F. Richard Moore
Department of Music, and Center for Research in Computing and the Arts
University of California, San Diego
La Jolla, California 92093–0037, USA
frm@ucsd.edu

Dreams of Computer Music—Then and Now

Computer music was first created nearly 40 years ago in universities and research laboratories by people dreaming of magical instruments for music. These instruments were like genies in that they could grant wishes—musical wishes—to anyone who possessed the required "bottle" of knowledge about computers and music. These dreams also had a corresponding technology in the real world, but more than anyone would have dared to wish at the beginning, the technology of computer music was to prove itself unique; computers are possibly the only thing in existence that have actually gotten cheaper over the years while simultaneously becoming more powerful. Now nearly everyone who wishes it has the means to create computer music—but do they have the dreams?

Precursors, Science Fiction, and Music

Dream: Power Over Sound and Music
Reality: Purely Imaginary

As strange as it may sound, the idea that observation should be the basis of knowledge was not always commonplace. The great philosopher and statesman Francis Bacon gave much impetus to the development of modern science by suggesting that observation of nature should be the basis for knowledge. At the time of Francis Bacon, the church, the state, and the nobility all had another theory; they believed that authoritative statements by high-ranking officials should be the basis for knowledge and truth, rather than observations that could be made by just about anybody. The real revolution of science was the replacement of authority with observation as the basis for truth. In fact, the very term *revolution* comes from Nicolaus Copernicus's observations, first published in 1514, that led him to suggest that the earth revolves around the sun rather than vice versa, an observation that directly contradicted the teachings of the Roman Catholic church. In 1633, just a few years after Bacon's death, the church condemned Galileo Galilei to life in prison on a charge of "vehement suspicion of heresy" for teaching Copernicanism. Francis Bacon's philosophical works were held in high esteem by many important scientists, including Robert Boyle, Robert Hooke, Isaac Newton, and Thomas Hobbes, so much so that in the 18th century, Voltaire and Denis Diderot considered Francis Bacon to be the father of modern science.

In 1626, about two years before his death, Francis Bacon wrote *The New Atlantis*. This is, ostensibly, the first work of what we today call science fiction. Bacon was both a practical man and a dreamer. In *The New Atlantis*, he described a society stumbled upon by shipwrecked sailors, a kind of island utopia somewhere in the Atlantic Ocean. As the sailors were shown around the wonders of this place, Bacon describes many aspects of nature being studied. One of these places of study was described with the following words.

> We have also our sound-houses, where we practice and demonstrate all sounds, and their generation. We have harmonies which you do not, of quarter-sounds, and lesser slides of sounds. Diverse instruments of music, likewise to you unknown, some sweeter than any you have; together with bells and rings that are dainty and sweet. We represent small sounds as great and deep; likewise great sounds extenuate and sharp; we make diverse tremblings and warblings of sounds, which in their original are entire. We represent and imitate all articulate sounds and letters, and the voices and notes of beasts and birds. We have certain helps which set to the ear do further the hearing greatly. We have also diverse strange and artificial echoes,

Computer Music Journal, 20:1, pp. 25–41 Spring 1996

> reflecting the voice many times, and as it were tossing it: and some that give back the voice louder than it came; some shriller, and some deeper; yea, some rendering the voice differing in the letters or articulate sound from that they receive. We have also means to convey sounds in trunks and pipes, in strange lines and distances.

In this small paragraph from *The New Atlantis*, Francis Bacon described much of the modern field of computer music almost perfectly. In 1624, Bacon was already dreaming of fundamental power over sound and music. What is remarkable is not so much the dream itself, for Bacon was simply trying to list everything that he could imagine doing with sound, but that it would take hundreds of years to achieve most of it. Alexander Graham Bell invented the "trunks and pipes" and "strange lines" to convey sounds over distances—otherwise known as the telephone—only in the 19th century. Hearing aids, which Bacon described as "certain helps which set to the ear do further the hearing greatly" are still a matter of intensive research, though much has been achieved recently through the use of neural implants and microelectronics. Modern recording studios would be lost without their "diverse strange and artificial echoes," and computer music techniques like the phase vocoder allow us to "represent small sounds as great and deep; likewise great sounds extenuate and sharp," among other things. The important thing is that Bacon imagined—he dreamed—of being able to "practice and demonstrate all sounds, and their generation." As with Leonardo da Vinci's visions of flight through the air, and Jules Verne's visions of travel through space, without dreams, reality could not possibly follow.

Dreams in this sense are a pure combination of desire and imagination without necessity of the means to achieve something in reality. Dreaming is imagining what is desirable without having to worry about how to achieve it. The purpose of dreaming is therefore to discover what we wish to do. Only when we awaken do dreams become attached to the typical clauses and conditions starting with "if only...."

I wish now to consider the development of the field of computer music in terms of the dreams of several of those who were active in its establishment. While Francis Bacon was dreaming in 1624 of doing what many now take for granted, I consider the real history of a separate field called computer music to have started in the mid-1950s with the pioneering work of Lejaren Hiller and Max Mathews. Obviously such a treatment as this must be somewhat speculative, for most of what remains is the work of such people rather than records of their dreams. But there is ample evidence that dream they did, and I suggest that it is these dreams—every bit as much as the work—that form the basis for what is now an established and increasingly important field of inquiry.

Musique Concrète and *Elektronische Musik*

There were many other precursors to computer music. Thaddeus Cahill's 200-ton Dynamophone (also called the Telharmonium), invented around 1906, toured the United States in several railroad boxcars, astonishing and delighting audiences with its amazing ability to create musical sounds through the use of huge electromagnetic inductors. Leon Theremin invented his famous mood-altering instrument about 1924, around the time of the Ondes Martenot used by Olivier Messian and the Trautonium used by Paul Hindemith. Sometime in the 1930s the great conductor Leopold Stokowski wrote, "I can see coming ahead a time when the musician who is creator can create directly in tone not on paper." In the late 1940s, with the advance of magnetic wire and tape, it became possible to manipulate sounds on recordings in a manner similar to the manipulation of images on photographic paper. Such manipulations gave rise to the school of *musique concrète* by the Groupe de Recherches Musicales at the national radio station of France in the early 1950s, with the work of Pierre Shaeffer and Pierre Henry and others. *Musique concrète* was to be based on the "concrete" sounds of nature rather than the "abstract" sounds of traditional musical instruments.

Also in the early 1950s, at the Cologne Radio Sta-

tion in Germany, Herbert Eimert and Robert Beyer began to generate sound waves from their elemental building blocks—sine waves—according to the famous mathematical theory of Jean-Baptiste Joseph Fourier. Because the sine waves were created electronically, the result, as practiced by composer Karlheinz Stockhausen and many others, was collectively known as *elektronische Musik*. In the United States, Vladimir Ussachevsky and Otto Leuning produced the first concert of what they called "tape music" at the Museum of Modern Art in New York City on 28 October 1952.

Computer Music: The First Decade, 1955–1965

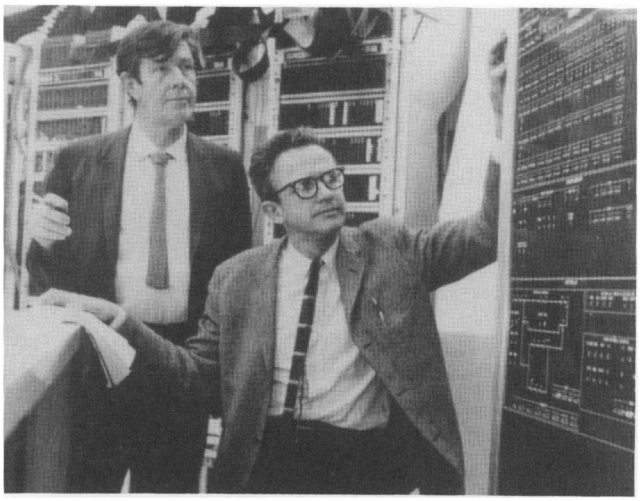
Dream: Mind of Science/Heart of Art
Reality: Existence Proofs

Hiller and the ILLIAC

True computer music did not begin, however, until a young research chemist and composer named Lejaren A. Hiller began to investigate ways to apply a process known as stochastic modeling to understanding the process of musical composition. Stochastic methods had been successfully used for molecular modeling in chemistry for some time, and computers were a perfect vehicle to exploit such techniques. Computers of the mid-1950s were not yet capable of sensing and producing arbitrary sounds. They were, however, capable of the high-speed calculations that made otherwise laborious stochastic modeling a practical reality. Because he worked at the University of Illinois, Lejaren Hiller had access to one of the earliest examples of what we would now call a supercomputer: the ILLIAC. He is shown in Figure 1 standing at the console of the ILLIAC with his collaborator John Cage.

Lejaren Hiller's basic idea was simple: have the computer generate a list of random numbers representing anything we like about music, such as pitches, rhythms, dynamics, etc., then program a set of "rules" by which one choice would be allowed to follow another. For example, if we wish to compose a melody, we might number the pitches on a piano keyboard from 1 to 88, then (arbitrarily)

Figure 1. John Cage and Lejaren Hiller at the console of the ILLIAC computer they used in their collaborative piece HPSCHD, realized at the University of Illinois in Urbana in 1968. (This photograph originally appeared in Source: Music of the Avant-Garde *2(2), 1968, and is reprinted with permission of the Source Archives, Music Library, University of North Texas, Denton, Texas, USA.)*



set the first note to number 40 (middle C). Given that first choice, the rules might disallow any skip greater than an octave, so if the computer chooses a second note at random less than 28 or greater than 52, the choice would be rejected and the computer must try another, and another, until a choice is found that doesn't break any of the "rules." Such a technique is called a "Monte Carlo method." The hard part of this procedure is coming up with rules that are both possible to obey and that give results that sound musical. This is much more difficult than it may appear at first glance; so difficult, in fact, that Hiller tried several alternatives to the whole procedure, most notably those based on the mathematics of Andrei Andreevich Markov. Using Markov's techniques, one examines any set of existing music, perhaps that of a composer generally regarded to be reasonably good, such as Mozart. The basic idea of Markov modeling is as follows. Looking through one or more pieces by Mozart, one can compute the probability that the note middle C will be followed by D, or E, or E-sharp, or F-sharp, etc. Once these probabilities are determined by analyzing existing music, the computer can be programmed to produce music that has exactly the same statistics, yet is different from the analyzed music. If the statistics are based on Mozart, the computer would be more likely to follow middle C with D than, say, F-sharp. Such techniques can be extended to pairs of notes, or triplets

of notes, and so on, yielding ever more restrictive probabilities, while still allowing some freedom of choice. Eventually one can imagine doing such things as feeding in Beethoven's nine symphonies and asking the computer to compose the Tenth Symphony that Beethoven might have written if he had had the time.

Lejaren Hiller's dream was not only to compose music with a computer, but to understand both how a composer thinks and even what makes one piece of music good and another poor. Because such work had never been attempted before, the dreams clearly (and typically) overshot any possibility of realization. This problem turned out to be much harder than anyone realized, before anyone had attempted to solve it, as problems often do. Nevertheless, the dream was no less than grandiose—deep insight into the intelligence of the human mind, in this case, the musical intelligence. The computer allowed an experiment that most would say failed (if we listen to the resulting music), but I contend that the experiment succeeded in producing one of the most brilliant dreams ever had about music. The dream exceeded even those of Francis Bacon, whose brilliant imagination stopped short of including scientific investigation of the process of human musical thought. Sound example 1 (CD index 2) is the second movement of Lejaren Hiller and Leonard Isaacson's 1957 piece *Illiac Suite for String Quartet*.

Granted, Lejaren Hiller's music may sound pretty bad, but does that make it good science? Were Hiller and his associates actually using the scientific method suggested by Bacon and Galileo and others, or were they simply playing with the new technology of computers? The following words, written by Lejaren Hiller in the record liner notes about the *Illiac Suite*, address this exact question.

> Should a person listen to these two pieces as he might "ordinary music?" Yes, I think, but with this important qualification: They are much more didactic than expressive compared to most music. These pieces are truly experimental because they are concerned with revealing process as well as being final product. They are embodiments of objective research results. They are laboratory notebooks. Sometimes the results have surprised us because a compositional routine seems less effective than expected, sometimes more so. If I were to delete everything that disturbs me aesthetically, I would falsify the research record. So, for the present, my objective in composing music by means of computer programming is not the immediate realization of an aesthetic unity, but the providing and evaluation of techniques whereby this goal can eventually be realized. For this reason, in the long run I have no personal interest in using a computer to generate known styles either as an end in itself or in order to provide an illusion of having achieved a valid musical form by a tricky new way of stating well-known musical truths.

Mathews: MUSIC I (1957)

Even in 1876, Alexander Graham Bell knew that sound pressure waves could be converted into analogous electrical signals, and vice-versa, through the use of microphones and loudspeakers—that's why we call them *analog* signals. The advantage of electrical signals is that they travel much faster through wires than sound does through air, allowing electrical signals to travel much further in a small amount of time; hence the name *tele* (far off) plus *phone* (sound). By continually reconnecting a network of wires efficiently, the telephone system has probably done more to shrink the size of the world than the airplane. When electrical signals travel through wires, however, they pick up the electrical equivalent of dust and dirt, called noise. If the wire is long enough, the signal becomes so contaminated that it is no longer possible to understand the original sound. Practically the entire story of telecommunications is about ways to prevent, or at least to slow, this inevitable noise degradation of signals as they are transmitted from one place to another.

Digital technology was developed in so many

ways and places that there is no particular person identified with its origin. By the 1950s it was already known that digital signals could be transmitted from one place to another far more reliably than analog electrical signals such as those in Bell's original telephone. In a digital signal, information about, say, a sound, is encoded as a series of binary digits, or "bits," each of which can have one of only two possible values ("on/off," "yes/no," "1/0," etc.). When a bit is transmitted electrically from one place to another, it picks up noise just like any signal. The receiver, however, only needs to be able to distinguish whether a "yes" or a "no" was originally transmitted to have essentially perfect knowledge of the original signal. As long as it is not too powerful compared to the "on/off" part of the signal, the noise can be eliminated completely, making digital transmission and recording far superior to analog for many purposes. This fact was not lost on the telephone company, and in the mid-1950s a young telecommunications engineer and amateur violinist named Max Mathews was hard at work at AT&T Bell Telephone Laboratories investigating its potential. At that time the telephone company was interested only in speech. From a technical standpoint, however, music is just like speech with a few more frequencies—both are structured sound.

Using a special device called a digital-to-analog converter, a computer can control the movements of loudspeakers with a precision never before available. Max Mathews and others reasoned that because computers could be used to investigate improvements in telephone speech analysis, processing, and synthesis, they could also be used to improve the analysis, processing, and synthesis of musical sounds. Mathews therefore wrote the first music-synthesizing programs in about 1957, turning the computer into a new kind of musical instrument that was capable, in principle at least, of producing any sound that could come from a loudspeaker. Along with Lejaren Hiller's efforts to use computers to investigate musical intelligence and composition, the field of computer music was now born. The liner notes of *Music from Mathematics*, the first published recording of computer music, explained it as follows.

> On this recording, we illustrate another advancement in the realm of tools available to the music-maker: the computer and the digital-to-sound transducer. This new "instrument" combination is not merely a gadget or a complicated bit of machinery capable of producing new sounds. It opens the door to the exploration and discovery of many new and unique sounds. However, its musical usefulness and validity go far beyond this. With the development of this equipment carried out at the Bell Telephone Laboratories, the composer will have the benefits of a notational system so precise that future generations will know exactly how the composer intended his music to sound. He will have at his command an "instrument" which is itself directly involved in the creative process. In the words of three of the composers whose works are heard on this recording, man's music has always been acoustically limited by the instruments on which he plays. These are mechanisms that have physical restrictions. We have made sound and music directly from numbers, surmounting conventional limitations of instruments. Thus, the musical universe is now circumscribed only by man's perceptions and creativity.

Especially given this vision of the computer as a completely general musical instrument, perhaps the most obvious question one can ask about his original computer music sound-synthesis program is, How did it sound? The answer is simple: in musical terms, it sounded rather terrible. For one thing, the audio quality of 1950s digital audio technology was poor by today's standards (as today's technology will seem soon enough). Also, the musical quality was extremely stilted and unnatural due to the simplistic means for specifying the musical sound, which left out the performer altogether. Finally, expectations placed on the output of early computers were often unrealistic. Who would have expected that the first musical sounds produced by a multimillion-dollar example of our most advanced technology would sound more like a child's

first violin lesson than the pinnacle of musical evolution? Despite its potential to act as a completely general musical instrument, the computer's first music lesson sounded no better than yours or mine—and (not knowing how yours sounded), perhaps a lot worse.

For the remainder of computer music's first decade, most work occurred in a few laboratories and universities where expensive and hard-to-use computers were available for this purpose. Programs were typically written in assembly language using punched cards or paper tape. To run a single computer program often took a full day or more, and the results were usually disappointing, assuming the computer itself worked properly (which was often not the case). Printed output was usually on large fan-folded sheets of computer paper. If the computer also produced a sound, it was in the form of a large computer tape containing millions of numbers representing the sound in digitized form. If the sound-synthesis program succeeded in producing the tape (which one often determined by watching the tape move while the program ran), it then had to be carried to a separate machine specially designed to feed these numbers at high speed through a digital-to-analog converter, to convert the numbers into voltages. This signal could then be processed and fed into a loudspeaker (in the mid-1960s, computer music researchers from Princeton University had to drive to Bell Labs—a 150-km round trip—about once a week with one or two such tapes). One always had to remember to turn on the reel-to-reel tape recorder before starting the conversion into sound, or the whole effort might be lost if the computer tape broke. The results typically fell into these categories: (1) silence (the most common result), (2) a long, loud noise (also a common result), (3) a patterned noise sounding nothing like expected (usually the result of a programming error), or (4) a thin, weak, pale, sickly musical sound (which happened in perhaps one of every five attempts). The sounds may have been sickly and pale, and the work may have been laborious and painstakingly slow, but these results proved that computer music existed. Now all it needed was improvement!

Computer Music: The Second Decade, 1965–1975

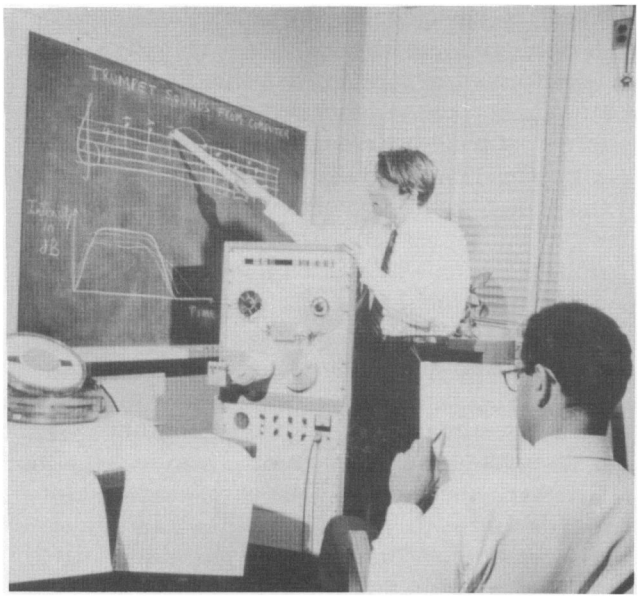
Dream: Truly Musical Sounds
Reality: Research

Risset and the Trumpet

It is difficult to make music on a musical instrument that does not sound like a musical instrument. Therefore, one of the first problems to be tackled seriously in the field of computer music was that of synthesizing sounds with truly musical characteristics. Two basic approaches to this problem are either to imitate the sounds of traditional musical instruments or to synthesize completely new sounds that are musically satisfying. Both of these problems are difficult, but for different reasons. Imitations are always tricky, because even the slightest discrepancy will alert the listener to try to find others. Elsewhere I have called this the "oleomargarine effect" by analogy with another famous example of how substitutes rarely work perfectly. The other approach, that of synthesizing new sounds with musical properties, is also difficult because listeners have such ingrained habits of thought regarding what sounds are "musical." Despite years of the "liberation of noise" and other campaigns by modern musicians, most people simply have different aesthetic expectations when they hear traditional musical instruments, such as violins and guitars, as opposed to sounds never before encountered. A kind of "categorical perception" occurs in which some sounds are placed into the "musical" category while others (such as speech and noise) are not. Some interesting sounds have been discovered, however, most notably an entire category of "acoustical illusions," named for their ability to "fool the ear" in ways analogous to the manner in which optical illusions "fool the eye."

By the mid-1960s, computer sound synthesis had advanced to the point where musical researchers could begin to address the problem of musical tone color, or timbre. Based on the assumption that it would be necessary to understand what makes the sounds of traditional instruments "musical" to be-

Figure 2. Jean-Claude Risset demonstrating a trumpet tone synthesized by computer at Bell Telephone Laboratories in 1965. (Photograph courtesy of Bell Telephone Laboratories.)



gin with before new ones could be invented, Jean-Claude Risset, a physicist and composer working with Max Mathews at Bell Labs, investigated the sound of the trumpet. What he learned confirmed what had been suspected for some time—that there is not a single recipe for the entirety of any musical sound. Rather, the recipe—the relative proportions of sine waves that are added together to form a complete sound—must change in characteristic ways from moment to moment within a sound, even from the beginning to the end of a single note. In technical terms, the spectrum of the sound waveform must vary over time in the correct way. The equivalent problem for cooking might be to try to invent a food that tastes and feels like a green apple when you bite into it, then gradually becomes more like roast beef as you chew it, then tastes and feels like port wine as you swallow it. That is essentially what all musical sounds do, except that taste and feel are replaced by timbre, the time-varying mixture of frequency components that fuse to form a single musical sound. Figure 2 shows Risset at Bell Labs in 1965 presenting his work on trumpet tones.

Max Mathews's synthesis programs allowed people to specify the physical properties of virtually any musical sound, but musicians are almost always after a particular set of musical characteristics, usually a matter of perception. One of the things early computer music researchers quickly learned was that our knowledge of the relationship between the physical and psychological characteristics of sound—psychoacoustics—is about as limited as our knowledge of every other aspect of the human mind. Looking at waveform plots, for example, we soon learned that fairly large-looking differences between two sound waves don't necessarily correspond to large-sounding differences, and the opposite is also true (apparently human beings can both see *and* hear because we get truly different information about the environment from each of these senses).

Jean-Claude Risset's investigation into trumpet sound led to a key insight into the sounds of virtually all brass musical instruments. The physical operation of brass instruments causes the amplitude, or strength, of the sound to build up, reach a more-or-less steady state, then die away at the end, all in more-or-less exponential fashion (this is a common characteristic of many physical systems). Risset determined that the bandwidth of the sound—essentially the number of harmonics—grows in direct proportion to the amplitude of the sound. While that simple fact is not all there is to brass-instrument sounds, it is so important that virtually any sound with this characteristic will sound brass-like, or at least more like brass than any other type of instrument.

Despite its limitations, Jean-Claude Risset's "brass tone hypothesis" was a real breakthrough, and an encouragement that at least some important characteristics of musical timbre could be understood. Risset celebrated his discovery by featuring his synthetic trumpet in computer-generated portions of the incidental music he wrote for *Little Boy*, a play that tells the true story of the pilot who flew the plane that dropped the atomic bomb on Hiroshima. The flight crew had named the bomb "Little Boy." Risset used his computer-synthesized trumpet to accompany sequences in the play in which the pilot dreamed about happier times in a jazz club he had frequented. The pilot—who later

went insane—was tormented by nightmares in which he "became" the bomb as it fell endlessly toward its target. Here Risset found an eloquent application for a computer-synthesized acoustic illusion. Based on recently understood principles of psychoacoustics, he created a long tone whose pitch gradually descends forever without getting any lower (like the barber pole stripes that appear to go down forever without getting anywhere) to accompany the pilot's dream of endless falling. Sound example 2 (CD index 3) is an excerpt from Jean-Claude Risset's 1968 *Suite from Little Boy.*

Ruiz and Physical Models

Jean-Claude Risset's success was based on understanding the physical characteristics of brass instruments and human perception. Max Mathews's programs allowed users to specify the physical characteristics of virtually any musical sound. Lejaren Hiller borrowed stochastic methods used in modeling molecular physics to model the music composition process. A young graduate student of Hiller's, Pierre Ruiz, working at the University of Illinois and later at Bell Labs with Risset, took computer music synthesis a big step further. Physicists have known for some time how to write differential equations that model the behavior of physical systems, including those of musical instruments. The trouble is that such equations quickly grow more complicated than any person can solve in a general mathematical way. The computer, however, can be programmed to simulate the moment-to-moment results of such equations even when they cannot be solved generally. If the equations correctly model the behavior of the musical instrument, then the computer should be able to compute the resulting sound. Ruiz applied this approach, called *physical modeling*, to the violin.

While this may be said of all computer music sound-synthesis techniques, physical modeling is truly a technique in which mathematical equations become musical instruments. The computer, in effect, "plays" the formulae in ways that no human ever could, thereby obtaining sounds. There are many problems with this technique, not the least of which is that it requires intimate familiarity with mathematical physics. Another problem with physical modeling is that it is difficult even for the computer, because an enormous amount of computation must usually be done to obtain even modest results. This fact relegates most physical modeling to the exotic realm of supercomputers. But the advantages of physical modeling are also very real, especially if one is interested in synthesizing the sound of traditional musical instruments. Pierre Ruiz's early work in this field was so demanding that, despite obtaining the best synthetic violin tones made at the time, he left the field of computer music altogether, as have many others when they discovered the true cost of some dreams. One might have expected someone with Ruiz's mathematical abilities to investigate other issues in science or technology, but such was not the case. He went into experimental theater.

Music V and GROOVE

When I first arrived at Bell Labs in 1967, after studying music composition and performance at Carnegie-Mellon University and the University of Illinois, one of my first jobs was to write the computer program that processed computer tapes containing sounds. I also became deeply involved in Mathews's last sound synthesis project, called Music V because it was the fifth version of that program he had designed over about ten years. While Music V embodied many ideas and technical advances of interminable interest to programmers, perhaps its most important feature was that it was the first such program written in a portable programming language—in this case FORTRAN IV—rather than machine-specific assembly language. This allowed Music V to be carried from one computer to another and still stand a fairly good chance of working. As a public service to the new field of computer music, Bell Labs gave the Music V program away for free, primarily to universities that owned suitable computer systems such as IBM 7094s, Honeywell/GE 645s, and Control Data 6600s. Keep in mind that a typical computer system was the size of a small house, and cost many

Figure 3. The IBM 650 computer (first delivered in 1954). (From S. Augarten, 1984. Bit by Bit—An Illustrated History of Computers. *New York: Ticknor and Fields, p. 210. Reused by permission.)*

Figure 4. The high-performance CDC 6600 computer of 1963. Other examples of this generation of machines were the IBM 7090 or 7094 and the GE 635/645, on which Music V was developed. (From Augarten 1984, p. 219. Reused by permission.)

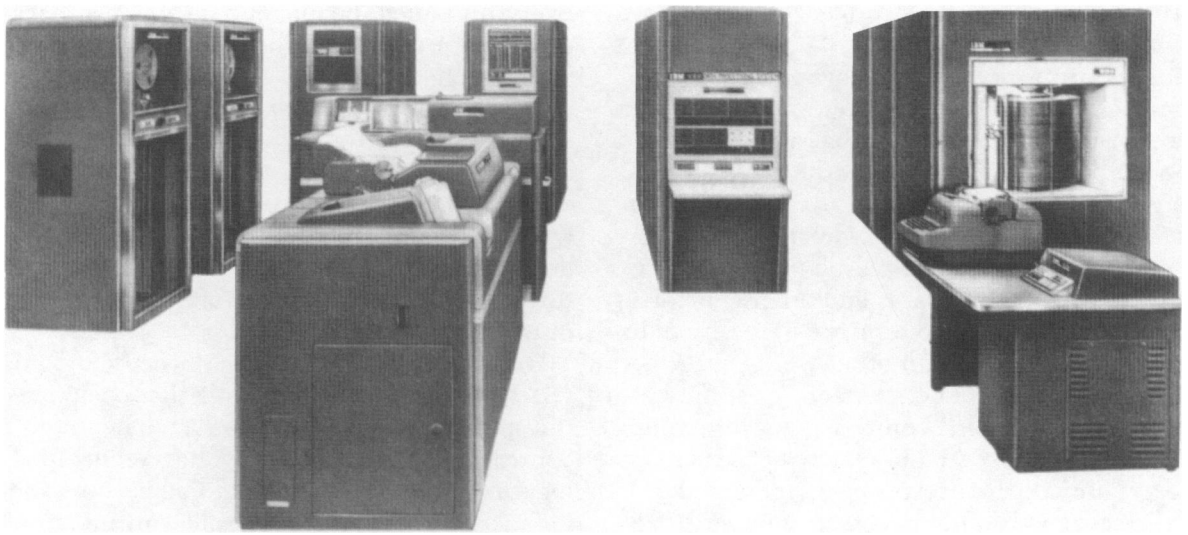


Figure 3

millions of dollars. The Music V program was distributed in several boxes, each having a capacity of roughly 2,000 punched cards (don't worry—they were numbered in case someone dropped them!).

Music V was a flexible and powerful tool for both making computer-synthesized music and for studying the relationship between the physics of sound and its perception. It was also clear that the *modus operandi* of Music V, which consisted of typed descriptions of one or more virtual musical "instruments" in a special language designed for that purpose, followed by a typed list of notes to be played on these instruments, mimicked well the concepts embodied in traditional written music. The program treated its input as the specification for a sound, which it then proceeded to synthesize.

What this paradigm leaves out is anything in particular having to do with a performer's contribution to music. When a performer plays a piece of music, a great deal happens, most of which is poorly indicated—if at all—in a written score. There are usually noteworthy differences between reading the script for a play and seeing it performed on stage or screen. In both theater and music, these differences are called *performance*. The long list of names at the end of a movie testifies to all the people who have added something to the book, which usually

Figure 4

exists before the movie-making process even begins. A performance is filled with nuances and inflections associated with such things as the general historical period and style of a piece, the performer's personal "interpretations," and the physical characteristics of the listening space. In the case of music, the physical characteristics of an instrument also contribute a great deal to the resulting sound. Without these refinements, a piece of music usually tends to sound flat and lifeless, as when a

primitive computer speech synthesizer reads a love poem.

There are two general approaches to solving the performance problem in computer music. One is to write more complicated scores that are designed to specify the precise final sound of the finished music—a much more complex task than writing a traditional musical score. The second approach is to allow computer music to be played in the traditional manner with keyboards and other controllers, thereby altogether avoiding the necessity of understanding performance. In other words, we can either try to understand performance or simply "capture" it. While some composers have been up to the first task, most are not, partly because many composers are not themselves performers, but mostly because we do not necessarily understand what happens when performers perform. Because it involves additional understanding, the first approach is a matter of further research, even today. The second approach avoids the necessity for understanding by simply capturing aspects of performance and using them directly. As we will see later, this "capturing" approach has also been applied successfully in the area of sound synthesis.

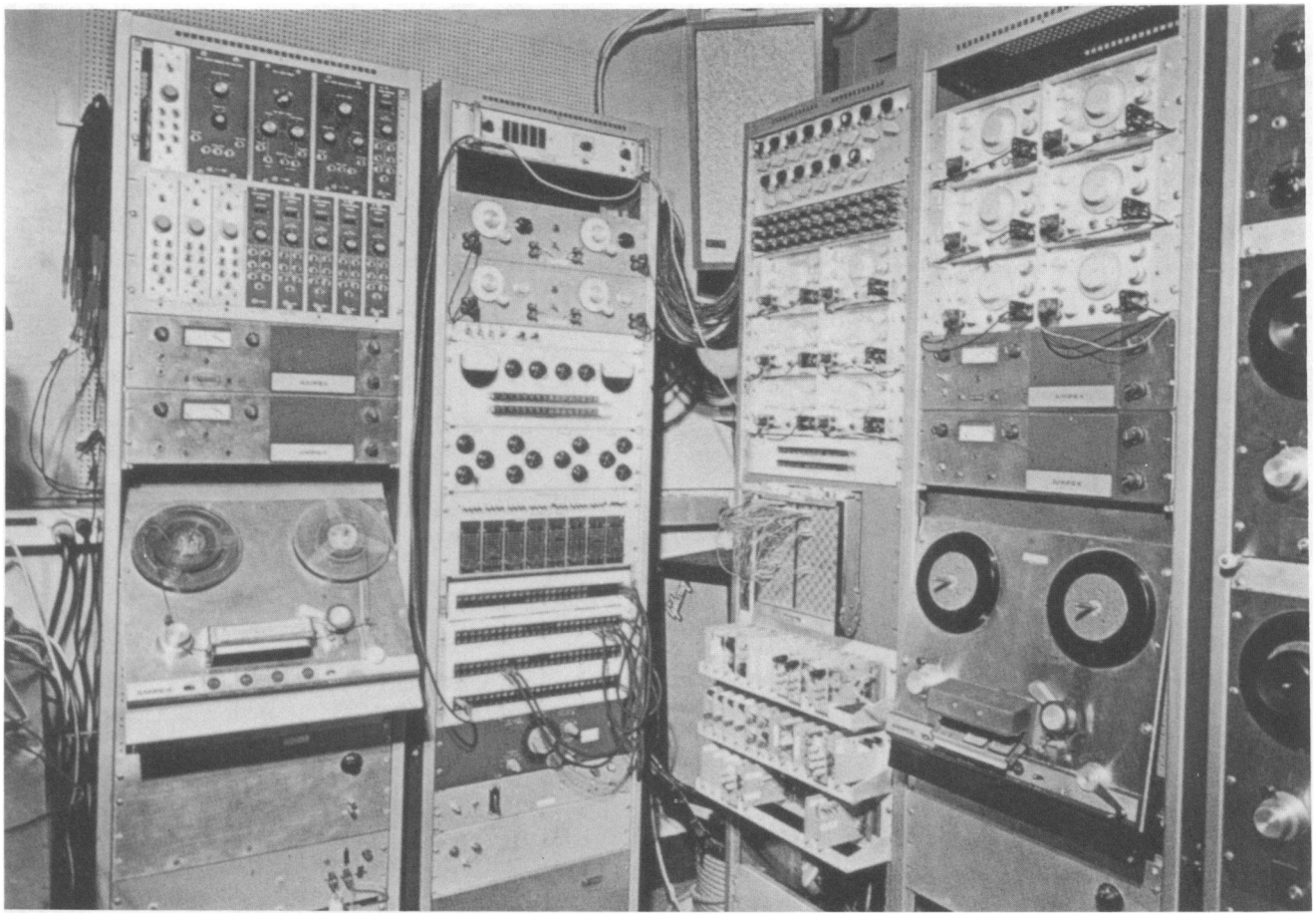
As a pianist and percussionist as well as a composer, I was equally interested in the computer music problems of sound synthesis and performance. The only trouble was that by the late 1960s there was no performance capability in computer music, which suggested to Max Mathews and me that we should work on one. The result was a system we called GROOVE, for Generated Real-time Operations On Voltage-controlled Equipment.

At that time, there were no digital computers capable of synthesizing sound in real time. Music V was an extremely powerful tool, but it worked independently of real time. This means that it synthesized the specified sounds as quickly as possible, but not as the sound was heard. Non-real-time sound synthesis is similar to creating an animated film; all the individual drawings must be completed before the film can be viewed and the motion implicit in those frames can be experienced. While virtually anything can appear on those frames, the synthesis process generally takes much longer than viewing the final result in real time. Everything must be planned in such a situation—every gesture, every scene—no spontaneous "improvisation" is possible. If computer music was ever to sound as lively and expressive as other music, we needed a way to account for what performers contribute to music. Since we didn't really understand performance well enough to specify it with written instructions (and we still don't), we had to resort to the capturing approach—and to capture what performers do, a computer music system must operate in real time.

For the GROOVE system, Mathews and I used a relatively small computer (one that occupied only the space of an apartment rather than a house) to control another roomful of voltage-controlled analog synthesis equipment that had become the basis for such popular (at that time) electronic music devices as the Moog synthesizer (of *Switched-On Bach* fame). It is shown in Figure 5. The voltages that controlled the pitches and amplitudes of the synthesis equipment were obtained through a bank of 14 digital-to-analog converters attached to a Honeywell DDP-224 computer. These voltages were updated by the computer at a rate of between 100 and 200 times per sec instead of the 10,000 to 50,000 times per sec needed for direct digital sound synthesis. Though the sound potential of the analog synthesis equipment was far more limited than direct digital synthesis, by using digital-to-analog converters to control only such things as pitch and amplitude rather than the actual sound waveform itself, we gained a factor of about 100 in available computation time, and 5 or 10 msec computers—even 1969 computers as big as apartments—can do a fair amount of computing! We then built and connected piano-like keyboards, pedals, joysticks, knobs, and switches to the computer, all of which could affect the sound being synthesized in real time according to the user's program. Like a player piano, the GROOVE system recorded the actions we made pushing keys and twisting knobs, rather than the sounds produced. For the first time, we could truly play the computer like a musical instrument—even though it still didn't really sound like one.

The result was that for the first time computers could be involved in improvisation and every other

Figure 5. The analog side of the early–1970s GROOVE system, developed at Bell Telephone Laboratories by Max Mathews and the author. The left-most equipment rack contains an Ampex two-channel tape recorder and a collection of Moog analog synthesizer modules above it. The other racks include a variety of analog voltage-controlled oscillators (with the large frequency tuning knobs) and filters, and patch bays for configuring the interconnection of these elements.



kind of spontaneous musical decision-making. These are things that trained musicians are extremely good at, and, like walking, are far easier to do than to explain. Much research on real-time systems for computer music has been done for the purpose of circumventing deeper questions regarding how performers perform. Fortunately, the very existence of real-time computer music systems has made it feasible to study performance in ways never before possible. Robert Willey, a bright young graduate student of mine at University of California, San Diego, recently earned a doctorate in music by studying the effect of instrument response delay on performers. This is just one of the many questions that can now be asked and answered using real-time computer music systems.

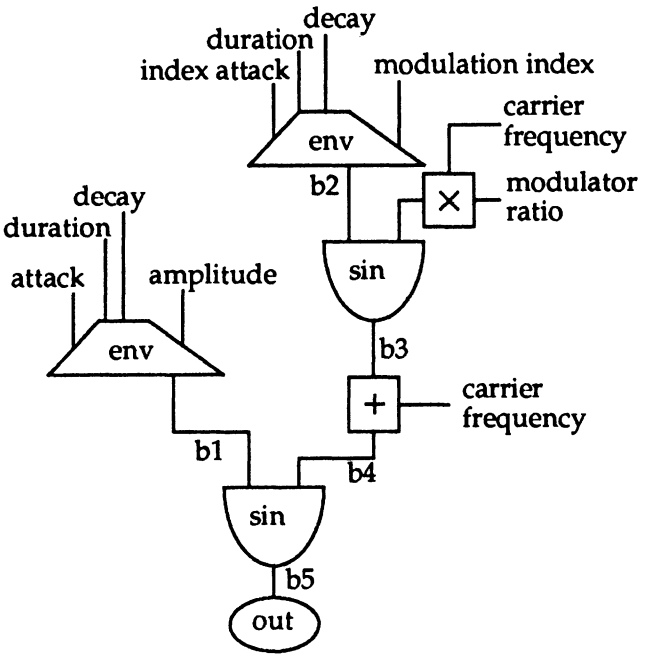
The CD that accompanies this issue includes three examples from the GROOVE system. The piece at CD index 4 is an excerpt from Emmanuel Ghent's work *Phosphones*, and indices 5 and 7 are two short GROOVE pieces by Laurie Spiegel.

The example at index 6 on the CD was produced on the real-time interactive synthesizer developed by Hal Alles and co-workers at Bell Telephone Laboratories; it is an excerpt from one of Laurie Spiegel's improvisations on the machine.

Frequency Modulation

In the late 1960s, another young graduate student of music named John Chowning was using Music V at Stanford University to investigate the musical effects of extreme vibrato. Due to a programming

Figure 6. A flow chart of a Music V-style instrument for frequency modulation (FM). The two oscillators have icons that resemble half-circles; the trapezoidal figures are time-varying envelope generators.



accident (as he himself describes it) he specified a sound with a vibrato rate much larger than a normal vibrato, which is usually about 5 or 6 vibrations per sec. In fact, the vibrato he specified was as high or even higher than the fundamental frequency of the sound itself—something no singer or traditional instrument could possibly produce. Much to Chowning's surprise, what came out was something that didn't sound like it had a vibrato at all. Instead, the result was a completely different tone quality. The simple process of vibrato, when carried to such an extreme, can be understood by borrowing a theory from radio transmission, called *frequency modulation* (FM). Even though FM radio transmission had been well understood since the 1930s, no one ever thought of listening directly to the modulated waveform itself before, and that's not how John Chowning thought of it either. What he did was accidentally transfer the FM theory from radio to musical psychoacoustics, and *voilà!* a fascinating new music synthesis technique was born.

Of course, FM is not musically interesting because its theory comes from radio. It is interesting because it is very simple to compute (one need only modulate the frequency of one sine wave with another—a simple and efficient process). Furthermore, two things that are easy to control in FM are the *modulating frequency*, which corresponds to the vibrato rate, and something called the *modulation index*, which is simply related to the vibrato depth (the extent to which the frequency varies above and below the main, or *carrier frequency*). It turns out that the ratio of the modulating and carrier frequencies determines whether the resulting musical timbre has a harmonic or inharmonic spectrum, which is an extremely important musical characteristic in perceptual terms. Furthermore, the modulation index is directly related to another perceptually important quality of sound—its bandwidth, or "brightness." Figure 6 shows the block diagram of an FM instrument.

Before John Chowning's wonderful discovery, no one had any idea that such a simple and efficient synthesis process could provide such straightforward control over many perceptually important characteristics of musical sound. Furthermore, no one had any idea that such a discovery in the computer music field could be so profitable. Stanford University patented the discovery, which was later licensed to the Yamaha Corporation, giving Yamaha a distinct early advantage in the computer music instrument marketplace. For several years, this patent produced royalties in excess of any ever issued to Stanford University. The discovery of FM synthesis—more than any other single development—was responsible for the proliferation of computer music from universities and laboratories into studios, onto stages, into concerts, onto recordings, and ultimately, into homes all over the world. Sound example 7 (CD index 8) is John Chowning's 1972 piece *Turenas*.

Computer Music: The Third Decade, 1975–1985

Dream: Proliferation
Reality: Development

The Digital Revolution

In 1960 the first *integrated circuits* were fabricated, marking the real beginning of the digital revolution

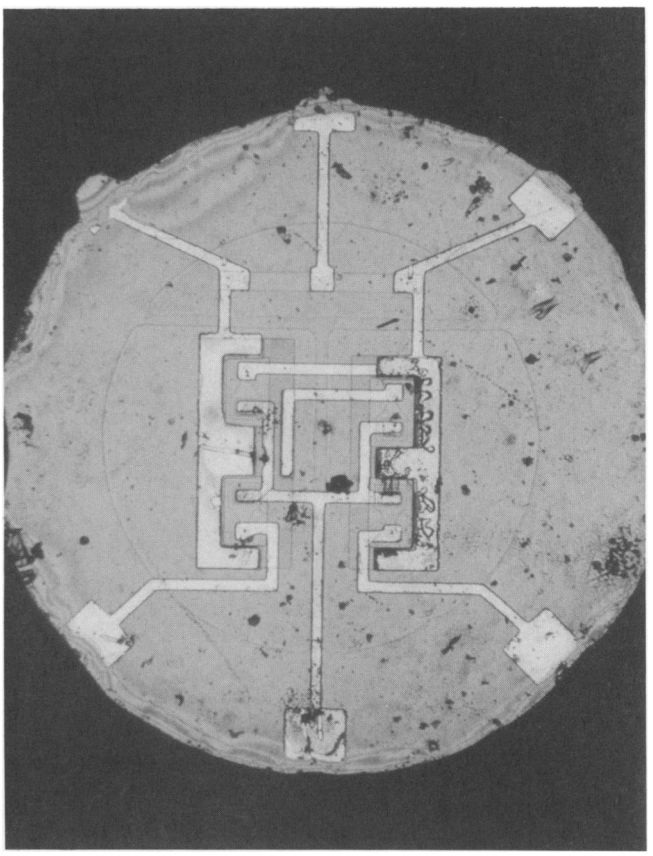


Figure 7. A 1961-vintage integrated circuit (a dual flip-flop) constructed by the Fairchild Camera and Instrument Corp. The white lines are aluminum interconnects; the four conical shapes around the center of the figure are the transistors. The entire device is 0.06-inch in diameter. (From Augarten 1984, p. 224. Reused by permission.)

(see Figures 7 and 8). About 10 years later, in 1971, Marcian Hoff of Intel Corporation invented the microprocessor, placing an entire computer central processor unit (CPU) on a single chip of silicon about the size of a fingernail.

Such facts gave rise to a well-known maxim in the semiconductor industry known as *Moore's Law* (after Gordon Moore of the Intel Corporation). Moore's Law states that the complexity of integrated circuits is proportional to 2^{Y-1960}, where Y is the current year. In other words, every year (starting in 1960) the complexity of integrated circuits doubles, meaning that roughly twice as many gates, bits of memory, or whatever, can be put on a single chip as compared with the year before. A corollary to Moore's Law states that cost is approximately proportional to the square root of complexity, meaning that the cost of doing any particular thing with integrated circuits will be cut in half about once every two years.

Moore's Law and its corollary have continued to be valid (more or less) ever since they were introduced. They are important because they codify one of the most important aspects of computer technology, namely, that it is getting cheaper as time goes by. Not only that, it is getting cheaper at a remarkable rate. Computer systems aren't getting cheaper quite so fast, of course, because not everything in a computer is an integrated circuit. The cost of an equivalent computer is only falling at the rate of about one half every three years, still an incredible price reduction compared with virtually everything else in the known universe.

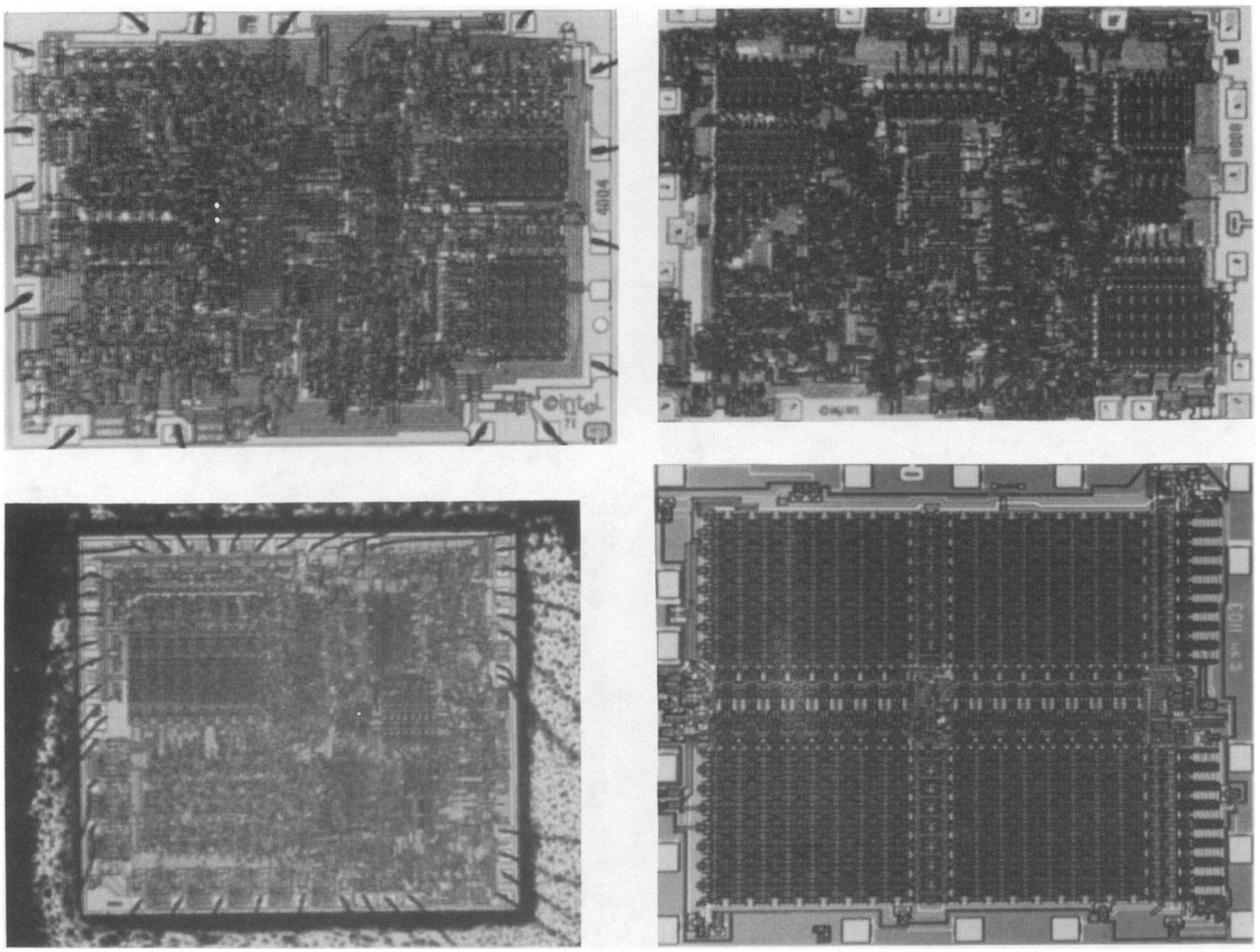
There is a dark side to Moore's Law, however. It says that for the same price, complexity will increase by a factor of four every two years or so. In other words, computers keep getting more and more capable, so much so that after a computer is just a few years old, it can no longer compete with newer models in either performance or cost. At some point it actually becomes cheaper to replace an old computer than to upgrade or even repair it. Moore's Law is therefore an important basis for the dreaded phenomenon known as *technological obsolescence*. Computer designs progress by "generations" that are only about three or four years long. A "last-generation" computer may still be usable for many things, but by the time a computer is two generations old it becomes nearly useless, like last month's newspapers.

Machines, Machines ...

The same principles hold true for any devices based on integrated circuits, like digital music synthesizers. With the beginning of the real-time digital music synthesis era, research in computer music was booming. In 1975 I had temporarily left Bell Labs to learn how to make digital synthesizers at Stanford University, where I studied computer engineering and built the FRMbox, one of the world's first all-digital music synthesizers shown in Figure 9.

Computers were still the size of cars and small trucks, and, much as with the GROOVE system,

Figure 8. Three historic microprocessor chips from Intel—a 4004 (1970; left-top), 8008 (1972; right-top) and 8080 (1974; left-bottom), and an Intel 1103 1k-bit memory chip (1970; right-bottom). (From Augarten 1984, p. 266. Reused by permission.)



one entire computer had to be dedicated to control my synthesizer. Its design was documented in my dissertation, and represented a kind of digitization of the analog synthesis portion of the GROOVE system. Peter Samson of the Systems Concepts company also built a large digital synthesizer that became the basis for much music created at Stanford University's Center for Computer Research in Music and Acoustics (CCRMA) which John Chowning and others founded around that time. Pierre Boulez created the Institute de Recherche et Coordination, Acoustique/Musique (IRCAM) in Paris, which opened in 1978. At IRCAM, a brilliant young nuclear physicist named Giuseppe Di Giugno started designing a digital synthesizer, working first with Luciano Berio, and later with Pierre Boulez and many others. Hal Alles built a research digital music synthesizer at Bell Labs around one of the new LSI-11 microcomputer systems, shrinking the size of the control computer from pickup truck to TV set. In 1979 I left Bell Labs to found the Computer Audio Research Laboratory (CARL) project at the University of California, San Diego, an effort dedicated to doing everything possible to sound and music with computers, and vice versa.

All of these efforts to build computer music instruments, among several others, were research projects that consumed fairly large efforts and bud-

Figure 9. The 1975 FRMbox digital synthesizer developed by the author at Stanford University. This was one of the first all-digital sound synthesis machines, developed around the same time as the Systems Concepts Digital Synthesizer (or "Samson box") used at CCRMA, and the "4" series of machines built at IRCAM in Paris.

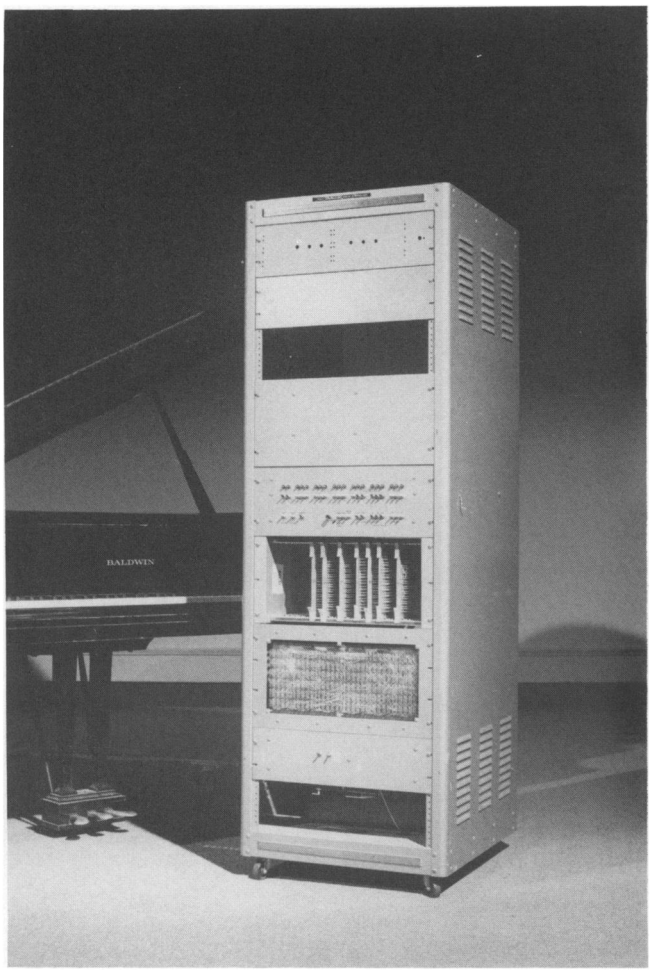
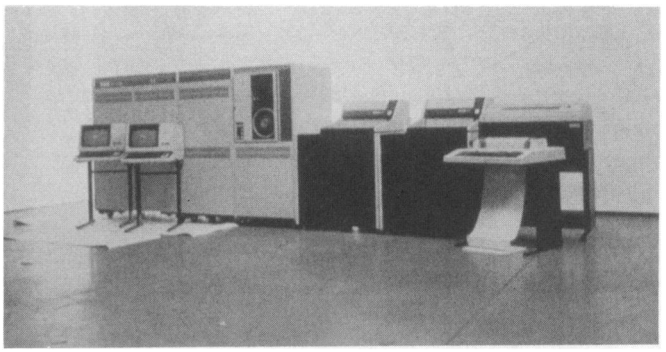


Figure 10. A Digital Equipment Corp. DEC VAX-11/780 of the late 1970s. (From Augarten 1984, p. 258. Reused by permission.)



gets, especially if the cost of the computer was included. The Digital Equipment Corporation VAX-11/780 (see figure 10) computer system that was the mainstay of the CARL project for many years cost about half a million dollars (US$), though it did serve many users simultaneously (albeit not too well, by today's standards). While it did not seem unreasonable that a major research effort in music might cost as much as a single grand opera production, tensions rose as with the costs of these efforts conflicted with the needs of individual musicians who were used to working on more modest budgets. In those days we wanted real-time digital computer music synthesis, and thought in terms of how many grand pianos it would cost. We dreamed of the day when real-time computer music synthesis would cost no more than a single grand piano, making it possible for a dedicated individual musician to afford one. Then computer music might begin to be on the same social basis as the rest of music, and we were sure that it would proliferate widely after that.

Our dreams of proliferation were far too modest. Within a few years, the Yamaha Corporation had developed inexpensive chips that could do much of what the large, expensive research prototypes could do. The first digital music synthesizer marketed in the United States was the Synclavier, designed by Sidney Alonso and others at New England Digital Company, a small computer company. Such machines, which appeared in the late 1970s, carried price tags similar to average automobiles, and were fairly limited in their capabilities.

But nothing could really compare on the price-performance curve with the first good, cheap synthesizer, the Yamaha DX-7, introduced in 1983. It cost about US$ 2,000. Based on John Chowning's FM synthesis algorithm, within a few years the DX-7 had sold more than 500,000 units to the international market. It was not only the world's most popular digital music synthesizer, but possibly the world's most popular musical instrument.

In addition to the DX-7, another key development in computer music was the agreement among several important manufacturers to a voluntary standard for connecting digital music synthesizers to each other (regardless of brand!) and to the new, small personal computers. The resulting Musical Instrument Digital Interface, or MIDI, allows people

to combine the features of different digital synthesizers and computers in various ways—a critical means of combating technological obsolescence. Whatever their shortcomings, MIDI and the DX-7 created a proliferation that was completely unprecedented in the field of music. Many other instruments and companies followed.

One device that deserves special mention in this context was the Fairlight Computer Music Instrument (CMI). Just as it is possible to solve the performance problem without understanding it, it is possible to solve the sound synthesis problem in a similar manner. The Fairlight was the first of a new genre of digital music instruments that works by recording sounds in its digital memory (a process called "sampling") in such a way that they can be played back at various pitches and amplitudes under control of a keyboard. The so-called sampling synthesizer could put a recording of virtually any sound under any key, though there are some limits to how much pitch modification can be done without degrading the sound.

Computer Music: The Fourth Decade, 1985 to the Present (and Beyond)

Dream: Interfacing Mind with Reality
Reality: Virtual

In about 1985 the complexion of computer music began to change; what was research oriented was becoming market oriented. Following the tremendously successful proliferation of computer music instruments from the laboratories and into homes and studios, shifts began to occur that are still profoundly affecting the music field in general. Computers have continued to shrink and get cheaper at rates predicted by Moore's Law to the point where they are now commonplace. In fact, computers, telephones, fax machines, television, radio, cinema, and print media all appear to be converging into a single information technology that is inherently centered around real time, telecommunicating, and audio-visual computer-based information exchange. Explosions in information storage, processing, and display have already combined with interchange to create what is now called the World-Wide Web (WWW), but that is just the beginning. In effect, the interconnected computers of the world are beginning to wake up, just as their users are beginning to realize how powerful an instrument of mind (as Marvin Minsky called them) computers really are. As we prepare to enter the 21st century, computers are empowering individuals in ways reminiscent of the scientific revolution—no longer is the world's most powerful technology available only to those in privileged positions.

Computer music has also jumped out of the laboratory and into the fire. Traditional musical instruments continue to get more expensive (like everything else) while computer music instruments just keep getting better and cheaper. Most new pianos sold are now digital, and this trend shows no signs of abating. Computer music is no longer an esoteric corner of music but has in fact become part of the mainstream, at least in terms of the marketplace.

The new face of computer music is as shallow as it is broad, however. Rather than promoting research into basic musical questions, most computer music now merely substitutes newer, cheaper technology for strings and pipes. In that sense the field of music has effectively "absorbed" computer music with little or no effect on what music is produced or our understanding of it.

Perhaps this is as it should be. Technology is changing much more rapidly than people.

After all is said and done, the dreams of today are not much different from those of our ancestors, except, of course, that we have many more opportunities to realize those dreams than they did.

Perhaps too, it is difficult to realize what a dream is about while one is in the midst of having it. Since my own interest in computer music over the years has been largely oriented toward creating new capabilities, perhaps the best way to access current "dreams" is to examine what capabilities still do not exist but need to.

Clearly, the technology of computer music will continue to improve and get cheaper. General-purpose computers are becoming so fast that soon we will have to slow them down—rather than

speed them up with specialized-but-limited hardware—in order to achieve real-time performance. This means that soon we will be able to retain the generality and flexibility of the Music V model and have real-time interaction, too. Computers are—or computing is—becoming an essentially free resource. This inevitably will put more and more of our experience of the world into the "computer-mediated" category, meaning that our distinction between reality and what we now call virtual reality will gradually disappear. Rather than being a "dial-a-dream" situation, the real world will actually be under the power and control of the mind.

As an example of this, consider the "fly-by-wire" systems in modern aircraft. Instead of directly controlling the airplane, a pilot operates controls that are sensed by the computer, which in turn flies the airplane. In effect, the pilot's input only expresses his wishes—the computer then responds to these wishes in a manner that is screened for such things as flight safety and passenger comfort. One result is that far more complex airplanes can be built than ever could be flown by an unassisted human pilot. Another result is that airplanes sometimes crash due to programming errors.

Imagine now a computer-based music machine that senses the musical desires of an individual listener. The listener might simply turn a knob one way when the computer plays something the listener likes, the other way when the computer does something less likable. Or, better yet, the computer could sense the listener's responses directly using, say, body temperature, pulse rate, galvanic skin response, pupil size, blood pressure, etc. Imagine what music would sound like if it continually adapted itself to your neurophysiological responses, for as long as you wish. Such music might be more addictive than any known drug, or it might cure any of several known medical disorders.

Of course, such a music machine cannot be built today but that is precisely my point. Without such dreams we have no idea how to direct our increasingly marvelous technology. Without dreams we can only repeat what has already happened. Dreams are the only true agents of change.