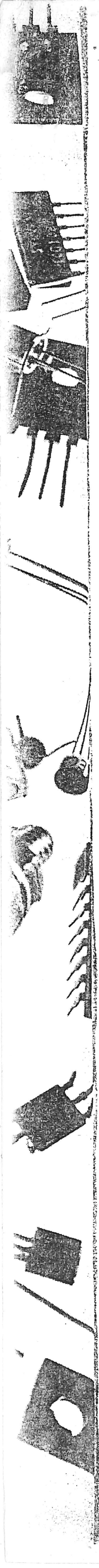
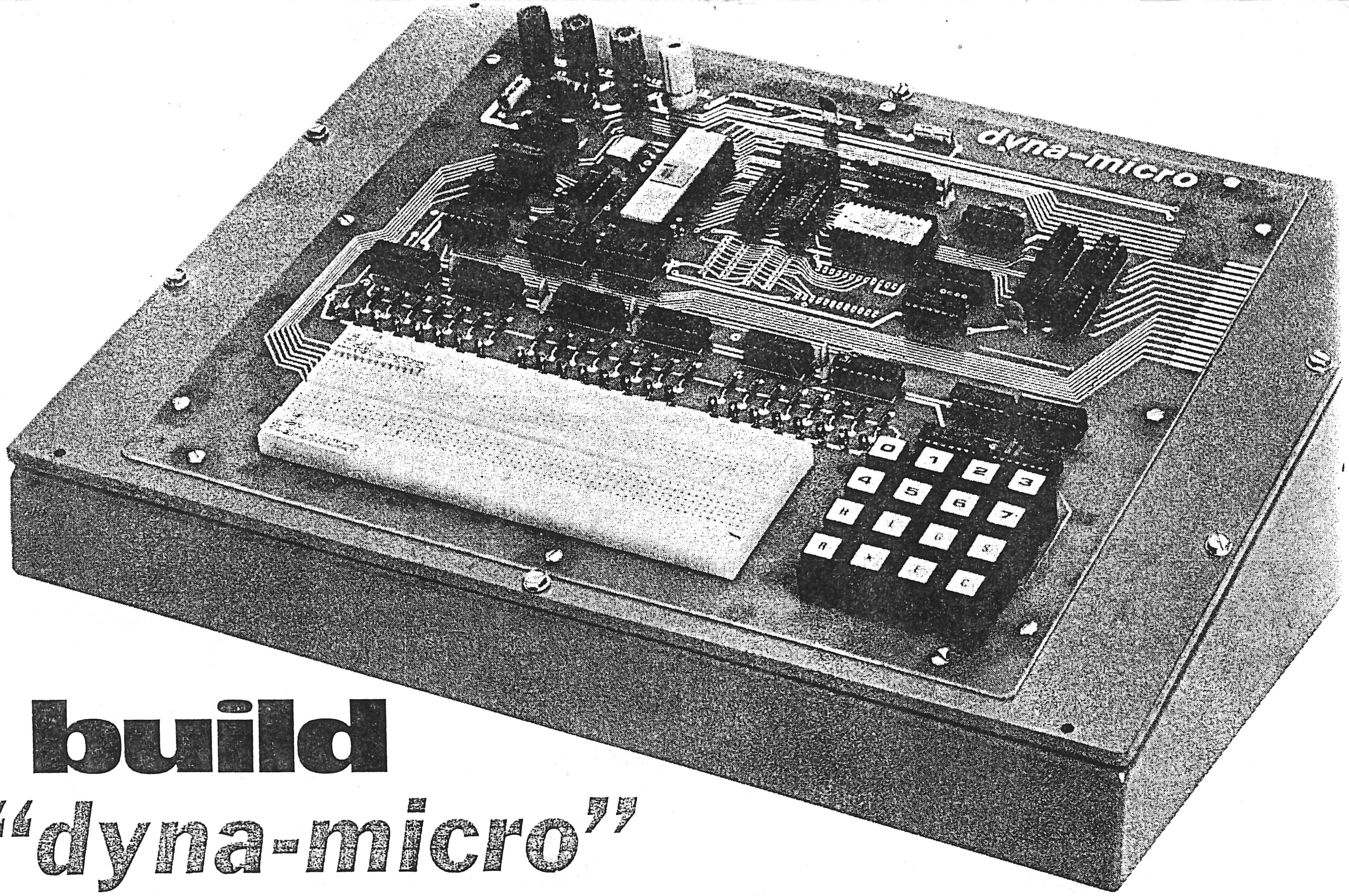




build "dyna-micro" an 8080 microcomputer

Complete with keyboard for data entry, LED readout of the address and data, breadboard socket for experimenting, 500-bytes of PROM and 500-bytes of RAM, expandable to 65K and self-contained power supply

JOHN TITUS

MANY EXPERIMENTERS, HOBBYISTS AND professionals are interested in learning about microcomputers and how they work, but the cost of a development system can be high and even inexpensive systems may require an expensive peripheral such as a teletypewriter to get them to work. None of the available systems have any easy hardware breadboarding capability and none have a series of experiments to teach you both interfacing and software fundamentals.

The Dyna-Micro is a complete microcomputer using the 8080A microprocessor chip. It isn't a stripped-down version of another system since it was designed specifically for the beginner. This system has been designed to teach you about microcomputers whether you're a high-school student or a digital design engineer—both will learn a great deal about computers. With the Dyna-Micro, you won't be spending your time wire wrapping a prototype or debugging a complex "hobby" system, you'll be doing hardware and software experiments to learn more about the microcomputer revolution.

The Dyna-Micro has many features that make it easy to use. These features include:

- Complete, software controlled "front-panel."
- Fully-encoded keyboard input containing 15 keys.
- Three, eight-bit output ports with LED indicators.
- Complete solderless breadboarding area that provides easy access to all the bus signals.
- 500-bytes of Read/Write memory (R/W) and 500-bytes of Programmable Read Only Memory (PROM).
- Edge connector provided for future memory expansion.

The Dyna-Micro won't run FORTRAN, BASIC, editors, assemblers or other complex software in its present form. It isn't meant for that. You can, however, expand the memory if you want to. By the time you finish the experiments, you will know how this is done.

All the Dyna-Micro functions, including the keyboard, are contained on a single 10" × 12" printed-circuit board. The power supply is external.

Using the Dyna-Micro

The Dyna-Micro is one of the easiest

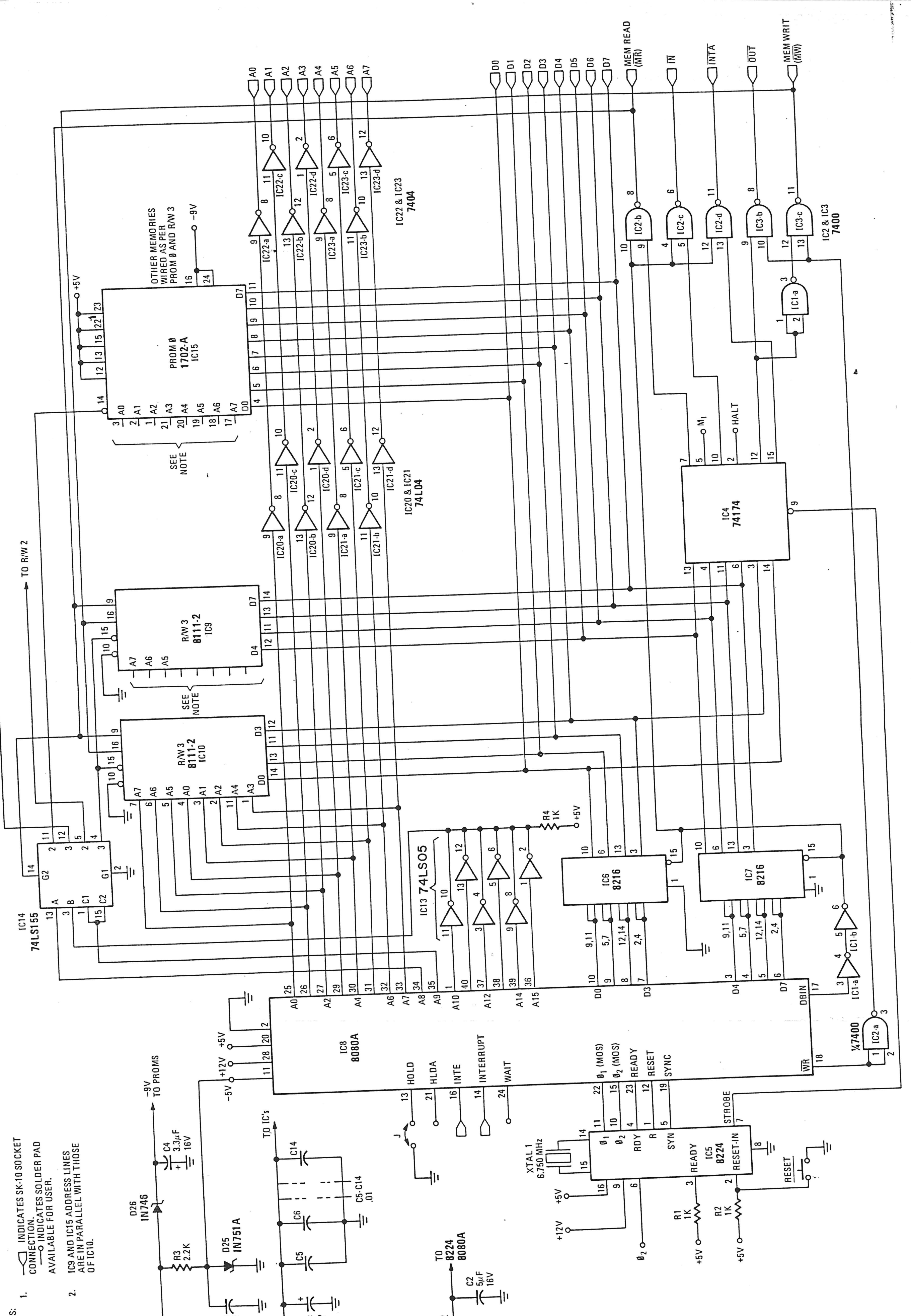


 FIG. 1—CENTRAL PROCESSOR shows 8080A microprocessor and associated components. Power supply is external while voltage regulator is on-board.

to use microcomputers since it uses a software controlled "front-panel." Data is entered through a 15-key keyboard and data is displayed by LED's. The keyboard is used to enter address and memory data, to examine the contents of any memory location and to start your program from any location desired. The software to do this is preprogrammed in a 1702A PROM and is called the Keyboard Executive or KEX program.

Since the 8080A microcomputer uses 16-bits of address and 8-bits of data, the LED display registers are divided into groups of 8-bits each. The KEX program uses OUTPUT 1 for the highest 8-bits of address (HI) and OUTPUT PORT 0 for the lowest 8-bits of address (LO). OUTPUT PORT 2 is used to display data. With 16 address-bits, up to 65,536 words (commonly referred to as 65K) of memory may be addressed and added to the basic 8080 system. The Dyna-Micro uses only 1K of memory, more than enough for all the experiments and for most of your own needs when getting started. The memory is segmented, with the first 500-bytes being PROM and with the next 500-bytes being R/W. Table 1 shows how the memory is divided between PROM and R/W.

The output ports and the keyboard input are used in conjunction with the KEX program, but since they are not

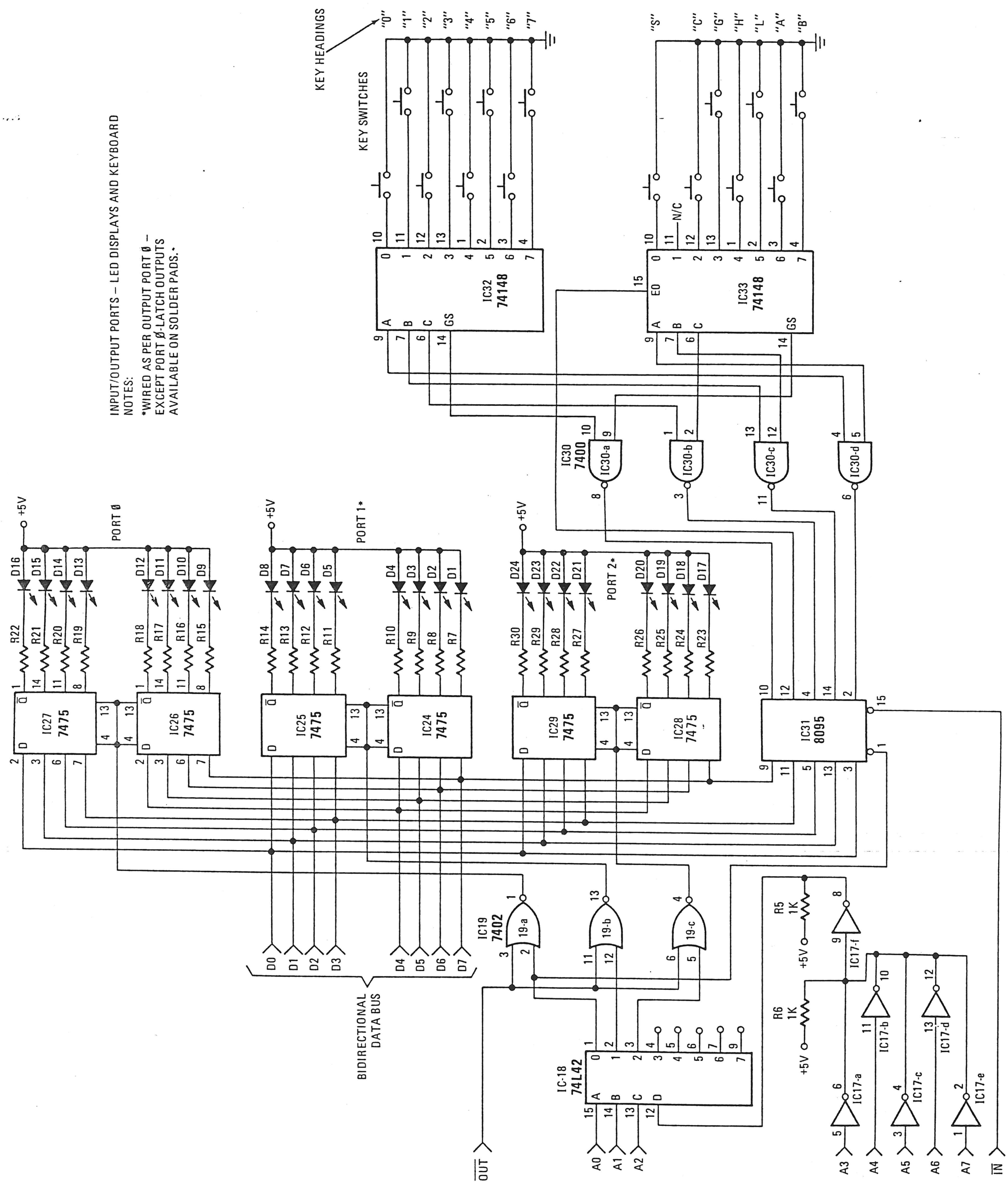
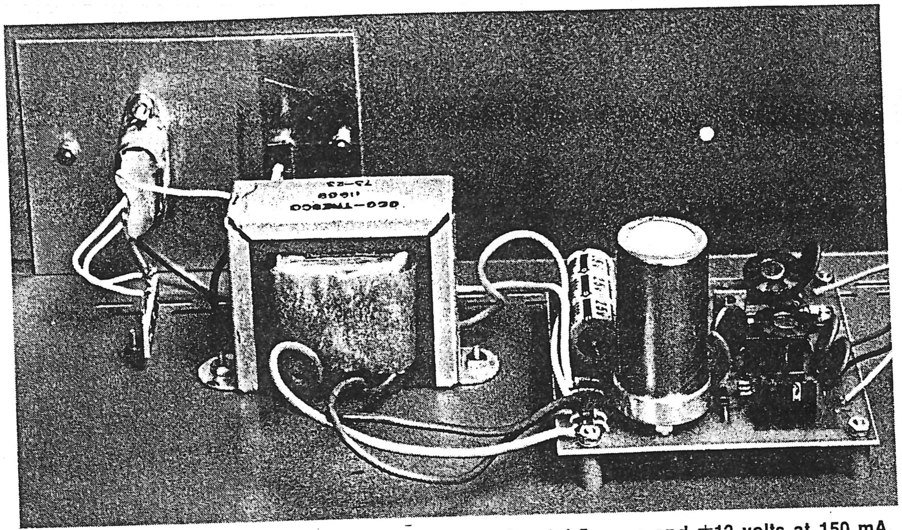


FIG. 2—INPUT/OUTPUT SECTION of the Dyna-Micro contains the keyboard and LED display.

hardwired, you can use them for data input and output with your own software. This is easily done with the software as we'll see later. Additional input and output ports are easily added by breadboarding them on the SK-10 socket or connecting them to the edge connector.

Interfacing experiments are all done on the SK-10 socket. All you will need to do the experiments are some no. 24 jumper wires and the necessary integrated circuits. No additional soldering or wire wrapping is necessary with the Dyna-Micro.



THE DYNA-MICRO POWER SUPPLY provides +5 volts at 1.5 amps and ±12 volts at 150 mA. You can design and build your own or purchase assembled supplies from surplus dealers.

Construction

The Dyna-Micro is constructed on a single printed-circuit board and it will be built and checked-out in stages to assure proper operation. These checks are fairly simple and all that is required is a voltmeter and a means of detecting for TTL level pulses. A monostable and LED will work quite well for this purpose. An oscilloscope is not needed, but might be helpful if you have one.

The schematic diagram of the Dyna-Micro is divided into two sections–the central processor unit and the I/O (*I*nput/*O*utput) section. The schematic diagram of the central processor unit is shown in Fig. 1 and the I/O section is shown in Fig. 2.

All the components of the Dyna-Micro, including the keyboard, are contained on a single 10" × 12" printed-circuit board. A double-sided board is used to minimize the number of jumper wires necessary for construction.

The power supply should be constructed first. If an assembled supply is purchased, it should be tested. The power supply must be capable of providing +5 volts at 1.5 A, and ±12 volts at about 150 mA. The power supply may be purchased from one of the many suppliers or from a surplus house. It should be ready before construction proceeds since it will be used to check the sections as we go along.

Mount the capacitors, resistor and Zener diodes in the voltage regulator section of the PC board. Be sure that the binding posts are in the upper-left corner when the board is in front of you. The +12 volts is used only by the 8080A and the 8224 crystal clock IC. The −12 volts is used by the voltage regulator circuitry to obtain the −5 volts for the 8080A chip and the −9 volts for the 1702A type PROM's. After soldering in the parts, connect the power supply and check for the voltages at the power supply to be sure that there aren't any shorts. Check for +12 volts at pin 28 of the 40-pin socket and at pin-9 at IC5 (both sockets will not be soldered in at this time). You can also check for −5 volts at pin 11 of the 40-pin socket. The −9 volts will not be present unless a PROM is in one of the sockets.

The clock circuit uses an Intel 8224 integrated circuit. This is a crystal oscillator that provides the proper MOS clock levels for the 8080 system. It also contains circuitry for a TTL level clock (Ø2), RESET and

(Continued on page 74)

PARTS LIST

R1, R2, R4, R5, R6—1000 ohms, ¼ W, 10%
R3—2200 ohms, ¼ W, 10%
R7-R30—220 ohms, ¼ W, 10%
C1—33 μF/6.3V electrolytic
C2—5 μF/50V electrolytic
C3, C5-C14—0.01 μF disc ceramic
C4—3.3 μF/16V electrolytic
D1-D24—Small red LED (Hewlett-Packard 5082-4484, Monsanto MV5075B, or equal.)
D25—1N751A, 5.1V Zener
D26—1N746, 3.3V Zener
IC1, IC22, IC23—SN7404 Hex inverter
IC2, IC3, IC30—SN7400 Quad 2-input NAND gate
IC4—SN74174—Quad type-D flip-flop
IC5—8224 Clock generator (Intel)
IC6, IC7—8216 Bus driver (Intel)
IC8—8080A CPU (Intel) (Must be "A" version)
IC9, IC10—8111-2 RAM memory (Intel)
IC13, IC17—SN74LS05 Open-collector hex inverter
IC14—SN74LS155 Dual 2-to-4 line decoder
IC15—1702A PROM memory (Intel)
IC18—SN74L42 BCD to decimal converter
IC19—SN7402 Quad 2-input NOR gate
IC20, IC21—SN74L04 Hex inverter
IC24-IC29—SN7475 Bistable latch
IC31—DM8095 or SN74365 Buffer
IC32, IC33—SN74148 8-to-3 line priority encoder
S1-S16—Keyswitches with legends
XTAL—6.750 MHz crystal, HC-18/U holder
Solderless breadboard—SK-10 IF18
Chassis—10" x 12" x 3" (Bud type AC-413)
Misc.—Binding posts, IC sockets, hardware

Optional IC's for expanded memory
IC11, IC12—8111-2 RAM memory
IC16—1702A PROM memory

Keyswitches are available from Solid State Systems, Inc., P.O. Box 617, Columbia, MO 65201. Order type LM or LFW-LT, with legends shown in Fig. 7.

Breadboarding socket is available from Circuit Design, Inc., Box 24, Shelton, CT 06484.

Power supplies are available from ....

The following kits are available from Circuit Design, Inc., Box 24, Shelton, CT 06484. Phone 203-735-8774. All kits, except MMD-1IC, MMD-1 PROM, and MMD-1 RAM come complete with construction details, experiments, and tutorial material.

#MMD-1CBK—Etched, plated-through PC board, keyboard parts and breadboarding socket. $125 postpaid.

#MMD-1K—Complete kit of parts including 1702A PROM preprogrammed with KEX software. $350 postpaid.

#MMD-1A—Completely assembled and tested system. $500 postpaid

#MMD-1IC—Microprocessor IC set includes one 8080A CPU, one 8224 clock generator, two 8216 bus drivers, two 8111-2 RAM memory, one 1702A PROM preprogrammed with KEX. $100 postpaid.

#MMD-1 PROM—Additional 256-word PROM (1702A). $40 postpaid.

#MMD-1 RAM—Additional 256 words of RAM (8111-2). $15 postpaid.

A user's group has already been formed for the Dyna-Micro. Interested people should contact:
Dr. Frank Settle, Jr.
Digital Directions
Box 1053
Lexington, VA 24450

(continued from page 38)

constantly be fed to the bus. With these two pins grounded, apply power and depress the keys, one at a time. The binary data for each keyswitch will be indicated on the LED's at all of the output-ports simultaneously. Note that the most significant bit, D7, will be on whenever one of the keys is depressed. This is often called a 'flag' since it is used to flag down the computer and tell it that one of the switches is ready with data.

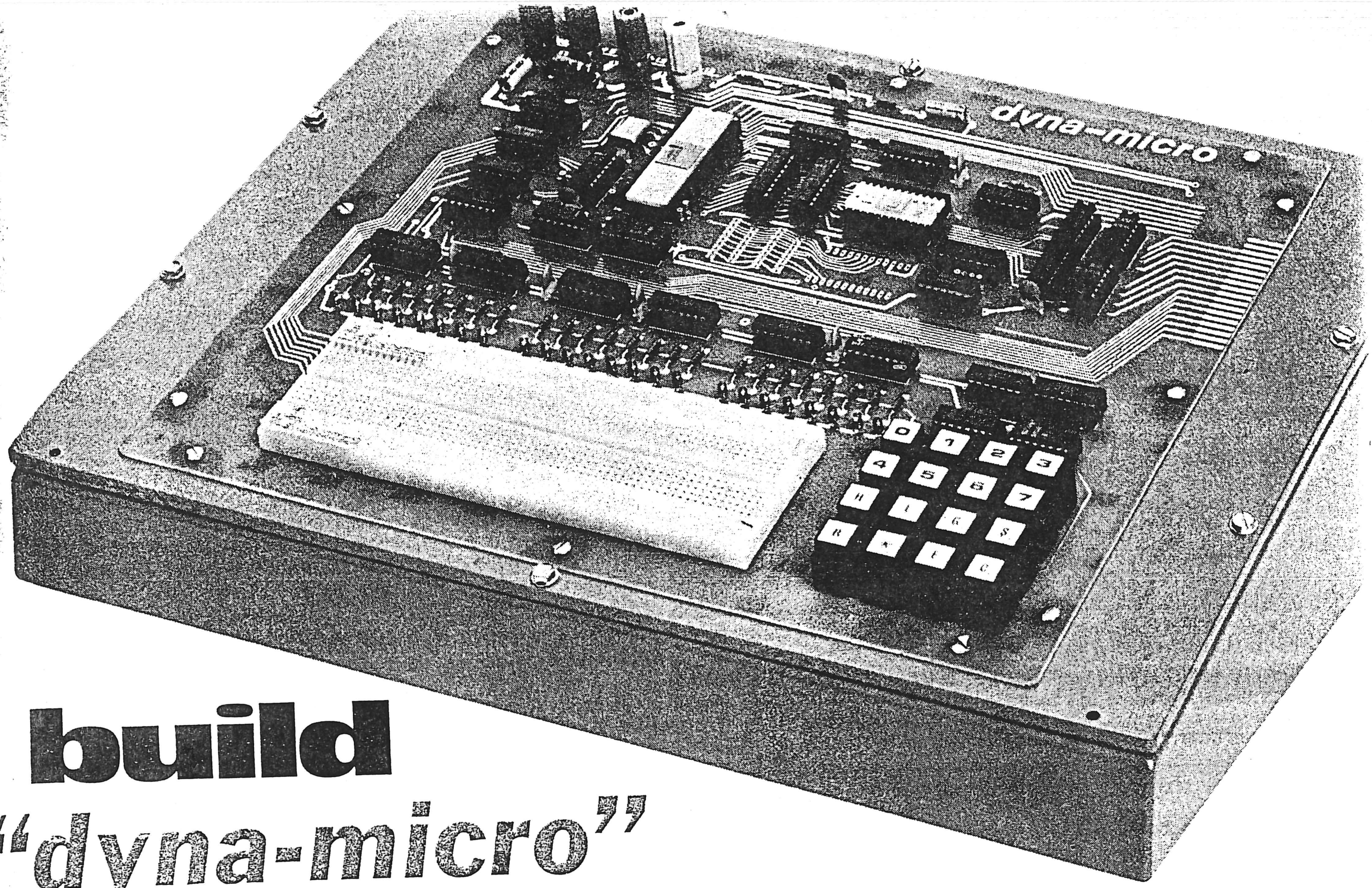
The lower-left reset (R) key will not output any data since it is hardwired to the 8224 clock chip. You can check this key's operation by testing the voltage at pin 1 of IC5, the 8224 clock chip. It should normally be at about zero volts and will rise to 3 volts when the R key is depressed.

If all the keys operate correctly, peal off the protective backings and apply them to the tops of the keys.

After the power supply, voltage regulator, clock, LED display and keyboard sections have been tested, install the remaining parts. Remember to add the 0.01 μF decoupling capacitors. If you are only going to be using 256 words of R/W memory, at least to start, be sure that it is correctly installed in the locations allocated for IC9 and IC10. If you have not already done so, obtain one of the 1702A PROM's with the Keyboard Executive (KEX) software in it. Preprogrammed PROM's are available (see parts list) and if you already have a 1702A PROM, it may be programmed from the listing provided. There are currently a number of 1702A type PROM's available. These are the 4702A and the 8702A. These are pin-for-pin equivalents of the 1702A, but their access times are slower. The 1702A PROM's or equivalents should have a maximum access time of about 1.3 ms to work with the Dyna-Micro. If you purchase a "surplus" PROM, be sure that these conditions are met.

The PROM containing the KEX software must be placed in the location allocated for IC15. It will not work correctly if placed in the location for IC16 since the addresses will be incorrect.

Next month, the foil patterns, component placement diagram, schematic diagram of the monostable-LED circuit and the final check procedures will be given. **R-E**



build "dyna-micro" an 8080 microcomputer

Complete with keyboard for data entry, LED readout of the address and data, breadboard socket for experimenting, 500-bytes of PROM, 500-bytes of RAM, expandable to 65K and self-contained power supply

JOHN TITUS

LAST MONTH, PART-1 OF THIS ARTICLE described the operation of the Dyna-Micro and presented the schematic diagram and construction details.

This month, the foil patterns and component placement diagram are presented along with a description of how to use the Dyna-Micro.

Final check

All integrated circuits should now be in place, hopefully in sockets. Remove the 8080A IC and check for the correct voltages at its socket. You should find +5 volts at pin 20, +12 volts at pin 28 and −5 volts at pin 11. You should also check the PROM for −9 volts at pins 16 and 24. A PROM must be in one of the sockets for the −9 volts to be present.

Turn on the power with all the IC's in place and check the power supply voltages. They should be at their preset levels of +5 and ±12 volts. If these are correct, your Dyna-Micro system should be operational.

Depress the R key. The LED displays should now indicate 003 (0000011) at the HI and 000 (00000000) at the LO. The OUTPUT PORT 2 LED's may have some random data present. If this doesn't happen, remove the power and carefully check your system. Things to check for are solder bridges, cold solder-joints, unsoldered IC pins and incorrect IC orientation. Plated through holes don't have to be soldered unless there is a component or other lead going through them. Also check for +5 volts and ground at all the IC's.

If the LED's display the correct pattern, depress and release the S key. Each time that this key is pressed, the LO address information should be incremented by 1. If this doesn't happen, check the keyboard encoder section and the I/O sections.

If the LED's are operating correctly, enter some data (0 through 7) from the keyboard. The binary codes for these keys will be entered in the Data-Register (OUTPUT PORT 2) display in the three least-significant bits. You will note that as new data is entered, the old data is shifted to the left where it finally disappears as more new data is entered. The actual operation of the KEX software to input and output data will be discussed later.

How to use the Dyna-Micro

The Keyboard Executive software is the "heart" of the Dyna-Micro system. It allows you to examine data or pro-

PARTS LIST

R1, R2, R4, R5, R6—1000 ohms, ¼ W, 10%
R3—2200 ohms, ¼ W, 10%
R7-R30—220 ohms, ¼ W, 10%
C1—33 μF/6.3V electrolytic
C2—5 μF/50V electrolytic
C3, C5-C14—0.01 μF disc ceramic
C4—3.3 μF/16V electrolytic
D1-D24—Small red LED (Hewlett-Packard 5082-4484, Monsanto MV5075B, or equal.)
D25—1N751A, 5.1V Zener
D26—1N746, 3.3V Zener
IC1, IC22, IC23—SN7404 Hex inverter
IC2, IC3, IC30—SN7400 Quad 2-input NAND gate
IC4—SN74174—Quad type-D flip-flop
IC5—8224 Clock generator (Intel)
IC6, IC7—8216 Bus driver (Intel)
IC8—8080A CPU (Intel) (Must be "A" version)
IC9, IC10—8111-2 RAM memory (Intel)
IC13, IC17—SN74LS05 Open-collector hex inverter
IC14—SN74LS155 Dual 2-to-4 line decoder
IC15—1702A PROM memory (Intel)
IC18—SN74L42 BCD to decimal converter
IC19—SN7402 Quad 2-input NOR gate
IC20, IC21—SN74L04 Hex inverter
IC24-IC29—SN7475 Bistable latch
IC31—DM8095 or SN74365 Buffer
IC32, IC33—SN74148 8-to-3 line priority encoder
S1-S16—Keyswitches with legends
XTAL—6.750 MHz crystal, HC-18/U holder
Solderless breadboard—SK-10 IF18
Chassis—10" x 12" x 3" (Bud type AC-413)
Misc.—Binding posts, IC sockets, hardware

Optional IC's for expanded memory
IC11, IC12—8111-2 RAM memory
IC16—1702A PROM memory

Keyswitches are available from Solid State Systems, Inc., P.O. Box 617, Columbia, MO 65201. Order type LM or LFW-LT, with legends shown in Fig. 7.

Breadboarding socket is available from Circuit Design, Inc., Box 24, Shelton, CT 06484.

The following kits are available from Circuit Design, Inc., Box 24, Shelton, CT 06484. Phone 203-735-8774. All kits, except MMD-1IC, MMD-1 PROM, and MMD-1 RAM come complete with construction details, experiments, and tutorial material.

#MMD-1CBK—Etched, plated-through PC board, keyboard parts and breadboarding socket. $125 postpaid.

#MMD-1K—Complete kit of parts including 1702A PROM preprogrammed with KEX software and power supply. $350 postpaid.

#MMD-1A—Completely assembled and tested system. $500 postpaid

#MMD-1IC—Microprocessor IC set includes one 8080A CPU, one 8224 clock generator, two 8216 bus drivers, two 8111-2 RAM memory, one 1702A PROM preprogrammed with KEX. $100 postpaid.

#MMD-1 PROM—Additional 256-word PROM (1702A). $40 postpaid.

#MMD-1 RAM—Additional 256 words of RAM (8111-2). $15 postpaid.

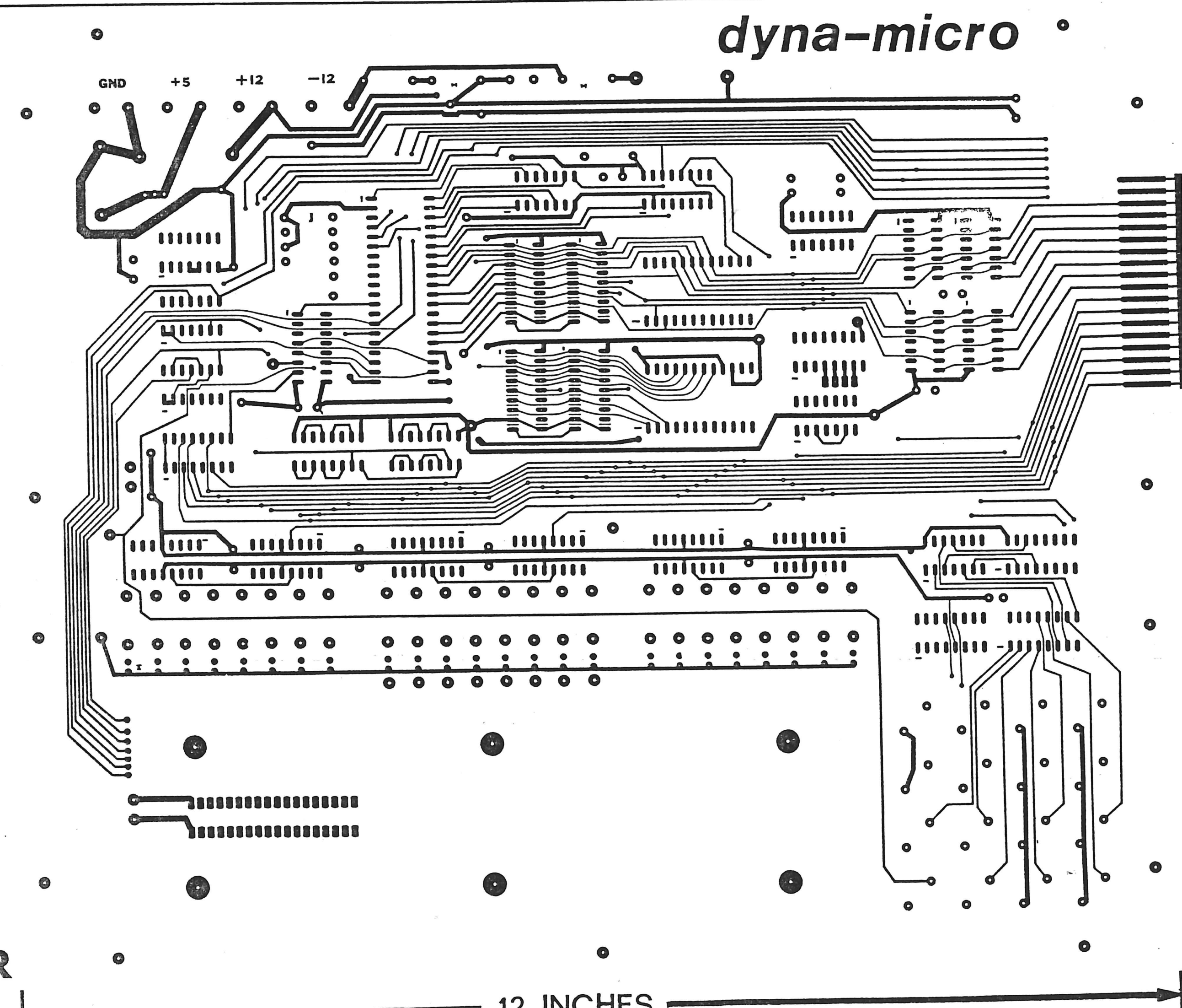


FIG. 3—FOIL PATTERN of the component-side of the double-sided board. The actual board measures 10 × 12 inches (254 × 305 mm.)

gram steps and to change data or program steps stored in the R/W (read/write) portion of the memory. We can also specify any address and start the program there.

The data keyswitches are labeled 0-7. When any one of these keys are depressed, data is entered in the corresponding octal format. The H key designates the HI address and the L key designates the LO address. The G represents Go and S represents See and Store. Three keys are not used by the KEX—A, B and C. The R key will always reset the computer and restart the KEX program. All manual data entry is through the keyboard in the basic Dyna-Micro system.

Whenever you want to start the system, depress keyswitch R. This will reset the KEX program and address the first location in the R/W section of the memory. This is HI=003 and LO=000. If you will only be using 256 words of R/W memory to get started, it must be in the locations allocated for IC9 and IC10. The KEX will not function without R/W memory.

To enter data, whether it will be used for new data or to address a memory location, simply depress the numbered keys as you would on a calculator. Data will be entered into the three least-significant right-most) LED's and it will shift to the left as more data is entered. If a mistake is made, simply re-enter the data. Mistakes are shifted out and lost. The data-register LED's will display the data just entered from the keyboard and this may be used as the HI address by depressing H, or it may be used as the LO address by depressing L. These keys will transfer the data to the proper LED display register and it will be used by the 8080A to address a new memory location.

Whenever a new HI or LO address is specified by depressing either the H or L keys, the KEX program will always display the contents of the specified memory location on the data-register LED's. To examine the contents in the next location, depress the S key. By depressing the S key again and again, we can examine the contents in sequential memory locations. It should be noted that this S function follows increasing memory locations, *not* the sequential *flow* of a program.

To change the contents in a R/W memory location, simply load the address using the data input keys and the H and L keys. The old data presently in the location will immediately appear on the data-register LED's. Enter the new data into the data register using the numeric keys and then enter it into the R/W location by depressing the S key. After S is depressed, the new data is stored and the address is automatically incremented by one to address the next memory location. The data from the next location is now displayed on the data-register LED's.

The S key has two functions, both See and Store. How can we tell the difference? If the data has changed we will store it and see the next location. If the data hasn't changed, it will be stored in the same location that it originally came from and then the contents from the next memory location will be displayed. When we store old data back to the same location, we can't really see

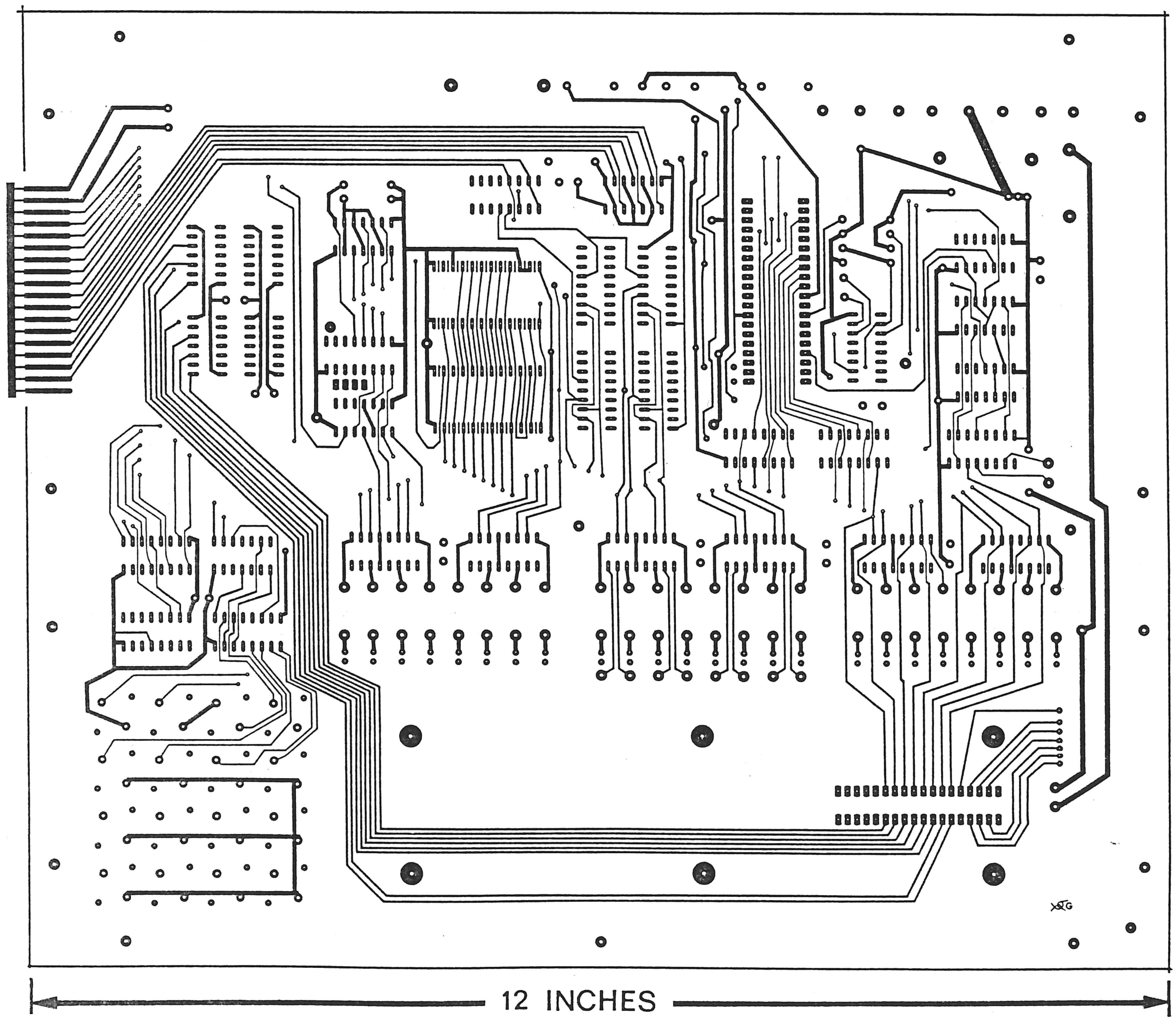


FIG. 4—FOIL PATTERN of the bottom of the double-sided board. A board with plated-through holes is recommended.

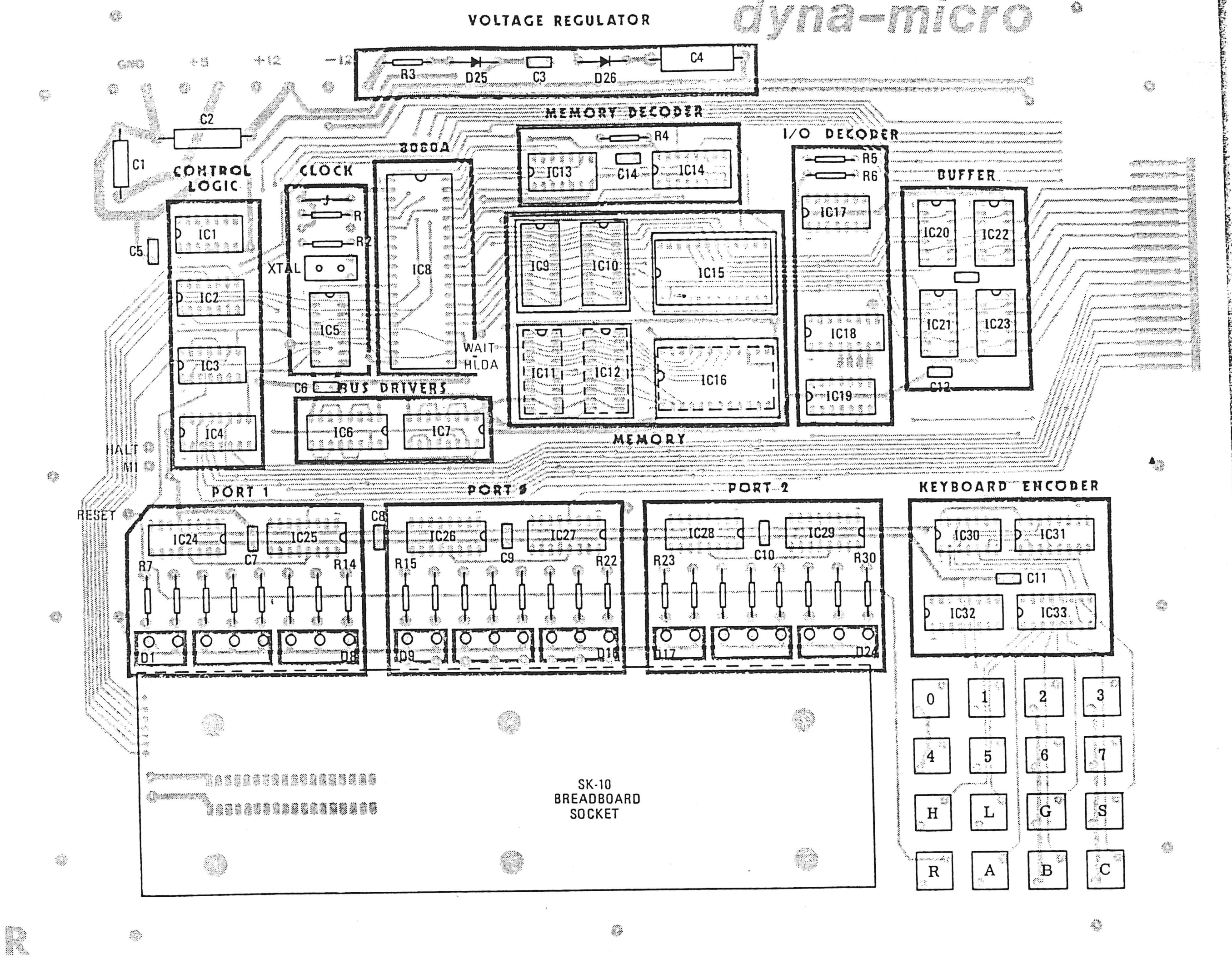


FIG. 5—COMPONENT PLACEMENT diagram. The board is shown divided into functional sections.

any change, but this is exactly what the KEX does. It displays data from a specific location, allows us to make changes and then puts it back. If no changes are made, the old data is restored to its memory location.

Once a program is entered into the computer through the keyboard, we can start it by loading our starting address and actuating the G key. This will transfer control from the KEX software to the program that we want to run. Starting addresses are loaded in the same way as previously described. Starting addresses don't have to be in R/W memory, but can just as easily be in PROM.

If your program starts at the first location in R/W memory (003 000) you can simply depress R followed by G. We can do this because KEX always resets the address back to this first R/W location.

Keys labeled A, B and C are not used by the KEX program. It should be remembered that the three LED output ports and the keyboard are not hardwired for use with the KEX program only. They are available for you to use in your programs. All fifteen keys may be used in any way you like, using software.

TABLE 1—MEMORY ALLOCATION

HI	LO	
000	000	KEY PROM
000	377	
001	000	OPTIONAL PROM
001	377	
002	000	OPTIONAL R/W MEMORY
002	377	
003	000	R/W MEMORY
003	377	
004	000	AVAILABLE FOR USER EXPANSION
377	377	

How KEX operates

The keyboard Executive software is contained in a single 1702A type PROM in the location allocated for IC15. This contains all the necessary software to operate the keyboard and the LED displays. This is our software controlled "front panel", since the keys and LED's perform functions determined by the KEX software.

Whenever the R key is depressed, the 8080A CPU will start to execute the program that starts at location 0. Looking at the software listing for the KEX program (Table II), you will see that immediately after starting at location 0, the software instructions cause the computer to jump to location HI=000, LO=070 (HI=000 throughout the KEX program) where we start the program by pointing to the first R/W memory address (003 000.) The address and the data in that location are displayed on the three output ports. This is done between POINTA and POINTC in the program (see Table II). The software between POINTC and POINTD will do the necessary tasks to input new data from the keyboard and shift the data onto the LED's. The shifting is done inside the 8080A with software instructions. Doing this by hardware would require

(continued on page 84)

BUILD A COMPUTER
(continued from page 44)

many more IC's, but it takes relatively few software steps.

The software routines at POINTD, POINTE, POINTF and POINTG make up what is called a command decoder. The software decodes the keyswitches into real actions. Depressing H or L causes the data temporarily stored in the 8080A as numeric key inputs to be output to either the HI or LO set of LED's. The S key causes the current or new data to be put back into the current memory location. Depressing G causes the computer to use the HI and LO address as the starting point for a new program.

The TIMOUT and KBRD software subroutines have specific tasks. The TIMOUT will count its way through various loops for about 10 milliseconds, while the KBRD subroutine will input a code from the keyboard. The KBRD subroutine has some unique features that illustrate an interesting hardware-software tradeoff. The keyswitches used in the Dyna-Micro are not bounce free, so that when the switches are opened or closed, they can often remake or re-break the contacts. This can be confusing to the computer since it can't distinguish between a real switch closure and a bounce. We don't want

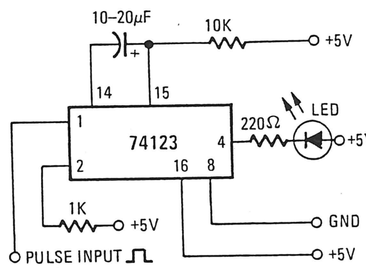


FIG. 6—MONOSTABLE-LED circuit is used for check-out of the Dyna-Micro.

0	1	2	3
4	5	6	7
H	L	G	S
R	A	B	C

FIG. 7—KEYBOARD LEGENDS are oriented as shown.

A user's group has already been formed for the Dyna-Micro. Interested people should contact:
Dr. Frank Settle, Jr.
Digital Directions
Box 1053
Lexington, VA 24450

WE GOOFED

WE SURE DID! Last month we ran the first part of this construction article on the Dyna-Micro and we erred. It seems that one or more gremlins (little green people) found their way into our editorial offices and stole—that's right, stole—74 lines of text from the article. The text was stolen between the last paragraph on page 36 and the opening paragraph on page 74. To correct this evil crime, we are reprinting the two paragraphs in italics along with the missing text. To prevent this from happening again, we have tightened security around our offices.

The clock circuit uses an Intel 8224 integrated circuit. This is a crystal clock oscillator that provides the proper MOS clock levels for the 8080 system. It also contains circuitry for a TTL level clock (Ø2), RESET *and* READY inputs. Construction of the clock circuit begins by inserting components R1, R2, the 6.75 MHz crystal, IC5 and the jumper. Good quality IC sockets are recommended for all the integrated circuits.

After inserting and soldering the parts, the clock section should be checked. To do this, apply power and check for voltages on the 8224 chip. You should observe +5 at pin 16, +12 at pin 9 and ground at pin 8. Clock operation can be checked at pins 10 and 11. These are the MOS level outputs that swing between +12V and ground. The signals can be observed with a good scope. The TTL output on pin 6 can also be checked with a scope or with a monostable-LED circuit. The monostable circuit can be constructed on the SK-10 socket before it is added to the system.

With the 6.75-MHz crystal, the output frequency will be 750 kHz. This is slower than the maximum 2-MHz frequency that the 8080A will operate at, but a slow frequency was chosen to allow for slow access times of the PROM's.

The output ports are constructed next by inserting and soldering all the LED's (D1 through D24). Be sure to observe the polarity as shown by the symbol near D1. Add all the 220-ohm resistors (R7 through R30) and insert the SN7475 latches. Be careful to orient the IC's in their sockets. The foil pattern has a small mark near pin 1 for all the IC's.

With the latches and parts installed and soldered, apply power to the system. All of the LED's should light. If any are not on, check the associated SN7475 latch. With power still applied, ground the input pins to the latches, one at a time. Since all the inputs are on eight common data bus-lines, only eight inputs must be grounded to check all the LED's and latches. Ground pins 2, 3, 6 and 7 on IC24 and IC25. One of the LED's in each group of eight should go out, one at a time. If this doesn't happen, again check the SN7475's.

The keyboard section consists of 16 keyswitches—15 are used to input data and one is hardwired to the 8224 chip to reset the Dyna-Micro. The keyswitch closures are encoded by two SN74148 octal encoders and the encoded binary data is gated onto the bus through a three-state DM8095 or SN74365 IC.

Insert and carefully solder the keyswitches to the printed-circuit board and then insert the four integrated circuits, IC30 through IC33.

The keyboard section is tested by monitoring the data on the LED's. Carefully ground pins 1 and 15 on the three-state driver, IC31. This will cause data from the keyswitch encoders to *constantly be fed to the bus. With these two pins grounded, apply power and depress the keys, one at a time. The binary data for each keyswitch will be indicated on the LED's at all of the output-ports simultaneously. Note that the most significant bit, D7, will be on whenever one of the keys is depressed. This is often called a 'flag' since it is used to flag down the computer and tell it that one of the switches is ready with data.*

the computer to sense each bounce as a key closure so we would like some way to filter them out. Additional circuitry including latches, clocks and monostables could do this for us, but it complicates the system. We can also do the debouncing via software.

The KBRD subroutine will recognize any key closure, but it will only input the key codes after being sure that the key is closed and not bouncing. It does this by waiting after sensing a closure and then rechecking the switch to be sure it is still closed. It also checks when we release a key to be sure that it has stopped bouncing before it tries to sense another key being depressed by the user. We have traded some additional software steps for a great deal of hardware. Since there was plenty of PROM left, it was easy to include.

The TIMOUT and KBRD software

(Table II is on page 86)
(Text continues on page 90)

Text continues on page 90

TABLE II—KEYBOARD EXECUTIVE (KEX) PROGRAM

```
                                 *000 000
000 000  303                     JMP
000 001  070                     START
000 002  000                     0

                                 /JUMP UP TO R/W MEMORY TO
                                  BE USED BY RESTARTS &
                                  VECTORED INTERRUPTS
                                 *000 010
000 010  303                     JMP
000 011  010                     010
000 012  003                     003
                                 *000 020
000 020  303                     JMP
000 021  020                     020
000 022  003                     003
                                 *000 030
000 030  303                     JMP
000 031  030                     030
000 032  003                     003
                                 *000 040
000 040  303                     JMP
000 041  040                     040
000 042  003                     003
                                 *000 050
000 050  303                     JMP
000 051  050                     050
000 052  003                     003
                                 *000 060
000 060  303                     JMP
000 061  060                     060
000 062  003                     003

                                 /BEGINNING OF MAIN PROGRAM

                                 *000 070
000 070  061   START,            LXISP   /SET STACK POINTER
                                          TO TOP OF R/W MEM.
000 071  000                     000
000 072  004                     004
000 073  041                     LXIH    /INITIAL VALUE FOR
                                          H & L
000 074  000                     000
000 075  003                     003
000 076  116   POINT A,          MOVCM   /LOAD MEM DATA INTO
                                          TEMP DATA BUFFER
000 077  174                     MOVAH   /OUTPUT HI TO LED'S
000 100  323                     OUT
000 101  001                     001
000 102  175                     MOVAL   /OUTPUT LOW TO
                                          LED'S
000 103  323                     OUT
000 104  000                     000
000 105  171   POINT B,          MOVAC   /OUTPUT TEMP. DATA
                                          BUFFER DATA TO
                                          LED'S
000 106  323                     OUT
000 107  002                     002
000 110  315   POINT C,          CALL    /WAIT & INPUT NEXT
                                          KEY CLOSURE
000 111  315                     KBRD
000 112  000                     0
000 113  376                     CPI
000 114  010                     010
000 115  322                     JNC     /JUMP IF KEY WAS <
                                          010
000 116  134                     POINT D /(0-7, OCTAL
                                          DIGIT)
000 117  000                     0
000 120  107                     MOVBA   /SAVE KEY CODE
000 121  171                     MOVAC   /GET OLD VALUE
000 122  027                     RAL     /ROTATE 3 TIMES
```

```
000 123  027              RAL
000 124  027              RAL
000 125  346              ANI      /MASK OUT LEAST
                                    SIG. OCTAL DIGIT
000 126  370              370
000 127. 260              ORAB     /OR IN NEW OCTAL
                                    DIGIT
000 130  117              MOVCA    /PUT NEW DATA
                                    BACK INTO BUFFER
000 131  303              JMP
000 132  105              POINT B
000 133  000              0
000 134  376  POINT D,    CPI
000 135  011              011      /"L" KEY
000 136  302              JNZ      /JUMP IF NOT AN "L"
000 137  145              POINT E
000 140  000              0
000 141  151              MOVLC    /PUT BUFFER DATA
                                    IN L
000 142  303              JMP
000 143  076              POINT A
000 144  000              0
000 145  376  POINT E,    CPI
000 146  010              010      /"H" KEY
000 147  302              JNZ      /JUMP IF NOT AN "H"
000 150  156              POINT F
000 151  000              0
000 152  141              MOVHC    /PUT BUFFER DATA
                                    IN H
000 153  303              JMP
000 154  076              POINT A
000 155  000              0
000 156  376  POINT F,    CPI
000 157  013              013      /"S" KEY
000 160  302              JNZ      /JUMP IF NOT "S"
000 161  170              POINT G
000 162  000              0
000 163  161              MOVMC    /PUT TEMP. DATA
                                    INTO MEMORY
000 164  043              INHX     /INCREMENT H & L
000 165  303              JMP
000 166  076              POINT A
000 167  000              0
000 170  376  POINT G,    CPI
000 171  012              012      /"G" KEY
000 172  302              JNZ      /JUMP IF NOT "G"
000 173  110              POINT C
000 174  000              0
000 175  351              PCHL     /GO EXECUTE PGM
                                    POINTED TO BY
                                    H & L

                          /THIS 10 MSEC DELAY
                           DISTURBS NO REGISTERS OR
                           FLAG

                          *000 277
000 277  365  TIMOUT,     PUSHPSW  /SAVE REGISTERS
000 300  325              PUSHD
                          LXID     /LOAD D & E WITH
                                    VALUE TO BE
000 301  021                        DECREMENTED
000 302  046              046
000 303  001              001
000 304  033  MORE,       DCXD     /JUMP IN THIS
                                    LOOP UNTIL
000 305  172              MOVAD    /D & E ARE BOTH
                                    ZERO
000 306  263              ORAE
000 307  302              JNZ
000 310  304              MORE
000 311  000              0
000 312  321              POPD
000 313  361              POPPSW   /RESTORE
                                    REGISTERS
000 314  311              RET

                          /THE KBRD ROUTINE
                           DEBOUNCES KEY CLOSURES
                          /AND TRANSLATES KEY CODES
```

Table II continues on page 89 (text continues on page 90)

(continued from page 87)

segments have been set up as subroutines and can be used in your software and in the experiments. Each of these subroutines may be started with a CALL instruction, 315. The TIMOUT subroutine does not affect any of the registers or flags and it only serves to delay the software flow by 10 ms.

An important distinction between the 8008 and the 8080 processors is in the use of subroutines. In the 8008, return-pointer addresses were stored in the 8008 IC itself. In the 8080, these return-pointer addresses are stored in a portion of the R/W memory. This is called a "stack" area. Whenever a subroutine is used, we want to execute the subroutine and then return back to the normal program flow. These return addresses are very important to the computer since they provide the only link between the subroutine and the main program. If we are to store them in a portion of R/W memory, the computer must know where this storage area is if it is to be able to use the addresses properly. In the KEX software, this is preset to be the top of the R/W memory with instructions at locations 070, 071 and 072. The LXISP instruction loads an internal 8080 stack-pointer register to HI=004, LO=000. Since the stack-pointer register is *decremented* to point to a new location before anything is stored, the first stack location will be HI=003, LO=377. Check your 16" bit binary numbers if this looks a little confusing.

You can use the stack as set up by the KEX (generally a good idea) or you can put your own stack anywhere you want, just by using the LXISP instruction. Remember to avoid the stack area when writing your programs. Remember, too, that you can't put the stack in an area of non-existent memory or in PROM.

You will use the stack area and you'll see how it can also be used to temporarily store data. This will be covered in the software modules. Let's see how the TIMOUT and KBRD subroutines can be used in our own software. We will use the software stack already set up in KEX.

Let's input a keyboard character, add a constant to the binary code for that character and display the result. We would first CALL the KRBD subroutine to input a binary key-code, then add the constant and display the result. The software listed in Table III will do this:

You can enter this with the KEX program. Depress the R key and start entering data. Enter 000 at location 003 004 so we'll first add zero to the codes. This will let us check what values are assigned to each key. Write down the codes. Go back and restart

TABLE III

HI	LO	INSTR	MNEMONIC	
003	000	315	CALL	/Input keyboard character
003	001	315	KBRD	/Subroutine's LO address
003	002	000	0	/Subroutine's HI address
003	003	306	ADI	/Add the following DATA to
003	004	???	DATA	/the contents of register A
003	005	323	OUT	/Output data from register A to
003	006	000	000	/device 000 (LEDs)
003	007	303	JMP	/Jump back to program at
003	010	000	000	/LO address = 000
003	011	003	003	/HI address = 003

TABLE IV

Assume HI = 003 throughout this program

LO	INSTR	MNEMONIC	
000	006	MVIB	/Load register B with the following data
001	370	370	/Data; time constant
002	315	CALL	/Call TIMOUT subroutine at
003	277	277	/LO address = 277
004	000	000	/HI address = 000
005	005	DECB	/Decrement B by 1
006	302	JNZ	/Jump if result is not zero to
007	002	002	/LO address = 002
010	003	003	/HI address = 003
011	076	MVIA	/Load register A with the following data
012	377	377	/Data; all 1's
013	323	OUT	/Output to device,
014	000	000	/device 000 (LED's)
015	166	HLT	/Stop once you reach here

KEX and change the value in 003 004 to, say, 005. This will add 5 to each code. Restart your software and see if this is the case. Congratulations, you have just done your first software experiment! The instructions at 003 003 and 003 004 could be changed to do other things to the data. Can you suggest one?

The 10 ms delay routine, TIMOUT, can be useful when we want a software delay that is in multiples of 10 ms. The software routine listed in Table IV will delay an output of all l's on OUTPUT PORT 0 by about 2.5 seconds after the program is started. Try it. Can you see how the time delay might be shortened? Can you see any use for programs like this?

The keyboard input subroutine, KBRD, is called at address 000 315 and the time delay subroutine, TIMOUT, is called at address 000 277.

Next month, this construction article concludes with a description of the 8080A microprocessor and how the Dyna-Micro works. This will include an explanation of how the memory is accessed and how the Dyna-Micro selects input/output devices via software. R-E

build an 8080 microcomputer

The first two parts of this article (May and June, 1976) described the Dyna-Micro and presented the construction details.

This third and concluding part of the article discusses the 8080A and describes how the Dyna-Micro works.

The 8080A microprocessor

The 8080A microprocessor is an eight-bit parallel central processing unit (CPU). Unlike the earlier 8008 and 8008-1, the 8080A is a more powerful, easy to use device. The 8008 was an 18-pin device and many things were multiplexed on the same bus. The 8080A is a 40 pin package that offers many advantages over the earlier devices:

- A simplified bus that has separate address (16 bits) and data (8 bits) buses.
- Three state and TTL compatible buses.
- Simplified control.
- Improved interrupt and stack.
- More instructions.

While the power and clock requirements for the 8080A are more complex than for the 8008, they are easily met.

How the Dyna-Micro works

The Dyna-Micro uses the same basic computer functions found in most other computers, large or small. These are:

- Central processing unit (CPU)
- Memory (R/W and PROM)
- Control Logic
- Input/Output Control
- Clock and power supplies

The 8080A has 24 output-lines, 16 of which are used for address information and eight of which are a bi-directional data bus taking data to and from the CPU. The data flow on the data bus is controlled by the DBIN (Data Bus IN) signal from the 8080A chip on pin 17. Two bus buffers, Intel type 8216, are used to boost or buffer the data bus outputs and to protect the 8080A's inputs. The DBIN signal switches the 8216 IC's for either input or output of data on the bus.

The system's operation is controlled by signals from the 8080A at various times as synchronized by the clock frequency. An Intel 8224 clock chip with a 6.75-MHz crystal is used to provide the 750-kHz MOS clock levels to the 8080A. These clocks are slightly out of phase and do not overlap. While the 8080A chip can operate at frequencies up to 2 MHz, the Dyna-Micro's clock has been set to 750 kHz so that it will work with the relatively slow 1702A type PROM's. You can increase the speed of the Dyna-Micro by using a higher-frequency crystal. The 8224 chip divides the crystal's frequency by nine, so an 18 MHz crystal will give the 2 MHz output needed for the highest operating speed.

The clock chip also synchronizes the READY input with the clock so that system can slow down for slow memories and I/O devices. The RESET input from the R key is also input to the 8224 IC where it, too, is synchronized to the clock and output to reset the 8080A.*

The 8080A's data bus will also contain some STATUS information that is used by the control logic section to synchronize its control operations. The STATUS information is output and is available when the SYNC signal is output from the 8080A. The 8224 clock IC gates the SYNC pulse from the 8080A with the main system clock to provide a very short STSTB (Status Strobe) signal that 'catches' the status information in the SN74174 latch. The status information indicates data output, data input, halt condition, memory read, memory write and a special condition that signals the start of the next instruction, M1. Four of these status signals are gated with other system signals to provide the $\overline{IN}$, $\overline{OUT}$, $\overline{MW}$ and $\overline{MR}$ signals used throughout the Dyna-Micro. These signals are used as indicated below:

- $\overline{IN}$ and $\overline{OUT}$—used to synchronize data flow to ($\overline{IN}$) and from ($\overline{OUT}$) the 8080A CPU. They are used with external devices for data transfer.
- $\overline{MR}$ and $\overline{MW}$—used to indicate to the memory that the computer is executing a Memory Read or a Memory Write.
- $\overline{INTA}$—indicates that the computer has acknowledged an interrupt.
- HALT and ML—latched and provided for the user. These are not used in the Dyna-Micro, but are described in the experiment modules and are available for your use.

When the 8080A addresses memory, one and only one address must be specified on the 16-bit address lines. Since the 1702A and the 8111-2 memories only have eight lines to address their 256 locations, the remaining eight lines must be used to select one and only one of the memory blocks. We can select from two blocks of PROM, two blocks of R/W or possibly no blocks on the Dyna-Micro if we have expanded the memory off the PC board.

Address lines A0 through A7 are applied directly to the memory IC's on the Dyna-Micro card. Address lines A10 through A15 are gated in a six input NOR gate IC13. When this gate's output is a logic 1, it indicates that we are selecting one memory location within the first 1024 locations. Address lines A10–A15 are equal to logic 0. This would be one of the locations in either R/W or PROM, since these are located within the first 1024 locations. If the NOR gate produces a logic 0, at least one of the address inputs (A10-A15) must be at a logic 1, indicating that we are now addressing memory outside the blocks on the Dyna-Micro. Address bits A8 and A9 and the memory read signal ($\overline{MR}$) are gated together in an SN74LS155 decoder to select the proper block of the four possible on the Dyna-Micro card. The 8111-2 memories input the memory read and memory write signals to control the flow of data from and to the CPU. The 1702A PROM has only a chip-select input, so the memory read synchronizing signal is used in the SN74LS155 to gate data from the 1702A onto the bus at the correct time. Table V shows the memory selection.

TABLE V—MEMORY SELECTION

NOR Output	A9	A8	$\overline{MR}$	Memory
0	X	X	X	NONE
1	0	0	0	PROM 0
1	0	1	0	PROM 1
1	0	0	1	NONE
1	0	1	1	NONE
1	1	0	X	R/W 2*
1	1	1	X	R/W 3*

*$\overline{MR}$ and $\overline{MW}$ applied independently to the 8111-2 chips.

X = Don't care condition, logic 0 or logic 1.

Data flows to and from the memory on the bidirectional data bus and the 16-bit address bus specifies each location. This address bus is also used to help control the flow of data to and from external devices. The eight least-significant or LO address bits can specify one of 256 possible peripheral devices. The address bits are decoded in

(continued on page 85)

*For more information about these signals and IC's, see the *8080 Microcomputer System User's Manual*, Sept. 1975 from Intel Corp., 3065G Bowers Ave., Santa Clara, CA 95051.

BUILD A COMPUTER
(continued from page 48)

the Dyna-Micro to select one of four possible onboard devices. These are the three groups of LED's and the keyboard. Address bits A3–A7 are all gated together in IC17, an OR gate made from a series of open-collector inverters in an SN74LS05 chip. If the 8-bit LO address specifies a value greater than seven, the OR gate's output becomes a logic 1. Address bits A0–A2 are used in IC18, an SN74L42 decoder chip to provide a one-out-of-eight code for devices 0 to 7. The OR gate's output is used to enable the decoder for the device coded 0–7, but to disable it for other codes. The decoding scehme is shown in Table VI.

TABLE VI—ON-BOARD I/O DEVICE DECODER

OR Gate	A2	A1	A0	Device
1	X	X	X	NONE
0	0	0	0	LED Port 0, Keyboard
0	0	0	1	LED Port 1
0	0	1	0	LED Port 2
0	0	1	1	Available for user
0	1	0	0	Available for user
0	1	0	1	Available for user
0	1	1	0	Available for user
0	1	1	1	Available for user

X = Don't care condition. Can be either a logic 1 or a logic 0.

You will notice that device-code 0 selects both the LED PORT 0 and INPUT PORT 0 (the keyboard.) How can the computer output and input data at the same time. It can't, and some means must be used to distinguish between them. The Control Logic section outputs two control signals, $\overline{\text{IN}}$ and $\overline{\text{OUT}}$ that indicate whether data will be input to the CPU or output to an external device. Both signals are exclusive, never taking place at the same time. The pulses are generated by the 8080A and the Control Logic section under control of the software. Only when we execute an I/O type instruction will either of these signals be generated. The device codes for the SN7475 latches are gated with the $\overline{\text{OUT}}$ signal in IC19, and the $\overline{\text{IN}}$ signal is gated with the keyboard device code in the DM8095 buffer chip associated with the keyboard, IC31. Data is input or output only when we have a valid device code *and* the presence of either the $\overline{\text{IN}}$ or $\overline{\text{OUT}}$ signals. **R-E**