60¢ ■ JULY 1972

Radio-Electronics

FOR MEN WITH IDEAS IN ELECTRONICS

For better stereo-use
TEST TAPES AND RECORDS

Build with op-amps
ACCURATE MULTIMETER

Modern radio techniques
SSB MADE EASY

Careers in electronics
TECHNICIAN IN THE FIELD

SUPERCLOCK—NEW DIGITAL TIMEKEEPER

VTR's—
Yesterday—
Today—
Tomorrow

A GERNSBACK PUBLICATION

later time zone chips; this was a feature added to the basic chip. For code 0000, the output drives the "20" on the readout. For any other code, the output is a "1" if the AM-PM in the selected zone is *different* from the input zone, and the output is a "0" if the AM-PM in the selected zone is the same. This "same-different" output can be used with an external EXCLUSIVE OR gate to change your AM-PM lights or leave them the same as the input code, giving you automatic AM-PM conversion anywhere in the world.

The "20" input to the display may be grounded instead of connected to the time zone chip if you *never* use the 2400-hour output. Otherwise, you have to rig up switching that automatically connects the "20" input on the display readout to the time zone on code 0000 and to ground on all other codes. This is easily done with the time zone selector switch. The AM-PM output is always right, even on 0000 for a 12-hour clock, but it should be switched out with a 2400-hour input on the 0000 code selection.

The readouts used do not have internal zero blanking. Transistor Q1 is used to erase the leading zero in the display for the 12-hour modes. It does so by jumping the input code into a disallowed state ("12" or "14") if a "1" is not present on the input, thus blanking the display. Q1 must be disabled on the 0000 program if you are using the 2400-hour mode of the clock, or if you simply like the zero in front.

A six-station pushbutton selector sets up the time zone code, as well as an "add-subtract" signal used in the external AM-PM circuitry. If the zone select code needs only 0, 1, or 2 ONES in its code, you can wire these directly to the needed *separate* switch terminals. If you need more than two connections per switch you have to add diodes, pointing from +5 to the select lines, to eliminate any "sneak paths" that short out the code. The Zone Expand outputs may be used with twelve or 24 buttons to get any time zone in the world instead of just six.

Construction

A printed circuit board is essential for this project. You can get one commercially, or you can etch and drill your own, using the full-size guides of Fig. 2 and the component location guide of Fig. 3. Debug the circuit as you build it, rather than assembling everything at once. Thus, you'll probably want to build up the power supply and a source of TTL 1 pps pulses before you begin. Or you can build the time base module described later in this article. Either way, be sure you have a good power supply and a valid source of input signals before you begin assembly.

Start with the resistors and bypass capacitors. Use all the excess leads for the numerous jumpers which mount exactly as shown on the component or bare side of the board. Decide whether you want 12 or 2400-hour operation and then add the five program jumpers as shown on the marked PC board. Jumpers go to the *left* for 12-hour and to the *right* for 2400-hour operation. The ten pin connectors may next be trimmed and soldered in place. Note that they project from the foil side. This is followed up by the readout sockets. Note that some types of socket are offset slightly. Unless they all "point" the same way, you will end up with either a crooked readout or a non-uniform and too-close-together spacing of one readout group.

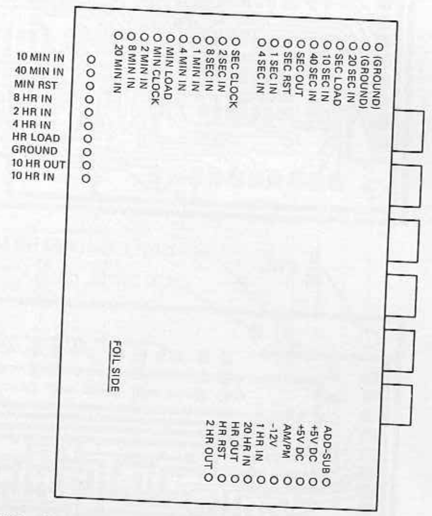


FIG. 4—CONNECTIONS on clock board mate with corresponding terminals on the board with the support circuitry.

Apply power and monitor the current. It should be zero or very low. Now remove the power and insert IC1. Reapply power and measure the voltage progressively on pins 5,9,2, and 12, while applying a 1 pps signal to the "SEC CLOCK" input. You should get a progressive division by 2,4,8, and 10, indicating that the counter is working, with the outputs changing in BCD fashion every one, two, four, eight or ten seconds. If all is well, remove the power, insert the readout (numeral "up" chip "down") in the seconds slot, and slowly bring up the supply voltage, noting the current, which should be around 100 mA. The readout should count seconds. At this point it's a good idea to check the other five readouts carefully for identical operation. Be very careful to keep the readouts right side up.

After this, you can add the ten seconds counter and readout, followed by the minutes and ten minutes counters and readouts, one at a time, and picking up only 100 mA extra per stage. Note that the MINUTE CLOCK input must be *externally* connected to either the SECONDS OUT terminal or to your pulse source.

If all is well, you can add IC5,6, and 7, and jumper the "HR OUT", and "HR RESET" terminals externally. Check the outputs with a high-speed clock to see that they are counting properly. Some *temporary* jumpers may be added from the time zone input to the time zone outputs to get the readouts to operate. You also have to temporarily short OR circuit resistors R1 and R2 and ground the collector (or short collector to emitter) on Q1. The hours counter should now work and current should be around 700 mA. You might like to check the operation of both the 12- and 2400-hour modes by changing the jumpers back and forth.

The blanking short may now be removed and Q1 added. This should erase the leading zero on the display. After this, remove all the temporary jumpers and add diodes D1 and D2, along with the selector switch. Arrange the switch and diodes to get the desired time code, (full details next month). Be sure to use diodes every time you have to use one switch contact for two purposes. Now check the -12-volt supply to see that it is the right voltage and properly applied to the PC board.

The time zone ROM chip is a reasonably rugged device, but exceptionally careless handling can damage it. It should be left in its protective carrier until ready for insertion. Then solder it in place rapidly with a small soldering *iron.* Be exceptionally careful to observe the code dot and notch, for **reverse polarity will damage this chip permanently!** Once you are certain all is well, punch the 0000 code button (or leave all the buttons *up*) and apply power.

If all is well, the circuit should behave as if the chip wasn't there, with the input time being directly displayed. Now check out the various buttons to verify that the chip is operating correctly. This should complete the assembly and debugging of your main clock board. If desired, AM-PM light-emitting-diodes (LEDs) may be added to the holes shown in the upper left corner of the board.

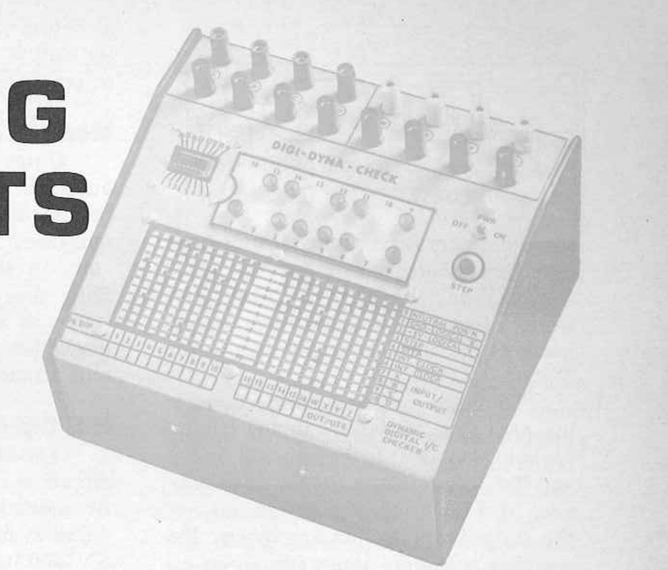
Next month, we conclude with more construction details, describe options and show how to set up the push buttons for the time zones. **R-E**

DYNA-CHECK CORRECTION

The author has called our attention to the following errors in the parts list and diagrams of the IC tester in the May issue:

Resistors R6, R7, R8, R9, R10, R12 and R14 should be 1000 ohms instead of 100 ohms as noted. The indicator lamps (LM1 through LM17) are rated at 5 volts—not 4 volts—at 50 mA. Switch S22 is not a toggle switch. It is a momentary-type push-button.

BREADBOARDING DIGITAL CIRCUITS



Experimenting is the quickest way to develop an understanding of digital logic circuits and their application. All this is simplified by breadboarding.

by JACK CAZES

THE FIRST TWO PARTS OF THIS SERIES told how to build the *Digi-Dyna-Check* digital integrated circuit tester and how to use it for dynamically checking a variety of IC types, both in- and out-of-circuit. This was done by simulating actual operating conditions: that is, making the IC "do its thing". We've seen that the matrix programming switch allows us to set up any combination of input/output conditions readily. Thus, the Digi-Dyna-Check is virtually obsolescence-proof.

Now, let's see how to use this programming capability to "breadboard" some simple digital circuits, using readily available IC's. The matrix switch will provide "jumpers", or circuit-patch connections. It will also establish input and output logic levels. The 16 monitor lamps will, as before, indicate the logic states at all points in our circuits, *simultaneously.* The input-output binding posts will be used to patch in external circuits and test equipment—scope, oscillator, or other instrument. Matrix positions 7, 8, 9, and 10 will, therefore, be used both as jumpers and as connections to the "outside world" via their binding posts A, B, C, and D.

The circuits we're going to build should serve, merely, as examples of how to breadboard with the Digi-Dyna-Check, making full use of its matrix capability. Operating power will always be supplied to the IC by connecting its ground pin to Position 1 (GND) and its V_{cc} pin to Position 2 (+5V). Matrix sliders that correspond to pins that are not used in a given circuit are left in the NEUTRAL position.

The experiments have been divided into sets, to offer relatively simple, basic computer circuits (logic gates) first, memory devices (flip-flops) next, and finally, circuits that use combinations of several flip-flops to manipulate numbers (counters and dividers).

Gates

The basic logic circuits in digital computers are known as *gates.* Their output voltage may be low or high, depending on the voltage levels at their inputs. We have examined the various input-output combinations for the common gates in the second article in this series. These are summarized, for two-input gates, in the following *truth table.* A high level is represented by a "1" and a low level by a "0".

INPUTS		OUTPUTS				
A	*B*	*NOT**	*NAND*	*AND*	*OR*	*NOR*
0	0	1	1	0	0	1
0	1	—	1	0	1	0
1	0	—	1	0	1	0
1	1	0	0	1	1	0

*NOTE: The NOT gate, or INVERTER, as it is usually called, has only one input. The output logic state is always the opposite of that which is present at its input.

These gates can all be synthesized with combinations of 2-input NAND gates. Let's see how this is done, using an SN7400 integrated circuit (Fig. 1) as our source of NAND gates. There are four 2-input NAND gates in a single SN7400. Plug this IC into the test socket of your Digi-Dyna-Check, making certain that pin 1 of the IC is in pin 1 of the socket. (There should be two unused socket positions). The internal connections for an SN7400 are in Fig. 1. The gates have been numbered for ease in referring to them later on. Provide power to the IC by moving the matrix sliders corresponding to pins 7 and 14 down to positions 1 (GND.) and 2 (+5V), respectively.

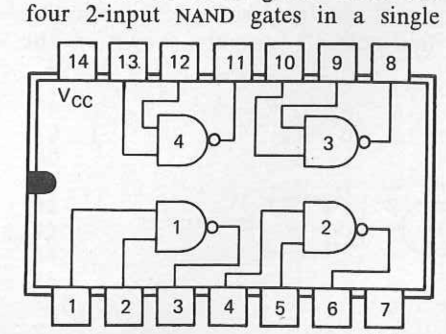


FIG. 1—IC SN7400

NOT gate

A NOT gate, or INVERTER (Fig. 2)

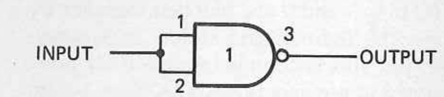


FIG. 2—*NOT* GATE (INVERTER)

can be made by merely tying the two inputs of a NAND gate together for use as a single input. The output will then always be in the opposite logic state to that of the input.

If you'd like to synthesize a NOT gate for use in an external circuit, move the matrix sliders corresponding to pins 1 and 2 to position 7, and the slider for pin 3 to position 8. (See Fig. 2.) The input to your inverter is now at binding post A and its output is at binding post B. The lamps corresponding to the input and output pins will indicate the logic levels at all times.

AND gate

An AND gate (Fig. 3) may be a NAND gate followed by an INVERTER.

Connect the output of NAND gate 1 to both inputs of NAND gate 2 by moving the matrix sliders for IC pins 3, 4, and 5 to position 7 (thereby using this matrix position as a jumper, or patch wire).

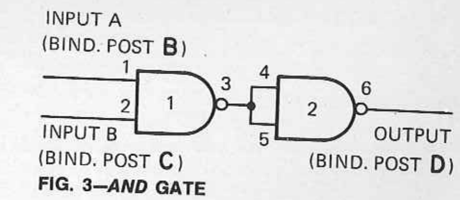


FIG. 3—*AND GATE*

Matrix sliders for IC pins 1 and 2 can now be set for each of the four input combinations shown in the *truth table* while observing the resultant output levels at the lamp corresponding to pin No. 6.

To make external connections to the AND gate, move the inputs (sliders for pins 1 and 2) to positions 8 and 9, and the output (slider for pin 6) to position 10. This brings the two inputs and the output out to binding posts. Remember that here too, even when external connections are involved, the lamps will still indicate logic levels at their locations in the circuit.

OR gate

Three NAND gates and two jumpers are required to build an OR gate. (Fig. 4).

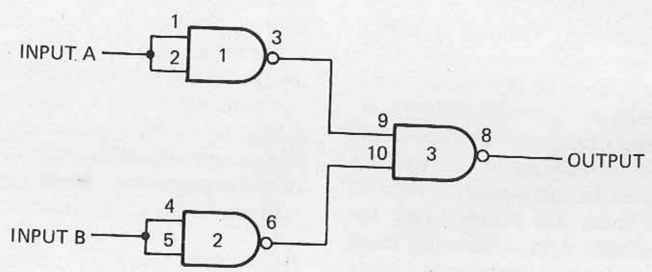


FIG. 4—*OR GATE*

IC pins 3 and 9 are patched together by moving their matrix sliders to position 7; put sliders for pins 6 and 10 at position 8 to connect these pins. Now, set up the input combinations given in the *truth table*, using the sliders for pins 1 and 2 as input A, sliders for pins 4 and 5 for input B. The lamp corresponding to pin 8 will then always indicate the output level for your OR gate.

NOR gate

The OR gate that we discussed above can be converted to a NOR gate (Fig. 5) by adding an inverter to its output. Set up an OR gate as above, and then move the sliders for pins 8, 12, and 13 to position 9. Enter input conditions as before (for the OR gate), but monitor the output with the lamp corresponding to pin 11.

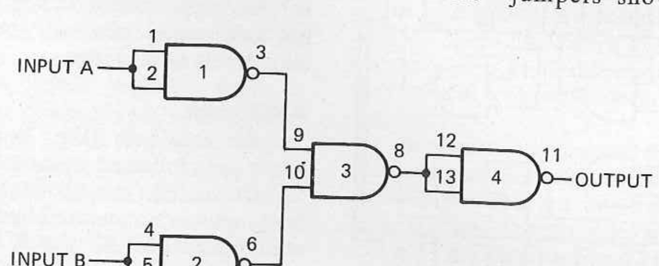


FIG. 5—*NOR GATE*

Memory circuits

Gates do not have memories; the output of a gate at any given time depends only on the status of its inputs *at that time!* The output of a bistable circuit, on the other hand, depends on both the *present* and *previous* input states, as we have already seen in our discussion of the J-K flip-flop in the second article in this series.

R-S flip-flop

The most basic bistable element, or circuit, is called an R-S flip-flop. It can be assembled with two NAND gates wired as shown in Fig. 6, using the same SN7400 integrated circuit that we used to build the basic gate circuits.

This circuit is used in your Digi-Dyna-Check as a contact "bounce" eliminator for the STEP switch. A momentary 1 to 0 input pulse entered alternately at the SET and RESET inputs will cause the R-S flip-flop to switch back

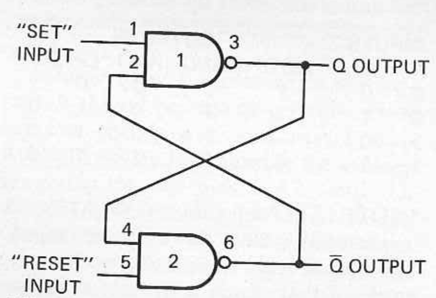


FIG. 6—R-S FLIP-FLOP

and forth, from one output state to the other, and back again.

Move the matrix sliders for pins 2 and 6 to position 7; move sliders for pins 3 and 4 to position 8. This provides the "cross" jumpers shown in the schematic. Enter a 1 to 0 pulse at one of the inputs, then at the other, either by moving its corresponding slider from position 2 (logic 1) to position 1 (logic 0), or by using matrix position 4 in conjunction with STEP button. The Q and Q̄ (not-Q) output states can then be observed at lamps corresponding to their IC pins. Notice that a given input will produce a change in the output conditions only if the outputs are in specific states prior to entering the input pulse.

R-S-T flip-flop (gated memory)

Let's add a gated input to our memory circuit (R-S flip-flop) so that it will respond to input pulses only during a specific interval of time. We can do that by adding two NAND gates to our basic R-S flip-flop. See Fig. 7.

Make the following connections:

Set Sliders For Pins	*to Position*	*Remarks*
2 and 6	7	Jumper
3 and 4	8	Jumper
1 and 8	9	Jumper
5 and 11	10	Jumper
10 and 12	3	Connects the STEP button to the CLOCK input.

Enter the SET and RESET input combinations shown below, using the matrix sliders for pins 9 and 13, with positions 1 (logic 0) and 2 (logic 1), and observe the resultant output states after a clock pulse is entered.

Inputs at time, n (before clock pulse) Reset	Set	*Output, Q, at time* n *+ 1 (after clock pulse)*
0	0	Q_n (No change)
0	1	1
1	0	0
1	1	Indeterminate

We see, here, that the output state changes only when the clock input *first* goes from 0 to 1, and *not* when the input information (reset and set) changes.

A wave shaping circuit

A novel application for an R-S *latching memory* is in converting various wave shapes to rectangular waves having very fast rise and fall times suitable for triggering digital devices such as counters, flip-flops, shift registers, etc. The input signal amplitude must be such that its maximum and minimum voltage levels are above and below the threshold level for triggering the shaping circuit; this is around 1 volt in this case.

A single NAND gate must be added to the basic R-S flip-flop, connected as an input inverter. This is done to ensure that the R and S signal inputs are always complementary, i.e., in opposite logic states.

Breadboard the wave shaper as follows using the Digi-Dyna-Check:

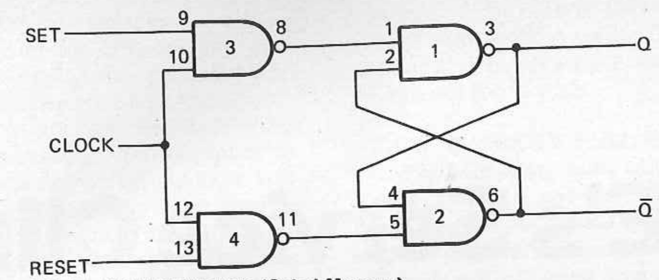


FIG. 7—R-S-T FLIP-FLOP (Gated Memory)

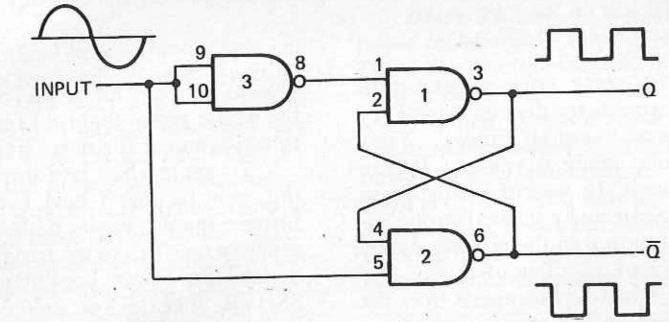


FIG. 8—WAVE-SHAPING CIRCUIT

Set Sliders For Pins	*to Position*	*Remarks*
2 and 6	7	Jumper and Q output
3 and 4	8	Jumper and Q output
1 and 8	9	Jumper
5, 9, and 10	10	Jumper and input

Connect a sine-wave generator with an output of about 3 volts peak-to-peak to the input (binding post D) and observe the complementary rectangular waves produced at the outputs (binding posts A and B). Many other input wave shapes will also work with this circuit.

Counters, shift registers, dividers

These more complex digital circuits generally consist of multiple flip-flops connected together in various configurations to perform specialized numerical manipulations. We can use our Digi-Dyna-Check to breadboard a couple of these and see how they operate:

Four-stage binary counter. A single flip-flop has two stable states, and thus is able to "count" to two. Two flip-flops connected in series with the output of the first connected to the input of the second will count to 2 × 2, or 4; three flip-flops will count to 8, and four of them will count to 16 (in binary notation, of course). Plug an SN7493 integrated circuit into the test socket, making sure that pin 1 of the IC is in pin 1 of the socket. Since this is a 14-pin IC, two socket pins will remain unused. The SN7493 (Fig. 9) contains four flip-flops, three of which are internally connected together as a three-stage binary counter. The fourth flip-flop can either be used alone or together with the others to form a four-stage binary counter, capable of assuming 16 stable states (counting to 16).

We'll set up the four-stage version:

Set sliders for pins	*to Position*	*Remarks*
10	1	Ground
5	2	+5V (V_{cc})
12 and 1	7	Jumper
14	3	STEP button, to input pulses to be counted
2 and 3	2, then to 1 then set	to clear the counter, and it up for counting.

Now, input count pulses by pressing and releasing the STEP button once for each count and observe the lamps for the outputs of flip-flops A, B, C, and D at lamps 12, 9, 8, and 11, respectively. They should change, with each input count, according to the following *truth table:*

Count	Outputs: / Lamps:	*A* 12	*B* 9	*C* 8	*D* 11
0		0	0	0	0
1			0	0	0
2		0	1	0	0
3		1	1	0	0
4		0	0	1	0
5		1	0	1	0
6		0	1	1	0
7		1	1	1	0
8		0	0	0	1
9		1	0	0	1
10		0	1	0	1
11		1	1	0	1
12		0	0	1	1
13		1	0	1	1
14		0	1	1	1
15		1	1	1	1
16 (or 0)		0	0	0	0
			etc.		

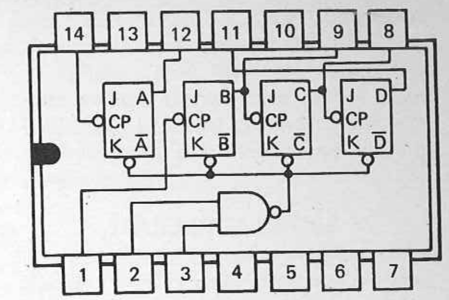


FIG. 9—PROGRAMMABLE DIVIDER

The counter can be reset to zero and made ready for starting the count again, at any time during its counting cycle, by moving the sliders for pins 2 and 3 (reset inputs) to position 2 (logic 1) and then back to position 1 (logic 0). Of course, here again, if you want to count pulses from an external source, you can move the slider for pin 14 to position 8, thereby connecting the counter input to binding post B. The external signal can now be entered at this binding post.

A programmable divider. A DM8520 integrated circuit is used in this experiment (Fig. 10). It contains

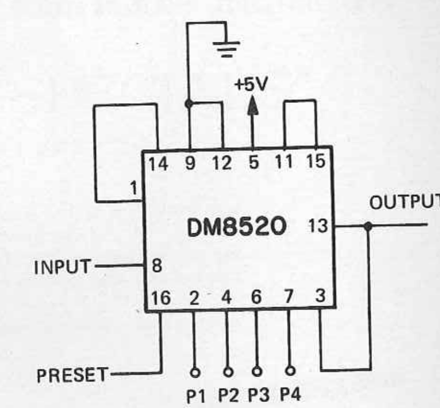


FIG. 10—PROGRAMMABLE DIVIDER

four flip-flops, internally wired together to form a 4-bit shift register. In addition to the count input and output, there are four parallel "programming" inputs and a "preset" input that can be used to preset the register to divide by any number from 2 to 15. Thus, for example, if the register is programmed to divide by ten, a single output pulse will be produced for every ten input pulses entered.

Plug a DM8520 IC into the test socket and make the necessary connections:

Set sliders for pins	*to Position*	*Remarks*
9 and 12	1	Ground
5	2	+5V (V_{cc})
11 and 15	7	Jumper
1 and 14	8	Jumper
3 and 13	9	Jumper, and output at binding post C
8	10	Input at binding post D

(continued on page 94)

BREADBOARD DIGITAL CIRCUITS
(continued from page 61)

Now set the sliders for the four program inputs, P1, P2, P3, and P4 (sliders for pins 2, 4, 6, and 7, respectively) to the combination of logic 0 and logic 1 inputs selected from the following programming table for the desired divisor:

To divide by N =	Settings P1	P2	P3	P4
2	1	1	1	0
3	1	1	0	0
4	1	0	0	0
5	0	0	0	1
6	0	0	1	0
7	0	1	0	0
8	1	0	0	1
9	0	0	1	1
10	0	1	1	0
11	1	1	0	1
12	1	0	1	0
13	0	1	0	1
14	1	0	1	1
15	0	1	1	1

Move the slider for pin 16 (preset) to position 2, then back to position 1, to enter the program. The circuit will produce one output pulse for every N input pulses.

Now that you've become proficient in programming and know how to use the matrix switch effectively, you should be able to devise and breadboard many circuits of your own, based on the increasing variety of integrated circuits on the market today. A number of them contain a multiplicity of circuits on a single chip, that can be patched together in relatively short order. A quick scan through any digital IC manual should reveal many interesting circuits, some of which perform very complex numerical manipulations. **R-E**

The Digi-Dyna-Check has literally taken off since the first article in this series appeared in the May issue. Many of you were stymied the first month because we neglected to tell you where to obtain the most essential parts. The great demand has necessitated some changes in the kit make-up as described last month. Here is the listing of kits and their sources:

The following kits of parts are available from The Electronetics Co., Inc., P.O. Box 278, Cranbury, N. J. 08512.

DDC-2, a revised programming manual listing more than 500 commonly used digital integrated-circuit devices with their pin-connection diagrams is available for $2.75.

DDC-3, a kit containing the integrated circuits used in these experiments (one each SN7400, SN7493 and DM8520) is available for $6.95 including postage and insurance.

A complete kit of all parts for building the Digi-Dyna-Check, including an improved power supply, LED lamps, programming and assembly manual and a case are available for $79.50 from MITS Corp., 2016 San Mateo Avenue N.E., Albuquerque, New Mexico 87110.