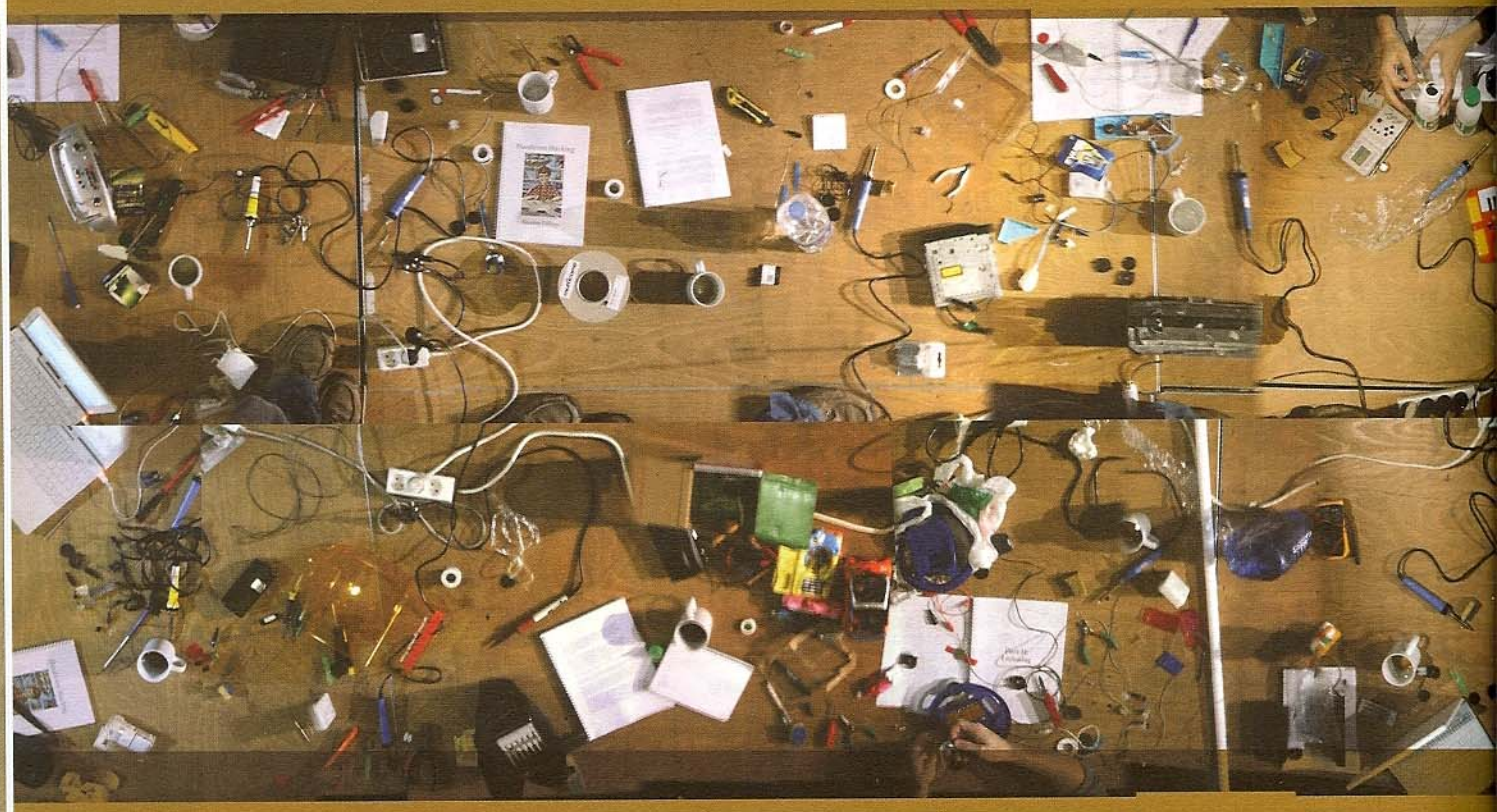
Searching for the Perfect Beep

Nicolas Collins

I built my first working circuit in 1972. I was 18 and it was an oscillator. For several months I had been messing around with a second-hand Tandberg reel-to-reel tape recorder containing a hidden, undocumented switch that, when thrown, induced delicious, semi-controllable, distinctly un-Scandinavian swoops of feedback. I was smitten by the siren call of electronic music, but unable to afford any of the instruments available at the time: synthesizers – from Moog or Arp or Buchla – were playthings of pop stars and universities.



Integrated Circuits, on the other hand - the guts of those costly machines – were getting cheaper in inverse proportion to their sophistication. New chips contained 90% of a functional circuit designed by someone who really knew what he was doing; the remaining 10% could be filled in by someone clueless – like myself. The trick was finding the right chips: in the days before the World Wide Web, information was so much more compartmentalized, with precious few leaks. When data did trickle down from the engineers to amateurs, through tech-porn magazines with titles like Popular Electronics or Wireless World, it was passed from hand to hand like samizdat literature.

My first chip was a Signetics SE/NE 566 Phase Locked Loop. Intended as the bleating soul of a Touch Tone telephone, this was an 'oscillator on a chip' – perhaps not quite so versatile as one from Robert Moog's hand, but at 5USD it was considerably cheaper. Years later I discovered that this same IC was the heart of Paul DeMarinis' first circuit, an electronic sruti box, and David Behrman's extraordinary 100-oscillator home-made synthesizer - this one chip may have been to the development of American electronic music as the Stratocaster was to the rise of rock and roll. Mine sat regally in the center of an overly large circuit board, encased in a phenomenally ugly (yet to me very professional-looking) metal box, with a crinkly matte-black finish, festooned with orange Dymo labels, officiously designating a few mismatched knobs, switches and jacks as 'pitch,' 'on,' 'output,' etc.

Ugly or not, this box not only made electronic music from the moment it was turned on, it also twisted truisms that might otherwise scare off a young experimentalist: ignorance is bliss, two wrongs can make a right, and anything worth doing is worth doing wrong – the house rules of hardware hacking, coined long before the emergence of the 'For Dummies' imprint.

The next fall I started college at Wesleyan University, where I studied composition and performance with Alvin Lucier, and picked up electronic skills any way I could: I scrutinized circuits in proper engineering journals like a first-year rabbinical student, with only a nodding acquaintance of Hebrew, gazing at the

Talmud; I stole bench space in the physics lab; I drank a lot of cheap beer with Ron Kuivila; I sat at David Behrman's feet through an invaluable guest residency; I joined David Tudor's 'Composers Inside Electronics,' the Masons of silicon; and I nagged Bob Bielecki for advice when all else failed.

My approach to design lay somewhere between the time-honored tradition of 'reverse engineering' (take something apart, copy it, make a variation, see if it still works, try another variation, etc.) and joining the simian typing pool working on Shakespeare (random component substitution.) I blew up a lot of chips, never acquired a decent grounding in general electronic theory, but became quite proficient at a few specific types of circuits that were useful to me (if no-one else) musically.

Near the end of the 1970s the first affordable microcomputers came on the market. Cajoled by the visionary Jim Horton, a handful of musicians invested in Kim-1s – a single A4-sized circuit board that resembled an autoharp with a calculator glued on top (DeMarinis intended to program his to sing 'Oh Susannah.') Programming this thing in machine language (and storing the program as fax-like tones on a finicky cassette tape recorder) was an arduous, counter intuitive, headache-inducing process, but coding offered one great advantage over building circuits: it was easier to correct a mistake by re-programming than by re-soldering. Over the next ten years Apple, Commodore, Atari and others introduced machines that whose increasing sophistication (and eventual introduction of disk drives) gradually reduced the angst-factor of programming, and home-made circuits faded into anachronism.

Like the distinguished Dr. Jekyll, or that cousin who leased out the ancestral home as the set for a porn film, I maintained a secret life of hardware even as I programmed away. I usually found that a circuit or two hanging off the computer spiced things up a bit, but I wasn't getting any younger, and I felt there was no need to call attention to my saddle shoes in an era of Doc Martins.

Cut to 2000: the millennium flops over, the computers keep running, the only sign of impending disaster is the coronation of a pretender to the American throne. I'm teaching in a computer-centric art school (The School of the Art Institute of Chicago) and I find myself repeatedly helping students seeking non-software solutions to design problems: 'How do I get the computer to tell when someone sits on the chair?' 'How do I make noises that can hang in the branches of trees?' 'How can I make a hydrophone?' Eventually I am persuaded to offer a summer class in what has by now become a forgotten black art: Hardware Hacking. It makes me feel like a living national treasure, like the oldest kimono maker in Japan, but students love it (my wife responds, witheringly, 'what do you expect when you offer a course in "Gameboy for Credit"?')

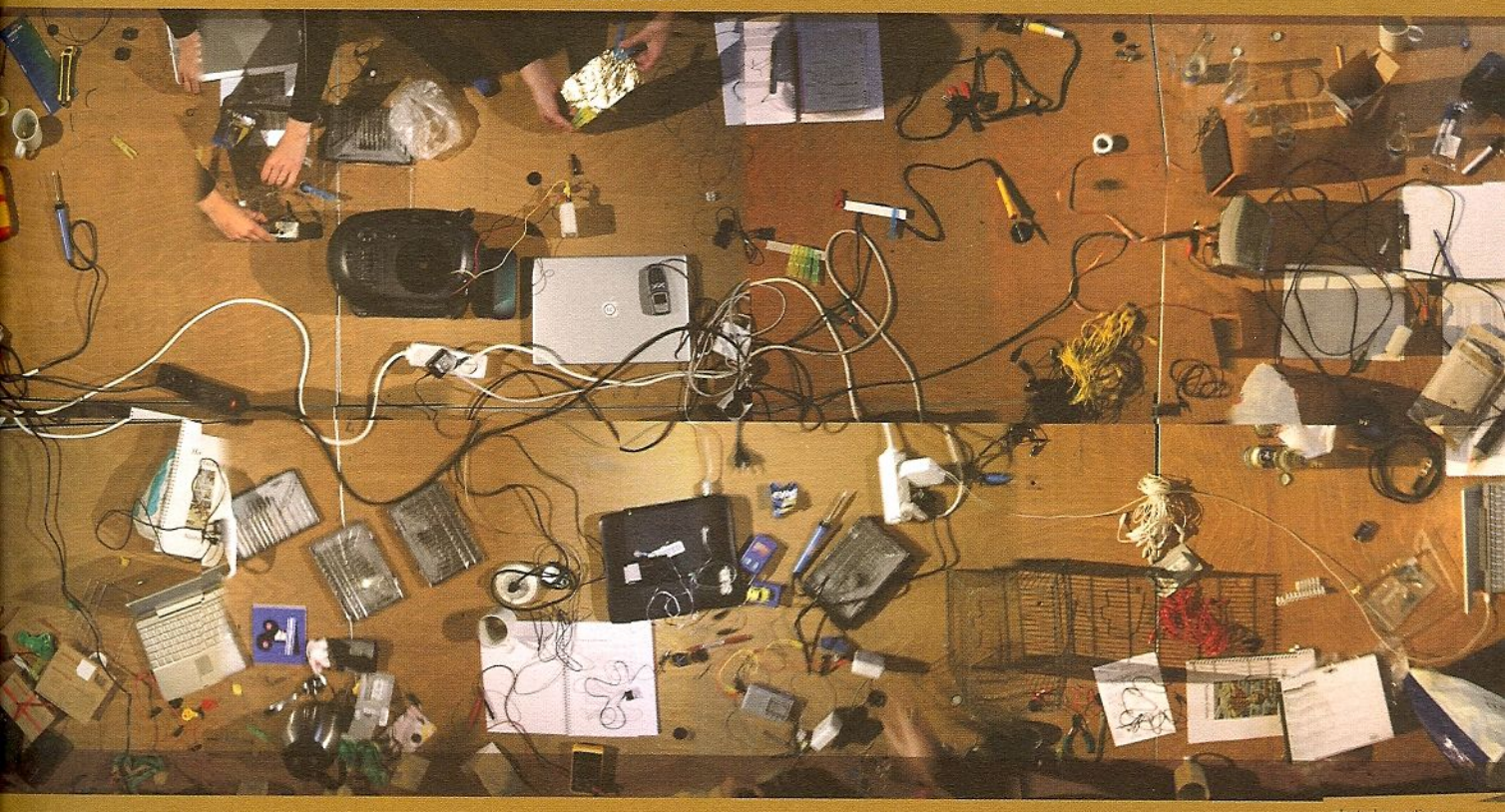


photo by *nadine*

The course begins with listening: making contact mikes and piezo drivers, experimenting with coils and tape heads, twitching speakers with batteries. We lick our fingers and lay them gently on a radio circuit board: small currents flowing through our skin create feedback paths that tip the circuit into oscillation and transform the radio into a touch-sensitive synthesizer in the style of the infamous STEIM Cracklebox. We open and re-wire toys in the tradition of Reed

Ghazala's 'circuit bending.' Digital logic chips are misused to build simple oscillators, distortion circuits, gates and panners. We listen to the video signals from cameras and video games, and hack LCD-based toys to create miniature pixel animations. We finish up, exhausted, with 'glue' circuits: simple mixers, amplifiers and power supplies that can be used to pull everything else together.

There's always this beautiful moment (usually around the time of discovering the ticklish spot that causes the radio to swoop and warble) where euphoric self-confidence sets in.

The emphasis is on instruments that are easy to build, robust, tactile, performable and permutationally rich. We make extensive use of photocells, direct contact with the circuit board, pressure pads and other intuitive, gestural interfaces. There's always this beautiful moment (usually around the time of discovering the ticklish spot that causes the radio to swoop and warble) where euphoric self-confidence sets in. Everyone leaves happy, fearless, and an obvious threat to the electronic possessions of roommates, lovers and children.

My class handouts grew into a hand-made textbook, which expanded with each offering of the course. The book escaped into the wild, and bit Phil Hallet at Sonic Arts UK, who arranged a tour of 'Hacking Workshops' in England and Northern Ireland in January, 2004. It crossed the Channel and lodged at STEIM, where I was asked – in a truly coals-to-Newcastle moment – to offer a workshop in the very birthplace of the Cracklebox. Guy van Belle showed the book to Annemie Maes, who invited me to present a workshop for x-med-k/*nadine*. The project ran October 4-9, 2004, and provided a sweaty contrast to the software workshops they had been offering previously.

I finish this essay as I sit in the legendary studios at Mills College in Oakland, CA, having just finished the most recent workshop. The subject continues to have new-found relevance to a generation of artists brought up on digital technology and software tools; here in the birthplace of microcomputer music, savvy programmers have fallen under the spell of the simplest of circuits – as they did in Brussels and Belfast, Norwich and Chicago. The handbook has evolved to the point that Routledge has asked to publish it. I hope I'm not going to wallow in nostalgia forever, but for now the beeps are sounding pretty good. ¶

photos by so-on

photos by so-on