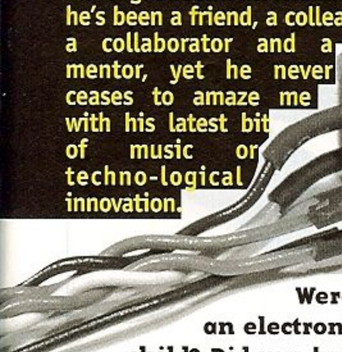
Nicolas Collins

The Unconventional Has Been Conventional For So Long

by Robert Poss

Photo by Arno Lingerak

Nicolas Collins's book, *Handmade Electronic Music: The Art Of Hardware Hacking*, is a brilliant, hands-on guide to electronic music making. I've known Nic since the mid-1970s; he's been a friend, a colleague, a collaborator and a mentor, yet he never ceases to amaze me with his latest bit of music or techno-logical innovation.

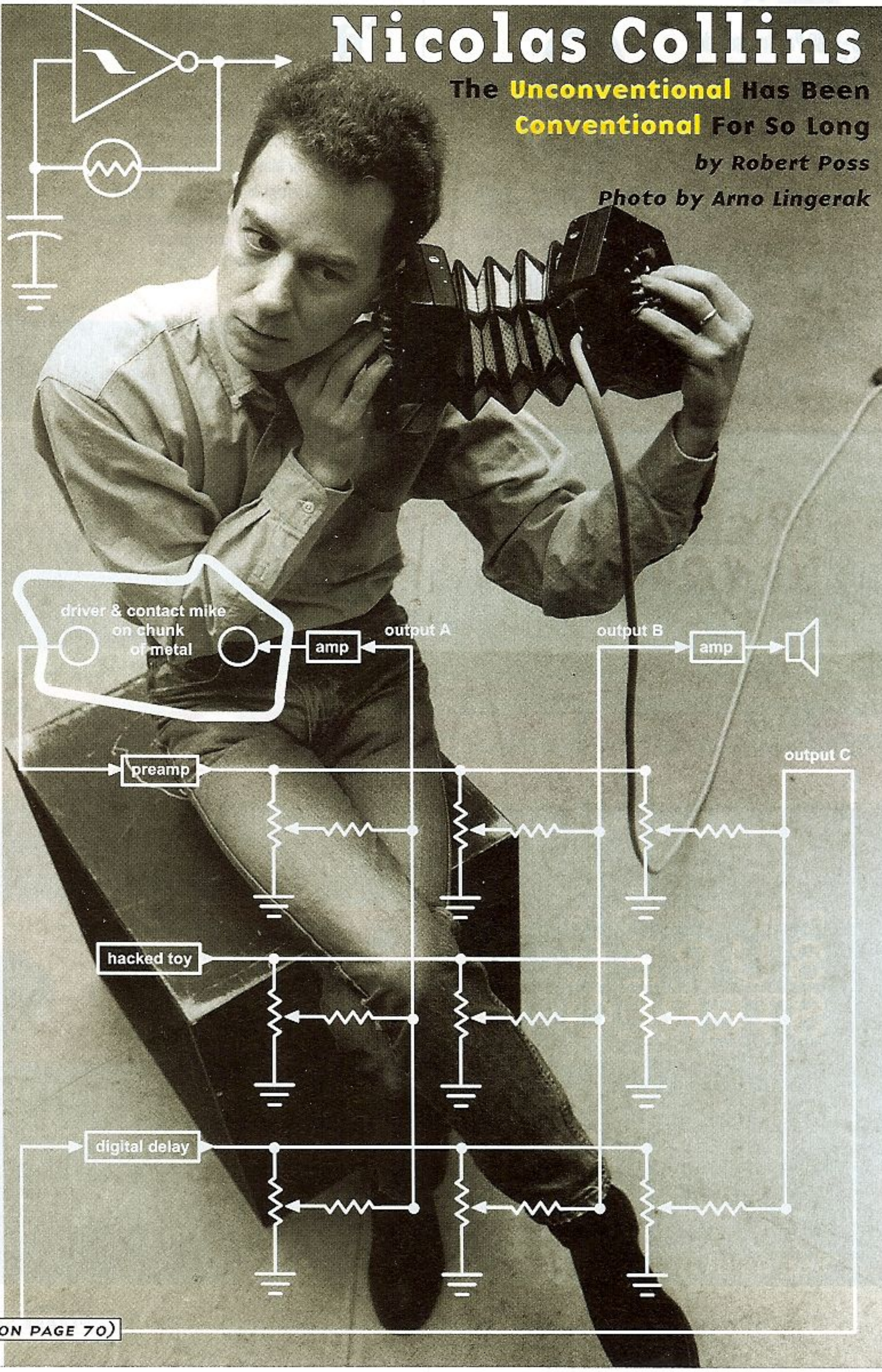


Were you an electronics hobbyist as a child? Did you build crystal radio sets and fuzz boxes and hang out at Radio Shack?

Not really. For an academic art historian, my father was quite the handyman – good carpenter, plumber, car mechanic, basic household electrical – and these were the skills he tried to pass on to his kids. He was never happier than when something broke down or got clogged up. I got a toolbox for my sixth birthday, and on Sundays we repaired appliances, built shelves, and went to the dump instead of church. I think my having learned to solder and patch together a component stereo were, for my father, the most impressive achievements of my very expensive college education. In high school I briefly tried playing electric guitar (I was terrible, even for a 13-year old), which got me interested in electronic sound. I bought an old used Tandberg reel-to-reel tape recorder and discovered a hidden switch, accessed by a screwdriver through the back panel, that when thrown set up some kind of undocumented feedback loop from output to input - adjusting the volume controls while recording manipulated weird, totally un-Scandinavian shrieking oscillation, further modulated by head echo (3 speeds!) and tape saturation limiting – very cool. I was hooked. I have a theory about the rise of American homemade electronic music: it involves the Signetics NE/SE566, a chip I became aware of in 1972 in my last year of high school.

When did you make the jump from simple analog circuits to the world of digital electronics?

Almost from the start. For a long time one of the biggest problems for non-engineers making audio circuits was power supplies – they were difficult to build, and always seemed to contribute excessive hum and other noise. Early integrated circuits [ICs] consumed a fair amount of power, only functioned within a narrow range of supply voltages, and required *two* "symmetrical" supplies. "CMOS" [Complementary Metal Oxide Semiconductor] was the first family of digital ICs designed for low power operation over a wide range of voltages – they run for days on a 9-volt battery. CMOS chips became available on the hobbyist market in the mid-1970s. A number of us sound guys figured out how to make noises with these chips, which was surprisingly easy, if never intended by the designers. In fact, when it came time to develop a course in "electronics for dummies" at my art school almost 30 years later, I went back to these mid-'70s circuit designs - they run the first time, every time - and they now form the core of my book on hardware hacking. Because they were *digital* chips, they suggested "digital ideas". My own circuitry started to go in the direction of "proto-computer instruments". I built circuits that combined sound generation with logical operations of various sorts. I was inspired by the "coordination" scores of Christian Wolff from the early 1960s – I saw his pieces like "For 1, 2, or 3 People" [1964] as presaging computer music: all those notations that instructed the players not to play notes by following a metronomic time base, but by linking their actions to those of other players as if by logic circuits. So I built instruments that could only be played by multiple players – sounds were generated when two players did the same thing at the same time - for example, pressing the same key on two separate keypads ["ANDS", 1978]. I took a course in computer programming in college in 1974 or so. I figured this would be the future of electronic music, so I should learn it, but sitting in front of a mainframe seemed miles away from the concert stage. A few years later Paul DeMarinis said I should get a KIM-1, one of the first generation of affordable, single-board microcomputers. I demurred, saying I didn't like computers. Paul replied, "Don't think of it as a computer, think of it as a big, expensive digital chip." That did it. I bought a VIM – a slightly turbocharged



(Continued on page 70)

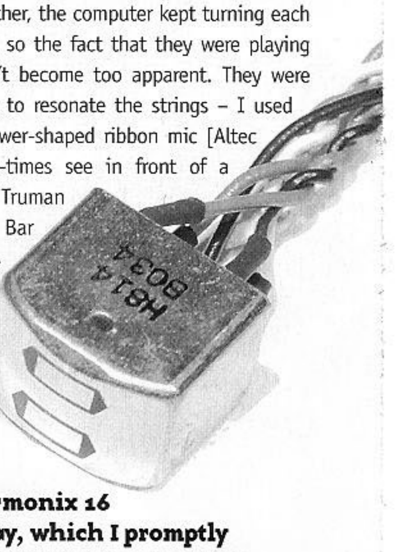
KIM: 1 MHz 8-bit CPU, 1 k RAM (expandable to 4 k!), 6-digit numeric LED display, a small calculator-style keypad for writing programs, audio jacks for connecting a cassette recorder for saving and retrieving data files using finicky fax-like tones. It really was not so different from designing circuits - programming machine code makes you think about waveforms, about voltages going on and off one at a time, hopefully in the right sequence – very different from Max, SuperCollider or Pro Tools. I couldn't figure out how to make a D/A converter, so my "computer music" consisted of direct digital output from the computer – just 1s and 0s, square waves at different frequencies. To vary the timbre I'd make trains of different patterns of ons and offs. My first computer music was as much about the hardware of the machine as it was about the software I wrote. It was pretty raw. I listen now to pieces I did then and think, "Wow, that is so much more extreme than anything I've done since" [like "Little Spiders", 1980]. Most of the composers of my generation who started soldering their own circuit instruments in the 1970s went over to programming by the end of the decade; they rode the wave from KIM to AIM to Commodore 64 to Apple II to Atari to Macintosh and never looked back at hardware. I always kept one foot in hardware and one in software. One thing I learned from David Tudor was that music seems to get more interesting when you add more connections: duos generally are more engaged than solos; a guitar sounds better though a distortion pedal and amp than it does unplugged; and lots of cheap circuits patched together with some feedback loops sound *way* more interesting than any MIDI keyboard. Unfortunately, computers tend to be permutationally impoverished in the connection department. Yeah, I know someone's going to say, "what about plug-ins?" But it's not the same without real patch cords. So after a few initial forays into pure, computer-generated sound I settled into an impractical, but richer strategy of combining programs I wrote with hardware I built or hacked. Around 1980 I got interested in early hip-hop DJs like Grandmaster Flash, and I wondered what it would sound like to have a dozen turntables cross fading on beat, instead of just two. So I built a crude automated mixer that could detect the beat in 16 channels of audio material and cut between any two channels when they came into sync on a downbeat. The performance consisted of constantly "feeding" the mixer with different audio material – tape loops, drum machines, electronic toys, radios, etc. – and letting the computer work out a perfectly seamless, if illogical, mix of these otherwise unrelated sounds – a sort of schizophrenic form of dub ["Is She/He Really Going Out With Him/Her/Them", 1982]. Later I used the system to tighten up the punch-ins on a more overtly rhythmic record of mine [*Devil's Music*, 1986] and a few tracks on an LP I produced for your band, Western Eyes [*Western Eyes*, 1984].

In the 1980s, after a number of years working in "pure" electronic music and acoustics and feedback-based music, you started tinkering with electric guitars. How did that come about, and what exactly is a "backwards electric guitar"?

Moving back to NYC in 1980 after college I was slowly drawn into a music scene that was more about *playing* than soldering or programming. I realized that guitar bands had a serious theatrical edge over knob-twiddlers, so I decided quite pointedly to make instruments and pieces that could compete for attention on the noisy New York stage. My first strategy was the "backwards electric guitar": I connected amplifier *outputs* to the pickups of some pawnshop guitars via some cheap matching transformers so that sounds plugged into the amp would resonate the guitar strings and turn the instrument into a mechanical filter – like a spring reverb, only one you could "play" by fretting chords and muffling strings - as if an EBow had an input jack. I asked some musicians to stand on stage and work their left hands while I sent a whole bunch of different sounds into the strings. It looked just enough like any other band of the time that we managed to open for Rhys Chatham at CBGBs in 1982, but it didn't really sound like one ["A Letter From My Uncle", 1984].

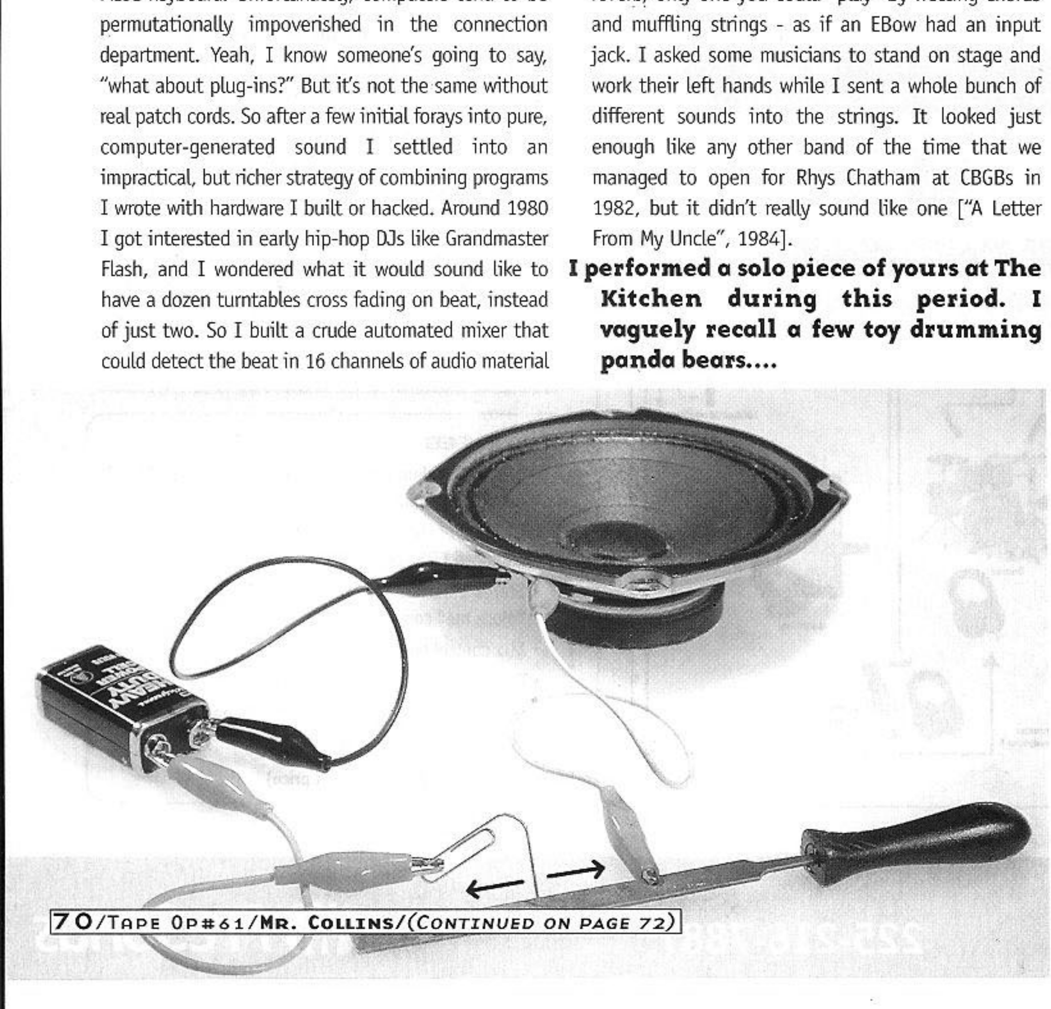
I performed a solo piece of yours at The Kitchen during this period. I vaguely recall a few toy drumming panda bears....

Yeah, you of all people should know how difficult it is to find and keep a good drummer. The six bears worked pretty well together, the computer kept turning each one on and off, so the fact that they were playing waltz time didn't become too apparent. They were mic'ed and sent to resonate the strings – I used that conning tower-shaped ribbon mic [Altec 639] you some-times see in front of a ranting Harry Truman ["Killed In A Bar When He Was Only Three", 1982].



You were the first person I knew who owned an Electro-Harmonix 16 Second Delay, which I promptly borrowed. This was in 1982 or '83. How did *Devil's Music* come about?

As music technology went digital, got cheaper and proliferated in the 1980s I realized that for a non-engineer such as myself, it was more efficient to modify commercial devices to suit one's personal needs, rather than continue to build from scratch. Hacking gained an advantage over honest design. Let me digress for a moment - from the day I started studying with Alvin Lucier in college I was paralyzed by John Cage – a nice enough guy, but he ruined my life. Under the sway of his maxim that "any sound can be a musical sound," I found it impossible to choose any *one* sound over another – they all seemed equally valid. My first way out of this conundrum was *feedback* - "the infinite amplification of silence" seemed a natural way to make sound without making any decision - plug in a mic, turn up, and let the architecture compose the music. I did a lot of feedback pieces in the 1970s ["Pea Soup", 1974-76, revived and reworked 2002] and periodically return to the stuff when I'm stuck ["Second State", 1981; "Charlotte Aux Poires", 1997; "Mortal Coil", 2002]. When Electro-Harmonix made the 16 Second Delay around 1980 - the first affordable, malleable DDL – I saw that live sampling could be another route to "making music without making a sound." I bought one of the first ones out the door of Electro-Harmonix, and made a piece called "Vaya Con Dios" [1984] using the loop as a way to accumulate, intercut and modulate samples from a Reagan speech and some old pop ballads – a ham-fisted political piece about America's clandestine war in Nicaragua. What bugged me about delays and echo, however (and still irritates me today), is their periodicity/repetitiveness – I've never been interested in a steady beat, and DDLs out of the box are way too steady. So I built a very simple 99-cent "stuttering circuit" - whenever there was a peak in the signal connected to the input of the delay it would reverse the playback direction of the loop – it was as if you had an "auto-scratch" function on a turntable that would reverse its direction on every



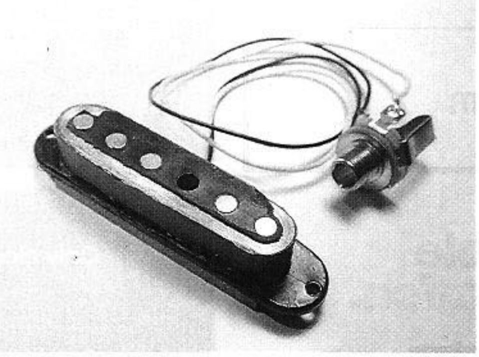
(Continued on page 72)

beat from the other turntable, or were scrubbing tape back and forth across a tape head when editing. The hack was easy - much easier than building a sampler from scratch, or writing my own sampling code for any of the personal computers of the time. I added the same circuit to a couple of "Super Replays" (Electro-Harmonix's next product, a 1-second triggerable sampler), and made a piece called "Devil's Music" [1985]. I'd grab samples of radio stations on the fly, and the circuit would retrigger the loops to create a wonderful sea of shifting rhythmic patterns that made the sample seem much more complicated (and longer) than they were. Within a few years, hip-hop's hiccoughing retriggered samples would become all too familiar, but in 1985 it was still fresh. In 2002 John Corbett, a friend of mine in Chicago, asked me to revive the piece for an all-night, multi-DJ performance at The Empty Bottle. The circuits were rusting away in an attic on Cape Cod, so I threw together a quick and dirty program in Max/MSP that emulated pretty well the original boxes - complete with stuttering circuit and slightly reduced bandwidth. Subsequently I've emailed an improved program to DJs and lap-toppers in the UK, Germany and Italy, and we've done nice long nights with multiple performers, segueing and mixing together. It's the closest I'll ever come to writing a song that can be covered creatively.

How did you come to start teaching hardware hacking and write the book?

We moved to Amsterdam and I had an invitation to be the visiting artistic director of STEIM, a foundation that does specialized music technology research for artists. For four years I coordinated research, organized the visiting artists program and programmed concerts. From there we moved to Berlin, where I had a wonderful composer's residency under the auspices of the DAAD. Money got tight and I decided maybe a job wouldn't be a bad idea, and I was lucky enough to get my first regular teaching position in the Department of Sound at The School of the Art Institute of Chicago. It's a great job – a sound department in an art school (one of the only ones in the USA) attracts a very eclectic bunch of students, much more experimental than you'll find in most music schools. When students realized that I knew something about the ancient black art of circuitry, they started asking advice, mostly about interfacing the real world to their computers. I decided we needed a course in "electronics for dummies". I start with making contact mics, using telephone taps to sniff electromagnetic fields, wiring up tape heads to "scratch" data cards. Then we open up transistor radios and emulate the infamous STEIM "Cracklebox" by licking our finger and touching the circuit board until we find feedback paths that cause it to swoop and squeal (in a most tactile, playable fashion). We do some basic "circuit bending" - hacking toys to make non-toy sounds. We build from scratch a handful of very simple, guaranteed-to-work circuits based on digital logic chips: oscillators, preamps, distortion circuits, tremolos, gates, panners, etc. We finish up with simple passive mixers and hacking sensors onto cheap game pads to make computer instruments. The emphasis is on new ways of *hearing* things (through alternatives to normal microphones, for example) and making highly *playable* gizmos – doing easily in simple circuitry what's difficult to do with computers. It was an interesting process, after almost 30 years of making my own circuits, to settle on designs that could be taught with an absolute minimum of theory, no special test equipment, could run on batteries, and that – although individually very simple — could be combined in multiple permutations to yield sophisticated, unexpected results. The class handouts eventually got spiral-bound, and this workbook escaped into the wild – like some invasive walking catfish. It got into the hands of a few producers in Europe who asked me to do workshops, and then things kind of snowballed. I've done a dozen workshops from Brussels to Beijing in the last two years, and an editor at Routledge asked me to develop a book for publication. I added an audio CD, a dozen sidebar profiles of interesting people working with homemade circuitry and a lot of nice pictures.

What are some of the most popular hacks?



That's hard to say – everyone has a favorite. Turning the radio into a synth with wet fingers is a real liberating experience for someone who has never messed with hardware – a roomful of novices with open radios has a distinctly speaking-in-[silicon]-tongues quality. Contact mics are always fun – some of the most unassuming objects (such as Slinkies) make the most extraordinary sounds - as is scratching credit cards with a tape head. Six oscillators on a single 20-cent chip make a glorious din. The workshops and the book try to provide something for everyone, and move along fast enough that people get exhausted before they get bored.

One of the things I like about your approach is that there are loads of people building their own fuzz boxes, treble boosters and tremolos, but your stuff is, even at its most conventional, unconventional.

Empowering as it is, "circuit bending", as inspired by the wonderful Reed Ghazala, has its peculiar orthodoxies. Looking at my instruments as I was setting up a demonstration at the Bent Festival at The Tank Gallery in NYC in 2004, an audience member inquired, "are they bent or hacked?" When I looked baffled he elaborated: "'Bent' means you have *no* idea what you are doing when you open up the circuit; 'hacked' means you have *some* idea." I don't care about those kinds of distinctions. I just try to find the best tool for the job, bought or made, acoustic or electronic, analog or digital, software or hardware, hacked or bent. In general, because I'm not an engineer, or a great programmer or wealthy, I look for the *simplest*, cheapest solution; but because the solution has to serve "art" rather than commerce, I also want a solution that is *rich*. I've never *had* to build "a fuzz box", so if I need to distort something I might or might not end up with a fuzz box. Throughout my workshops and the book, for example, I emphasize the importance of doing things "wrong", backwards, of trying *anything* at least once. I chose designs and components that were cheap, easy to get running, capable of operating over with a wide range of component substitutions and unlikely to burn out no matter how you hook them up. This led to some very non-standard devices and designs – there are no "op-amps" or transistors anywhere in the book, for example. I guess unconventional has been conventional in my own work for so long that I don't even think about it anymore. ✇

www.nicolascollins.com

More on Nicolas at tapeop.com/magazine/bonus.php

Illustrations by Simon Lonergan from Nicolas Collins, Handmade Electronic Music - The Art of Hardware Hacking (Routledge, 2006)

Nicolas Collins – Trombone Electronics

by Robert Poss

What exactly is/are "trombone-propelled electronics?" The backwards guitar pieces were fun to watch because of all those "horny guitars" (as my wife dubbed the highly pronged pawnshop specials that met the under-$50 Collins budget criterium.) "Devil's Music" almost rocked, and had a vaguely DJ-look that seemed to justify the low-key performance. Then in 1986 or so I started hacking an early, pre-DSP digital reverb - the Stargate from Ursa Major - to do both the live sampling I'd gotten hooked on and some very weird digital signal processing. It sounded cool, but took me back to the Tudor-era minimal knob-twiddling performance style. For reasons of theatre and pedagogy I decided I needed a BIG controller, a HUGE knob, so that the tiny gestures of nudging electronics would visually amplified. I remembered how in early science fiction movies ordinary everyday things like door knobs would be huge - like the wheels on hatches in submarines; then Star Trek substituted tiny keypads and voice response. I wanted to go back to Buck Rogers-scale. I thought, "What I need is a really big slide pot." I had an old trombone lying around the loft. I took an optical shaft encoder - essentially half a mouse, like the continuous rotation knobs used these days for data entry on synths and control surfaces - and mounted it on the back of the trombone, then ran a retractable dog leash around the encoder's knob to the slide, so that when I moved the slide the leash would pull in and out, and the knob would turn. I attached a small keypad to the slide where I could reach it with my fingers. By pressing the keys as I moved the slide I could increment and decrement two dozen values in my computer program – just like clicking and dragging with a mouse, only without having to look at a computer screen. The computer used these values to control the behavior of the signal processing: record a live sample, play it, reverse it, slow it down, add multiple echo taps, scrub through it, etc. The digitally processed sounds could then play back through the speaker coupled to the mouthpiece, so they could be further processed *acoustically* by moving the slide, using mutes, aiming the thing around the room, etc. The trombone-speaker was kind of lo-fi (it sounded like an old gramophone) but kind of charming, and it transformed the whole electronic system into *almost* an acoustic instrument (with the press of a button I could also channel the output to the PA when I wanted a beefier sound.) I had initially assumed that, as with most of my previous gizmos, I would use this thing for one or two pieces and then build up something new for the next one, but the trombone had surprising longevity – maybe because it was more like a proper instrument and less like a score embodied in silicon. Although in no way as powerful and flexible as a real trombone, it was surprisingly versatile. I used it in several composed pieces over the next five years, and it also served as my entry into the world of improvised music. Perhaps because it looked like a musical instrument I found that improvisers were happy to play with me. Self-processing had been a part of the vocabulary of amplified instruments such as electric guitars. There were some wonderful early practitioners of synthesizer improvisation - I'm thinking in particular of Richard Teitelbaum and, later, Bob Ostertag - and a handful of DJs - such as Christian Marclay - were visible in the 1980s; but I think I was one of the first on the scene to play a “signal-processing instrument” – an instrument that was mute until someone else started playing and I could sample and process them. The novelty must have been appealing: I played hundreds of concerts with musicians like Tom Cora, Peter Cusack, Shelley Hirsch, George Lewis, Ben Neill, Zeena Parkins, Elliott Sharp, Christian Marclay, and John Zorn, and in 1989 did a CD of 42 short improvised duets with 15 of them [*100 Of The World’s Most Beautiful Melodies*]. Because a Commodore 64 microcomputer was embedded in the Ursa Major chassis to translate button pressing and slide movement into signal processing parameters, I could hook up a monitor and disk drive whenever I wanted to tweak or expand the program. But it was tedious work. I was starting to feel that the instrument had reached the limits of its evolution when it was run over by a taxi at Schipol Airport in Amsterdam in 1994. I patched it together and nursed it along for another month before it finally expired, rather dramatically, during a live radio performance, while I built up a new instrument based on the STEIM SensorLab (a small computer that translates sensor data into MIDI) and a MIDI-controlled effects unit. As with the original version, this one suited me quite well until I reached the limits of its DSP vocabulary and the bandwidth constraints of MIDI, around 2001. By then I was happy to retire both the instrument and the whole concept of “live sampling and signal processing” (which had by this time become quite commonplace) but a few years later I dusted it off for a last performance and realized that I still enjoyed the basic premise of an alternate controller with an acoustic voice. So I started working on Rev. 3.0 and got it to the point of playing out (still held together with rubber bands) in the spring of 2005. It uses a clever tiny box by Sukandar Kartadinata [the Gluion, www.glui.de] to gather sensor data from the instrument, but all the signal processing is now done in easy-to-program-and-update Max/MSP – which I hope will give this instrument a longer shelf life before obsolescence than its predecessors, and expand the signal processing palette beyond what I had in the past, and what’s available commercially.

At some point in the late 1980’s I recall you doing unspeakable things to portable CD players. You were certainly the first composer to work with the skipping CD, though many others followed in your footsteps. What did you do?

As I said I’ve been interested in DJ culture since the early 1980s. I really admired the way this affordable, low-tech sampling instrument was so quickly incorporated into both pop and experimental music. I worked with a few DJs - primarily Christian Marclay - and thought about picking it up myself, but was put off by the sheer weight of the gear. I was forever trying to reduce the amount of junk I toured with, and adding turntables and vinyl seemed a step backwards. So when CDs emerged in the mid-1980s I thought, “Hmm, CDs and Discmen are lighter, and I bet it’s possible to hack the player.” Yasunao Tone had just started “wounding” CDs with Scotch tape to make his fabulous, edgy, proto-Glitch music [1985], but I was interested in modifying the *player* to make a performable, turntable-like instrument, rather than altering the CDs themselves. I called up a Sony parts distributor in the USA and sweet-talked a service manual out of them for the cheapest Discman I could find. On the largely incomprehensible schematic I found a signal designated “mute”. I figured on a trombone a mute is an expressive tool, but in audio mute means “off” – what did they want turn off, and why? I followed the signal upstream to the control chip and ripped off that leg. Now the CD never muted; it “scratched” the laser across the CD, like flicking a tonearm. Pausing the player produced a beautiful, off-kilter skipping rhythm – not as regular as a skipping record, it had a sort of swing to it. I went through my modest CD collection, scrounged cut-out bins and discovered that I really liked the effect of the hack on late Renaissance and early Baroque music: the timbre of “original instruments” with the hard-edge glitch of the digital click (I called it “digital claves”) superimposed. I wrote a bunch of pieces based on this sound and using the hacked CD player as an instrument. In “Broken Light” [1991], musicians in the Soldier String Quartet used foot pedals to scratch through a CD of Italian Baroque string music, suspending fragments in unstable pause-loops, while playing against the resulting pulsing chords. The control interface was pretty crude: they couldn’t control *exactly* where the CD would loop, only get within a second or two, so there was a quasi-indeterminate aspect to the unfolding harmonies that made it more challenging (and rewarding) than playing against a backing tape of similar material. I did a number of similar pieces for instrumentalists and hacked CD players, then finally wrote a CD-free piece for a Berlin chamber ensemble that mimicked the artifacts of the skipping CDs purely acoustically. [“Still (After) Lives”, 1997, based on “Still Lives”, 1993] By the late 1990s the sounds of malfunctioning CD players were pretty familiar -- from Oval and Microstoria, through Micro-House and Glitch, to Madonna of all people, that off-kilter rhythm had insinuated itself into mainstream pop – but there was a time when it the hacked CD made fresh sounds and provided a nice structure for recycling sound.