puts it out of the input range of the TI TPS793 series LDO, which was used in the first revision X1 boards. This regulator was replaced by the TI REG113 series LDO, a change made possible by the removal of the need for a 1.5 V rail.

## 3.7 Future Work

Although significant improvements in the tracking and feel of the slide were made in the X2 revision, the bulk of the mechanism limits the usability of the instrument. As stated previously, the main reason for the size is the desire to enclose the slide mechanism within the instrument body without having a ribbon cable connecting to a moving reader. The X3 revision, currently in progress, addresses this issue in a completely different way, by utilizing a short-range wireless connection and a battery powered reader mechanism. Unlike the previous instrument housings, the new design has a fixed optical strip and a moving reader. This reader includes a very small lithium polymer battery, which is managed by a Maxim MAX1555C single cell battery charger. The battery charger is connected to terminals that rest on spring clips when the slide inside the instrument body is at the lowest position, which allows the battery to charge while the instrument is not in use. This battery powers both a PIC18F2331 (directly from the battery) as well as a small boost converter to power the 5 V optical encoder. Although a dsPIC would have been preferred in this configuration, the 8-bit PIC18 has significantly lower power draw and allows for a wider input voltage range (from 2 V to 5 V), removing the need for a step up/down regulator (the lithium polymer cell has a nominal voltage of 3.7 V but can drop below 3 V as the battery wears). Additionally, it draws significantly less power, requiring approximately 1 mA to run at 4 MHz, while the dsPIC requires more than 10 times this. Because the PIC18 does not do significant processing, the lack of 16-bit math support is not a significant issue. The wireless transmission mechanism is a simple LED connected to the UART of the reader's processor, which the main board reads via a high-speed photo-transistor connected to the main processor's UART. With careful mechanical design (to avoid light leakage), this setup allows for transmission speeds greater than 1 megabit per second within the instrument. One unknown with this design is the battery life of the reader. Initial estimates with a 100 mAh battery suggests a battery life of about 2 hours before a recharge is needed, which may not be enough for a practical musical instrument. Even with a highly efficient microcontroller, significant power is needed to drive the high-brightness LED (approximately 20 mA at the battery voltage) as well as the boost converter. The practicality of such a configuration, therefore, depends on the real-world battery life as well as the charge time; for example, if it is possible to significantly charge the battery during an intermission in a concert.

# Chapter 4

# CONCLUSION

This thesis described one potential implementation of an electronic continuous pitch wind controller for musical performance. We have not discussed, except in the narrow contexts of pitch and MIDI encoding, the various synthesis algorithms that may be adapted to such a controller. While the goal of the controller's design is to make it possible to interface to many diverse synthesizers, in practice many synthesizers are fundamentally not designed to be played from a continuous pitch controller. For example, one of the most popular synthesis methods currently employed is sampling, which, simply put, takes many recordings of the sound, finds the nearest recording to the pitch being played from the controller, and pitch shifts the nearest recording to match the the desired pitch. When playing a glissando using any continuous pitch controller, it is clear that there is no obvious way to switch samples, therefore necessitating a large pitch shift of a single sample, which cannot evolve in timbre. Furthermore, it is very difficult to manipulate a recorded sound in real time through control streams from the player, because the synthesizer has very little understanding of the structure of the recorded sounds. It is difficult, for example, to de-tune the attack of a recorded sound depending on the slope of the breath input, which is a reasonable thing to do in order to simulate the character of a plucked instrument. Any sort of manipulation is likely to require significant human "tagging" of each sample, as well as significant frequency domain re-sampling [13].

One area we are currently researching is a type of additive synthesis using high-Q bandpass filterbanks. This method essentially involves filtering a wide-band signal through a bank of IIR bi-quad sections, which are individually controlled (in center frequency and Q) by parameters derived from the control input. This algorithm differs from sine wave additive synthesis in that the output of each filter is not a perfect sine wave because the filter itself is relatively weak. This effect allows for better representation of acoustic instrument sounds for a given number of bands compared to the same number of sine waves because the noise characteristics of instruments (such as the breath noise of a flute) are more easily represented. Furthermore, the filter state performs significant interpolation on the output by not allowing instantaneous changes of frequency or Q, which reduces the necessary update rate for filter parameters and removes the need for linear interpolation of parameters.

We have also only hinted at the starting point for the mapping of instrument control dimensions to synthesis parameters. Discussions on generalized strategies for performing this mapping can be found in [14], and a presentation of a particular mapping for the synthesis method described previously can be found in [15].

The eventual goal of any instrument design is to design a sound that rivals existing acoustic instruments, whether through sheer emulation – as in the case of a digital piano – or by offering a unique playing method or sound. It should be noted that all acoustic instruments have a significant player base while essentially only having a single "patch" (that is, the native sound of the instrument), whereas electronic instrument often have hundreds of patches or programs. It is clear that acoustic instruments have a large appeal because current electronic instruments, despite being able to produce a great variety of sounds, cannot respond to expressive player control with the same subtle manner that acoustic instruments can. Therefore, we continue research into designing control and synthesis methods that allow electronic instruments to rival their acoustic counterparts in expressiveness.

# Appendix

# PHOTOGRAPHS OF THE INSTRUMENT

Photographs of the complete instrument may be found in the supplemental file named **photos.zip**.

# REFERENCES

[1] C. Roads, “Early electronic music instruments: Time line 1899-1950,” *Computer Music Journal*, vol. 20, no. 3, pp. 20–23, 1996. [Online]. Available: http://www.jstor.org/stable/3680817

[2] T. Rhea, “The ondes martenot, an early milestone in the development of electronic keyboards,” *Keyboard*, vol. 10, p. 14, 1984.

[3] R. H. Dorf, *Electronic Musical Instruments.* Mineola, NY: Radiofile, 1968.

[4] C. Scholz, “A proposed extension to the midi specification concerning tuning,” *Computer Music Journal*, vol. 15, no. 1, pp. 49–54, 1991. [Online]. Available: http://www.jstor.org/stable/3680386

[5] M. M. Wanderly, “Gestural control of music,” *Proceedings of the International Workshop on Human Supervison of Control in Engineering and Music*, vol. 1, pp. 101–130, 2001.

[6] L. Haken, E. Tellman, and P. Wolfe, “An indiscrete music keyboard,” *Computer Music Journal*, vol. 22, pp. 30–48, 1998.

[7] E. G. Shower and R. Biddulph, “Differential pitch sensitivity of the ear,” *The Journal of the Acoustical Society of America*, vol. 3, no. 2A, pp. 275–287, 1931. [Online]. Available: http://link.aip.org/link/?JAS/3/7/2

[8] *MIDI Musical Instrument Digital Interface Specification 1.0*, International MIDI Association Std., 1983.

[9] G. Loy, “Musicians make a standard: The midi phenomenon,” *Computer Music Journal*, vol. 9, pp. 8–26, 1985.

[10] M. Wright and A. Freed, “Open sound control: A new protocol for communicating with sound synthesizers,” 1997. [Online]. Available: http://hdl.handle.net/2027/spo.bbp2372.1997.033

[11] A. Freed and D. Wessel, “Communication of musical gesture using the aes/ebu digital audio standard,” *Proceedings of the 1998 International Computer Music Conference*, vol. 1, pp. 220–223, 1998.

[12] G. Scavone and A. da Silva, “Frequency content of breath pressure and implications for use in control,” *Proceedings of the 2005 National Conference on New Interfaces for Musical Expression (NIME05)*, vol. 1, pp. 93–96, 2005.

[13] E. B. Egozy, “Deriving musical control features from a real-time timbre analysis of the clarinet,” M.S. thesis, Massachusetts Institute of Technology, 1995.

[14] A. Hunt and R. Kirk, “Mapping strategies for musical performance,” in *Trends in Gestural Control of Music*, M. M. Wanderly and M. Battier, Eds. Paris, France: Ircam - Centre Pompidou, 2000, pp. 231–258.

[15] E. Eagan, Z. Cheng, and L. Haken, “Musical synthesis using modal filter banks for expressive performance,” qualifying exam paper, Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, 2008.