BUILD THE ALTAIR 8800 MINICOMPUTER

PART TWO

Practical use of the computer, including programming

BY H. EDWARD ROBERTS AND WILLIAM YATES

LAST MONTH, we discussed the various subassemblies used in the basic Altair 8800 computer, went into details on how it is assembled, and listed a few applications. Here, we will describe a test program to be used in checking operation and then focus on practical uses and go through a software example to familiarize you with some operating procedures.

Test Program. The following simple program is used for initial testing of the computer's operation. It also illustrates how a program is loaded and run. The selected program will add two numbers stored at address locations 128 and 129 and store the result at address location 130. The procedure is as follows:

1 Set the power switch to ON and momentarily toggle the RESET switch. (Note: Excluding the power switch, all bottom-row switches on the front panel are spring-loaded, momentary-action types. The switches automatically return to their center-off positions when released from either of their operate positions. When instructed to operate any of the bottom row switches, momentarily throw it to the position indicated and release it.)

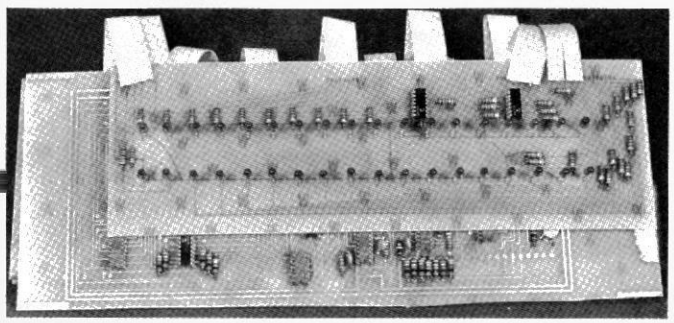
2 Set address switches *A0* through *A15* all to the 0 positions (down). Operate the EXAMINE switch, which should cause address LED's *A0* through *A15* to extinguish to indicate that location 0 is ready. (Some of the data LED's, *D0* through *D7*, might be illuminated, indicating the current contents at location 0.)

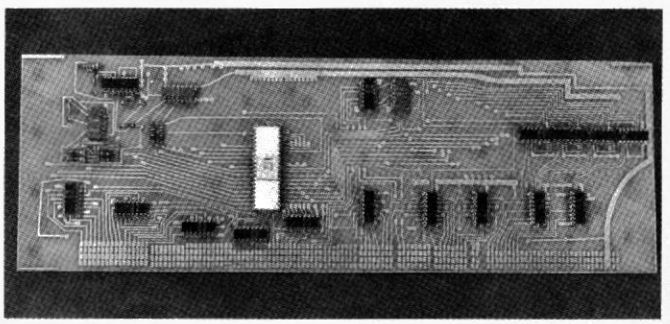
3 Next, store the load accumulator instruction at location 0 by using the binary number for 58 (00111010). Set this binary input up by using switches *D0* through *D7*, with a 1 represented by the switch in the up position and a 0 with the switch in the down position. Hence the switch sequence for 00111010 would be: *D7* down, *D6* down, *D5* up, *D4* up, *D3* up, *D2* down, *D1* up, *D0* down. Store this number at location 0 by operating the DEPOSIT switch. The *D0* through *D7* LED's should now match these settings, with a lighted LED indicating a 1 and a darkened LED indicating a 0. None of the *A0-A15* LED's should be on indicating location 0. The load accumulator instruction now tells the computer that the next two entries will be an address number (16 bits). Upon program execution, the data stored at that address number will be transferred to the accumulator.

4 Address numbers, such as address 128, are expressed in 16-bit binary format. The least-significant bits (last eight) are stored in the first memory location following the load accumulator instruction, while the most-significant bits are stored in the second memory location. Set *D0* through *D7* for 10000000 (128) and operate the DEPOSIT NEXT switch. This number is now stored, in binary form, at memory location 1. (*A0* LED should be lit indicating location 1.) Set *D0* through *D7* all to 0 and operate the DEPOSIT NEXT switch. The all-zero binary number is now stored at memory location 2 (*A1* LED is lit) and the computer has been instructed to put the contents of address 128 into the accumulator.

5 To add a second number to the current number stored in the accumulator, the computer must be instructed to transfer the current number to one of the general-purpose registers. In this example, we will use register B. The instruction used is "move A to B," where A is the accumulator. The code for this instruction is 01000111, set up with switches *D0* through *D7*. Operate the DEPOSIT NEXT switch. The instruction "move A to B" is now stored at memory location 3. (*A1* and *A0* lit.)

6 Now, instruct the computer to load the data from address 129 into the accumulator. This procedure is identical to that outlined in steps 3 and 4 above. Set switches *D0* through *D7* for 00111010 and operate the DEPOSIT NEXT switch. The load accumulator instruction is now stored at memory location 4. (*A2* lit.) Set *D0* through *D7* for





10000001 (129) and operate the DEPOSIT NEXT switch to store this number at memory location 5. (*A2, A0* lit) Then set *D0* through *D7* all to 0 and operate the DEPOSIT NEXT switch to store the all-zero number at memory location 6 (*A2, A1* lit).

7 Store the add instruction at memory location 7 by setting *D0* through *D7* for 10000000 (128) and operating the DEPOSIT NEXT switch. When executed, this instruction adds the number in the accumulator to the number stored in register B and places the result in the accumulator (*A2, A1, A0* lit).

8 To store the result at address 130, first store the instruction at memory location 8 by setting *D0* through *D7* for 00110010 and operating the DEPOSIT NEXT switch (*A3* lit). Set *D0* through *D7* for 10000010 and operate the DEPOSIT NEXT switch. The least-significant eight bits of address 129 are now stored at memory location 9 (*A3, A0* lit) Set *D0* through *D7* to 0 and operate the DEPOSIT NEXT switch. The most-significant eight bits of address 129 are now stored at memory location 10 (*A3, A1* lit).

9 A program that adds the contents of address 128 to the contents of address 129 and stores the result in address 130 has now been loaded into the computer. With the use of a "jump" instruction, you can now create a program loop that will direct the computer back to memory location 0 and allow repeating this addition procedure continuously for as long as desired. Store the jump instruction at memory location 11 by setting *D0* through *D7* for 11000011 and operating the DEPOSIT NEXT switch (*A3, A1, A0* lit). Set *D0* through *D7* to 0 and operate the DEPOSIT NEXT switch *twice*. The 16-bit address 0 is now stored at memory locations 12 and 13 (*A3, A2, A0* lit).

Before we can run this program, we have to load the two numbers we want added into addresses 128 and 129. For example, if we wanted to add 12 to 8, the procedure would be as follows:

Set address switches *A0* through *A15* for 0000000010000000 (128) and operate the EXAMINE switch (*A7* lit). Set *D0* through *D7* for binary 12 (00001100) and operate the DEPOSIT switch (*A7* still lit). Set *D0* through *D7* for binary 8 (00001000) and operate the DEPOSIT NEXT switch. The binary numbers for 12 and 8 are now stored at address locations 128 and 129, respectively (*A7, A0* lit).

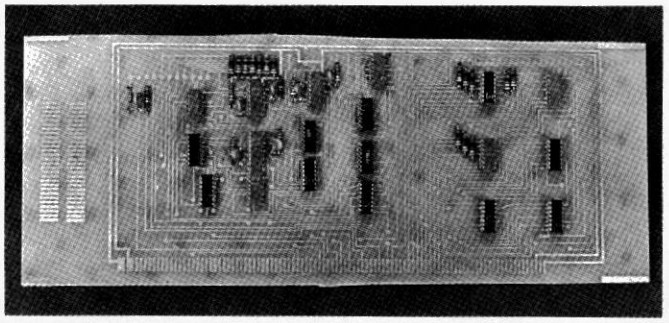
Set address switches *A0* through *A15* to 0 and operate the EXAMINE switch (all A LED's are off). Operate the RUN switch, and the program will execute at a rate of about 30,000 times per second. Operate the STOP switch. Set the address switches to address 130 (10000010) and operate the EXAMINE switch. LED's *D0* through *D7* will display the sum of the two numbers added, which is 20, in binary format (00010100).

Basics of Programming. If you have never done any programming, it may seem a little mysterious at first, but the basic ideas of programming

GLOSSARY OF COMPUTER JARGON

Access time — Time interval between the instant at which information is called for storage and the instant at which delivery is complete.
Accumulator — Part of the logical-arithmetic unit of a computer used for intermediate storage, to form algebraic sums, or other intermediate operations.
Address — Label, name, or number identifying a register, location, or unit where information is stored.
Assembler — Translates input symbolic codes into machine instructions.
Bit — Abbreviation of binary digit; a single character in a binary number.
Buffer — Isolating circuit used to avoid reaction of a driven circuit upon its driving circuit.
Byte — Group of binary digits usually operated upon as a unit. Usually shorter than a word.
Clock — Time-keeping device used to synchronize the computer.
Data — Basic elements of information which can be processed or produced by a computer.
Hold — Function of retaining information in one storage device after transferring it to another device, in contrast to clear.
Instruction — Coded program step that tells the computer what to do for a single operation in a program.
Interrupt — Break in the normal flow of a system or routine such that the flow can be resumed from that point at a later time.
Jump — Depart from the normal sequence of executing instruction in a computer (synonymous with branch).
Memory — Storage. A device that holds information that can be extracted at a later time.
Processor — Device capable of receiving data, manipulating it, supplying results usually of an internally stored program.
Programming — Art of reducing the plan for the solution of a problem to machine-sensible instructions.
Register — Device for the temporary storage of one or more words to facilitate arithmetical, logical, or transferral operations.
Stack — Portion of a computer memory and/or registers used to temporarily hold information.
Subroutine — Set of instructions in machine code to direct the computer to carry out a well-defined mathematical or logical operation; a part of a routine.
Word — Set of characters that occupies one storage location and is treated by the computer as a unit and is transported as such. Word lengths are fixed or variable, depending on the particular computer being used.

Definitions were extracted from "Computer Dictionary" by Charles J. Sippl and Charles P. Sippl, published by Howard W. Sams & Co., Inc., The Bobbs-Merrill Co. Inc., Number 20943, 484 pages, $8.95 (in Canada $11.95).



Shown at far left is the display board atop the control board, with cables that connect to other boards. The central processor unit is shown in the center, and the control board at near left. Not shown is memory board, which holds 17 IC's.

MACHINE INSTRUCTIONS

Instruction	Binary Code (for instruction)	Octal	Comment
IN 6	11011011 (IN)	333,006	Bring data from input 6 and store in register A (accumulator).
MOV B,A	01 (MOVE) 000 (B) 111 (A)	107	Take A and move its contents to B.
IN 30	11011011 (IN) 00011110 (30)	323,036	Bring input 30 into accumulator
ADD B	10000 (ADD) 000 (B)	200	Add contents of A to B. Put results in A.
OUT 128	11010011 (OUT) 10000000 (128)	323,200	Transmit contents of accumulator to output 128.

are really very straightforward and easy to master. The procedures that are always used consist of the following:

Defining the Problem. This is by far the hardest part of the programming. Don't worry about the computer or the computer language when doing this part of the preparation. Simply decide what is required to do the job you want to accomplish.

Establishing an Approach. The computer and computer language have nothing to do with this step, either. It involves outlining a step-by-step procedure to achieve the desired results and getting it down on paper.

Writing the Program. Once you are familiar with programming, you will find that this step is the simplest. It is merely a matter of translating step 2 into the appropriate language.

There are many books available on programming. Some of them are quite good and are particularly useful for learning techniques such as flow programming, looping, etc. However, in essence, they can all be boiled down to the three steps above.

Software Example. To get a feel for what programming the Altair 8800 is like, let's go through a sample program, which is similar to the test program that we first went through to check out the computer operation. Assume that we want to take the data available from input channel 6 and input channel 30 and add them, placing the result in output channel 128. The machine instructions are shown in the box.

The first instruction simply stores the data from channel 6 in register A (the accumulator). The next instruction moves this data from register A to register B. This clears A for the next input. The third instruction brings the data from input channel 30 into the A register. The fourth instruction adds the contents of register A (data from channel 30) to register B (data from channel 6) and puts the results back into register A. The final instruction transmits the answer from A to output channel 128. Total computer time used to perform this operation with the Altair 8800 is 18 microseconds. To put it another way, the computer could perform 56,000 of these operations in one second.

The instructions could be entered into the processor in one of three ways. The first and easiest would be with the use of an assembler. This is essentially a piece of software that converts alphanumeric symbols to machine language (binary code). For example, the assembler would convert our first instruction (IN 6) to the correct binary code. The problem with using an assembler is that you need a computer terminal for an input device and the assembler itself requires about 6000 words of memory storage. If extensive program development is to take place, the assembler is a good tool to have.

The next easiest method of entering the instructions is with the use of the Very Low Cost Terminal featured in the December 1974 issue of POPULAR ELECTRONICS. With this terminal, the instructions could be entered by using the octal code. The procedure would be to write the program in assembly language and then enter the corresponding code for each instruction. This system, while not being as fast as the use of an assembler is less expensive.

The third method, using front panel entry, is of course inexpensive but time consuming.

This has been only a brief summary of the programming procedures for the computer. Complete programming information is provided with the Intel 8080 integrated circuit and with the Altair 8800 computer kit. ◈

EXPANDING THE COMPUTER

In describing the assembly of the Altair 8800 Minicomputer in last month's article, it was noted that the interior of the cabinet provides plenty of room for expansion. The room can be used to add many functions to the basic computer. For example, the present memory board in the Altair 8800 can be expanded with the addition of three 256-word memories (Kit 8802-MS available from the manufacturer, MITS at $34 per 256-word memory). Further additions require an expansion mother board having four connectors that can accommodate any four memory or input-output (I-O) cards. This expansion board (Kit 8800-EB) is available for $44, while a 4K dynamic memory card (Kit 8840-MC) costs $198. Various other kits—a vectored interrupt card and a real-time clock, among them—are also available.