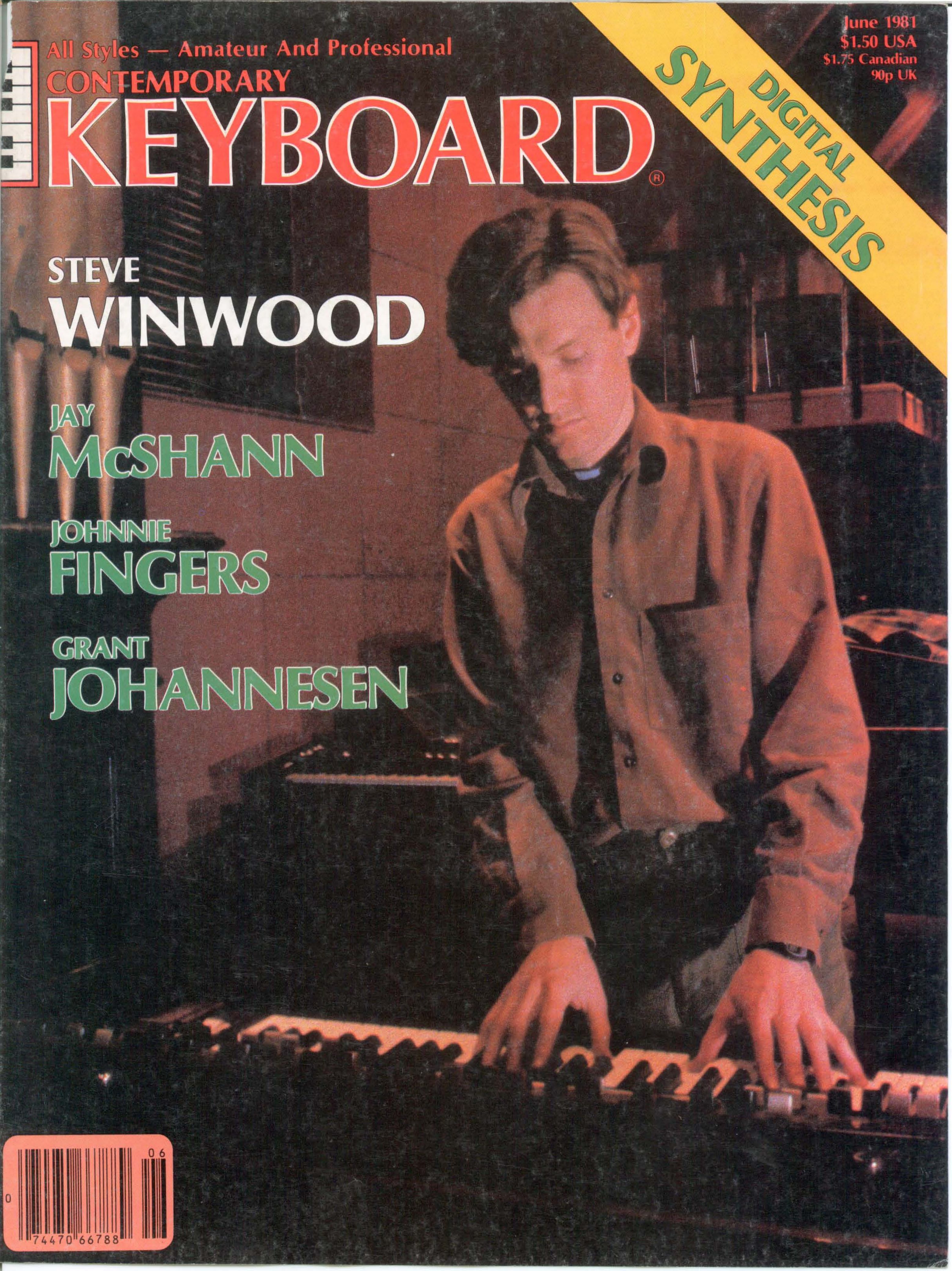
All Styles — Amateur And Professional

CONTEMPORARY KEYBOARD®

June 1981
$1.50 USA
$1.75 Canadian
90p UK

DIGITAL SYNTHESIS



STEVE WINWOOD

JAY McSHANN

JOHNNIE FINGERS

GRANT JOHANNESEN

FROM THE PUBLISHER

WELL, YOU ALMOST didn't have a May issue last month. It seems that our humble (occasionally) editor, Tom Darter, was a bit distracted from his appointed task of whatever it is he does.

His wife Sibyl decided it was time to give birth, and damn the deadlines, full speed ahead! Or rather, nature gave her her own deadline, and husband Tom had to choose his allegiance. Wisely, he elected to turn most of the actual work on the May issue over to Jim, Bob, Dominic, and Ellayn. It became their job not just to get the issue off to press, but to make it appear to the publisher that dear Tom had actually been working those last few days surrounding the arrival of 7-lb, 8-oz Lisa.

Both deliveries, *Keyboard's* and Lisa's, had their strained moments, but it's nice to be able to report that both saw the light of day on time and in good spirits. Congratulations to Tom, Sibyl, big sister Erika, and Lisa Darter. And a few hefty pats on the collective back to the *Keyboard* staff.

* * * *

This issue contains the sixth installment of Dominic Milano's new column, Keyboard Report. What do you think? Where are we succeeding, and where are we letting you down? Are we on target when it comes to selecting products you're interested in? Are there certain areas of evaluation we consistently overlook? We try to be as fair as we can, but everybody has biases. Are our biases showing?

As regular readers know, we at *Keyboard* pay a great deal of attention to your opinions. So take a minute, please, to drop me a card about Keyboard Report. We want to make it the most valuable product-purchasing aid you could want, and we can only succeed if we have your input.

* * * *

For $4\frac{1}{2}$ years author, pianist, composer, critic, and music historian Leonard Feather has been writing our Piano Giants Of Jazz column, but this issue marks the last installment in that series. During this time, Leonard has introduced hundreds of thousands of readers to pianists ranging from Ammons to Zawinul, presenting their history as well as an analysis of their music. Leonard won't be leaving *Keyboard's* pages altogether, though, we're happy to report. He's agreed to lend his knowledge and talent to a series of feature articles, the first of which will appear later this year.

Also winding up in this issue is the Electronic Perspectives column penned for more than four years by electronic music historian and synthesizer design consultant Tom Rhea. Tom's column has documented the fascinating odyssey of the development of electronic musical instruments, from the Trautonium and the Theremin to today's digital synthesizers. Tom will

Continued on page 75

A GPI Publication

CONTEMPORARY KEYBOARD®

Vol. 7, No. 6 June 1981

The Magazine For All Keyboard Players

President/Board Chairman
L.V. Eastman
Publisher
Jim Crockett
Associate Publisher
Don Menn

Editor
Tom Darter
Assistant Editors
Jim Aikin
Bob Doerschuk
Dominic Milano
Art Director
Wales Christian Ledgerwood
Advertising
Ron Taff (Director)
Judy Pauli (Asst.)
Pat Nicosia (Acct.)
Office Coordinator
Deby L. Cronk

Music Coordinator
Jim Ferguson
Typesetting
Leslie Bartz
Staff Photographer
Jon Sievert

Circulation
Jay Johnson (Sales Dir. Dealers.)
Joan Kreamer (Fullfillment. Mgr.)
Lucienne O'Connor, Peggy Shea
Subscriber Services
Gloria Smith (714) 485-6535
Box 28836, San Diego, CA 92128
Dealer Services
Jake Hunter (408) 446-1105
20605 Lazaneo, Cupertino, CA 95014

Administration
Ellayn M. Evans, Frances Jackson
Susan Johnson, Jenn Kohler, Cheryl Matthews
Tom Murphy, Suzi Peterson
Bonnie Schoenemann, Peter Yu
Mai Sano-Lindstrom (Chief Accountant)
Laurie Walters (Personnel)
Public Relations
Gretchen Horton

Columns

Departments

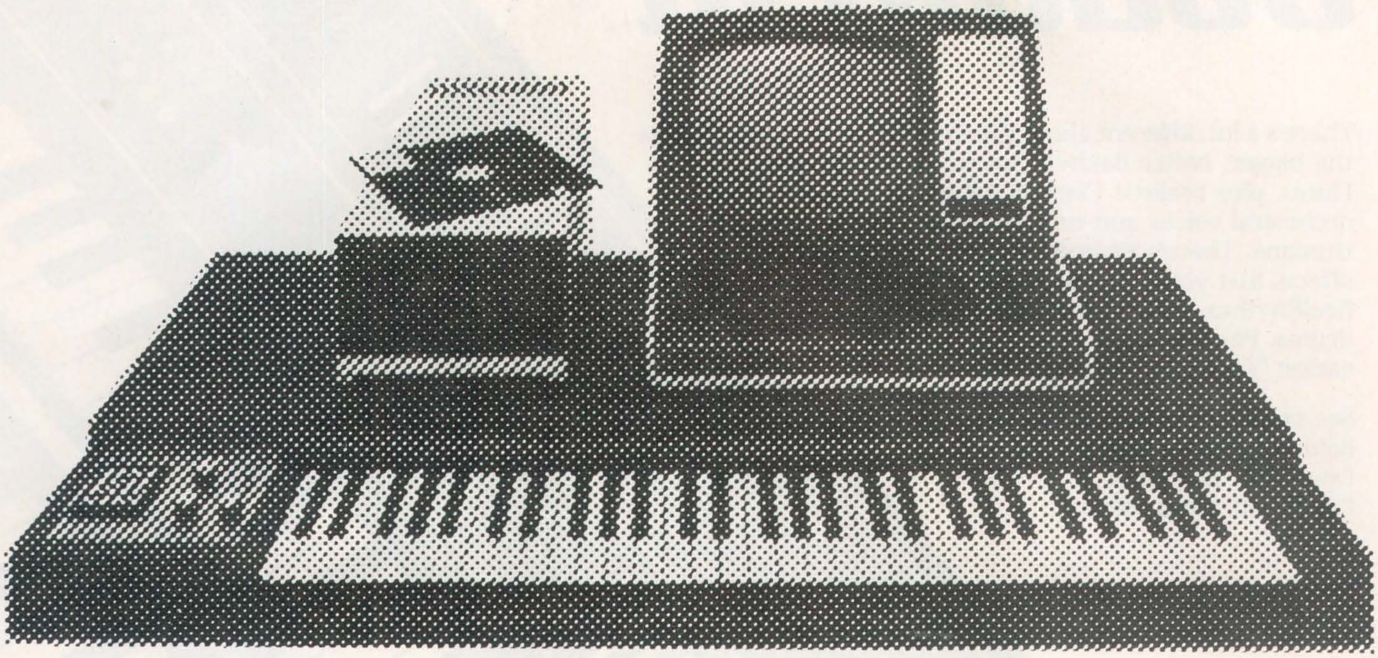
Cover photo by Adrian Boot

Also from GPI: Guitar Player Magazine, Tom Wheeler, Mng. Editor • Frets Magazine, Roger Siminoff, Editor

CONTEMPORARY KEYBOARD (ISSN 0361-5820) is published monthly by GPI Publications, which also issues *Guitar Player* and *Frets* magazines. Copyright © 1981 by GPI Publications. All rights reserved. Controlled circulation at Long Prairie, Minnesota. Canadian postage paid at Winnipeg, Manitoba. **POSTMASTER:** Please send form 3579 to P.O. Box 28836, San Diego, CA 92128. **EXECUTIVE, EDITORIAL AND ADVERTISING OFFICES:** 20605 Lazaneo, Cupertino, CA 95014; (408) 446-1105. Publisher assumes no responsibility for return of unsolicited manuscripts, photographs, or artwork. **SUBSCRIPTIONS:** $17.95 per year, outside U.S. and Canada $23. Address all subscription correspondence to Subscription Dept., P.O. Box 28836, San Diego, CA 92128 Allow 7-9 weeks for new subscriptions or change of address. Send mailing label plus old and new addresses for address change.

basics of digital sound, part two:

digital synthesis



By John Strawn

DURING THE LAST couple of years, the big news in music technology has been the application of digital electronic equipment to sound. The use of large general-purpose computers in the generation of signals that can be processed to produce sound dates back more than twenty years; and more recently, advances in microcircuitry have made practical the manufacture of small digital synthesizers whose hardware and software are dedicated specifically to musical applications. This article is not a survey of current commercially available digital synthesizers, but rather an introduction to the theory behind their operation. In the first part of this article, which appeared in *Keyboard* in March, we discussed the basics of the related field of digital recording. This month we will go on to take a look at exactly how computers and other digital equipment can be used to generate new sounds. But first, we need to review briefly the terms that were introduced in the first article.

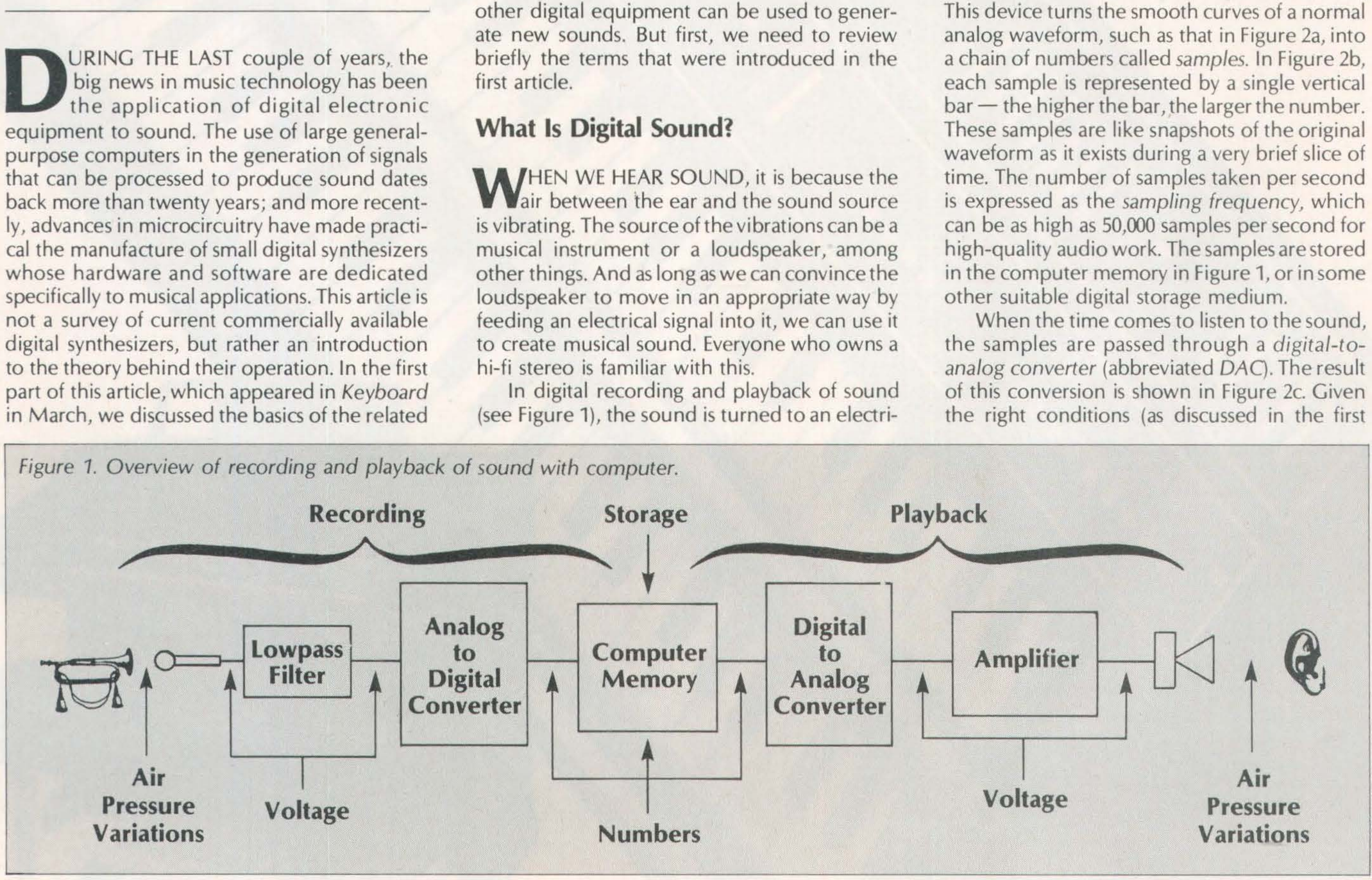
What Is Digital Sound?

WHEN WE HEAR SOUND, it is because the air between the ear and the sound source is vibrating. The source of the vibrations can be a musical instrument or a loudspeaker, among other things. And as long as we can convince the loudspeaker to move in an appropriate way by feeding an electrical signal into it, we can use it to create musical sound. Everyone who owns a hi-fi stereo is familiar with this.

In digital recording and playback of sound (see Figure 1), the sound is turned to an electrical signal by a microphone, and this signal is processed by an *analog-to-digital converter*. This device turns the smooth curves of a normal analog waveform, such as that in Figure 2a, into a chain of numbers called *samples*. In Figure 2b, each sample is represented by a single vertical bar — the higher the bar, the larger the number. These samples are like snapshots of the original waveform as it exists during a very brief slice of time. The number of samples taken per second is expressed as the *sampling frequency*, which can be as high as 50,000 samples per second for high-quality audio work. The samples are stored in the computer memory in Figure 1, or in some other suitable digital storage medium.

When the time comes to listen to the sound, the samples are passed through a *digital-to-analog converter* (abbreviated *DAC*). The result of this conversion is shown in Figure 2c. Given the right conditions (as discussed in the first

Figure 1. Overview of recording and playback of sound with computer.



DIAGRAMS BY LACHLAN

installment of this article) it is possible to reconstruct the original signal exactly from the samples stored as digital numbers.

Digital Synthesis

NOW THAT WE HAVE REVIEWED some of the terms used in working with digital sound, let's examine how a computer or other digital equipment can be used to *create* sound. In other words, we will split off and ignore the part of Figure 1 labelled "recording." The computer will still be providing numbers for the digital-to-analog converter from its memory, but now the numbers will be created by the computer itself rather than played back from a recorded input. To simplify matters, we will look at the problems involved in synthesizing a simple signal such as that shown in Figure 2a. This is technically known as a *sine wave,* and it forms an important building block in many synthesis techniques. The samples representing a sine wave can be calculated in a computer using formulae from trigonometry. The simplest way of thinking of computer synthesis would be to imagine a computer program which would cause the computer to calculate one value of the sine waveform for each sample, and then send that sample to the DAC. This process would be repeated as long as the composer wanted to hear a sine wave. The frequency of the sine wave (and thus the pitch of the note) would be determined by how often the sine wave is repeated during a given time (expressed as the number of *cycles per second*).

There are many reasons for not using the computer in this way, however. One is simply that of efficiency. If the same sine waveform is going to be used over and over again, it seems wasteful to make the computer calculate the same numbers every time. A simpler method is to have the computer calculate the numbers for one cycle of a sine wave and then store these numbers in a list inside the computer memory, as shown in Figure 3. Such a list is called a *wavetable.* After the computer has filled the wavetable with numbers, it can *look up* the number it needs for each sample, and send the number to the DAC. This process of *table lookup* is much quicker than calculating the value for each sample from scratch. Obviously, we can calculate and store any waveshape we like in the wavetable. For purposes of discussion, we will continue to assume that the wavetable contains a sine wave, since synthesists who work with computers often prefer processing and modifying sine waves to defining other waveshapes.

Figure 2a. This waveform is known as a sine wave because it can be calculated using trigonometric formulae for the sine of an angle. It represents one of the simplest building blocks available in sound synthesis.

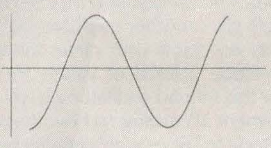


Figure 2b. A sampled version of the sine wave in Figure 2a, as it might appear at the output of the analog-to-digital converter in Figure 1. Each vertical bar represents one sample. Each sample is stored inside the computer memory as a number which represents the height of the vertical bar.

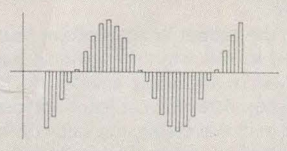


Figure 2c. Reconstruction of the sampled version of the waveform given in Figure 2b, as it might appear at the output of the digital-to-analog converter in Figure 1. Roughly speaking, the tops of the samples are connected to form the waveform which eventually reaches the listener's ear.

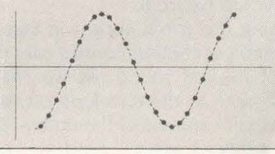
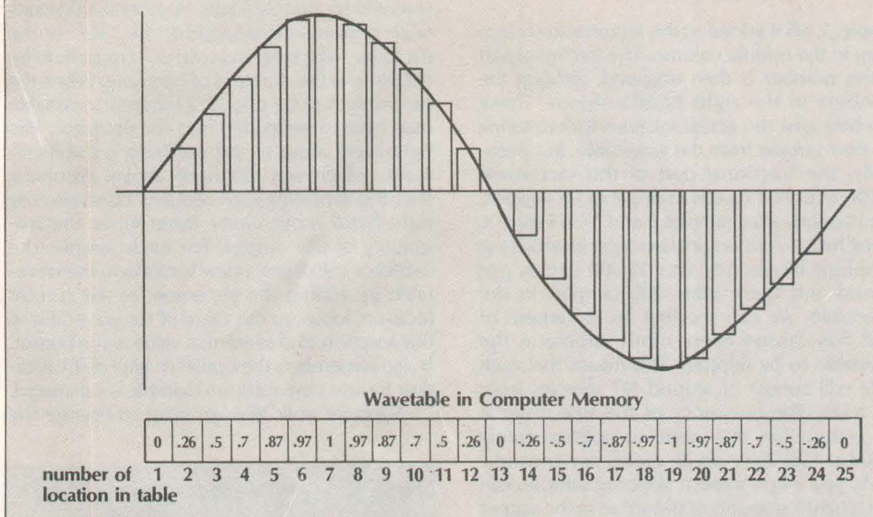


Figure 3. It is possible to synthesize the waveform shown at the top by calculating and storing the values of the waveform in a wavetable, as shown below, and then reading out the values of the wavetable one at a time.



Wavetable in Computer Memory

	0	.26	.5	.7	.87	.97	1	.97	.87	.7	.5	.26	0	-.26	-.5	-.7	-.87	-.97	-1	-.97	-.87	-.7	-.5	-.26	0
number of location in table	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Suppose, then, that the value of the first sample is given by the first number in the wavetable in Figure 3. For each new sample to be produced by our simple digital synthesizer, we will take the next sample from the wavetable. When we get to the end of the wavetable, we simply go back to the beginning and start reading out the samples again. What would be the frequency of the sound produced by doing this? If the sampling frequency were 10,000 cycles per second, and there were 200 numbers in the table, then there would be 10,000/200 = 50 cycles per second, corresponding to a note near the bottom end of the keyboard. On the other hand, if the sampling frequency were 50,000 cycles per second and the table contained 500 entries, the output would be at 100 cycles per second, since 50,000/500 = 100. The frequency of the note thus depends on the sampling frequency and the length of the wavetable.

What if we want to change the frequency of the output signal? One way would be to change the sampling frequency; this is in fact the method used in the Synclavier and Fairlight synthesizers. Another solution would be to skip some of the samples stored in the wavetable. For example, if we only take the even-numbered samples, then we would go through the table twice as fast. This would raise the pitch of the output signal by an octave. If we skip two samples and take every third sample, the pitch would be raised even further (by an octave and a fifth, to be exact). The number of samples skipped is related to a number called the *increment.* The increment is added to the location we are currently looking at in the table (this location is sometimes called the *phase*) in order to calculate the location at which we will find the next sample to read and sent to the output. In the simple example in which we were reading every sample from the table, the increment is 1. If we read only the odd- or even-numbered samples, the increment will be 2.

If we want to change the frequency by some amount smaller than an octave, we will have to use an increment between 1 and 2. Suppose, for example, that we use an increment of 1.125. The successive locations in the lookup table would be calculated as shown in Figure 4. For each new

Figure 4. This table shows the process of finding successive samples by using wavetable lookup. When the increment is greater than 1, some samples will be skipped.

Sample number	Location in table Calculated	Actual
1	1.000	1
2	2.125	2
3	3.250	3
4	4.375	4
5	5.500	5
6	6.625	6
7	7.750	7
8	8.875	8
9	10.000	10
10	11.125	11
11	12.250	12
12	13.375	13
13	14.500	14
14	15.625	15
15	16.750	16
16	17.875	17
17	19.000	19

John Strawn graduated from the Oberlin Conservatory in 1973, where he studied organ with Haskell Thomson and electronic music with Ron Pellegrino and Robert Moore. After two years at the Technische Universität in Berlin on a Fulbright scholarship, he spent another year abroad in various electronic music studios on a Thomas Watson Fellowship. Since 1976, he has been a Ph.D. candidate at the Computer Center for Research in Music and Acoustics (CCRMA) at Stanford University, working with John Chowning. In addition, he has served as one of the editors of Computer Music Journal *for the past three years.*

sample, 1.125 is added to the accumulated value given in the middle column. The fractional part of this number is then dropped, yielding the numbers in the right-hand column. These numbers give the actual location for retrieving the new sample from the wavetable. But eventually, the fractional part of this increment builds up until it causes a sample to be skipped. This happens after samples 8 and 17 in Figure 4. Going back to our original example in which the sampling frequency was 10,000 cycles per second and there were 200 samples in the wavetable, we can see that an increment of 1.125 has caused every ninth sample in the wavetable to be skipped. This means that each cycle will consist of around 177 samples from the table. The frequency of the sine wave at the output has thus risen to approximately 10,000 x 1.25/200 = 56.25 cycles per second.

As you might expect, skipping samples can lead to small amounts of distortion in the output signal. This is called *table lookup noise,* and there are various ways of dealing with it, but they're too complicated to go into here.

Controlling The Parameters Of Sound

SO FAR, WE HAVE SEEN how to produce a simple sine wave at a fixed frequency. Since the maximum value of this wave does not change in time, the signal would not change in loudness either. In fact, the computer would continue to create this simple-minded note from the time the program was started until it was stopped. This is not very useful for making music!

In general, composers who work with digital synthesis speak of the following parameters of sound: frequency, amplitude, timbre (or tone color), duration, location in space (left/right, front/back, distance), and even the motion of an apparent sound source relative to the listener. There won't be enough space to discuss all these in detail here. But to give you a better idea of how some of these facets of sound can be controlled in digital synthesis, let's consider the control of amplitude and duration of a note.

At this point, it will be helpful to introduce some notation which is often used as a form of shorthand in publications on digital sound synthesis. One of the first programs to gain widespread use in computer synthesis is called MUSIC V. An important concept in this program is the *unit generator,* abbreviated *u.g.* To put together an "instrument" for sound synthesis, the composer connects these unit generators together. (In MUSIC V these generators don't exist as separate hardware, but are aspects of the computer program. Some commercially available digital synthesizers, on the other hand, do have unit generators that exist as dedicated pieces of hardware.) When a unit generator produces a number at its output, that number can become the input of another unit generator. By shuffling the numbers around in this way, the computer "connects" the unit generators.

One important unit generator is the *oscillator,* shown in MUSIC V notation in Figure 5a (the symbol for each generator has a unique shape). The oscillator accepts two control inputs and produces one output. In fact, this generator works exactly like the wavetable lookup discussed in the previous section. Before a given oscillator can be used to produce sound, the computer must store numbers in the wavetable reserved for that oscillator. In general, this wavetable remains unchanged for the "life" of the oscillator, which for now we will consider to be the same as the duration of one note. When the next note is to be played, a different wavetable may have to be loaded into the oscillator. The right-hand input to the oscillator is the increment, which was discussed above. Assuming that the sampling rate remains constant, the right-hand input alone determines the frequency of the output. For each sample, the oscillator calculates a new location in the wavetable by adding the increment to the current location, looks up the value of the wavetable at that location, and sends that value to the output. It also remembers the updated value of the location for use when the next sample is calculated.

Suppose now that we want to change the loudness of the output of the oscillator. We could conceivably connect the output of the oscillator to the input of another unit generator, called a *multiplier,* which is shown in Figure 5b. The multiplier simply takes two numbers, multiplies them together, and sends the product to its output. The resulting *patch,* as such connections are called by analog synthesists, would appear as shown in Figure 6. The other input to the multiplier would be some number, let's say between 0 and 1. When the number is 0, of course the output of the multiplier (and hence of the entire patch) would be silence. When the number is 1, the output of the oscillator would be unchanged by the multiplier. Any number between 0 and 1 would produce a somewhat softer output from the multiplier.

Figure 5a. MUSIC V notation for an oscillator.

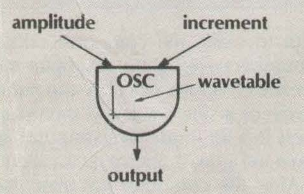


Figure 5b. A two-input multiplier.

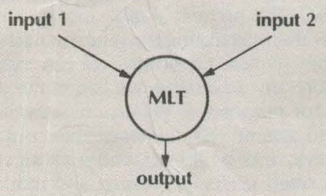
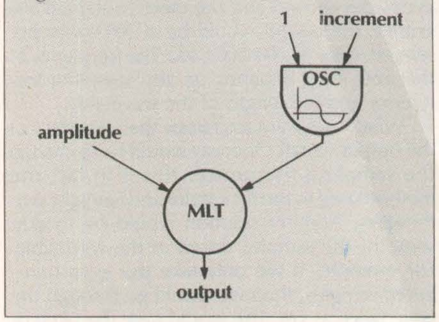
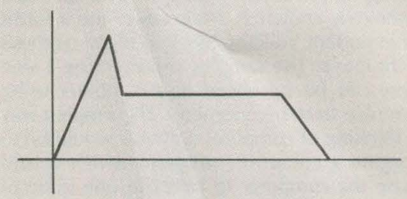


Figure 6. A simplistic way to control amplitude, expressed in the notation shown in Figure 5.



The loudness of a "note" produced by a traditional acoustic musical instrument varies during the duration of the note. Typically, the note starts with zero amplitude, works its way up to some maximum value, and dies down again more or less slowly to zero. A simplified representation of such an *amplitude envelope,* as such a pattern is called, is shown in Figure 7. (If you've worked with analog synthesizers, you're already familiar with the idea of envelope shapes.) If we could somehow hook up such an envelope to the second input of the multiplier in Figure 6, we could use this combination to start producing notes. In other words, we could multiply each sample of the sine wave from the oscillator by some value from an appropriate part of the amplitude envelope. We are now getting closer to controlling our simple oscillator in musical terms: If we can control the duration and maximum amplitude of the envelope, then the envelope will control the note for us.

Figure 7. This graph, also called an amplitude envelope, shows a simplified representation of the way in which the amplitude of a note changes throughout its duration.



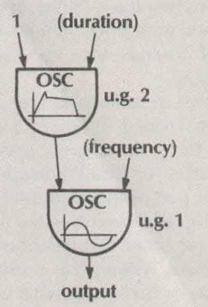
The problem becomes one of getting a value of the amplitude envelope for each sample of the output. And the solution may be obvious by now. We'll take another oscillator, but this time we'll fill its wavetable with values for the amplitude envelope instead of values for the sine wave. For this second oscillator we will calculate the increment according to how long we want the envelope to last — with a smaller increment, it will take the computer longer to go through the envelope wavetable. Now we have the desired control over the duration of the note. For each sample, the second oscillator will produce at its output a value derived from the amplitude envelope. We will then take this value and use it as the left-hand input for the multiplier.

Since the output of an oscillator must often be scaled in this way, the process has been streamlined by including a left-hand input in the oscillator. After the oscillator has looked up a sample in its wavetable, the value of the sample is multiplied *inside the oscillator* by whatever appears at the left-hand input. (In Figure 6, this input was set to a constant number.) Since this multiplication can be done inside the oscillator itself, there is no need for the extra multiplier. We can simply hook up the output of the envelope oscillator to the left-hand input of u.g. 1, as shown in Figure 8.

By using the patch shown in Figure 8, we have gained yet another control over the parameters of musical sound. As we change the left-hand input to the *envelope oscillator* (u.g. 2), the *overall* loudness of the note will change. A small number there will produce a soft note, and so on. This gives us the desired control over the loudness of the note.

As you might imagine, it is possible to connect oscillators and other unit generators in a wide variety of ways in order to make (hopefully musical) sound. For example, if we wanted to make a glissando on a note, another envelope could be added to the increment (frequency) input of the sine wave oscillator (u.g. 1) in Figure 8. We could also take several such sets of oscillators and add together their outputs on a sample-by-sample basis to create a more complex (and hopefully more interesting) sound.

Figure 8. The upper oscillator is used as an "envelope generator" to control the amplitude of the sine wave generated by the lower oscillator.



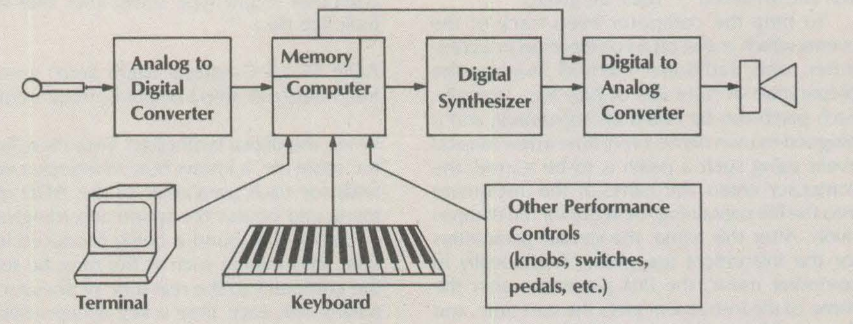
Real-Time Digital Synthesis

THERE IS ANOTHER important aspect of digital synthesis. Figure 9 shows the individual steps which might be necessary for calculating one sample of sound in one oscillator. The important point here is that each step in Figure 9 takes some amount of time to perform. For example, it might take .002 seconds to perform the calculations in steps 1 and 3 in Figure 9. For reasons we won't discuss here, it would typically take the computer even longer to do the multiplication in step 5. But if we are using a sampling frequency of 50,000 samples per second, the time available to create one sample is only 1/50,000 = .00002 second! Apparently, it would be impossible to have the computer do all the calculations necessary for one sample in one oscillator, not to mention the calculations necessary for all the other oscillators and unit generators.

Figure 9. Summary of the steps necessary for one oscillator to calculate one sample.

1. Add increment to current wavetable location to obtain new wavetable location.
2. If the new location lies past the end of the wavetable, correct the location by subtracting the wavetable length.
3. Store the new location for use in calculating the next sample (the new location for this sample becomes the current location for the next sample).
4. Find the value in the wavetable at the new location.
5. Multiply that value by the amplitude input.
6. Send the product to the output.

Figure 10. Overview of a digital recording and synthesis facility. The composer/performer would communicate with the digital synthesizer using keyboards and other controls as well as a computer terminal, all of which are connected to the computer. Sound can be recorded and stored in the computer for processing and later playback through the digital-to-analog converter. Synthesis is accomplished by special-purpose hardware controlled by the computer. This hardware produces streams of numbers which are converted to sound by the digital-to-analog converter.


Analog to Digital Converter
Memory
Computer
Digital Synthesizer
Digital to Analog Converter
Other Performance Controls (knobs, switches, pedals, etc.)
Terminal
Keyboard

In the early days of digital sound synthesis, the solution was to allow the computer to take as much time as it needed to calculate one sample, and to store the samples in a file inside the computer. Only after the samples for all or part of a composition were calculated could the samples be sent to the DAC so the composer could hear the sound. This might mean that the computer would have to calculate all night in order to produce a few minutes' worth of music! Of course, if some small mistake had been made, the entire process would have to be repeated. Even though this was a laborious process, to say the least, composers such as Jean-Claude Risset, J.K. Randall, and John Chowning were able to create full-length computer-generated works of music.

As computers have become smaller and digital technology has become less expensive, it has become practical to build special-purpose devices — *digital synthesizers* — which are fast enough to do all the calculations necessary for one or more samples within the time allowable for one sample. This is known as *real-time* synthesis. An overview of a real-time analysis/synthesis system is given in Figure 10. One obvious advantage of a real-time synthesizer is that controllers such as keyboards and footpedals can be attached to it, so that the sound can be modified as it is being produced. Another advantage of having a computer as part of a synthesizer is that the computer can quickly repatch the system according to the user's needs.

Communicating Musical Ideas To The Computer

THE TRADITIONAL WAY of thinking about constructing a piece of music is to imagine the composer selecting various notes and writing a score or chart which directs the performers to play the notes one after another. Sound synthesis with computer allows us to introduce a more generalized idea, that of a *musical event,* which might be much longer than one note typically lasts, or more complicated (for example, an event might consist of several notes in a bundle).

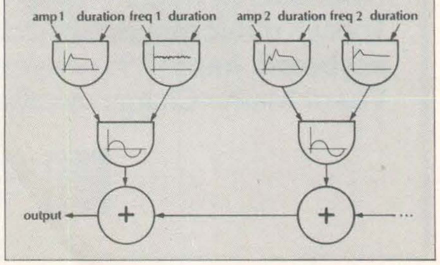
In programming a synthesizer to produce musical events, it is often helpful to think in terms of some model of the sound to be produced. One method of deriving such models is to analyze traditional musical instruments. Using the techniques of digital signal processing, a large amount of data can be collected about the instrument. When these data have been analyzed, a patch such as that shown in Figure 11 can be used to re-synthesize the original sound. If the amplitude and frequency envelopes controlling the sine oscillators are given suitable values, it is possible to generate tones which are very difficult to distinguish in listening tests from the digital recording of the original tone. Of course, reproducing traditional instruments is only one of the tricks of computer music. But by doing this, we can test whether our models of sound seem to work, and then use the model as a basis for exploring non-traditional sounds and compositional structures.

If we were using the patch shown in Figure 11, then for each musical event produced using this model, we would need to specify the following parameters for each set of oscillators:

frequency
overall amplitude
amplitude envelope
begin time of amplitude envelope
duration of amplitude envelope
frequency envelope
begin time of frequency envelope
duration of frequency envelope
where to put the output (which channel)

Continued

Figure 11. The patch shown in Figure 8 has been expanded to form a simplified additive synthesis instrument. Each sine oscillator is modified by amplitude and frequency envelopes. The outputs of the two oscillators are added together to make one sample. More three-oscillator units like these might be added to the patch to make more complicated sounds.



If there are 15 oscillators, and each oscillator requires these nine items, then 135 pieces of information must be specified for just one sound event! In addition, the patch — that is, the configuration of oscillators, adders, multipliers, and whatever — must be given.

To help the composer keep track of the events which make up a composition in a computer, one traditional method involves the preparation of *note lists* or *play lists.* Typically, each patch can be called an *instrument*, and is assigned its own name. Every time a new musical event using such a patch is to be started, the composer enters the name of the instrument into the file containing the score for the composition. After this name, the various parameters for the instrument are given. Traditionally in computer music, the first parameter after the name of the instrument gives the start time, and the second parameter gives the duration for the event. In the case of the patch shown in Figure 11, the same begin time and duration might be used for all the envelopes. Subsequent parameters will have different meanings, depending on the nature of the instrument.

To give a concrete example, suppose the composer wants to invoke a musical event starting 15 seconds after the beginning of the composition. The event is to be a note 1.5 seconds long. The sound will be created by an instrument which plays two oscillators, as shown in Figure 11; we will call this instrument ADD, because the synthesis technique which this patch implements is known as *additive synthesis.* The frequencies (increments) will be called freq1 and freq2, and the amplitudes will be amp1 and amp2 for the first and second oscillators, respectively. Likewise, the envelopes will be called ampEnv1, ampEnv2, freqEnv1, and so on. The output will be sent someplace called Right, which means that the event will occur only on the right-hand speaker of a stereo setup. The end of an entire *instrument call* is marked by a semicolon. In the score, then, the composer might type some lines that would look like this:

```
ADD 15 1.5 C-natural Right amp1 ampEnv1
freq1 freqEnv1 amp2 ampEnv2 freq2 freqEnv2;
```

When the digital synthesizer finds these lines in the score file, it knows how to interpret what it finds for each parameter of the ADD instrument, and creates the sound accordingly.

When the sound is being produced in real time, information such as this must be sent by the computer to the real-time synthesizer. This means that each time a key is depressed, the computer must calculate the frequency, find the right envelopes, calculate the increments, set up the patch, and then start everything up. As you can see, the commercial digital synthesizers already on the market have to deal with many problems which (hopefully!) remain invisible to the user. For example, if the computer is not fast enough to transmit all of this data to the synthesizer itself, then the keyboard response will be sluggish, or a thick chord will sound arpeggiated. (This is the kind of thing to listen for when checking out digital keyboard synthesizers.) Overcoming these problems is a test of the programmer's skill and ingenuity, and the real-time playability of an instrument depends on the effectiveness of the solutions implemented.

Digital vs. Analog Synthesis

IT WOULD BE IMPOSSIBLE to detail all of the advantages of digital synthesis in comparison with older synthesis techniques (such as orchestras, pipe organs, and analog synthesizers). This is *not* meant to imply that someday digital synthesis might replace traditional musical instruments. Rather, the point is that musicians are fortunate in having access to digital synthesis because it allows them to explore musical areas that were previously inaccessible.

A major advantage of digital synthesis is related to the fact that sound in its digital form is represented as a string of numbers. An enormous amount of work has been devoted to creating computers to be used for working with numbers. If the proper set of specifications can be prepared, the computer can be instructed to create almost any imaginable string of numbers. This means that the techniques of digital synthesis are not inherently limited in terms of the kinds of sounds that can be created and manipulated.

Another advantage is that once a number has been stored inside the computer, the number does not change (assuming that the equipment is functioning properly). This tenacity on the part of the computer applies to individual samples as well as to the timing, loudness, timbre, and other features of events within the composition. When the composer wants to hear today or tomorrow what was created during yesterday's work, it will still sound the same. This can be a major advantage over traditional musical instruments, which tend to go out of tune, play differently under varying conditions of temperature and humidity, and so on.

Finally, digital synthesis can be used to explore areas which were difficult even to imagine in previous generations. What would it sound like if the human voice were crossed with a clarinet or an oboe to create a "talking woodwind"? What are the esthetic implications of structuring each individual sound in a composition to match the structure of the overall composition? Clearly, we can imagine whatever possibilities we like, and the techniques of digital synthesis give us a powerful tool for realizing them.

Summary: Digital Recording & Synthesis

CURRENTLY, DIGITAL synthesizers tend to cost between $15,000 and $30,000. Fortunately for the musicians who are already working in this area, or who would like to start, digital components (chips) are becoming cheaper and faster, thanks to the heavy emphasis on the research and development of digital electronics for commercial and military applications. The spin-off of this for the music world will be cheaper, more powerful digital synthesizers in the not-too-distant future. We can expect that commercially available digital synthesizers will follow the pattern established by their analog cousins; that is, the giant companies will tend to concentrate on push-button instruments that the user can operate with no knowledge whatever of the processes being carried on behind the front panel. In other words, we can expect to see a generation of souped-up preset organs with the word "digital" prominently displayed on the front panel. The most exciting benefits of advanced technology, though, will go, as they do already, to those who are willing to learn enough about their equipment to envision and then explore these new horizons.