Physical Livecoding with littleBits

James Noble
Victoria University of Wellington, NZ
kjx@ecs.vuw.ac.nz

ABSTRACT

littleBits (littleBits.cc) is an open-source hardware library of pre-assembled analogue components that can be easily assembled into circuits, disassembled, reassembled, and re-used. In this demonstration, we will show how littleBits — and the KORG littleBits SynthKit in particular — can be considered a physically-embodied domain specific programming language, and thus how assembling or improvising music with littleBits circuits is a tangible form of livecoding.

1. Introduction

littleBits (littleBits.cc) is an open-source hardware library of pre-assembled analogue components that can be easily assembled into circuits, disassembled, reassembled, and re-used (Bdeir 2009). Designed to inspire and teach basic electrics and electronics to school-aged children (and adults without a technical background) littleBits modules clip directly onto each other. littleBits users can build a wide range circuits and devices with "*no programming, no wiring, no soldering*" (Bdeir 2013) — even extending to a "Cloud Module" offering a connection to the internet, under the slogan "*yup. no programming here either* [sic]" (littleBits 2014).

The littleBits system comes packaged as a number of kits: "Base", "Premium", and "Deluxe" kits with 10, 14, and 18 modules respectively; and a series of booster kits containing lights, triggers, touch sensors, and wireless transceivers. littleBits have recently introduced special purpose kits in conjunction with third party organisations, notably a "Space Kit" designed in conjunction with NASA, and a "Synth Kit" designed in conjunction with KORG that contains the key components of an analogue modular music synthesizer.

In spite of littleBits' marketing slogans, we have argued that littleBits —- and the littleBits Synth Kit in particular (Noble and Jones 2014a; Noble and Jones 2014b) — as a live physically-embodied domain specific programming language. If building littleBits circuits is programming, then performing music with the littleBits Synth Kit (configuring modules to construct an analogue music synthesizer, and then producing sounds with that synthesizer) can be considered as a music performance by live coding (McLean, Rohrhuber, and Collins 2014) — especially as the circuit construction typically occurs simultaneously with sound production.

The demonstration draws directly on previous demonstrations given at the FARM workshop in Uppsala (Noble and Jones 2014a) and the VISSOFT conference in Victoria (Noble and Jones 2014b).

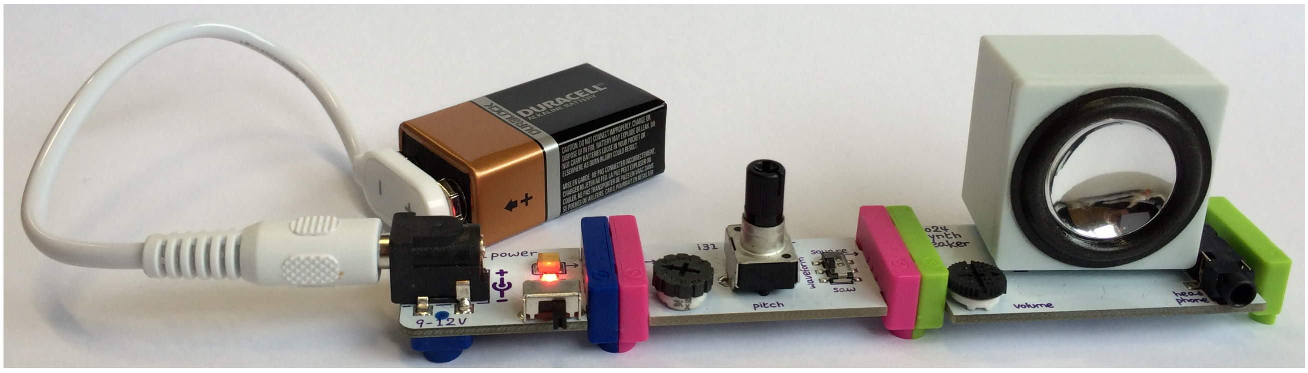

Figure 1: *A simple littleBits Synth Kit circuit. From left to right, the three modules are a power source, an oscillator, and a speaker.*

2. The littleBits SynthKit

Figure 1 shows the one of the simplest circuits in the littleBits synth kit — indeed, the simplest littleBits circuit that can actually make any sound. This circuit is composed of three simple modules — a power module on the left, an oscillator module in the centre, and a speaker module on the right. The power module accepts power from a nine volt battery (or a 9V guitar pedal mains adapter) and provides that power to "downstream" modules — as seen in the figure, littleBits circuits flow in a particular direction, and all modules are oriented so that this flow is left to right.

3. Livecoding with littleBits

If building and configuring circuits with littleBits can be considered as a form of embodied, tangible, programming, then performing music "live" with littleBits can be considered as a form of livecoding — performance programming to produce music (McLean, Rohrhuber, and Collins 2014; Blackwell et al. 2014; Magnusson 2014) — or in this case both *building* and *playing* synthesizers as a live performance. In this section we describe our practice livecoding littleBits, and compare and contrast with typically textual livecoding (inasmuch as typical livecoding practice can be supposed to exist).

This demonstration (Noble and Jones 2014a; Noble and Jones 2014b) draws on the author's experience livecoding/performing littleBits with "Selective Yellow", an experimental improvisation duo of indeterminate orthography drawing on New Zealand's heritage of experimental music practice (Russell 2012; McKinnon 2011), yet seeking to recreate (electronically) all the worst excesses of free jazz with all the enthusiasm of antisocial teenagers meeting their first MOS6851, while maximising the time required to set up equipment.

Selective Yellow performances typically employ a number of different synthesizers or sound generators as well as littleBits, ranging from digital toys (Kaosscilators, Buddhamachines) to semi-modular analogue and MIDI digital synthesizers, played with a variety of controllers (wind controllers, monome grids, knob boxes etc) — while eschewing a primary role for digital audio workstation software and computer-based virtual instruments. Selective Yellow is still a relatively young project, probably only Grade 2 as evaluated by Nilson (Nilson 2007).

Livecoding with littleBits involves two main activities that are tightly interleaved in a performance, first building the circuits by clipping modules together, and second "playing" the resulting synthesizer by turning the shafts, thumbwheels, switches, the "keys" on the keyboard module to actually generate sound. Generally a performance — or rather the portion of the performance improvised upon littleBits — starts with the smallest possible sound-generating circuit, typically the single unmodulated oscillator in figure 1. Once the littleBits are assembled (and the speaker module's output patched into the sound system) we can manipulate the oscillator's pitch and output waveform. Depending on the context of the improvisation, the possibilities of such a straightforward sound generator will be more or less quickly exhausted, at which point the performer will disassemble the circuit, insert one or more additional modules (a second oscillator, a filter, or perhaps a keyboard or sequencer module) and then continue playing the resulting circuit. In this overall pattern, littleBits livecoding is similar to some textual livecoding, where performers typically start with a single texture and then build a more complex improvisation over time.

While the circuit building and playing are conceptually separate activities, an advantage of the physical nature (and careful design) of the littleBits components is that the two activities can be very tightly interleaved. Indeed, with more complex circuits (or more than one Synth Kit) its is quite possible to keep part of a circuit playing and producing sound (such as a sequencer driving an oscillator) while building/editing another branch of the same circuit — adding in a second oscillator controlled by the keyboard module with an independent output route, perhaps, or adding in a modulation path to a filter that is already making sound in the main circuit. Again, this overall dynamic is also found in some textual livecoding performances (see e.g. the SuperCollider jitlib (Collins et al. 2003)). Of course, because of the underlying simplicity of the analogue synthesizer embodied within the littleBits modules, the sounds produced by littleBits Synth Kit are much less complex than the sounds that can be produced by a general-purpose laptop running a range of digital synthesis or sampling (virtual) instruments, although, being purely analogue, they have a piercing tone all of their own.

In the same way that programmatic live coders generally display the code on their laptop screens to the audience of the performance (Collins et al. 2003), Selective Yellow projects an image of the desk or floor space where the little bits circuits are being assembled. The projected image not only seeks to dispel "dangerous obscurantism" (Ward et al. 2004) but also to illustrate how the source is being generated - especially as some modules include LEDs as feedback to the performer. The sequencer module, for example, lights an LED to indicate the current sequencer step, and other littleBits modules can also be used to provide more visual feedback on circuits where that is necessary.

This projected display seems particularly useful for audiences when the performer is "debugging" their circuit (live). Here again the physicality of the littleBits modules comes to the fore, so there is something for the audience to see: the easiest way to debug a littleBits circuit is just to pull it apart, and insert a speaker module after each module in the circuit in turn, listening to the sound (if any) being output by each module. Often this lets the performer understand (and the audience to notice) that there is no sound output from a littleBits circuit, allowing the performer either to readjust the module parameters, or to re-assemble the circuit in a different design, if not producing the desired sound, at least producing something.

4. Acknowledgements

Thanks to Chris Wilson, for being the other half of Selective Yellow. Thanks to a wide range of anonymous reviewers for their comments. This work is partially supported by the Royal Society of New Zealand (Marsden Fund and James Cook Fellowship).

5. References

Bdeir, Ayah. 2009. "Electronics as Material: littleBits." In *Proc. Tangible and Embedded Interaction (TEI)*, 397–400.

———. 2013. "LittleBits, Big Ambitions!" `http://littlebits.cc/littlebits-big-ambitions`.

Blackwell, Alan, Alex McLean, James Noble, and Julian Rohrhuber. 2014. "Collaboration and Learning Through Live Coding (Dagstuhl Seminar 13382)." *Dagstuhl Reports* 3 (9): 130–168.

Collins, Nick, Alex McLean, Julian Rohrhuber, and Adrian Ward. 2003. "Live Coding in Laptop Performance." *Organised Sound* 8 (3) (December): 321–330.

littleBits. 2014. "Sneak Peek: the Cloud Block." `http://littlebits.cc/cloud`.

Magnusson, Thor. 2014. "Herding Cats: Observing Live Coding in the Wild." *Computer Music Journal* 38 (1): 8–16.

McKinnon, Dugal. 2011. "Centripetal, Centrifugal: Electroacoustic Music." In *HOME, LAND and SEA: Situating Music in Aotearoa New Zealand*, edited by Glenda Keam and Tony Mitchell, 234–244. Pearson.

McLean, Alex, Julian Rohrhuber, and Nick Collins. 2014. "Special Issue on Live Coding." *Computer Music Journal* 38 (1).

Nilson, Click. 2007. "Live Coding Practice." In *New Interfaces for Musical Expression (NIME).*

Noble, James, and Timothy Jones. 2014a. "[Demo Abstract] LittleBits Synth Kit as a Physically-Embodied, Domain Specific Functional Programming Language." In *FARM.*

———. 2014b. "Livecoding the SynthKit: littleBits as an Embodied Programming Language." In *VISSOFT.*

Russell, Bruce, ed. 2012. *Erewhon Calling: Experimental Sound in New Zealand.* The Audio Foundation; CMR.

Ward, Adrian, Julian Rohrhuber, Fredrik Olofsson, Alex McLean, Dave Griffiths, Nick Collins, and Amy Alexander. 2004. "Live Algorithm Programming and a Temporary Organisation for Its Promotion." In *READ_ME — Software Art and Cultures*, edited by Olga Goriunova and Alexei Shulgin.