OSI's Superboard II

The author saw the ads, bought the board and wrote the following review.

Bruce S. Chamberlain
1982 W. Bayshore Road, #42
Palo Alto CA 94303

Most of you have probably seen the full-page ads for the Ohio Scientific Superboard II, but before buying one you would like to know more about it. So I decided to review the Superboard II single-board computer. At $279 for a board with CPU, 4K RAM, regular 53-key alphanumeric keyboard, Kansas City Standard cassette tape interface, video interface, 2K monitor and 8K Microsoft BASIC in ROM, it has to be the biggest bargain around. All you need to make it operational is a 5 volt 3 Amp power supply and a video monitor or a television with a radio-frequency converter. If you add a cassette recorder to the system for program and data storage, you will be in business.

I live in Palo Alto, California, the heart of Silicon Valley. There are numerous computer stores around, but none of them carry Ohio Scientific products. While down in Los Angeles recently, I checked the computer stores there. The first store, a Byte Shop, told me that the nearest Ohio Scientific distributor was located about a mile away in Huntington Beach.

At the distributor's I looked at the board and read a pamphlet called "The Challenger IP Technical Report." I was so impressed by what Ohio Scientific claimed for their Superboard II that I ordered one. You can have a complete system for $529, which includes the Superboard II and a power supply in a cabinet, a television modified for direct video and a cassette recorder. This is the equivalent of computer systems costing $800 to $1000.

Hardware

When I ordered the Superboard II, the salesman said I would receive it in about three weeks. Much to my surprise, it arrived in two and a half weeks. The single printed circuit board with keyboard was packed in a box of Styrofoam and wrapped in aluminum foil. Packaged this way the board will survive the roughest handling unscathed. The box also contained cassette and video cables, four pamphlets, referenced at the end of this article, and a cassette with six programs: a math tutor for addition, subtraction, multiplication and division; a trigonometry tutor; a teaching program for children on counting from one to ten; a program for balancing your checkbook; a video game called "Star Wars"; and 20 historical questions about various presidents.

Looking over the 12 by 14 1/2 inch printed circuit board, I was surprised to see that it had been hand-soldered. This board has no solder mask or silkscreened legends. The board contains one 40-pin IC, eight 24-pin ICs, ten 18-pin ICs, twenty-seven 16- and 14-pin ICs, one 8-pin IC and ten empty sockets (for eight RAM ICs and two buffer ICs) for a total of 57 ICs. A few resistors, capacitors, a crystal and the keyboard completed the package. The board also has a fuse and reverse protection diode, so if you plug the 5 volt power supply in backwards (shudder), the fuse, rather than all the ICs, blows. The parts for the RS-232 interface and power supply are not provided with the board. In addition there are pads for stuffing four 16-pin ICs for prototyping.

The keyboard is a 53-key ASCII-type keyboard. It features a scanned, rather than decoded, array, so keys can be programmed for user functions. With the shift-lock key down (the shift-lock key is mislabeled, it actually puts the keyboard in a different mode rather than locking it in the shift mode), the keyboard generates uppercase and numerals. Holding either shift key down will generate punctuation marks.

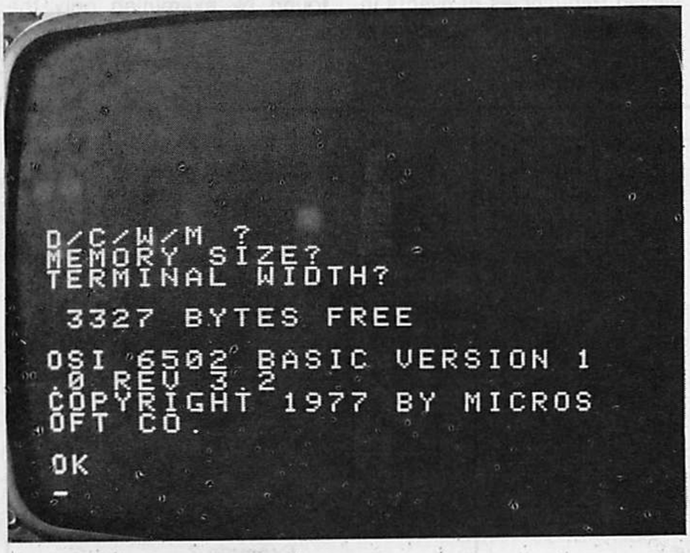

Photo 1. Display after the start-up procedure.

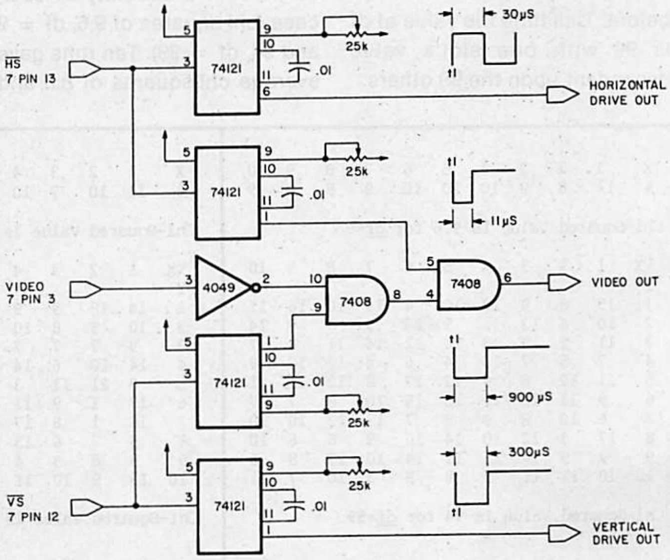

Fig. 1. The interface circuit.

Percent Probabilities							
99	95	90	50	10	5	1	0.1
Chi^2 for df=9							
2.1	3.3	4.2	8.3	15	17	22	28
Chi^2 for df=99							
69	77	81	99	117	123	134	147

Table 1. Probabilities in percent that the chi-square of a random set with the same degrees of freedom (df) will have a value greater than the one given.

therefore, not independent.

The third part of the program (lines 180-320) is similar to the second part except that it examines the set of random numbers as 1000 pairs to identify whether the value of the first digit of a number affects the value of the first digit of the next consecutive number in the set. Note that this approach supplants the other technique in that it examines overall distribution as well as pair relationships. It would *not*, however, reliably expose a pattern involving relationships between nonconsecutive numbers.

Each of the 1000 consecutive pairs of random numbers in the set is evaluated to determine into which of the 100 slots it falls. The slots are arranged in a 10 by 10 matrix, N2(Y,X), according to the value of the first digit, as before. The counter value for this slot is then incremented by one.

The y-axis of 1 to 10 denotes the first number in the set, and the x-axis similarly denotes the second. The average expected value in each slot is 10, and the chi-square is calculated as before. This time the value of df is 99 with one slot's value dependent upon the 99 others.

Application

The application of the program involves simply running it (it will take a minute or two) and then examining the values of chi-square obtained. Table 1 gives the probabilities associated with different chi-square values for df = 9 and df = 99. The value given is the probability in percent that the chi-square obtained is smaller than that expected from a random set of data.

Your values should hover around 50 percent. A probability of less than 10 percent indicates nonrandomness (especially if obtained repeatedly), and a value repeatedly greater than 90 percent indicates that the set is more evenly distributed than would be expected. The latter would occur, for instance, if the set consisted of equal frequencies of the ten digits (chi-square would be 0).

Three examples of the program's output are shown to illustrate a few final points. Example 1 is a straight test of our BASIC random-number generator and gives typical results near 50 percent probability in each case (chi-squares of 9.6, df = 9 and 94, df = 99). Ten runs gave average chi-squares of 8.8 and 96 (with standard deviations of 2.3 and 4, respectively, for the statistics-minded reader). Casual examination of the printed values in the second array indicates that they vary greatly from 10. However, the variation is what is expected statistically.

Example 2 shows the result of using RANDOMIZE before every twentieth use of the RND (1) function in line 40 instead of just once in line 20. Note that rather large inequalities of distribution occur in both tests, with probabilities of 1 percent and much less than 0.1 percent that the set of "random" numbers is actually random. Apparently, overuse of RANDOMIZE with Xitan's SuperBASIC is detrimental.

This result is unlike that obtained using Rogers' unnamed BASIC interpreter. Use of RANDOMIZE before each iteration of line 40 (not shown) results in chi-squares of several thousand due to most of the "random" numbers' falling into a few slots in each array.

Observation of the integral frequency percentages shown in the first array of Example 2 does not suggest great disparity from the expected value of ten. However, there is only one chance in 100 that this distribution is actually random. This illustrates the advantage of using a quantitative approach to data analysis. Chi-square tests are often used to demonstrate the significance of differences between the expected and observed frequencies of events in other applications; their probability values for different degrees of freedom can be found in statistical handbooks.

Example 3 demonstrates the advantage of examining the distribution of pairs. An alteration was made in the random-number-generating portion of the program, which made each twentieth random number in the set equal to the previous one (5 percent repetition). Such an alteration does not significantly affect the overall distribution of single values as shown by the chi-square value of 11 for the first array. However, the chi-square value for the second array is 190, giving a probability of much less than 0.1 percent that the array is random.

In fact, it is possible to detect the expected increase in values of the array on the diagonal from upper left to lower right, where X=Y. Some dependency of a random number on the one obtained previously would be a likely result of nonrandomness in a BASIC randomizing operation that carries a seed through the same series of manipulations for each random number generated. For this reason, numbers generated by this method are properly called "pseudorandom," since they are related by whatever operations are used in their generation.

We hope we've convinced you that such patterns of relatedness between consecutive numbers may be observed through the examination of pair distributions but cannot be found by examining only the distribution of single random numbers. ■

```
X    1  2  3  4  5  6  7  8  9 10
%   11  8  9 10 10 10  9  9  9  9

Chi-Squared Value Is 9.6 for df=9

\X   1  2  3  4  5  6  7  8  9 10
Y
1   15  8  9 19 10  4 14 10 16 11
2   10  6 13 11  9 13  6  6  8 14
3   11  5  9  8 10 12 14 11  9  7
4    7  5  7 15 14  6  8 12 10  9
5   11 12  8  9 12 17  8 12 11 11
6    9 11 10 12 10 15 10  8  7 17
7    6 10  8  9  9  7 13 12 10 10
8   17  3 12 10 14 10  9  6  6 10
9    9  9  9 11 11 14 10 13  8  6
10  10 11 11  8  8  9  3 10  7 11

Chi-Squared Value Is 94 for df=99
```

Example 1.

```
X    1  2  3  4  5  6  7  8  9 10
%   11 10  7 10  8  9 10  9 10 11

Chi-Squared Value Is 23 for df=9

\X   1  2  3  4  5  6  7  8  9 10
Y
1   14 18  9  9  7  9 12 10  6 16
2   10  9  8 10  6 18 17 11 12  8
3    9  7  7  7 11 11 10  3  6  8
4   14 10  6 14  9 11  8 18 10 13
5    8 21 11  3  8  4 14 10  9 11
6   17  1  9 11  3  5 11 18 10 11
7   16  1  8 17  3 11  8 12 18  6
8    4  7  6 13  3 14  4  6  7 20
9    9  8  5  4 15 13 17  8 12 19
10  13  9 10 11  5  7 18  6 11 10

Chi-Squared Value Is 2.0E+02 for df=99
```

Example 2.

```
X    1  2  3  4  5  6  7  8  9 10
%    9  9 11  9  9 10  9  9 11  9

Chi-Squared Value Is 11 for df=9

\X   1  2  3  4  5  6  7  8  9 10
Y
1   17  8 11 11  7 12  4  7  9  9
2    8 22  6  9 14  6  6  4 10  9
3    9  7 20 11 12  9 11 16  8  9
4    7  7 11 25  7 10  5  5  7  6
5   15  6 14  7 17 10  9  5  9  8
6   10 10  8 11  9 16 11 17 12 11
7    8  7  7 11  8 12 18  7 11  7
8   10  8  9  4  7 12 10 21 10  6
9    4  6 10  9 11 12  7  8 27 14
10  12  5 14 11  6  4  5 12  9 15

Chi-Squared Value Is 1.9E+02 for df=99
```

Example 3.

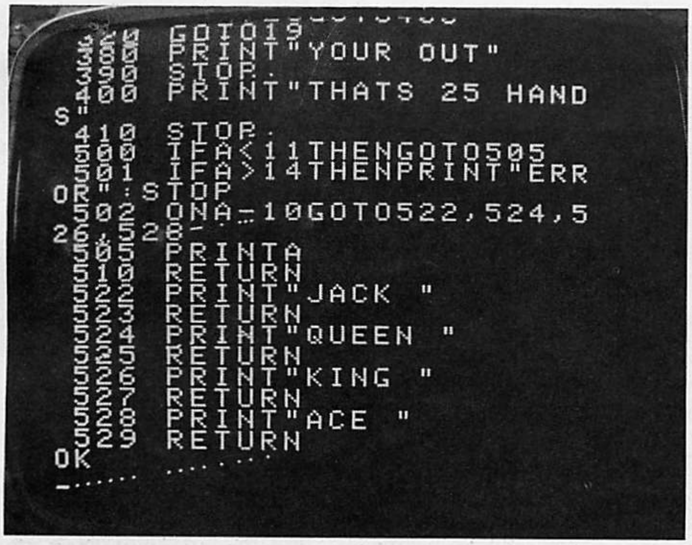

Photo 2. Diagonal bright spots affect the display.

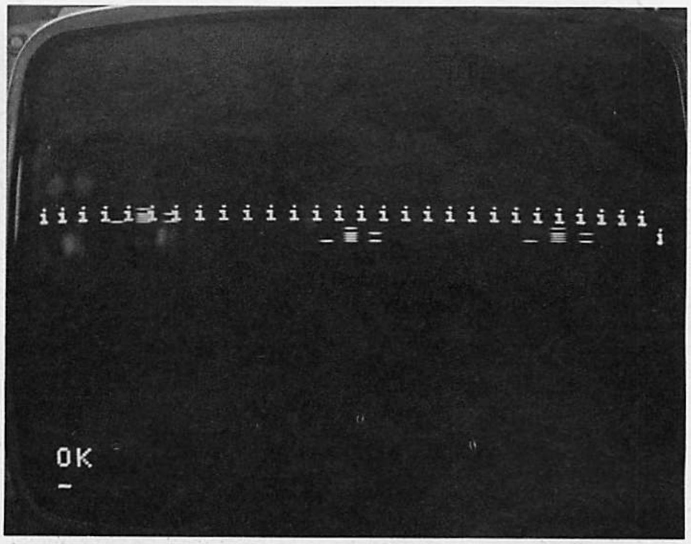

Photo 3. 27 characters per line with an additional four characters on retrace.

The shift-lock key must be in this position for programming and normal system operations.

With the shift-lock key up, the keyboard generates lowercase characters. However, the right and left shift keys act differently. Holding down the left shift key generates uppercase characters and numerals. Holding down the right shift key generates uppercase punctuation. Except when you want lowercase, keep the shift-lock key down. There is a 12-pin Molex connector beside the keyboard for hooking other keys, joysticks, etc., across the keyboard switches for games and other control functions.

The microprocessor is the popular 6502, which is used in KIM-1, Apple II and PET. The board has ten 2114s with sockets for eight more RAM ICs, which are available for $69 from Ohio Scientific. Eight 2114s are used as 4K user memory, giving enough memory to get started writing programs in BASIC. The other two 2114s form 1K of video memory, the contents of which are shown on the television screen.

Input and output consists of a composite video interface that will give you 30 characters by 30 lines on a television without overscan; you will only get 24 by 24 with an ordinary television set because of overscan. Overscan means the beam starts off screen and finishes off screen, both horizontally and vertically.

The other interface is a Kansas City Standard cassette interface, which operates at 300 baud. Ohio Scientific supplies two cables that plug into the back of the board: One plugs into the microphone jack and the other into the earphone jack of a cassette recorder.

To load a tape you type LOAD, start the tape and push RETURN. As it is loading, the screen displays each line of the BASIC program. This way you see what is actually being loaded, and you know immediately if it is loading incorrectly. To save a program, you type SAVE, push RETURN, type LIST, start the tape and push RETURN, causing the tape to store the program as listed, line by line. What could be easier? In all the tapes I loaded and unloaded, I did not have one error and I was using an inexpensive cassette player.

The input/output lines are all through two 12-pin Molex connectors across the back of the board. There is no parallel port. There is a 40-pin socket near the microprocessor IC for expansion. Ohio Scientific sells an expansion board 610 that plugs into this socket. The board mounts over the main board and contains 8K RAM with sockets for another 16K of RAM, a mini-floppy disk interface that will control two mini disk drivers, a real-time clock and expansion interface for a

Photo 4. The interface circuit solved the problem of the diagonal bright spots.

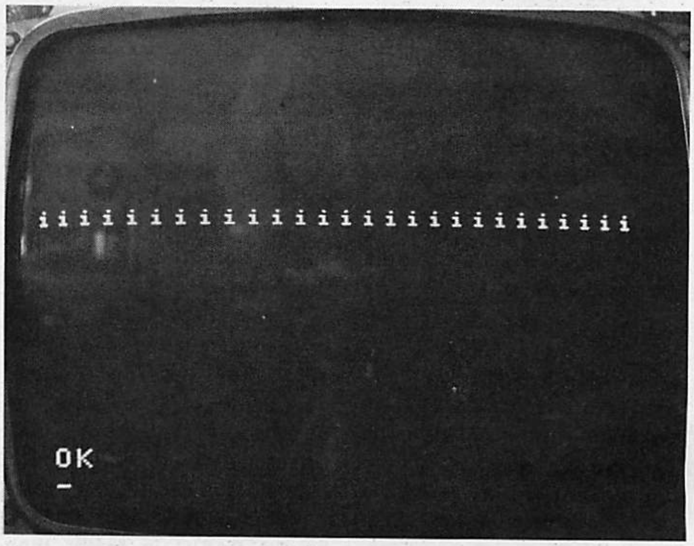

Photo 5. The interface circuit eliminated the retrace characters.

Photo 6. Graphics characters program.

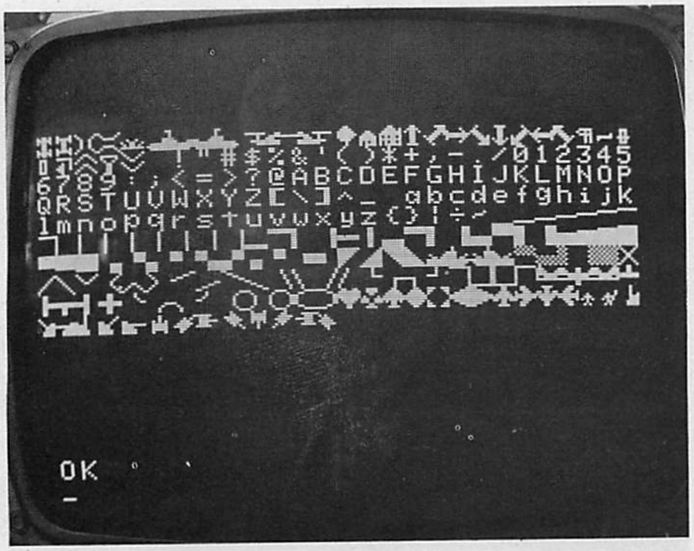

Photo 7. Sample run.

Model 620 bus adapter. The Model 620 is a connector board to connect the 610 to Ohio Scientific's 48-line bus backplane. Thus you can expand to use all of Ohio Scientific's 48-pin boards, including A/D, D/A, RS-232 port, parallel ports, PROM boards and prototype boards. All this makes it easy to expand to as big a system as you could want.

Firmware

Firmware is a read only memory (ROM), which has been programmed with software. The Superboard II has 10K of firmware in five 2K ROMs. One ROM has the system monitor, which contains input-output subroutines for a cassette, video display and keyboard.

Also contained in the ROM is a complete machine-code monitor that allows you to examine memory locations, load machine code, execute machine code programs and load machine code programs from cassette tape, but the directions do not say how to put machine code programs on the tape in the first place.

The other four ROMs contain 8K BASIC for the 6502 by Microsoft. This is a 6 1/2-digit precision BASIC with full scientific notation, trigonometry functions, string manipulations, logicals and many other useful features. Ohio Scientific claims its 8K BASIC is faster than other personal-computer BASICs. To check this out, I decided to use the benchmark programs by Rugg and Feldman published in the June 1977 *Kilobaud*. After running the programs five times each and averaging the times (I used an ordinary stopwatch), I got a total of 126.6 seconds for all seven of the benchmarks. This placed it third in a field of 25, so draw your own conclusions.

Ohio Scientific does not supply a handbook on how to program in BASIC, but only how to use their version of it. They do recommend several books on BASIC programming. What impresses me about their BASIC is that you don't have to use any spaces in the BASIC statements. This will save a lot of space-bar punching.

The Superboard II has 256 different graphics characters, including uppercase and lowercase, numbers, punctuation marks and all kinds of special graphics characters including several for use in playing games. Ohio Scientific's "The Challenger Character Graphics Reference Manual" provides an outline and decimal code for each of the 256 characters as well as a code number for each location on a 25 by 25 grid.

To put a character on the screen you merely type POKE (location code), (character code) punch RETURN, and the character appears on the screen at the location specified. "The Challenger Character Graphics Reference Manual" gives examples of how to move a character around the screen. By using an offset code, you can move the character in 12 directions on the screen.

So far Ohio Scientific has good marks for:

- fast delivery (maybe even a record)
- good packing
- good price (I paid only $279 for the board; they paid the shipping)
- good workmanship
- full-size keyboard
- 8K BASIC on ROM
- machine code monitor
- reliable and easy-to-use cassette interface
- 256 graphics characters.

Documentation

The documentation could be better. Instead of four pamphlets, there should be one user manual with better schematics. The schematics were about the worst I have seen, and I have been an electronics technician for five years. The schematics are presented as 13 separate drawings, two to a page. It is hard to trace a line that goes off the drawing. You must go through all the drawing looking for the letter and number code of that line. Since they can go to more than one page, you have to check all of the pages. Input/output lines are not labeled except by the edge-connector number. There are three exceptions to this: The microphone output of the cassette interface is labeled MIC, and the RS-232 interface is labeled RS-232 OUT and RS-232 IN. The two transistors of the RS-232 interface are not identified, except that one is an NPN and the other is a PNP.

Video Interface

The technical report and the salesman both said separate sync and video signals are available, but I could find nothing in the documentation or on the printed circuit board to back this up. This was important, because I have a "Ball Brother's" 12-inch CRT monitor that needs separate horizontal drive, vertical drive and video inputs.

In order to check the waveforms with a scope to see if they match the monitor's requirements, I had to first find the character generator, then trace the signal to the composite video output, which was labeled J2 pin 12, then trace back to where two inputs are labeled HS and VS. The vertical sync matched. The horizontal sync was inverted and too short.

By using a 74121 one-shot to invert and lengthen the horizontal sync, I received a readable display on the CRT (see Photo 1). As you can see, the display is quite readable. However, when programming and running programs, I kept getting diagonal bright spots (see Photo 2).

Using the POKE command to explore exactly how many characters I had on the screen, I discovered 27 characters per line. I also discovered an additional four characters on retrace (see Photo 3). This was allowable, since I could avoid poking characters in retrace locations, but the diagonal bright spots were not.

Examining the specifications for the monitor shows four waveforms: horizontal drive, horizontal blanking, vertical drive and vertical blanking. Since the monitor only has inputs for horizontal drive, vertical drive and video, I assumed the two blanking waveforms controlled the video reaching the monitor. I designed and built another interface (see Fig. 1) to try out this assumption. The interface worked, as you can see by comparing Photo 2 to Photo 4 and Photo 3 to Photo 5.

Photo 6 shows the listing of the program I wrote to show all of the graphics characters. Photo 7 is a run of the program. I changed the program to separate each character with a space first horizontally, then vertically and then both horizontally and vertically (Photos 8, 9 and 10).

With a normal television or monitor, you will get 24 lines by 24 characters, but since the screen is rectangular, the graphics characters are rectangular. This also happened with both of my interfaces. I did not find this acceptable. By adjusting the vertical frequency of the monitor, which stretched the display vertically, I was able to get a square graphics character. If you compare Photo 11, which was taken before I adjusted the monitor, to Photo 1 you should be able to see the difference.

Ohio Scientific could improve their video interface by having more characters per line, so that the graphics characters would look right. With 27 characters by 24 lines you have 648 total characters on the screen, but you have enough RAM for 1024 characters, so you are wasting 376 characters. Even with 30 characters by 30 lines for a total of 900 characters, you would still waste 124 characters.

Photo 1 shows the display after start-up procedure. You push four keys, BREAK, C, RETURN and RETURN, after the power is on to get into BASIC. Note the fourth line, 3327 BYTES FREE. The start-up firmware tests the RAM memory. The number 3327 tells you all of your 4K of RAM is good and you have 3327 words of memory to write programs in. The firmware uses 769 words of memory. With 8K of RAM, the screen would display 7423. If you have a program in memory and you push BREAK, to get back in BASIC without losing your program, you push W, and you are

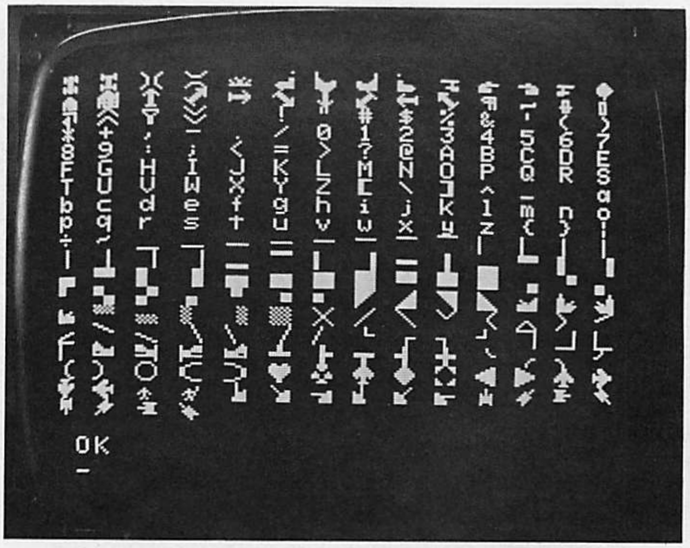

Photo 8. Program display characters separated by a horizontal space.

Photo 9. Program display characters separated by a vertical space.

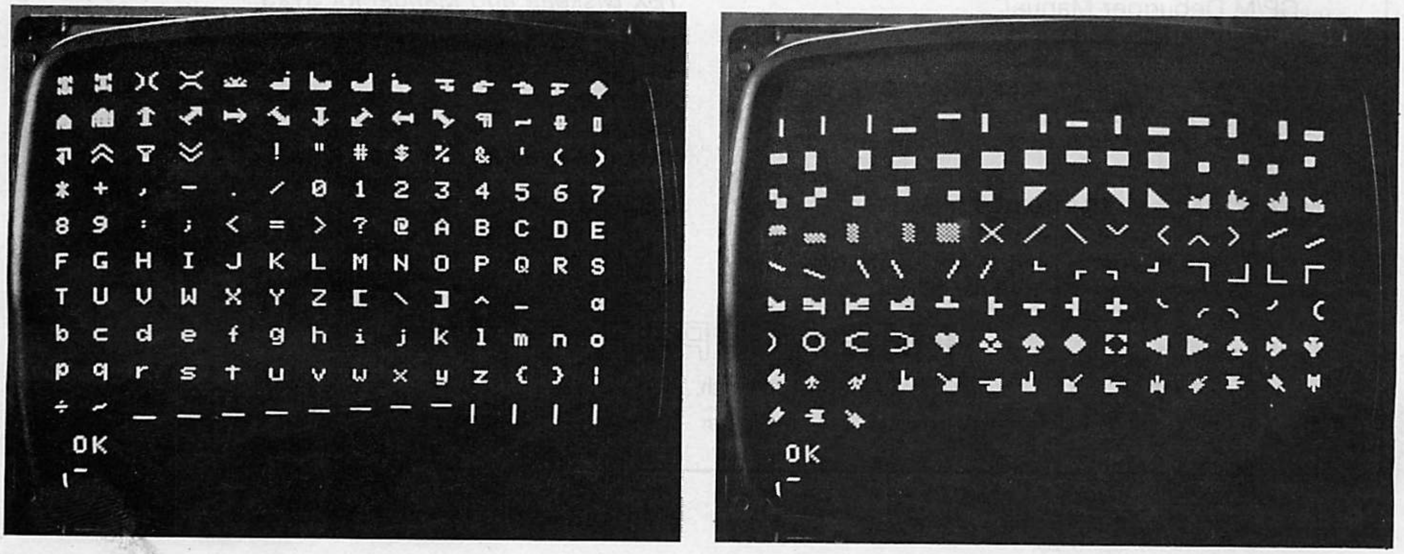

Photo 10. Program display characters separated by both horizontal and vertical spaces.

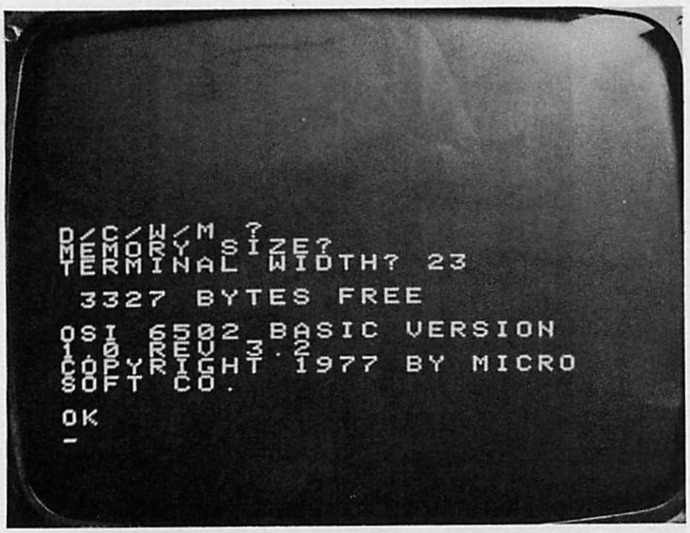

Photo 11. With a rectangular TV screen or monitor, you get rectangular graphics characters.

back in BASIC. Pushing BREAK and M puts you in the monitor so you can program in machine code.

Conclusion

Ohio Scientific's Superboard II computer system is less expensive than comparable systems, and its BASIC is fast. I think they will sell a lot of these, especially if they get them in the computer stores. I think the Ohio Scientific Superboard II is the best buy available for both beginner and expert. It is very inexpensive, especially if you already own a power supply, monitor and cassette recorder, as I did. It is also easier and less expensive to expand than other computer systems. I only wish they would supply better schematics and improve their video interface.■

References

Tom Rugg and Phil Feldman, "BASIC Timing Comparisons," *Kilobaud*, June 1977, p. 66.
"The Challenger IP Technical Report," Ohio Scientific pamphlet.
"Superboard II Challenger IP Users Manual," Ohio Scientific pamphlet.
"The 8K BASIC-in-ROM Reference Manual," Ohio Scientific pamphlet.
"The Challenger Character Graphics Reference Manual," Ohio Scientific pamphlet.

✓Reader Service—see page 179