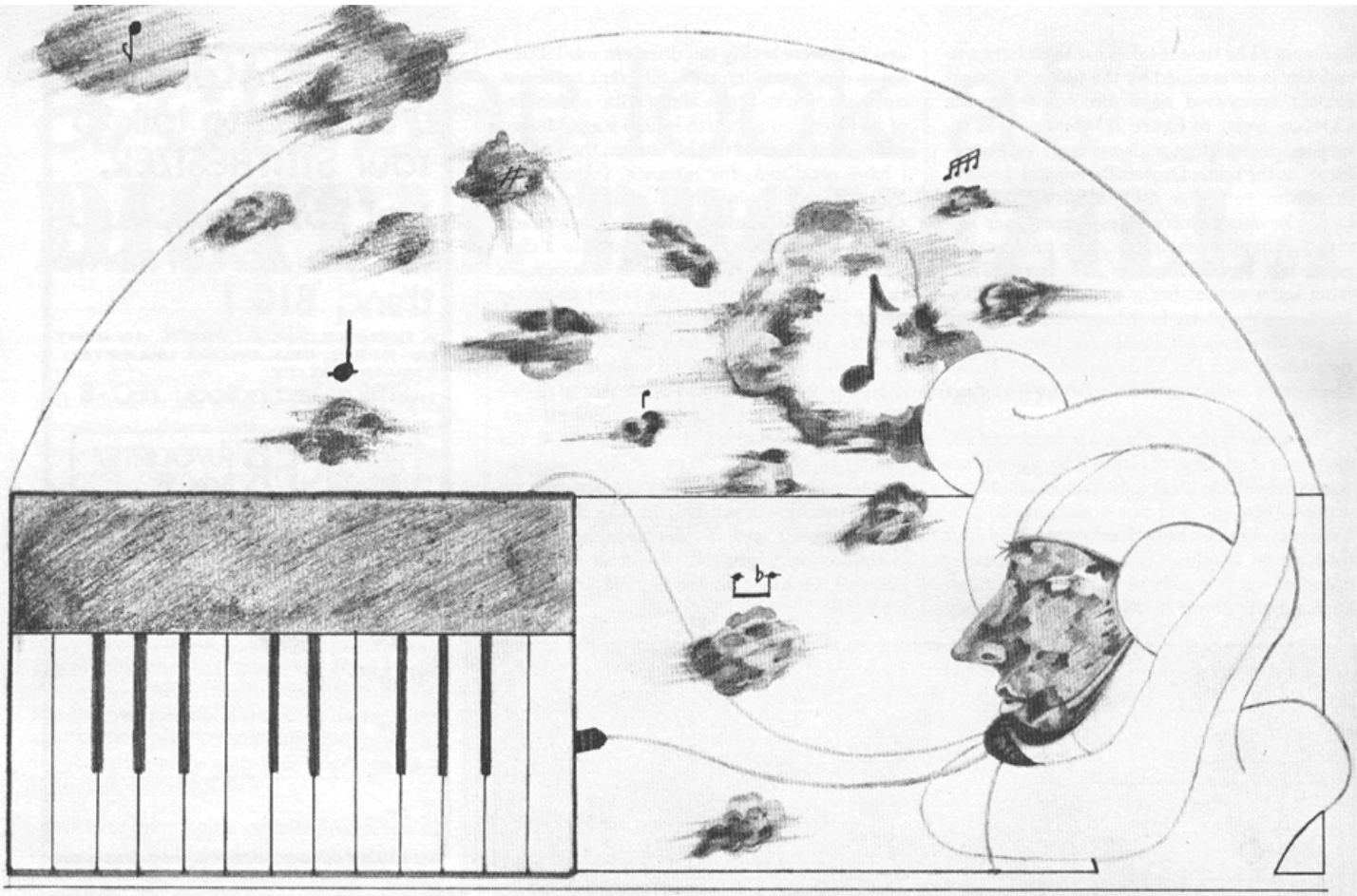


ILLUSTRATIONS BY GEORGE SANDERS (SANDO)

ANALOG PROGRAMMING:
Alternatives to the One Note Organ

by Eric Valinsky

Small keyboard synthesizers such as the Minimoog are useful to the performing musician in that they make instantly available the most common electronic sounds. They are nevertheless limited in flexibility. The synthesizer is pre-programmed such that the depression of a key instigates an event (note) whose pitch and other parameters are controlled directly from the key. Thus, a synthesizer of this type is basically a keyboard instrument capable of producing only one event at a time—a One-Note Organ! Only large modular synthesizer systems allow what is called *analog programming*.

Analog programming is achieved through the production and routing of two classes of *programming voltages*, namely DC *control voltages* and *timing voltages*. The former are usually positive voltages, static or slowly varying, which are used to set the operating parameters of the synthesizer system (e.g. frequency of an oscillator, gain of an amplifier, etc.).

In addition, two types of timing voltages are generally available. A *trigger pulse*, or simply *trigger*, is a rapid rise then fall of voltage level used to initiate events or, as will be seen later, to change the state of the system. The second type of timing voltage is the *gate signal*, or *gate*, a rapid rise to a high voltage level which is sustained for a certain time period (the *on-time*), followed by a sharp drop in voltage to ground (0 volts). Gate signals are also used to initiate events, but will also impart duration to the event corresponding to the gate's on-time. Graphical representations of these programming voltages are shown in figure one.

Programming voltages are generated by numerous modules within the synthesizer system, including keyboards, sub-audio oscillators, and sequencers. An example of a *programming voltage generator* (PUG) which will be used later is the envelope follower. This module produces a dc control voltage output proportional to the amplitude of the audio signal input. In addition, if the input signal, hence dc voltage, exceeds a set threshold level, a trigger and a gate can be produced. The gate is on as long as the threshold is exceeded. Thus the envelope follower produces timing as well as control voltages. The output behavior of the envelope follower is summarized graphically in figure two.

Many types and variations of programming voltage generators exist. In fact, each synthesizer system has unique methods of generating and utilizing programming voltages. It is therefore beyond the scope of this article to provide an overview of analog programming techniques. Instead, a few general programming techniques will be explored through a progression of examples. First, however, one additional programming concept will be explained.

An event is an undefined term in music, but it has at least three properties. The first is its character, which enables a listener to distinguish the event from another event, or from the continuum of non-events, another undefinable. Secondly, it has a starting time, and finally, a duration. A device or collec-

tion of modules which produces an event is called an *event generator*. (EG). It is with the interaction of event generators with each other and the outside world that analog programming is concerned.

What exactly constitutes a musical event is a point of disagreement among composers. However, the event generator concept is independent of event criteria. Thus, an event generator generates an event, no matter what is considered an event.

Examples of Analog Programming

Example One: Direct Programming

For the first example, we will return to the One-Note Organ, for which a possible configuration is given in figure three. As mentioned earlier, a single note (event) at a time is produced, whose pitch, timing and duration is controlled by the keyboard. The system structure consists of one event generator controlled by a keyboard and is depicted in figure four, a simple and straightforward configuration.

Note the blocks within the rectangle labeled "event generator." The amplifier stage represents the modules used to initiate and terminate the event, which could be a VCF or the on or off state of the power switch, as well as a conventional VCA controlled by an envelope generator as in the case of our poor inflexible synthesizer. Notice also, in this example, that certain modules are always on: the oscillator is perpetually oscillating, the filter is continually filtering. However, the event itself will only be on (i.e. heard) when the final amplifier stage is on. As will be seen later, we can obtain control signals from the continuously operating portion of the generator which are independent of the on-off state of the event.

Example Two: A Simple Program Loop

A sequencer can be used to generate a *program loop*, as shown in figure five. The sequencer is clocked slowly and provides triggers which initiate EG I and EG II when the corresponding sequencer stage is reached. The recurring cycle of events which results forms a simple, uncontrolled program loop. It is assumed in this and the following examples that the durations of the events are independent of the on-times of the corresponding sequencer stages. In addition, more than two event generators can be used, up to and including the number of sequencer stages.

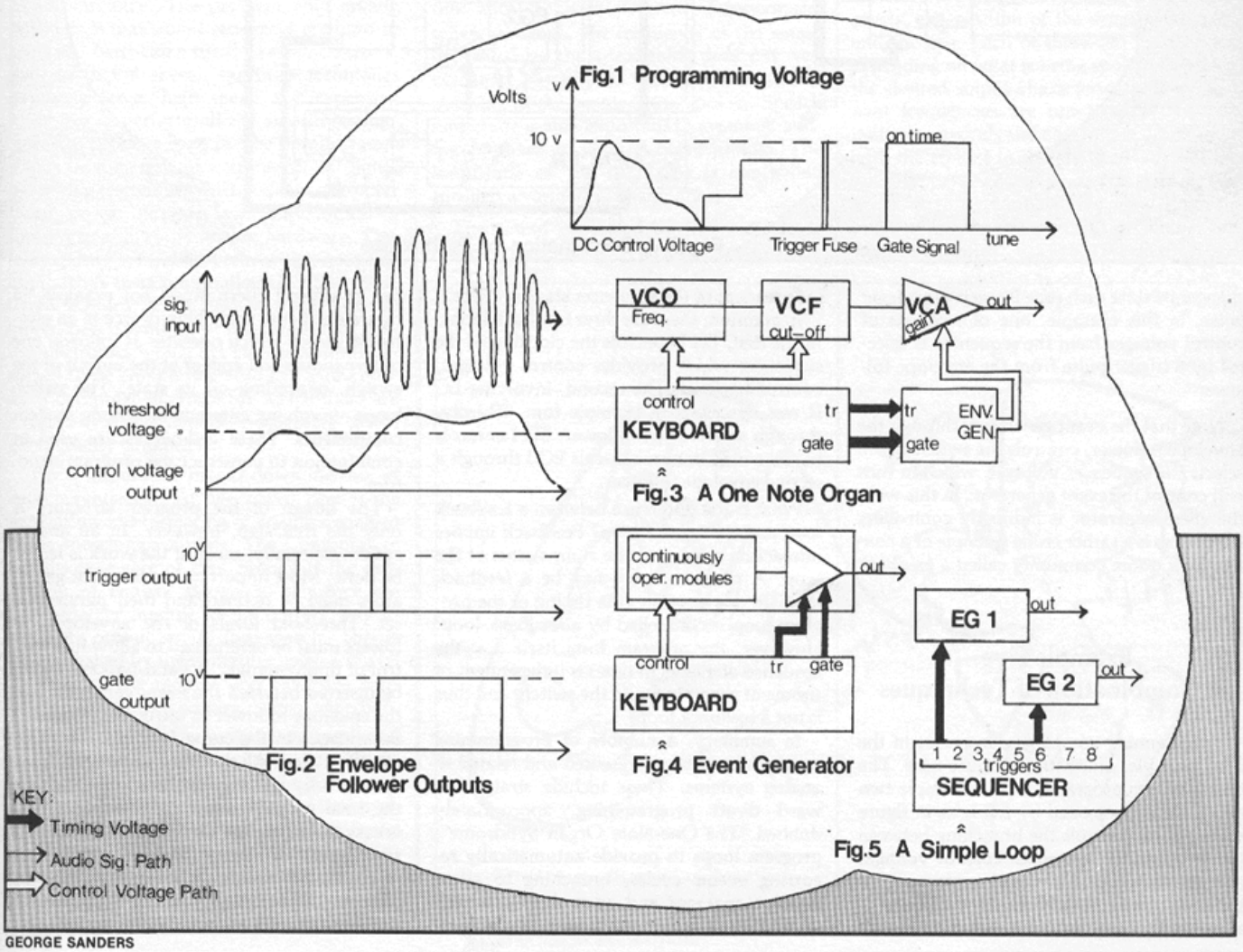
Example Three: Controlling the Program Loop

In figure six, an envelope follower processes the continuous signal from each event generator. When the amplitude of the continuous signal from EG I exceeds the threshold level, a trigger is generated which starts the sequencer clock, thus initiating the loop. The loop is terminated by a trigger controlled by EG 11, which stops the clock.

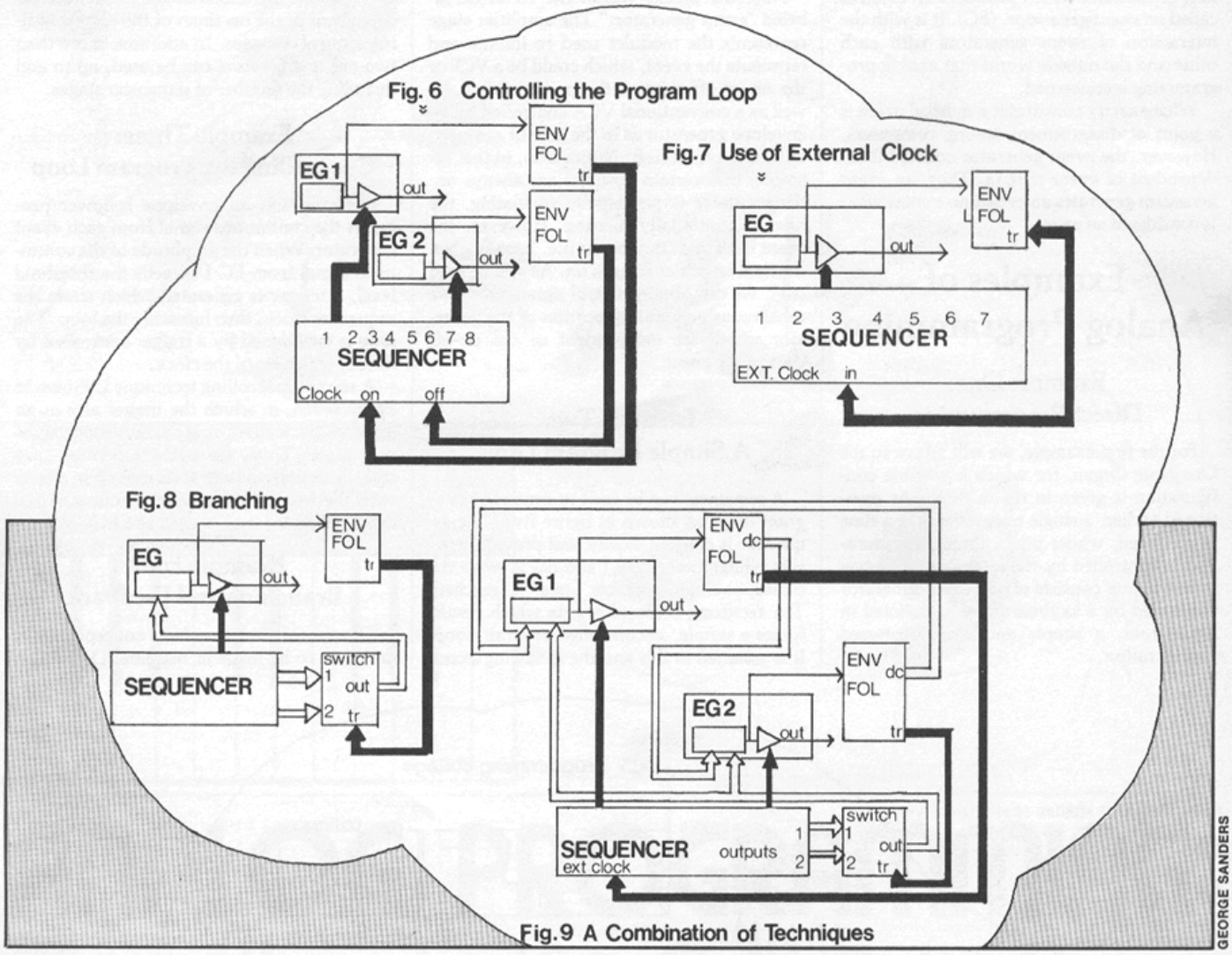
A second controlling technique is shown in figure seven, in which the trigger acts as an external clock advancing the sequencer to the next stage. Thus, the event generator controls the initiation time of its events. In effect, with the help of the envelope follower, the event generator itself is used as a PUG.

Example Four: Branching and Feedback

The use of the *branching* concept allows decisions to be made in realtime. Depending



GEORGE SANDERS



changes its state each time it receives a trigger pulse. In this example, one of two sets of control voltages from the sequencer is selected by a trigger pulse from the envelope follower.

Note that the event generator, through the envelope follower, controls the switch which selects the sequencer voltages, which in turn will control the event generator. In this way, the event generator is indirectly controlling itself. This is a rather crude example of a *control loop*, more commonly called a *feedback loop*.

Example Five: A Combination of Techniques

Programming complexity increases in the final example, illustrated in figure nine. The timing of the program loop of example two and three is controlled by EG I, as in figure seven. EG II controls the branching between the two sets of sequencer control voltages as in example four. The use of sequencer control voltages is nontrivial, since again it is assumed that the durations of the events are independent of the sequencer stage on-time.

In addition, there are three feedback loops. In the first, EG I controls the clocking of the sequencer which provides control voltages, controlling EG I. The second, involving EG II was described in example four. Thirdly, through an envelope follower, EG I controls EG II, which in turn controls EG I through a second envelope follower.

What is the difference between a feedback loop and a program loop? Feedback implies interaction among all the components of the loop. A program loop may be a feedback loop. In this example, the timing of the program loop is controlled by a feedback loop. However, the program loop itself (i.e. the sequence of events in time) is independent of the event generators and the switch, and thus is not a feedback loop.

In summary, a number of programming concepts have been presented and related to analog systems. These include straightforward direct programming, appropriately dubbed "The One-Note Organ Syndrome"; program loops to provide automatically recurring events cycles; branching to allow decision making; and control or feedback on its state, a system can automatically select one of several alternatives. For example, in figure eight, the branching device is an electronic switch which operates as follows: one of two inputs will appear at the output of the switch, depending on its state. The switch loops, involving interaction among system components. These techniques are used in combination to construct the program structure.

The design of the program structure is only the first step, however. In an analog system especially, most of the work is left to be done. Most importantly, the event generators must be defined and their parameters set. Threshold levels of the envelope followers must be determined to allow fine control of timing signals. A band-pass filter may be inserted between the event generator and the envelope follower to introduce frequency dependence of the control signals. This also must be tuned. All this must be done with the system fully patched and operating so that the final musical result can be selected. It is easy to lose sight of it, but it is the musical result which is most important. The beauty of analog programming lies in the ability to enhance the complexity of the final musical result. ⌐∿∿⌐

VOCAL SYNTHESIS

by Peter Hillen
consultant, National Semiconductor

An article on computer generated speech wouldn't seem right without a reference to Star Trek or 1984. That was it. The fact is that computer speech is here today and is in reach of electronic music performers both from an availability and a cost stand point.

The first speech synthesizers were a mechanical replica of the vocal tract. Soon after followed the electronic synthesizers using analog circuitry. The problem with analog synthesis is that sound sequences are hard to generate. Next came the digital synthesizers. Entirely digital speech synthesis techniques require a large, high speed and expensive computer to perform all the algorithms necessary to produce speech. The breakthrough comes in the form of a trade off in digital vs. analog technique which reduces computer speed, power and cost with the addition of some special purpose analog hardware. One such speech synthesis sytem which uses this approach is the Computalker (PO Box 1951, Santa Monica, Calif. 90406). The Computalker speech synthesizer is a card full of analog electronics which fits into computers designed to be compatible with the S-100 hobbyist computer interface for which there are a number of suppliers of inexpensive ($500) good quality computers.

The Computalker breaks down the task of speech synthesis into two parts. First is the generation and formating of analog signals to make speech-like sounds. The Computalker circuits take care of this. Second is the generation of control signals to sequence the Computalker module in the correct way to make its output sound like speech. This is accomplished by the computer using a digital interface to the module and extensive software to follow the complex rules of speech.

Let us examine how we humans speak and break the contributing functions into blocks which the computer can control. For reference, a schematic of the whole vocal synthesis system is shown in Figure 1. It is the electronic analog of the vocal tract which starts at the vocal cords and ends at the lips. As we go through the derivation you will find that the electronic counterparts of our vocal tract are made up of the very familiar electronic music functions of voltage control.

First, the vocal cords or vocal folds. The vocal folds open and shut one end of the vocal tract interrupting the flow of air from the lungs. These short bursts of air serve as the oscillator in our vocal system. The form of these waves looks like a half wave rectified sine wave as is shown in Figure 2. The electronic analog of the vocal folds is a variable frequency oscillator. It is variable because the frequency at which each of us use our vocal folds is different. Furthermore, when we speak, the frequency of the sound generated by the vocal folds does not stay constant. Its variation is used to increase the meaning of the words being spoken. Speech amplitude is also important to meaning, adding emphasis to what is being spoken. The amplitude of the oscillator is controlled through a variable gain amplifier.

The sound generated by the vocal folds makes its way up the vocal tract to the lips. The vocal tract can be treated as a straight tube. From physics, it is known that a characteristic of a tube such as this is to be resonant at odd multiples of the wave length of the tube. From the analysis of speech, it has been found that only the first three of these resonances need be considered when developing a model for speech. If you have looked into your mouth lately, you may have noticed it doesn't look like a straight tube. All along the way, there are things which can change the characteristics of the sound. Some to consider are the shape of the mouth, the nasal cavity, the position of the tongue, the teeth and the lips. Each of these can be modified depending on what is being spoken to deliver the desired sound characteristics. The resonant frequencies are not fixed at the odd multiple intervals and can vary in accordance with the change in the physical shape of the tract. These resonances are simulated in Figure 1 as three variable frequency filters in series with the vocal cord oscillator and amplitude control. This path is called the oral

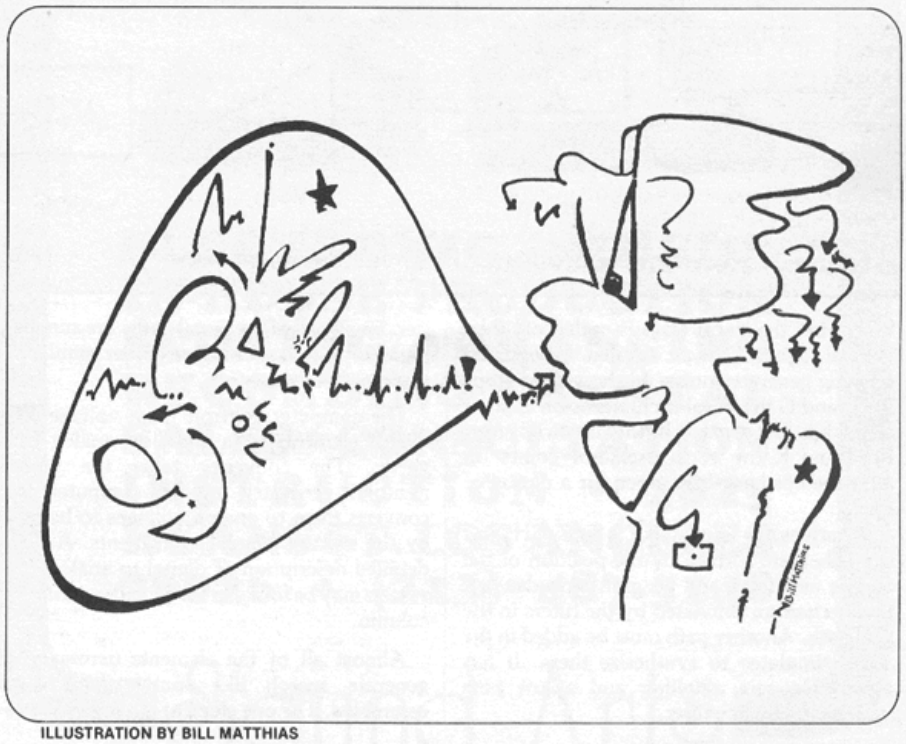


ILLUSTRATION BY BILL MATTHIAS