**Alex Thorogood**
The Australian National University
Center for New Media Arts
Faculty of Arts
121 Childers Street
Acton, 0200
Australia
u4118319@anu.edu.au

# Algorithm Gadget: An Improvisatory Rhythm Generator

## Abstract

*This paper describes an algorithmic drum performance device for musical expression. The idea is to provide maximum variation and dynamics with minimal input. An algorithm programmed into a PIC micro-controller acts as the 'brains' of the device taking data from inputs and forming an equation with the variables, outputting the result to an audio drum chip. The drum chip was derived from a basic musical toy, which was chosen for its low cost, loaded sample set and easy implementation. A deciding factor in choosing the components and configuration was to reduce the overall size of the device so as to make usage possible in either a larger live performance setting, or whilst sitting in a smaller area (e.g bus) for personal entertainment.*

## Introduction

Algorithmic composition has been widely used as a structure for musical composition and creativity to varying ends throughout styles/genres since numbers were related to sound (Pythagoras circa 580BC). Although this "automation" in composition has in the most part been used in solid compositions (e.g. canonic composition 15th c., serial music circa. 1920). It was not until the jazz movement that live and improvised rhythm and melody, tied in with known structures broke the compositions from the page. The circumventing of electronic and computer art/music performance has made use of tools that extend and supplement the idea of the composition as a written work, then further engaging with intangible elements to create intrinsically compelling musical structure and sound aesthetics.

The intention here was to create an algorithmic improvisation hardware device and look at what aspects of the design and build would be conducive to streamlining thought and process in live performance. An algorithm was needed that could provide a wide ranging and dynamic output from minimal input and still be loaded into a small package. This paper describes the motivation, design, implementation and preliminary evaluation of a novel algorithmic percussion device meeting these goals and some directions for further exploration of this notion.

## Motivation

The project was prompted by experiences in performance using multi modal interfaces. During live performance little time is reserved for programming rhythm sections 'on the fly' as most of the time attention is on creating a number of other motifs simultaneously. Typically during a live collaboration a number of aspects of a sound piece are being processed at the same time and a rhythm section is required. These concurrent requirements can make the thought process of the work disjointed and other performance tasks may be put aside as programming a drum machine takes the mental thread.

In exploring the possibilities of composing with equations in the area of live performance and the most efficient means of achieving such a thing, it was hoped that it would reduce the composition times of processing musical parts and produce varying and dynamic outcomes. It was important to keep to the idea of instrumentation by means of ubiquitous computing (Weiser et al. 1999), hence the ideology of a hardware instrument over a software application.

A sonically diverse range of sound sets was desired as outputs from the drum performance device. This transcends the procedure of pattern formation and focus is to given to the aesthetic of sound.

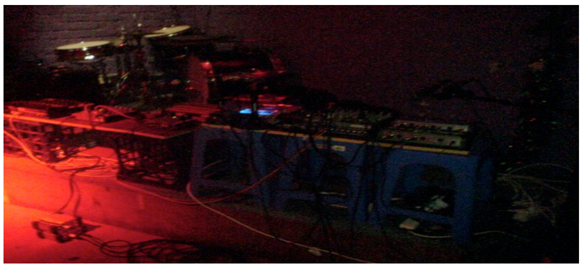

***Figure 1***. *Typical performance setting.*

## Build

Starting with a short brief of what was the required outcome and given the known technologies to work with, it was then a matter of assembling the data into a workable product concept.

A flexible design approach was used to meet these objectives. This approach meant having the drum device evolve during the process, thus leaving an amount 'creative control'. This was achieved through a flexible circuit design. The idea of circuit design flexibility is to allow for all the necessary elements to be included, leaving space for additional features and allowing further ideas to be easily patched in as the construction takes place or later when revisions to the unit are desired.

Following from this framework a list of possible components was compiled to form a starting point for the project.

| | |
|---|---|
| *50k & 100k potentiometer* | *1 x 3.5mm stereo socket* |
| *3 x 330R* | *1 x toggle switch* |
| *1 x 10kR* | *1 x battery pack* |
| *1 x 100uf cap* | *jiffy box* |
| *4 x LED* | *proto board* |
| *1 x PICAXE 08M* | *1 length of pin strip* |
| *1 x audio drum IC* | *toy buttons (for knobs)* |

***Table 1.*** *Final component list.*

## Process

The proto board was cut into different sized pieces to separate the component blocks to fit at different locations in the jiffy box. A main board was cut with eighteen(18) three(3) hole buses and fifteen(15) two(2) hole buses. This was more than adequate for the number of components to be installed and left room for further additions to be made if required. The board was then scanned and printed onto paper to make component layout easier. The board for the LED strip was cut to accommodate LED's with separate positive rails and a shared 0V rail. The board for the PIC chip was made so that the IC could be positioned on the outside of the box. This positioning allowed for easy removal for re-programming and added to the aesthetic of the design. The associated components were then soldered to the boards along with connecting wires for their input/output.

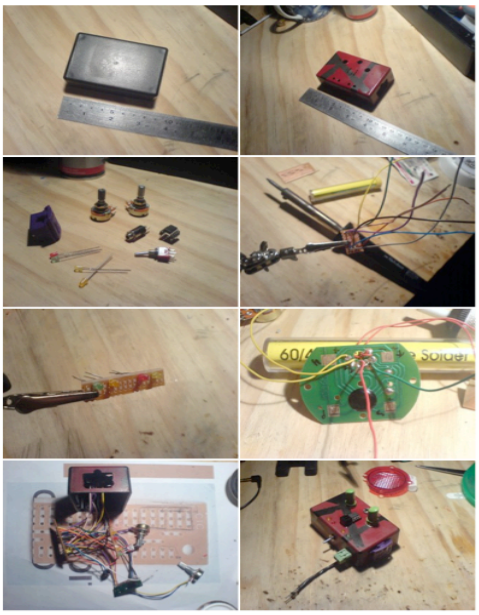

***Figure 2.*** *Pictures of the build*

## Affordance

Before bread boarding, the separate component blocks and additional components were fitted in the jiffy box and holes drilled where necessary. It was necessary to identify the intended use and the ways that the unit may be used in performance when placing the components to the box, allowing for interesting and free flowing interaction with a simple and uncluttered process.

The spacing was done in a way to allow a variable (*n*) to be changed by the left potentiometer and inputted (to the PIC) by a momentary switch located in the centre within a single hand position, then letting the hands return to the default position with minimal movement to further play with the circuit bend (right potentiometer).

It was decided not to make other inputs available so as to not clutter the interaction process and to stay close to the desired resolve of minimal input. This type of bi-manual manipulation for the pattern structure and sound mutation allows for interesting and free flowing interaction. Cognitive studies show the richness and efficiency that this type of simple interaction can deliver (Leganchuk et al 1998 ) (Buxton and Myers 1986). The visual feedback (LED strip) was placed to allow for reference with eyes in the same view as the input.

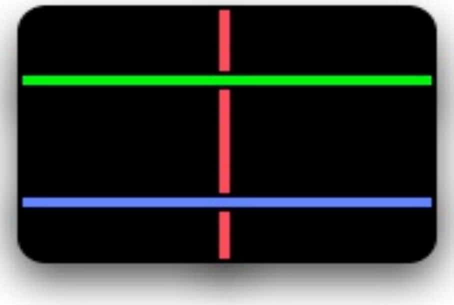

***Figure 3.*** *Affordance map of control layout*

## PIC abstraction

The PICAXE 08M micro-controller is an eight(8) pin package that contains a minimum number of features that were attractive for the project. These are: one ADC input, one digital input and three general input/output pins. The most easily implemented audio output from the PICAXE 08M[1] is a variable PWM, giving unfavorable results for the intended outcome. Thought was given to an electronic toy drum (the IC running the toy has four sounds: kick, snare, hi-hat, tom) and augmenting this with known circuit bending techniques (Ghazala. 2005) then implementing the hack into the current ciruit design.

Assignment of pins was as follows: ADC input to left 50k potentiometer, digital input to momentary switch and general in/out pins to input of the audio drum IC.

## Interaction layer

A user interacting with the drum device would be given access to a single button and two knobs which allows a

[1] Software and documentation – www.picaxe.co.uk/ (Referenced 28-03-06)

series of values to be entered by first turning a knob to an entry value then placing the value into the equation by pushing the button. When deciding on an appropriate program to implement a number of factors were essential. Firstly there has to be a means of gaining enough data entries to make the equation dynamic and variable. This was realised by using a polled interrupt from the digital input which reads the value of the ADC input into variable ($x_1$) and setting a switch ($s$) to one(1). On return from interrupt, if switch ($s$) is equal to one(1) then program jumps to a sub routine that resets switch ($s$) to zero(0) then moves to sub routine determined by the value of an internal variable ($x_2$=0) (branch number) where ADC input ($x_1$) is placed in the corresponding variables($n_1$ ,$n_2$ ,$n_3$) and internal variable ($x_2$) (branch) is incremented every time a new variable is stored, then returning back to main program. When internal variable ($x_2$) (branch) is reached ($x_2$) is returned to zero(0).

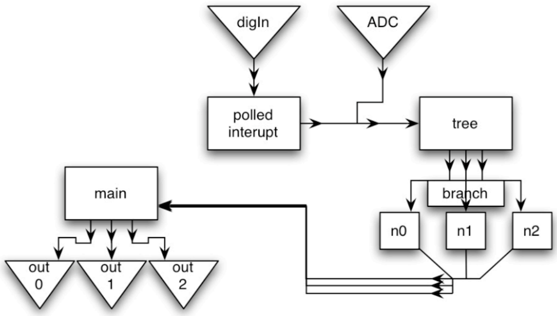


***Figure 4.*** *Abstract of program schematic*

## Implement an Algorithm

Input from the user was translated into various drum sounds and patterns. To do this an algorithm was needed that would take the input data using it to produce both a time scale and drum output trigger. This polymorphic behaviour of the output was intended to create a musical correlation between time and sound. At the same time, the generated data set was expected to produce an aleatoric music so that the feeling of the minimal instrumented interaction would still be alive. Execution of the equation would be done every 60ms or equivalent to 8th notes at 120bpm. This ran the risk of producing static sets of patterns that would be unfavourable for the intended purposes. Thus tempo deviations and perceptual "flaws": tatums per measure (Blimes 1993) were needed to make the drum device sound "alive".

To produce these tempo deviations, given the types of input and structure/restrictions of the program language to be used, the pulse length was used to give each pattern seemingly different characteristics. Variations of the equation were tested out in Processing (Beta103)[2] with a GUI interface for dynamic input/output to sketch the operation of the BASIC program. This was done to reduce the number of times the micro was required to be re-programmed and minimise the risk of damaging the PIC device.

The length of pulses to implement rhythmic timing is computed using the following.

$$r_n = (r_{n-1} + x + yz) \% m$$

where $r_n$ is the output value at time $n$ , variables $x,y,z$ are inputs and $m$ is a constant which is the maximum output value.

The value $r_n$ was then used to evaluate the pin output $d_n$ for the pulse:

$$d_n = r_n \% c$$

$d_n$ is the pin number to send the pulse to, $r_n$ is the pulse length and $c$ is a constant relating to the number of pins for trigger output.

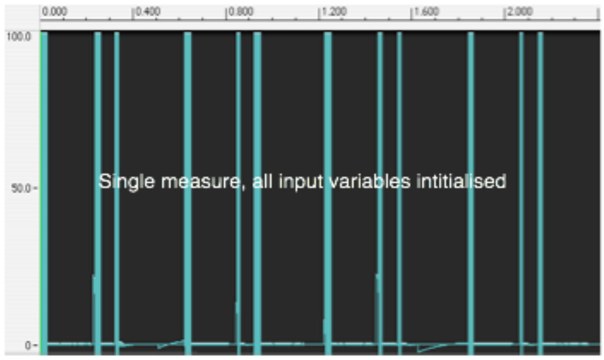


***Figure 5.1.*** *Timing resulting in contemporary structure.*


***Figure 5.2.*** *Timing with a higher stochastic identity.*

2 Software and documentation – www.processing.org/ (Referenced 28-03-06)

## Experience

When completed the drum device was given to a number of people to try. This was to see if, as an instrument, it was understood in the intended manner. Initial reactions were of curiosity as the design generally looks like it "does something". People used the input knobs to try and do something to the sound but were not sure how they were affecting the sound. Nor did they realise that the button was used to enter data. The minimal input design for data entries proved obscure for non-instructed users. If the intention was to release the drum device to a number of people then symbols on the panelling may prove useful. Further studies of this will need to be carried out if a wider implementation is to be undertaken.

Using the algorithm, when all variables are loaded and the equation set in motion, the output produced a wide range of results. Depending on the input the resulting patterns would either seem to lose their rhythmic integrity or have a stable contemporary structure that held some familiar characteristics. The device was taken 'out on the road' and used for live performance at *The Make It Up Club* (Bar Open, Melbourne, 14/02/2006). Throughout the performance the device held up for its intended purpose in complimenting the varying motif of the improvised arrangement. It was also shown to extend the idea of a quick improvisatory tool in that further augments to the sound were carried out using the rapid repetitions of some cycles that the algorithm would fall into. Using basic equalisation the ornament added to the total motif with a new sonic texture. Without treatment the sound set of the drum IC sounded gritty (which was favourable for the performance) and no control was given to switching the individual drum sounds on or off. If a greater control of each drum sound is desired, a simple implementation of a toggle switch should be added for each drum sound.

The layout of input controls was simple and did not clutter headspace during performance. However, the total workable area proved to be at times a little fiddly. Future similar devices should include more ergonomic attributes so as to fit more comfortably in the hands.

The successful execution of patterns could be done quickly within movement changes of the improvised composition. The solid build of the unit proved to be useful for not having to be overly concerned with damaging the device. This idea of a robust technological instrument has been shown with the boomBox (Allen 2005), where it is a premise that musical gesture/interaction can be tough and at times aggressive.

## Conclusion

This paper describes the requirements, design, implementation and first use of a rhythm generator that creates an interesting musical structure with minimal input from the performer. With future improvements to the algorithm, more advanced structures of rhythm may be achieved. The completed drum device shows promising direction for the avenue of small hardware units for improvised algorithmic composition. When used in live performance the system was found to fulfil the desire of making a rhythmic sound layer without taking large amounts of cognition from other ongoing performance tasks. Taking into account the restricted parameters of the micro, a future approach will be to use a package with a richer feature set. The expectation is that this will generate a greater range of functions thus adding a higher level of performance interaction.

## References


Allen, Jamie. 2005. "boomBox" Proceedings of the *2005 International Conference on New Interfaces for Musical Expression.* Vancouver.

Ariza, C. 2005. "Navigating the Landscape of Computer-Aided Algorithmic Composition Systems: A Definition, Seven Descriptors, and a Lexicon of Systems and Research" Proceedings of the *International Computer Music Conference.* San Francisco.

Blimes, Jeff A. 1993. "Tequniques to Foster Drum Machine Expressivity" Proceedings of the *International Computer Music Conference.* Tokyo.

Buxton, William. and Myers , Brad A. 1986. "A study in two-Handed Input" *CHI '86 Proceedings.* Massachusetts.

Ghazala, Reed. 2005. *Circuit-Bending: Build Your Own Alien Instruments.* Wiley, Indianapolis.

Leganchuk, Andrea. Zhai, Shumin. and Buxton, William. 1998. "Manual and Cognitive benefits of Two-Handed Input: An Experimental Study" *ACM Transactions on Computer-Human Interaction.* December. 5, 4.

Meadows, Eddie S. ed. 1995. *Jazz Research Materials: A Selected Annotated Bibliography.* New York: Garland Science.

Pope, Stephen Travis. 1995. "Fifteen Years of Computer-Aided Composition" *Computer Music Journal, And CNMAT.* Dept. Of Music, U.C. Berkley.

Weiser, M. Gold, R. Brown, J.S. 1999. "The Origins of ubiquitous computing research at PARC in the late 1980s" *IBM Systems Journal.* 38, 4.