Superseque: A Full Featured Analog Sequencer

by: Thomas Henry

Thomas Henry is a contributor to several magazines including Radio-Electronics, Compute!, Keyboard, *and* Run. *He played guitar professionally for ten years in order to put himself through school; currently he is an Assistant Professor of Computer Science at Mankato State University in Minnesota, and is wrapping up an implementation of the FORTH language that has extensive editing, graphics, and error-detection routines. Aside from computers and electronics, Thomas has a passion for Victorian literature, Sherlock Holmes, and the works of Oscar Wilde.*

Sequencers have been with us from the earliest days of voltage-controlled synthesizers and have always held an important position in the studio. Sequencers are especially useful nowadays since much modern music depends on intricate or rapidly changing rhythmic events.

With the advent of inexpensive personal computers, digital sequencers are now plentiful and cost less than the old analog types. So why build an analog sequencer? There are several reasons. First, Superseque is low-cost and easy to build; you don't have to write any software for it . . . just throw together a bunch of common parts. Second, due to the influx of MIDI equipment there are a lot of analog synthesizers being "dumped" for very low prices, and Superseque will work with any of these devices that follow a 1V/octave control voltage standard. Finally, if you currently use a non-computer controlled synthesizer and don't want to adapt it to computer operation, then an analog sequencer is the best way to go. (Of course, if your system already includes some "computer-ready" circuitry such as scanning keyboards, digital-to-analog converters, and the like, then you would best be advised to pursue a computer generated sequencer that interfaces with these devices.)

Introducing Superseque. This project is called Superseque because it includes just about every generalized feature you could possibly want in an analog sequencer. Superseque is a sixteen stage, dual channel sequencer with onboard control voltage summing. It also provides individual gate outputs for each stage and a master precision trigger output. Additional features include a programmable length input, manual and trigger reset inputs, manual and trigger clock inputs, and the capability to generate one-shot, "circulate," or automatic reversing patterns. Just about every feature is brought out to a jack, control or switch so that it is possible to completely reconfigure Superseque simply by arranging patch cords.

Older sequencer circuits were often designed around available circuitry, not the needs of musicians. For example, how many sequencers have you seen that used the 4017 CMOS decade counter? While this is certainly an easy to find chip and is perfect for some applications, it usually leads to a dreary sequencer. To create Superseque, I wrote down all of the features that I wanted as a musician and then designed a circuit to implement these features, without any preconceptions as to what components would be needed. As it turns out, the result isn't all that much more complex than a 4017 sequencer, nor is the cost that much greater. After all, let's face it—the biggest share of your money will be tied up in the pots and knobs, not the supporting circuitry!

If this sounds appealing to you, then let's get cooking and see not only how Superseque works, but how you can build it for your own system.

How Superseque works. Since this is a large circuit, the schematic has been divided into two sections, the *logic circuitry* and the *output circuitry.* You will probably want to leave the logic circuitry alone, since not a whole lot more can be added to it. On the other hand, the output circuitry may be modified and customized as you see fit to include more or less features. We'll talk about this in just a bit, but for now let's examine the logic circuitry which forms the heart of Superseque.

Like all sequencers, something has to count input pulses and provide appropriately related output signals; in Superseque, IC5 (see **Fig. 1**) does the counting. This IC is a CMOS 4516, "divide-by-16 synchronous binary up-down counter." Don't let the words fool you; this is just a fancy way of saying that the chip can count up to sixteen events in either direction.

Notice that there are four output lines on the chip (pins 2, 14, 11, and 6). These lines output a number from 0000 to 1111 in binary format. Since a binary number is difficult to use directly for sequencer control, IC6 takes on the job of transforming the binary number into something more convenient. IC6 is a CMOS 4514 one-of-16 decoder; input a four-bit number and one of the sixteen output lines goes high while all the others stay low. The 4516 and the 4514 work well in tandem since the one chip needs what the other has to offer!

Let's now look at some of the details. J1 is a clock input. Typically, a rectangular waveform is fed into this

". . . the sequencer can step through the various stages until the last one, where it stops . . . this is a great way to generate 'instant arpeggios' which play at superhuman speed!"

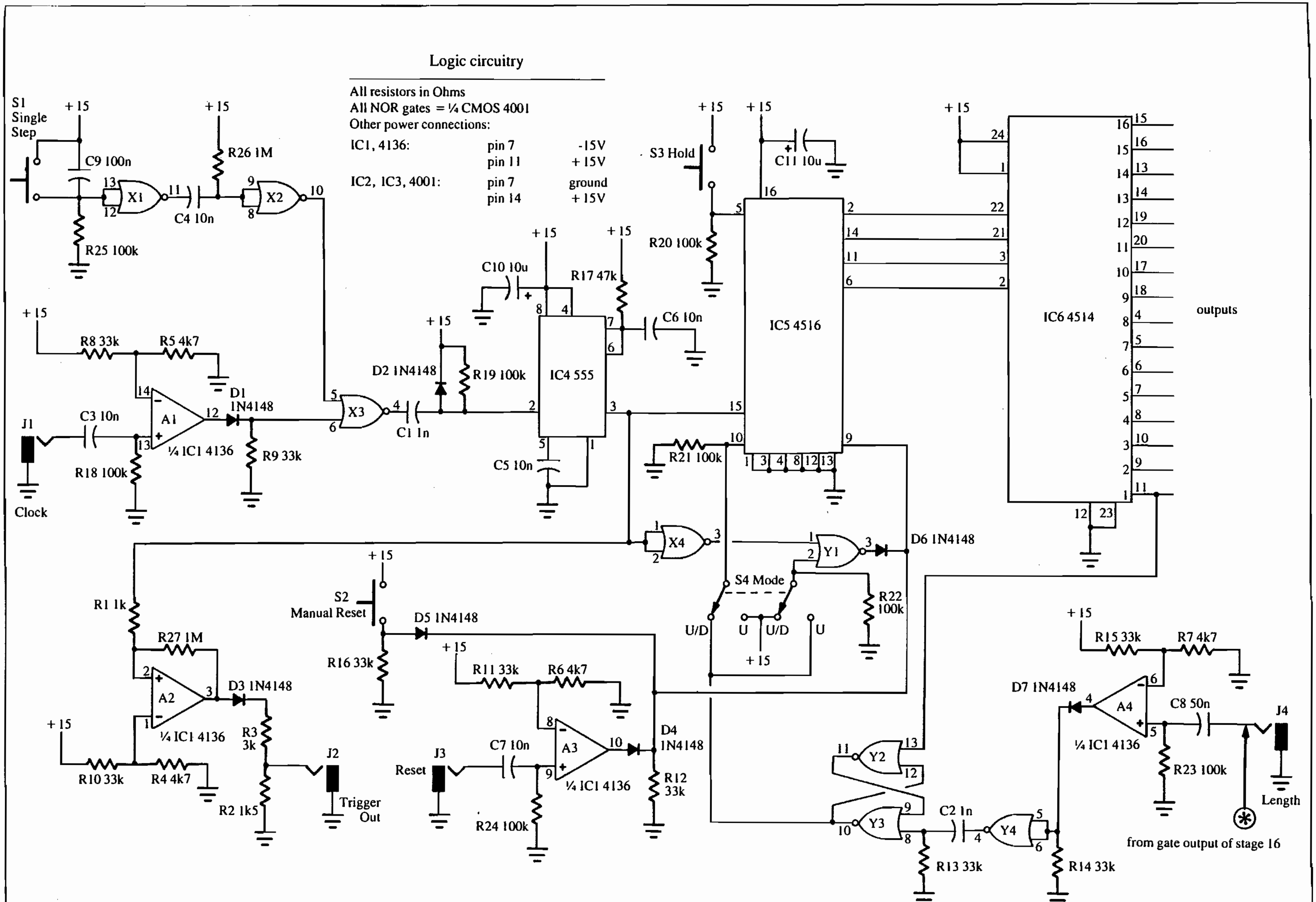

Fig. 1 Superseque schematic

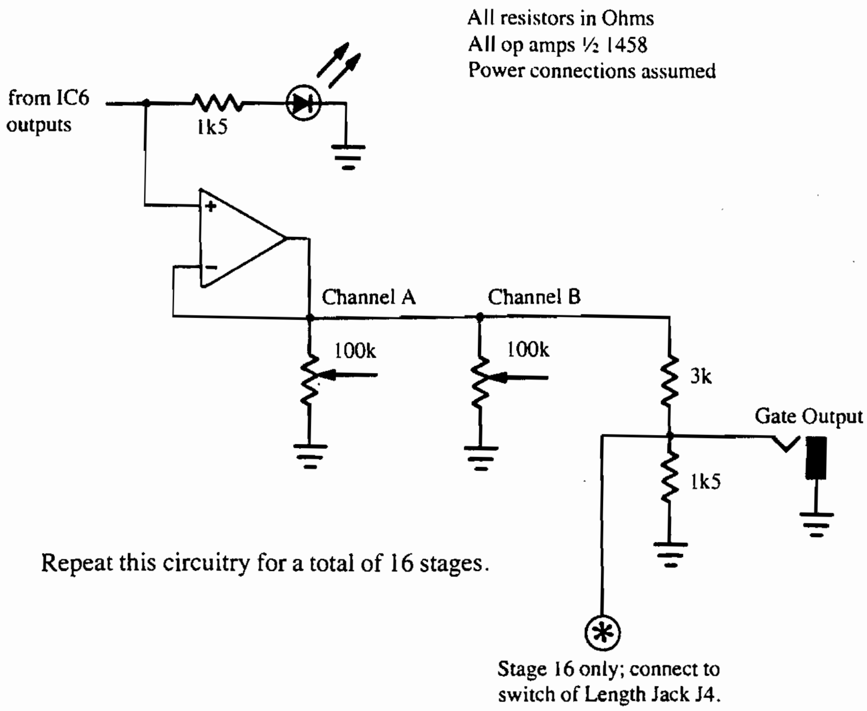

Fig. 2 Output interface circuitry

jack, perhaps from a VCO or LFO. The duty cycle of the input signal isn't important, since the differentiator composed of C3 and R18 trims the pulse width to about 1 millisecond. Likewise, R5 and R8 set a threshold for comparator A1, so that any input signal with an amplitude of two Volts or greater will fire the circuit. D1 restricts A1's output swing to positive excursions only, thus keeping negative going signals out of the subsequent CMOS circuitry.

S1 lets you single step through Superseque's 16 stages. Notice that this switch shorts out the capacitor in the RC combination of C9 and R25. This helps to eliminate the problem of contact bounce in a very simple, yet effective fashion. When S1 is pressed, the output of inverter X1 goes low and this fires the "half-monostable" comprising C4, R26 and inverter X2. The output of X2 goes high for about 10 milliseconds. A fairly long time constant is chosen here, again to minimize the effects of contact bounce back in S1.

The outputs of both the clock and single step circuitry are coupled to NOR gate X3. If either signal goes high, then the output of X3 will go low, firing the monostable composed of IC4 and associated components. IC4, which is a 555 timer, is set up to provide a 1.2 millisecond output pulse at pin 3, and this becomes the main clock signal for the rest of Superseque. During the breadboard stage of Superseque, trial and error demonstrated that the 4516 counter chip is very fussy about what sort of waveform clocks it. The purpose of the 555 is to provide a clean clock signal, whether the origin was a single step pulse or a clock input at J1.

As mentioned, pin 3 of the 555 is the master clock output. It feeds pin 15 of IC5 and thus steps the counter along. The clock also goes to comparator A2 and associated circuitry. A2 trims the voltage swing of the clock to a 0V to +5V range and this signal is then presented to J2 as a trigger output. This output can be used to fire envelope generators, synchronize drum units, or tie in other sequencers for tandem operation.

So far, all we have seen is the drab part of the design which any circuit must have to be called a sequencer. Let's skip ahead now to the reset section, for this is where the fun begins. NOR gates Y2 and Y3 are cross-coupled to form an R-S flip-flop. The first output stage of the decoder (at pin 11 of the 4514) is sent to the set input of the flip-flop, pin 13 of Y2. When stage 1 goes high, at the start of a sequence, the flip-flop is set and so the Q output, at pin 10 of Y3, goes high. Hang on to that notion for a minute, while we backtrack a bit and see what happens at the reset input of the flip-flop.

J4 is Superseque's "sequence length" input. Patching a cord from one of the sequencer outputs to this input programs the sequence length. Notice that J4 is a closed circuit type jack; with no cord plugged in, the circuit defaults to the maximum length of sixteen steps. As usual, A4 and associated circuitry shape the input signal up into something more usable by the CMOS chips (compare this to the clock input circuitry surrounding A1). The output of comparator A4 is inverted by Y4, and then the signal is capacitively coupled via C2 to the flip-flop's reset input. Capacitive coupling is used here since various timing considerations dictate that the flip-flop respond to an edge, not a level.

Okay, stage one of the sequencer sets the flip-flop, while the last stage (determined by the length input, J4) resets it. Now suppose that S4, the mode switch, is in the up/down position (as it is on the schematic). In this case, pin 2 of Y1 is at +15 Volts and this guarantees that the output, at pin 3, is low. This in turn holds the reset input of the 4516, pin 9, at ground throughout. The net effect is that the reset pin of the 4516 plays no part in the operation when in the up/down mode.

But look at the other half of DPDT switch S4. The output of the flip-flop is now directly coupled to pin 10 of the 4516. Pin 10 is the up/down selector for this chip. When this input is high, the counter counts in an upward direction; when it is low, the counter counts downward. We now have all of the pieces necessary to put the puzzle together. When the sequencer is turned on, the counter will count upward and when the last stage is hit, the flip-flop is reset. This pulls pin 10 of the 4516 low and the counter now reverses direction and counts downward. Then stage one is hit, the flip-flop changes state again and the counter counts back up. Thus we have arrived at means for generating an up/down mode of operation.

With S4 in the up mode position, pin 10 of the 4516 is pulled high, and this guarantees that the counter will count only in the up direction. Notice too that the flip-flop is now coupled to the reset pin of IC5, at pin 9. You shouldn't have too much trouble convincing yourself that the counter will now count up until it hits the highest stage, then resets to stage one and starts over again. Incidentally, by using the flip-flop composed of Y2 and Y3, and by making the reset inpu

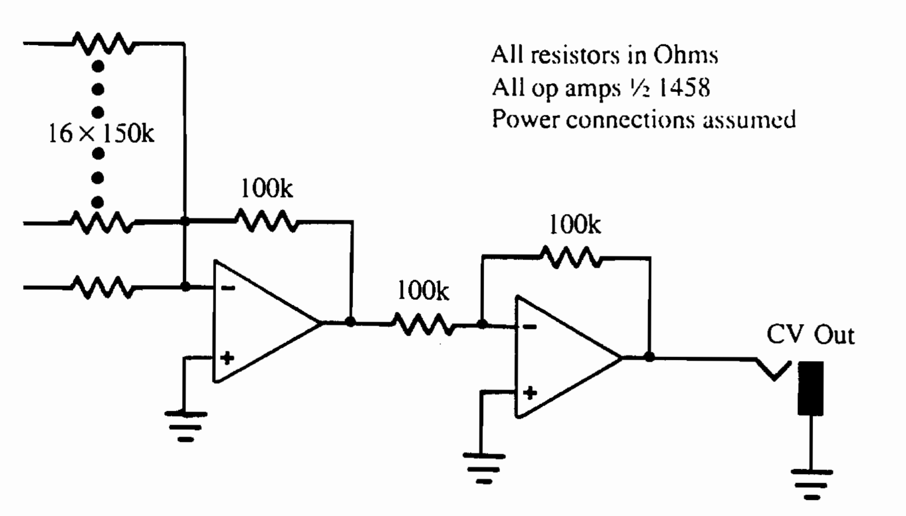

Fig. 3 Control voltage summer circuitry

edge sensitive, any unusual timing anomalies are completely avoided. Every stage is on for an equal amount of time since the reset signal is recognized only at the time of the *next* clock pulse. This is a small, but important, point and took over half of the design time to work out. The net effect is that the sequencer works like you would expect it to; there are no strange effects to hamper your musical feelings.

The entire sequencer can be reset by an external pulse via jack J3. Likewise a reset can be forced by pressing switch S2. Notice that diodes D4, D5 and D6, in conjunction with R12, form an OR gate so that any one of these three signals can reset the counter, IC5, at pin 9. This is an instance of "Mickey-Mouse Logic" as detailed by Don Lancaster in the *CMOS Cookbook* (Indianapolis: Howard W. Sams, 1977). Despite the weird name, Mickey-Mouse Logic works very well in this situation and keeps the IC package count down.

S3 is a hold switch; press it and the counter stops counting, release it and the count resumes. This comes in handy when adjusting the sequencer, since it gives you a way to stop the action momentarily.

Well, that pretty much covers the logic circuitry of Superseque. Although most of the functions should be clear from the schematic, it probably isn't at all obvious how to best use them in a musical setting. We'll cover that in just a bit, but let's first take a quick look at the output circuitry.

Interfacing Superseque to the real world. Since CMOS chips are fairly sensitive to outside interference, I decided to completely isolate them with op amps. Besides beefing up the current drive, using op amps means that you won't have to worry about shorting outputs or hooking two outputs together accidentally. Again, this is just part of good musical engineering, ensuring that the performer can concentrate on music rather than worrying about things at a circuit or component level.

Refer to **Fig. 2**, which shows the interface circuitry for one of IC6's sixteen outputs. As shown in the schematic, the output is buffered by an op amp then drives two pots and a gate jack; but feel free to alter this arrangement as you see fit. Two channels (and hence two pots per stage) seem about right, for then you can use one potentiometer to set the pitch of musical notes, while the second can be used to set the duration of the notes. The gate output, of course, can be used to fire envelope generators or any other type of time related circuitry. And by the way, to help you keep track of things, notice that an LED is hooked up to the output of each stage as well. The blinking light patterns are very inspiring!

After isolating each stage and hooking up the various LEDs, pots and jacks, you should have quite an array of controls. But we're still not done yet. Refer to **Fig. 3**, which illustrates the control voltage summing network. The wiper of each pot from **Fig. 2** is sent to this summer via a 150K resistor. If you used two pots for each stage (as recommended), then you will need two of these summer circuits, one for Channel A and another for Channel B. Once the voltages from each stage are summed into the summer, the result is inverted by one additional op amp so that the output swings in the positive quadrant (from 0V to +10V). Notice that by using 150K input resistors and a feedback resistor of 100K, the signal swing of each stage is reduced from +15V peak to +10V peak. This keeps the controls from feeling too "touchy," yet still lets you cover ten octaves if you're driving standard 1V/octave systems.

By the way, it might have occurred to you that an attenuator could be placed between the two op amps in **Fig. 3**, thus allowing you to scale the output as desired. I didn't add this feature since my synthesizer makes provisions for scaling elsewhere, but feel free to add it in if you think you need it. Likewise, you might want to add a pot which feeds in a variable offset voltage to the second op amp; this would allow for setting the range of the output.

Incidentally, like all other circuits presented in "Practical Circuitry" the inputs of Superseque have 100K impedances while the outputs have 1K impedances. Gate and trigger voltages swing from 0V to +5V, and the control voltage output covers a 0V to +10V range. These are all standard values for most synthesizers.

Finally, as a small point of interest, I tried using an inverted power supply for the logic circuitry (−15V and ground) so that the extra op amp required to invert the control voltages could be eliminated. Things didn't work out pleasantly at all! While I did get the circuit working with an inverted supply, other ephemeral problems popped up which made it not worth the effort. Anyway, op amps are cheap so all of the extra work simply didn't justify the savings. By the way, you can use ordinary 1458 dual op amps for the control voltage circuitry since you don't have to worry about bandwidth, noise, slew rate, or other critical parameters.

Building Superseque. It sure took a lot of fast talking to explain how Superseque works, even though the basic circuit action is fairly straightforward. Let's switch gears now and see how to build it.

Superseque's complete parts list is at the end of this article; note that the parts are all easy to find and not too expensive. I picked up the CMOS chips from JDR Microdevices (1224 S. Bascom Ave., San Jose, CA 95128) and the pots and jacks from PGS Elec-

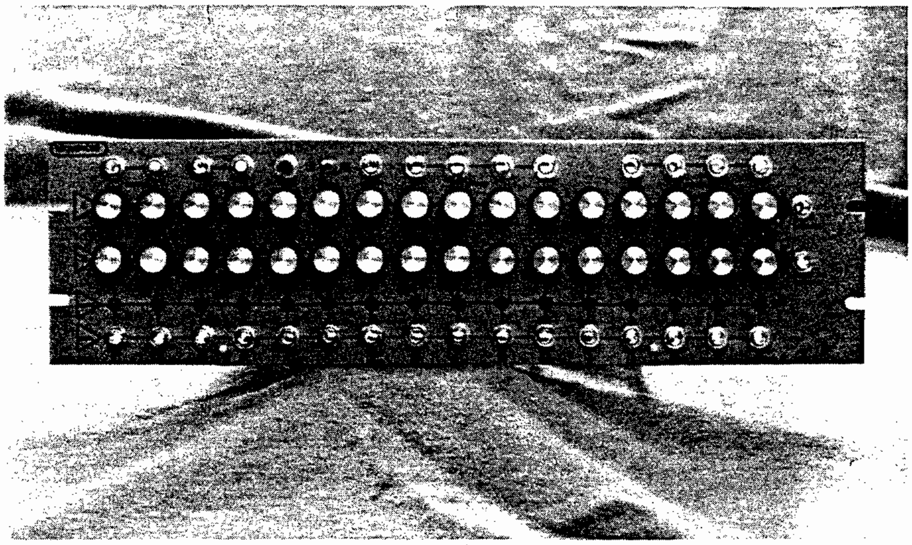

Fig. 4 Superseque front panel

tronics (Route 25, Box 304, Terre Haute, IN 47802) in case you're wondering.

To help keep point-to-point wiring down to a minimum, printed circuit boards were used in the construction of Superseque. (Don't kid yourself too much, though; no matter how you build this circuit, there's going to be a lot of wiring!) One circuit board contained the entire logic portion of Superseque, while two more boards, identical to each other, covered the output control voltage circuitry. However, don't feel that you have to go with printed circuit boards, since there really aren't any critical features in Superseque. As a matter of fact, in many ways wire wrapping may be the best way to go.

The entire circuit was mounted behind a standard 5¼" by 19" rack panel. This requires a bit of planning to squeeze in all of those pots, LEDs, and jacks, but I found that things worked so well that I even had room for a 1 by 4 multiple! Although the panel is very tightly packed, it is still quite attractive and easy to use. Refer to **Fig. 4** and **Fig. 5** which show the front and back, respectively, of the prototype.

Now, here's a bit of advice. Don't attempt to build this project in one sitting. Although Superseque is not really a high-tech circuit, it is very repetitious. When you find yourself getting tired, knock off for the night or take a break. Do whatever it takes to keep a clear head about you at all times and keep your patience intact. You'll be glad in the long run when the circuit works perfectly right off the bat, obviating the need for any tiresome troubleshooting! I found that I could build the circuit in several four hour sessions without going too batty.

How to use Superseque. As you may have noted while looking over the schematics, Superseque has no internal source of triggers, so you will have to provide these from an outboard VCO or LFO. My "Precision Controller Clock" circuit, which appeared in the special "E" issue of *Electronotes*, May 1984, is a good choice since this circuit was designed especially with sequencers in mind. Another good choice is any exponential VCO from your synthesizer. The reason for using a VCO with an exponential input will become clear in just a moment.

Patch the square wave output of the VCO (or Precision Controller Clock) into the clock input of Superseque, at J1. Now decide how long a sequence you want and patch a cord between the gate output of the desired stage and the length input at J4. Using no cord gives a default length of sixteen. Next, connect the control voltage output of Channel A to the 1 V/octave input of an audio VCO, and the trigger output, J2, to an envelope generator. If your envelope generator also needs a gate signal, use a paralleled signal from the

Continued on page 46

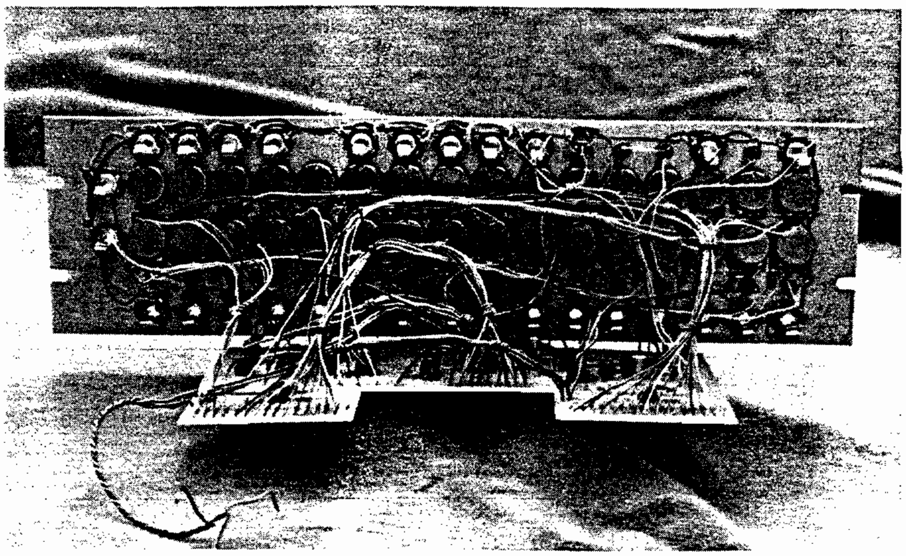

Fig. 5 Superseque (rear view)

SUPERSEQUE *from page 30*

clock input for this. Finally complete the patch with any further audio processing you might want (such as filters, phasers, VCAs and so on).

Now the control VCO sets the timing of Superseque, with each stage firing sequentially for an equal duration. Of course it would be nice to set the timing individually for each stage, and this is indeed possible using the second channel of Superseque. Simply connect a patch cord from the control voltage output of Channel B to the 1V/octave input of the control VCO. Now the setting of each potentiometer in Channel B adjusts the timing for its respective stage independently of the other stages. For an excellent treatment of this technique, refer to John Duesenberry's article, "Rhythmic Control of Analog Sequencers" (*Polyphony,* September/October 1978, pp. 26–29).

Changing S4 alters the Superseque sequence from an up/down pattern to an "up-then-reset" pattern. Notice that in either case the sequencer is in a free-running mode; however, if your VCO has a stop or gate-off mode, or if you're using the Precision Controller Clock mentioned above, you can implement a one-shot pattern with the following arrangement. Run a patch cord from the gate output of the final sequencer stage to the stop input of your VCO or controller. (Since the final stage also goes to the length input at J4, you'll have to use a multiple to split the signal in two.) Next patch a keyboard trigger to the start input of your VCO or controller and also to the reset input of Superseque. Now when you hit a key on the keyboard, the sequencer will step through the various stages until the last one, where it stops. It should be clear that this is a great way to generate "instant arpeggios" which play at superhuman speed! By the way, the reset input is used in this patch so that the pattern always starts predictably from the first stage.

There are countless other patches possible with Superseque, and I can't even begin to go into them here. You'll have to do a little experimenting on your own to find some of them, but that's half the fun of it. Suffice it to say that Superseque has many, many possibilities, so be sure to let your mind run untrammeled when thinking of new ideas for it. Along the lines of exotic patches, you might want to consider how Superseque can be tied into Micro-Drums (see "Practical Circuitry" in the August and October 1983 issues of *Polyphony*). It can be done, and in fact the flexibility is such that either Superseque *or* Micro-Drums can be the master, while the other is a slave! Anyway, you get the idea; Superseque is a super sequencer. Try it and see!

PARTS LIST

Resistors (¼ Watt, 10% tolerance)

R1	1k
R2	1k5 (1.5k)
R3	3k
R4–R7	4k7 (4.7k)
R8–R16	33k
R17	47k
R18-R25	100k
R26,R27	1M

Capacitors (15 or more working Volts)

C1,C2	1n (0.001 uF)
C3–C7	10n (0.01 uF)
C8	50n (0.05 uF)
C9	100n (0.1 uF)
C10,C11	10u
C12,C13	100u

Semiconductors

D1–D7	1N4148 or equivalent
IC1	4136 quad op amp
IC2,IC3	4001 quad NOR gate
IC4	555 timer
IC5	4516 up/down counter
IC6	4514 1–of–16 decoder

Mechanical Parts

J1–J3	¼" open circuit phone jack
J4	¼" closed circuit phone jack
S1–S3	SPST pushbutton switch
S4	DPDT toggle switch
Misc.	Wire, solder, front panel, hardware, sockets, etc.

(Note: C12 and C13 do not appear on the schematic, but should decouple the two power supply lines right where they enter the circuit board.)

If you intend to build the output circuitry for Superseque, as outlined in the text, then you will need the following additional parts:

(32)	1k5	resistor
(16)	3k	"
(32)	150k	"
(6)	100k	"
(32)	100k	linear potentiometer
(10)	1458	dual op amp
(16)	LED	

and various knobs, jacks, wire, etc. EM