SUPERBOARD II™

CHALLENGER 1P™

USERS MANUAL

Preliminary

Aug. 1978

Ohio Scientific Inc.

Table of Contents

UNPACKING AND ASSEMBLY INSTRUCTIONS

INTRODUCTION

This information outlines procedures to be followed during unpacking and assembly of Ohio Scientific microcomputer systems. Please follow these instructions carefully, preferably, read them completely BEFORE opening any cartons. You'll then be assured an up-and-running system with a minimum of problems. Please don't be guilty of the adage, "When all else fails, read the instructions"!

SYSTEM ARRIVAL

A CHALLENGER system may be delivered in from one to six boxes. Ohio Scientific normally ships via United Parcel Service (UPS). However, large or bulky items may require shipment by air or around freight carriers. Further, due to weight restrictions, the entire shipment may arrive over several days. Each package of the total shipment will be marked "(number) OF (total boxes)", i.e. 1 OF 3, 2 OF 3, etc. Equipment in each box will be accompanied by manuals and other materials pertinent to that equipment.

SHIPMENT EXTERNAL CHECK

Via a carton count, determine that you have the entire shipment. Inspect the boxes for signs of rough handling such as punctures, crushed sides, etc. If such damages are detected, check the contents of the box, preferably without removing the equipment. If the contents have sustained damage, this will make the determination of liability easier. In such cases, notify the carrier immediately.

UNPACKING

Carefully remove the system components from their boxes and SAVE ALL PACKING MATERIALS! These may be needed later to transport or ship components of the system. Most components require no further unpacking. However, a few components contain internal packing materials which must also be removed. These are listed below:

CHALLENGER C2-8P, CHALLENGER III - remove the 4 screws retaining the top cover (lower side edges), and lift the cover off. Compress and remove the foam from between the circuit boards. Cassettes, cables or other small components may be shipped within this cabinet. Remove these also. Be careful not to disturb any wiring, components, or PC boards. Replace the cover and screws.

FLOPPY DISK DRIVES - remove and save the cardboard dummy diskette which protects the disk head from vibration. The dummy should be re-installed when the disk drive is transported.

Other accessories such as CRT terminals, cassette machines, and line printers are shipped separately. Their boxes will contain the manufacturer's operating instructions, warranty cards, etc.

Check all unpacked components of the shipment to make sure the shipment is correct and complete.

WARNING!

NEVER OPEN ANY CABINET WHEN THE UNIT IS PLUGGED IN. NEVER ENERGIZE OR OPERATE ANY UNIT WITH THE TOP OR PROTECTIVE COVER REMOVED. YOU MAY CREATE A POTENTIAL SHOCK HAZARD BECAUSE THE COMPUTER AND PERIPHERAL EQUIPMENT HAVE EXPOSED 11Ø VOLT WIRING WITHIN. ALSO, THE UNSHIELDED LOGIC CIRCUITRY MAY EMIT RADIO FREQUENCY INTERFERENCE.

POLLED KEYBOARD NOTE

Several Ohio Scientific microcomputers employ a polled (software scanned) keyboard. These keyboards can be identified

by a latching "SHIFT LOCK" key immediately to the right of the "+;" key. The SHIFT LOCK must be latched in the "down" position before the machine can be reset and BASIC can be entered.

SETTING UP THE COMPUTER

Welcome to the world of personal computing! You now have a computer that was technically impossible just a few years ago! We at Ohio Scientific think you'll find your computer an interesting and entertaining device for years to come.

So that you can get the computer operating as quickly as possible, we have provided detailed instructions to assist you. Although a computer is a relatively rugged solid-state device, it may still be damaged if you fail to observe power supply, accessory, or safe-operating requirements. Therefore, follow the instructions carefully. Better yet, read all the instructions BEFORE you turn to the computer. Once you are familiar with these procedures, you can explore other areas of personal computing at your own pace.

The information we'll cover concerns the following:

1. Supplying a source of power to the computer. The Challenger 1P requires a 3-wire grounded 110V outlet. The Superboard II requires a 5V @ 3A DC power supply.
2. Connecting a display device such as a closed-circuit TV monitor or ordinary television.
3. Use of a cassette recorder to play pre-recorded programs into the computer.
4. Use of a cassette recorder as a program storage device.
5. Computer activities you can turn to after you have mastered these basic skills.

POWER SUPPLY CONNECTIONS

CHALLENGER 1P ONLY

1. The Challenger 1P must be plugged into a grounded 3-wire 110V receptacle. This assures that the computer's cabinet is thoroughly grounded, which protect both the computer and you from possible damage or shock.

2. Optionally, you can run a wire from the computer's cabinet to a good ground such as a cold water pipe, and ONLY THEN use a two wire adapter on the computer's power cable.

3. THESE ARE THE ONLY ACCEPTABLE POWER CONNECTIONS FOR THE CHALLENGER 1P.

WARNING

ANY OTHER POWER CONNECTIONS FOR THE CHALLENGER 1P OTHER THAN OUTLINED ABOVE MAY ULTIMATELY PRESENT A SHOCK HAZARD OR LEAD TO COMPUTER DAMAGE VIA STATIC DISCHARGES. SUCH DAMAGES ARE SPECIFICALLY NOT COVERED UNDER THE WARRANTY.

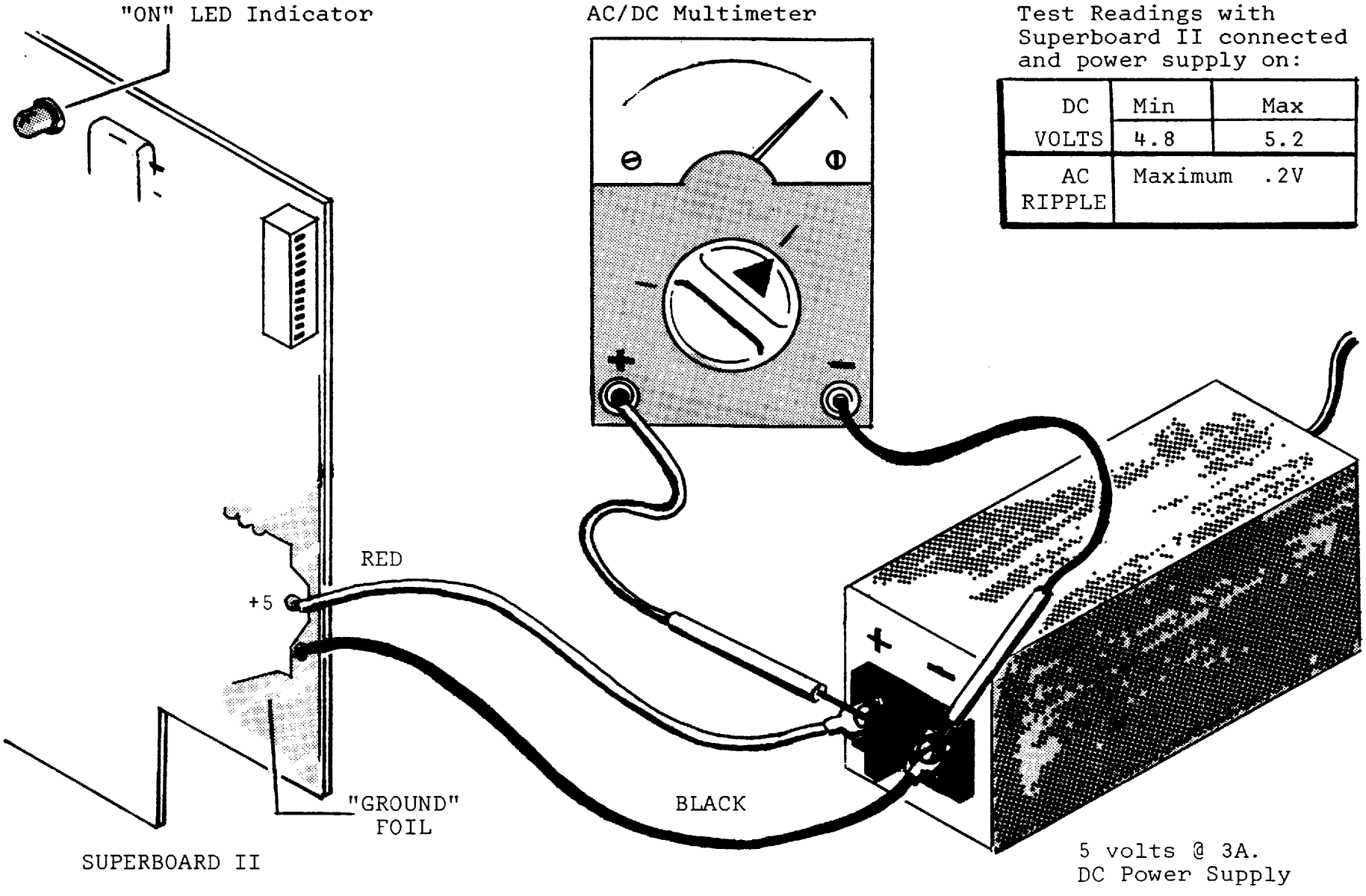


FIGURE 1. SUPERBOARD II POWER SUPPLY CONNECTIONS

SUPERBOARD II ONLY

1. Consult Figure 1, Superboard II power supply connections. The power supply MUST BE a 5 volt @ 3A (minimum) regulated supply (± 5% maximum ripple). No other supply will be adequate. You'll also need an AC/DC multimeter.

2. Choose a work area which is free of all foreign matter, paper clips, or any other conductive materials. The computer or power supply could be damaged if these come in contact with the foils on the board.

3. Observe that the Superboard II should not be treated roughly, flexed, bent, or otherwise abused.

4. Make sure the power supply is not plugged in.

5. Connect the RED and BLACK wires from the Superboard II to the + and - terminals.

6. Set the AC/DC multimeter to a DC voltage range to let you measure 5 volts accurately. (A range of 0-5, 0-6, or 0-10 volts is adequate.)

7. Next, you will observe the "ON" LED indicator on the Superboard II, and the power supply's voltage under load. Briefly turn on the power supply and observe that the "ON" LED glows. If not, turn off the supply immediately and check your power supply leads. They may be reversed. Go back to Step 1.

8. Again, turn on the power supply and measure the DC voltage. It must be between 4.8 and 5.2 volts. A reading less than 4.8 indicates that the power supply probably lacks the required voltage or current capability. A power supply delivering more than 5.2 volts may damage the computer. Turn off the power supply.

9. Set the multimeter to measure a voltage of about 0.5 volts AC (probably the lowest AC range).

10. Turn on the power supply and measure the AC voltage. This represents ripple and must not exceed 0.2 volts AC.

11. If everything checks out, proceed to the video display hook-up.

VIDEO DISPLAY CONNECTION

There are three different methods of attaching a video display to the Superboard II and Challenger 1P computers. These are outlined as follows:

1. Preferred method - connect the supplied computer video cable to the high impedance (Hi-Z) input of a closed-circuit TV video monitor. Ohio Scientific offers the Model AC-3 12" monitor which is ideal for this application. The unit doubles as a television when the video cable is disconnected.

2. Connect the supplied computer video cable to an "RF modulator" which is, in turn, connected to a standard television's antenna terminals. RF Modulators are inexpensive and allow you to use almost any television with your computer.

3. Have a standard AC transformer-operated television modified to accept direct video entry. This requires special safety precautions which will be explained later.

CLOSED-CIRCUIT VIDEO MONITOR CONNECTION

1. Refer to Figure 2. Attach the supplied video cable to the computer as shown. With the Superboard II this cable is part of a computer video/cassette recorder cable assembly.

2. Connect the other end of the cable to the high impec input of the video monitor. The AC-3 monitor has a RCA-type phono jack input. On other monitors, a hig impedance - low impedance selector switch is sometir present, or there may be two or more inputs. Consu the manufacturer's instructions.

3. Observe the manufacturer's power recommendations. the monitor has a 3-wire grounded plug, connect it a properly grounded 3-wire AC outlet.

4. Turn on the computer and monitor.

5. Allow the monitor to warm-up. You should see the s filled with random graphics characters, alphabet, e

6. If necessary, adjust the VERTICAL and HORIZONTAL cc to obtain a stable picture.

RF MODULATOR/STANDARD TV CONNECTION

1. Refer to Figure 2. Review the manufacturer's instructions included with the RF modulator.

2. Connect the computer video cable to the computer as shown. With the Superboard II, this cable is part of a computer video/cassette recorder cable assembly.

3. Connect the video cable to the RF Modulator.

4. Connect the modulator to the television's antenna terminals (consult modulator instructions).

5. Plug in the television and computer.

6. Turn on the computer, television, and modulator (consult modulator instructions).

7. At this point you will have to select the proper TV channel and possibly adjust the television's fine tuning slightly (consult modulator instructions).

8. When the television warms up you should observe a screen filled with random graphics characters. If the picture is not stable, adjust the television's VERTICAL or HORIZONTAL controls as needed.

MODIFICATION OF A TELEVISION FOR DIRECT VIDEO ENTRY

1. A standard television may be modified to act as a video monitor. However, this conversion requires detailed knowledge of television circuitry, and will likely require a schematic of the television to be converted. Consult a qualified service person.

WARNING

ANY TELEVISION CONVERSIONS MUST BE PERFORMED ONLY BY A QUALIFIED PERSON, SUCH AS A TV SERVICEMAN. LETHAL VOLTAGES ARE PRESENT WITHIN THE TELEVISION. INCORRECT CONNECTIONS MAY PRESENT SHOCK HAZARDS OR DAMAGE THE COMPUTER. SUCH DAMAGE IS NOT COVERED BY THE WARRANTY.

2. The television to be modified must be an AC-transformer operated television. Several solid-state TV sets are now available which can be operated from 110V AC, or from a 12 volt source such as a car cigarette lighter. These televisions can usually be converted easily. Some older "AC-DC" tube-type televisions are "hot chassis" types; that is, one side of the power line is connected to the chassis. These televisions do not have transformers and MUST NOT be used for conversions. Shock hazards are

present, and the computer may be damaged. These televisions are suitable for RF modulator operation. Refer to that section in this manual.

3. More characters per line can be displayed on the screen if the picture is "shrunken" slightly. On most 110V AC/ 12V DC televisions, this can be accomplished by adjusting the television's power supply regulator to give a lower voltage. Brightness will also be diminished, but this can be restored via the TV BRIGHTNESS CONTROL. Refer this adjustment to the service person at the time of conversion.

4. When the power supply voltage is adjusted, the picture may require re-centering. Equal borders can be restored to the screen by adjusting the picture tube centering coils. Refer this adjustment to the service person at the time of conversion.

5. When the television has been modified, it may then be treated as a video monitor and connected to the computer. Refer to that section of this manual.

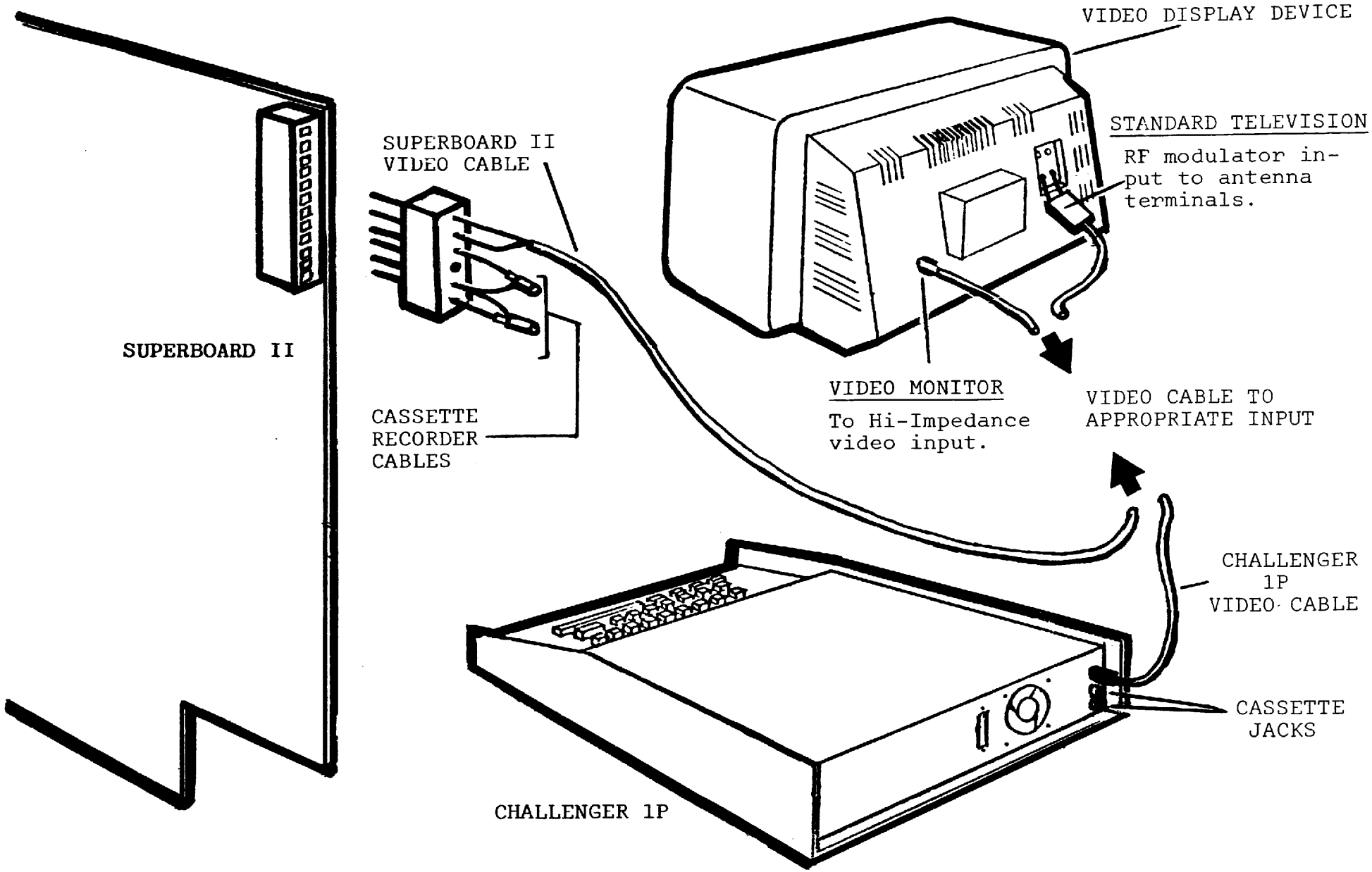
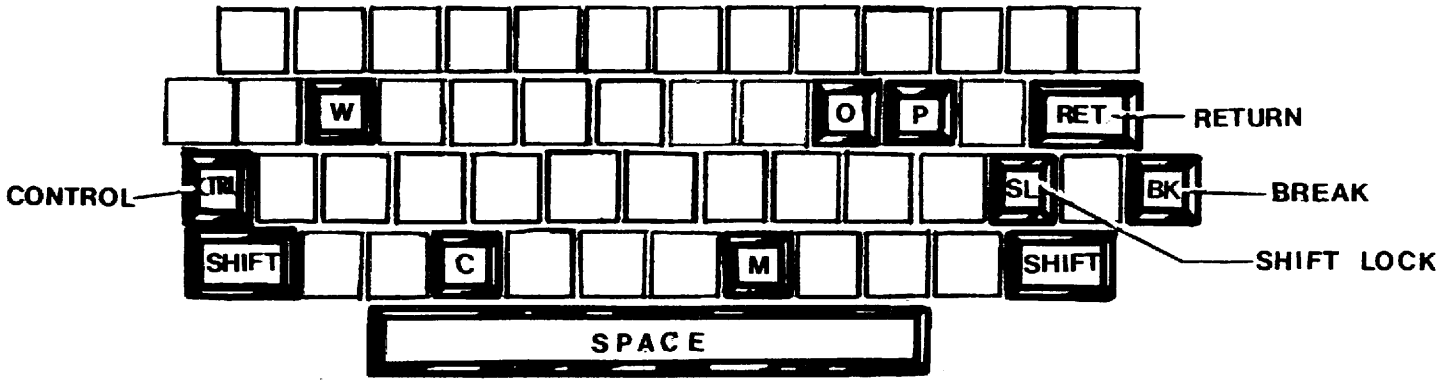


FIGURE 2. CHALLENGER 1P AND SUPERBOARD II VIDEO CONNECTIONS



1. <> - Brackets - Instruct user to press key whose label is contained between the brackets. DO NOT type in word between brackets.
2. SHIFT LOCK - (latching key) - must be in the locked (depressed) position before BASIC may be entered; or capital letters, numerals, etc., may be entered.
3. <BREAK> - Places computer in the "RESET" state any time after system is powered up.
4. C - May be pressed after <BREAK> . Initializes computer and clears system RAM.
5. W - May be pressed after <BREAK> except when computer is first powered up (C must be used). Initializes computer, DOES NOT clear system RAM. Any programs in RAM are preserved.
6. M - May be pressed after <BREAK> . Initializes computer, clears system RAM. Computer enters machine language monitor.
7. < SPACE > - provides a space when pressed.
8. < RETURN > - must be entered after a line is typed. Typed material is then stored in program memory space.
9. < SHIFT O > - Press <SHIFT> first, add O - erases last character typed.
10. < SHIFT P > - Press <SHIFT> first, add P - erases current line being typed. Provides a "@" carriage return and line feed.
11. < CONTROL C > - Press <CONTROL> first, add C. Program listing or execution is interrupted, "BREAK IN LINE XXX" is printed.

FIGURE 3. OPERATION NOTES - OSI POLLED KEYBOARD.

GETTING YOUR COMPUTER "UP AND RUNNING"

INTRODUCTION

These instructions will help you get your computer running in the computer language "BASIC". This language is permanently stored in the computer's memory, and can be quickly brought up when the computer is turned on.

Several of these instructions for bringing up BASIC contain words or letters which are bound by brackets "<" and ">", such as <BREAK> and <C>. The brackets indicate that a keyboard key labeled with the word or letter must be pressed. Do not type in a word contained between the brackets letter-by-letter. See Figure 3.

BEFORE YOU POWER-UP...

1. Check that all power supply connections are correct.
2. Make certain that your video monitor or television is connected to the computer properly.

GETTING INTO BASIC

These instructions should be followed very closely. If, at any time, the computer or television/monitor does not respond as indicated, turn off the power to both and review all hook-up procedures, wiring, etc.

1. Turn on the computer.
2. Turn on the television or monitor. After a short warm-up you will observe the screen filled with random characters.
3. Press <BREAK>. The prompt C/W/M? or D/C/W/M? will appear in the lower left corner of the screen.
4. Press <C>. The screen will scroll up one line and ask "MEMORY SIZE?"

5. Press <RETURN> . The screen will scroll up another line and ask "TERMINAL WIDTH?"

6. Press <RETURN> . The computer will reply:

```
XXXX BYTES FREE

OSI 6502 BASIC VERSION 1 REV. 3.2

COPYRIGHT 1977 BY MICROSOFT CO.

OK
```

Pay particular attention to the first line, "XXXX BYTES FREE". "XXXX" represents a memory test which is performed each time the computer is turned on. Typically, "XXXX" is "3327" for a computer with 4K RAM, and "7423" for a computer with 8K RAM. If you get a number other than the correct one for your system, do not use the computer. Refer to your authorized Ohio Scientific dealer for help.

PROGRAM EXAMPLE

The following program example demonstrates some of the more fundamental concepts of BASIC. This program may be entered once the computer replies "OK". Enter the program exactly as it appears, including all punctuation, etc.

```
10 PRINT "HELLO!  I'M YOUR NEW COMPUTER!" <RETURN>
20 PRINT <RETURN>
30 END <RETURN>
```

Now, check the program to be sure you have entered it correctly. Type in the word LIST and <RETURN> . This instructs the computer to print out the program as stored within the computer's memory.

LIST <RETURN>

To have the computer execute ("RUN") the program, type in:

RUN <RETURN>

The computer should then print:

HELLO! I'M YOUR NEW COMPUTER!

The BASIC language makes it easy to modify ("EDIT") a program. Errors within a line may be corrected by retyping the line. Additional statements may be incorporated into a program by sequencing the new line numbers within the existing program. The following additions to the example program demonstrate these editing concepts.

5 FOR X=Ø TO 3Ø <RETURN>

25 NEXT X <RETURN>

To examine the program as amended, type LIST <RETURN>.

To execute the new program, type RUN <RETURN>.

Refer to one of the many BASIC programming texts now available for an in-depth study of BASIC.

CASSETTE STORAGE TECHNIQUES

CONNECTION

A standard cassette tape recorder, such as the Model AC-2 offered by Ohio Scientific, may be used for program storage and playback. The recorder must have a microphone input jack and an audio output jack (usually labeled "EARPHONE" or "SPEAKER"). The cassette recorder should be connected to the C1P or Superboard II computers as shown in Figure 4 or 5. The necessary cables are supplied with the computer.
Note that the placement of the microphone and audio output jacks may vary with different cassette recorders. If you plan to use a recorder other than the AC-3 be certain your connections are correct.

PLAYING BACK A PROGRAM

The following steps show how to load the computer with a program stored on cassette.

1. Check for correct cable connections between the recorder and computer.
2. Rewind the cassette so that the tape "leader" is visible on the take-up spool.
3. Turn on the computer and get into BASIC as indicated by the letters "OK" in the lower left corner of the screen.
4. Type in NEW <RETURN> . This erases anything now stored in the computer.
5. Type LOAD. Do not press <RETURN> .
6. Turn on the recorder to PLAY the tape.
7. As soon as the tape (dark brown) begins to wind onto the spool, press <RETURN> .

8. Shortly, the program will begin listing on the screen. When program loading is complete, the following will appear in the lower left corner of the screen:

```
OK

S↵ ERROR

OK
```

9. Press <SPACE> .

10. Press <RETURN> .

11. To inspect the program, type in LIST <RETURN> .

12. To execute the program, type RUN <RETURN> .

RECORDING A PROGRAM

These instructions show how to record a program contained in the computer's memory.

1. Check for correct cable connections between the recorder and computer.

2. Use a new or thoroughly erased cassette. This will minimize "noise" and other problems associated with old cassettes.

3. Have your program ready in the computer.

4. Rewind the cassette so that the tape "leader" is visible on the take-up spool.

5. Type SAVE <RETURN> .

6. Type LIST. Do not press <RETURN> .

7. Turn on the cassette recorder in the RECORD mode. When the tape (dark brown) begins winding onto the take-up spool, wait 5 seconds and press <RETURN> .

8. Observe the program listing on the screen. When the last line of the program is listed, wait a few seconds and turn off the recorder.

9. Type in LOAD <RETURN> .

10. Press <SPACE> < RETURN> .

11. Label the cassette. If you wish to protect the contents from accidental erasure, break out the appropriate "record protect" tab located on the cassette's rear edge.

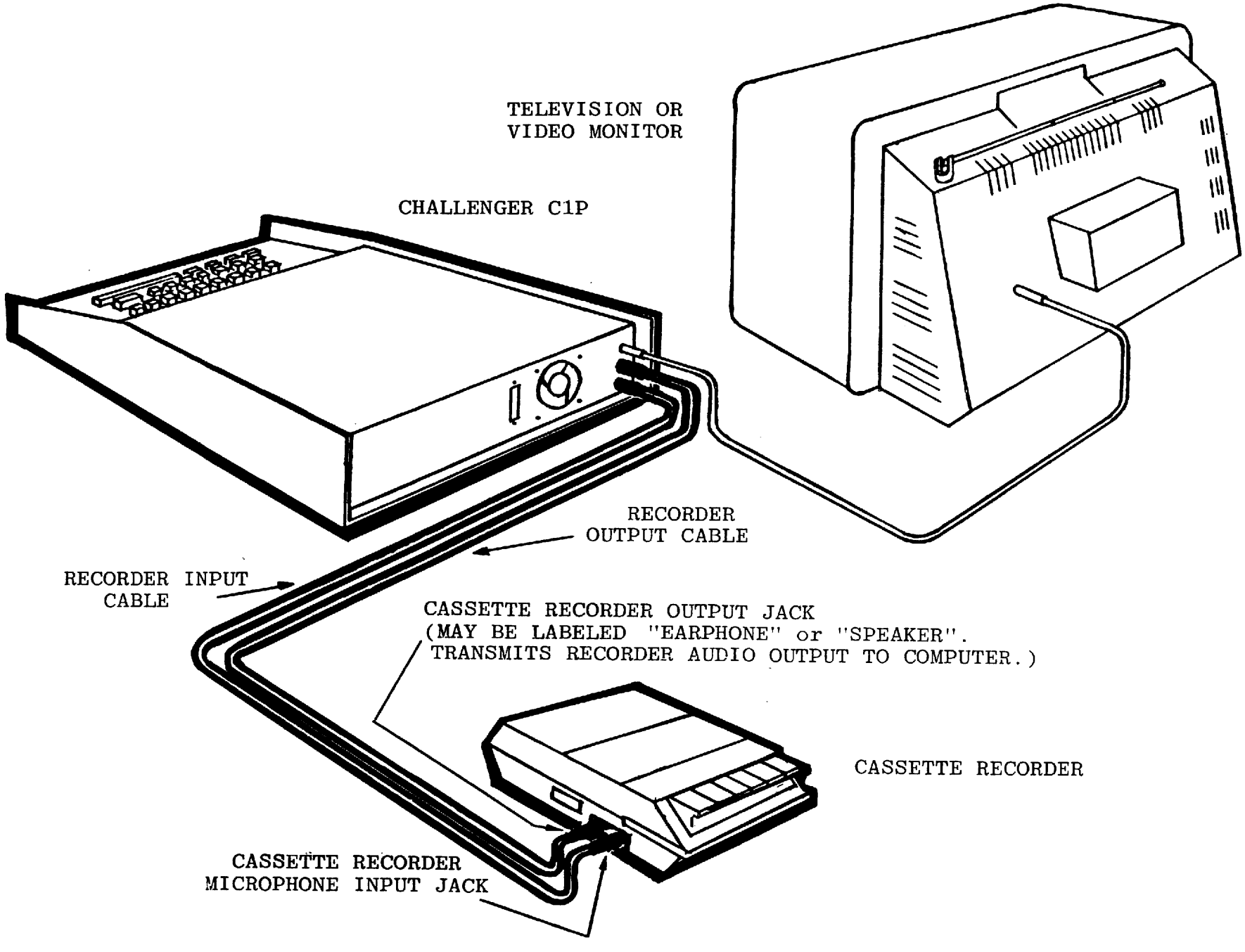


FIGURE 4. CHALLENGER C1P CASSETTE RECORDER CONNECTIONS

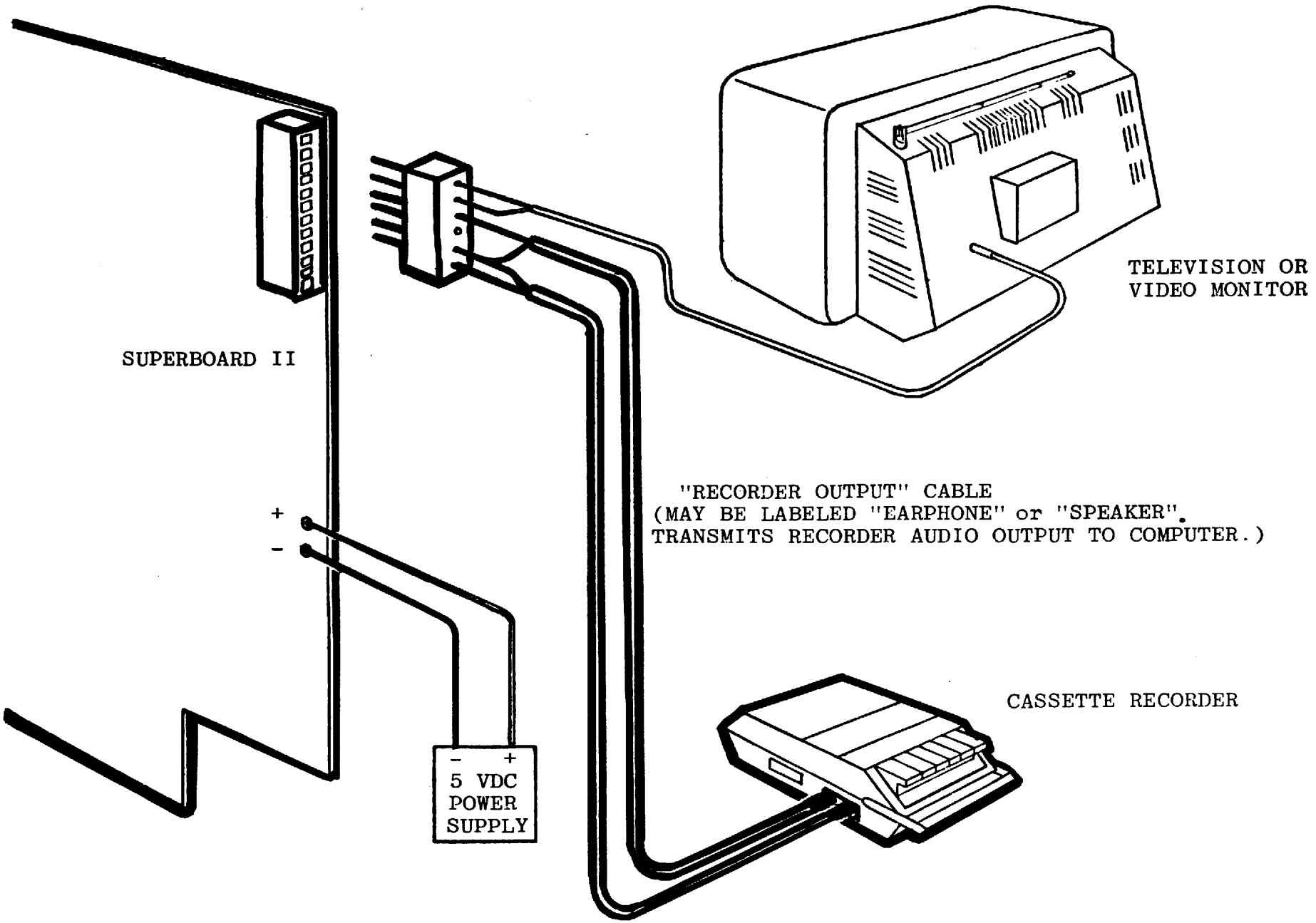


FIGURE 5. SUPERBOARD II CASSETTE RECORDER CONNECTIONS.

Using BASIC Program Cassettes

At this point, you have set up your computer system and should be able to run and utilize commercially available program tapes for the Challenger 1P and Superboard II. Several programs on cassette are provided with the unit and a large library of applications software is available from Ohio Scientific and your local Ohio Scientific dealer. Your Ohio Scientific computer can provide you with years of enjoyment and benefit by utilizing commercially available applications software alone. However, you may wish to learn how to program yourself and may even wish to explore and expand its hardware capabilities. The following sections of this manual are concerned with expanding your horizons in both programming and the hardware capabilities of the computer. The next section specifically deals with an introduction to the programming language BASIC. This section should be used in conjunction with the accompanying Ohio Scientific BASIC Reference Manual and possibly a text book on the programming language, BASIC.

INTRODUCTION TO SMALL COMPUTER SOFTWARE

In order for a computer to perform even simple operations, it obviously needs a means by which the user can communicate instructions to it. Any such means which consists of a set of rules to convey information, is called a computer language. The numerous languages in use today offer a wide range of specific applications and varying degrees of understandability for the user. They can provide direct communication with the computer at the complex level of machine language, or enable the programmer to use an indirect communication by means of a higher level language which corresponds more closely to human speech.

Machine level languages are really the most practical device from the computer's point of view, because when you use them, you are really speaking the computer's own jargon, and are thus making more efficient use of memory space. On the other hand, when you use an upper level language, every instruction you give, in what resembles "plain English", has to be converted into one or more separate instructions in a machine level language. Therefore it is obviously less wasteful in terms of time and effort to write in machine language, and skip translations from other languages altogether. The major drawback in machine languages, however, is that they are difficult for the average person to learn.

Machine languages consist of binary codes used in all of the commands which are entered by the programmer. These codes are 8-bit groups of on-or-off switches, which in various combinations, serve as instructions for the computer. Although the majority of users will have no need or desire to learn these combinations, there are some who, for one reason or another, will want to program their computer directly. Since it is quite troublesome to commit several dozen combinations of numbers to memory, a system of abbreviations (mnemonics) has been devised which exactly correspond to machine language instructions. The program which converts these mnemonics into machine language is called an Assembler. By following the instructions provided with your computer, you can make use of the Assembler and write a program in mnemonic code. This is directly translated into the binary object code which the machine understands. After you gain proficiency, you can even begin to use the actual object code to do your programming, examine and change memory locations, etc., and thus be in ever greater control of your machine.

Upper level languages, in contrast to machine level languages, are much easier for humans to master. Nevertheless, every upper level language has to originate at the machine language stage, and usually represents a long, tedious effort on the part of the author or authors of that language. Probably the most common upper level language is BASIC (Beginner's All-Purpose Symbolic Instruction Code). An 8K version of BASIC written by Microsoft, Inc. (i.e., it occupies 8×2^{10} locations in memory) is used in all of OSI's 6502 computers. Because of BASIC's popularity, simplicity, and versatility, OSI has made it a standard feature in its product line, either by placing it in a computer's permanent memory (also known as Read Only Memory [ROM], which does not "forget" once the power is turned off), or by reserving special tracks for it on floppy or hard disks. In addition, in OSI products, BASIC is always immediately available to use, because it comes up automatically the instant the computer is reset. Therefore, the programmer is free of the burden of manually bringing in BASIC, which would demand that he be thoroughly versed in the computer's internal thinking processes and machine language. The fact that BASIC comes up automatically is very convenient for computer programmers, most of whom probably have programs they would like to run or write in BASIC.

There are a large number of publications available which describe in detail the commands and functions of BASIC. While this introduction can in no way duplicate

such excellent manuals as Schmidt's outline series Programming with BASIC (McGraw-Hill), it can at least give you some insight into the method for writing your own programs in BASIC.

Refer to the instructions provided with your individual unit to bring up BASIC in the OSI Challenger. Establish the memory size and terminal width for your particular program. When you see an OK appear on your video monitor or terminal, the computer is ready to start accepting BASIC commands from the keyboard.

Every statement in your program must begin with a statement number. These need not be typed in numerical order, since the computer will automatically rearrange them according to statement number when you have finished typing the program. But they must be numbered in the same order in which they are to be run. In OSI's 8K BASIC for the 6502, a variable can consist of one or two characters. If longer variables are to be used, BASIC will recognize only the first two characters. The first character in a variable must be alphabetic. The second character, if present, may be either alphabetic or numeric. Functions, commands, etc., already used by BASIC must not be employed as variables. In order to set a variable equal to a desired value, e.g., Z equal to 10, you use the LET statement, as follows:

```
(line number ⟶) 20 LET Z=10
```

Since LET is optional in OSI's 8K BASIC, you may also type:

```
20 Z=10
```

You may wish the value of the variable to change each time you run the program, without having to rewrite the whole program every time. To take advantage of the option to alter the values of variables, you make use of the INPUT statement, for example, as follows:

```
10 INPUT A,B,C
20 LET X=A
30 LET Y=B+C
```

In this way you can cause X and Y to take different values each time the program is run. Later, when you do run the program, you will see a ? on the terminal. You then type the values for A,B, and C which are relevant to the particular program. If there are no other INPUT statements in the program, it will begin to run immediately with the values you have entered, unless some built-in control prevents this. If the program contains additional INPUT statements, BASIC will keep asking you (by means of a ?) to input whatever data it needs to run the program, until each INPUT statement has been answered.

It can be that a variable has a value which is to change at a regular rate during the course of a single program run. This will require you to set up a loop which makes calculations using these increasing or decreasing values each time a new value is employed. For this you need to use a FOR-NEXT loop. This loop begins with a FOR statement and ends with a NEXT statement. The FOR statement identifies the initial and final values of the variable in question, and includes the constant amount of increase or decrease:

```
100 FOR Z=10 to 20 STEP 3
110 LET A=Z+(2*4)
120 NEXT Z
130 (resumption of program)
```

Step 3 means an increment of 3 upon each pass through the loop. Therefore, the above FOR-NEXT loop will be run four times, namely, when Z=10,13,16, and 19. When the value of Z exceeds 20, BASIC resumes the program by going to the first statement following the FOR-NEXT loop. In addition to signifying the end of the loop, the NEXT statement also contains the variable identifying which loop it terminates. As you may later discover, this is most useful in nesting one loop inside another.

Sometimes you will want the program statements to be run in a different order, if a certain condition is met. In order to change the order of execution, you may use the IF . . . GOTO statement, for example,

```
100 IF X=10 GOTO 150
110 (another program line)
140 (another program line)
150 LET Y=X+5
```

Here, the IF . . . GOTO diverts execution to a non-consecutive statement, line 150, omitting lines 110 and 140, provided only that the value of X is equal to 10. If X is not equal to 10, the program would resume with line 110. A simple GOTO command may also be employed without an accompanying IF, if no condition must first be met.

An IF . . . THEN statement is used to jump to a statement other than the one directly following. It can also be used to issue any other statement allowed in BASIC. For example,

```
100 IF X>10 THEN PRINT "X IS GREATER THAN 10."
```

This will cause the terminal to display X IS GREATER THAN 10 only if X>10. The program will then proceed as normal, with the next consecutive line. If X≤10, the program will, of course, proceed as normal, ignoring the PRINT command.

A PRINT statement will cause the terminal to display whatever follows. If you type:

```
100 PRINT A
```

the value of some previously defined variable A will be printed. If you type:

```
100 PRINT "A"
```

the simple letter A will be printed.

The END statement terminates the program and allows you to run the program, change it, or start to write a new program. As in the case of the LET statement, the END statement is optional in OSI's 8K BASIC.

If you want to erase a program and start a new one, simply type NEW and enter your next program.

BASIC is provided with a large number of mathematical functions, such as sine [SIN(X)], square root [SQR(X)], and absolute value [ABS(X)]. These functions automatically cause the computer to calculate the pertinent value without figuring by the user. For example, in the following statements:

```
100 X=121
110 PRINT SQR(X)
```

the value of the square root of 121 will appear on the terminal when the program is run.

At any time while entering your program, or after you have finished entering it, you can list all the statements up to that point by typing LIST. You can thus list the whole program, or by typing a specific line number after LIST, such as LIST 140, you can display just the one line. If you desire to see a certain block of program lines only, then you can specify the desired range, such as:

```
LIST 100 TO 200
```

If you want to correct a line previously typed, simply type the correction, using the same line number. It is recommended that you number the program statements by jumps of 10 rather than consecutively, so that you can later easily insert additional lines if you wish. To do this, type a line number which falls between the interval where you want the new statement to appear, and add the missing line. If you want to delete a line, simply type the line number, then ⟨return⟩. By using the LIST command, you can easily verify any changes you have made. This will cause every program line to scroll up the terminal, with each line number in consecutive order. If you want to stop the scrolling, type Control-C, examine the listing to your satisfaction, then type CONT (=continue), after you see BREAK IN LINE XX on the terminal.

The following example gives an illustration of editing procedures: Suppose you want to modify the following statements:

```
 90 INPUT A
100 LET X=2*A
110 PRINT "THIS IS A PROGRAM."
120 PRINT "EXAMPLE"
```

If you want to insert a line Y=A between lines 90 and 100, you could, at this point, type:

```
91 LET Y=A
```

If you want to delete line 110, simply type 110, then ⟨return⟩. If you want line 100 to read LET X=3*A, simply type the correction, using the same line number. At any time you could confirm the alteration by typing LIST.

Following these corrections, if you are ready to run the program, type RUN ⟨return⟩. If your program contains any INPUT statements, you will now see a ? on the terminal. Type in the data desired, as explained above, and the program will run. Following program execution, you can start over again by typing RUN, or enter a new program by typing NEW.

The following sample program demonstrates the INPUT, LET, PRINT, GOTO, and END statements, the FOR-NEXT loop, and the IF . . . GOTO command, as well as the SQR function.

Problem: Print the square root of a number; increase the number by five six times, and each time print the square root. If the largest square root is less than twice the first square root, indicate this. Otherwise, indicate only the fact that the program prints square roots.

After the programmer has typed the above program in BASIC, he will see an OK on the screen, signifying that the computer is ready for the next command from the user. If he wants to run the program, he types RUN. The computer will show a ? on the terminal. The user types on the keyboard that number with which he wants to begin the program. The six values (with constant increments of 5, see line 40) will scroll up the screen, each accompanied by its square root. If by chance you have made an error in typing (not including improper spacing), you

will probably see an error message on the screen. If you do, simply edit the line containing the error, as explained above. You can always run the program again by typing RUN. Here are the lines of the program:

```
 10 INPUT A
 20 LET Y=SQR(A)
 30 LET Z=A+30
 40 FOR X=A TO Z STEP 5
 50 PRINT "THE SQUARE ROOT OF";X;"IS";SQR(X)
 60 PRINT
 70 NEXT X
 80 IF SQR(Z)<2*Y GOTO 110
 90 PRINT "THIS PROGRAM PRINTS SQUARE ROOTS."
100 GOTO 120
110 PRINT "THE LAST ROOT IS LESS THAN TWICE THE FIRST ROOT."
120 END
```

All of the special features of Ohio Scientific's 6502 8K BASIC are described in the OSI 8K BASIC Users Manual. For a more fundamental introduction into BASIC, refer to any of the following books:

Gottfried, B.S.: Programming with BASIC, Schaum's Outline Series, McGraw-Hill, New York, 1975.

Gottfried, B.S.: BASIC Programmer's Reference Guide, Quantum Publishers, New York, 1973.

Greunberger, F.: Computing with the BASIC Language, Canfield Press, San Francisco, 1969.

Kemeny, J.G. and T.E. Kurtz: BASIC Programming, 2nd ed., Wiley, New York, 1971.

Short BASIC Programs

The following short BASIC programs are provided here to allow you to gain some experience with your computer through fully debugged programs which are known to be working. These programs in no way depict the total capability of your computer. They are simple programs which are very short to facilitate manual entry. Each of the programs can be entered in your computer as listed. Remember to type NEW before entering each program. This clears out the computer's workspace. You can substitute a ? for the word PRINT. Ohio Scientific's 8K BASIC allows you this particular shorthand notation wherever the word PRINT occurs. Before you try to write lengthy programs of your own in BASIC, try modifying or customizing any of these progams to get a good feel for how BASIC works.

PROGRAM 1: Number Guess

In this program the computer generates random numbers, and you try to guess what the number is. When you guess the correct number, the computer tells you how many attempts you took to arrive at the correct number.

```
10 PRINT "I WILL THINK OF A"
15 PRINT "NUMBER BETWEEN 1 AND 100"
20 PRINT "TRY TO GUESS WHAT IT IS"
25 N=0
30 X=INT(RND(56)*99+1)
35 PRINT
40 PRINT "WHATS YOUR GUESS  ";
50 INPUT G
52 N=N+1
55 PRINT
60 IF G=X THEN GOTO 110
70 IF G>X THEN GOTO 90
80 PRINT "T00 SMALL, TRY AGAIN ";
85 GOTO 50
90 PRINT "T00 LARGE, TRY AGAIN ";
100 GOTO 50
110 PRINT "YOU GOT IT IN ";N;" TRIES"
113 IF N>6 THEN GOTO 120
117 PRINT "VERY GOOD"
120 PRINT
130 PRINT
140 GOTO 10
150 END
```

PROGRAM 2: Heads-Tails Flipping

This program exercises the RND function of the computer by producing heads and tails. The long-term average out of many runs of this program should be approximately fifty percent heads, fifty percent tails.

```
5 REM HEADS/TAILS FLIPPING
10 Y=1
20 C=0
30 X=1
40 F=INT(RND(45)*2)
50 IF F=1 GOTO 80
```

```
60 PRINT "T";
70 GOTO 100
80 C=C+1
90 PRINT "H";
100 X=X+1
110 IF X<51 GOTO  40
120 PRINT
130 PRINT C;" HEADS OUT OF 50 FLIPS"
132 PRINT
133 PRINT
135 Y=Y+1
140 IF Y<11 GOTO 20
150 END
```

PROGRAM 3: ESP Test

This is another number-guess program where you are simply guessing heads or tails as the computer flips a coin. The computer keeps constant tabs on how many right and wrong answers you have given.

```
10 REMESP TESTER
15 REMTYPE E TO END
20 H=1
25 W=0
30 T=0
35 C=0
37 E=10
40 F=INT(RND(12)*2)
42 IF F=0 THEN A$="H"
43 IF F=1 THEN A$="T"
50 PRINT "H OR T ";
60 INPUT X$
70 PRINT
80 IF X$=A$ THEN GOTO 100
83 IF X$="E" THEN GOTO 150
85 W=W+1
87 PRINT "WRONG"
90 GOTO 120
100 C=C+1
110 PRINT "RIGHT"
120 PRINT "W=";W;" R=";C
130 PRINT
140 GOTO 40
150 PRINT "BYE"
160 END
```

PROGRAM 4: Power Generation

This program generates powers of two up to the mathematical limit of the computer. It demonstrates the fact that BASIC automatically reverts back to scientific notation (E-format) when numbers are more than about six digits long up to a maximum of 10 to the 32d power. BASIC can also handle fractions as small as 10 to the -32d power.

```
5 PRINT
7 PRINT
10 PRINT "POWERS OF TWO"
20 PRINT
30 PRINT "POWER          VALUE"
40 X=0
50 Y=1
60 PRINT X, Y
70 Y=Y*2
75 X=X+1
80 IF X=126 THEN GOTO 100
90 GOTO 60
100 END
```

PROGRAM 5: Decimal-Binary Conversion

It is important that the user become familiar with binary in hexadecimal notation if he is to master machine language or assembly language.

```
50 PRINT
60 PRINT
70 PRINT "DECIMAL TO BINARY"
80 PRINT "   CONVERTER"
90 PRINT
93 PRINT
95 PRINT
100 INPUT X
101 IF X<0 THEN GOTO 330
102 IF X>32767 THEN GOTO 330
104 PRINT
105 PRINT "X=";
110 Y=16384
120 A=INT(X/Y)
130 IF A=0 THEN GOTO 200
140 PRINT "1";
150 X=X-Y
160 GOTO 300
200 PRINT "0";
300 Y=Y/2
310 IF INT(Y)=0 THEN GOTO 320
315 GOTO 120
320 GOTO 90
330 END
```

PROGRAM 6: Prime Number Generation

Try to figure out how this program works.

```
10 PRINT"PRIME NUMBER"
11 PRINT"GENERATOR"
13 Y=2
```

```
15 A=1
17 GOTO 80
18 X=1
20 X=X+1
50 Z=INT(Y/X)
60 IF INT(Z*X)=Y GOTO 85
70 IF X*X>Y GOTO 80
75 GOTO 20
80 PRINT A,Y
82 A=A+1
85 Y=Y+1
90 GOTO 18
100 END
```

PROGRAM 7: Acey-Deucy

This is a longer program that should be fun to play. Once you get this program in and running, it would be wise to store it on audio cassette or disk for future use.

```
10 PRINT "ACEY-DUCEY"
12 PRINT "YOU WILL GET 25 HANDS"
13 H=1
15 PRINT
17 T=100
19 PRINT "YOU HAVE $";T
20 X=INT(7*RND(67)+6)
21 IF X>12 THEN GOTO 20
30 Y=INT(X*RND(23)+1)
31 IF Y>=X THEN GOTO 30
32 IF Y=1 THEN Y=2
40 A=X
50 GOSUB 500
60 A=Y
70 GOSUB 500
80 PRINT
100 PRINT "YOUR BET";
110 INPUT B
111 IF B<=T THEN GOTO 120
112 PRINT "YOU DONT HAVE THAT MUCH"
113 GOTO 100
120 Z=INT(13*RND(99)+2)
121 IF Z>14 THEN GOTO 120
130 A=Z
140 GOSUB500
150 PRINT
160 IF Z<=Y GOTO 200
170 IF Z>=X GOTO 200
180 PRINT "YOU WIN"
181 PRINT
182 PRINT
190 T=B+T
195 GOTO 300
200 PRINT "YOU LOSE"
201 PRINT
202 PRINT
```

```
210 T=T-B
220 IF T<=0 GOTO 380
300 H=H+1
310 IF H>25 GOTO 400
320 GOTO 19
380 PRINT "YOUR OUT!"
390 STOP
400 PRINT "THATS 25 HANDS"
410 STOP
500 IF A<11 THEN GOTO 505
501 IF A>14 THEN PRINT"ERROR":STOP
502 ON A-10 GOTO 522,524,526,528
505 PRINT A;
510 RETURN
522 PRINT "JACK ";
523 RETURN
524 PRINT "QUEEN ";
525 RETURN
526 PRINT "KING ";
527 RETURN
528 PRINT "ACE ";
529 RETURN
```

PROGRAM 8: Multiplication Quiz

This demonstrates the use of the computer as a teaching aid.

```
10 PRINT "MULTIPLICATION QUIZ"
13 N=0
15 C=0
16 I=0
20 X=INT(RND(56)*13)
30 Y=INT(RND(54)*13)
40 Z=X*Y
50 PRINT
60 PRINT X;"*";Y;"=";
70 INPUT W
75 PRINT
80 IF W=Z GOTO 120
90 PRINT "STUPID!"
91 PRINT "THE ANSWER IS";Z
100 I=I+1
110 GOTO 140
120 PRINT "YOU ARE RIGHT!"
130 C=C+1
140 PRINT C;" ARE RIGHT"
150 PRINT I;" ARE WRONG"
160 N=N+1
170 IF N<=9 GOTO20
180 IF C>=6 GOTO 190
183 PRINT"YOU FLUNKED!"
184 PRINT "PRACTICE!"
185 GOTO 13
190 IF C>=9 GOTO200
195 PRINT "YOU DID OK"
198 GOTO 210
```

```
200 PRINT"NICE JOB!"
210 PRINT "TRY AGAIN?"
220 INPUT T$
230 IF T$="Y" GOTO 13
240 END
```

PROGRAM 9: Fahrenheit-Celsius and Celsius-Fahrenheit Conversions

```
10 PRINT "THIS PROGRAM C0NVERTS"
20 PRINT "FAHRENHEIT TO CENTIGRADE"
30 PRINT "AND VICE-VERSA"
40 PRINT
41 PRINT "TYPE THE TEMPERATURE TO CONVERT"
42 PRINT "A COMMA AND A"
43 PRINT "  0 TO GET FAHRENHEIT
44 PRINT "  1 TO GET CENTIGRADE
50 C=0
60 F=1
70 INPUT X,Y
75 IF Y>1 GOTO 250
80 IF Y=1 GOTO 200
90 A=(9*X)/5+32
100 PRINT "   =";A;"F"
110 PRINT
120 GOTO 70
200 A=(5*(X-32))/9
210 PRINT "   =";A;"C"
220 PRINT
230 GOTO 70
250 END
```

PROGRAM 10: Subroutines

A simple demonstration of the use of subroutines.

```
10 REMOSI ADVERTISING PROGRAMS
20 GOSUB 500
22 GOSUB 500
25 GOTO 100
30 PRINT"     XXX   XXX   X"
35 RETURN
50 PRINT"    X   X X      X"
55 RETURN
70 PRINT"    X   X     X  X"
75 RETURN
80 PRINT"    X   X  XXX   X"
90 RETURN
100 GOSUB 30
110 GOSUB 50
115 GOSUB 50
120 GOSUB 50
```

```
130 GOSUB 80
140 GOSUB 70
150 GOSUB 70
160 GOSUB 70
170 GOSUB 30
180 GOSUB 500
190 GOSUB 690
200 GOTO 20
500 X=1
510 PRINT
520 X=X+1
530 IF X>8 GOTO 550
540 GOTO 510
550 RETURN
690 Z=1
700 Z=Z+1
710 IF Z>30 GOTO 720
715 GOTO 700
720 RETURN
```

Lower Case

Ohio Scientific's computer systems are capable of generating lower case characters as well as numerous graphics characters. Under normal operation, the shift lock key is in the depressed or recessed condition. It must be in this state for normal systems level software to operate. With the shift lock key down, depressing any alphabetic, numeric or punctuation key on the keyboard will cause the keyboard to generate upper case alphabetics and numerics. By depressing the left or right shift key in conjunction with another key, punctuation and special control codes will be generated. For example, depressing the shift key and the 5 key together generates a per cent (%) sign. Depressing the shift key and the P key together generates a commercial at (@) which is recognized by BASIC as being the line delete code.

Shift Lock Key Up

The shift lock key can be released for certain special applications. Specifically, to generate lower case characters as part of literal strings in BASIC and for use in conjunction with word processing software. With the shift lock key in the up position, the keyboard will act considerably different than with it in a locked position. With the shift lock key up. only standard alphabetic characters will generate expected results. Specifically, depressing any alphabetic key will cause the generation of a lower case alphabetic character. In this mode of operation, the left shift key has a different function than the right shift key. Depressing the left shift key in conjunction with alphabetic or numeric keys generates upper case alphabetics and numerics. The right shift key in conjunction with other keys generates upper case punctuation. For example, depressing the 5 key without either shift key generates "garbage". Depressing the 5 key in conjunction with the left shift key generates numeral 5. Depressing the 5 key in conjunction with the right shift key generates the per cent (%) sign. As stated in numerous other places, the shift lock key should be kept in a depressed or locked mode except when lower case characters are explicitly desired.

Advanced Features

Auto Repeat

The Challenger keyboard has a built-in auto repeat feature. By depressing any key and holding it down, first that character will be generated once and then after approximately one-half second, the character will be repeated at a rapid rate.

Programming and Graphics

The Challenger keyboard is directly read out via microcomputer and can be programmed for special functions. The computer system is also capable of a wide range of graphics and gaming displays in

addition to standard upper and lower case characters. Refer to the Challenger Character Graphics Reference Manual for a complete discussion of graphics, and programmable keyboard operation of the computer.

Onwards Towards New Horizons

This is the end of the formal portion of the manual. The remaining sections are specific appendices and deal with the hardware and advanced software topics of the computer system. The following table can act as a guide line towards expanding your computer horizons in areas of more complex software and hardware investigations.

A. More on BASIC

Your Ohio Scientific microcomputer system is capable of storing files on cassette. It is also capable of having program key functions and elaborate graphics. Discussion of the cassette data file capabilities is in the BASIC reference manual. The Ohio Scientific character graphics manual covers the programming of the keyboard and graphics capabilities in detail.

B. Machine Code

This manual includes documentation on the 65V monitor PROM which is built into your Ohio Scientific computer. This allows you to examine memory, load and run machine code programs. Ohio Scientific offers an excellent book HOW TO PROGRAM MICROCOMPUTERS which covers the machine language programming techniques and procedures for the 6502 which is used in this computer and the 6800 and Z80 which is utilized in the Challenger III triple processor system. Ohio Scientific also offers an extended monitor for debugging machine programs, an interactive assembler/editor which will operate on your computer provided it has 8K or RAM or more.

C. Hardware Expansion

The small computer hardware appendix of this manual and the Challenger 1P Technical Report provide an in-depth discussion of the hardware configuration of your computer and its expansion capabilities. By adding more memory and a mini-floppy system, you will be able to broadly expand your computer's capability. By adding a mini-floppy, you will gain the ability to instantly load and save programs and to have random access data files. Your system can be further expanded to be plug-compatible with the OSI 48 line BUS giving you a wide range of real world interface capability.

D. Other Advanced Topics

Your purchase of an Ohio Scientific small computer entitles you to a one year subscription to the Ohio Scientific Small Systems Journal. The journal covers a broad range of topics for personal and small business users of Ohio Scientific computers. It typically contains software ideas and new products which will be of interest to you as an owner of an Ohio Scientific small computer system.

INTRODUCTION TO SMALL COMPUTER HARDWARE

Small computers are made up of several modules, or blocks. The first of these, the microprocessor, is an integrated circuit much like those used in modern watches and calculators. It performs the function of a large computer, which a few years ago would have been prohibitively expensive. This integrated circuit makes the whole field of personal computing possible and affordable.

Next, one must have some memory, which can be in the form of ROM, PROM, EPROM, or RAM. The first three devices provide permanent storage of programs and data, that is, they do not "forget" when the power is turned off. RAM provides modifiable storage, that is, programs and data can be written in and read out repeatedly. However, almost all types of RAM "forget" whenever the power is turned off. Therefore, RAM is used for temporary storage, and ROM, PROM, and EPROM are used for permanent storage of programs which will not change. Generally, a small computer will have a large amount of RAM for general purposes, and very little PROM or ROM. It does need some of the latter, to give it some intelligence when it is first turned on, and this is usually in the form of a monitor program which allows the user to load additional programs from some external device such as a tape recorder into RAM. Today, the most advanced computers put BASIC, the most commonly used programming language in ROM. This has only recently become possible because BASIC requires approximately 8,000 bytes of ROM, which had been a costly feature until now.

Along with memory, the microprocessor requires some form of I/O device (Input/Output), that is, some way of "talking" to the outside world. The computer communicates through interfaces such as the ACIA-Based Serial Interface and the PIA-Based Parallel Interface to external devices called peripherals, such as CRT terminals, Teletypes, paper tape readers, paper tape punches, line printers, and audio cassettes. Other types of interfaces include D/A converters and A/D converters.

The microprocessor communicates with its interfaces and memory with a series of wires, or lines, called buses. There are generally three buses in any microcomputer: an address bus, a data bus, and a control bus. These three buses are combined in what is called a system bus.

The address bus is generally made up of 16 lines. The microprocessor always is the signal generator for this bus. The 16 lines carry specific addresses, that is, 16-bit binary words which select a memory or I/O location. This location can be thought of as a post office box, and the address word can be thought of as the box number. The microprocessor can, therefore, through its address bus, specify memory or I/O locations.

It can place data in these locations, or read memory from them via an 8-bit wide data bus. Its 8-bit width indicates that the microcomputer can read or write one byte at a time. With ROMs, PROMs, and EPROMs, the microcomputer only reads what is already in those locations, and acts accordingly. In the case of RAM and some I/O locations, the microprocessor can also place data in these locations. Unless the computer has a large amount of ROM, it will generally be very "stupid" when first turned on. That is, its main memory, or RAM, has nothing of value in it. The user must enter a program which the microcomputer can then execute in its RAM memory. It does this by use of a PROM Monitor Program, that is, a short program which the computer runs, allowing it to take data from some interface, and ultimately from some peripheral, and place it into its operating memory, or RAM. It can then later perform functions and write or store additional programs based on this stored program. The typical peripherals used for this are a video display interface, and a keyboard, or a Teletype or CRT terminal. Additional mass storage devices, such as paper tape readers, audio cassettes, and floppy disks, are utilized for storage of programs.

The OSI system utilizes a 48-line system bus made up of an address bus of from 16 to 20 lines (depending on the CPU board used in the system), an 8-line data bus, a 7-line control bus, power connections, and spare lines for user connections. The system utilizes 8" x 10" PC boards plugged into an 8-slot backplane, which spaces the system boards one inch apart. For very small computers, the Model 500 can be used without a backplane board as a stand-alone computer, that is, it can be populated with the microprocessor, PROM and ROM memory, RAM memory, and a serial interface, so that it can function as a complete computer by itself. For larger systems, however, a backplane board and additional system boards are used.

It is necessary for anyone servicing or building an OSI system to be somewhat familiar with the 48-line bus utilized by the computer. This bus is outlined on page

Glossary of Small Computer Terms

ACIA- (Asynchronous Communications Interface Adapter) An IC used for serial data transfer between a device such as a small computer and a serial terminal.

A/D- (Analog/Digital) refers to changing an analog signal to a digital signal which the computer can use.

Backplane Board- (Sometimes called mother board) allows simple interconnection between small computer boards using the same bus.

Bit- The smallest amount of data possible; a bit is expressed as a high or low (on or off) state (normally 1 or 0).

Bus- Refers to the set of foils or wires needed to interconnect between system boards provided that the pattern of how each of the connections is used is consistent for all system boards.

Byte- 8 bits of data. The most fundamental microprocessor commands are organized into sets of 8 bits (i.e. bytes).

CPU- (Central Processing Unit) the portion of a microprocessor which does the actual arithmetic calculations and decision making.

D/A- (Digital/Analog) Refers to changing digital signals (from the computer) into analog signals.

EPROM- (Erasable Programmable Read Only Memory) information stored in an EPROM IC can only be removed by special light sources or specific voltages (depending on the type of EPROM). Through the use of a special programming device, the user can store a set of information in the EPROM after it has been erased.

Hardware- that part of a computer consisting of actual electronic circuitry, printed circuit boards, case, and power supply as opposed to software which is the set of commands the hardware is executing.

I/O- (Input/Output) refers to bringing information into the machine in a form it recognizes and allowing the machine to transmit information. In other words, communicating with the outside world.

Memory- a general term referring to parts of the computer where information is stored.

Microprocessor- a large IC (electronic part) which functions as the CPU of the microcomputer. The 6502 on Ohio Scientific's 500 board is a microprocessor.

PC Board- (Printed Circuit Board) a card with foils (electronically conductive pathways) connecting electronic components which are mounted on the board.

PIA- (Peripheral Interface Adapter) IC used for parallel data transfer.

PROM- (Programmable Read Only Memory) Memory which can have information stored on it once, but, is not normally changeable.

RAM- (Random Access Memory) the data stored in this type of memory is easily changed by the user while the machine is in use (unlike ROM, PROM, EPROM) However, it is erased whenever electrical power is turned off.

ROM- (Read Only Memory) preprogrammed, unchangeable memory.

Software- programs or instructions that the machine will execute,

48 LINE SYSTEM BUS OUTLINE

B1 - low true WAIT When pulled low by a system board, causes processor clock to slow down to speed of approximately 500KHz on most processor boards. This is used to service slow memory and I/O devices.

B2 - NMI (non-maskable interrupt) When brought low, a non-blockable interrupt occurs, causing the processor to stop its operation and service this interrupt, that is, go to a specific memory location and start executing an interrupt service routine.

B3 - IRQ (interrupt request) An interrupt which can be masked by the processor, that is, the processor can choose to ignore this interrupt under program control. If the interrupt is not masked, it will cause the processor to stop executing the program it is in, and jump to a different location.

B4 - DD (data direction) When pulled low by system board, it changes the direction of the 8T26 buffers on the CPU board, and thus switches the processor from outputing data to the bus to listening to the bus.

B5 - D0
B6 - D1
B7 - D2
B8 - D3
B9 - D4
B10 - D5
B11 - D6
B12 - D7

Bi-directional eight-bit wide data bus for communication of data between the processor and system boards.

B13
B14
B15
B16

Upper data bits on some systems

B17 Optional reset line used to clear all PIAs and similar I/O circuitry in the system.

B18 spare line

B19
B20
B21
B22

Memory management address lines: Lines 21 and 22 are used on systems with a 500 CPU Board; all 4 are used with the 510.

B23 +12 Power connection

B24 -9 Power connection

B25
B26

+5 Power connection

B27
B28

Ground Connection

B29 - A6
B30 - A7
B31 - A5
B32 - A8
B33 - A9
B34 - A1
B35 - A2
B36 - A3
B37 - A4
B38 - A0

Ten low-order address lines

B39 - Ø2 Used to clock external circuits or external I/O interfaces, such as the A/D converter.

B40 - R/W (read/write) Originates at the microprocessor and specifies read or write operations on the data bus.

B41 - VMA (Valid memory address) Only used in conjunction with the 6800 microcomputers. The 6502's always have this line high.
B42 - Ø2-VMA Master timing signal for enabling memory and I/O in the system.
B43 - A10
B44 - A11
B45 - A12
B46 - A13
B47 - A14
B48 - A15

Six high-order address lines

Your Challenger 1P or Superboard II unit utilizes its own unique internal BUS structure. This BUS is fed out to a 40 pin IC socket and optionally fed to the 610 expander board. The 610 expander board has full buffering and signal condition circuitry to adapt it to the OSI 48 line BUS. The physical adaptation is via a model 620 ribbon cable and BUs adapter board which will plug into any OSI backplane. So although the Superboard II and Challenger 1P systems do not directly support the OSI 48 line BUS, they can be expanded to handle the OSI 48 line BUS system. Refer to the Challenger 1P Technical Report for detailed discussion of the 600 board's expansion capabilities by the 610 and 620 and discussion of suitable OSI 48 line BUS accessories for the system.

65V PROM MONITOR

INSTRUCTIONS

Anyone wishing to become proficient in programming at the machine language level should be well acquainted with the PROM Monitor which is necessary to examine and change the contents of memory locations. The Monitor used in conjunction with the 6502 video system is the OSI 65V PROM Monitor.

As explained above, OSI's 8K BASIC comes up when the computer is reset. When this is done, the letters "C/W/M?" or "D/M?" appear on the video screen or terminal. If you wish to enter a program by means of the Monitor instead of BASIC, type an "M" on the keyboard. This brings up the Monitor and starts it in the Address Mode, that is, the mode in which you can specify memory addresses or locations to simply examine their contents. Appearing on the screen are four digits in hexadecimal notation followed by two spaces and, finally, another two digit number, also in hex. The four digit number is a location, and the two digits are the contents of that location. To examine another address, type the address on the keyboard. The same address will appear on the terminal, as will the corresponding contents of that address.

If you wish to change those contents and thus enter programs using the Monitor, you must exit the Address Mode and get into the Data Mode. To do this, type a slash (/). Then type any two hex characters, and they will be inserted into that location as its new contents. The normal procedure for entering programs via the Monitor is to use consecutive memory locations. While still in the Data Mode, you can open the next address by typing the return key. You can do this continually, each time altering the memory contents according to the needs of the program you are writing. If you want to jump to a non-consecutive location, you need to get back into the Addressing Mode by typing a period (.). Then type the new address you want. Type another slash to get back into the Data Mode and continue as before until you want to open a non-consecutive address. Extreme caution should be used whenever the Monitor is in the Data Mode as you are directly manipulating the computer's memory.

If you wish to enter a program from an audio cassette instead of manually from the keyboard, first get into the Address Mode, then turn on the cassette. Let the tape advance to the point where the program of interest begins, and type L. This transfers control to the audio cassette, such that all ASCII commands are supplied by the cassette instead of by the keyboard. The L command also puts the Monitor into the Data Mode. If the contents of ØØFB are ØØ, the Monitor will accept commands from the keyboard.

If the cassette does not load ØØFB (hex) with ØØ, to transfer control back to the keyboard, press reset. Otherwise, commands are accepted from the Audio Cassette UART.

To run any program which you have entered via the Monitor, get into the Address Mode, i.e., type a period, type the starting address of the program, and type a "G."

Label	Program Entry Points
VM	FE00 - Restart Location FE0C - Bypasses UART and Stack Pointer initialization and the clearing of decimal mode but does clear the screen.
IN	FE43 - Entry into address mode, bypass initialization
INNER	FE77 - Entry into data mode, bypass initialization
Label	**Subroutines**
OTHER	FE80 - Input an ASCII character from Audio Cassette UART
LEGAL	FE93 - Returns stripped ASCII number if 0-9 or A-F. Otherwise returns a FF.
INPUT	FEED - Input an ASCII character from keyboard

Required Hardware

The 65V Monitor requires as a minimum the following hardware: an OSI Model 400 board with a 6502 microprocessor, 1,024 words of RAM memory located from ØØØØ to Ø3FF, and the 65V monitor itself. It also requires an OSI Model 440 Board populated for alphabetic display and keyboard input. The 440 Video Board must be located at DXXX which will automatically locate the keyboard input at DFXX.

The keyboard must be a seven-bit high true ASCII keyboard with a positive or negative going strobe pulse at least 100 microseconds long.

The 65V Monitor will additionally support input from a generalized serial communications subsystem of an OSI 430 board located at FBXX. Specifically, the monitor contains a load program for a 430 board-based audio cassette interface. The same program can be used with a 430 board configured for digital cassette or ASCII teletype input.

Commands

Address Mode Commands:
/ - Change to Data Mode
G - Go -- Jump to location seen on screen and execute program found there.
L - Transfer control to audio cassette.

Data Mode Commands:
. - Change to Address Mode
RETURN - Open next address. In other words, increment location pointer by 1.

If the 65V is in address mode, typing 0 - 9 or A - F will cause that number to be rotated into the LSD of the location pointer. Typing a 4 causes 0123 XX to become 1234 XX.

If it is in Data Mode, the number is rotated into the data contents and memory is thus modified. This permits the easy correction of errors. If, for example, the user typed 0478 when intending to look at location 047B, he would simply type 047B.

All of the non-command keys and non-hexadecimal characters are ignored by the monitor.

65V Demonstration Program

The following is a program which may be entered using the 65V Monitor from the keyboard or audio cassette. An "*" indicates a return key depression.

.0002 Loads the ASCII Message Starting at Location 0002
/4F * 53 * 49 * 20 * 36 * 35 * 56 * 2E * 5F

.0200 Loads the Main Program at 0200
/ A9 * 02 * A2 * 00 * 20 * 00 * 03 * A2 * 00 * 20 * ED * FE * 9D * 24 * D2 * E8 * 4C * 09 * 02

.0300 Loads the Subroutine at 0300 to Output an ASCII Character String.
/85 * 00 * A9 * 00 * 85 * 01 * A0 * 00 * B1 * 00 * C9 * 5F * F0 * 0A * 9D * E4 * D1 * E8 * E6 * 00 * D0 * F2 * E6 * 01 * 60

.0200G Loads the Starting Address of the Program and Execute it.

You should see the message "OSI 65V." on the screen. Now, you may type any keys and they will be echoed just below the message. Press reset to re-enter the 65V Monitor.

If this were entered off of the audio cassette, it would be self-loading and auto starting. Since the cassette is in complete control, it can load the starting address and execute the program without user interruption.

WARRANTY AND TROUBLESHOOTING HINTS

Ohio Scientific fully-assembled products are covered by a limited warranty. Challengers are covered for a period of sixty days against defects in materials and workmanship to the extent that any malfunction not caused by abuse, misuse, or mishandling will be repaired or corrected without charge to the owner provided that the unit is returned postpaid to Ohio Scientific within sixty days of day of receipt by the user.

Beyond this sixty-day period, up to one year from day of receipt by the user, the system is further warranted against defects in materials to the extent that Ohio Scientific will repair or replace them, charging only for labor on the portion of the electronic component that is manufactured by Ohio Scientific, without charge for the part(s). This warranty includes power supplies and floppy disk drives. It specifically excludes hard disks, terminals, video monitors, audio cassettes and some keyboards. Ohio Scientific's only obligation under these terms, in either case, is to repair the unit and return it once it has been delivered postpaid to Ohio Scientific. Typical turn-around time under this warranty is two to three weeks plus shipping time from the factory. Ohio Scientific cannot be held responsible for delays beyond its control such as those caused by shipping or long delivery of replacement components, e.g., floppy disk drives, etc.

Ohio Scientific reserves the ultimate authority to determine what constitutes in-warranty repair in circumstances where circuit modification, abuse, misuse, or shipping damage occur. The warranty is also subject to the use of proper packing material in any returns. This is the only warranty expressed or implied by Ohio Scientific and the only warranty which any Ohio Scientific agent is authorized by Ohio Scientific to give in conjunction with the product. Any maintenance or extended warranties that the end user may entertain with an Ohio Scientific representative or dealer are solely between that representative and the customer and are in no way authorized or supported beyond the extent of the above stated warranty by Ohio Scientific. The support of such warranty or maintenance contract is the sole responsibility of the agent offering the warranty.

Ohio Scientific software offers absolutely no warranty. The software is always thoroughly tested and thought to be reasonably bug-free when released. Ohio Scientific maintains a full staff of software experts and will endeavor to correct any serious bugs that may be discovered in the software after release in a reasonable amount of time. However, this is a statement of intent and not a guarantee in such matters.

TROUBLESHOOTING

If you encounter any difficulty in procedures in this manual, first refer to the following troubleshooting guides. If they do not provide sufficient help for you to solve your problems, proceed to the end of this section.

1) Order does not seem complete. First check to see that all packages specified have arrived. Carefully look over the packing lists, manuals, and this manual to determine what is supposed to be present in your system. If you have further doubts, check with the dealer or representative from whom you purchased your system, or the factory, <u>if and only if you ordered directly from the factory</u>.

2) Unit(s) mechanically damaged in shipment. Report damages or losses immediately to carrier. All units are shipped by Ohio Scientific fully insured. Under no circumstances should you ship the unit back in such condition as it would then be impossible to determine where the unit was damaged. This can cause a long drawn-out dispute with the carrier especially if the unit was transported by different carriers.

3) User has difficulty in following manual because of high level of technology involved. Suggestions: Obtain assistance from your local Ohio Scientific dealer or representative. If you ordered factory direct, or, are at a considerable distance from the dealer, contact your local hobby club and see if any members can assist you. Hobby club members are generally very willing to help out, which is a major reason they are in the club. Current club activities are listed in BYTE, Kilobaud, Interface-Age, and bi-weekly publications such as ON_LINE. Any local computer store should be able to assist you in becoming a computer club member.

4) Reset light does not illuminate on power-up. Carefully check power connections. Check to see if unit is plugged in, that the power switch is on and that power is present at the power outlet. If so, turn the unit off and unplug it. Check the 2 amp fuse at the back of the unit and check the reset light itself by pulling the lens cap out and making sure that the lamp is properly seated in its socket.

5) Reset switch is dimly lit or not lit at all after you have checked with the above procedures. Carefully inspect the PC board portion of the computer for foreign matter such as a wire cutting or something leading out from the PC board. Also check to see that all PC boards are properly seated, and that any ribbon cables are properly seated in their sockets. If the unit light is only dimly lit, remove about half of the PC boards. If the light comes up to full brightness with these out, put those boards back in and pull the other ones out. If the same condition occurs, it means that there is a power supply malfunction and that the unit will have to be returned for repair. If the power supply folds back when some PC boards are out, and not with others, you should be able to isolate the board causing the foldback. That board most likely has foreign matter across it, causing the short on the board.

6) Power supplies look fine, but computer doesn't seem to reset at all or properly. Symptoms: nothing comes out on serial terminal or screen doesn't clear on video system. Solution: again, give the system a careful visual inspection. At this point, it would be invaluable to have access to another Ohio Scientific computer system by way of a dealer or another computerist. If neither is available, and you do not wish to or are not able to attack the actual circuitry of the system, it will most likely be necessary to return the unit for repair.

7) Computer appears to reset properly, but there is no response to the keyboard. On serial systems, carefully check your interface. Make sure the terminal is set up for full duplex and no parity. On video systems, carefully check your wiring. Make sure the keyboard strobe is properly connected. It is possible that the keyboard strobe pulse is too short for the computer's polling routine to see. The keyboard's strobe should be at least 5 ms long for proper operation.

8) Last key strobe on video systems is not acknowledged until next key is depressed. Solution: Keyboard strobe is set wrong. Change polarity via the jumper on the 440 Board.

9) Keyboard operation erratic; some keys are missed. Solution: Keyboard strobe is too short. Strobe pulse must be lengthened, usually be adding capacitance to the strobe circuit on the keyboard. Strobe pulse should be on the order of 5 to 10 ms for proper operation.

10) System works fine in machine code, but in BASIC you consistently receive SN error message (Syntax error). Carefully refer to the example given in the BASIC User's Manual.

11) Floppy disk does not appear to work properly. Consult OS-65D Version 2.0 Manual for any problems concerning the Floppy Disk Drive.

12) Audio Cassette Interface does not appear to work. Carefully look at your connections between audio cassette recorder and computer making sure that you are connecting the microphone jack to the cassette interface output and the speaker jack to the cassette interface input. Be sure that you are using shielded cables and that there are no shorts or opens in your cables. Volume controls should be mid-range, tone controls should be mid-range. Be sure you are using high-quality tape and that you are running the cassette on fresh batteries or 110V AC.

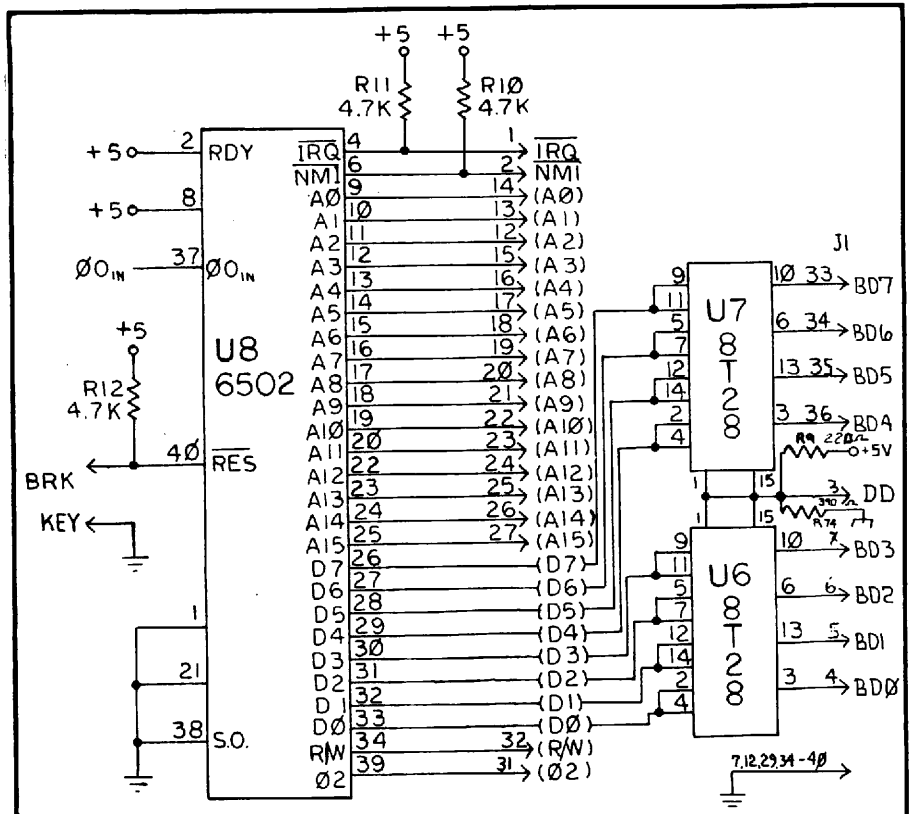
13) If you experience errors when playing back cassettes, try adjusting your own control upward or downward from midrange when recording and playing back cassettes. You may also need to adjust your tone controls. The audio cassette is generally extremely reliable with most medium-quality tape recorders. If your system is subject to a noticeable or an appreciable error rate, it is quite likely that your tape recorder is not suitable for use as a computer storage device. We highly recommend the Panasonic RQ309 tape recorder for such applications.

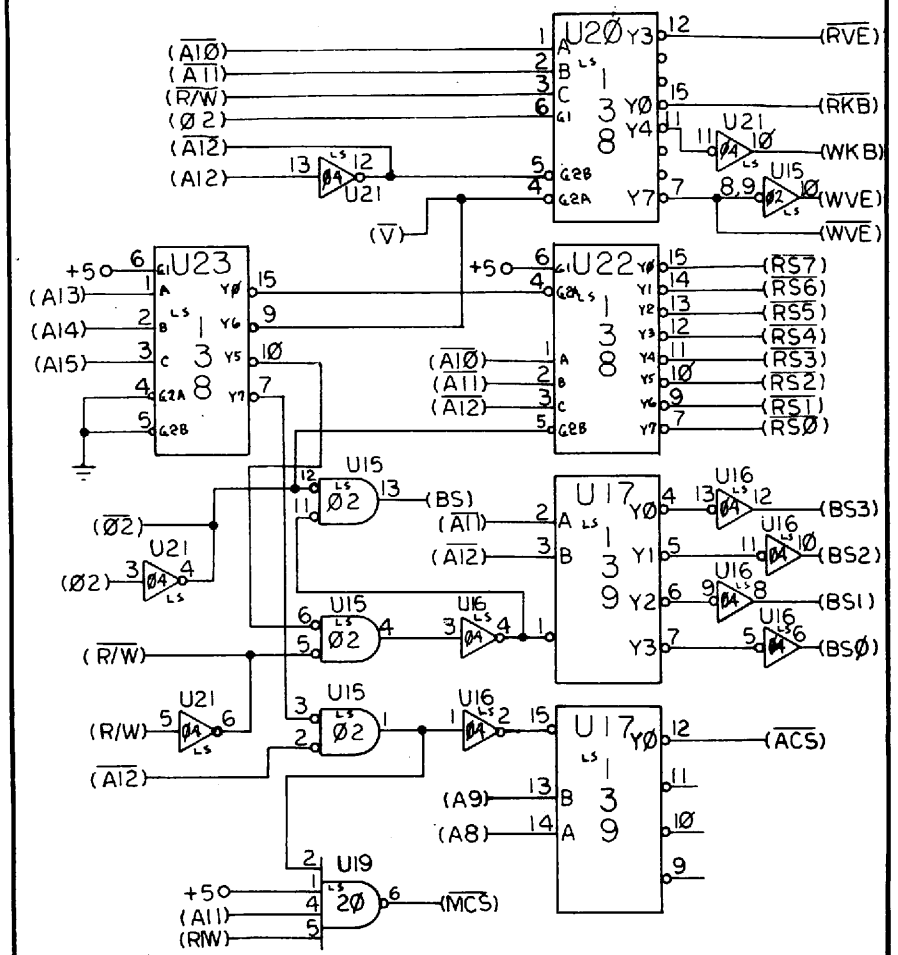
In Case of Difficulty

If you encounter a problem with your system, first carefully look over the trouble-shooting hints in your procedures. The great majority of problems encountered on new computers result simply from the user's unfamiliarity with the computer system. If you decide that you cannot resolve the problem yourself, contact the representative or dealer from whom you purchased the computer. If you purchased it directly from the factory, contact the factory at 216-562-3101. This number is for customer service. Your local OSI dealer representative should be able to help you by providing guidance on operating procedures, and in the case of an actual computer malfunction, should be able to substitute PC boards and subassemblies to isolate the problem. He should then also provide the service of getting the replacement or repair for the malfunctioning unit. If you ordered the computer factory-direct, then you must call Ohio Scientific to obtain a return authorization number before returning any unit for in-warranty repair. When you call, please immediately state that you currently own Ohio Scientific equipment and are experiencing difficulty with it. By stating this, you will quickly be routed to the service department. Please have your situation well organized so that you can present it to the service department personnel. They may then be able to help you immediately, or may need to consult with engineers and production staff to see if there is a simple solution to your problem, which will require another telephone call. If it is then decided that there is a hardware problem which must be repaired, you will then be offered a return authorization no., so that you can send the faulty equipment back for repair.

All equipment returned for repair must have a return authorization no., and must be packaged in the same containers or equivalent containers upon return. That is, all Challenger products are to be shipped in double containers with at least one inch of standard packing material lining all six sides of the inner container to preclude the possibility of puncture and damage. All packages must be stuffed with packing material so that the computer cannot vibrate or move around in the box; that is, it should pass a manual shake test, where there is no shift of load when you hold the box and shake it. Computers not shipped in double-wall containers or double boxes will not be accepted for

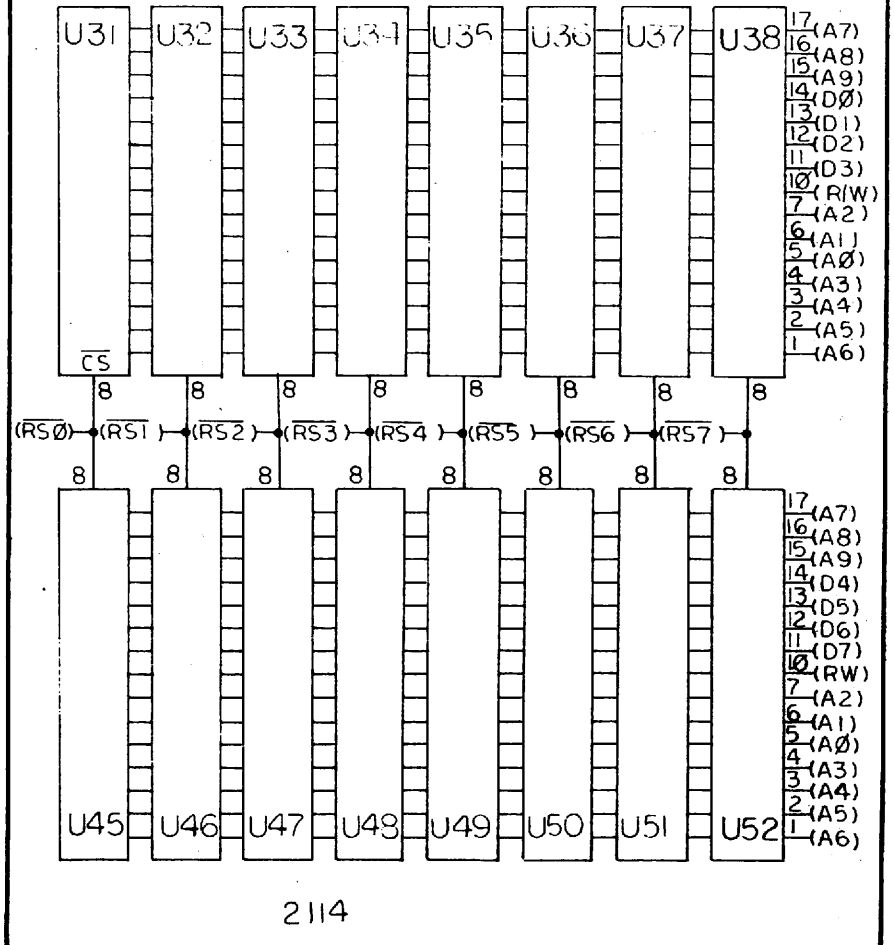
in-warranty repair by Ohio Scientific, since it is our experience that such units are often damaged in transit. The preferred method of shipping is by UPS insured. Please enclosed a note clearly stating the problems you have encountered with the unit when you return it for repair, and be sure to include the return shipping address on that note.





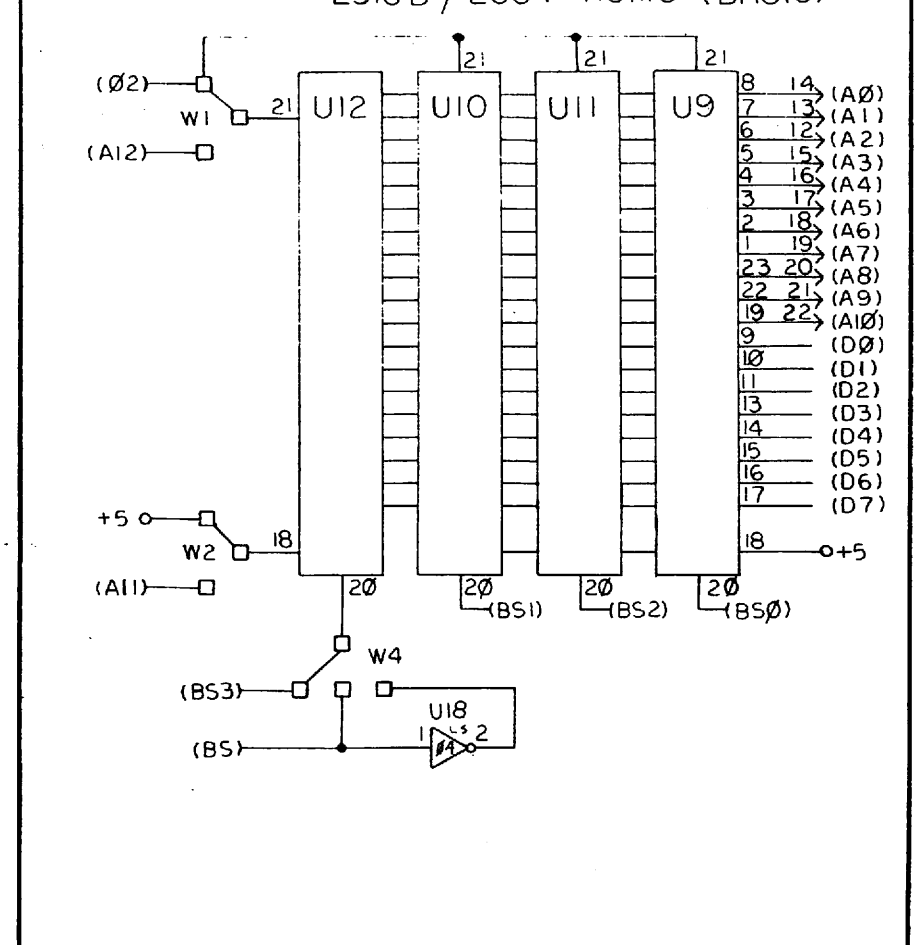
OHIO SCIENTIFIC			product name/number SUPER BOARD 2 600	
date 5 SEP 1978	revision B	page	status	sheet 1 of 13

OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 2 of 13

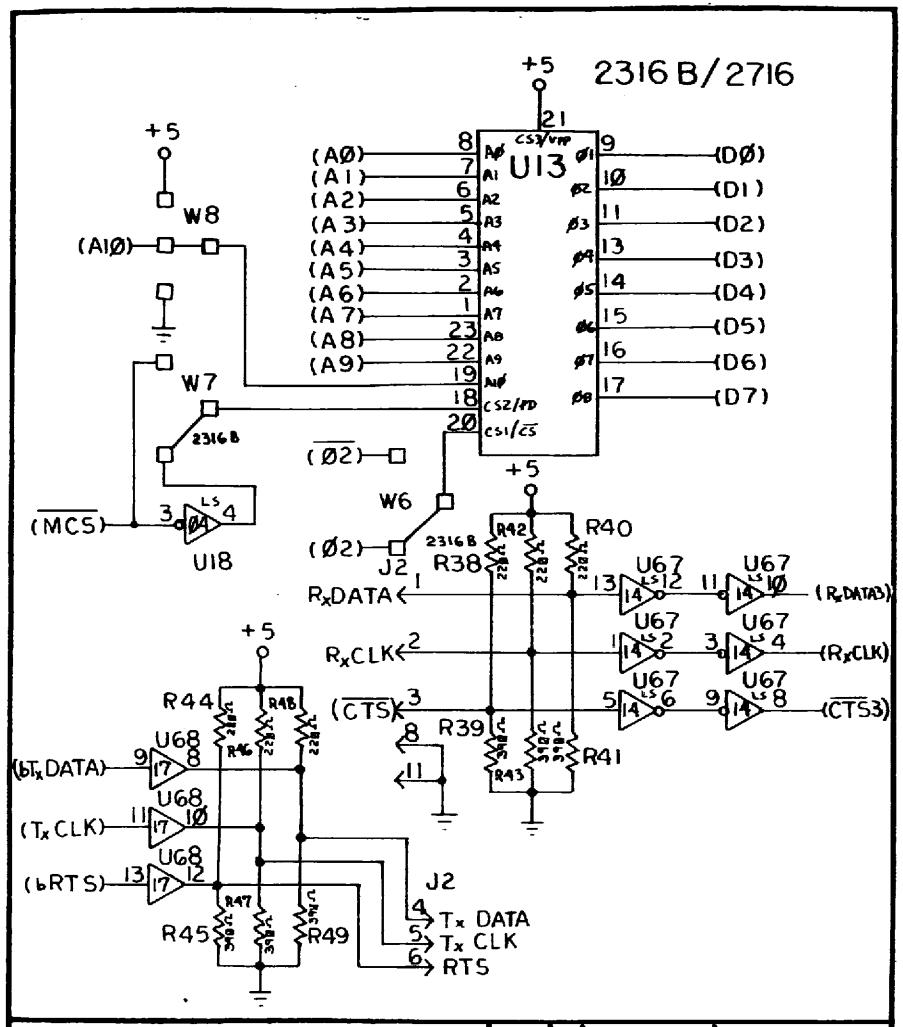


2114

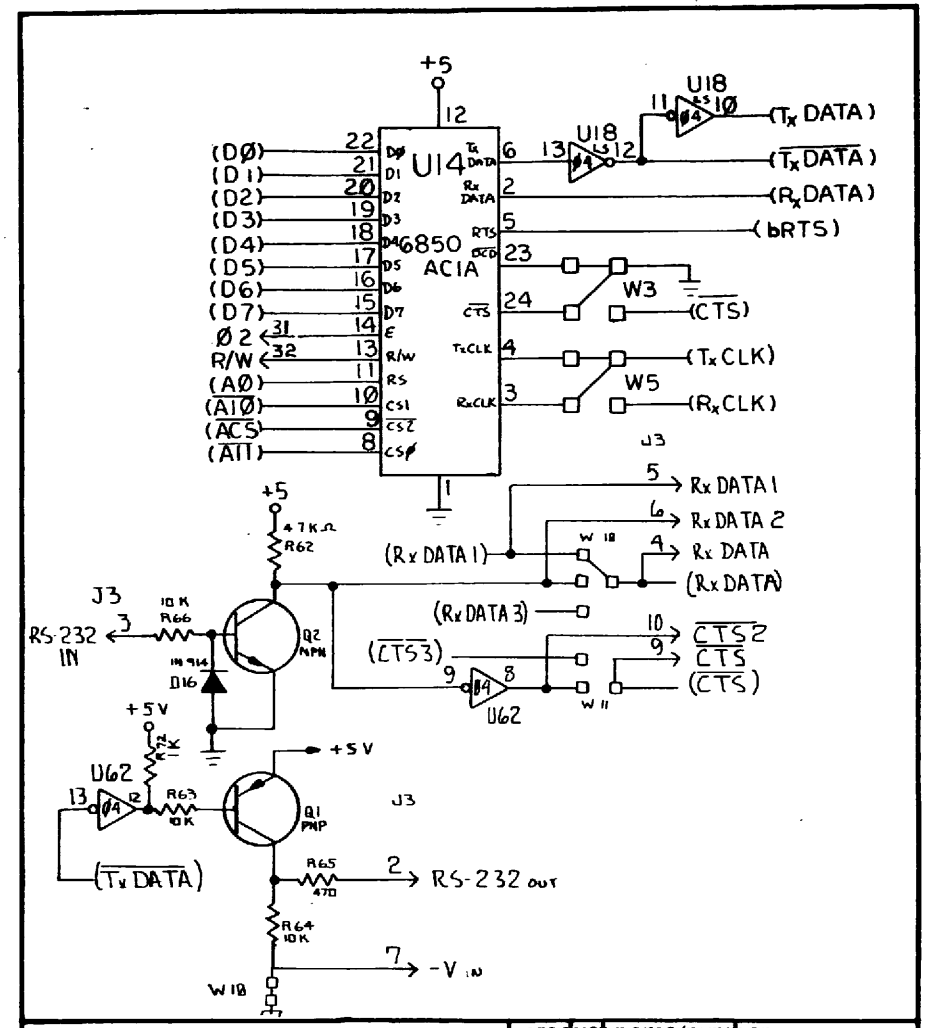
OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 3 of 13



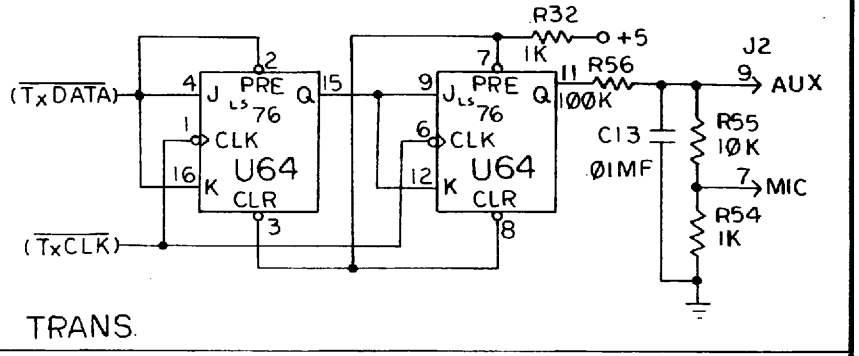
OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 4 of 13

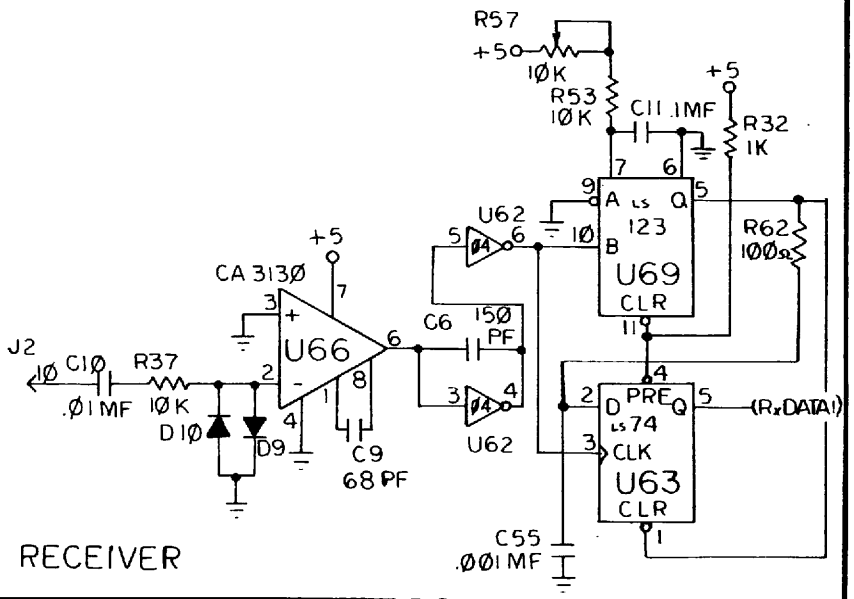


OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 5 of 13

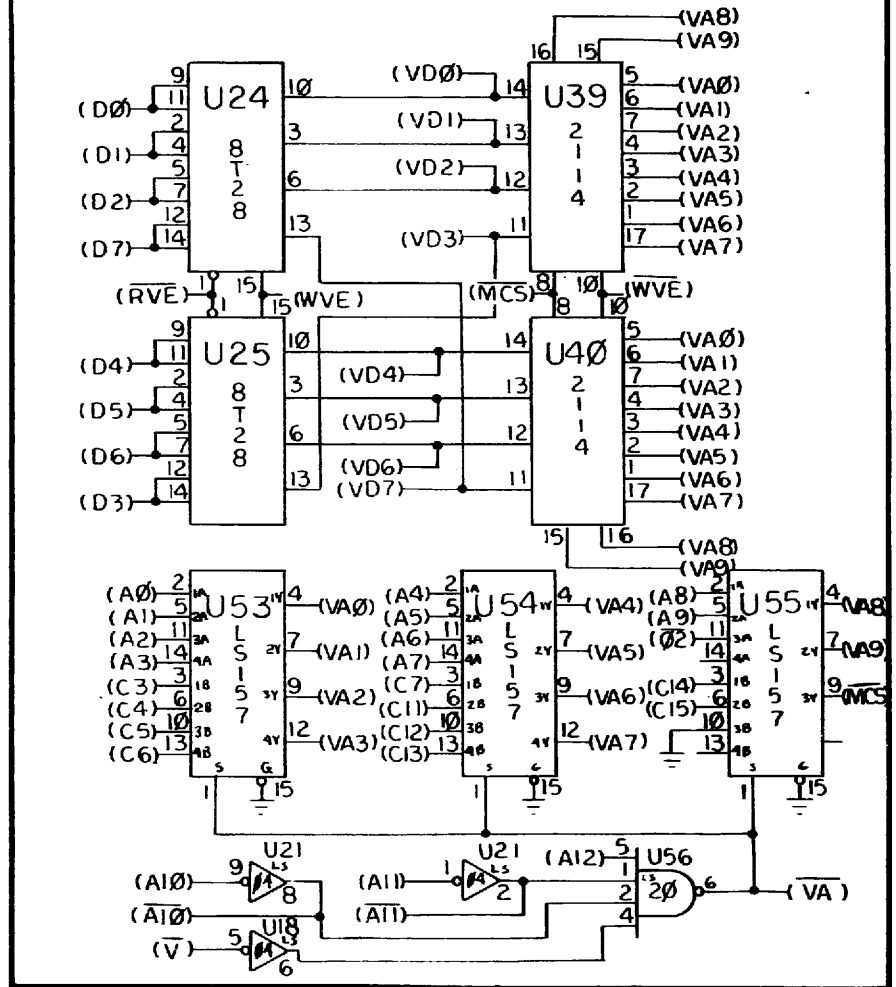


OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 6 of 13

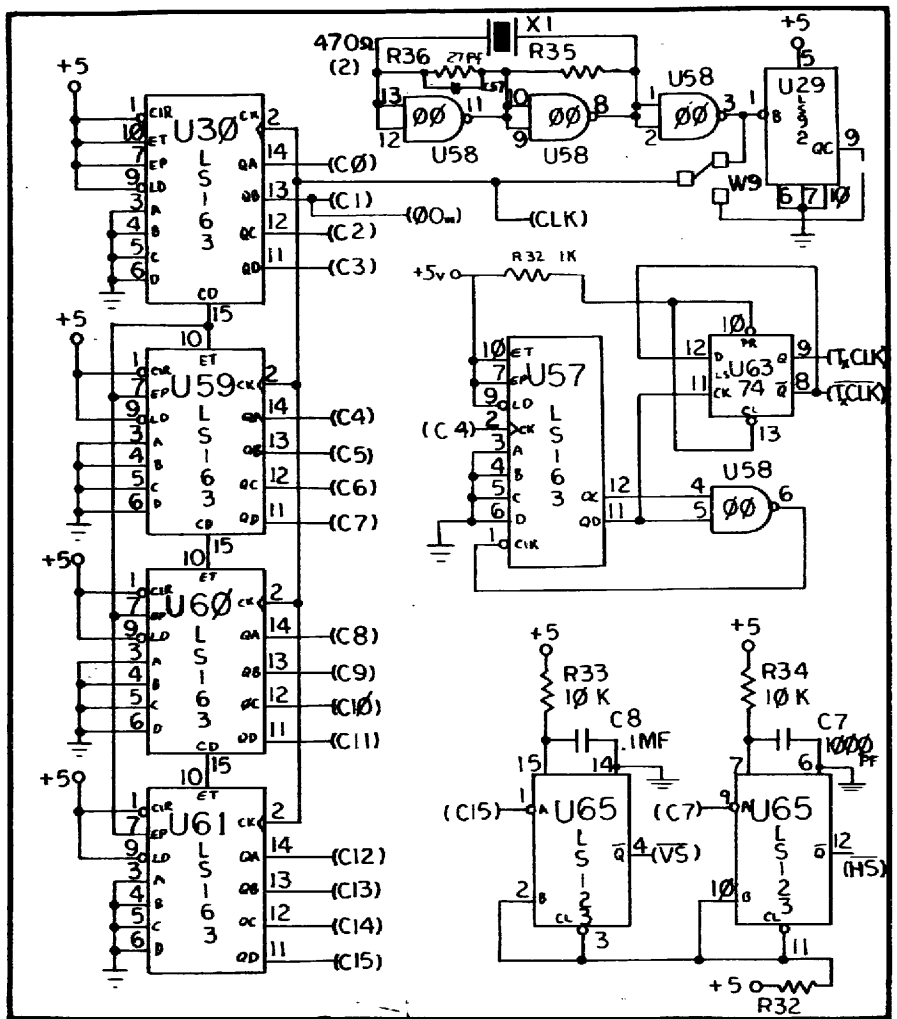




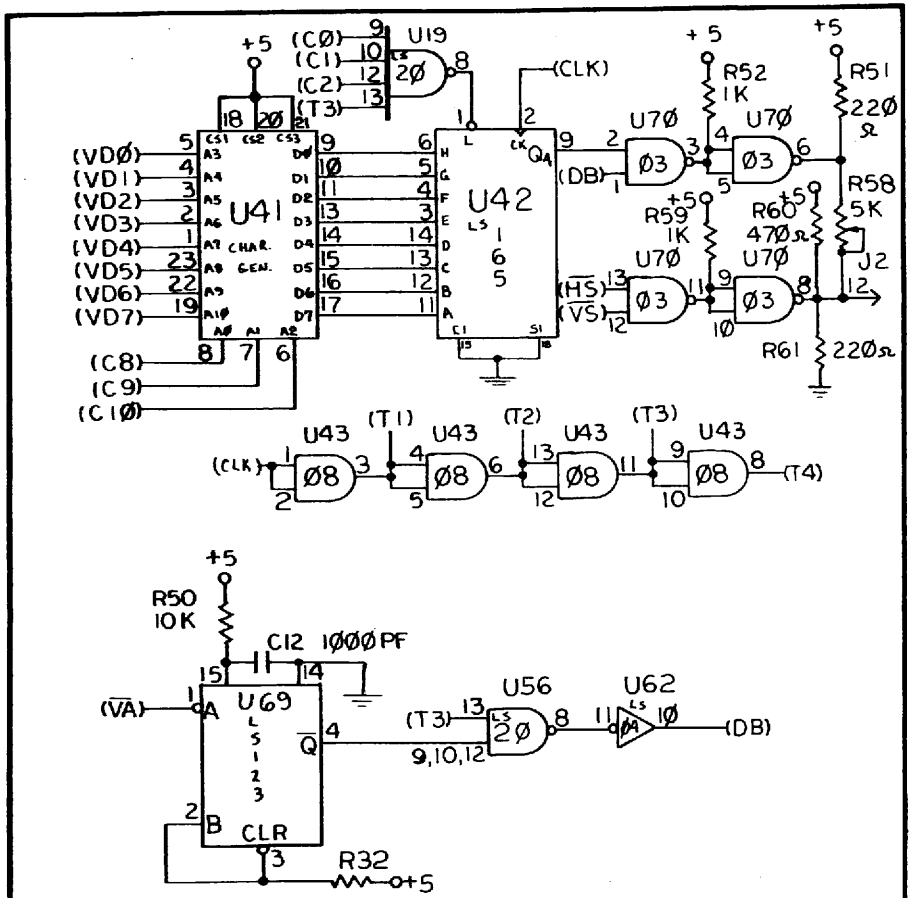
OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 7 of 13



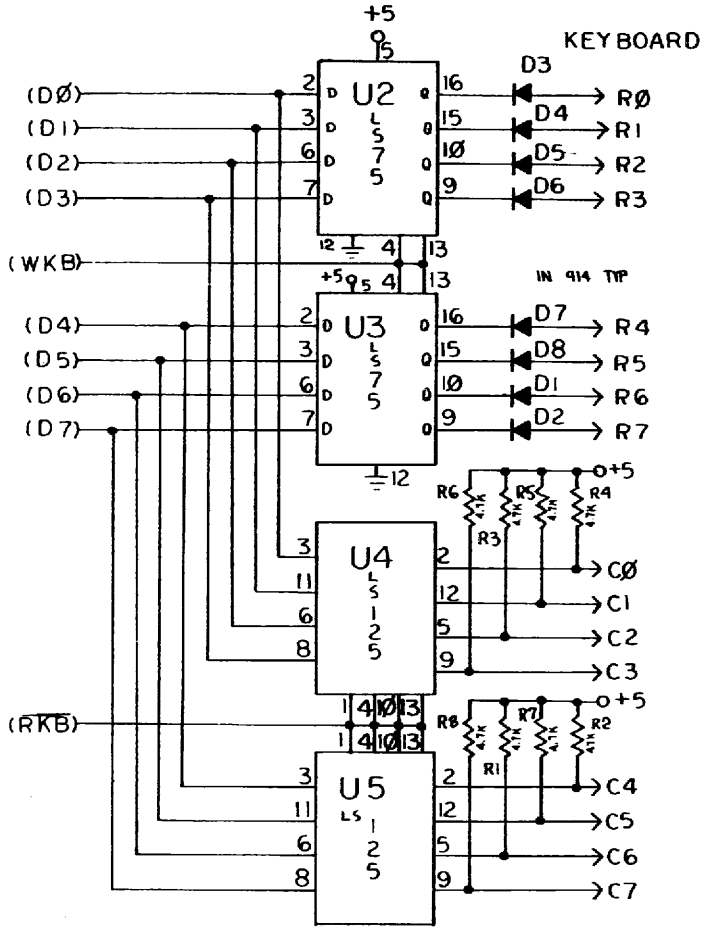
OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 8 of 13



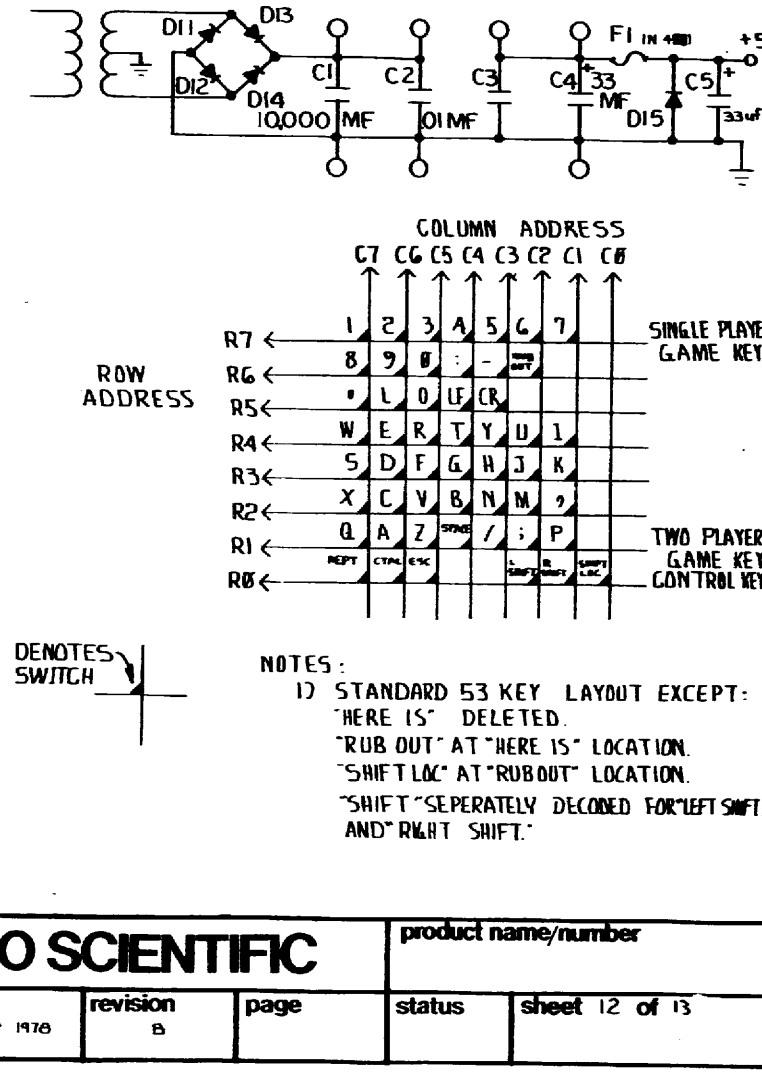
OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 9 of 13



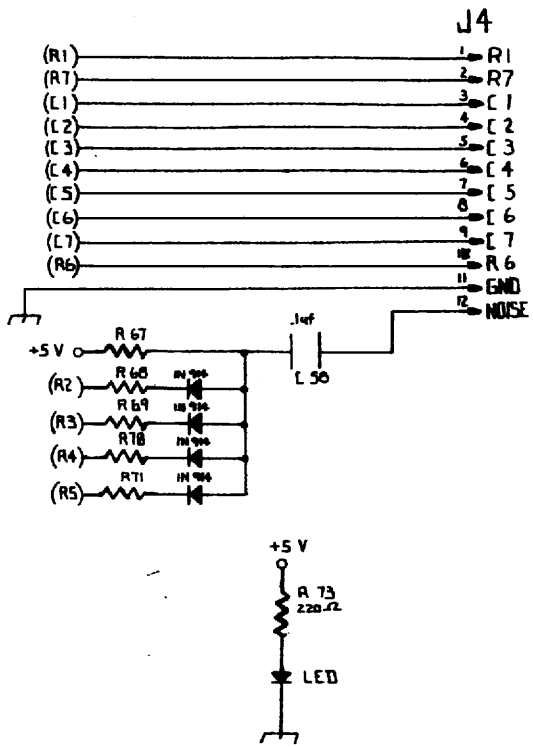
OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 10 of 13

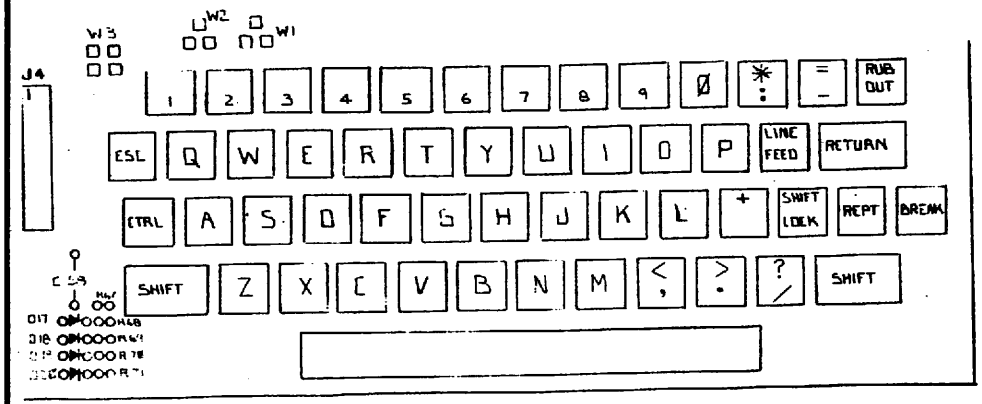


OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 11 of 13

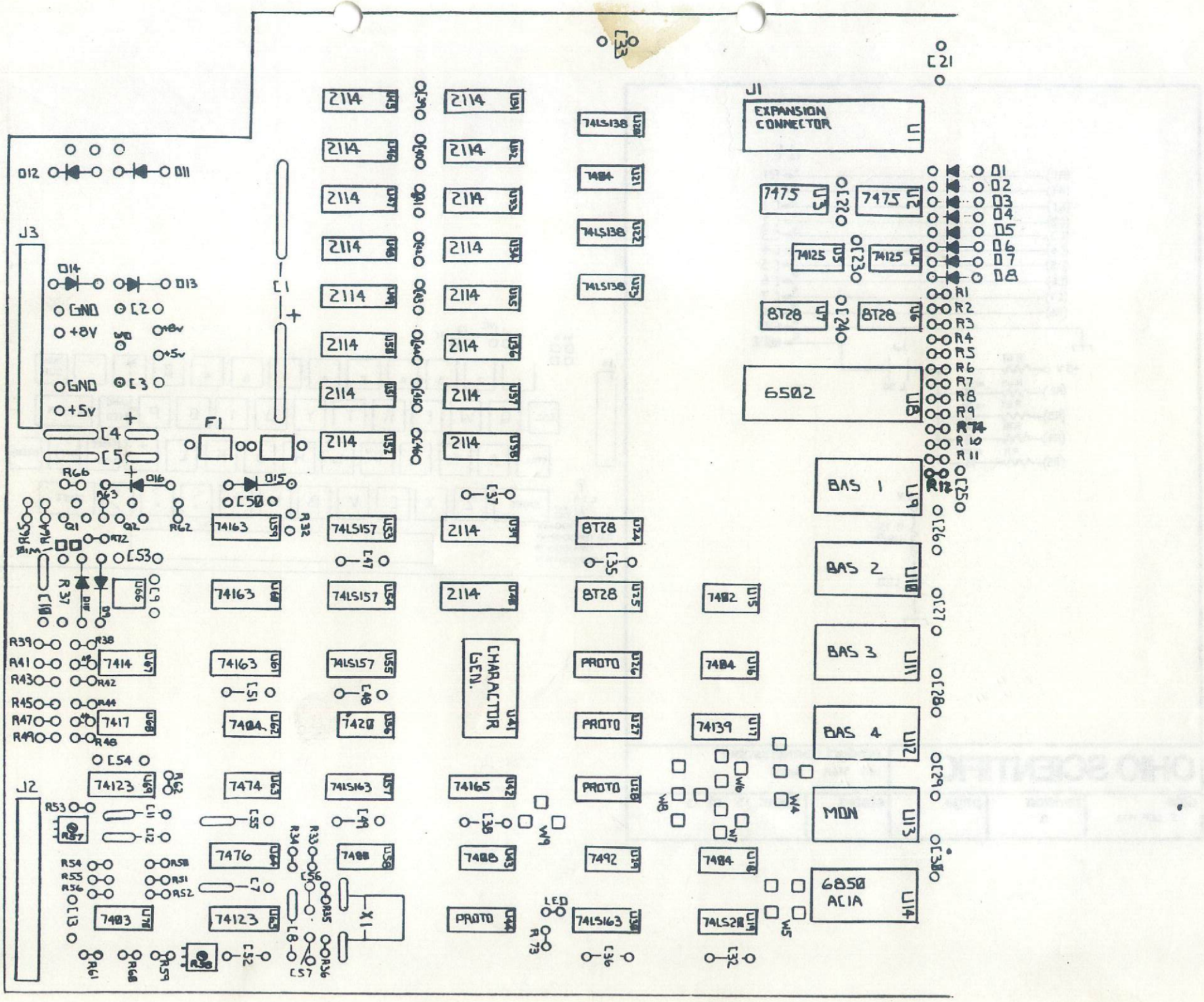


OHIO SCIENTIFIC			product name/number	
date 5 SEP 1978	revision B	page	status	sheet 12 of 13





OHIO SCIENTIFIC			product name/number JOY STICK + SOUND	
date 5 SEP 1978	revision B	page	status	sheet 13 of 13


J1
EXPANSION CONNECTOR
U1
J2
J3
F1
C1
2114
74LS138
74S04
7475
74125
8T28
6502
BAS 1
BAS 2
BAS 3
BAS 4
MON
6850 ACIA
CHARACTER GEN.
74163
74LS157
74S02
74S04
74139
74123
7474
74LS163
74165
7476
74S08
7492
74S04
74LS28
7414
7417
7420
PROTO
7403
LED
X1
D1
D2
D3
D4
D5
D6
D7
D8
D11
D12
D13
D14
D15
D16
GND
+8V
+5V
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12

Challenger I-P Memory Map (BASIC-in-ROM Configuration)

0000 - 00FF	Page Zero
0100 - 01FF	Stack
*0130	NMI Vector
*01C0	IRQ Vector
0200 - 0221	BASIC Flags & Vectors
*0203	LOAD Flag
*0205	SAVE Flag
*0218	Input Vector
*021A	Output Vector
*021C	Control C Check Vector
*021E	Load Vector
*0220	Save Vector
0222 - 02FA	Unused
0300 end of RAM	BASIC Workspace
A000 - BFFF	BASIC-in-ROM
D000 - D3FF	Video RAM
DF00	Polled Keyboard
F000 - F001	ACIA Serial Cassette Port
F800 - FBFF	ROM
FC00 - FCFF	ROM - Floppy Bootstrap
FD00 - FDFF	ROM - Polled Keyboard Input Routine
FE00 - FEFF	ROM - 65V Monitor
FF00 - FFFF	ROM - BASIC Support
*FFFA	NMI Vector
*FFFC	Reset Vector
*FFFE	IRQ Vector

Challenger I-P Memory Map Under P-DOS

0000 - 00FF	Page Zero
0100 - 01FF	Stack
*0130	NMI Vector
*01C0	IRQ Vector
0200 - 0221	BASIC Flags & Vectors
*0203	LOAD Flag
*0205	SAVE Flag
*0218	Input Vector
*021A	Output Vector
*021C	Control C Check Vector
*021E	Load Vector
*0220	Save Vector
0222 - 02FA	Unused
02FB - 20FF	P-DOS Workspace Pointers
- 22FA	BASIC Workspace under P-DOS (8K)
2300 - 317D	P-DOS
317E - 3FFF	Free
	End of 16K
4000 - 7FFF	Free
	End of 32K
A000 - BFFF	BASIC-in-ROM
C000 - C003	Floppy PIA
C010 - C011	Floppy ACIA
D000 - D3FF	Video RAM
DF00	Polled Keyboard
F000 - F001	ACIA Serial Cassette Port
F800 - FBFF	ROM
FC00 - FCFF	ROM - Floppy Bootstrap
FD00 - FEFF	ROM - 65V Monitor
FF00 - FFFF	ROM - BASIC Support
*FFFA	NMI Vector
*FFFC	Reset Vector
*FFFE	IRQ Vector

Challenger I-P Memory Map Under 65DV3.0

0000 - 00FF	Page Zero	
0100 - 01FF	Stack	
*0130	NMI Vector	
*01C0	IRQ Vector	
0200 - 0221	BASIC Flags & Vectors	
0200 - 22FA	Transient Processor Area Under 65DV3.0 for 9 digit BASIC Assembler/Editor	
2300 - 317D	65DV3.0 Drivers	
317E - 3FFF	Free	
	End of 16K	65DV3.0
4000 - 7FFF	Free	Object Code Workspace
	End of 32K	
A000 - BFFF	BASIC-in-ROM	
C000 - C003	Floppy PIA	
C010 - C011	Floppy ACIA	
D000 - D3FF	Video RAM	
DF00	Polled Keyboard	
F000 - F001	ACIA Serial Cassette Port	
F800 - FBFF	ROM	
FC00 - FCFF	ROM - Floppy Bootstrap	
FD00 - FDFF	ROM - Polled Keyboard Input Routine	
FE00 - FEFF	ROM - 65V Monitor	
FF00 - FFFF	ROM - BASIC Support	
*FFFA	NMI Vector	
*FFFC	Reset Vector	
*FFFE	IRQ Vector	

MOS TECHNOLOGY, INC.
VALLEY FORGE CORPORATE CENTER (215) 666 7950
950 RITTENHOUSE ROAD, NORRISTOWN, PA. 19401

PRELIMINARY

DATA

SHEET

MAY, 1976

MCS6500 MICROPROCESSORS

The MCS6500 Microprocessor Family Concept ----

The MCS6500 Series Microprocessors represent the first totally software compatible microprocessor family. This family of products includes a range of software compatible microprocessors which provide a selection of addressable memory range, interrupt input options and on-chip clock osscillators and drivers. All of the microprocessors in the MCS6500 group are software compatible within the group and are bus compatible with the M6800 product offering.

The family includes five microprocessors with on-board clock oscillators and drivers and four microprocessors driven by external clocks. The on-chip clock versions are aimed at high performance, low cost applications where single phase inputs, crystal or RC inputs provide the time base. The external clock versions are geared for the multi processor system applications where maximum timing control is mandatory. All versions of the microprocessors are available in 1 MHz and 2 MHz ("A" suffix on product numbers) maximum operating frequencies.

Features of the MCS6500 Family

- Single five volt supply
- N channel, silicon gate, depletion load technology
- Eight bit parallel processing
- 56 Instructions
- Decimal and binary arithmetic
- Thirteen addressing modes
- True indexing capability
- Programmable stack pointer
- Variable length stack
- Interrupt capability
- Non-maskable interrupt
- Use with any type or speed memory
- Bi-directional Data Bus
- Instruction decoding and control
- Addressable memory range of up to 65K bytes
- "Ready" input
- Direct memory access capability
- Bus compatible with MC6800
- Choice of external or on-board clocks
- 1MHz and 2MHz operation
- On-the-chip clock options
  * External single clock input
  * RC time base input
  * Crystal time base input
- 40 and 28 pin package versions
- Pipeline architecture

Members of the Family

Microprocessors with On-Board Clock Oscillator

- MCS6502
- MCS6503
- MCS6504
- MCS6505
- MCS6506

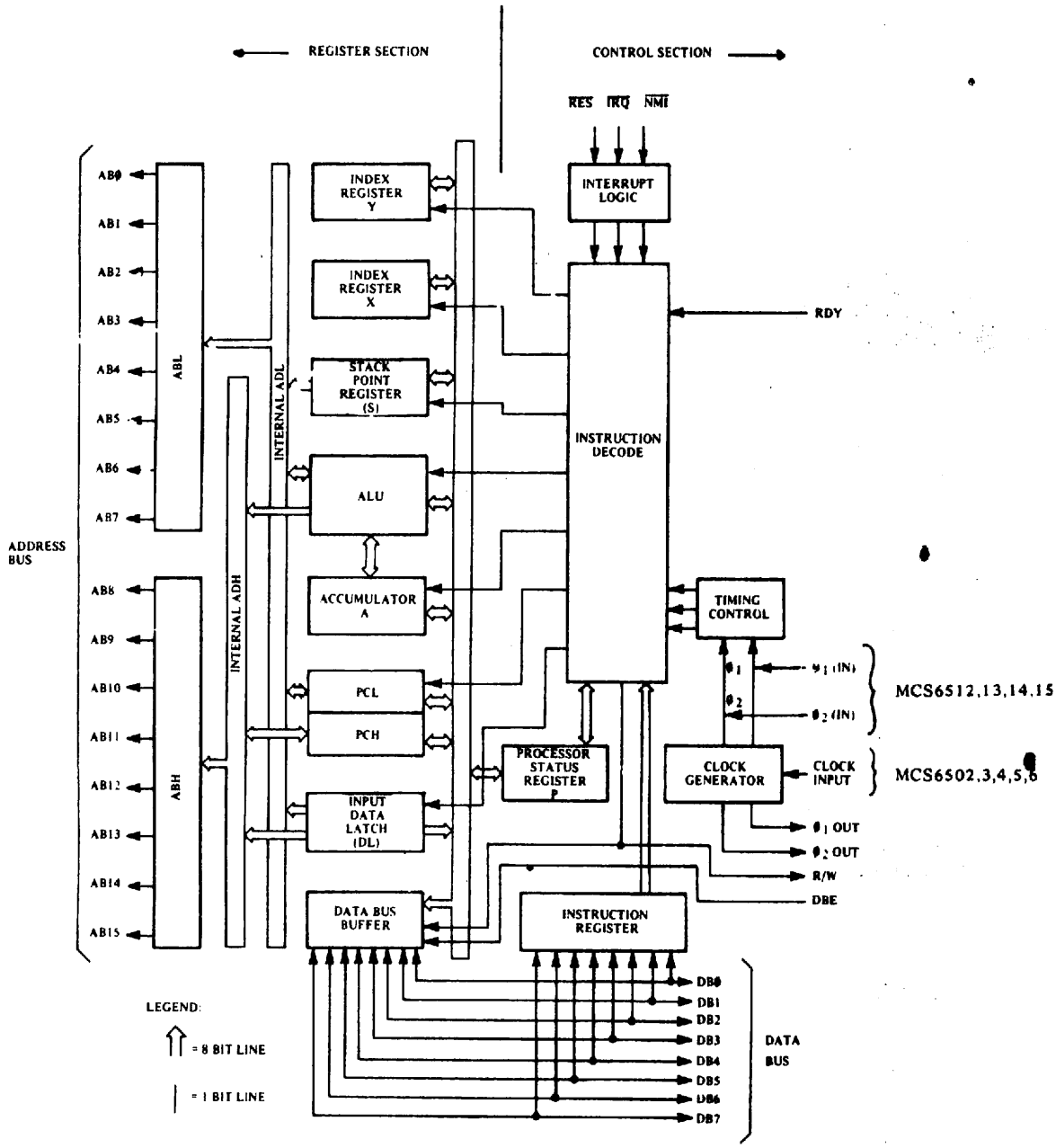
Microprocessors with External Two Phase Clock Input

- MCS6512
- MCS6513
- MCS6514
- MCS6515

Comments on the Data Sheet

The data sheet is constructed to review first the basic "Common Characteristics" - those features which are common to the general family of microprocessors. Subsequent to a review of the family characteristics will be sections devoted to each member of the group with specific features of each.

COMMON CHARACTERISTICS



Note: 1. Clock Generator is not included on MCS6512,13,14,15

2. Addressing Capability and control options vary with each of the MCS6500 Products.

MCS6500 Internal Architecture

COMMON CHARACTERISTICS

INSTRUCTION SET – ALPHABETIC SEQUENCE

ADC	Add Memory to Accumulator with Carry
AND	"AND" Memory with Accumulator
ASL	Shift left One Bit (Memory or Accumulator)
BCC	Branch on Carry Clear
BCS	Branch on Carry Set
BEQ	Branch on Result Zero
BIT	Test Bits in Memory with Accumulator
BMI	Branch on Result Minus
BNE	Branch on Result not Zero
BPL	Branch on Result Plus
BRK	Force Break
BVC	Branch on Overflow Clear
BVS	Branch on Overflow Set
CLC	Clear Carry Flag
CLD	Clear Decimal Mode
CLI	Clear Interrupt Disable Bit
CLV	Clear Overflow Flag
CMP	Compare Memory and Accumulator
CPX	Compare Memory and Index X
CPY	Compare Memory and Index Y
DEC	Decrement Memory by One
DEX	Decrement Index X by One
DEY	Decrement Index Y by One
EOR	"Exclusive-or" Memory with Accumulator
INC	Increment Memory by One
INX	Increment Index X by One
INY	Increment Index Y by One
JMP	Jump to New Location
JSR	Jump to New Location Saving Return Address
LDA	Load Accumulator with Memory
LDX	Load Index X with Memory
LDY	Load Index Y with Memory
LSR	Shift One Bit Right (Memory or Accumulator)
NOP	No Operation
ORA	"OR Memory with Accumulator
PHA	Push Accumulator on Stack
PHP	Push Processor Status on Stack
PLA	Pull Accumulator from Stack
PLP	Pull Processor Status from Stack
ROL	Rotate One Bit Left (Memory or Accumulator)
ROR	Rotate One Bit Right (Memory or Accumulator)
RTI	Return from Interrupt
RTS	Return from Subroutine
SBC	Subtract Memory from Accumulator with Borrow
SEC	Set Carry Flag
SED	Set Decimal Mode
SEI	Set Interrupt Disable Status
STA	Store Accumulator in Memory
STX	Store Index X in Memory
STY	Store Index Y in Memory
TAX	Transfer Accumulator to Index X
TAY	Transfer Accumulator to Index Y
TSX	Transfer Stack Pointer to Index X
TXA	Transfer Index X to Accumulator
TXS	Transfer Index X to Stack Pointer
TYA	Transfer Index Y to Accumulator

ADDRESSING MODES

ACCUMULATOR ADDRESSING - This form of addressing is represented with a one byte instruction, implying an operation on the accumulator.

IMMEDIATE ADDRESSING - In immediate addressing, the operand is contained in the second byte of the instruction, with no further memory addressing required.

ABSOLUTE ADDRESSING - In absolute addressing, the second byte of the instruction specifies the eight low order bits of the effective address while the third byte specifies the eight high order bits. Thus, the absolute addressing mode allows access to the entire 65K bytes of addressable memory.

ZERO PAGE ADDRESSING - The zero page instructions allow for shorter code and execution times by only fetching the second byte of the instruction and assuming a zero high address byte. Careful use of the zero page can result in significant increase in code efficiency.

INDEXED ZERO PAGE ADDRESSING - (X, Y indexing) - This form of addressing is used in conjunction with the index register and is referred to as "Zero Page, X" or "Zero Page, Y". The effective address is calculated by adding the second byte to the contents of the index register. Since this is a form of "Zero Page" addressing, the content of the second byte references a location in page zero. Additionally due to the "Zero Page" addressing nature of this mode, no carry is added to the high order 8 bits of memory and crossing of page boundaries does not occur.

INDEXED ABSOLUTE ADDRESSING - (X, Y indexing) - This form of addressing is used in conjunction with X and Y index register and is referred to as "Absolute, X", and "Absolute, Y". The effective address is formed by adding the contents of X or Y to the address contained in the second and third bytes of the instruction. This mode allows the index register to contain the index or count value and the instruction to contain the base address. This type of indexing allows any location referencing and the index to modify multiple fields resulting in reduced coding and execution time.

IMPLIED ADDRESSING - In the implied addressing mode, the address containing the operand is implicitly stated in the operation code of the instruction.

RELATIVE ADDRESSING - Relative addressing is used only with branch instructions and establishes a destination for the conditional branch.

The second byte of the instruction becomes the operand which is an "Offset" added to the contents of the lower eight bits of the program counter when the counter is set at the next instruction. The range of the offset is -128 to +127 bytes from the next instruction.

INDEXED INDIRECT ADDRESSING - In indexed indirect addressing (referred to as (Indirect,X)), the second byte of the instruction is added to the contents of the X index register, discarding the carry. The result of this addition points to a memory location on page zero whose contents is the low order eight bits of the effective address. The next memory location in page zero contains the high order eight bits of the effective address. Both memory locations specifying the high and low order bytes of the effective address must be in page zero

INDIRECT INDEXED ADDRESSING - In indirect indexed addressing (referred to as (Indirect),Y), the second byte of the instruction points to a memory location in page zero. The contents of this memory location is added to the contents of the Y index register, the result being the low order eight bits of the effective address. The carry from this addition is added to the contents of the next page zero memory location, the result being the high order eight bits of the effective address.

ABSOLUTE INDIRECT - The second byte of the instruction contains the low order eight bits of a memory location. The high order eight bits of that memory location is contained in the third byte of the instruction. The contents of the fully specified memory location is the low order byte of the effective address. The next memory location contains the high order byte of the effective address which is loaded into the sixteen bits of the program counter.

COMMON CHARACTERISTICS

Clocks (ϕ_1, ϕ_2)

The MCS651X requires a two phase non-overlapping clock that runs at the Vcc voltage level.

The MCS650X clocks are supplied with an internal clock generator. The frequency of these clocks is externally controlled. Details of this feature are discussed in the MCS6502 portion of this data sheet.

Address Bus (A_0-A_{15}) (See sections on each micro for respective address lines on those devices.)

These outputs are TTL compatible, capable of driving one standard TTL load and 130pf.

Data Bus (D_0-D_7)

Eight pins are used for the data bus. This is a bi-directional bus, transferring data to and from the device and peripherals. The outputs are tri-state buffers capable of driving one standard TTL load and 130pf.

Data Bus Enable (DBE)

This TTL compatible input allows external control of the tri-state data output buffers and will enable the microprocessor bus driver when in the high state. In normal operation DBE would be driven by the phase two (ϕ_2) clock, thus allowing data output from microprocessor only during ϕ_2. During the read cycle, the data bus drivers are internally disabled, becoming essentially an open circuit. To disable data bus drivers externally, DBE should be held low.

Ready (RDY)

This input signal allows the user to single cycle the microprocessor on all cycles except write cycles. A negative transition to the low state during or coincident with phase one (ϕ_1) will halt the microprocessor with the output address lines reflecting the current address being fetched. This condition will remain through a subsequent phase two (ϕ_2) in which the Ready signal is low. This feature allows microprocessor interfacing with low speed PROMS as well as fast (max. 2 cycle) Direct Memory Access (DMA). If Ready is low during a write cycle, it is ignored until the following read operation.

Interrupt Request ($\overline{IRQ}$)

This TTL level input requests that an interrupt sequence begin within the microprocessor. The microprocessor will complete the current instruction being executed before recognizing the request. At that time, the interrupt mask bit in the Status Code Register will be examined. If the interrupt mask flag is not set, the microprocessor will begin an interrupt sequence. The Program Counter and Processor Status Register are stored in the stack. The microprocessor will then set the interrupt mask flag high so that no further interrupts may occur. At the end of this cycle, the program counter low will be loaded from address FFFE, and program counter high from location FFFF, therefore transferring program control to the memory vector located at these addresses. The RDY signal must be in the high state for any interrupt to be recognized. A 3KΩ external resistor should be used for proper wire-OR operation.

Non-Maskable Interrupt ($\overline{NMI}$)

A negative going edge on this input requests that a non-maskable interrupt sequence be generated within the microprocessor.

$\overline{NMI}$ is an unconditional interrupt. Following completion of the current instruction, the sequence of operations defined for $\overline{IRQ}$ will be performed, regardless of the state interrupt mask flag. The vector address loaded into the program counter, low and high, are locations FFFA and FFFB respectively, thereby transferring program control to the memory vector located at these addresses. The instructions loaded at these locations cause the microprocessor to branch to a non-maskable interrupt routine in memory.

$\overline{NMI}$ also requires an external 3KΩ register to Vcc for proper wire-OR operations.

Inputs $\overline{IRQ}$ and $\overline{NMI}$ are hardware interrupts lines that are sampled during ϕ_2 (phase 2) and will begin the appropriate interrupt routine on the ϕ_1 (phase 1) following the completion of the current instruction.

Set Overflow Flag (S.O.)

A NEGATIVE going edge on this input sets the overflow bit in the Status Code Register. This signal is sampled on the trailing edge of ϕ_1.

SYNC

This output line is provided to identify those cycles in which the microprocessor is doing an OP CODE fetch. The SYNC line goes high during ϕ_1 of an OP CODE fetch and stays high for the remainder of that cycle. If the RDY line is pulled low during the ϕ_1 clock pulse in which SYNC went high, the processor will stop in its current state and will remain in the state until the RDY line goes high. In this manner, the SYNC signal can be used to control RDY to cause single instruction execution.

Reset

This input is used to reset or start the microprocessor from a power down condition. During the time that this line is held low, writing to or from the microprocessor is inhibited. When a positive edge is detected on the input, the microprocessor will immediately begin the reset sequence.

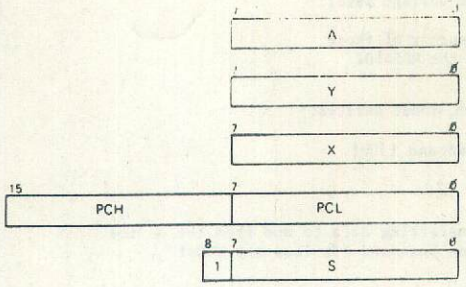
After a system initialization time of six clock cycles, the mask interrupt flag will be set and the microprocessor will load the program counter from the memory vector locations FFFC and FFFD. This is the start location for program control.

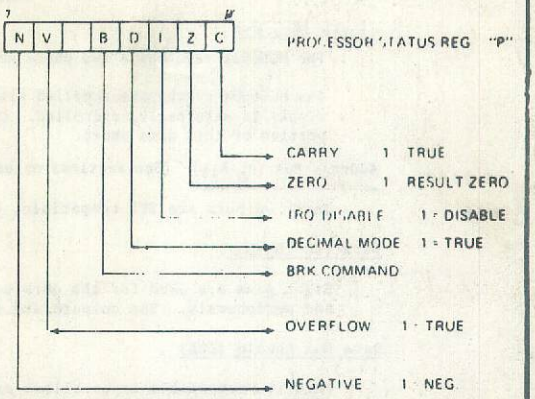
After Vcc reaches 4.75 volts in a power up routine, reset must be held low for at least two clock cycles. At this time the R/W and (SYNC) signal will become valid.

When the reset signal goes high following these two clock cycles, the microprocessor will proceed with the normal reset procedure detailed above.

COMMON CHARACTERISTICS

PROGRAMMING MODEL





INSTRUCTION SET – OP CODES, Execution Time, Memory Requirements

INSTRUCTIONS		IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM.			IMPLIED			(IND,X)			(IND),Y			Z,PAGE,X			ABS,X			ABS,Y			RELATIVE			INDIRECT			Z,PAGE,Y			CONDITION CODES					
MNEMONIC	OPERATION	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N	Z	C	I	D	V
ADC	A+M+C → A (4) (1)	69	2	2	6D	4	3	65	3	2							61	6	2	71	5	2	75	4	2	7D	4	3	79	4	3										✓	✓	✓	-	-	✓
AND	A∧M → A (1)	29	2	2	2D	4	3	25	3	2							21	6	2	31	5	2	35	4	2	3D	4	3	39	4	3										✓	✓	-	-	-	-
ASL	C 7 … 0 ← 0				0E	6	3	06	5	2	0A	2	1										16	6	2	1E	7	3													✓	✓	✓	-	-	-
BCC	BRANCH ON C=0 (2)																															90	2	2							-	-	-	-	-	-
BCS	BRANCH ON C=1 (2)																															B0	2	2							-	-	-	-	-	-
BEQ	BRANCH ON Z=1 (2)																															F0	2	2							-	-	-	-	-	-
BIT	A∧M				2C	4	3	24	3	2																															M_7	✓	-	-	M_6
BMI	BRANCH ON N=1 (2)																															30	2	2							-	-	-	-	-	-
BNE	BRANCH ON Z=0 (2)																															D0	2	2							-	-	-	-	-	-
BPL	BRANCH ON N=0 (2)																															10	2	2							-	-	-	-	-	-
BRK	(See Fig 1)													00	7	1																									-	-	-	1	-	-
BVC	BRANCH ON V=0 (2)																															50	2	2							-	-	-	-	-	-
BVS	BRANCH ON V=1 (2)																															70	2	2							-	-	-	-	-	-
CLC	0 → C													18	2	1																									-	-	0	-	-	-
CLD	0 → D													D8	2	1																									-	-	-	-	0	-
CLI	0 → I													58	2	1																									-	-	-	0	-	-
CLV	0 → V													B8	2	1																									-	-	-	-	-	0
CMP	A-M (1)	C9	2	2	CD	4	3	C5	3	2							C1	6	2	D1	5	2	D5	4	2	DD	4	3	D9	4	3										✓	✓	✓	-	-	-
CPX	X-M	E0	2	2	EC	4	3	E4	3	2																															✓	✓	✓	-	-	-
CPY	Y-M	C0	2	2	CC	4	3	C4	3	2																															✓	✓	✓	-	-	-
DEC	M-1 → M				CE	6	3	C6	5	2													D6	6	2	DE	7	3													✓	✓	-	-	-	-
DEX	X-1 → X													CA																											✓	✓	-	-	-	-
DEY	Y-1 → Y													88																											✓	✓	-	-	-	-
EOR	A ⊻ M → A (1)	49	2	2	4D	4	3	45	3	2							41	6	2	51	5	2	55	4	2	5D	4	3	59	4	3										✓	✓	-	-	-	-
INC	M+1 → M				EE	6	3	E6	5	2													F6	6	2	FE	7	3													✓	✓	-	-	-	-
INX	X+1 → X													E8	2	1																									✓	✓	-	-	-	-
INY	Y+1 → Y													C8	2	1																									✓	✓	-	-	-	-
JMP	JUMP TO NEW LOC				4C	3	3																												6C	5	3				-	-	-	-
JSR	(See Fig 2) JUMP SUB				20	6	3																																		-	-	-	-	-	-
LDA	M → A (1)	A9	2	2	AD	4	3	A5	3	2							A1	6	2	B1	5	2	B5	4	2	BD	4	3	B9	4	3										✓	✓	-	-	-	-

MNEMONIC	OPERATION	IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM			IMPLIED			(IND,X)			(IND),Y			Z,PAGE,X			ABS,X			ABS,Y			RELATIVE			INDIRECT			Z,PAGE,Y			CONDITION CODES					
		OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N	Z	C	I	D	V
LDX	M → X (1)	A2	2	2	AE	4	3	A6	3	2																			BE	4	3										✓	✓				
LDY	M → Y (1)	A0	2	2	AC	4	3	A4	3	2													B4	4	2	BC	4	3													✓	✓				
LSR	0 → 7 … 0 → C				4E	6	3	46	5	2	4A	2	1										56	6	2	5E	7	3													0	✓	✓			
NOP	NO OPERATION													EA	2	1																														
ORA	A∨M → A	09	2	2	0D	4	3	05	3	2							01	6	2	11	5	2	15	4	2	1D	4	3	19	4	3										✓	✓				
PHA	A → M_S S-1 → S													48	3	1																													
PHP	P → M_S S-1 → S													08	3	1																													
PLA	S+1 → S M_S → A													68	4	1																									✓	✓				
PLP	S+1 → S M_S → P													28	4	1																									(RESTORED)					
ROL	C 7 … 0				2E	6	3	26	5	2	2A	2	1										36	6	2	3E	7	3													✓	✓	✓			
ROR	C 7 … 0				6E	6	3	66	5	2	6A	2	1										76	6	2	7E	7	3													✓	✓	✓			
RTI	(See Fig 1) RTRN INT													40	6	1																									(RESTORED)					
RTS	(See Fig 2) RTRN SUB													60	6	1																													
SBC	A-M-C̄ → A (1)	E9	2	2	ED	4	3	E5	3	2							E1	6	2	F1	5	2	F5	4	2	FD	4	3	F9	4	3										✓	✓	(3)			✓
SEC	1 → C													38	2	1																													
SED	1 → D													F8	2	1																													
SEI	1 → I													78	2	1																													
STA	A → M				8D	4	3	85	3	2							81	6	2	91	6	2	95	4	2	9D	5	3	99	5	3														
STX	X → M				8E	4	3	86	3	2																													96	4	2						
STY	Y → M				8C	4	3	84	3	2													94	4	2																					
TAX	A → X													AA	2	1																									✓	✓				
TAY	A → Y													A8	2	1																									✓	✓				
TSX	S → X													BA	2	1																									✓	✓				
TXA	X → A													8A	2	1																									✓	✓				
TXS	X → S													9A	2	1																													
TYA	Y → A													98	2	1																									✓	✓				

(1) ADD 1 TO "N" IF PAGE BOUNDARY IS CROSSED

(2) ADD 1 TO "N" IF BRANCH OCCURS TO SAME PAGE
ADD 2 TO "N" IF BRANCH OCCURS TO DIFFERENT PAGE

(3) CARRY NOT BORROW

(4) IN DECIMAL MODE Z FLAG IS INVALID
ACCUMULATOR MUST BE CHECKED FOR ZERO CONDITION

X INDEX X
Y INDEX Y
A ACCUMULATOR
M MEMORY PER EFFECTIVE ADDR
S MEMORY PER STACK POINTER

✓ TEST FOR CONDITION
- NOT MODIFIED
0 MEMORY BIT
1 MEMORY BIT

N NO. CYCLES
NO. BYTES