A Microcomputer-controlled Synthesis System for Live Performance

Martin Bartlett
Department of Music
University of Victoria
Victoria, B.C., Canada

In my experience with electronic music, the studio environment has always presented a major problem. Music is a social phenomenon; the musical act reaches fruition in performance. To work in the studio producing works which have their existence only as recordings seems artificial to me. Yet the cost and technical sophistication of electronic music facilities, particularly where computation is involved, seems to imply that "Mohammed comes to the mountain," and this approach has been further emphasized by the development of general-purpose computer music systems which are as far as possible without personality of their own and can supposedly serve the needs of a variety of composers. Furthermore, even in those institutions which can afford such facilities it may be impossible to have a system dedicated totally to musical tasks, presenting additional problems of access.

Martin Bartlett (left), a moment of illumination.
(photo credit: Joel Chadabe)

Now I have nothing against the existence of sophisticated centralized facilities, and am delighted with the rare opportunities I have to work with them, but I believe that if the potential of computer music is to be realized, they should be complemented by small personalized systems capable of being used in live performance. My own approach to the implementation of such a system is the concern of this paper.

The "Black Box"

In 1969 I began building a personal analog synthesis system. This was the Black Box which has gone through many metamorphoses in its working life, and is still in use. It is only recently, however, that it has seemed feasible to extend the controlling means of this instrument with a microcomputer.

The Black Box is a collection of fairly conventional analog modules. Many of the circuits were designed by Berkeley engineer William Hearn, with whom I collaborated from 1969 to 1972, though I am responsible for the overall configuration of modules, the construction, and various designed or undesigned "quirks." The basic premise was a low-cost live-performance voltage-controlled synthesizer capable of complex timbral textures and spatio-location in four channels. A large number of modules were built; the instrument has ten voltage-controlled oscillators, four voltage-controlled filters (two of them doubling as quadrature oscillators), eight envelope generators, twenty voltage-controlled amplifiers in a four-channel distribution matrix configuration, two four-channel sequencers, in addition to noise generators, envelope followers, multipliers, and phase-locked loops. Absolute linearity and predictability of operation was in some cases sacrificed to module density, giving this instrument its own personality and sound, and condemning me to eternal bondage to the soldering iron. There is no keyboard, the only tactile input being potentiometers and switches. The performance philosophy is to realize by means of small adjustments the acoustic potential of a given patch in the way that an Indian musician might exploit the pitch and duration associations of a particular *raga* or *tala*. The instrument is in a continuous state of change, with modules being retired and new ones constructed as my musical ideas and performance aims change.

With the absence of tactile input to which I have referred, the generation of complex event sequences has always been a problem, and has been attained on occasion by miraculous combinations of patching and parameter setting which seem almost beyond my control. The computer offered the possibility of transcending this problem, though the actual implementation of a practical system seemed to present other problems even more daunting. In particular, many of the available low-cost microcomputer systems seemed to suffer from "peripheralitis"—in other words it was obvious that I could not achieve a practical working system without the additional expense of memory boards, interfaces, terminals, and so on. The decision I ultimately made, at the time, incidentally, of the last computer music conference (2nd International Computer Music Conference, USCD, La Jolla, October, 1977), was to start with an absolute bare-bones, minimum system, and allow expansion to be governed by hands-on experience. This approach led me to choose the MOS Technology KIM-1, my choice being influenced by the work of several composers associated with the Contemporary Music Center at Mills College, in particular James Horton, Paul DeMarinis, David Behrman, and Rich Gold, all of whom have managed to make interesting and individual use of this particular microcomputer system.

So beginning in October 1977, working largely unaided except for what I could glean from the literature, I set about implementing the KIM-1 as a control-voltage generator.

I will not dwell on the first two months of this process, spent largely in gazing blankly at the programming manual awaiting illumination. (I had no previous programming experience.) On December 19, 1977, however, at 2:34 a.m., PST, the penny dropped, and what had been obscure was now revealed: I understood the essentials of writing and debugging 6502 machine code.

Interface

Having at least a working understanding of the basics of programming the KIM-1, the next step was the construction of a D/A. Early experiments were conducted with a very simple single-channel D/A designed by Hal Chamberlin, and described in *BYTE*, September 1977¹. With this interface working, I began writing control programs based on stochastic algorithms, primarily for oscillator control and generation of timing functions. Since it was obviously important to have as many output channels as possible, and in addition to have the capability of analog input to the computer, I looked for a suitable design for a multi-channel interface. In January 1978 I built an eight-channel multiplexed A/D and D/A to the design of Douglas Kraul, also, described in *BYTE*, June 1977². This interface uses a single inexpensive D/A for both A/D and D/A modes, with 8-bit resolution and a maximum bandwidth of approximately 100 Hz per channel, which, though modest in comparison with more elaborate installations, compares favourably with the sequencers available with most analog synthesizers.

The KIM-1 seems to me an elegant and versatile little machine. In addition to the 6502 CPU, the system includes two 6530 arrays providing 2K bytes of ROM for the system monitor, a hexadecimal keyboard, TTY interface, LED display, cassette interface, and programmable timer. In addition, the board accommodates 1K bytes of RAM, to which, until recently, all my programs have been restricted.

As well as the 1K of on-board RAM, each 6530 contains an additional 64 bytes of RAM, and it is in this memory space that the program resides to refresh the D/A sample-and-holds and enable the A/D or D/A modes of the interface, which it does on an interrupt basis deriving timings from the programmable timer. Some trial and error experimentation was needed to establish the optimum timing for this interrupt, especially since the programmable timer is used also as a system clock for timing output voltages. A system refresh and timing-pulse of 200 Hz was ultimately decided upon, although part of the control program provides for altering this frequency as a function of an analog input.

Programming

The programs written for the system so far are for the generation of control voltages, mainly for oscillators, and trigger-pulses to fire the synthesizer's envelope generators. Since the timing-pulses are generally made to coincide with the generation of a new output voltage (the combination is called a "note" in musical terminology), I found it convenient to link these functions, so that four of the D/A channels output voltages, and the other four output the trigger-pulses. This gives, therefore, four independent control channels.

The timing process must allow independent timings for each channel. The algorithm to accomplish this is diagrammed in the flow chart of Fig. 1. Decrementing the channel timing counters is done as part of the same interrupt routine which updates the D/A's, so one's control over the output timing can be implemented in two ways: by programming the 6530 interval timer as part of the program or from an A/D input; or by choice of numbers loaded into the duration tables.

Frequency and Duration Tables

Both analog output levels ("frequencies") and durations are chosen from tables. Parameters may be read from these tables either sequentially, randomly, or by means of a number of "random walk" algorithms to be described later. Each output may choose data from its own separate table, or one table may be shared by two or more outputs. It is also provided that the choice of the table in use by a given output may be a function of an input voltage from one of the A/D channels.

Analog output levels are specified by numbers between 00 and FF(Hex). The D/A is adjusted so that 30(Hex)=1 volt=1 octave on the exponential-input oscillators. This gives frequency control of 48 bits per octave, or 1 bit= $\frac{1}{4}$ semitone, for a full range of a little over 5 octaves. This can hardly be called a high degree of resolution, but has proven adequate so far, especially since, as hinted above, the oscillators are not models of precision themselves.

As already outlined, the duration tables contain hexadecimal values which effectively act as dividers of the programmable timer frequency. The most usual means of accessing the duration data so far has been by random choice of values from the table. Probability distributions are governed by the table length and weighted by duplication of data values in the table: a typical situation might be a table of 32 data elements, containing only four different values, distributed, say, 1, 3, 8, and 20.

Figure 1

Random walks

One of the first "automatic melody" algorithms written for this system was the random walk diagrammed in the flowchart of Fig. 2. Each time the duration counter for a given channel reaches zero a subroutine (NEWOUT2) is called which begins by choosing a random number between 0 and 3. If the result is zero, the new output value is the same as the previous one (a "repeated note"). If it is one, the frequency table index pointer is incremented by one (the next "higher" note); if two, it is decremented by one (the next "lower"); and if three, a new pointer value is chosen randomly. Table values comprising different pitch sets, or tables arranged with irregular increments naturally yield widely different musical results.

Figure 2

A recent article on stochastic music in *Scientific American*³ interested me in the possibility of writing a random walk program based on $1/f$ distribution. Whether my algorithm generates a statistically correct $1/f$ sequence or not I cannot say, but the results are musically interesting, having that peculiar combination of volition and purposelessness which seems so characteristic of computers when they are feeling playful. This subroutine (NEWOUT3) is diagrammed in the flowchart of Fig. 3. A counter is incremented each time the subroutine is called. The lowest three bits of the counter value are tested, and, depending on which bits are set, one, two, or three new random numbers between 0 and 7 are stored in three memory locations. The contents of these locations are summed and the result is the frequency table pointer value.

Figure 3

A/D Input

The system's eight A/D inputs are provided with voltage-divider potentiometers for manual control, and jacks for control-voltage input from the synthesizer. The limited frequency response of the interface makes it impractical to use these inputs for audio-frequency signals. Nevertheless, there is potential for access to program parameters through these inputs.

As with the D/A's the inputs are adjusted so that five volts in gives the full-scale value of 255 (FF Hex). One of the A/D inputs is usually used to control the frequency of the programmable timer, giving overall system timing control. The range of control is adjusted so that even at the slowest setting there is no perceptible droop in the output from the sample-and-holds, while the fastest setting still allows sufficient time for interrupt servicing and program execution. In this, as in other timing problems, I have used a trial and error approach.

At present, the A/D inputs are used in the following ways: first, to determine which of a number of output-voltage selection algorithms are in use (i.e., should the table pointer step sequentially through the table? Move randomly? Or should it employ one of the algorithms already described?)

Second, A/D inputs are used to choose which of a number of data tables is being addressed by the pointer. These include tables containing different pitch sets ("modes") including microtonal ones, and other tables loaded by random operations. Third, some of these A/D inputs determine choices between a number of timing possibilities: choosing durations from a list, regular time intervals, and so on, including the option of whether or not trigger pulses will coincide with output voltage changes ("notes") or operate on a slower time scale ("phrases"). Fourth, one of the A/D inputs controls the frequency of the programmable timer which governs the data acquisition rate and also counts down the individual clocks for the output channels, giving overall system timing control. All these inputs may be addressed either from the pots or the analog inputs, giving the possibility of structural variation by feedback from synthesizer control voltages. It is these A/D inputs which make possible rapid and radical changes in control voltage structures and hence in the musical output.

Musical Uses

Since February 1978 I have been using this system in live performance—solo, as part of an electronic-instrument improvisation ensemble—the Networks Orchestra, and more recently in a composed work for eight instruments and live electronics.

It always functions very much as an addition to the capabilities of the analog synthesizer, at which timbral and dynamic control as well as structural decisions are exercised. As such I find it a very useful extension of the Black Box's capabilities, and considerably more versatile than an analog sequencer.

In *Air*, my recent composition for eight instruments and live electronics, the KIM-1 Black Box has a rather satirical role, as a "pseudo-instrument." Since the same algorithms determined many of the pitch-time decisions in the instrumental parts as are used by the computer, computer and instruments engage in a kind of mutual parody.

Further Developments

The system has recently been expanded beyond its original minimal configuration by the addition of an 8K memory board with two extra output ports, sockets for four 2716 EPROM's and an EPROM programmer. This permits assembly-language programming and an editor, which after a year of loading hexadecimal machine code on the KIM's calculator-type keypad, seems a great luxury. Other envisioned improvements are a faster data-acquisition system, and, of course, more extensive and ingenious programming.

As I get to know the capabilities of this system, I am impressed at the complexity and diversity of the music I can make with it. The countless hours of fumbling, debugging hardware and software, have been rewarded by the partial realization of a long term goal: to perform a lively and intricate music full of timbral, tonal, temporal and spatial variety, and to do so with my own resources wherever I may happen to be. The portability, personality and minimum hardware cost of the system are persuasive advantages; its limitations are readily apparent and are accepted as part of that personality. As always, the musical results are the only real test of a system's validity.

References

1. Chamberlin, H., "A Sampling of Techniques for the Computer Performance of Music," *BYTE*, Vol. 2, No. 9, September 1977.
2. Kraul, D., "Designing Multichannel Analog Interfaces," *BYTE*, Vol. 2, No. 6, June 1977.
3. Gardner, M., "Mathematical Games: White and Brown Music, Fractal Curves and One-over-f Fluctuations," *Scientific American*, April 1977.

The microcomputer-controlled synthesis system for live performance. In the foreground, the KIM-1 is shown; in the background, the Black Box analog synthesizer.

(photo credit: Joel Chadabe)