New Expressive Percussion Instruments

by

Roberto Mario Aimi

S.B., Biology, Massachusetts Institute of Technology (1997)

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2002

Author .....
Program in Media Arts and Sciences
August 16, 2002

Certified by .....
Tod Machover
Professor of Music and Media
Thesis Supervisor

Accepted by .....
Andrew Lippman
Chairman, Department Committee on Graduate Students

New Expressive Percussion Instruments

by

Roberto Mario Aimi

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
on August 16, 2002, in partial fulfillment of the
requirements for the degree of
Master of Science in Media Arts and Sciences

Abstract

This thesis presents several new percussion instruments that explore the ideas of musical networks; playing, recording, and developing musical material; continuous control over rhythm and timbre; pressure sensing; and electronic / acoustic hybrids. These instruments use the tools of electronics and computation to extend the role of percussion by creating new ways for people to play percussion alone, together, and in remote locations. Two projects are presented in detail. The Beatbugs are a system of eight hand-held networked instruments that are designed to let children enter simple rhythmic motifs and send those motifs to be developed further by the other players. Results from three workshops and performances are discussed. Preliminary results are also presented for the Remote Drum Network, a system that lets people play drums together over the internet even in high latency situations by synchronizing their audio streams and delaying them to match each player's next phrase.

Thesis Supervisor: Tod Machover
Title: Professor of Music and Media

New Expressive Percussion Instruments

by

Roberto Mario Aimi

Thesis Readers:

Thesis Reader .....

Joseph A. Paradiso
Principal Research Scientist
MIT Media Laboratory

Thesis Reader .....

Hiroshi Ishii
Associate Professor of Media Arts and Sciences
MIT Media Laboratory

Acknowledgments

Thanks to:

Tod Machover

Gili Weinberg

Kevin Jennings

Peter Colao

Tristan Jehan

Michael Lew

Matt Wright

Mary Farbood

Emily Cooper

Bruno, Clair, and Marco Aimi

The whole Toy Symphony team

Special thanks to my readers, Joe Paradiso and Hiroshi Ishii.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Throughout history, musical instruments have changed to reflect new technology. This is no less true for percussion instruments, which have always been an expression of available materials and building techniques. Since the development of electronics, many people have applied the new technology to percussion by modifying old percussion instruments and creating new ones, but most of these efforts have failed to take advantage of the unique possibilities of electronics and computation.

This thesis explores the potential for electronic percussion instruments to do things that traditional instruments can not. The concepts of musical networks; recording, playback, and variation of musical material; continuous control of timbre and rhythm; pressure sensing; and electronic-acoustic hybrids are developed through a series of new percussion instruments. Two projects, the Beatbugs, and the Remote Drum Network are presented in detail.

The Beatbugs are a network of eight hand-held instruments that are designed to let children play simple rhythmic motifs and to send those motifs to other players who can choose to develop them further or to create their own motifs. The system of motif and variation can be used to create and perform larger-scale collaborative compositions. The Beatbugs were jointly developed by myself and another MIT Media Lab researcher, Gili Weinberg as one part of Tod Machover's Toy Symphony project, which is briefly described in section 3.4.

The Remote Drum Network is a system that lets people play drums together

over the internet, even in high latency situations in which true real-time playing and collaboration would be impossible. The system uses variable delays to align the player's musical phrases.

1.1 Background

There are some aspects of traditional percussion instruments that make them different from other instruments. Because of the quick attack of percussion sounds, they lend themselves well to rhythmic playing, and it is easy for an audience to correlate the playing gestures to the resulting sound. Although the physical materials and construction of all acoustic instruments are closely related to how they sound, percussion instruments are unique in that the playing impulse is transferred directly to the resonator (often comprising the entire instrument) rather than through a complex driving system such as a vibrating string or reed. Because of this, the shape and material of the resonator has significant influence over the sound of the instrument [30].

The development of percussion parallels the development of human culture, dating back to prehistoric times, and remaining with us until the present [2]. As percussion technologies developed, new instruments have been added to the palette of possible sounds, but few instruments disappeared completely. Though some were relegated to ceremonial roles in isolated communities, other early percussion instruments have thrived for centuries and are still in use. The single-headed frame drum, first appearing in Babalyonian reliefs dating to 2000 BCE, is used today in the form of the tambourine, Irish bodhran, and the Brazilian pandeiro, among many others [22]. A thorough history of percussion is far beyond the scope of this thesis, but I would like to highlight the possible origins of percussion, and to discuss a more modern example of the development of the drum set to give a context for the brief history of electronic percussion that follows.

1.1.1 Prehistory of percussion instruments

Humans are apparently unique among primates in our ability to perceive and create rhythmic meter. The ability to entrain our movements to an external timekeeper, such as a metronome or drum, enables us to play music and dance in groups, and the ability most likely evolved in parallel with the cultural development of group dance and music rituals [6].

Percussion instruments are believed to be among the earliest musical instruments. The first instances of percussion were probably not instruments at all, but hand claps, stamping on the ground, or hitting the body [25, 30]. Any object can be struck or scraped, and percussion (the act of hitting things) was necessary both to fashion tools and to hunt game. The process of striking flint with a wood or bone baton to create a sharp edge is in itself rhythmic. Some theorize that such objects, though designed for utilitarian purposes, were among the first objects used to make music [2].

Percussion music developed from purely utilitarian roles to more ceremonial uses in groups, and the range of percussion instruments grew from the paleolithic rattles, scrapers, rubbed shells and stamped pits to include the membrane drums, xlyophones, mouth harps, and clappers found in late neolithic archaeological sites (shown in table 1.1, adapted from [33]) [25]. Prehistoric man made use of the materials around him. Early percussion instruments have been found that were made from wood, hides, bone, clay, and stone, generally representing the array of available materials at the time [2].

Idiophones	Aerophones	Membrano-phones	Chordophones
	–Early Stratum–		
rattles	bull-roarer		
rubbed shell	ribbon weed		
scraper	flute without		
stamped pit	holes		
	–Middle Stratum–		
slit-drum	flute with	drum	ground-harp
stamping	holes		ground-zither
tube	trumpet		musical bow
	shell trumpet		
	–Late Stratum–		
rubbed wood	nose flute	friction-	
basketry rattle	cross flute	drum	
	transverse	drum stick	
xylophone	trumpet		
jaw's harp			

Table 1.1: Chronology for early instrumental development

1.1.2 A more recent example: Development of the modern drum set

Skipping ahead several thousand years (and through almost all of recorded history), one of the most recent and successful innovations in percussion has been the development of the modern drum set. By the late 1800s, marching bands were being replaced by smaller ensembles that performed exclusively indoors. One drummer had to play the role of many, and some drummers developed a technique of using sticks to play the bass drum, snare, and a small Turkish cymbal mounted on the bass drum simultaneously. Others played the bass drum with a foot, freeing their hands to play the cymbal and snare. Various pedal systems had been developed that allowed the player to hit the bass drum and a cymbal simultaneously, but in 1909, Chicago drummer William Ludwig patented what would become the typical bass drum pedal design. By the 1920s, the bass drum pedal became a standard part of the drum set, which also included a Turkish cymbal, Chinese cymbal, woodblock, cowbell, and Chinese tom toms [28, 5].

Through the 1920s and 30s, a broad range of new instruments were added to the percussionist's arsenal, but only a few stayed into the 40s: the Chinese tom tom evolved into the high and floor toms that we now know, and Turkish cymbals became more common, largely displacing the Chinese cymbals. The low boy (resembling a low hi-hat) was added to emphasize the prominent backbeat of Chicago style Dixieland jazz, but quickly evolved into the more versatile hi-hat [5].

By the mid-1940s, the jazz drum kit became standard, and was used in a wide variety of musical styles. The classic drum kit consists of a pedal-operated bass drum with attached toms and suspended ride and crash cymbals, a hi-hat operated with another foot pedal, a floor tom, and a convertible snare drum with a disengageable snare [28].

Since then, drum set designers have incorporated modern materials, such as synthetic drum heads, and used new materials for drum shells such as metal, fiberglass, and acrylic. Drummers have incorporated traditional percussion instruments from

around the world. More recently, chain drive and double bass drum pedals, rototoms, and octobans have been added to expand the possible rhythmic and timbral palette further. Electronic drum pads and triggers have been integrated into many players' drum sets, and purely electronic sets have been marketed as drum set replacements [28, 5].

1.1.3 Electronic percussion

Although some electronic percussion instruments existed before 1960, notably Leon Theremin's Keyboard Electronic Timpani (1932) [14], modern efforts to incorporate electronics into percussion instruments began in the late 1960s as modular synthesizers became more common. Musicians and engineers began experimenting with attaching transducers to pads that they could hit. By plugging the resulting waveform into a modular synthesizer, they could use the trigger output to gate a synthesizer sound [10].

In 1973, Moog introduced what was possibly the first commercial percussion controller. The Moog Percussion Controller Model 1130 was a drum with a sensor in the drum head that could drive the Moog modular synthesizer [23]. The PAiA Programmable Drum Set, released in 1975, was one of the first self-contained electronic drum devices (Figure 1-1). Its sounds were made by sending impulses into almost-oscillating filters to make them ring. Also credited with being the first programmable drum machine, it featured touch pads that responded to skin capacitance. A modification was available to approximate velocity-sensitive pads by measuring changes in skin capacitance and resistance, taking advantage of the fact that the skin's complex impedance is approximately proportional to finger pressure on the pad [16].

Figure 1-1: PAiA Programmable Drum Set

With the development of MIDI in 1983, a new set of percussion controllers became available that could be plugged into any synthesizer module, making them truly generic controllers. Simmons, DrumKat, and Roland offered a variety of drum controllers modeled after drum kits, marimbas, and in some odd multi-pad arrangements like the Roland Octopad and DrumKat's eponymous DrumKat [45]. Generic drum trigger boxes that could accept a range of triggers, usually including a built-in set of drum samples, include the Yamaha TMX, and the Alesis DM series. Moving against the trend of making the controllers, trigger units, and sound sources more generic and interchangeable, Roland's V-drums, first introduced in 1997, re-introduced the idea of specialized pads for a particular trigger unit and sound source. The V-drums can measure velocity and stick position, and dual-trigger pads can detect rimshots and cymbal chokes [32].

Figure 1-2: DrumKat

Some percussion controllers have ventured further from traditional drum designs. Don Buchla has made several percussion interfaces that depart substantially from simple emulation of drum kits. Buchla's Thunder is a drum intended to be played using fingers, and it can track the position of the depression made by the finger using a clever optical system on the back of its reflective mylar drum head [27, 23]. The position and velocity information can be mapped to any MIDI control, opening up a range of sounds and mappings not previously possible. Buchla's Marimba Lumina (Figure 1-3) is a marimba-like controller that can sense which mallets struck it, as well as where on the bars it was hit, enabling different mallets to be mapped to different timbres [39]. The Mathews/Boie radio drum used capacitive sensing to track the 3D position of transmitter batons close to the playing surface [4].

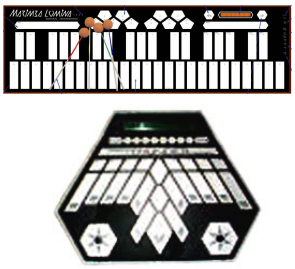

Figure 1-3: Buchla's Marimba Lumina and Thunder

Another innovative percussion instrument is the Korg WaveDrum which, unfortunately, was only sold for one year. The WaveDrum is notable in that it used the acoustic sound of the drum head to drive a synthesis algorithm, similar to the way in which the Korg G5 Synth Bass Processor uses the bass sound to drive its synth.

This blend of acoustic and electronic sound was very compelling, and by all accounts, highly responsive to subtle changes in playing [31].

Other important work includes Laurie Anderson's drum suit, which made drum triggers wearable [13], and Tod Machover's piece, *Towards the center*, which explored ideas of musical interdependence and autonomy by using a computer system to dynamically transform and remap the musical input from a keyboard and mallet controller, sometime combining the two to form a "double instrument," where control over different aspects of a complex musical output is split between the two performers [21]. Tina Blaine and Tim Perkis' Jam-O-Drum combined conventional drum triggers with projected graphics in a tabletop installation that encouraged group improvisation [3]. The Rhythm Tree, part of the "Brain Opera" project by the MIT Media Lab, is possibly the world's largest electronic percussion instrument, consisting of over 300 multi-zone and velocity-sensitive drum pads, each with an individual LED to provide visual feedback [24].

1.1.4 What's missing

With a few notable exceptions (including some mentioned above), what has not been well explored is whether technology can do more than replace or enhance traditional percussion instruments, but enable new ways of playing music that were not possible before.

In fact, electronic percussion instruments have the potential to do things that traditional instruments cannot. The instrument can record exactly what has been played, and it can modify, repeat, and develop recorded phrases. If connected in a network, it can send sound or control information to another instrument. Using such a network, people can play music together over a large distance or in the same room. One can take advantage of these strengths to make instruments that let people play music in new ways. Specifically, we can make instruments that blend the responsiveness, tactility, and intuitive playing style of traditional percussion with the networked collaboration, transformation of rhythmic patterns, and timbre manipulation possible using electronic instruments.

With these enhanced potentials come some associated risks. The interface can become so abstracted from the sound it makes that the player no longer feels in control. The ability to modify musical patterns after they have been played has no physical analog in traditional instruments, so it can be difficult to make these interactions seem intuitive.

1.2 Motivations / Approach

Given these potentials and challenges, I see my work as being guided by three general questions. Can the ability to modify, store, analyze, and reproduce the performance information enhance the musical experience? What are the creative implications of connecting instruments in a network? Most importantly, how can the physical interface best reflect the potentials of electronic instruments while maintaining the rich yet clear musical mappings and intuitive playing style found in acoustic instruments? Out of these general concepts and other constraints, some specific design goals can be applied to my two primary projects, Beatbugs and Remote Drum Network.

The goal for Beatbugs was to create an instrument for children and professional musicians that encourages creating and sharing of rhythmic motifs. It needed to provide feedback so that the player could understand what he or she was doing, and the sound needed to be of high quality for a concert audience. Since all players were together in the same space and could hear each other, network interactions needed to reinforce the connections between players without interfering with their normal connections as an ensemble, and those interactions had to be conveyed clearly to both the players and an audience. The interface needed to be simple enough to learn and understand in a week-long workshop, but rich enough to allow for expressive playing.

For the Remote Drum Network, the goal was to develop a system that allows people to play drums together in real time over long distances . Since the players were not in the same physical location, the network interactions needed to provide the entire musical connection between players while tolerating latencies associated with

communicating over a large distance. Rather than designing a new interface, my aim was to make the interface as minimal as possible to keep the player's focus on playing acoustic drums and listening to the other players.

The Beatbugs and the Remote Drum Network were informed by and based on several musical devices that I made over the past two years. Kaossfly, Echo Drum, Pressure Tambourine, and Drum Network were early sketches made to investigate the issues of electronic percussion. Together, they provide the foundation for the later projects and for future work.

Chapter 2

Early Work

When I first came to the Media Lab, I was interested in exploring acoustic sound, and how it could be combined with electronics and computation to make new hybrid instruments. Much of this interest was an outgrowth of my frustration with synthesizers and how difficult it is to control and manipulate their sound. I though that by coupling synthetic sound to a physical surface, I might be able to gain a degree of acoustic control over that sound. Through experimenting with different transducers, I became interested in using the head of a drum as a speaker while still letting it function acoustically, with the hope that it might provide a good way to integrate electronic sound into an acoustic instrument. This led to several drum projects, but also got me thinking about percussion in general, and about how people play music together. What follows is an account of a few early sketches that explore some of the ideas that became the core of the Beatbug and Remote Drum Network projects which will be described in the next two chapters.

To drive the drum head, I tested a variety of transducers, but two approaches proved most successful. One method (figure 2-1) was to remove a speaker's cone, spider, and basket, leaving the magnet and a loose voice coil. By attaching the coil to the drum head and the magnet to a truss under the head, the drum head

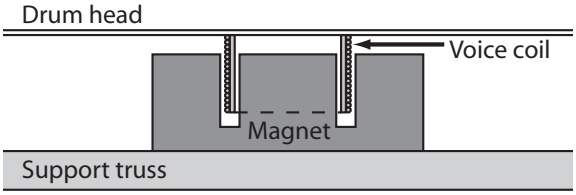

Figure 2-1: Using a voice coil to drive a drum head

could replace the cone and spider as a surface that both moves air to make a sound and keeps the voice coil aligned in the magnet's groove. Although this gave the most output and best bass response, some drawbacks of this design were that each drum needed a truss to support the magnet and it was difficult to align the voice coil so that it moved freely and didn't scrape against the magnet. A second, simpler approach was to use the driver from a flat panel speaker, based on a design by NXT [19]. The NXT driver is essentially a conventional speaker without the cone and basket, leaving only a voice coil, spider, and magnet. I attached the voice coil directly to the underside of the drum head and left the magnet free in the air. The inertia of the magnet was enough to give the voice coil something to push against. Although the NXT transducer could be mounted in locations that the spiderless design could not, it had less bass response and lower overall output than the first approach. With two adequate methods for turning a drum head into a speaker, it was time to think about musical applications.

2.1 Echo Drum

Figure 2-2: Echo drum

As a first test of the drum transducer, I made a drum that could repeat what was played on it by emitting the delayed sound from the drum head (figure 2-2). The Echo Drum consisted of a conventional MIDI drum trigger (Pintech RS-5) [17] and an NXT driver mounted under the head of a children's djembe [18], connected to an Apple Macintosh running a simple MIDI delay line written in Cycling 74's Max environment [9]. Hitting the drum would cause a delayed click to be played back through the drum head, and the resonance of the drum head and cavity made the click sound more like another drum hit. Setting the delay to 200 ms, the drum would echo anything played on it, but unlike a conventional echo unit, the echo was subject to the same control (by muting, pressing, letting it ring) as a player would have over acoustic drum hits. When the trigger threshold

was turned up high enough, the undamped drum would repeat echos by retriggering from its own output. Putting a hand flat on the drum head could completely mute the echos. This control over the echo made it feel inviting to play. The sense that the drum responded to hand pressure after it was hit was consistent with the way a drum can normally be muted, so it did not require any new playing techniques. One drawback of this design is that the delay effect was fixed, and there was no way to directly influence the electronic sound. The overall timbre was still that of a drum.

2.2 Pressure Tambourine

Through working on the Echo Drum, I became interested in having more control over the audio being played through the drum. I wanted to keep the gesture of pushing on the head to mute the echos, but I thought it would be interesting if the player could sustain a note by pushing harder on the head. The challenge was to sense pressure on the drumhead while still maintaining its acoustic and physical properties so it could still be played like a conventional drum. Since the drum head was under tension, pressure on the head caused more displacement of the head, so it seemed reasonable to sense head displacement as an indirect measurement of pressure.

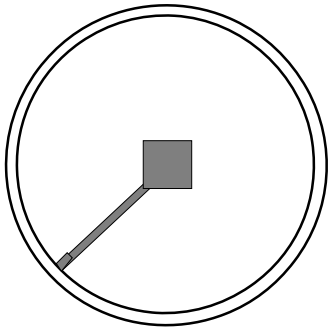

Figure 2-3: Underside of the drumhead with ground electrode

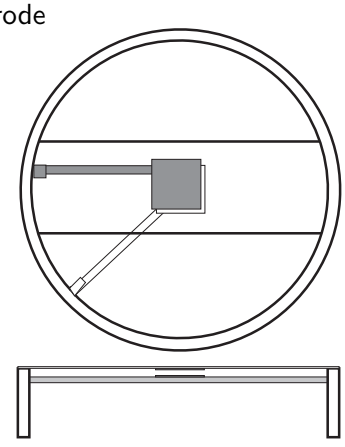

Figure 2-4: Underside of the drumhead with sensing electrode and plastic beam. Also in cross section

One way to measure displacement is to put an electrode plate on the object being measured, and fix another parallel plate to some reference point. The two electrodes form a capacitor with the air serving as a dielectric. When the plates move closer to each other, the capacitance between the plates increases [12]. By timing how long it takes to charge or discharge the capacitor through a large resistor, one can measure the capacitance. The closer the plates are to each other (and the

harder the player is pressing on the drum), the larger the capacitance and the longer it takes to charge or discharge.

I fitted a tambourine with a drum trigger and an NXT driver, as in the Echo Drum, and added electrodes to measure pressure. A Max patch received the trigger and sensor data and drove the Reason [26] software synthesizer and effects. The sound coming out of Reason was routed back through the drum. I mapped displacement of the drum head to delay time, giving the player some control over pitch, since the pitch of sound already in a delay line increases or decreases with changing delay time. To sense displacement of the drum head, I made a 2" x 2" square ground electrode on the underside of the drum using adhesive copper tape (figure 2-3), and ran a thin strip of the tape to the edge of the drum where it connected to the measurement circuit. I added a plastic beam to hold the sensor electrode, to which I affixed a 2" x 2" square of copper tape connected via a copper strip to the edge of the drum. The sensor electrode was on the side of the plastic beam facing the drumhead (figure 2-4). The beam was mounted so that the two electrodes were about 7 mm apart when the drumhead was at rest.

I used a simple two-pin PIC circuit to measure capacitance between the electrodes (figure 2-5). Pin b2 (35) goes to the sensor electrode on the plastic beam, and a 3MΩ resistor connects pins b2 (35) and b1 (36). The electrode on the drumhead was connected to the circuit ground. I chose the PIC 16F877 because I originally thought I would need to use analog inputs, but digital inputs work fine for this application. The PIC 16F84 would have been adequate, and a lot smaller.

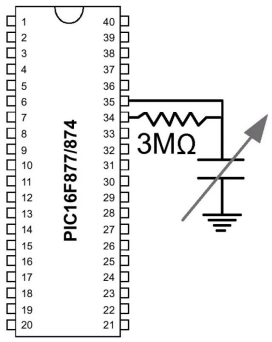

Figure 2-5: Simple PIC circuit

The role of the PIC was to measure how long it took to charge and discharge the capacitor, and to report that result via MIDI to the computer. Starting with both pins b1 (pin 35 in figure 2-5) and b2 (pin 34 in figure 2-5) set low, pin b2 gets set as an input. Pin b1 gets set high, and the PIC starts counting until pin b2 reads high. Next, the process is inverted: pin b2 is set to output and set high, and then set back

to an input. Pin b1 is set low, and the PIC counts until pin b2 also reads low (see table 2.1). The raw measurement is of low resolution, between two and three bits. To increase resolution, the measurement process is repeated 1000 times, and all the counts are added up to one number (usually between 18000 and 36000). The PIC converts that number to two 7-bit bytes and sends it out the serial port as a 14-Bit MIDI pitch bend. On the computer, a Max patch scales the 14 bit number back down to a range of 0-127, which can easily be mapped onto MIDI controllers.

The sensor output had some dead spots in the travel of the head. By putting weights on a 2" diameter disk, I was able to plot the sensor output versus the applied force (figure 2-6). The stair-step pattern is probably due to the relatively low resolution of the counter, and because there isn't enough noise to let the averaging process increase the effective resolution. One way to increase resolution would be to pulse pin b1 so that the capacitor charges more slowly, allowing more times through the testing loop before pin b2 flips. A faster clock-speed chip would also help.

step	**b1**	**b2**
1	output 0	output 0
2	output 1	input
3	wait until b2 reads 1	
4	output 1	output 1
5	output 0	input
6	wait until b2 reads 0	
7	repeat	

Table 2.1: Pins b1 and b2 measuring capacitance

The overall musical result was that hitting the drum produced an echo that came back from the drum itself. By pushing on the drumhead, the player could shorten the delay time and create a sustained drone sound. Since changing the delay time also changed the pitch of sound in the delay line, the faster the head was pushed, the greater the change in pitch, so players could get a glissando effect. Although the changing pitch effect was not originally intended, after hearing it in preliminary tests, I was able to enhance the effect by changing the range of possible delay times.

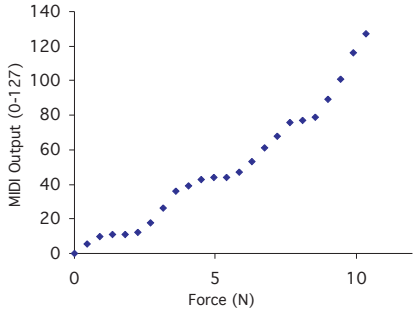

Figure 2-6: MIDI output vs. force

2.3 Drum Network

Figure 2-7: Drum Network

Inspired by Gili Weinberg's work on musical networks [42], I became curious as to whether the drum sensing and transducer work could be applied toward group playing. Working with Gili, I developed a system by which two drums could be connected to each other (figure 2-7). As with most of our work together, my contribution was primarily the physical system of drums, sensing, and triggers, while Gili worked on mapping the triggers and controls to sound. I attached drum triggers to two drums which I connected to a computer running Max. As was the case for the tambourine, the audio was routed back through the drums, but this time the output of one drum could be sent to the other. One drum was the same djembe used in the Echo Drum project, another was an Irish bodhran outfitted with a pressure sensor of the same design as the one used in the Pressure Tambourine.

We decided to use a simple echo model where whatever was played on the bodhran was delayed and sent to the djembe one measure later, and likewise from the djembe to the bodhran. Additionally, pressing on the head of the bodhran could control the pitch of the echo, allowing for a more interdependent manner of playing.

The djembe used the NXT driver, but the bodhran lent itself well to the spiderless driver because it already had wooden cross pieces under the drum which could be used to support the magnet. I removed the basket, spider, and woofer, leaving only the magnet and the voice coil. I attached the magnet to the cross pieces under the drum, and glued the coil tube to the underside of the head in an arrangement similar to figure 2-1. With careful alignment, the coil and the magnet didn't touch, allowing the drum head to ring normally. Although the big magnet made the drum substantially heavier, it was able to produce significant bass and loud overall output, especially since the bodhran had such a large head. The ground electrode for sensing pressure was mounted under the bodhran head, and slightly offset from the center of the drum to make room for the magnet. The sensing electrode was mounted on an

arm attached to the crosspieces under the drum.

Although both players were working together to create a musical phrase, the rhythmic and melodic roles were distinct enough from each other that the players didn't step on each other's toes. Unfortunately, although the musical roles were clear, and it was easy to play, it proved quite challenging to play well. The biggest problems were confusion between the direct acoustic sound of hitting the drum and the echoed sound, and it was easy to get into a musical rut of playing against your own echo rather than with the other person. This had the strange property of encouraging obvious and rather boring rhythmic phrases without any larger-scale structure. It was also hard to reproduce consistent pressure on the drum head, making control of the pitch quite difficult.

2.4 Kaossfly

In parallel to the drum projects, but in the purely electronic realm, I wanted to experiment with more accurate continuous controllers to modify rhythmic sound. The Kaossfly (figure 2-8) was a variation on work by Gili Weinberg called the musical Fireflies shown in figure 2-9 [41]. Since the Kaossfly is so closely based on the Fireflies, it is worth briefly describing how the Fireflies work.

Figure 2-8: Kaossfly

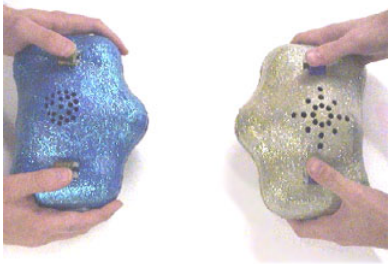

Figure 2-9: Fireflies

Fireflies were hand-held wireless instruments that could communicate using infrared beacons. A player could enter a simple sequence of accented and unaccented notes using two buttons. The sequence would play back in a loop and a second layer could be added in a new timbre. When two Fireflies were in range, they synchronized tempos to play together. By pressing one of the buttons while the Fireflies were pointed toward each other, they could trade the timbre associated with that button, for example, the hi-hat sound of Firefly 1 would be exchanged with the

snare drum sound of Firefly 2. When the Fireflies lost sight of each other, they would keep their new timbres.

Although the patterns that one could enter were constrained to simple accented sequences without regard to timing or how hard the buttons were hit, in user testing, children recognized that they had created the loops, and felt ownership of them. The role of the network was minor, limited to synchronization and timbre trading, but it didn't impede playing, and the interactions were clear.

One thing that was missing from the Fireflies was any continuous control that could make the monotonous general MIDI sounds more expressive. The Kaossfly was an attempt to address this deficiency. The Kaossfly was essentially a Firefly with a much better speaker housed in a new enclosure that also held the parts of an effects processor (a Korg Kaoss pad, hence the name Kaossfly) with a touch pad mounted on the outside of the enclosure. The small buttons of the Firefly were replaced by switches with large flaps that could be slapped or tapped, making the interface feel more percussive, even though they were still simple intermittent switches.

Adding an effects processor and touch pad allowed players to apply an effect to the simple Firefly pattern and to control different effect parameters such as filter cutoff and resonance. Delay with variable rate and feedback was one of the most successful effects since, at the extremes, it became a sound of its own, rather than just a transformation of the Firefly sound.

Still, some features were missing from both the Firefly and the Kaossfly. Players could only create simple sequences of loud and soft sounds; velocity and rhythm information was ignored. The Firefly / Kaossfly network required direct line of sight to work together, meaning that only pairs of players could interact. The interaction between players was also discrete—once the timbres were traded, the interaction was over until someone wanted to trade timbres again. The Kaossfly was also too big and heavy to be held comfortably, and worked better as a tabletop instrument.

The Beatbugs, which will be presented in the following chapter, attempt to address all of these issues, while scaling up the network to eight nodes, substantially improving the quality of the sound, and making the physical instruments robust enough to be used in workshops and on stage.

Chapter 3

Beatbugs

The Beatbugs are a network of bug-shaped percussion instruments that are designed to encourage a group of people to play music together. In their final version, the Beatbugs were played by eight children in workshops, open houses, and concerts. The general idea behind the Beatbugs is that players would be able to enter rhythmic patterns, manipulate and develop them, and collaborate with other players to perform and create music. The process of developing physical and software prototypes helped shape our ideas about how an application could best embody that concept.

The Beatbug project was a collaboration with Gili Weinberg. In general, I was responsible for the physical instruments, microcontroller programming, and development of the system. Gili was responsible for the Max software, he created the sounds, and he composed a piece for Beatbugs that was performed as part of Toy Symphony [38]. However, it was a true collaboration—we discussed practically every major design choice, and the physical system and the software were made to work together.

Much of the direction of the Beatbug project was guided by Gili's interest in musical interdependence between players, which he has studied extensively and explored through building many instruments [41, 44, 42]. Multiplayer musical interdependence refers to the way musical systems can redistribute musical control between players. A simple example would be a two-player system where one player controlled pitch, while the other controlled rhythm (as in the Drum Network). A more complicated example would be Gili's composition, *Squeezadelic* for the Table of Squeezables, where

five pressure-sensitive balls control the timbre and rhythmic stability of individual accompaniment parts while collectively controlling the timbre and scale of a sixth "melody" ball [44]. Interdependence between musical instruments can heighten the natural connection between players in a group, but it can also make it more difficult for players to understand what they are controlling.

The other critical collaborator on the Beatbug project was Kevin Jennings, a music educator and researcher at Trinity college and Media Lab Europe in Dublin, Ireland. Although Kevin joined the project late, he was instrumental in developing the pedagogical workshops and gave us invaluable guidance in development of the various Beatbug applications and modes. Kevin also ran all of the preliminary workshops and the bulk of the Beatbug workshops and rehearsals associated with the Toy Symphony performances in Dublin.

Some of the initial design goals of the Beatbugs were first expressed in a set of instruments called Simple Things. The Simple Things, created by MIT Media Lab researcher Josh Strickon, were hand-held devices capable of simple on-board synthesis that could be networked via infrared. Audience members at the 1999 Sens*bles symposium [35] used the Simple Things and a network of Sega Dreamcast game systems to "grab notes" played by an electric cello and manipulate them [47]. The Simple Things were an important influence on the form of the Beatbugs, while the Beatbug interface and musical function were inspired by the Fireflies and the Kaossfly. This chapter will describe both the "first generation" Beatbugs, which were early working prototypes of the Beatbug system, developed in September 2000, and the "concert version" Beatbugs which have been used in concerts and workshops since October, 2001.

3.1 First generation Beatbugs

Having some specific constraints on the project helped focus development. Several goals were present from the outset. Based on people's positive impressions of the Simple Thing, we wanted a hand-held music toy to be part of the Toy Symphony

project. The role of Beatbugs in that project will be described in section 3.4. From our experience with the Kaossfly, we also knew that we wanted more continuous control over the sound, and the instrument needed to be smaller if kids were going to hold it. If the toy were going to be featured in a concert with orchestras, it needed to have good sound. Based on preliminary tests of the Fireflies and Kaossfly with kids, we learned that one of the things they liked most about those instruments was that they were creating the music from scratch rather than just shaping preexisting music. Even though those toys only stored patterns of accented and unaccented notes, not rhythm or velocity, the children still recognized the stored pattern as their own.

In order to achieve these goals, we modified both the controller interface (figure 3-1) and the musical application. Physically, the first generation Beatbugs were a pair of baseball-sized, egg-shaped controllers. To provide a better tactile interface, the two buttons of the Firefly were replaced with two PZT piezo discs that responded to being hit rather than being pressed. Continuous control was added in the form of two resistive bend sensor "antennas" [11]. To make it easier for other Beatbug players to hear the sound, the speaker was moved from the top of the instrument to the front, facing away from the player. The whole Beatbug was dipped in a blue rubberized coating to make it more inviting to hold and hit the sensors. The system and application could support two Beatbugs.

Figure 3-1: The first generation Beatbugs

The application for the first generation Beatbugs consisted of an entry phase similar to that of the Fireflies, in which accented and unaccented notes were entered in a sequence, and a new real-time manipulation phase in which the sequences could be played at different speeds, volumes, and timbres, while new sounds could be triggered. In this phase, hitting the piezo pads triggered new drum sounds that could be played over the looped sequence. Unlike the first application, these new sounds did not create another looping layer; rather, they allowed players to play along with the loop in real time, letting the instrument be played longer, and providing a contrast to

the highly metronomic loops. By bending the two antennas, the player could change speed, volume, and timbre parameters of the looped sequence.

The left antenna provided discrete control of two tempi (normal and double speed) and continuous control of the cutoff of a resonant bandpass filter. The right antenna provided continuous control of reverb mix, volume, and the resonance of the same bandpass filter.

Since each antenna was independent and both could be controlled simultaneously, the player had access to any combination of the two timbre parameters. The looping sequence could be stopped at any time by pressing a stop button, and a new pattern could be entered. In the multi-player mode, the antennas of one Beatbug could control the sound of another Beatbug, while each player could still play his or her own real-time sounds over the pattern. By providing continuous control over the other player's sound, we increased the musical interdependence between players, allowing more interpersonal interaction than was possible with the Fireflies' discrete "timbre trade" (described on page 26).

Knowing that we would need good quality sound for a performance, we decided to produce the sound remotely on a synthesizer, and then send it back to be played through the Beatbug's speaker. This arrangement made for a fairly complicated system, shown in figure 3-2. The Beatbugs were tethered to a patch box by DB-9 cables which carried two drum trigger signals, MIDI, speaker-level audio, and power on five pairs of conductors. Each Beatbug contained a PIC microcontroller which measured the bend sensor positions and stop button state, and sent them as MIDI data to a MIDI interface. The drum triggers were connected to a Yamaha TMX MIDI drum module (also connected to the MIDI interface). Gili Wein-

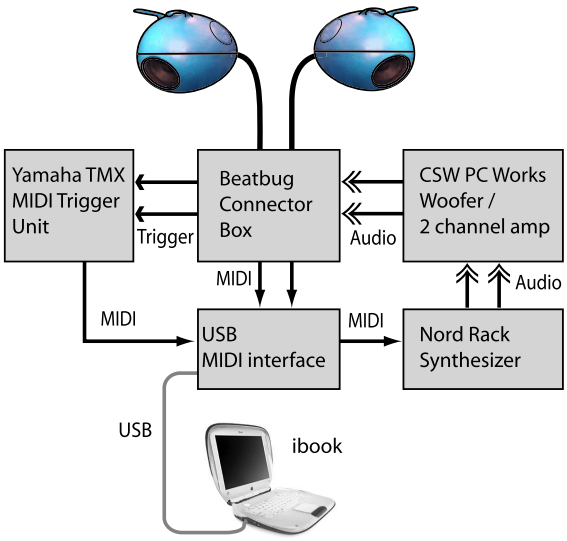

Figure 3-2: First generation system schematic

berg wrote the application in Max which ran on an Apple ibook. The application read the MIDI drum trigger and sensor data and triggered the Beatbug sounds (also designed by Gili) on a Clavia Nord Rack virtual analog synthesizer. Two-channel output from the Nord went to a Cambridge Soundworks PCWorks woofer and two-channel amp, where it was amplified and sent via the patch box to the speakers in the Beatbugs.

There were some advantages to the first generation Beatbug in comparison to the Firefly or Kaossfly. The physical interface was easier to hold and hit, and the piezos made hitting the Beatbug a richer tactile feeling. The bend sensor antennas were inviting to play, and had good resolution; adding continuous control over timbre made the resulting sound more dynamic and interesting. The general topology of having sound synthesis occur remotely, and sending the audio back to the Beatbugs allowed for high quality sounds, but also made it clear clear which Beatbug was associated with which sound.

There were still some problems. Since the bend sensors were covered in the same blue rubberized material that coated the whole Beatbug, they had some hysteresis problems, and were slow to spring back to a neutral position. Our original idea was to have a Beatbug stop playing a loop when a player hit both piezo disks at once. This turned out to be too confusing and difficult to convey to new players, so I added a "stop" button on the back of the Beatbug. Since the stop button was an afterthought, it didn't blend well with the rest of the case design, and seemed out of place in the application. Some other problems were that the Beatbugs didn't give any visual feedback to reinforce what was happening musically, and that they determined accents by reading whether the player hit the left or right piezo, limiting players to a binary choice of accented or unaccented notes rather than measuring the velocity of their hits. Similarly, it was frustrating for some players that the system only recorded the sequence of hits, not the rhythm. The interdependence between the Beatbugs was limited to having one Beatbug's antennas control another Beatbug's sound, and it wasn't clear how that kind of interaction could scale to accept more players.

3.1.1 First generation Beatbugs meet a large toy company

After seeing a demonstration of the first generation Beatbugs, a toy company became interested in commercializing them. Gili and I met with their engineers and designers to develop a prototype and to see if kids liked it. Throughout the development of the Beatbugs, we had been thinking about how they could be made into toys, but kept our designs oriented toward the specific requirements of workshops and performances. My original expectation was that the toy designers and engineers would use the Beatbugs to inspire a new design that was better-suited to be sold as a toy. Instead, they tested our existing and unmodified two-Beatbug system directly against other music-oriented toys.

Children were introduced to several commercially available toys and also shown our prototype Beatbugs paired with a non-working model of the toy company's Beatbug design. One child accurately described it as looking like a "frog wearing sneakers and you hit it in the lungs." The children were then asked to choose which toy they would want to buy from among all of the toys, including the Beatbugs. The Beatbugs proved less popular than several of the toys, including Bop-it Extreme, Simon, and the Christina Aguilera MusicClip. Based on the results of the test, the toy company decided to abandon further development of the Beatbugs.

What was most surprising to me was not that the test focused on whether the children would want to buy the toy, but that it did not test whether the toys were enjoyable to play. Meeting with the designers and engineers, it was clear that the cost and marketing constraints that are present in the toy industry place severe limits on what can be done. Music chips that are cheap enough to be used in toys typically store four-bit ADPCM (Adaptive Differential Pulse Code Modulation) audio at very low sample rates, which can yield intelligible speech, but very low fidelity music. It might not be a problem for a toy: the most popular toys in the test all had very low quality audio, and none of the children mentioned it as a flaw.

The designers had several proposals for improving the Beatbugs as toys. They felt that the bulk of the Beatbug functionality needed to be in its "stand-alone" mode,

since it is unlikely that other children already own Beatbugs. They also disliked the idea of a musical network and suggested that songs could be traded on small flash cards, and they wanted to add a genre switch to make it play in a rap, country, rock, or classical music idiom. I knew that the Beatbug design would need to change to become a good toy, but I did not anticipate the degree of the disconnect between my own goals for the design and the goals of the toy designers.

As discouraging as this episode was, I still believe that the ideas behind the Beatbugs could be used in a toy. By more thoroughly understanding what can be achieved using minimal hardware and by understanding the toy market better, we could develop a toy that is inexpensive and immediately appealing, but still offers musical depth.

Secure in the knowledge that the Beatbugs were not going to become the next Pokemon, we returned our attention to developing a new Beatbug design for workshops and concerts.

3.2 Beatbugs: concert version

For the concert version of the Beatbugs (figure 3-3), we tried to address the deficiencies of the first generation by changing the physical design of the controller, the design of the overall system of hardware and software, and the musical application with the goal that the Beatbugs could be used in workshops, open houses, and concerts.

Figure 3-3: Beatbugs (Concert version)

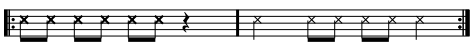

Figure 3-4: A typical eight-beat rhythmic motif (from *Nerve*)

One key idea was that we wanted to allow players to enter simple looping rhythmic motifs (figure 3-4) and to develop each other's motifs sequentially, rather than simultaneously. If two players were to control parameters of the same motif, it could be difficult for each of them to identify their own contribution. We decided that a better approach would be

to pass the entire motif to another player, so that only one player could manipulate it at a time. By choosing to either replace a motif with a new one, or to develop it further, players can control both the low-level rhythmic content and the high-level trajectory of the music, while maintaining a coherent interaction.

Gili developed several prototype applications based on passing motives between larger groups of players. To enable this kind of group playing, we needed to have a system that could support many more Beatbugs. Through playing the instruments ourselves and having kids and adults test them, we arrived at three main modes of operation, which will be discussed in section 3.3.

The physical instrument also needed to be designed in a more easily manufacturable and robust way, and there were several aspects of the interface that needed to be changed to work with the new applications. The process of redesigning the Beatbug system went through many iterations, and at each step, the physical design was informed by the software, and vice versa.

3.2.1 Design decisions

What follows is a brief description of some of the design decisions that occurred between the making of the first generation Beatbug and the concert version (described in detail in section 3.2.2). After making the first generation Beatbug, we considered a broad range of designs, from thumb piano interfaces to something resembling a video game controller. The Bug / egg shape of the first Beatbug was chosen mostly at random, but after building many models in different shapes, I decided to return to the basic look of the first generation. I wanted the Beatbug to look like it could have a personality to make its musical behaviors seem more autonomous even though they were being controlled by a central system.

I thought about having the speaker face upward at the player, but decided that it was important to be able to point the sound at the other players in order to emphasize the sharing aspect of the application. It also made the speaker look more like a mouth, which caused the Beatbug to appear more anthropomorphic.

We decided that an interface with one piezo that could respond to how hard it was

hit would be simpler than the two-piezo system used in the first generation. Rather than having the player hit the piezos with his or her thumbs, we moved the piezo to the top, so that the Beatbug could be hit more like a drum. We also considered having the antennas be used as rhythmic input devices by plucking them like the tines of a thumb piano, but opted for the drum metaphor because it had a bigger playing gesture that would be more visible to an audience and to other players.

I briefly considered making the Beatbugs wireless, which would have avoided the inevitable tangle of cables, but abandoned the idea because of a number of issues. The quality of sound that we could make on board was limited to wavetable synthesizers found in PC sound cards, prohibiting any continuous control over timbre. Sending control signals wirelessly would have been possible, but sending audio back to the Beatbugs would have needed either a potentially bulky and expensive real-time digital system such as 802.11 to stream audio back to the Beatbugs as in Adam Smith's WAI-Knot instrument [36], or small radio receivers (tuned to eight different frequencies) in each Beatbug, which would be at the mercy of the local RF environment. To make a system robust enough to tour, I decided to use cables instead of making it wireless.

It was important that each Beatbug be interchangeable so that if one broke during a concert or workshop, another could be substituted in its place. MIDI was a good protocol to use because many multi-port interfaces are available, removing the need to run the Beatbugs on a bus, which would have required a unique ID for each Beatbug.

In the end, I also decided to keep the general system topology of the first generation Beatbugs in which a computer running Max was responsible for all of the behavior, musical mappings, and sensor calibration while the role of the Beatbug processor was limited to sending uncalibrated control information and receiving messages to change lighting. We decided to use a software synthesizer and multi-channel audio interface because we found that we could create equally good sounds without needing a bank of external synthesizers. As in the first generation, audio from the computer was sent back to be played from each Beatbug's speaker. The downside to this approach is that the Bugs can not make any sound away from the central system.

I experimented with a variety of multi-segment LED and vacuum-fluorescent displays, with the thought that the application could be more of a self-contained game. We abandoned this approach because we wanted to emphasize making music as a group, and because we wanted to keep the player's attention on each other rather than on their own instrument. Ultimately, I decided instead to use LEDs to light the whole top of the Beatbug, which would be visible to the other players and the audience, and not just the one player. White LEDs could be flashed to indicate which Bug was playing, while LED clusters could smoothly change their color from green to yellow to red. The computer had complete control over turning on and off the white LEDs and controlled the brightness and the color of the clusters. This gave us some flexibility to find lighting schemes that best illustrated what was happening musically even as the musical application was still being developed.

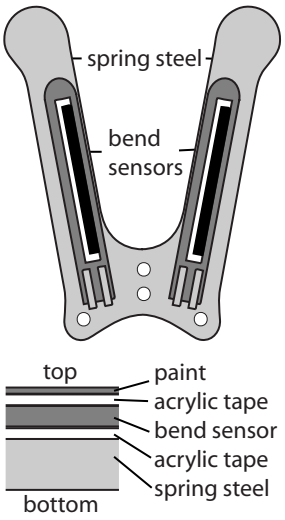

Figure 3-5: Bend sensor antenna diagram

Having determined the basic system topology and interface, there was still some design work left to do on the antennas and the Beatbug shell. The first generation Beatbug antennas suffered from hysteresis problems and had a tendency to kink, damaging the bend sensor. To solve these problems, I laminated the bend sensor to a spring steel substrate (figure 3-5).

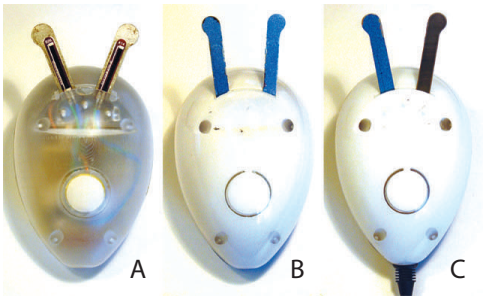

Figure 3-6: Three Beatbug antenna designs

The antennas went through three main design iterations. The first set of antennas, shown in figure 3-6a were short and wide. We found that they were too hard for younger players to bend, and that they were hard to play with one hand. The second antenna design (figure 3-6b) was thinner and more parallel. Although they were easier to bend, they were too far apart to be bent by children with short fingers. I redesigned the antennas in more of a V shape (figure 3-6c), so that they were close together at the Beatbug body (where shorter fingers can reach them) and further

apart at the ends for players with longer fingers.

The design of the Beatbug shell also went through several iterations, some of which are shown in figure 3-7. Figure 3-7a shows a clay model that the first 3D computer models were based on. The top is a piece of clear plastic from a toy easter egg, which saved my having to sculpt the top half. Our first computer models were drawn in Rhino, a 3D CAD program. Rather than importing our model into their system, the manufacturer [8] made their own model from scratch based on the clay model. The computer model was machined from a block of high density foam (figure 3-7b), which gave us a chance to test the feel of the Beatbug before printing an STL (stereo lithography) model. Urethane was poured over the STL to make a flexible mold that would be used to cast the Beatbug parts. The first parts were cast in a quick-set acrylic and painted grey to give them a better finish (figure 3-7c). We also had some cast in clear acrylic, shown in figure 3-7d. Since the STL model still had a stairstep-like finish, tiny grooves were transferred to the mold and cast in the clear acrylic, making it appear cloudy.

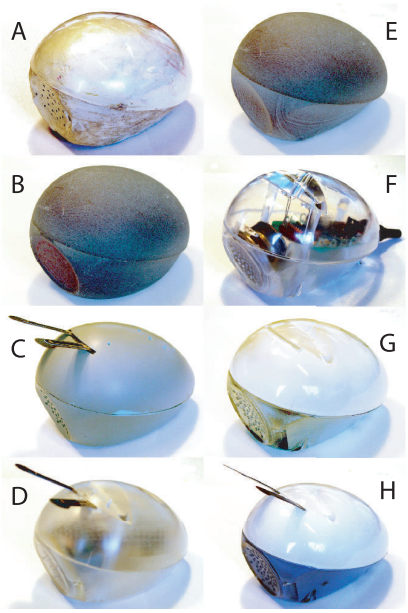

Figure 3-7: Development of the Beatbug shell

Using the clear and the grey shells, we assembled the Beatbugs to be tested in preliminary workshops with kids. Holes were drilled in the grey Beatbugs to make the LEDs visible to the players. Based on these workshops (described in more detail in section 3.4.1), we discovered that the bottom half of the Beatbugs were too big for people with small hands to hold. Another model was made that had a narrower base to accommodate smaller hands (shown in high density foam in figure 3-7e and compared to the wider design in figure 3-8). This time, the

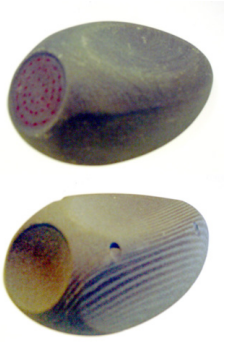
Figure 3-8: Models of the original base design (top) and the slimmer base (bottom)

new STL model was sanded and smoothed to remove the grooves. This resulted in a clear Beatbug (figure 3-7f). Although it was nice to see the circuit board inside of the Beatbug, the shell did not scatter the light of the LEDs, making them hard to see off axis, and glaringly bright on axis. We had a very light coat of white paint applied to the underside of the top to scatter the light (figure 3-7g). I tried different colors for the bottom half of the Beatbug. Metallic Silver, shown in figure 3-7g tended to highlight surface imperfections on the inside of the shell. Metallic Blue (3-7h) was the final choice for the Beatbug bottom.

3.2.2 Beatbug physical construction

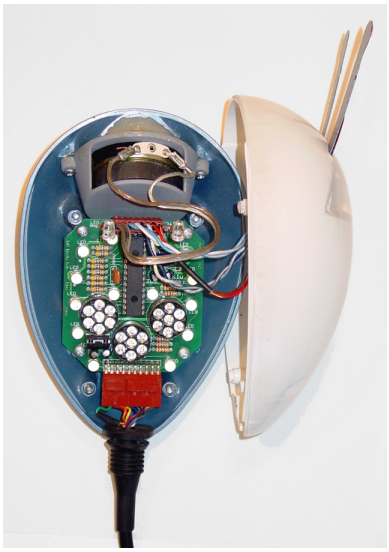

Figure 3-9: An open Beatbug

The final Beatbug design maintains the overall bug / egg shape of the first generation, but is slightly larger and has a sculpted bottom half to make it more comfortable to hold (figure 3-3). The Beatbug shell is made of a top and bottom half of clear cast acrylic, made for us by a fabricator [8] based on our physical and 3-D models. The bottom half was painted with a blue metallic paint on the inside (visible from the outside) and houses the speaker and the PC board. The region of the bottom half that covers the speaker has an array of small holes to let the sound out. The top half, which holds the antennas and the piezo trigger, received a very light layer of white paint on the inside to diffuse the light from white LEDs mounted on the PC board. The antennas were laser cut from 0.010" thick spring steel to which we laminated 2" Flexpoint bend sensors [11]. The antennas were then coated with blue metallic paint to match the color of the bottom half of the Bugs. Cable strain relief was provided by a custom rubber boot.

The PC board (seen in figure 3-9) in the Beatbug holds a PIC 16F876 microcontroller that reads the bend sensors, controls LEDs, and communicates with the central system via tail-like cables that carry MIDI, trigger, audio, and power. The antennas,

trigger, and speaker connect to the PC board via a Molex header for easier assembly. To give visual feedback to the players and the audience, LEDs highlight both the discrete hits and continuous manipulations. White LEDs are arranged in a circle around the perimeter of the board to light the top of the Beatbug. The PIC also uses pulse-width modulation to control the brightness of three green and red-orange LED clusters, which can give additional feedback to the player by glowing green, yellow, or red.

Bend sensor information is read by the PIC's 10-bit analog-to-digital converters, averaged over 256 measurements, and encoded as 14-bit pitch bends (all calibration occurs on the computer). The white LEDs are controlled by note-on and note-off messages, and the green and red-orange clusters are controlled by pitch-bend messages.

3.2.3 The Beatbug system

Software and application

While the Beatbug processor is responsible for operating the sensors and LEDs, a central computer system controls the actual musical interactions and behaviors. The application represents a significant departure from the application used in the first generation Beatbug. Players can play rhythms with different velocities, and send their motifs to other players to be developed further. This kind of sequential interdependence avoids some of the confusion that can occur when multiple players are simultaneously manipulating different parameters of the same music—it is clear that each player is the only person manipulating the sounds of his or her Bug. The application will be described in more detail under the subsections for each mode.

The "brain" of the system was written in Cycling 74's Max environment by Gili Weinberg and Gautam Jayaraman. By controlling all of the Beatbugs' behavior from the computer, we were able to quickly experiment with a much broader range of interactions than would have been possible if we had reprogrammed the Beatbugs each time we wanted to change a behavior. Similarly, all sound synthesis also occurs on the central computer system and plays through each corresponding Beatbug's speaker. (For performances or large-scale workshops, we supplement the direct sound

from the Bugs with a 2 or 8-channel PA). For the software synthesizer, we chose Propellerhead's Reason, which gave us a broad palette of timbres and effects with continuous control over many parameters of the sound. Gili also designed each of the sounds and timbre mappings used by the Beatbugs.

The rack

Eight Beatbugs can be plugged into one central rack, shown in figure 3-10. The rack consists mostly of standard, off-the-shelf equipment including an Apple Macintosh G4, a Mark of the Unicorn 2408 audio interface, two Emagic Unitor MIDI interfaces, a Lectrosonics PA-8 8-channel amplifier, an Alesis DM-5 Drum trigger unit, and a Yamaha 03d Mixer. The only non-standard device is a custom patch box, which provides power to the Beatbugs and converts each Bug's 10-pin Neutrik Minicon connector to MIDI in, MIDI out, trigger, and audio in. The entire system, including the mixer, and the computer, fits in a single Mixer rack.

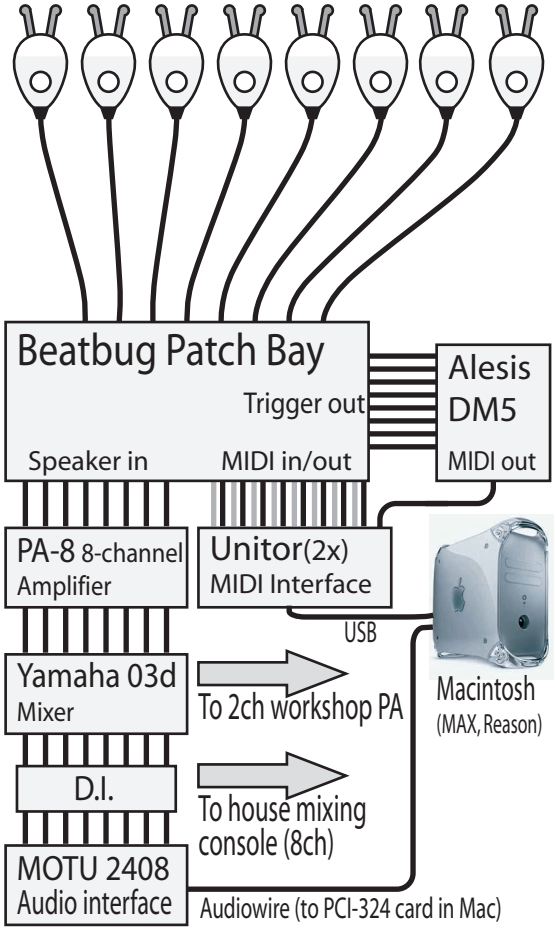

Figure 3-10: Beatbug rack schematic

The eight channels of Reason audio coming from the Macintosh's 2408 Audio interface connect to the 03d mixer via a bank of direct inject boxes that can split the audio to connect directly to the mixing board in a concert hall. All eight channels are mixed down to stereo for the workshop P.A. system, but the unmixed audio comes back out the four bus sends and four aux sends of the mixer into the Lectrosonics PA-8, which amplifies the sound to drive the speakers in the Beatbugs (connecting through the patch box).

When a Beatbug is hit, the analog signal from the piezo drum trigger is sent to the Alesis DM5, which outputs a MIDI note to the Macintosh via one of the Unitor Midi interfaces. The note's velocity is proportional to how hard the Beatbug was hit, and each Beatbug triggers a different note. This lets the Max patch also know which Beatbug was hit. Bending the antennas causes the PIC in the Beatbug to output a 14 bit MIDI pitch bend message through the patch box to the Unitor. This message is scaled down to 7 bits by a simple Max patch that automatically calibrates the Beatbug antennas. Sending a high-resolution, uncalibrated signal to the computer makes it easier to compensate for any changes in bend sensor output over time, and lets the antennas be easily replaceable.

3.3 System function

The Beatbug system lets players participate in the process of making and performing music in a variety of ways. Based on the results of preliminary workshops (described in section 3.4.1), three different interaction modes were developed, each one offering successively more sophisticated control of the musical output. The modes, called "Free-Play," "Drum-Circle," and "Snake," are introduced to children during workshops and open houses.

3.3.1 Free-play mode

This mode is designed to introduce the players to the Beatbugs. Each Beatbug works like a regular drum: hitting it makes a sound, and hitting it harder makes the sound louder and changes the timbre. All eight players can experiment, hitting the Bugs freely to familiarize themselves with the Bugs' response and sound (each Beatbug has a different timbre). They can also use this mode to practice playing rhythmic motifs which will be used in the other modes. The bend sensor antennas are not typically used in this mode, but they are still active, meaning that a good player could simultaneously manipulate timbre while hitting the Beatbug. In fact, Gili has gotten quite good at playing them in this mode. Although the Beatbug was designed

to be hit on the top, the entire shell is actually sensitive, letting a player manipulate the antennas with one hand while hitting the underside of the Beatbug with the other.

3.3.2 Drum-circle mode

Drum circle mode presents a more complex musical and social interaction and requires a session leader who, in addition to playing a Beatbug, also conducts and manages the group. The leader starts the session by generating a metronome beat (based on the tempo of the first four hits, or by choosing from a predefined setup.) While the metronome is playing back, the leader can enter a rhythmic pattern, drumming the Beatbug for a predefined number of measures (usually two 4/4 bars) after which the system automatically plays back the quantized recorded pattern in a loop. The quantization algorithm nudges the notes towards the closest quarter note, eighth note or quarter note triplet. When the entered pattern is played back (causing the white LEDs to flicker off as each note is played), the leader can manipulate the pattern by bending the two antennas (causing a proportional color change in the multicolor LED clusters).

The left antenna is used to transform the timbre through a predefined combination of filters, low frequency oscillators, frequency modulators, noise generators, and envelope parameters using Propellerhead's Reason Subtractor synthesizer [26]. Each Beatbug had its own distinct timbre and timbre transformation, letting players get used to controlling a particular sound. The right antenna adds rhythmic ornamentation to the pattern by controlling the level and feedback of a delay line. Every twelve measures, players are assigned delays of different rhythmic values including quarter notes, eighth notes, sixteenth notes, and triplets. We chose to use delay for the rhythmic manipulation since we believe that it lets players transform the pattern in a way that is still recognizable as having developed from the original pattern. Changing the pattern's individual notes might have made the motif sound too different from the original, obscuring the "motif-and-variation" aspect of the interaction.

When the leader feels that his variation is ready, he can hit his Beatbug again, which randomly activates another Beatbug in the network. The chosen Bug lights

up and its player can add a complementary rhythmic motif, which is looped and quantized in the same way. The new player can then manipulate his or her pattern, controlling different timbre and rhythmic parameters. As the session progresses, more and more players are randomly chosen to add their personal patterns to the drum circle and to add their own manipulations. The most recent Bug always plays louder than the others in order to highlight the new motif. After all the Bug players have entered their patterns, the system waits for a simultaneous hit by all eight players (conducted by the session leader) which brings all eight motifs up to the same level. Players are encouraged to create musical dialogs of call and response with each other, and they can continue to manipulate their patterns until the next simultaneous hit, conducted by the leader, which ends the music.

3.3.3 Snake mode

Snake mode, the most complex of the Beatbug interactions, forms the foundation for the piece performed in the concerts. Here, players can explore the musical network by creating motifs, sharing their motifs with others, and by reshaping the motifs of their peers. In this mode, after the first player enters the first motif, both the motif and the timbre associated with the first Bug are automatically sent to a different random Bug. which immediately begins playing back the received loop. The receiving player can decide whether to develop the motif further (by manipulating the timbre and rhythmic antennas) or to keep it (by entering and sending a new motif to the group). The function of the timbre and rhythm antennas are the same as in Drum-circle mode.

If a player decides that the received motif is ready and does not need further manipulation, he can enter a new pattern (in his Bug's default timbre). In this case, he keeps the received (transformed) pattern in his Bug at a soft background level, while his new pattern is sent to another random player, who becomes the new "head of snake." If the receiving player decides that the motif is not ready he can further manipulate it and hit his Bug to send his transformation to the next random Bug. Each time a pattern is manipulated by a new player, the antennas control different aspects of the timbre; for example, one player might control cutoff of a filter, while

the next controls resonance. The transformations are recorded and layered in each cycle until the pattern is considered complete and a new pattern is entered. Each player faces the same two options (develop the pattern, or create your own) when receiving a motif until all the players have entered their patterns (and have stored a quiet pattern in their Beatbug) The system then randomly groups different numbers of players in twos, fours, and all eight, highlighting the way the different patterns sound against each other.

We found that having the system randomly choose who a motif was sent to encourages the players who were not playing at that moment to pay attention to where the motif went, since it was a surprise and there was always a chance that they could receive it. In contrast, for *Nerve* (described in section 3.4), we decided instead to predetermine the order of which motifs were sent to which players. We felt that it would be easier for the players to rehearse if the general sequence of the piece was the same each time.

We found the sequence of modes to be important. By slowly building the complexity of the interaction, workshop leaders were able to focus the group on learning the new functionality of each mode. Free-play mode introduces the sounds of the Beatbugs and how to play simple rhythms on them. Drum circle mode introduces looping and using the antennas to manipulate timbre and rhythmic ornamentation. Snake mode adds the ability to send loops to another player and introduces the system's role in providing an underlying musical structure.

3.4 Beatbugs as part of Toy Symphony

The Beatbugs were developed as part of Tod Machover's Toy Symphony project. Toy Symphony brings together orchestras, children, musical toys (such as the Beatbugs) and a Violin Soloist playing a new "Hyperviolin" [48] in a series of concerts to "bridge the gap between professional musicians and children, as well as between audience and performers."[38] The children playing and singing in each concert are from the host city. Toy Symphony has been performed in Berlin, Dublin, and Glasgow, with other

performances expected in the United States and Japan. In the full Toy Symphony concert, the Beatbugs are featured twice, once in Gili Weinberg's piece *Nerve*, and once in Tod Machover's piece *Chorale*.

Nerve

The short piece *Nerve* was composed by Gili Weinberg as one particular manifestation of the Beatbug network [43]. Written for six children and two adult musicians playing Beatbugs, the piece follows the general parameters outlined in Snake mode. A player hits a Beatbug to start a constant bass pulse and then taps out a simple rhythmic motif. The motif is sent randomly to another player, who either reshapes the timbre and adds rhythmic ornamentation before sending the motif to another random player, or keeps it and plays a new motif, which is also sent at random to another player.

Polyphony grows as each player eventually enters a new motif until all eight Beatbugs are playing in the background. Although the motifs and play order are pre-composed, the players are free to manipulate the antennas in any way they choose. The piece ends in a Finale section in which the computer randomly groups different numbers of players for improvised solo sections. First in twos, then in fours, and finally with all eight, the original patterns are juxtaposed against each other, culminating in all of the patterns being played together.

Chorale

Tod Machover's composition, *Chorale* is a much larger piece of music that makes use of the whole orchestra, the Hyperviolin, a children's chorus, and music toys, including the Beatbugs. In *Chorale*, the Beatbugs usually operate similarly to Free-play mode, where each Beatbug hit triggers a sound. In some sections, an echo of the Beatbug sounds is gated by a keyboard player, while in other sections, the antennas are used to control the mix of several tracks of audio. Gili programmed some modes while I was responsible for the others and for integrating them into one master patch driven by keyboard triggers.

3.4.1 Preliminary workshops

Figure 3-11: Child playing a Beatbug in a workshop at the Media Lab

We held preliminary workshops with nine and ten-year-olds at the MIT Media Lab and at Media Lab Europe (MLE). The workshops were designed by Kevin Jennings, Gili Weinberg, and myself. Kevin was also the workshop leader. Over the course of these workshops, we developed the three main Beatbug modes, described in section 3.3, which were designed to help the workshop leader gradually introduce the features of the Beatbugs. These first workshops also helped us refine the application and the design of the Toy Symphony workshops which would precede each concert. The MLE workshop was much longer (almost three months). In total, thirty-six children were selected by a local children's center [1] to participate in the MLE workshops, in sets of eight at a time. The children in the Dublin workshop generally had more musical background than the children at MIT.

After getting a chance to experiment with the Beatbugs in Free-play mode and to hear their different sounds, children were invited to create short (four-beat) rhythmic motifs, first by clapping and then by hitting the Beatbugs. When all children were comfortable at this level, motif length and tempo were increased. Relationships between timbre and rhythm were highlighted to guide the children to create motifs that were appropriate to a particular timbre. The children were then introduced to Drum-circle and Snake modes. One surprise was that most of the children preferred to keep playing the same motif throughout the workshop rather than making a new one.

Children were encouraged to listen carefully to their own motifs and those of others in the group to develop an awareness of what elements exist in their sound environment. Listening skills, such as the ability to hear and perceive a single voice in a multi-part texture, were practiced by manipulating the antennae and directing the child's attention to the part of the texture that was changing. To maintain coherence,

the child who was "head of the snake" at any given time was encouraged to act as a conductor and indicate who should manipulate antennae by pointing the Bug at another child. Although initially self-conscious, the children quickly embraced this idea and were willing to take responsibility not just for their own musical part but also for giving direction to their peers in performance [43].

Seeing how kids played with the Beatbugs in a workshop situation, we learned several things:

- It was important to start with simple clapping exercises to get a sense for how much experience the group had with rhythm and to allow children to focus on the rhythmic patterns without being distracted by the technology.
- We quickly found that it was critical to give the workshop leader control over which Bugs were active. At the MIT workshop, one kid in particular kept hitting his Bug despite our efforts to get him to stop and listen to what the other kids were playing.
- The Beatbug cables needed to be longer so that the kids could be arranged in a semicircle while maintaining enough slack so they could still move and gesture with the Beatbugs.
- The bottom of the Beatbug was hard for kids with smaller hands to hold. I redesigned the bottom half to make it narrower at the base.
- Children played the Beatbug antennas in two main ways, using two thumbs, or using the first and second fingers of one hand. We decided that it was easier for the children to move from hitting the Bug to manipulating the antennas if they bent the antennas with their fingers.
- We discovered that the original antenna design that was robust enough for months of lab demos was quite easy for kids to break, prompting a redesign.

These workshops provided a good opportunity for debugging and refining the application well before the performances.

3.4.2 Toy Symphony Beatbug workshops and performances

To introduce the children to the instruments and to rehearse the music, each concert was preceded by a week of workshops and rehearsals in the host city. Toy Symphony has been performed in Berlin, Dublin, and Glasgow. As the situation in each city

was quite different, I will describe what happened in the workshops and performances city by city.

Toy Symphony Berlin

In Berlin, workshops were held in practice rooms adjoining the Sender Freies Berlin (SFB) concert hall. We were lucky to have had the help of a class of music education students from the Universität der Künste (UdK) Berlin, led by Constanze Rora, who were interested in the Toy Symphony workshops. Two student teachers, Judith Hoch and Sonja Fournes, helped with the Beatbug workshops.

We started the week with Gili leading the workshops, and having the music teachers translate, but the teachers quickly understood both the system and our goals for the performance, and took progressively more initiative until they were effectively running much of the workshop. Our original plan was that the teachers would play the Beatbugs with the children for the first half of the week, and then they would be replaced by two percussionists from the orchestra who would play in the last rehearsals and in the concert. We found that the childen had become accustomed to following the cues from one of the teachers, so we decided to have one teacher and one percussionist run the rest of the rehearsals and perform with the children in the concert.

The children were encouraged to develop a system of gestures to indicate if the "head of the snake" wanted accompaniment from the Beatbugs who had already captured quiet loops by entering their own motifs. The head would point his Beatbug at another player while manipulating the antennas and then pull back, encouraging the other player to move their antennas in response. By taking turns, the children also avoided a common problem that had emerged in previous workshops where everyone would manipulate their antennas all the time, making the background sound more constant and without any significant changes over time.

Before the concert, the audience got a chance to try out the Beatbugs and other music toys as well as traditional instruments from the orchestra in a two-hour open house. The orchestra members did a great job helping kids play their orchestral

instruments, and it was nice to see the new and traditional instruments presented together. Gili developed a special Beatbug mode for the open house in which one motif was repeatedly passed and manipulated, since entering motifs proved difficult for some audience members.

Dublin

In Dublin, we had the advantage that the eight kids who would play the Beatbugs in concert had been selected from the thirty-six kids who participated in the preliminary workshops at MLE, so they had months of more free-form (non-rehearsal) experience with the Beatbugs. Since they were already quite comfortable with the instruments and application, the rehearsals could focus on learning the motifs for *Nerve* and presenting the piece on stage.

Figure 3-12: A Beatbug player showing a child how to play a Beatbug after a Demo concert

Workshops were held at The Ark, a children's cultural center in Dublin [1]. Unlike children's museums which house content that children come to see, The Ark is built around a workshop model in which children come to participate in creative activities. This made The Ark an almost ideal place to hold Toy Symphony workshops. In addition to the rehearsal workshops, there were several open houses which featured mini-workshops for the general public, in which children got a chance to play the Beatbugs in Free-play, Drum-circle, and Snake modes. There also were "Demo concerts", in which the Beatbug players performed and introduced the three modes. After each Demo concert, the children who had been rehearsing the Beatbugs showed children from the audience how to play them, and let them try some of the different modes (figure 3-12).

Based on our experience in Berlin, we decided to have one workshop leader and one percussionist play in the concert, so that the kids could perform with an adult who had been involved in all of the rehearsals. As in Berlin, we introduced the idea that the "head" of the snake could gesture to the other Beatbugs to indicate that

he or she wanted accompaniment. Over the course of the workshops, these gestures became more stylized and theatrical, and the children added a "wave" motion at the end of the piece.

At the end of the week of rehearsals (and after months of more general Beatbug workshops), I was surprised to find that the children wanted to keep playing the Bugs and rehearsing the performance, and they insisted on doing a final run-through after we (the adults) were tired and ready to quit for the day.

Two open houses were held in the concert hall lobby, one before the concert and one during intermission. The open house before the concert was quite busy, but during intermission the lobby was so packed with people that the open house application used in Berlin didn't work very well. Since people couldn't see or hear that it was someone else's turn, they got confused when their Bug didn't do anything. They didn't listen to each other. We changed the application back to Free-play mode, which at least gave people feedback when they hit the Bugs, but didn't convey how the Bugs worked together. People were excited about playing the Bugs, but they didn't gain any new insight into how they had worked on stage.

The Dublin performance also marked the premiere of the second movement to Tod Machover's *Toy Symphony* piece, called *Chorale*, for orchestra, Hyperviolin, children's chorus, and music toys. The children's chorus played the Beatbugs using a special version of Free-play mode in which they could trigger percussive sounds and control the mix of other sounds using the antennas.

Glasgow

Although the children from Glasgow had much less musical background than the children in either Dublin or Berlin, the rehearsals started strong. The week before the concert workshops started, the Beatbug players were invited to perform a very short (1.5 minute) version of *Nerve* on the BBC children's television program, Blue Peter (figure 3-13). Even though they had only two days of rehearsal before the show, the children did very well and were excited to be on TV on a show they all watched.

The Blue Peter performance left the children in a good position to begin rehearsals of the full version of *Nerve*. Rehearsals were held at the Sacred Heart primary school in Bridgeton, a working class suburb of Glasgow, for one hour each day during the regular school schedule. Two musicians—a bass player and a trumpet player—from the orchestra participated in all of the rehearsals, and also showed the children their own instruments. Based on our experience in Berlin and Dublin, we decided that it was more important that the adults playing Beatbugs be excited about working with kids than that they be trained percussionists, and by requesting only percussionists, we had been unnecessarily excluding other musicians who would have a good rapport with kids.

Having heard of the playing gestures and the "wave" that the Dublin kids used, the Glasgow children pushed these gestures to an extreme by holding the Beatbugs as high and as low as they could. Gili and I were concerned that it would be distracting to the audience, but the musicians and kids insisted, and their intuition was probably correct. It looked smaller on stage, and it gave a nice visual climax to the piece.

Figure 3-13: Beatbugs on Blue Peter

What the Glasgow kids lacked in musical experience, they made up for in motivation. The Blue Peter performance helped get them focused on learning the motifs, and gave them a real head start when the rehearsals for *Nerve* began.

3.5 Outcomes

Over the Toy Symphony workshops and concerts, some problems presented themselves:

- The spring steel antennas turned out to be easy for the children to fatigue and break.
- Insufficient strain relief made it easy to accidentally unplug the header from the Beatbug circuit board by pulling hard on the cable.

- The multicolor LED clusters were bright enough to be seen by the players, but in the performances, it was hard for the audience to see them.
- Some audience members thought that there was a technological reason why the head of the snake gestured to ask the other players for accompaniment, assuming that there was some special wireless communication between Beatbugs, instead of just visual communication between the players.
- Although having the speaker face forward made it easier for other players to hear a Beatbug that was pointing toward them, it was harder for the players to hear their own Beatbugs.
- Some timbres were hard to hear once all of the Beatbugs were playing, making it difficult to tell what effect the antennas were having on the sounds.

In other ways, the Beatbugs were successful:

- Children and adults were able to understand even some of the most complicated modes.
- Children enjoyed playing the Beatbugs and wanted to keep playing them even after months of workshops
- Audiences seemed to understand the basic interaction between Beatbug players.
- The LEDs provided good visual feedback, and made the Beatbugs more inviting to hit.
- The Beatbug shells were durable and resisted extreme abuse.
- Having a speaker next to each player in the concert to amplify the sound of their Beatbug helped the audience localize the sounds from the players. This varied from hall to hall, and there were some seats, particularly the balcony area in the Dublin concert, where spatialization was lost because of bad room acoustics and masking from a monaural overhead cluster of speakers.

Technical observations

Making a system robust enough to tour and withstand extreme use by kids is hard. The plastic shells proved to be pretty unbreakable, but the antennas did break too often, and it was possible to pull the cable header from its socket with a hard tug on the cord. Gluing the rubber boot onto the cable should address this problem. The antenna design is a bit more difficult to fix. The next antenna design is wider where it meets the body of the Beatbug, which hopefully will avoid stress concentrations

that can lead to fracture. Using a slightly softer spring steel as a substrate should also help resist fractures, but it still needs to be hard enough to provide a good range of elastic deformation so that it can spring back to the same position each time.

Through working on the Beatbugs, I gained a new respect for how useful MIDI or similar protocols are. It made it much easier to use a remote computer to create the sounds and behaviors, while letting the microcontroller code be as simple as possible. On the other hand, it also reinforced how difficult it is to make good musical mappings between the playing gestures and the resulting sound. Looking back at the first drum project, the Echo Drum, there was something very intuitive and expressive about playing it that was missing from the Beatbugs.

The general approach of putting the computer between the Beatbug input (trigger, antennas) and the output (sound, LEDs) made it possible to continue development through the workshops. We were able to make changes to the Beatbug behaviors as new ideas emerged. It also made it easier to tailor different applications for disparate situations such as Demos, open houses, and Demo concerts. Having the sound come from each Beatbug also helpful in those situations because it made it easier to tell which sound was coming from which Bug. In the concerts, the eight monitor speakers gave similar localization cues, but the internal speakers were too small to be useful.

Musical and educational considerations

Nerve presented several conceptual challenges. The structure of the piece was intended both to demonstrate the interactions to the audience and to highlight what the system and the children were doing, but it also had to work as a piece of music in its own right. We wanted to give the players a sense that the music they created was their own, and that their gestures were meaningful, but we had the system control things like tempo, quantization, who the patterns were passed to, and the general sequence of different modes. The players controlled the timing of key events by hitting a Bug to pass the pattern to someone else, and by hitting the Bugs simultaneously to trigger the different pairings at the end of the piece. Each significant event was initiated by the players, not the system, but the order of those events was determined

by the system.

Although one can think of *Nerve* as a particular manifestation of the Beatbug system, the system itself was designed from the beginning with a performance in mind. The roles of composer, instrument designer, and player are therefore somewhat blurred. *Nerve* only specifies which particular motifs are played and their order, but allows the players to improvise accents, timbre variation, and rhythmic ornamentation, and to control the timing of major events. In comparison, Snake mode shares all of the underlying rules of *Nerve* but also lets players create their own motifs, and chooses which player a motif is passed to at random.

While musical success is difficult to measure objectively, *Nerve* was well received, mostly due to the children's performance and to Gili's ability to create a dramatic structure for the piece. A review of the Glasgow Toy Symphony performance reads, "A Beatbug ensemble from Sacred Heart Primary gave a balletic performance of Gil Weinbergs Nerve, throwing complex rhythmic surprises between one another like a game of pass the parcel." [40]

The Demo concerts in Dublin were more successful that I had expected. The space at The Ark was the right size for the event—the concerts had an informal open house atmosphere, but they were still performances, and their smaller, more intimate nature made it easier to understand what was happening. They were a good opportunity for the children to get used to performing, and it was a better demonstration of the Beatbugs than one could get in a normal open house. Often in open houses, there isn't enough time to let people develop any skill at playing Beatbugs, so they miss understanding what people who have practiced can do with them, and there is a disconnect between what they see in the concert and what they experience in the open house. At the end of the Demo concerts, audience members could ask the children how to play the Beatbugs, and get a chance to try them, giving them a chance to learn about the instrument directly from a player. Going forward with Toy Symphony, it might be worthwhile to try the Demo concerts again.

While there is no doubt that the children developed their performing and listening skills through their participation in workshops and performances, the specific contri-

bution of the Beatbugs is less clear. One could argue that any intensive musical exercise could produce similar results, regardless of the instrument. What makes the Beatbugs different from other instruments is the way they can encourage group interaction and collaboration, but they still depend on a skilled workshop leader to guide those interactions. Kevin Jenings describes his observations of the workshops:

> In the course of the workshops there was clear development in the children's performance at all levels. Stability in entering rhythm patterns against a pulse and also against a complex shifting texture, ability to deal with syncopation, use of accent and shaping of motives all improved considerably. Use of rhythmic and timbral manipulations became increasingly subtle and pointed. Interpersonal interactions such as making eye contact, looking, turning and pointing in order to facilitate musical events became completely intuitive and contextualized. When participants were asked about their learning experience in comparison to traditional music classes, many pointed out concepts such as the communal music making and peer-to-peer musical interaction that are rarely addressed in the early stages of learning to play an instrument. Others talked about being more aware of the other players in the group, listening to and following each other. [43]

Although the conditions were quite different in each city, all three sets of Beatbug players performed very well. I was particularly surprised and impressed by the children from Glasgow, who had much less musical background than the children from Berlin and Dublin, and who had a shorter workshop schedule (a fraction of the workshop time that the kids from Dublin had). They worked very hard, and having the Blue Peter performance was a great motivator. I felt that their enthusiastic presentation of *Nerve* in the Toy Symphony concert made it the best of the three Beatbug performances.

This was the largest of the projects represented in this thesis and accounted for the bulk of my work for the past two years. Development was pushed the farthest by having real deadlines for performances, by showing it to people throughout the development process, and by observing how they made music with it. Getting the Beatbugs out of the lab and into concert halls and schools gave many more people an opportunity to see and try the Beatbugs, and gave us a unique chance to see how they work in the real world.

Chapter 4

Remote Drum Network

4.1 Introduction

Although the Beatbug system focused on connecting players together in the same space, it does not address the question of what happens when players in different physical locations attempt to play together over a network such as the Internet. There are currently few ways for people to play music together over large distances. One of the biggest problems with playing music with other people over the Internet is latency. Even though there are high bandwidth connections between many locations, the latency is often too high to allow people to play music together in real time. If there is too much lag, players can have difficulty staying synchronized. The Remote Drum Network is a musical application for players in different locations that can tolerate high latency.

4.1.1 Latency

One way to think about network latency is to consider how far apart two musicians would have to be to experience the same latency due to the speed of sound in air, as is shown in equation 4.1 [20].

$$\text{separation} = 344 \text{ m/s} \times \text{latency} \tag{4.1}$$

Ignoring stalls, the network latency between my computer at home and a computer at the Media Lab is 6 milliseconds (ms) one way, which would correspond to approximately 2 meters. Adding another 10 ms for A/D and D/A buffering, the total equivalent distance would be 5.5 meters, which is not an unreasonable distance for performers in an ensemble.

Typical one-way network latency between a machine at the MIT Media Lab in the United States and Media Lab Europe in Ireland is approximately 50 ms, though peak latency can easily be double that. 50 ms is equivalent to two musicians standing 17 meters away from each other. Add the 10 ms A/D and D/A delay, and the equivalent distance grows to almost 21 meters, which is much further apart than most musicians play (both because of latency and physical constraints).

4.1.2 How have people tried to solve the latency problem?

Asynchronous systems

One approach to avoid latency problems is to send an entire piece of music to the other person, let them overdub their parts, and send it back.

The Rez Rocket application, made by Rocket Networks [29], is one example of an asynchronous system that lets people add or edit tracks in a piece of music, and send the updated piece back to their collaborators. Rez Rocket works with several digital audio and MIDI sequencing software packages including Protools and Digital Performer. The song files are sent to a central server which can be accessed by everyone working on a particular project when they request an update. Although it goes a long way toward creating a distributed studio environment, it does not give any sense of playing music in real time and improvising with other people. Each musician in effect plays to a static backing track, with no feedback from the other players until after the track has been recorded and copied to the server.

Phil Burk's WebDrum uses a different technique in which multiple players control a shared drum machine interface, and work together to edit a loop of music [7]. A particular track can be "owned" and edited by only one person at a time to avoid

conflicts that could occur if two people were attempting to manipulate the same track simultaneously. Each user sees the same pattern as it is being edited, but the metronomes on the client machines are not synchronized. If a player modifies a pattern by adding a note, for example, and the command is delayed by the network, the tracks stay synchronized and the new note is heard the next time through the loop. WebDrum also features a chat facility to socialize and discuss musical strategy. Although the actions of the other player occur in close to real time, the drum machine interface forces players to input rhythms graphically rather than playing them, which inhibits the more spontaneous gestural aspects of music such as dynamic, timbral, and rhythmic variations.

Playback Synchronization : Nagano Olympics

Another approach to playing music from remote locations is to synchronize the various audio streams by delaying them different amounts when they are played back. This approach was used in the opening ceremony of the 1998 Olympic games in Nagano, Japan. *Ode to joy* from Beethoven's 9th Symphony was performed by 88 orchestra members, 3,100 chorus singers, and eight soloists in seven locations spanning five continents. Seji Ozawa conducted the Nagano Winter Orchestra and the Tokyo Opera Singers at the Nagano Prefectural Cultural Hall. Audio from those ensembles and video of his conducting was sent by satellite and ISDN to the remote choruses. The choruses sang along, following Mr. Ozawa's conducting. Audio from the choruses was sent back to Nagano where it was buffered, and each stream was delayed to match the stream with the greatest delay using a special device made by NHK called the Time-Lag Adjuster. The original audio and video from Nagano was also synchronized with the new material. Once synchronized, a stereo mix was sent, along with video, to Minami sports park, where it was played back over loudspeakers and was joined by a 2,000-voice choir in the stadium [34].

Although this technique works well in a situation where one person is in control, the drawback of this approach is that the communication is only one-way. Mr. Ozawa never got to hear how the remote choruses responded to his conducting, and he was

unable to respond to them.

In the case of the *Ode to joy*, as with many kinds of music, it would be nearly impossible to close the loop. Since tempo and harmony change between phrases, and the phrases themselves are different lengths, hearing another player's delayed phrase would be confusing and out of place when heard at the same time as the next phrase. But there are other kinds of music where it might work better.

Why a drum circle might work differently than *Ode to joy*

In a typical modern drum circle, the problems introduced by delayed feedback are less apparent: the tempo is fixed or changes slowly, and one phrase tends to resemble the next. Delaying a player's sound by a full phrase might not be as confusing or unmusical in the drum circle situation as it would be in *Ode to joy*.

A modern drum circle is typically a spontaneous, self-organized event in which people gather to play drums. People tend to bring their own instruments, and often sit in an circlular arrangement to see and hear each other better. Although there is usually not an explicit leader, some players may take a leading role in establishing a beat or in shaping the direction of the group. The purpose is purely recreational—although professional percussionists may join in a drum circle, it is also accessible to amateurs, and it is not intended to be a performance. There is no set repertoire; rather, players engage in collective improvisation. Mickey Hart explains, "The drum circle is not a professional ensemble ... The ultimate goal is not precise rhythmic articulation or perfection of patterned structure, but the ability to entrain and reach the state of a group mind. It is built on cooperation in the groove, but with little reference to any classic styles" [15].

The innate and uniquely human ability to entrain our gestures to an external beat [6] is one part of what helps make drum circles so accessible to amateurs. The other part is that, unlike playing a more complicated instrument like an oboe, nearly any drumming technique can yield acceptable sound. Since the goal is not to master a particular technique, players are free to experiment and use whatever playing gesture they feel most comfortable with, while experienced players are still free to use more

challenging and virtuosic techniques.

The drum circle presents an interesting model for group musical collaboration. Because of its intrinsic repetition and slow tempo changes, I became interested in seeing if the drum circle model could be adapted to work with remote players by using phrase-length delays.

4.2 General approach/implementation

4.2.1 Using delay to synchronize

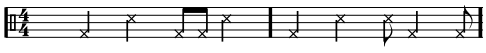

Figure 4-1: A simple 8-beat phrase

The core idea of the Remote Drum Network is a simple one: the system can tolerate significant latency if the players are willing to hear each other's phrases one phrase late. For this technique to work, there has to be a narrow definition of what a phrase is. In this case, a phrase is a short musical idea that can be repeated and that contains a fixed number of beats chosen by the players. A typical phrase could consist of two 4-beat measures (figure 4-1). An analogy could be drawn to the musical form of a round, in which each phrase is intended to be heard at the same time as the previous one.

Each player has a drum, a microphone (and mixer in some cases), speakers, and a computer connected to the Internet. Everyone plays to a fixed tempo, and each computer has its own metronome. Although the tempo of each computer's metronome is the same, the relative phase of each metronome is not—one could be ahead of or lag behind another.

The basic approach is to send audio between the different computers encoded with a time stamp that indicates what beat in the measure it was recorded on. Knowing when it was recorded, the receiving machine can play the sound back at the correct time, regardless of how long it took the packet to cross the network; for example, if player A hits a drum on his beat 1, player B will hear a hit sometime later, on her beat 1, and vice versa.

There are some potentially strange musical side effects of this arrangement. Since

each player hears a different combination of the other player's delayed phrases, their perception of the music can be quite different. If player A introduces a new musical idea; for example, by starting to play triplets instead of quarter notes, player B will not hear Player A's triplets until some time later (based on the offset between players). If player B responds immediately, Player A will not hear that response until two full phrases have gone by. Similarly, one player could perceive that they are taking turns playing alternate phrases, while the other player thinks that they are alternating between playing together for eight beats, and resting for eight beats.

4.2.2 Getting audio into and out of otudp

I chose to implement the drum system in Max/MSP using Matt Wright's otudp object and other custom MSP objects [46].

The first challenge was to figure out how to send audio between the computers. Matt Wright's otudp object takes a pointer to a buffer location and buffer length, and sends the contents of the buffer as a UDP packet. UDP is a connectionless protocol that doesn't acknowledge whether it received a packet or not, making it a good choice for applications which stream a lot of data, but which can tolerate dropped packets. Bandwidth is optimized since only data packets are sent [37].

Unfortunately, the buffers otudp uses are different from the buffers that Max uses, so to get audio into otudp, one would have to write a new Max object. Adam Smith had written a Max object called packetbuild~ for his WAI-Knot [36] project that put audio in a buffer to be read by otudp. He also wrote another object called packetparse~ that took otudp buffers and played them back as audio. I modified Adam's objects for my drum application by adding a header to the packet and by having the object keep track of the packet number, which it would reset on receiving a "bang" as a way of encoding the location of beat 1. (Max uses "bangs" to trigger events; for example,

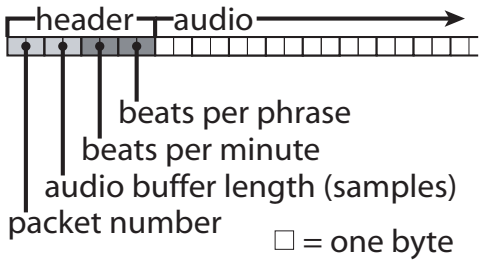

Figure 4-2: Audio packet with header

the metronome object in Max outputs a bang on every beat.) I also modified his receive object packetparse~ to output a bang when it received a beat 1 and to output tempo and beats per phrase in addition to audio to keep the computers synchronized.

The Remote Drum Network packet consists of a header, followed by the audio (figure 4-2). The header contains the packet number (which gets reset on beat 1), audio buffer length in samples, tempo (BPM), and number of beats per phrase. Each datum in the header is encoded in two bytes for consistency, even though tempo and beats per phrase wouldn't typically exceed 256, or one byte. The remainder of the packet contains audio. Although MSP represents audio internally using 24 bits, packetbuild~ scales each sample of audio down to two bytes, which still yields CD quality sound. To limit the complexity of the system, I decided to leave the audio uncompressed, though compression algorithms could be implemented in the future. The packet manipulating objects could be easily recoded to accept longer headers if more control data was needed to be sent between players.

With a timestamped audio infrastructure in place, I started to run some tests between computers.

4.2.3 A first attempt

In my first implementation, The role of packetparse~ was to receive incoming Drum Network packets, route the encoded audio to an MSP audio output, and to indicate when it received a packet with a packet number of zero, which would mean that the audio in the packet had been recorded on the sending computer's beat 1. Each time packetparse~ received a packet with a packet number of zero, it output a bang. The bang caused Max to start recording the audio coming out of packetparse~ into the beginning of an MSP buffer (of sufficient size to hold an entire phrase worth of audio). When the local metronome indicated beat 1, the buffer was played back. The overall effect was supposed to be that a drum hit on beat 1 of the sending computer would be played back on beat 1 of the receiving machine. Unfortunately, the timing of the metronome was unreliable enough that there was significant artifact at the beginning of each phrase, and often the first beat was not actually aligned with the start of the buffer.

This made it difficult to play with someone else, since the delay was constantly shifting. Playing in sync was no guarantee that the other player would hear it properly synchronized. Also, the system was unable to put out-of-order packets back in the correct order if they got shuffled in transit. There needed to be a way to get the received audio into a format that Max could access, one that could use the packet number information to keep the beats lined up, rather than relying on the timing of the metronome.

4.2.4 A solution: packetindex~ as a way to get at buffer~

A more successful approach was to take advantage of the knowledge of which packet contained beat 1, and to write it directly to the beginning of a buffer~ object, which can be played back by Max using the index~ object.

The packetindex~ object (at the bottom of figure 4-3) writes incoming packets into a buffer~, indexed by their packet number. This means that packet 4 goes in its slot (4) even if it gets there before packet 3, avoiding discontinuity artifacts when packets are received out of order. The other advantage of this approach is that if several packets are dropped, audio from the same part of the previous phrase fills in the gaps. This can still cause artifacts, but they are minimized, particularly with percussion signals that are mostly silence punctuated by individual hits. Since the drum hits themselves are more "noisy" than periodic, discontinuities are less apparent. Because hits in one phrase resemble the hits in the next, hearing a leftover drum hit on beat 2 doesn't necessarily ruin the next phrase, even if a significant number of packets were lost.

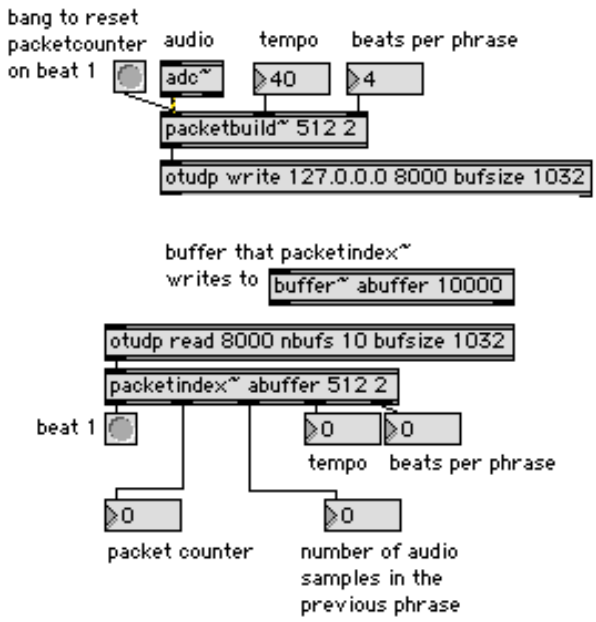

Figure 4-3: Inputs and outputs of Max objects packetbuild~ and packetindex~

Some artifacts remain: the number of packets in a phrase varies due to jitter in

the timing of the metronome on the sending machine, and it is possible that some leftover data can be played just before beat 1, causing a click.

Tristan Jehan generalized packetbuild~ and packetindex~ to be able to accept different header lengths and to let the user specify the length of the header with an optional argument, enabling the user to send more data. He also simplified the code quite a bit. The updated objects are available at `http://web.media.mit.edu/~tristan/maxmsp.html`.

4.3 A two-player implementation

4.3.1 How synchronization works

In the two-player implementation, each computer runs an independent metronome. Ideally, the two computers are offset by half of a phrase length, allowing the maximum buffer size on either machine (assuming symmetrical network latency, where $L = L'$, as shown in figure 4-4). Audio is sent from player A to player B in packets that are numbered starting with beat 1. Player B's computer puts the packets in order in a buffer, and delays playing it back until the corresponding beat occurs on player B's machine. If player B were to point her microphone at the speakers, player A would hear his own echo one phrase after he played it, representing a round trip delay of one phrase. The IP addresses and UDP ports of the two machines were hard-coded into the Max patches. There was no system for allocating ports or identifying clients.

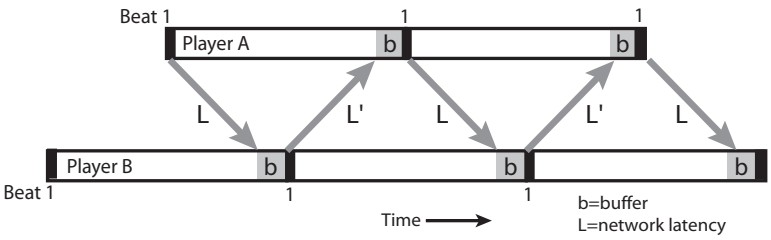

Figure 4-4: Two player model

4.3.2 Maximum network latency

The maximum network latency that the Remote Drum Network allows is a function of the tempo and phrase length. For a simple two player system, assuming that the two players are offset by exactly one half of a phrase length,

$$\text{maximum network latency(s)} = \frac{1}{2}\left(\frac{\text{beats per phrase}}{\text{beats per minute}} \times 60 \text{ s/min}\right) - \text{B} \qquad (4.2)$$

where B is the combined buffering and A/D latencies (in seconds), typically around 15 ms using the Macintosh sound manager. If the tempo is 140 beats per minute and there are 8 beats per phrase, the maximum allowable latency would be 1.7 seconds (one way).

4.3.3 Dealing with tempo, bpm changes

In my first implementation, when the player on the server changed the tempo and beats-per-measure settings (by moving sliders on the screen), the clients would be updated as soon as they received the packets. This caused significant artifacts since the audio already in the buffers was at the old tempo, so the ends of phrases would be cut off, or random data would be played from the end of the buffer beyond the phrase ending.

To address this problem, I had the receiving machine measure the offset between its own beat 1 and when it receives a packet stamped as beat 1. The tempo and beats per measure settings were then delayed by that amount, so if one player shifted the tempo on beat three, the other player would hear the tempo change on his beat three, without any extra audio artifacts.

One unexpected advantage of this approach is that MIDI note data could be sent in the same manner, and be synchronized with the audio (quantized to the nearest packet length, approximately 12 ms). Note-on and note-off messages were encoded in the packet header and delayed to match the beat they were originally played on. The quantization could be reduced significantly by time-stamping the MIDI message with its approximate position within the phrase. Such a system could enable players with lower bandwidth connections to play using MIDI, while other players used audio. Another advantage of using MIDI is that the pitch and attack of each note are known, enabling simpler beat tracking algorithms and systems of intelligent harmony, possibly extending the system to support more tonal instruments. Integrating audio and MIDI into the Remote Drum Network should be a subject of future work.

4.4 Multiple players

To scale the system up to allow multiple players, I needed a server to synchronize and mix the audio from each of the clients and stream it back to them. I also needed a way for the server to accept new client computers without already knowing their IP addresses.

4.4.1 Topology

I decided to use a star topology with a central server and several clients to minimize the overall network congestion (figure 4-5). For n players (counting the player on the server) network load is $2(n-1)$ audio streams at the server, and 2 streams at each of the clients. For a fully connected network, each player would have a network load of $2(n-1)$. Tempo and beats per phrase are set by the player on the server machine. The server provides custom audio mixes that get sent back to each client. Each mix contains the synchronized sound of everyone else minus the receiving player's sound.

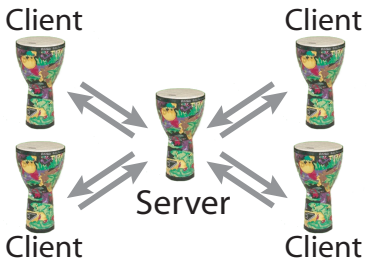

Figure 4-5: Remote Drum Network star topology

4.4.2 Multiple player model

In the multiple player model (figure 4-6), audio from the clients is sent to a server which sends custom mixes of everyone else's audio (but not their own) back to the clients. To support the same maximum network latency as in the two-player model, the clients have to be synchronized with each other, and the server has to be offset by half of a phrase length. In practice,

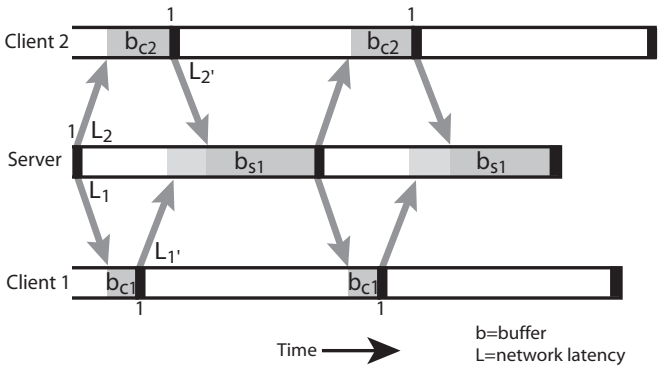

Figure 4-6: Server synchronizing two clients

the system works as long as the server metronome is shifted relative to the clients by at least the actual network latency. The server buffer is typically larger than the client's buffers to ensure that the audio from all the clients has been received before mixing it and sending it back to the clients. There is one significant drawback to this approach: although the connection between any client and the server is equivalent to the two-player mode with a round trip delay of one phrase, connections between clients have a round trip delay of two phrases. This gives the player on the server more control than the clients have over guiding the musical direction of the group.

4.4.3 Client and server implementation

To allow players to hear each other, audio from each of the clients is sent to a central server, which can host one additional player. The server has a fixed IP address, and each client is pre-programmed to connect to it. The server currently supports four clients and one player on the server.

Port negotiation

In order to allow any client to connect to the server, I implemented a system of port negotiation. To initiate a connection, the client sends a packet to a dedicated UDP port on the server. This packet contains the client's IP address and specifies the port on which the client would like to receive streaming audio. The server replies with the port that it would like the client to stream to. The client and server stream audio data to each other, and the client's metronome starts when it receives a packet that was sent on the server's beat 1 (packet number zero).

The server will reject new connection requests from machines that are already connected. When the server is full, it returns zero as the port number, and the client can try again later. If the server doesn't receive a packet encoded with beat 1 within three phrase lengths, the port will time out and remove the client from the active list, effectively removing dead connections and freeing up ports so that other computers can connect.

Interface

I chose to keep the Remote Drum Network's interface quite minimal to maintain the player's focus on the music. Each player has an on-screen mixer which can control the level of his own sound, the other player's sound, and the volume of the metronome. There is also a visual metronome to indicate the current beat.

4.5 Testing at MIT and between MIT and MLE

The system was tested with up to four players, both within the Media Lab and between the Media Lab in Cambridge, Massachusetts and Media Lab Europe (MLE) in Dublin, Ireland. Michael Lew, a researcher at MLE, set up a system there and also participated in the tests. Our tests with MLE had two players (one playing on the server) located at MIT and one at MLE. Apart from the logistics of setting up a time to play, the MIT-MLE tests were nearly identical to tests within the Media Lab. During the course of the tests, some packets were dropped, creating occasional pops in the audio, but the noise was not distracting.

We used the telephone quite a bit to discuss what we wanted to do between playing sessions, and it seems like having a way to temporarily disable the delay could let conversation happen over the same link (by talking into the microphones). Some text-based chat facility would have also been useful.

4.5.1 General observations

In general, participants enjoyed playing the system, and they felt that they were playing music with each other. The delay took some time to get used to, but with decent players who listened to each other, it worked. As in many kinds of improvisation, we found that it worked best when we took turns taking responsibility for keeping a steady background beat while the other players could play more ornamental solos. Social aspects, such as turn taking, had unexpected effects due to the delay. What one player might perceive as taking turns would seem like unison to another player.

Since there are no visual cues as to who is playing which hits, it helped when each drum was timbrally distinct—a djembe and bongos worked better together than two djembes.

Although having experienced drummers helped keep everyone playing together, actual drumming technique was not as important as having a good sense of tempo and being able to listen to the other players. In one trial with a novice player, it was very difficult to get ourselves synchronized, though I suspect we would have had some trouble playing without the delay as well.

The original design of the Remote Drum Network assumed that the players would be listening to a metronome click to stay synchronized. When the metronome was displayed as an audible click, it turned out to be harder to listen to the other players and had an isolating effect; each player was playing to the click, not with each other. Displaying the metronome visually allowed the timing to be more flexible, while achieving the main goal of keeping the tempo approximately correct.

Even without a metronome, there was a tendency to converge on tempi for which the delay caused a shift of an integral number of beats. If the tempo was close to correct, it tended to converge on the correct tempo as long as all of the players attempted to match what they heard, but made their changes slowly. Some players preferred to ignore the metronome completely, with good results.

Slow, deliberate changes that lasted more than one phrase were also necessary if a player wanted to suggest a new rhythmic idea to the group and get the others to follow him, or to make more large-scale changes to the music. Players were able to play some phrases in unison, but only by repeating the same phrase many times. The player on the server had a clear advantage due to the shorter delay, but the other players could convince the server player to act as a musical relay, reinforcing their suggestions to the rest of the group.

Increasing the delay to sixteen bars allowed for more complicated phrases, but it became harder to synchronize tempo since it took twice as long to hear someone's reply. Eight bars phrases were the best compromise between responsiveness and phrase complexity.

Call and response form worked in some special cases. Four beat calls and responses in 8 beat measures (one call and one response fit in one phrase length) worked fine as long as they were repeated enough to convince the other players to play in the gap. The same scenario worked poorly when the metronome was shifted to four beat phrases. If the leader initiated the call and the others played back during the gap, the leader heard the response while playing his next call, and the other players heard each other's responses at the same time as the next call.

Changing the number of beats played in a phrase without changing the system phrase length worked for short periods, but the background player eventually had to return to the correct phrase length. In these periods where players' phrases were a different length than the system phrase length, each player heard the other players on different beats than they intended, creating a kind of unintentional phase shift. For example, if the system was set to eight beats per phrase, and I played a six beat background phrase while everyone else attempted to play along, I would hear their phrases shifted by eight beats, which sounds like a whole six-beat phrase plus two beats of offset, so an accent played on beat 1 would sound to me as an accent on my third beat. The other players also would hear each other shifted by two beats. If I switched back to an eight-beat bar, within a phrase, the others would hear hits on the correct beats again.

In general, trials with three and four players worked similarly to those with two, but it was even more important to have unique timbres. Players had to be more sensitive to each other's playing and needed to play more deliberately to get the rest of the group to follow changes in the music.

Overall, the Remote Drum Network encouraged group collaboration by giving players a sense that they were playing music together. Rather than being an exact replica of the way people play drums together in the same physical space, the Remote Drum Network required different techniques of listening to the other players and anticipating their actions, which often led to musical surprises not encountered in a traditional drum circle.

Chapter 5

Conclusion

Electronic percussion instruments have great potential to extend the roles of traditional percussion into new areas that were not previously possible. This thesis pushes those roles along a few different axes:

- Development of local and remote percussion networks
- Using the instruments themselves to support the creation, recording, playback, sharing, and development of musical motifs to create a larger piece of music
- Giving players more levels of continuous control over timbre.
- Extending the joy of solo and ensemble playing to novices.
- Coupling electronic sound to surfaces that can be manipulated acoustically.

With these new potentials come some risks. For the Beatbugs, the ability to modify musical patterns after they have been played has no physical analog in traditional instruments, so it can be difficult to make these interactions seem intuitive. Similarly, for the Remote Drum Network, players have little experience with playing music using long delays so any intuition has to be built through playing the system itself. With any electronic instrument, much of the playability of the instrument comes down to the strength of the mappings between playing gestures and sound.

The Beatbug project gave me a unique opportunity to see how an instrument works outside the lab, and to see how my own intuitions can differ from those of other people. We held Beatbug workshops and performances in three cities as part of

Toy Symphony. Thousands of people have seen them, and hundreds of people have played them in open houses. One thing we learned is that although an interface can seem "intuitive" to the designer, people bring their own experiences and expectations to an instrument. One can try to help a player learn the interface by rewarding certain gestures. Hitting the Beatbug seems obvious once someone sees that it flashes and makes a sound when it is hit. The antennas were more difficult for people to understand because they only modify existing sound, they don't create any new sound themselves. Some people stopped trying to play after bending the antennas and not hearing anything. In that sense, the Beatbugs were not initially intuitive to players. However, having people understand how to play Beatbugs in an open house is only one of the goals. We found that after spending a few days working with Beatbugs in workshops, the children understood even the most complex aspects of the interaction, such as choosing to develop a motif further, or creating a new one.

Intuition is built through experience. Each time I play the Remote Drum Network, the other players and I get better at anticipating each other's music, telegraphing our intentions measures in advance, and shifting back to a simple rhythm when we start to lose synchronization.

This thesis is an exploration of different ways to apply electronics and computation to percussion. It has not yet resulted in a synthesis: one instrument that uses all of the ideas developed in these instruments. More development has to happen before such a synthesis is possible. The Beatbugs and the Remote Drum Network designs instead focused on a subset of those ideas to enable and reinforce specific desired interactions. For the Beatbugs, the design of the interface helped make it easier for the players and the audience to understand the relationship between what the players did and what they heard. For the Remote Drum Network, the phrase delay system was intelligible to the players, and the limitations of the system helped spark new musical ideas that would not have occurred in a traditional drum circle.

Chapter 6

Future work

Much remains to be done on Beatbugs, Remote Drum Network, and in the field of electronic percussion in general. The Beatbugs could benefit from more robust physical construction, especially if they were to be redesigned as an installation rather than as a performance instrument. Although Remote Drum Network can tolerate high network latencies, it currently requires significant bandwidth. Audio compression schemes could reduce the bandwidth, but sending MIDI-type trigger data could shrink the requirements even more, while opening up more possibilities for beat tracking and intelligent harmony. Looking toward the future, I see acoustic-electric hybrid instruments as a possible solution to the challenges of mapping multiple gestural and musical parameters in a coherent way.

6.1 Beatbugs: toward a robust installation and future applications

6.1.1 Materials and fabrication issues

Although the current Beatbug system is robust enough for workshops and performances, it is vulnerable in unsupervised situations, making it unsuitable for installations in museums and public spaces. The antennas need to be redesigned, or replaced with a less fragile system. Shorter, wider antennas would help limit how far a player

could bend them. The Beatbugs would also need to be mounted in a way that contains their cables and holds them at playing height to avoid tangles and to protect the cables from being tugged on.

6.1.2 Future application development

Installations

The current Beatbug application has a linear path suited for performing a piece of music. To support an installation in a museum context or in a public space, the application would need to be adapted to work in a much less structured environment. A Beatbug installation would need to be enjoyable for a range of numbers of players and should allow them to start playing and to leave at any time. The system of entering and sharing motifs might have to be simplified to support players with no musical background as well as players with more experience. In a workshop, a leader can introduce players to what the system does. In installations, there is not usually a leader, so the interactions need to be made more obvious to the players.

Toys

Our experience of trying to commercialize the Beatbugs showed us that to make a successful musical toy, some significant hurdles have to be overcome. Since toys are a particularly cost-sensitive market, the electronics have to be extremely inexpensive, limiting most electronic musical toys to crude 4-bit ADPCM samples at low bit rates. Implementing any continuous control over timbre would be difficult using the simple wavetable synthesizers found in most electronic music toys.

Most of the value of a toy has to be in its "stand alone" mode. If a toy is only interesting when you have seven other friends who already own the toy, it is unlikely to catch on as a commercial product. Although children are using computers more than ever, parents are still reluctant to buy toys that need to be tethered to a computer, making the Beatbug model of having a central hub that does all of the processing an unattractive topology.

Still, given the cost constraints, there are some techniques that could be used to make better-sounding musical toy. Specific timbres could be chosen that sound better at low sample and bit rates. Some simple filtering could be implemented acoustically by letting players shape the speaker cavity with their hands, and a matrix of pre-rendered wavetable samples could be created to approximate combinations of two timbre parameters. FM synthesis would be a good option for toys since the hardware requirements are minimal and the timbral range and possibility for continuous timbral manipulation are quite broad, but there are not currently any FM synthesizer chips that are inexpensive enough to be used in toys.

6.2 Reducing the bandwidth requirements of the Remote Drum Network

Although the Remote Drum Network lets people play music in high-bandwidth, high-latency conditions, there are many more lower-bandwidth, high-latency connections in the world.

Audio compression

The delayed phrase model allows enough time for the audio stream to be compressed and decompressed. Even a simple algorithm like ADPCM would reduce the bandwidth requirement to one quarter that of uncompressed audio, but one could use better, more processing intensive codecs like MP3 to cut the bandwidth requirements further.

Sending MIDI style trigger information

A more extreme reduction in bandwidth would be to only send trigger information with each drum hit. CNMAT's OpenSoundControl would be a good choice for this job, since it can take in MIDI data and prepare packets for OTUDP. Without good controllers and careful mapping, some of the intimacy that comes from sending actual

audio could be lost. One interesting commercial controller is the Roland Handsonic HPD-15, a hand drum controller with multiple pressure sensitive pads. It might be a good testbed for networked drum circle ideas.

It might also be possible to change the pitch of incoming notes to follow what is being played on each client machine, or to match an underlying structure, enabling the use of more tonal sounds and chord changes.

One could create heterogeneous Drum Networks in which some players send and receive trigger information over low bandwidth connections, while others send and receive audio. One possible client for a low bandwidth drum network could be a cellular phone. As phones incorporate more sophisticated processors, software synthesis could become possible, letting players send trigger and timbre control data to make expressive electronic percussion sounds using their phones. One interesting aspect of representing the data as triggers is that the system could change the timing of individual notes, perhaps applying a little quantization to make it harder for novices to divert the whole group. Beat tracking would also be easier, since it would require no audio analysis.

The bandwidth savings of sending trigger data come at a cost though. There is a risk, especially in a delayed phrase situation such as the Remote Drum Network, of losing the belief that the other people are actually playing music. Trigger data can lack the obviously human feel of actual audio, making the players feel more isolated rather than less.

6.3 Acoustic/electric hybrids

One of the things that makes the Remote Drum Network enjoyable to play is that it uses actual audio from the players. This both greatly broadens the possible range of sounds that the player can use, and gives the sound a more personal connection.

With the Echo Drum, Pressure Tambourine, Drum Network, and Remote Drum Network, I was working on ways to combine computation and network ideas with acoustic sound. This aspect was missing from the Beatbugs and although they each

had their own speakers and made sounds when they were hit, the sound was clearly electronic and the player had no actual acoustic control over the sound.

Through the development of MIDI, electronic musical controllers became abstracted from sound production. This was a powerful development since it enabled any parameter of the sound to be mapped to any function of the controller. It proved invaluable for the Beatbugs because it allowed the bulk of the processing to occur on the computer, while letting the Beatbugs themselves work as input and output devices. The problem with decoupling the controller and the sound source is that someone needs to create a mapping between playing gesture and sound, and creating rich and expressive mappings that still make sense is hard. It can be especially difficult to combine multiple sound and gestural parameters into groups that make sense physically and musically without having the mapping seem arbitrary.

In contrast, the sound of an acoustic instrument is dictated by its physical form. A player can gain rich and simultaneous control over many parameters of the sound through interaction with that physical form. On an acoustic guitar, one note can be played in many different ways; for example, the string can be played using a pick, fingernail or pad of a thumb. The string can be plucked closer to the bridge, or closer to the neck. The strings can be muted partially or fully by the player's palm or fingers. For each method of playing, there are infinite sub-variations, each involving control over many sound parameters at once.

To duplicate the range of sounds and manipulations possible in acoustic instruments, one approach has been to make an electronic controller with a large number of sensors and controls and to map each to a parameter of the sound generator. The difficulty with this approach is that someone has to choose the physical parameters and sound parameters that make sense together, which can be arbitrary, like the mapping of a midi knob box, or more metaphoric, like a wind controller's attempt to emulate a saxophone or clarinet). Most keyboards use the piano metaphor: keys specify which note to play, key velocity maps to note velocity, and pitch bend and modulation are mapped to two wheels on the left side of the keyboard. Any commercial sound module can be controlled by any keyboard, and the module is designed

to work with these mappings. Keyboards work well for piano and organ sounds, but founder when faced with sounds that need a greater degree of continuous control.

My hunch, developed through the experiences and projects described in this thesis, is that by bringing some parts of the process of sound creation back into the physical world, one could interact with the sound in the same ways possible with an acoustic instrument, giving a more coherent interdependence between different parameters. The playing gestures would not just be metaphors for acoustic playing, but real acoustic interactions with the sound. The drum work just begins to touch on this idea, but doesn't go nearly far enough. One of my goals for future work is to create these hybrid instruments to see if my hunch is correct.

Appendix A

Beatbug circuit schematics

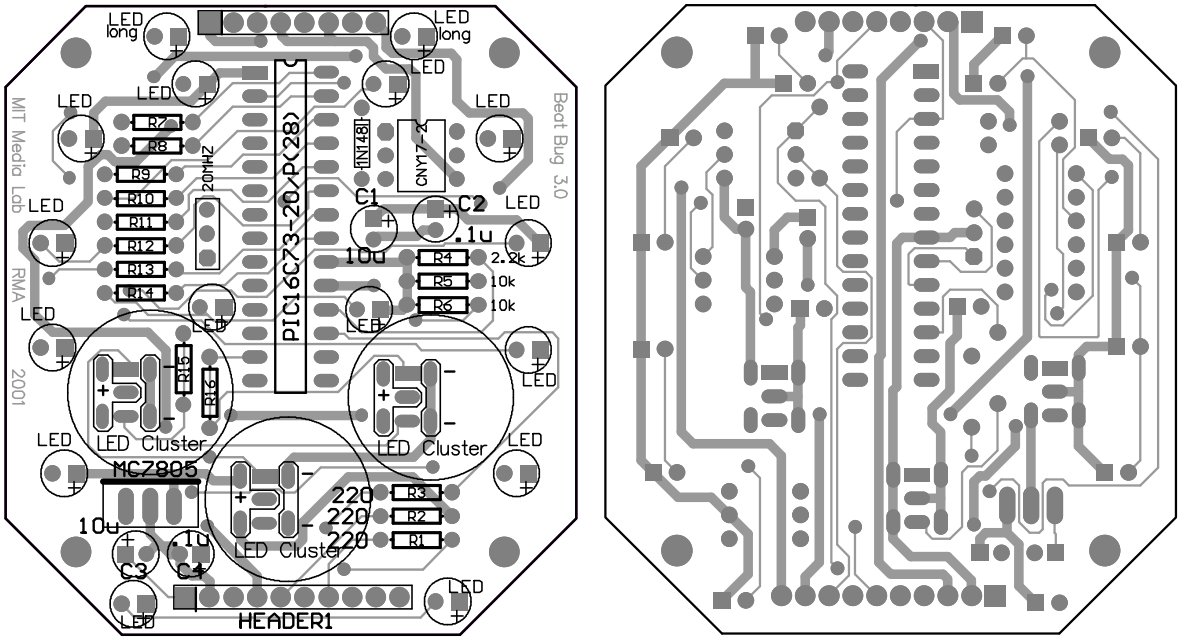

Figure A-1: Beatbug circuit layout

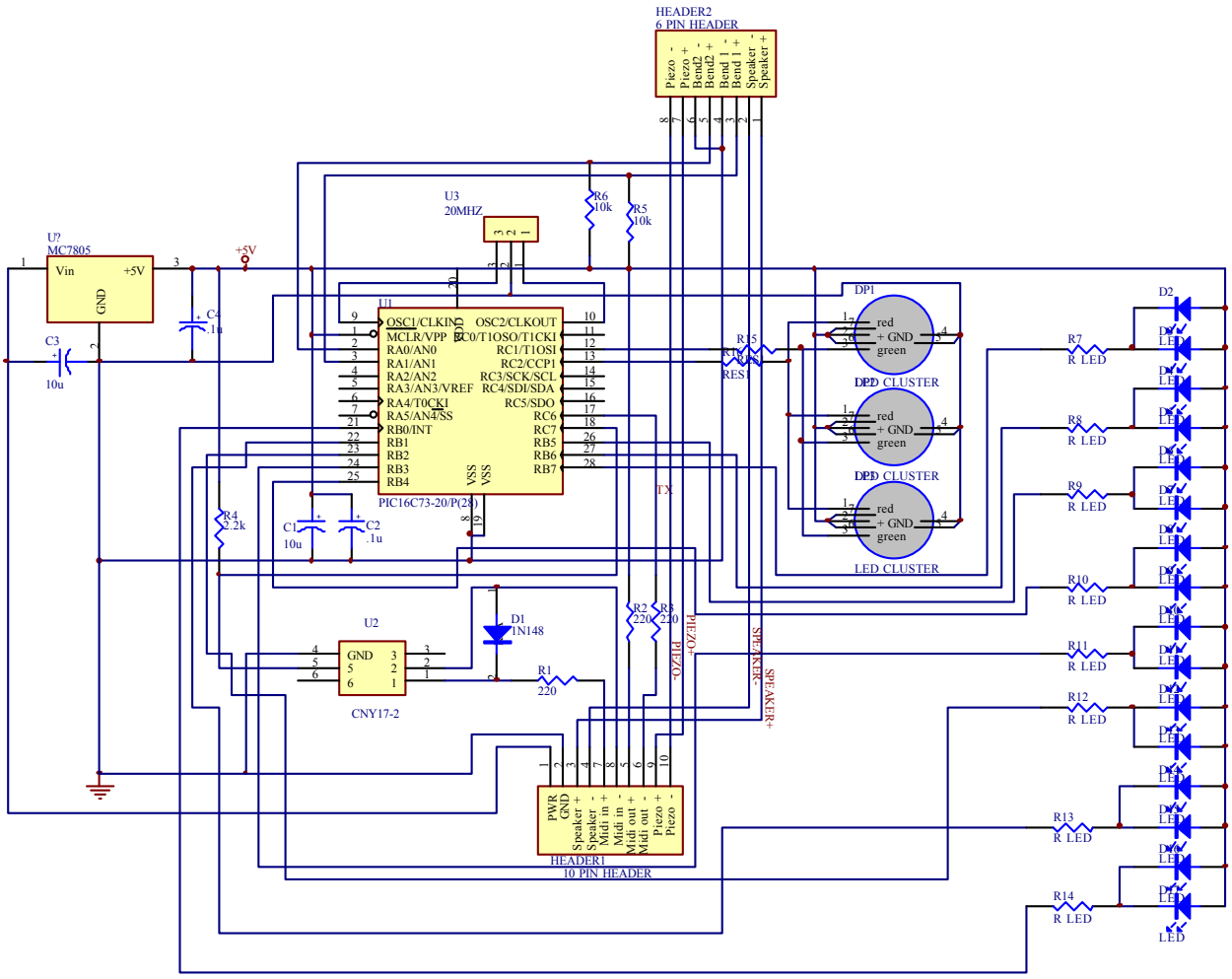

Figure A-2: The Beatbug circuit schematic

Bibliography

[1] The Ark. The ark, a cultural center for children. `http://www.ark.ie`.

[2] James Blades. *Percussion instruments and their history*. Books that matter. Fredrick A Praeger, New York, N.Y., 1970.

[3] Tina Blaine and Tim Perkis. The jam-o-drum interactive music system: A study in interaction design. In D. Boyarski and W. A. Kellogg, editors, *Proceedings of the ACM Symposium on Designing Interactive Systems.*, pages 165–173, New York, NY, 2000. ACM Press.

[4] Robert Boie. Radio drum. In S. Arnold and G. Hair, editors, *Proc. 1990 International Computer Music Conference*, pages 398–400, San Francisco, CA, 1990. International Computer Music Association.

[5] Robert B. Breithaupt. The drum set: a history. In John S. Beck, editor, *Encyclopedia of percussion*, number 947 in Garland reference library of the humanities, pages 173–185. Garland publishing, New York and London, first edition, 1995.

[6] Steven Brown, Björn Merker, and Nils Wallin. *The Origins of Music*, chapter 1: An introduction to evolutionary musicology, pages 3–24. MIT Press, Cambridge, Massachusetts, second edition, 2000.

[7] Phil Burk. Jammin' on the web: A new client/server architecture for multi-user musical performance. In Ioannis Zannos, editor, *Proc. 2000 International Computer Music Conference*, Berlin, Germany, 2000. International Computer Music Association.

[8] Cornerstone prototype development, Pawtucket, RI. http://www.cornerstonepd.com.

[9] Cycling '74, San Francisco, CA. http://www.cycling74.com.

[10] Hugh Davies. Electronic percussion. In S. Sadie, editor, *The new grove dictionary of musical instruments*. Macmillan, London, second edition, 1984.

[11] Flexpoint, Inc. Midvale, UT. http://www.flexpoint.com.

[12] Jacob Fraden. *Handbook of modern sensors : physics, designs, and applications*, volume 2. Springer-Verlag, New York, second edition, 1996.

[13] Gillian Gaar. Liner notes in *Talk normal: the Laurie Anderson anthology*.

[14] Albert Glinsky. *Theremin: ether music and espionage*. Music in American life. University of Illinois Press, Urbana and Chicago, 2000.

[15] Mickey Hart. Quoted in http://www.remo.com/drumcircles.

[16] PaiA inc. Touch switches. http://www.paia.com/touchsw.htm.

[17] Pintech inc. http://www.edrums.com.

[18] Remo inc. http://www.remo.com.

[19] Elac industries. Distributed mode loudspeakers. http://www.elac.com/english/innovation1.html.

[20] John Lazzaro and John Wawrzynek. A case for network musical performance. In *The 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2001)*, Port Jefferson, New York, June 2001. ACM, Academic Press.

[21] Tod Machover. Hyperinstruments: a progress report. Internal document, MIT Media lab, 1992.

[22] Jeremy Montagu. Tambourine. In S. Sadie, editor, *The new grove dictionary of musical instruments*. Macmillan, London, second edition, 1984.

[23] Joseph Paradiso. American innovations in electronic musical instruments. *New Music Box*, 6, October 1999. Available at `http://www.newmusicbox.org/third-person/index_oct99.html`.

[24] Joseph Paradiso. The brain opera technology: New instruments and gestural sensors for musical interaction and performance. *Journal of New Music Research*, 28(2):130–149, 1999.

[25] Gordon B. Peters. *The drummer: man*. Kemper-Peters Publications, Wilmette, Illinois, revised edition, 1975.

[26] Propellerhead software, Stockholm, Sweden. `http://www.propellerheads.se`.

[27] Robert Rich. Buchla thunder. *Electronic musician*, page 84, August 1990.

[28] J. Bradford Robinson. Tambourine. In S. Sadie, editor, *The new grove dictionary of musical instruments*. Macmillan, London, second edition, 1984.

[29] Rocket Network, Inc., San Francisco, CA. `http://www.rocketnetworks.com`.

[30] Thomas D. Rossing. *Science of Percussion Instruments*, volume 3 of *Series in Popular Science*. World Scientific, Singapore, 2000.

[31] Greg Rule. Keyboard reports: Korg wavedrum. *Keyboard*, 21(3):72–78, March 1995.

[32] Greg Rule. Roland v-drums - a new standard in positional head sensing, plus a megaton of punchy percussion samples. *Keyboard*, 24(4):77–84, April 1998.

[33] Curt Sachs. *The history of musical instruments*. W. W. Norton & company, New York, 1940.

[34] Larry Schindel. Placing the art before the sport: a united nations of hardware. *Live Sound*, April 1998.

[35] 1999 Sens*bles Symposium, Cambridge, MA `http://www.media.mit.edu/Sensibles`.

[36] Adam Douglas Smith. Wai-knot (wireless audio interactive knot). Master's thesis, MIT Media Laboratory, Cambridge, MA, September 2001.

[37] Willliam Stallings. *High speed networks: TCP/IP and ATM design principles*. Prentice-Hall, Upper Saddle River, New Jersey, 1998.

[38] Toy Symphony. `http://www.toysymphony.org`. Website for the Toy Symphony project.

[39] Larry the O. Nearfield multimedia marimba lumina: This is not your mother's midi controller. *Electronic Musician*, 16(6):138+, June 2000.

[40] Kenneth Walton. Classical music review: Toy symphony. *The Scotsman*, June 4, 2002. Available at `http://news.scotsman.com/archive.cfm?id=601862002`.

[41] Gili Weinberg. Expressive digital musical instruments for children. Master's thesis, MIT Media Laboratory, Cambridge, MA, June-August 1999.

[42] Gili Weinberg. *Interconnected Musical Networks Bringing Expression and Thoughtfulness to Collaborative Music Making*. PhD proposal, MIT Media Laboratory, 2002.

[43] Gili Weinberg, Roberto Aimi, and Kevin Jennings. The beatbug network: A rhythmic system for interdependent group collaboration. In Eoin Brazil, editor, *Proceedings of the NIME-02 Conference on New Interfaces for Musical Expression*, Dublin, Ireland, May 2002. NIME, Department of Computer Science and Information Systems, University of Limerick, Ireland.

[44] Gili Weinberg and Seum-Lin Gan. The squeezables: Toward an expressive and interdependent multi-player musical instrument. *Computer Music Journal*, 25(2):37–45, July 2001.

[45] Norman Weinberg. Electronic percussion. In John S. Beck, editor, *Encyclopedia of percussion*, number 947 in Garland reference library of the humanities, pages 191–193. Garland publishing, New York and London, first edition, 1995.

[46] Matt Wright. Open soundcontrol: A new protocol for communicating with sound synthesizers. In *Proc. 1997 International Computer Music Conference*, Tessaloniki, Greece, 1997. International Computer Music Association. OSC is available at `http://cnmat.cnmat.berkeley.edu/OSC/`.

[47] Sarah H. Wright. Pair of media lab events showcase toys and inventions. *MIT Tech Talk*, October 27, 1999. Available at `http://web.mit.edu/newsoffice/tt/1999/oct27/media.html`.

[48] Diana Young. The hyperbow controller: Real-time dynamics measurement of violin performance. In Eoin Brazil, editor, *Proceedings of the NIME-02 Conference on New Interfaces for Musical Expression*, Dublin, Ireland, May 2002. NIME, Department of Computer Science and Information Systems, University of Limerick, Ireland.