BIT BENDING: AN INTRODUCTION

Kurt James Werner,
Center for Computer Research in Music and Acoustics,
Stanford University, Stanford
California, USA
kwerner@ccrma.stanford.edu

Mayank Sanganeria,
Center for Computer Research in Music and Acoustics,
Stanford University, Stanford
California, USA
mayank@ccrma.stanford.edu

ABSTRACT

We introduce the technique of "Bit Bending," a particularly fertile technique for circuit bending which involves short circuits and manipulations upon digital serial information. We present a justification for computer modeling of circuit-bent instruments, with deference to the movement's aversion to "theory-true" design and associations with chance discovery [1]. To facilitate software modeling of Bit Bending, we also present a software library for modeling certain classes of digital integrated circuits. A synthesis architecture case study (frequency modulation via numerically controlled oscillators) demonstrates software modeling of Bit Bending in action.

1. INTRODUCTION

Before discussing Bit Bending, we will briefly review the context of circuit bending and the motivation for modeling musical instruments and audio effects in software, as well as a justification for computer modeling of circuit-bent instruments in particular.

1.1. History of circuit bending

Circuit bending is the process of creatively modifying or augmenting sound-producing electronic devices [2], typically toy instruments, guitar effect pedals, sound-producing children's toys, etc.

Circuit bending is primarily associated with experimental, noise, do it yourself (DIY), hacker, chiptunes, and multimedia art scenes. Recently, the practice has found a foothold in the mainstream and occupies an increasingly important musical and cultural niche [3]. High profile artists like Radiohead, the Flaming Lips, and Björk increasingly experiment with and utilize circuit-bent instruments [3]. Information about circuit bending is shared freely online [4, 5], and DIY project communities (including circuit bending interest groups) thrive in various major metropolitan areas through workshops, festivals, and meetups [6].

Interest in circuit-bent instruments stems from the rich and complex sonic palette they create, the nostalgic associations conjured by modifying old devices, the repurposing of old technology, and a preoccupation with the emancipation of noise in musical practice [7]. In the context of "clock bends" or the use of binary counters to manipulate data streams (a special case of Bit Bending that will be discussed in this paper), circuit bending brings its own non-linear flair to a tradition of multi-scale sound design and composition [8, 9].

Figure 1: *Circuit-bent Speak & Spell with MIDI control, Kurt James Werner, 2007*

Though the practice began in the 1960s [10] (and traces roots to Leon Theremin's experiments with radio tubes in the 1920s [11]), it is still understudied. An example of a circuit-bent device is shown in Figure 1.

1.2. Overview of modeling techniques

Models are used to predict the behavior of a system, and model-building is a very useful way of understanding the world [12: 2]. It is often desirable to model the behavior of musical instruments and effects in software [12: 3].

There are numerous approaches to modeling, which are broadly categorized as non-physical signal models (spectral models, etc.) [12: 4], and physical signal models (which leverage "an explicit representation of the relevant physical state of the sound source") [12: 6]. One could also categorize modeling approaches into "top-down" (or "feature synthesis") models and "bottom-up" (or "system synthesis") models. A top-down approach would identify high-level features of a system that are important / perceptible / desirable and then try to recreate them in a model. A bottom-up approach would seek to understand and model each small foundational component, inter-connecting them appropriately, assuming that if each component is accurately modeled, the high-level features will occur as a by-product.

1.3. Modeling circuit-bent instruments

Computer models of circuit-bent instruments are desirable for several reasons. Circuit bending is often performed on devices that are antiquated and fragile. A computer model is used for archival purposes, to preserve historical sounds and practices. Creating a circuit-bent instrument requires specialized knowledge and a significant time investment. A software model of a circuit-bent instrument (in an audio plugin, for instance) fits conveniently into various electronic music workflows, is not susceptible to physical damage or decay, and scales easily. A model's insusceptibility to electrical damage enables fearless experimentation.

Figure 2: *Circuit-bent Casio SK-1 and SK-5, Kurt James Werner, 2008/2010*

A successful model of a circuit-bent instrument requires a careful discussion and selection of modeling goals. To facilitate such a discussion, consider two of the most popular and sought-after devices for circuit bending, the Casio SK-1 (a sampling keyboard introduced by Casio in 1985, as shown in Figure 2), and the Speak & Spell (an educational toy introduced by Texas Instruments in 1978, which used the world's first single-chip implementation of linear predictive coding speech synthesis, as shown in Figure 1).

Given the relatively simple behavior of the SK-1 (its built-in timbres are represented digitally as pulse-code modulation (PCM) streams), and the small vocabulary of the Speak & Spell (around 200 words [13]), a good choice for modeling these devices might be "structured sampling [12: 5]," which balances the strength of sampling- and model-based synthesis methods. Given the low cost of memory, it would be simple to sample the timbres of the SK-1 and the small library of words the Speak & Spell has available. Playing these into a simple model of the analog amplifier and filter circuits and a model of the speaker of each instrument would yield a fairly robust and useful model of the original devices.

However, interest in these devices generally lies in their fertility as the object of circuit bending. Circuit benders employ many techniques to repurpose sound-producing devices, including shorting circuit paths together or to ground, replacing resistors with potentiometers (often used in "clock bends"), touching exposed parts of the circuitry with their fingers ("laying of hands" [14]), and adding additional circuitry. These modified circuits can be very unpredictable, produce incredibly varied sounds (making feature synthesis very difficult), exhibit highly non-linear behavior (invalidating signal processing techniques that make the assumption of linearity), and feature fluid signal routings (making simplifications by techniques such as commuted synthesis [12: 305] unavailable).

Because these circuit-bent systems have the potential to confound top-down modeling techniques, we adopt and recommend a bottom-up approach. The charm of creating and working with circuit-bent instruments and audio effects lies in accidental discovery and experimentation. Identifying high-level features as desirable is limiting, and a system-based approach is necessary to preserve the open-ended experimentation of circuit bending.

2. BIT BENDING

A discussion of the history and characteristics of Bit Bending will give some context to our software library and model of a circuit-bent 2-operator FM synthesizer.

2.1. History of Bit Bending

The technique of Bit Bending arose out of circuit-bent instruments created by one of the authors (Kurt James Werner) between 2008 and 2011. Werner worked on four circuit-bent instruments in particular (Casio SK-1 and SK-5, and Roland TR-505 and TR-626, as shown in Figure 3) which all featured banana jack patchbays connected to digital circuitry inside (ROM chips), allowing access to any number of reconfigurable patchbay connections. Further work on the Yamaha PSS-170 and PSS-270 involved putting switches and a patchbay in series with critical circuit traces (leading to the main FM synthesis chip), allowing them to be cut and rerouted at will, creating new and varied musical effects. At this time, the author had also been using homemade clock circuits to enable controllable, floating clock rates for all of these devices (enabling arbitrary transposition, in effect). Part of these clock circuits was a binary counter chip that allowed quick clock pitching by exact octave increments.

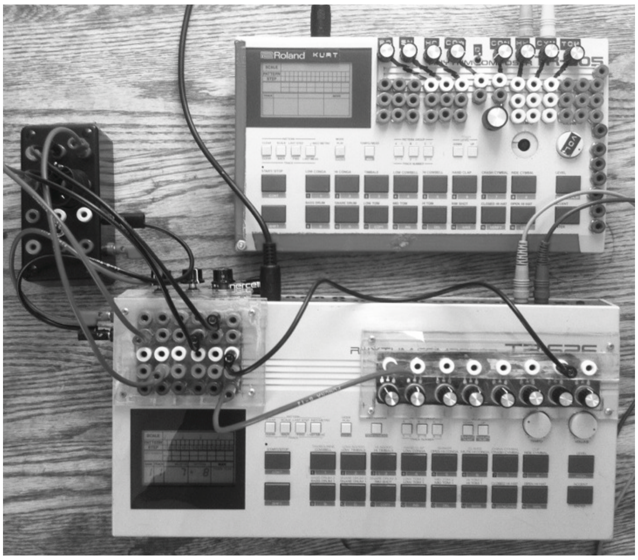

Figure 3: *Circuit-bent Roland TR-606, Kurt James Werner, 2009/2011*

Experiments with shorting jacks on the patchbay through binary counters (connecting one jack to the input pin of the counter, and connecting another jack to various output pins of the counter) yielded even more interesting results. Tapping different output pins of the counter often cause octave-like effects. For instance, moving toward a more significant bit off the counter often causes tonal components of the instrument's sound to drop by an octave but also causes secondary timbral and temporal effects. It is instructive to note that if an arbitrary binary signal is input to a counter, each successively more significant bit off of the counter will result in a roughly one-octave drop in the spectral energy, but also result in a signal that gets closer and closer to a square wave (as shown in Figure 4). These techniques were also extended to the use of non-counter digital logic chips.

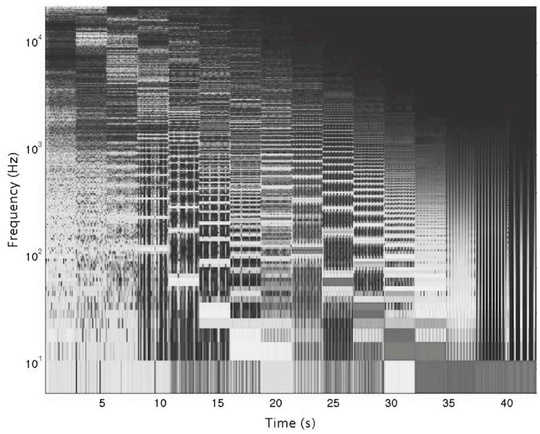

Figure 4: S*pectrogram of circuit-bent instrument with "Counter bend" tapped at successively more significant bits over time*

The exciting results from these initial experiments led to the establishment of a new technique for circuit bending: "Bit Bending." Bit Bending encompasses the insertion of one or more digital logic chips in series with an existing or added circuit trace, in the context of creating a circuit-bent instrument, as shown in Figure 5. So far, Bit Bending has typically been accomplished by housing digital logic chips (counters, NAND gates, etc.) in small ABS project enclosures, and using banana cables to communicate signals and draw power from the main board.

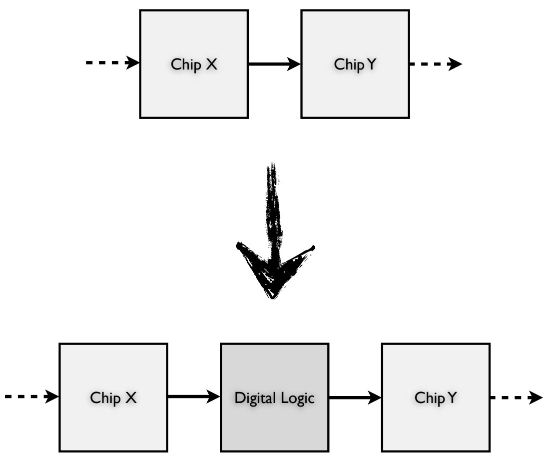

Figure 5: *Basic Bit Bending schema*

2.2. Characteristics of Bit Bending

In addition to creating sonically rich results that balance predictability and chaos, extending the typical circuit bending toolset, and providing musical commentary on the nature of digital information, Bit Bending also taps into a tradition of composition with logical operations. Among his many innovations, Greek composer and theorist Iannis Xenakis made extensive use of logical "sieves" on sets of data as a foundation for many of his compositions [15]. Composition with one bit and logical operations is discussed at length in [16].

Bit Bending was chosen as the object of a circuit-bent modeling approach for several reasons. Since Bit Bending involves logical operations on streams of binary data, one can reasonably assume ideal chip behavior. The assumption of ideal behavior means that Bit Bending (including the chips' internal logic and register behavior) is easily emulated in code. Also, many popular and sought-after devices for circuit bending have hybrid digital/analog circuitry. Analog circuit modeling techniques are well-established (though they may not easily accommodate the arbitrary re-routing of signals and atypical feedback paths endemic to circuit-bent instruments), though a general methodology for approaching the mangling and bending of digital signals is not. Hence, Bit Bending.

3. OUR MODEL

A software library for Bit Bending will make use of the assumption of ideal digital circuit behaviour and existing circuit design abstractions.

3.1. Register-transfer level

In digital circuit design, register-transfer level (RTL) is a design abstraction which models a synchronous digital circuit in terms of the flow of digital signals (data) between hardware registers, and the logical operations performed on those signals [17].

Our circuit modeling library is at the register-transfer level. Actions on the chip and register levels are rising edge-triggered by clock signals. Actions include chip-level actions (latching, performing computations, etc.) and register-level actions (register shifts, transmission and reception of binary information). Registers and all computations have a controllable degree of accuracy through the bit depth.

3.2. Architecture

We will now describe the architecture of our object-oriented software library for Bit Bending.

A real-world circuit involving RTL operations consists of a circuit board populated with interconnected chips, some edge-triggered, some non-edge-triggered, some register-based, some non-register-based.

Our library works by analogy to a real-world circuit. A circuit model involving RTL operations consists of a `CircuitBoard` object populated by many `Chip` objects, interconnected through associated `ChipInput` and `ChipOutput` objects (which may be registers or not). To advance time on a `CircuitBoard`, a tick function is called. A tick call to the `CircuitBoard` advances time on all of its associated chips by one sample, a concept from unit generator (UGen) based synthesis frameworks (such as the Synthesis Toolkit (STK) [18] and ChucK [19]). A schematic diagram is shown in Figure 6.

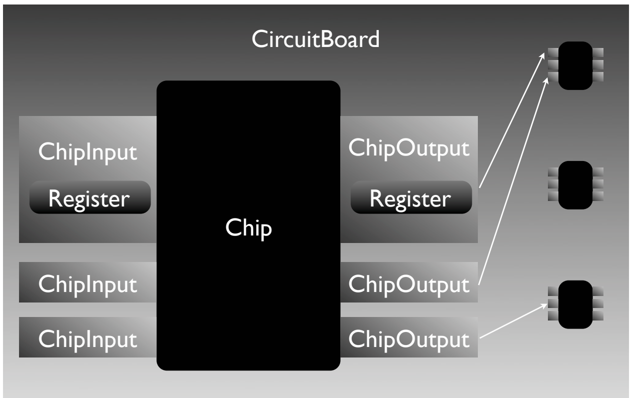

Figure 6: `CircuitBoard` *object schema*

3.3. Chip behavior

Real-world chips can treat input signals in as clocks, or as data streams. Our library utilizes different objects for each case.

Real-world chips are often rising edge-triggered (they perform computations when a particular pin sees a move from low to high voltage). In our library, actions on these types of chips (`FrequencyControlWord`, `PhaseAccumulator`, `Multiplier`, `ADSR`, etc.) occur when their associated `EdgeTriggeredInput` objects see a rising edge.

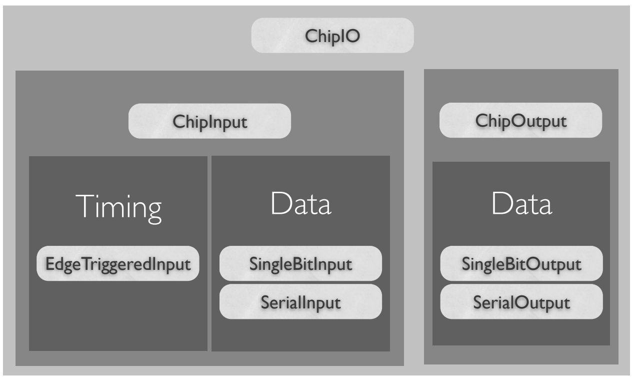

Figure 7: *Class hierarchy for* `ChipInput` *and* `ChipOutput`

Other real-world chips perform calculations continuously (they are asynchronous). In our library, actions on these types of chips (all `Gates`, `Counter`, `Timer`, etc.) are performed on every tick.

Real-world chips that perform complex operations often store a buffer of binary data in a register. `Registers` are associated with certain `ChipInputs` and `ChipOutputs`. These `ChipInputs` are further sub-classed as `SingleBitInputs` and `SerialInputs`, depending on the types of computation that the associated `Chip` performs. Similarly, `ChipOutputs` are sub-classed to `SingleBitOutputs` and `SerialOutputs`. A class hierarchy of `ChipIO` objects is shown in Figure 7.

Though most of the chip models emulate specific real-world chips (the components of the numerically controlled oscillator, logic gates, binary counter, etc.), several are simplified abstractions of higher-level synthesis concepts (such as an attack-decay-sustain-release (ADSR) envelope generator). A class hierarchy of `Chip` objects is shown in Figure 8.

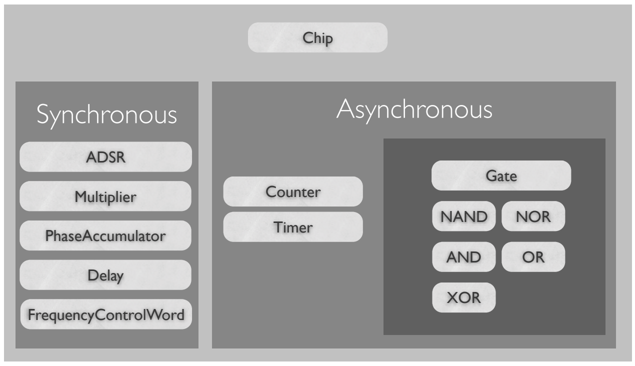

Figure 8: *Class hierarchy for* `Chip` *objects*

3.4. Case Study: 2-operator FM synthesis

Real-world devices that are amenable to Bit Bending are often very complex, involving interactions between various chips (ROM, CPU, dedicated integrated circuits, analog circuitry, etc.). A case study to demonstrate the veracity of Bit Bending must approach a simpler system. As a case study, we built a 2-operator Frequency Modulation (FM) synthesizer using numerically controlled oscillators (NCOs) [20].

3.4.1. Why FM Synthesis?

2-operator FM synthesis is a very flexible and powerful synthesis technique that uses a (usually audio-rate) “modulating” oscillator to control the frequency of a “carrier” oscillator [21].

FM synthesis was chosen as a case study for its simplicity, and since it is easily implementable through cascaded NCOs (which are easily expressible through our library). As well, one of the main strength of FM synthesis (intuitive control and complexity arising from simple and meaningful parameters) is well-aligned with the design ethos of Bit Bending.

3.4.2. Numerically controlled oscillators

Figure 9 shows the model of an NCO. It consists of a Frequency Control Word (FCW) chip, a Phase Accumulator chip, and a Sine Phase to Amplitude Converter (SPtAC) chip. A binary representation of the phase increment associated with the NCO’s frequency is stored in the FCW chip, and sent serially to the Phase Accumulator chip. Every sample, the Phase Accumulator’s stored phase value is incremented by this amount. The SPtAC chip indexes into a sine wavetable lookup based on this phase.

Our case study extends the basic NCO framework by cascading two NCOs together as shown in Figure 10, and by adding Multiplier and ADSR chips (corresponding to the voltage-controlled oscillator (VCA) and envelope generator (EG) in the typical expression of FM synthesis) on the output of the SPtAC chips. The Modulator ADSR chip corresponds to typical FM enveloped control of the Modulation Index (β), and the Carrier ADSR chip corresponds to typical enveloped control of the overall amplitude of the signal.

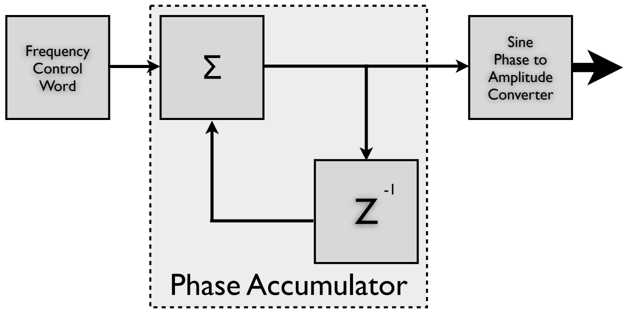

Figure 9: *Numerically controller oscillator block diagram*

We have produced a computer model of FM synthesis implemented with two cascaded NCOs using our software library for RLT. This working model is now ready to be Bit-Bent. Even a simple circuit can be bent or reconfigured in any number of ways. Here we present only a few of the possible Bit Bending modifications to this model.

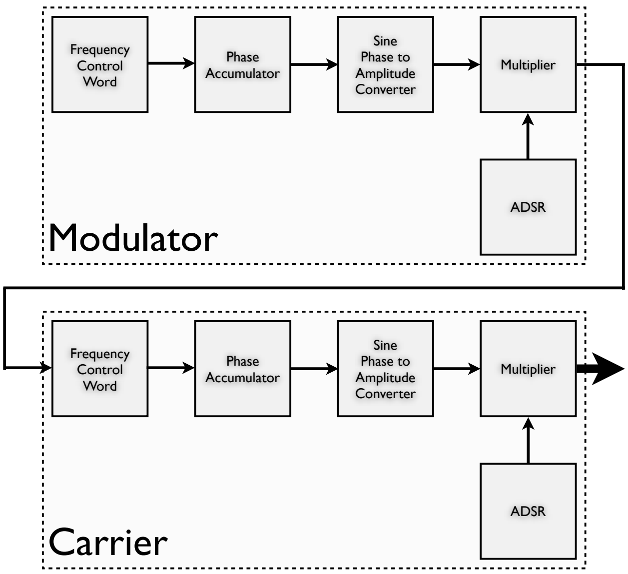

Figure 10: *Implementation of 2-operator FM synthesis with numerically controlled oscillators block diagram*

3.4.3. Bit depth bend

The entire model has a controllable bit depth. This bit depth characterizes many of the computations performed by the chips, and thus the overall behavior of the synthesizer.

Bit depth reduction ("bitcrushing") is a familiar digital audio effect, much used for simple emulation of the sound of historical digital signal processing hardware. Traditional bitcrushing merely represents an audio signal with fewer bits. Bit depth reduction in a Bit Bending context is much more complicated. Bit depth characterizes the frequency accuracy of the Frequency Control Word chip, the phase accuracy of the Phase Accumulator chip, the bit depth of the SPtAC chip, etc. This also effectively causes non-linear, system-endemic waveshaping on both the modulating and carrier wave.

3.4.4. Clock signal bend

Many of the chips in the model have rising edge-triggered actions. Normally, their clock inputs look to the global timing chip. However, cascading additional circuitry in series with this clock signal causes these actions to occur at different times.

A simple case of this would be to cascade a counter chip in series with the clock signal seen by the last chip in the signal flow. An n-bit counter keeps track of an n-bit binary representation of a number that increments when the input transitions from low to high. Taking different pins off of this counter would result in octave-drops of the (square wave) clock signal. This would result in the output chip being polled for sample data less often, and octave reductions in the Nyquist rate – in effect, sample rate reduction by octaves with aliasing.

Much more complicated routings can be imagined and implemented, resulting in highly non-linear sample rate reduction effects. Signal-dependent bandwidth reduction and aliasing artifacts that result from these techniques are unique and completely un-obtainable with traditional sampling rate reduction effects.

3.4.5. Counter bend

Less intuitive Bit Bending extensions to the model involve insertion of additional circuitry along the main signal path. As shown in Figure 11, we inserted a Counter chip at the interface between the carrier SPtAC output and the ADSR multiplier. Taking different pins off of this counter allows for complex, non-linear octave-like effects that are familiar from Bit Bending in hardware.

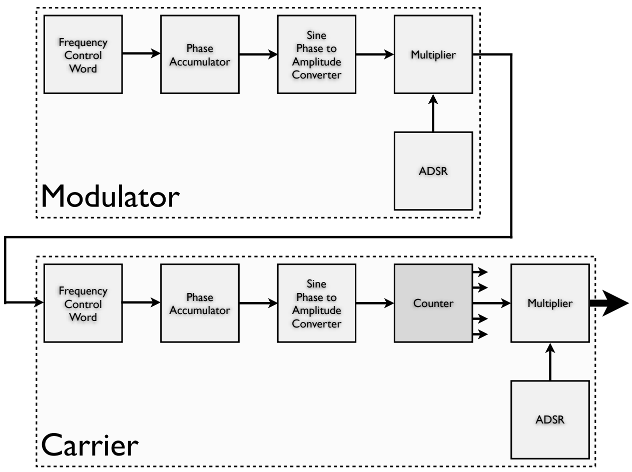

Figure 11: *"Counter bend" block diagram*

3.4.6. Delay bend

Another type of circuit element that can be inserted into the main signal path is a Delay chip. As shown in Figure 12, we inserted a Delay chip between the carrier phase accumulator and the wavetable lookup. The Delay chip "delays" the binary information flowing through it.

Delay bends are highly sensitive to the delay amount, and introduce signal-dependent digital noise, which can have a variety of effects depending on the bend's location in the signal path.

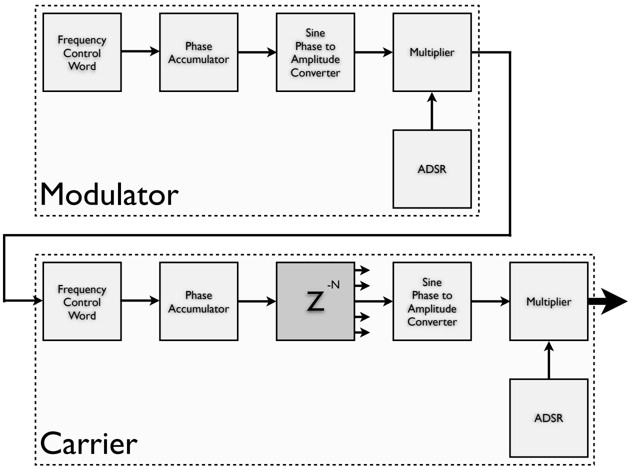

Figure 12: *"Delay bend" block diagram*

3.4.7. Conclusion

The effects obtained through Bit Bending alterations upon this model closely resemble the results obtained through Bit Bending on hardware devices (TR-505, TR-626, etc.). As the case study of the FM synthesizer shows, even simple Bit-Bent alterations and additions to the signal diagram create fantastic sounds that are controllable through very few parameters.

Audio examples can be found at:
http://ccrma.stanford.edu/~mayank/Bit-Bending/

4. PERMUT8

A simple version of this methodology seems to be employed in the design of one commercial audio effect plug-in designed by Sonic Charge, Permut8 [22]. Permut8 is a model of a 12-bit digital delay, where the delay line index can be "controlled by a programmable processor with an assortment of operators [23]," which include logical operations on their binary representation (AND, OR, XOR). Although it is not presented as such, Permut8's processing can be considered a simplified version of Bit Bending; its core operation is on a single binary number (the delay line index), rather than the communication between chips on the register level. Permut8 offers a powerful example of the artistic uses of modeled circuit-bent devices. While it is curiously marketed without reference to circuit bending, reviews (most of them glowing) make frequency note of the inspiration [24, 25, 26, 27, 28].

Permut8's developer Magnus Lindström has done an impressive job of balancing modeling goals and combining analog and digital modeling techniques with this plugin. Permut8 combines a very unique, bottom-up design methodology in its model of a bent digital delay processor with analog modeling techniques on the input and output stages (soft clipping to model analog saturation with low aliasing [29], analog filter models, etc.).

Permut8 is a very powerful example of how exciting and useful a computer model of a circuit-bent instrument or audio effect can be, as well as an instructive demonstration of a well-thought-out design.

5. CONCLUSIONS

We have successfully made a computer model of "Bit Bending." Our 2-operator FM synthesis case study demonstrates that computer modeling of circuit-bent instruments is possible and worthy of study. The model holds true to the sound and experience of working with circuit-bent instruments. This points to just one of many regimes in which circuit bending could be an interesting and fruitful object of study. The aesthetics of circuit bending, associated composition and live performance techniques, social context and ethnography of the circuit bending community, etc. are all other interesting facets.

6. FUTURE WORK

Bit Bending encompasses only a very small slice of commonly utilized techniques for circuit bending, and our case study explores only a small subset of the full scope of Bit Bending. Future work on computer modeling of circuit-bent instruments should focus on other common circuit bending techniques (clock bends, analog feedback, etc.), and integration of digital (Bit Bending, RTL, etc.) modeling methods with analog circuit modeling methods, to get closer to more complete models of bendable instruments. The case of feedback through digital circuits without buffering may necessitate the investigation of non-idealized chip behavior.

Great musical devices are often the result of successful mappings [30]. A powerful synthesis engine needs to be complemented by an expressive user interface to reach its full potential. Future work on Bit Bending will also focus on the creation of user interfaces that allow expressiveness while maintaining unbounded chance discovery and experimentation.

7. ACKNOWLEDGEMENTS

Thanks to John Granzow for allowing us to give an early version of this paper at the CCRMA Colloquium series in January 2013 [31]. Thanks to Jorge Herrera for helping us put a name to our new technique ("Bit Bending"), and its computer-modeled counterpart. Thanks for helpful and encouraging feedback from professors Julius Orion Smith III and John Chowning, and many others.

8. REFERENCES

[1] Q. R. Ghazala, "What is circuit-bending, anti-theory & clear-illogic," in *Experimental Musical Instruments*, vol. 14, no. 4, Jun. 1999.

[2] Q. R. Ghazala, "The Folk Music of Chance Electronics: Circuit-Bending the Modern Coconut," *Leonardo Music J.*, vol. 14, pp. 97-104, 2004.

[3] A. Fernandez and F. Iazzetta, "Circuit-Bending and the DIY Culture." Escole de Communivações e Artes, Universidade de São Paulo, 2011.

[4] Burnkit 2600, *Burnkit 2600*, [online] 2012, http://www.burnkit2600.com(Accessed: 14 February 2013).

[5] Casper Electronics, *Casper Electronics*, [online] 2012, http://www.casperelectronics.com (Accessed: 15 February 2013).

[6] S. Kuznetsov and E. Paulos, "Rise of the expert amateur: DIY projects, communities, and cultures," in *Proc. of the 6th Nordic Conference on Human-Comput. Interaction: Extending Boundaries*, 2010, pp. 295-304.

[7] L. Russolo, "The Art of Noise." Mar. 1913.

[8] K. Stockhausen, "...how time passes..." *Die Reihe*, vol. 3, 1957.

[9] C. Roads, "Time Scales of Music," in *Microsound*, 1st ed. Cambridge, MA: MIT Press, 2004.

[10] R. Ghazala, "Acknowledgements," in *Circuit-Bending: Build Your Own Alien Instruments*. Indianapolis, IN: Wiley, 2005.

[11] A. Glinsky, *Theremin: Ether Music and Espionage*. Urbana and Chicago, IL: University of Illinois Press, 2000.

[12] J. O. Smith, *Physical Audio Signal Processing*. W3K Publishing, 2010.

[13] B. Edwards, "30 Years Later, Richard Wiggins Talks Speak & Spell Development," blog, 16 Dec. 2008; http://www.vintagecomputing.com/index.php/archives/528.

[14] N. Collins, "Laying of Hands: Transforming a Portable Radio into a Synthesizer by Making Your Skin Part of the Circuit," in *Homemade Electronic Music: The Art of Hardware Hacking*, 2nd ed. New York, NY: Routledge, 2009.

[15] I. Xenakis, *Formalized Music: Thought and Mathematics in Composition (Harmonologia Series, No 6)*. Pendragon, 1992.

[16] V. Adán, "Discrete Time 1-bit Music: Foundations and Models," D.M.A. dissertation, Music Dept., Columbia Univ., New York City, NY, 2005.

[17] F. Vahid, *Digital Design with RTL Design, Verilog and VHDL*, 2nd ed. Wiley, 2010.

[18] P. Cook and G. Scavone. "The Synthesis Toolkit (STK)." In *Int. Comput. Music Conference*, Beijing, China, 1999.

[19] G. Wang, "The ChucK Audio Programming Language: 'A Strongly-Timed And On-The Fly Environ/Mentality," Ph.D. dissertation, Dept. of Comput. Sci., Princeton Univ., Princeton, NJ, 2008.

[20] J. Radatz, *The IEEE Standard Dictionary of Electrical and Electronics Terms*, IEEE Standards Office, New York, NY, 1997.

[21] J. Chowning, "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation." *J. of the Audio Eng. Soc.*, vol. 21, pp. 526-534, Sep. 1973.

[22] Permut8, *Sonic Charge*, [online] 2012, http://soniccharge.com/permut8 (Accessed: 10 February 2013).

[23] Permut8 user manual, *Sonic Charge*, [online] 2012, http://cdn.soniccharge.com/public/Permut8 User Guide.pdf (Accessed: 10 February 2013).

[24] Sonic Charge Permut8 review, *Plugin Boutique*, [online] 2012, http://www.pluginboutique.com/articles/137 (Accessed: 10 February 2013).

[25] Sonic Charge releases Permut8 effect, *Music Radar*, [online] 2012, http://www.musicradar.com/news/tech/sonic-charge-releases-permut8-effect-556508 (Accessed: 10 February 2013).

[26] S. Raoofi, "Sonic Charge Permut8", (Attack Magazine), [online] 2012, http://www.attackmagazine.com/reviews/gear-software/sonic-charge-permut8 (Accessed: 1 Apr. 2013).

[27] Sonic Charge Permut8 'Excels In Digital Noises Of All Kinds', *Synthtopia*, [online] 2012, http://www.synthtopia.com/content/2012/08/07/sonic-charge-permut8-excels-in-digital-noises-of-all-kinds (Accessed: 2 Apr. 2013).

[28] P. Kirn, "Sonic Charge Permut8: All Your Lo-Fi Digital Mangling Effects in One Place," (Create Digital Music), [online] 2012, http://createdigitalmusic.com/2012/08/sonic-charge-permut8-all-your-lo-fi-digital-mangling-effects-in-one-place (Accessed: 1 April 2013).

[29] H. Thornburg, "Antialiasing for Nonlinearities: Acoustic Modeling and Synthesis Applications." In *Int. Comput. Music Conference*, Beijing, China, 1999.

[30] R. Henke. Music 223M. Class Lecture, Topic: "Parameterization and Mapping." CCRMA, Stanford University, Stanford, CA, Apr. 8, 2013.

[31] Kurt Werner and Mayank Sanganeria: BitBending - circuit bending brought to software, *CCRMA*, [online] 2013, http://ccrma.stanford.edu/events/kurt-werner-and-mayank-sanganeria-bitbending-circuit-bending-brought-software (Accessed: 12 April 2013).