Nicolas Collins
Before Apple There Was Kim – the Microcomputer, Music and Me
September, 2009

Background

For as long as I've been making music I've been making music with electricity. I began as a talentless player of electric guitar, then started messing around with a reel-to-reel tape recorder. I built my first circuit in high school, and was fortunate to have had the opportunity to learn from such legendary silicon luthiers as David Behrman and David Tudor while I was a student at Wesleyan University.[1] My college years (1972-76) were an interesting time in the co-evolution of electronic technology and its cultural applications: many useful modules (oscillators, amplifiers, etc.) were becoming available as Integrated Circuits, which a non-engineer could interconnect with an almost Lego-like ease to build up musical systems that would have been daunting in the earlier days of the transistor.

The 1970s also saw the rise of the microcomputer. Integrated Circuits shrank room-filling mainframes down to suitcase-size and smaller, and took computing power out of the hands of IBM technicians and made it accessible to a much larger (and more diverse) base of users and experimenters.[2] Shopping for parts, one was a likely to find digital logic chips, such as an AND gate or a chip to add two numbers, side-by-side with analog opamps or oscillators. In keeping with this proximity of analog and digital building blocks, several composers built instruments at this time that foreshadowed interactive computer music in their combining sound generators with logical decision-making circuitry.

The earliest example of which I am aware is Gordon Mumma's *Hornpipe* (1967) which used what Mumma referred to as "cybersonic circuitry" (essentially an analog computer) to analyze room acoustics and control the behavior of filter circuits.[3] In Paul DeMarinis' *Pygmy Gamelan* (1972), pseudo-random patterns generated by digital circuits triggered resonant filters to produce pleasing algorithmic music. For his *Cello With Melody-Driven Electronics* (1974) David Behrman built an interactive sequencer out of digital logic chips so that a musician could control the tuning of oscillator chords by playing specific pitches. My 1979 composition *ANDS* was inspired by Christian Wolff's "coordination" scores (such as *For 1, 2 or 3 People*[4]): it incorporated CMOS logic circuitry that detected coincidences in key-presses

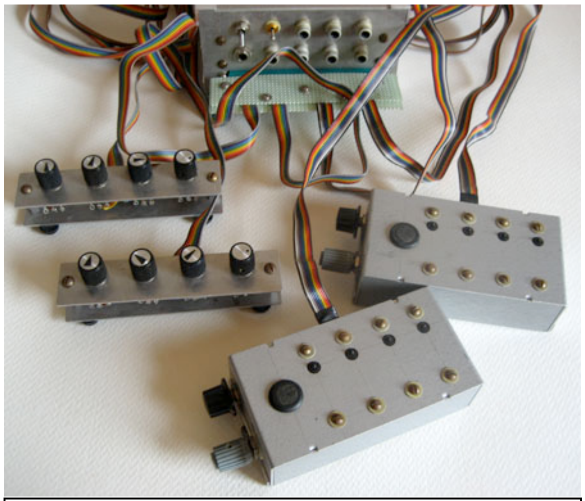
Figure 1: Circuitry for *ANDS*, Nicolas Collins, 1979.

on the performers' small keyboards, as well as generating sound from digital shift registers rather than traditional oscillators.[5]

The Kim-1

When the Kim-1 microcomputer was introduced by MOS Technology in 1976 (primarily as a demonstration platform for the 6502 MOS's 6502 microprocessor that was the heart of the computer), a number of young composers decided to hang up their soldering irons and learn programming. The Kim was so much less threatening than traditional computers – from its folksy name to its resembling, in Paul DeMarinis' words, "an autoharp with a calculator glued on top" – and it was cheaper than any synthesizer at the time ($245-.)[6] For those who had been experimenting with incorporating logic chips in their musical circuits, the Kim provided the missing element: memory. Now it was possible to program a time-varying sequence of events (i.e., embed a score in software), and to incorporate conditional decisions made on the basis of incoming information (such as branching to different parts of the score according actions performed by the players.)

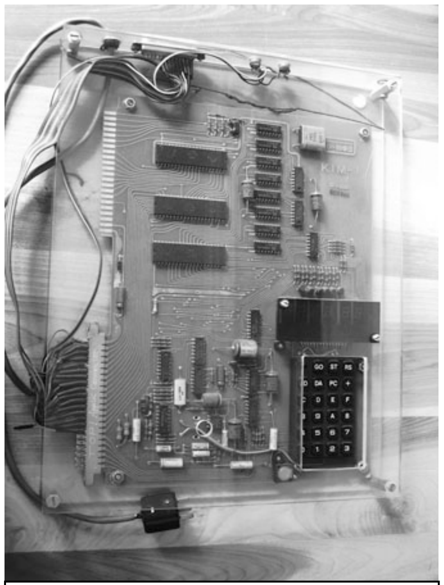

Figure 2: David Behrman's *Kim 1*, 1977.

David Behrman, for example, immediately extended the premise of his *Music for Homemade Synthesizer with Pitch Sensitive Windows* to take advantage of this larger memory space; he replaced his earlier logic circuitry with the Kim, which controlled his bank of homemade analog oscillators through a Digital-to-Analog converter (DAC.)[7] As the name suggests, a DAC was essential for making the connection between the digital chips of the computer and any analog sound-producing circuitry, and this proved to be a serious obstacle for many composer/programmers. There were no of-the-shelf DACs that could just plug into a slot on the Kim; they had to be built up from surprisingly expensive (at that time) chips, and the circuit designs were not simple. As the Kim user-base grew so did the number of small companies producing peripherals for the computer, but in the early days owners were at the mercy of their immediate community. As it happened, Behrman was teaching at CCM (Center for Contemporary Music) at Mills College at the time, surrounded by like-minded autodidacts who applied their considerable collective energy to solving each other's problems; his DAC arose from this remarkable pool of talent[8].

The Vim and me

I was introduced to the Kim by Rich Gold on a visit to the Bay area in 1977, and was persuaded by Rich and his friends to invest in the brand-new "Vim 1" from another 6502 microprocessor manufacturer, Synertek.[9] As the name might suggest, and the photographs prove, the Vim was the next generation Kim, souped-up with more memory and interface options[10]. Unfortunately, an on-board DAC was not included as one of those improvements, and I neither had regular access to Behrman's clever buddies, nor could afford any commercially available ready-to-run DAC boards. On the other hand, the VIM had three 6522 programmable interface chips, each of which included a very versatile shift register, so I went back to the sound-generating method I had developed in *ANDS*.

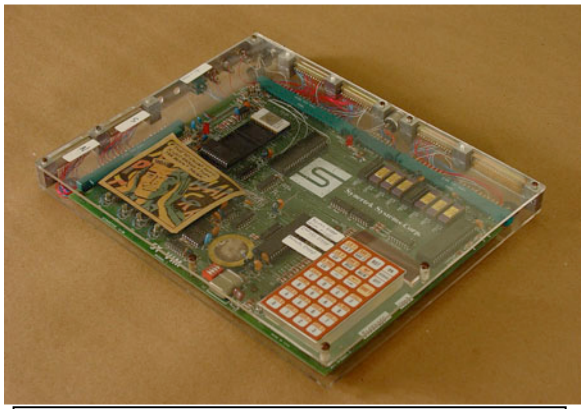

Figure 3: Synertek *VIM* microcomputer, Nicolas Collins, 1978.

Figure 4: Nicolas Collins and Susan Tallman perform *Little Spiders* at The Kitchen, NYC, 1982.

My early computer music (such as *Little Spiders*, 1981) was characterized by the ambiguous pitch center and shifting overtone structure of these serial binary waveforms, and a Wolff-like performer interaction similar to that of *ANDS*, but expanded to take advantage of the Vim's hardware and software capabilities[11]. In taking this approach I unwittingly reached back to the dawn of computer music: the designers of CSIRAC, Australia's first computer (1949) attached a speaker directly to its data bus, so that engineers could track a program's progress by ear – in the days before video screens this was a convenient way to check whether the software was running smoothly or had hung up somewhere[12]. My shift register music sounded very similar to CSIRAC's acoustic monitor (dubbed "The Hooter" by its users). The early Kim owners similarly made use of the shift-register-based interface chips like the 6522 as a cost-efficient alternative to DACs – these characteristic sounds crop up in the work of The League of Automatic Music Composers (the first networked computer band, forerunner of The Hub), most notably in the music of Jim Horton, a band member who is credited with buying the first Kim for musical use[13]. The soundtrack of many of the first computer games was generated by shift registers or direct binary pulse wave output, and the growing nostalgia for these early games has led to a revival of interest in raw computer sound: for his *1-bit Music* project, Tristan Perich has programmed simple microprocessor chips to play back full compositions using direct binary output, and packaged an edition of the circuits in CD boxes as an alternative to distributing recordings of the music[14].

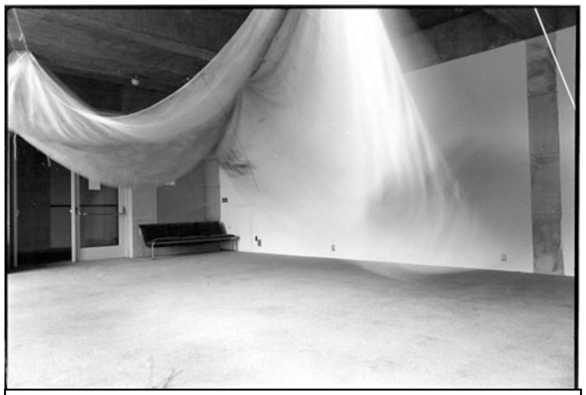

Figure 5: Installation of *Niche*, Wesleyan University, 1979.

Despite its timbrel richness there was nonetheless a certain sameness to the pulse-train output from the Vim. From the start I sought affordable ways of interfacing the computer to other devices. My longstanding interest in architectural acoustics led me to develop two rather different works. For the installation *Niche* (1979) I constructed a room size tent from old canvas sails hung from the ceiling with ropes and pulleys; the Vim controlled electric winches that raised and lowered different parts of the tent, and computer-generated sound made audible the resulting changes in room acoustics. A chance encounter with an old article on the use of mercury delay lines for computer memory (a technique employed in the 1951 UNIVAC) prompted my research into the application of architectural reverberation as "forgetful computer memory."[15] In 1979 I bought one of the first "Speak & Spells" by Texas Instruments, along with an interface that transformed the toy into a computer-controlled speech synthesizer[16]. In my pieces *Room To* Let (1979) and *The Time It Takes* (1980) I played bursts of Morse code into the room, and recorded the sound back into the computer as the reverberation died away; each iteration of the Morse was sent to the Speak & Spell, and one heard words repeat and change as the room "forgot" the decaying data.

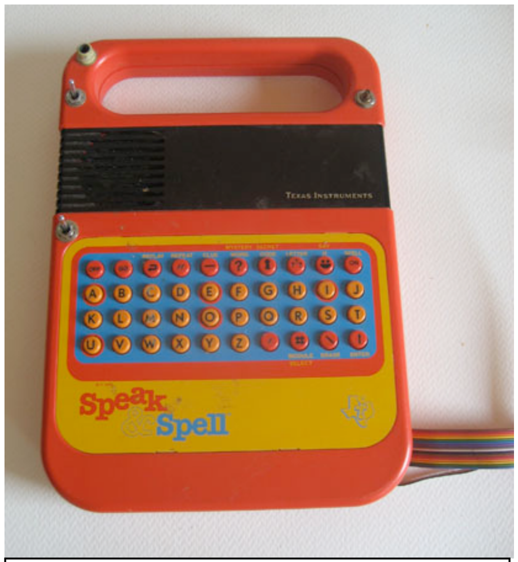

Figure 6: *Speak and Spell* modified to function as a computer speech synthesizer.

The 6522 chip that contained the shift register also included several "parallel ports": connections to the outside world that could be programmed either to output a two-state logical voltage (0 volts for "false", 5 volts for "true") or to detect such a binary signal coming from an external device. While not a substitute for a proper DAC (the port could not deal with the "in-between" voltages of an continuously changing analog signal), these ports were very useful for connecting to two-state devices, such as the switches in the keyboards used in *Little Spiders* (each key toggled between 0 and 5 volts as it closed). In 1982, inspired by the virtuosic turntable cutting of early Hip Hop DJs like Grandmaster Flash, I built an automated mixer (analog circuitry connected to the computer) that detected the beat in the sounds coming into any of the 16 inputs, and cross-faded channels whenever two of them came into sync – my goal was to emulate a Ganesh-like multi-armed DJ. In *Is She/He Really Going Out With Him/Her/Them* I ran the program while connecting to the mixer a series of different sound sources, some of them having a distinct pulse (drum machines, tape loops of snippets of Pop songs, circuits from electronic toys) and other being rhythmically free (cassettes of people speaking); the computer program generated an uncannily seamless mix from this ocean of sound.[17] I subsequently used this mixer system to help mix a record by a local NYC rock band[18], as well as the multi-track sessions from my 1985 LP, *Devil's Music*[19]. I built a variation on the mixer for Alvin Lucier for a revival of his 1965 composition, *Music For Solo Performer* (1965); the device detected patterns in his brainwave activity and automatically channeled the amplified alpha waves amongst 16 channels of percussion that were resonated by the low-frequency bursts[20]. For my backwards guitar ensemble I built a computer-controlled audio matrix switcher (1984) with which the performers could route different sound sources to resonate the strings of their instruments[21].

Although most of my early computer music was based on the kind of direct binary input and output interfaces described above, in 1981 I finally managed to build myself a 2-channel DAC. I used it to control two voltage-controlled analog bandpass filters that were modulating feedback between microphones and speakers. Because of a programming error on my part, the subroutine for one filter had an extra instruction, and therefore took ever so slightly longer to execute; as a result, what would have been a rather predictable parallel sweeping of two sets of feedback overtones devolved into a rich sheering texture of frequency modulation, distortion and beating patterns[22]. (I've never been a great programmer, but some of my mistakes have had brilliant consequences.)

Beyond the Vim

In the early 1980s I supplemented my Vim with a Rockwell AIM 65, a slightly more powerful platform for the same 6502 microprocessor. The AIM had a full ASCII keyboard and a modest alphanumeric display (in contrast to the VIM's calculator-style keypad and display), and it included a tiny on-board thermal printer that spat out my program as if it were a cash register receipt; it also came bundled with an assembler software that made programming much easier. I would develop my code on the AIM, then burn an EPROM for insertion in the more compact Vim for concert use. This working habit presaged one of the ironies of the evolution of computer technology in the 1980s: although the parade of personal computers through the decade (TRS80, Commodore 64, Atari, IBM PC, Apple II, and finally the Macintosh) provided ever-increasing power and ease of use, these advances came at the cost of portability. While enthralled by the ever-increasing capabilities of the new machines, I was loath to schlepp a video monitor and chunky computer on tour. I kept thinking back wistfully to my underpowered but Vogue-sized Vim (it was not until the Macintosh Powerbook of 2005 that I owned a computer as light and compact as the one I bought in 1978). Accordingly, I chose the slightly contrarian path of embedded microcontrollers over more general-purpose machines.

In 1986 I began developing the instrument that came to be dubbed "trombone-propelled electronics". It began with a homemade digital signal processor built up from a slightly obsolete digital reverb by Ursa Major known as the *Stargate*[23]. Pre-DSP, the *Stargate* processed digitized audio through a chain of discrete digital TTL logic chips. I embedded the motherboard from a *Commodore 64* personal computer[24] inside the *Stargate's* rack-mount chassis, pulled various key chips from the reverb, and connected the computer's parallel ports to the empty sockets. By hooking up a keyboard, monitor and disc drive I could program the *Commodore* to emulate the correct behavior of these chips (i.e., simulate reverberation), or perform my own weird variations (live sampling, looping, raspy time stretch.) The intersection of these two machines yielded unusual signal transformations. When I finished a day's programming I'd burn an EPROM to insert in the *Commodore*; disconnect the keyboard, monitor and disc drive; seal the *Stargate* chassis; and carry a relatively portable instrument to the gig.

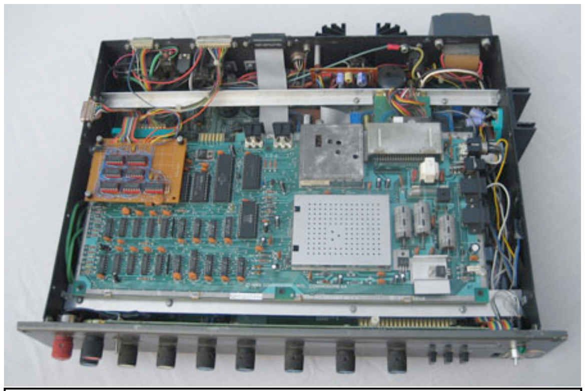
Figure 7: Ursa Major *Stargate* with additional circuitry, including Commodore 64 motherboard on top.

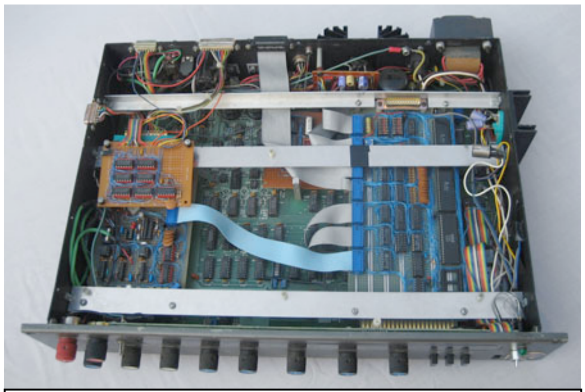
Figure 8: *Stargate* with Commodore removed, showing interface circuitry.

To control this signal processor I mounted a rotary shaft encoder (essentially half a mouse) on the back crook of the trombone and coupled it to the movement of the slide via a retractable dog leash. I attached a small keypad to the slide where it could be played by the fingers of the right hand. By pressing keys and moving the slide I could change any parameter of my program as easily as clicking and dragging icons on a computer screen – the slide became my mouse. I coupled a speaker (a high-frequency driver from a PA horn) to the mouthpiece such that the sounds of the *Stargate* could be sent through the bore of the trombone – moving the slide, manipulating a plunger mute, and aiming the instrument around the performance space added an additional lever of *acoustic* transformation to the digitally processed signals.[25]

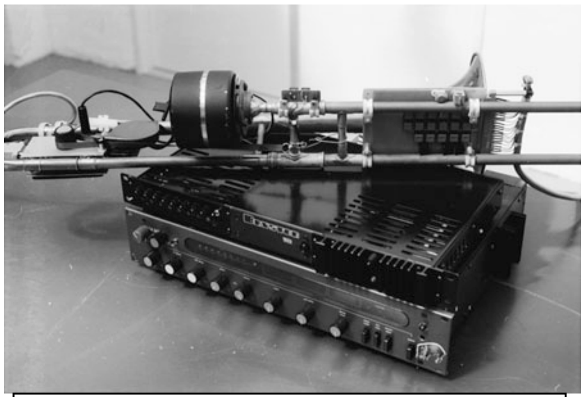
Figure 9: Trombone-propelled electronics, 1986, showing modified Ursa major *Stargate*; Bryston *2B-LP* chassis with amplifier channel in right half, computer-controlled mixer in left; trombone with leash, breath control, speaker, and keypad.

My embedding of an auto-booting computer motherboard foreshadowed the proliferation of PIC chips, Arduinos and other application-specific microcontrollers that began in the 1990s as developers of musical instruments, interactive installations, and portable computer-based products sought lightweight, cost-effective substitutes for general-purpose PCs. In the music world one of the first of these dedicated microcomputers was STEIM's "SensorLab," developed in 1992 as a musician-friendly, programmable sensor-to-MIDI converter.[26] Although many SensorLab owners used the device as a sophisticated analog-to-digital front end for their Macintosh, and did the bulk of their computer music programming in Max, the paperback-size STEIM box was a fully-programmable stand-alone computer. Programs were written in Spider (a proprietary, C-like language that STEIM developed) using a text editor on a Macintosh. The user programmed individual subroutines to be triggered by specific sensor activity -- i.e., pressing Switch 1 sends a note, twisting a pot sends Pitch Bend data, etc; Spider contained enough instructions for rather sophisticated programming. The final code was downloaded to the SensorLab, where it ran without the need of the larger external computer (much like my method of burning EPROMS for the Commodore 64 motherboard embedded in the Ursa Major).

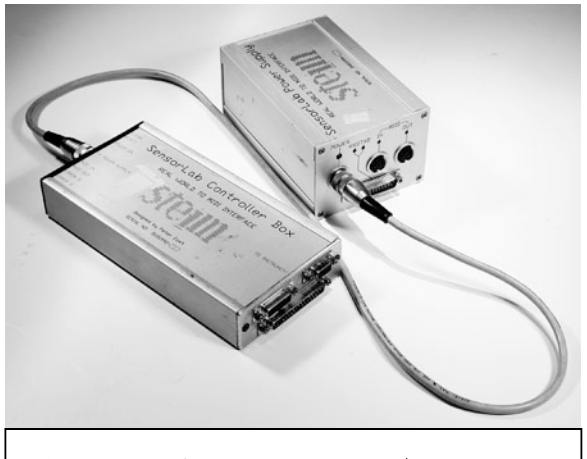
Figure 10: STEIM *SensorLab*

I became an early "power-user" of the SensorLab. I used it first in a Concertina-based MIDI controller that was intended as a six-voice extension of the trombone: magnets and Hall-effect sensors measured the flex of the six sides of the bellows; keys on each side were replaced with switches; and the SensorLab translated all this data to control a DSP system that played back through a speaker inside the instrument. Despite several years of hard work I never managed to make interesting music with this instrument. On the other hand, when my original trombone-propelled electronics were crushed beneath a Dutch taxi in 1994, I built a second system incorporating the SensorLab and a MIDI-controlled DSP, which served me well for ten years.

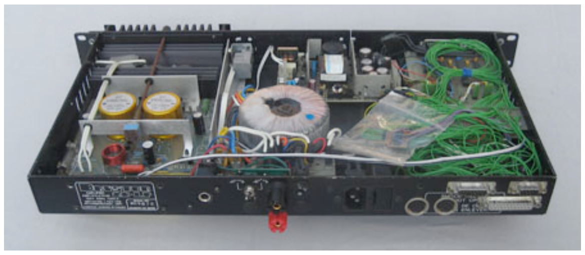

Figure 11: Trombone-propelled electronics rev. 2: open Bryston chassis (rear view) showing amplifier channel on left, SensorLab circuit boards on right.

The Laptop

As alluded to earlier in this essay, by the beginning of the millennium two curves by which computer performance can be evaluated crossed critical thresholds: CPU power and speed reached the point where affordable machines could process audio (and video, for that matter) without the need for additional specialized hardware; and this power could be shrunk into a truly portable package. The immediate and most obvious result of this shift was the emergence of "Laptronica" as a musical form that extended well beyond the pseudo-academic confines of "Computer Music."[27]

At this time I shifted my programming focus to writing code on a Macintosh Powerbook that actually ran on the Powerbook, rather than using the Mac as a development platform for embedded processors like the SensorLab. This shift paralleled a revival in my engagement with circuitry that was prompted by my students' desire for a class in "pre-digital" hardware hacking[28]. My first programs, ironically, were recreations and extensions of compositions of mine from the 1970s and 1980s that had relied on specialized electronic circuits, rather than computers – with the increasing ubiquity of common computer platforms, I began to regard software "cloning" as a way to give old-fashioned electronic music the kind of portability usually associated with scored instrumental music. I wrote software for *Pea Soup* (1974) and *Devil's Music* (1985) in Max/MSP, for distribution as stand-alone applications by downloading from my website[29].

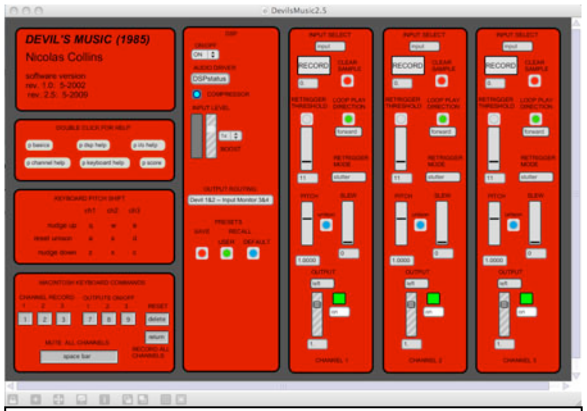

Figure 12: Screen-shot of software version of *Devil's Music*, 2009.

Subsequently I've programmed a number of pieces that could not have existed outside of the digital domain, but the overall arch of my engagement with computers has only served to reinforce the intertwined relationship of software and hardware. Computers as they are used in music are instruments, after all, and each instrument has its voice, whether it is a Kim, a Vim, a Commodore, a Mac, or a homemade assemblage of chips.

---

[1] See Nicolas Collins, "A Solder's Tale -- Putting the 'Lead' Back in Lead Users". *IEEE Pervasive Computing -- Special Issue on The Hacking Tradition: Lead Users in Pervasive Computing*, July-Sept. 2008.
[2] See Jeffrey Zygmont, *Microchip: An Idea, Its Genesis, and the Revolution It Created*. Perseus Publishing, Cambridge, MA, 2002.
[3] On Gordon Mumma, *Live-Electronic Music*. Tzadik CD, 2002.
[4] Christian Wolff, *For 1, 2 or 3 People*, 1964. On Barton Workshop, *Wolff: Works for Trombone*. EtCetera CD, 2005. In his scores from this period Wolff notated ensemble interaction using instructions similar to digital logic – I have always regarded these pieces as "proto-computer music."
[5] A shift register outputs a binary sequence of 1s and 0s, like a train of square waves with occasional gaps; by loading different numbers into the register one changes the pattern of 1s and 0s, and in turn alters the timbre of the sound, similar to selecting different waveshapes or filtering on a traditional analog synthesizer.
[6] See http://oldcomputers.net/kim1.html and http://www.commodore.ca/products/kim1/mos_kim1.htm.
[7] Behrman's first compositions for the Kim, *Figure in a Clearing* (1977) and *On the Other Ocean* (1977), are available on David Behrman, *On the Other Ocean*. Lovely Music (LP 1978, CD 1996).
[8] A good insider's history of the Bay Area computer music scene can be found at http://crossfade.walkerart.org/brownbischoff/.
[9] See http://oldcomputermuseum.com/sym_1.html.
[10] Synertek was soon forced to change the computer's name to "Sym" on the grounds of trademark violation.
[11] Nicolas Collins, *Little Spiders* (1981) on Nicolas Collins & Ron Kuivila, *Going Out With Slow Smoke*, Lovely Music LP, 1982.
[12] Paul Doornbusch, *The Music of CSIRAC*, Common Ground Publishing, Melbourne, 2005.
[13] John Bischoff, Jim Horton, Tim Perkis, Paul DeMarinis, Rich Gold and David Behrman, *The League of Automatic Music Composers 1978-1983*, New World Records CD, 2007.
[14] http://www.onebitmusic.com/
[15] T. Kite Sharpless, "Mercury Delay Lines As A Memory Unit", in *Proceedings of a Symposium on Large-Scale Calculating Machinery, 7-10 January, 1947*, pp. 103-109.
[16] The SP-1 Speak & Spell Interface kit was made by Dave Kemp at East Coast Micro Products, one of the many small businesses started in the late 1970s to serve the growing market for microcomputer peripherals.
[17] *Is She/He Really Going Out With Him/Her/Them* (1982), on *Going Out With Slow Smoke*. Two decades later I encountered a young British composer sharing my name (Nicholas Collins) who was developing software to perform similar

automatic cross cutting on digital music files; see his publication references at http://www.informatics.sussex.ac.uk/users/nc81/researchml.html and my account of our interaction at http://www.nicolascollins.com/collinscup.htm.
[18] Western Eyes, *Western Eyes*, Trace Elements Records LP, DATE????
[19] *Devil's Music*, Trace Elements Records LP, 1985.
[20] Alvin Lucier, *Music For Alpha Waves, Assorted Percussion and Automated Coded Relays*, on *Imaginary Landscapes*, Nonesuch Records, 1989. Lucier gave a few performances of *Music For Solo Performer* using this automated mixer in the early 1980s, but -- although he had anticipated a future role for a computer when he composed the piece in 1965 – he was never really comfortable with this solution to routing the alpha waves; he soon abandoned the device and chose to use a new title for the version released on the Nonesuch recording, so as not to confuse it with his original composition.
[21] See Nicolas Collins, "A Brief History of the 'Backwards Electric Guitar'" (unpublished, available at http://www.nicolascollins.com/texts/BackwardsElectricGuitar.pdf) and *A Letter From My Uncle*, on Nicolas Collins, *Let The State Make The Selection*, Lovely Music LP, 1984.
[22] *Second State* (1981), on *Going Out With Slow Smoke*.
[23] For information on the history of Ursa Major and their products see http://www.sevenwoodsaudio.com/UrsaMajor_Documents.htm
[24] For more information on the Commodore 64 see http://www.c64.com/.
[25] For more information on my trombone-propelled electronics see Nicolas Collins, "Low Brass -- The Evolution of 'Trombone-Propelled Electronics'" in *Leonardo Music Journal*, Vol. 1, 1991 (available at http://www.nicolascollins.com/texts/lowbrass.pdf); and Nicolas Collins, "Trombone-Propelled Electronics", (unpublished, available at http://www.nicolascollins.com/texts/TrombonePropelledElectronics.pdf.)
[26] See http://www.steim.org/steim/sensor.html.
[27] See, for example, the sweaty man in boxer shorts who goes by the name of Girl Talk.
[28] Nicolas Collins, *Handmade Electronic Music – The Art of Hardware Hacking*. Routledge, New York, 2009. See also http://www.nicolascollins.com/make.htm.
[29] http://www.nicolascollins.com/software.htm