OHIO SCIENTIFIC

The C1P Users Manual

TABLE OF CONTENTS

APPENDIX

TABLE OF FIGURES

SECTION 2

GENERAL INTRODUCTION

Ohio Scientific's Challenger 1P line is the most economical of the Ohio Scientific family of microcomputers. In spite of its economical price, the C1P includes many deluxe features usually found only in more expensive systems. The standard model Challenger 1P features BASIC-in-ROM and is an attractive fully packaged personal computer ready to run as delivered. The basic Challenger 1P Series 2 includes a standard audio cassette interface and is capable of sound, music and voice output via a built-in digital to analog converter. The basic system can be easily and economically expanded to include up to 32K of RAM memory, dual mini-floppy drives, printer, modem, and color display.

The C1P Series 2 personal computer is specifically designed for the first-time personal computer user and for use in educational environments. Its advanced features allow a wide range of home applications including, for example, the following.

PERSONAL OR HOME COMPUTERS

Challenger 1P's advanced character graphics, noise-free display, programmable keyboard, and extremely high speed BASIC make it capable of spectacular video displays, cartoons, animated advertisements, and elaborate computer games. Ohio Scientific offers an extensive library of one and two player video games very similar to conventional arcade games as well as a standard complement of computer-type games. Ohio Scientific's software library also includes examples of cartoons, advertisements, and educational games which make extensive use of graphics and programmable keyboard inputs. The computer's fast program execution makes such applications a snap to program.

PERSONAL FINANCES

Challenger 1P's floating point decimal arithmetic capability in conjunction with its cassette storage abilities make it practical for many forms of personal financial aid and analysis. Ohio Scientific's cassette library includes a check book balancing program, savings account program, and annuity and loan analysis programs. Budget planning aids include home ownership cost analysis and expense accounting. A complete home budget system is available for use on the C1P MF Series 2 disk system. Demonstration programs provide personal calendars, phone directory, address book, and other personal services such as dietary analysis.

It should be pointed out that a mini-floppy disk is a practical necessity for some of the advanced applications mentioned above. These capabilities can, however, be effectively demonstrated on a cassette system. As in all applications, the ease of programming in BASIC, along with the convenient features of decimal arithmetic capability and cassette storage, make user-generated applications in these areas easy to program.

SCIENTIFIC CALCULATIONS AND ADVANCED MATHEMATICAL ANALYSIS

Challenger 1P's BASIC has full advanced arithmetic capability, including trigonometric functions, logarithms, exponentiation, and full scientific notation. These features are available in the immediate mode of operation as well as the stored program mode. For instance, a user can quickly turn the computer on, type in an equation as a single line, and press return to get an answer. The computer can double as an advanced scientific calculator with much greater ease of use than any available calculator.

The program storage and alphanumeric capability of the Challenger 1P make it extremely valuable to engineers, students, and educators for solving scientific, engineering and mathematical analysis problems. Ohio Scientific's cassette library includes several advanced mathematics oriented programs including a programmable calculator simulator and a mathematical function library. The library also includes applications programs such as definite integrals, statistical analysis, and other complex mathematical functions. In general the Challenger 1P will be hundreds of times faster than the most powerful scientific calculators in "number-crunching" applications.

EDUCATION

Challenger 1P series personal computers are extremely versatile in educational computing applications. Once the user gets involved in the educational applications of these machines, he will quickly consider computers a necessity in the educational process.

Young children from kindergarten to grade six are especially attracted to computers. As the child's reading ability develops they quickly master the elementary operations of the computer. It is not at all unusual for six year old children to respond to mathematical problems on a personal computer. Children's natural fascination with computers in conjunction with the 1P's cartoon-like interactive capability make the computer highly valuable in a modern educational environment. Programs which teach, tutor and drill students in virtually all areas of education can be easily programmed on the Challenger 1P system. Ohio Scientific has a full library of several types of educational games which can be used as an example in programming such applications. These programs range from a simple "Sesame Street" type arithmetic cartoon through mathematical drills, to word games such as "Hangman" where the gallows, noose and person are actually constructed graphically on the screen as the child attempts to guess the letters of a word.

Another broad area of education is in teaching computer fundamentals. The Challenger 1P utilizes the most popular upper level language, BASIC, in a very complete and concise implementation. With the Challenger 1P the user can teach or learn BASIC in conjunction with any of the common available text books on the BASIC programming language. The 1P series machines have full machine code accessibility including the machine code monitor so that advanced students can enter, edit and execute machine code programs. A very fast and interactive assembler/editor is available to run on Challenger 1P machines so that students can be introduced to the concepts of assembler programming and editing.

ADVANCED APPLICATIONS

There are many other applications of the basic 1P machines that have not been mentioned here. The Challenger 1P MF system provides the user with the extra convenience of virtually instantaneous loading and storing of programs on mini-floppy disks. The addition of a mini-floppy disk drive to the Challenger 1P also provides convenient construction and access of data files. Using the file capabilities of the C1P MF, an educator can develop an interactive textbook with a quick access data base for any educational topic. In the home, the data file operation of the mini-floppy makes the Challenger 1P a deluxe personal service computer giving the user easy access to phone numbers, personal calendar, addresses and other file-type information.

The Challenger 1P is available in an uncased version as the Superboard II. For a personal computer enthusiast on a limited budget, the Superboard II offers nearly all of the features of the basic Challenger 1P at a fraction of the cost. Setting up a Superboard is discussed in section three. Essentially all that is required, other than a little time, is a 5 volt @ 3 A (minimum) regulated power supply and an AC/DC voltmeter. As with the standard Challenger 1P, the Superboard II includes both a standard audio cassette interface and a video display interface.

The remainder of this section gives an overview of the Challenger 1P system. Although the presentation is reasonably nontechnical, it uses several terms which are part of the standard vocabulary of computers. The meanings of many of these terms will be clear from the context in which they are used. Appendix 1 includes a computer glossary which summarizes the meaning of most of these technical terms.

Like all small computers the Challenger 1P is made up of several modules. The most important of these, the microprocessor, forms the heart or CPU (Central Processing Unit) of the C1P system. This microprocessor is an integrated circuit much like those used on digital watches and calculators. It performs all of the logical and arithmetic operations required by the computer. The Challenger 1P system is based upon the 65Ø2 microprocessor.

In addition to the microprocessor, any computer system requires memory for storage of data and programs and input/output (I/O) devices to allow it to communicate with the user. Memory in a computer can be thought of as a collection of post office boxes each having a specific address or box number. The addressing scheme used by the Challenger 1P system is discussed in Appendix 2. Two basic types of memory are present in the C1P. They are referred to as ROM (for Read Only Memory) and RAM (for Random Access Memory). Each memory location (post office box) can contain one piece (or BYTE) of data at any given time. The numeric value of the data at any memory location is restricted to the range Ø to 255. This restriction arises from the fact that each BYTE is eight BITS (binary digits) in length. Appendix 2 gives a brief introduction to the binary and hexadecimal number systems.

When the microcomputer is operating, it can read (or PEEK) the contents of the memory location or it can write (or POKE) a new value into a memory location. The contents of ROM memory is preprogrammed and unchangeable by the user. Thus the user can read (or PEEK) the contents of ROM memory but cannot POKE a new value into ROM memory. All models of the Challenger 1P line include BASIC, the most commonly used programming language, in 8K (K is an abbreviation for kilo, for computers 1K = 1Ø24 bytes) of ROM. RAM memory provides modifiable storage comprising the workspace, that is an area in memory where programs and data can be written in and read out repeatedly. Generally, RAM "forgets" (is erased) when the power is turned off.

The simplest means of communication between the user and the C1P is via the keyboard and the television or video monitor. Sections three through six describe setting up and running your C1P using only keyboard input and the video display. The conventional computer style 53-key keyboard supports both upper and lower case characters. Each key has full automatic repeat. Holding down any key transmits first one character and then, after approximately a half second delay, a repeat factor of five characters per second. The keyboard of the Challenger 1P series is a polled or software scanned keyboard. This allows the user to program individual keystrokes for specific functions. This feature, which is described in detail in section twelve, is especially useful in real time video game applications. The built-in video display interface is capable of generating 256 distinct characters including upper and lower case letters as well as graphics and gaming character elements. The display format is 32 rows × 32 columns of 8 × 8 dot or pixel characters, but due to the overscan present on normal television equipment you will actually see 24 rows × 24 columns. The Series 2 models in the Challenger 1P line feature a program selectable 12 row × 48 column character display option. Section nine includes a detailed discussion of the character display capabilities of the Challenger 1P system.

All models in the Challenger 1P line include a high reliability audio cassette interface allowing inexpensive cassette storage and playback of programs. The procedures for cassette storage and playback of programs are described in detail in sections seven and eleven.

All Challenger 1P cassette based computers come with a demonstration library on cassette (the C1P sampler) which gives the user some insight into the capability of the computers. This demonstration cassette contains six programs which demonstrate various aspects of personal computing. The cassette includes an advanced video game called "Star Wars" which runs in real time, a check book balancing program, a mathematics skill drill for children, and an example of an educational program. Also included are "Counter" which is designed to be a child's first introduction to a computer, and a sample of a tutor program called "Trig Tutor" which shows the use of graphics in tutoring complicated concepts. Ohio Scientific offers a full library of very economical cassette programs for the Challenger 1P system.

The standard cassette based C1P Series 2 has 18K total RAM/ROM with 8K of workspace. Although the workspace can be expanded to 32K, 8K is a practical upper limit for cassette based computers because of the load time for programs from cassette into an 8K workspace. As the user upgrades to 20K of RAM, he will find it desirable to convert his system to a mini-floppy disk based system. The C1P MF is a standard C1P with a 61Ø floppy disk and memory board, an extra power supply and a single mini-floppy. A cassette based C1P can be expanded to a C1P MF at a total cost just slightly more than purchasing a C1P MF outright.

The C1P MF is supplied with 30K total RAM/ROM, a single 90K byte fast access mini-floppy and two disk operating systems. PICO DOS allows the operation of ROM BASIC and cassette originated programs on diskette. OS-65D is a powerful business and development oriented system with 9-digit BASIC by Microsoft. OS-65D also supports an optional interactive Assembler/Editor, an optional text editor and both random access and sequential data files. The use of these two operating systems on the Challenger 1P is discussed in section eleven. With the C1P MF system, the user can load and run programs from diskette in a fraction of a second.

Ohio Scientific offers a wide range of educational, personal, entertainment and small business software on diskette. Ohio Scientific also offers "OS-MDMS," a mini data base management system, for use on the C1P MF. This data base management system allows the user to store collections of information on diskette for instant recall without requiring any programming knowledge.

Generally, Ohio Scientific floppy diskette software is much less expensive than cassette software simply because of the much lower cost of mass duplicating diskettes. For instance, a typical Ohio Scientific applications mini-floppy will have ten programs on it and cost a fraction of what the individual programs would cost if purchased separately on cassette. If a large software library is contemplated, the mini-floppy system will not only provide much faster program LOAD and SAVE capabilities but will also be more cost effective than purchasing a large number of audio cassettes.

The 63Ø I/O Expander board is available for addition to either the C1P or C1P MF. This board provides the C1P with the state-of-the-art in input/output capabilities rivaling the most expensive small computer systems available today. This board allows easy interface with joysticks, remote keypads, AC remote control units, home security system and more. It also substantially enhances the video display capabilities of the Series 2 models in the Challenger 1P line by allowing the display of up to 16 colors with any of the standard 256 graphics characters. The features of the 63Ø I/O Expander board are described in sections fifteen through nineteen.

SECTION 3

UNPACKING INSTRUCTIONS

This section details the procedures to be followed during the unpacking and assembly of a Challenger 1P system. The instructions given here are intended to supplement the introductory manual supplied with each computer.

Your Challenger 1P system is shipped from the factory in a carton designed to provide maximum protection from damage in transit. Nevertheless you should inspect the box carefully for signs of rough handling such as punctures or crushed sides. If there is external evidence of damage, check the contents of the box carefully. (If possible without removing the equipment.) If the contents have sustained damage notify the carrier immediately.

The system should be unpacked carefully and all packing material should be saved. These materials may be needed later to transport or ship the system. If your system is diskette based, remove and save the dummy cardboard diskette which protects the disk head from damage during shipment. This dummy diskette should be saved to be used later if the system is shipped or transported.

Assembling a C1P system requires essentially the same precautions as hooking up a stereo system. Although the Challenger 1P is a relatively rugged solid-state device, it may be damaged if you fail to observe power supply, accessory or safe operating requirements. As with all electronic equipment, it is recommended that you take time to read all the instructions carefully before you turn on the computer. Once you are familiar with these procedures, you can explore other areas of personal computing at your own pace.

Figure 1A is a diagram of the rear panel of the standard model of the Challenger 1P Series 2. This diagram will be referred to repeatedly throughout sections three through fourteen. If your Challenger 1P Series 2 is equipped with the 63Ø I/O Expander board then the rear panel will appear as in Figure 1B. Sections fifteen through nineteen will frequently refer to this diagram.

POWER SUPPLY-CHALLENGER 1P

The Challenger 1P requires a three-wire grounded 11ØV outlet. Although adapters are readily available which allow plugging a three prong plug into a non-grounded two-wire outlet, such an adapter m-u-s-t n-o-t be used unless you run a ground from the frame of the computer to a good ground such as a cold water pipe. This will insure that the computer's cabinet is thoroughly grounded and will protect both you and the computer from possible damage or shock.

POWER SUPPLY-SUPERBOARD II

The Superboard II requires a 5 volt @ 3 A (minimum) regulated power supply with a +−5% maximum ripple. Reasonable care must be exercised when working with the Superboard II. Your work area should be clear of paper clips or other conductive material which might accidentally contact the foil on the board. The board should not be flexed or bent.

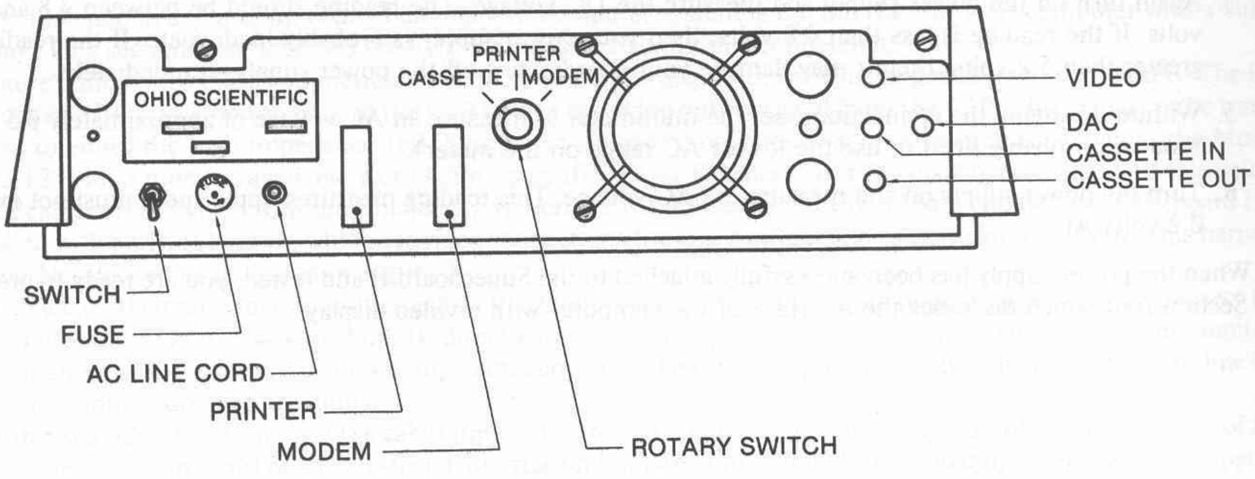


Figure 1A: C1P Series II Standard Rear Panel

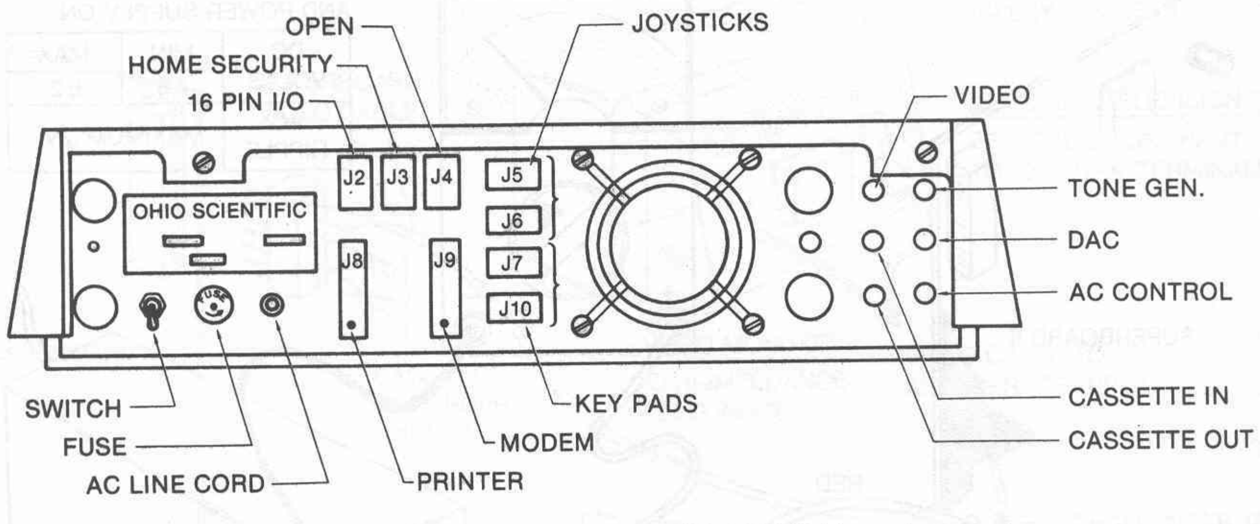
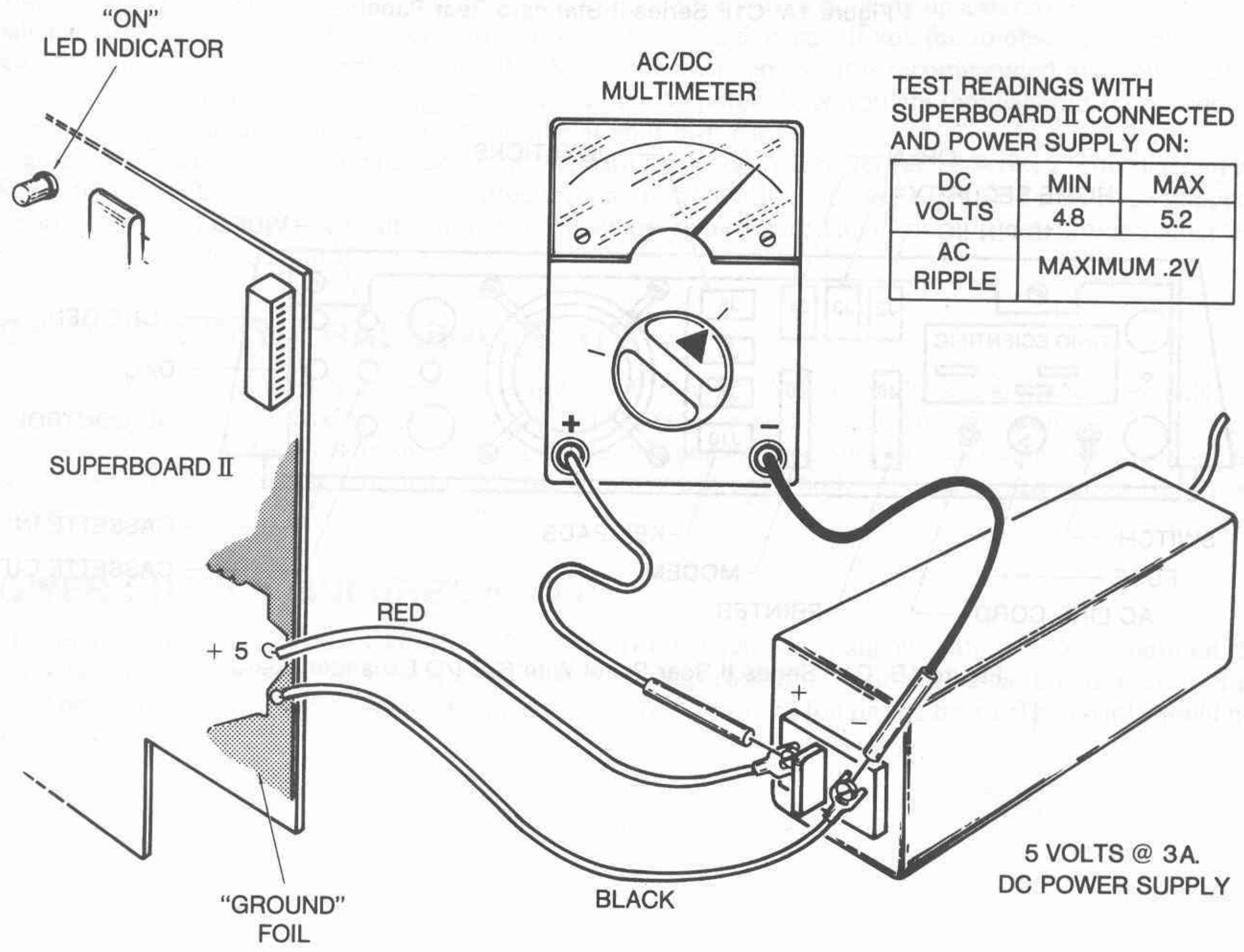


Figure 1B: C1P Series II Rear Panel With 630 I/O Expander Board

Figure 2 illustrates the following sequence of steps involved in connecting a Superboard II to a power supply:

1. With the power supply unplugged, connect the RED and BLACK wires from the Superboard to the + and − terminals of the power supply.
2. Attach an AC/DC multimeter to the terminal of the power supply and set the multimeter to a DC range which will accurately measure 5 volts (a range of Ø-6 volts or Ø-1Ø volts should be acceptable.)
3. Briefly turn on the power supply. The "ON" light (a red LED) should glow. If it does not, then turn off the power supply and check your connections to make sure they are not reversed.
4. Again turn on the power supply and measure the DC voltage. The reading should be between 4.8 and 5.2 volts. If the reading is less than 4.8 volts, then your power supply is probably inadequate. If the reading is greater than 5.2 volts, then it may damage your board. Turn off the power supply—immediately.
5. Without changing the connections, set the multimeter to measure an AC voltage of approximately Ø.5 volts (you will probably need to use the lowest AC range on the meter).
6. Turn the power supply on and measure the AC voltage. This reading measures ripple and it must not exceed Ø.2 volts AC.

When the power supply has been successfully attached to the Superboard II and tested, you are ready to proceed to Section four which discusses the interface of the computer with a video display.



TEST READINGS WITH SUPERBOARD II CONNECTED AND POWER SUPPLY ON:

	MIN	MAX
DC VOLTS	4.8	5.2
AC RIPPLE	MAXIMUM .2V	

Figure 2: Superboard II Power Supply Connections

SECTION 4

CONNECTING YOUR UNIT TO THE VIDEO DISPLAY

The first step in assembling your Challenger 1P computer system is the interface of your computer with a video display. This section describes two possible methods of making this connection.

Figure 3 illustrates these two methods of attaching a video display to the Challenger 1P or Superboard II. The top (or top left) RCA jack on the back of the C1P carries the video output signal from the C1P. This signal can be transmitted to either the high impedance (HI-Z) input of a closed circuit television video monitor (such as the Model AC-3 12" video monitor available from Ohio Scientific) or an RF modulator for display on a standard television. Three cables are provided with the Challenger 1P Series 2 for the audio and video connections. The Superboard II is supplied with a wiring harness which provides connections for video output and cassette input/output. This harness should be attached as indicated in Figure 3.

With a closed circuit video monitor such as the Model AC-3, use the cable supplied with the C1P to connect the video output jack on the back of the C1P directly to the video input jack on the back of the monitor. On monitors other than the AC-3, if there is a high impedance-low impedance selector switch or two or more inputs follow the monitor manufacturers instructions.

With a standard television, use the cable supplied with the C1P to connect the video output jack on the back of the C1P to the "video in" port of a video-to-RF interface modulator and follow the manufacturers instructions supplied with the modulator.

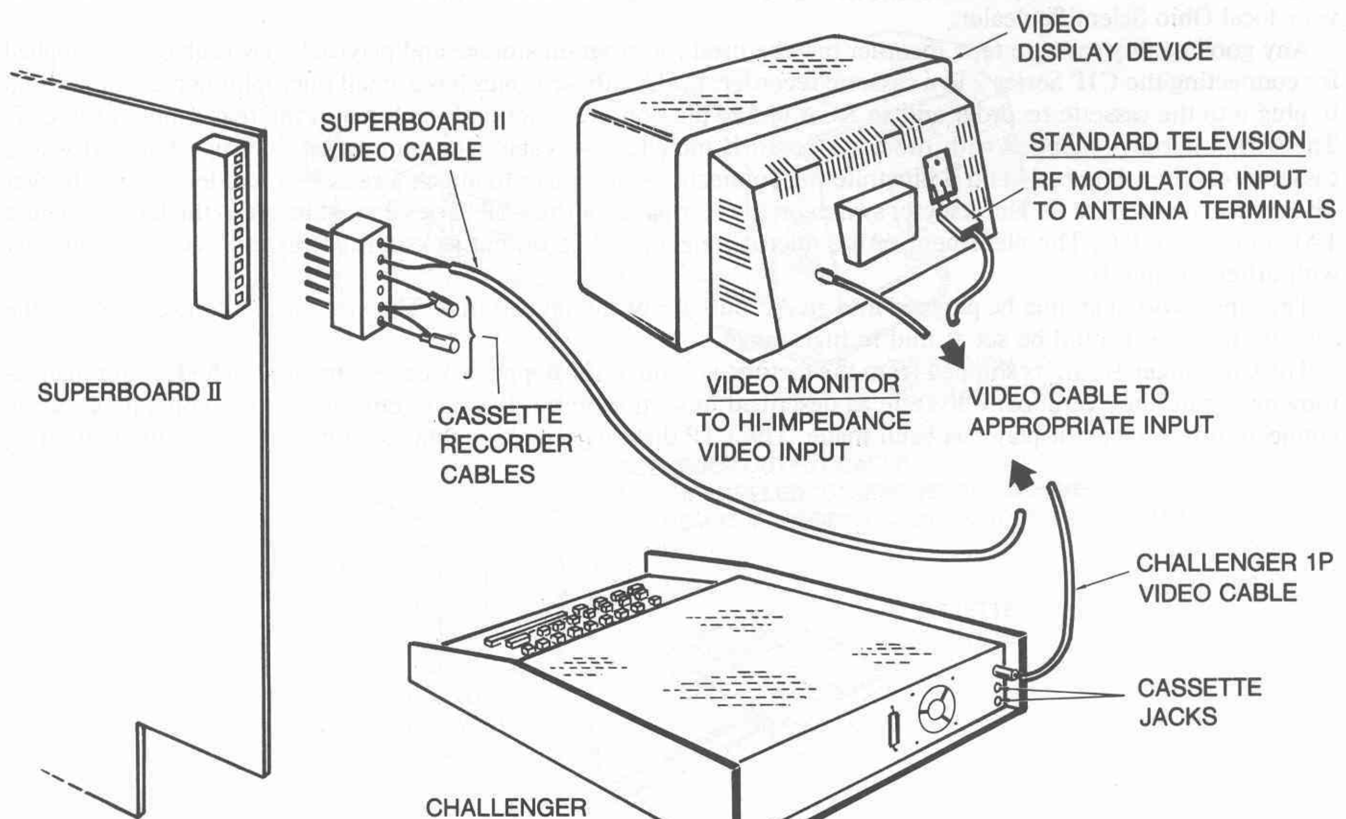

VIDEO DISPLAY DEVICE
SUPERBOARD II VIDEO CABLE
STANDARD TELEVISION
RF MODULATOR INPUT TO ANTENNA TERMINALS
SUPERBOARD II
CASSETTE RECORDER CABLES
VIDEO MONITOR
TO HI-IMPEDANCE VIDEO INPUT
VIDEO CABLE TO APPROPRIATE INPUT
CHALLENGER 1P VIDEO CABLE
CASSETTE JACKS
CHALLENGER 1P

Figure 3: Challenger 1P and Superboard II Video Connections

SECTION 5

CONNECTING THE UNIT'S FLOPPIES OR CASSETTE SYSTEM

All models of the Challenger 1P line of computers, including the Superboard II, include an audio cassette interface. This interface allows a standard audio cassette recorder to be used for program storage and playback. Although cassette I/O is not as convenient as disk I/O, it provides an inexpensive means of building a permanent library of programs. Moreover, a large library of applications software is available on cassette from Ohio Scientific through your local Ohio Scientific dealer.

Any good quality cassette tape recorder may be used for program storage and playback. Two cables are supplied for connecting the C1P Series 2 to a cassette recorder. Each of these cables has a small microphone plug on one end to plug into the cassette recorder and an RCA phono plug on the other end to plug into the rear panel of the C1P. The wiring harness supplied with the Superboard II includes two cables for connecting the Superboard II with a cassette recorder. Figures 4 and 5 illustrate the connections necessary to attach a cassette recorder to a Challenger 1P and to a Superboard II. The selector switch on the rear panel of the C1P Series 2 must be set to the left (see figure 1A) for cassette I/O. The placement of the microphone and audio output jacks on the cassette recorder may vary with different brands.

The tape recorder should be plugged into an AC outlet, not run on batteries. The volume and tone controls of the cassette recorder should be set at mid to high range.

The Challenger 1P MF is shipped from the factory with the mini-floppy disk drive already attached. Other than removing the dummy cardboard diskette as described in section three, these systems are ready to operate once the connection to a video display has been made. The C1P disk upgrade kit contains all necessary documentation.

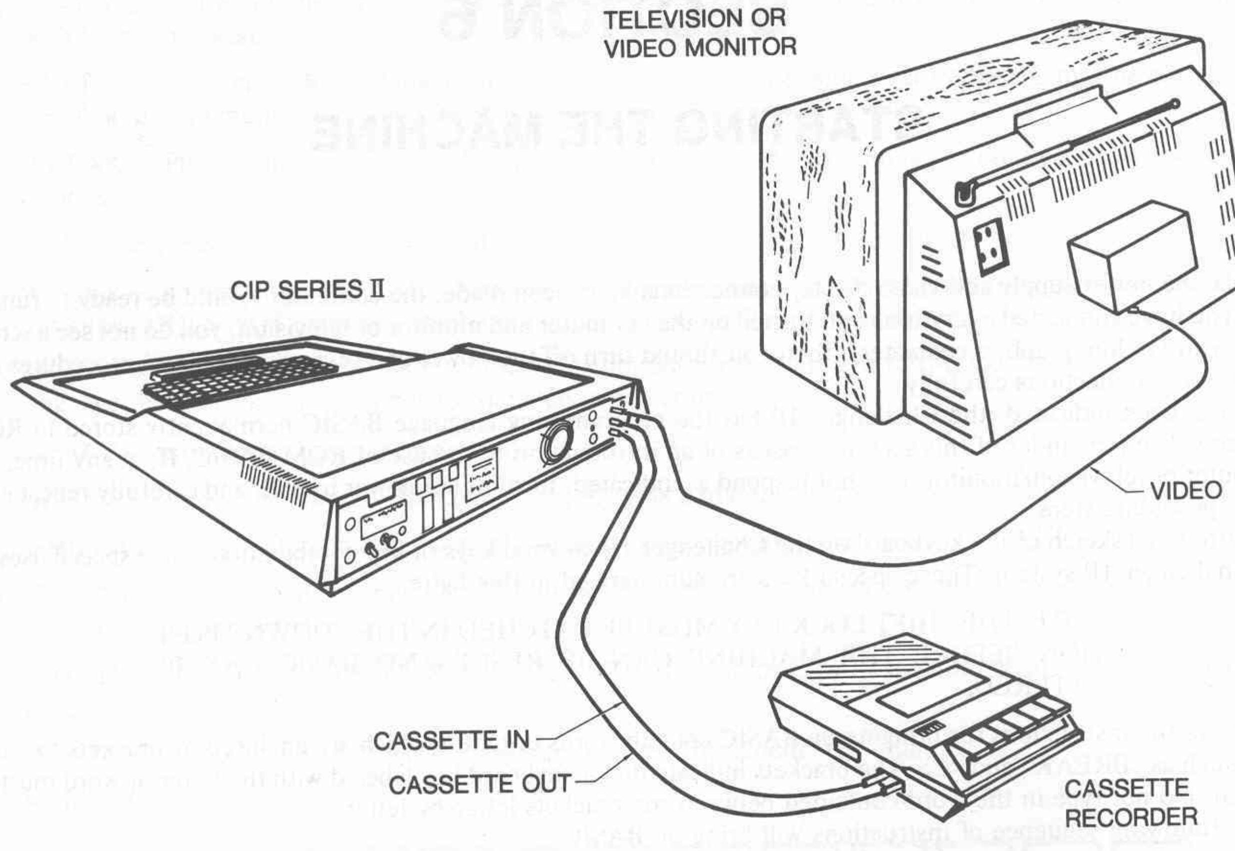


Figure 4: C1P Cassette Recorder Connections

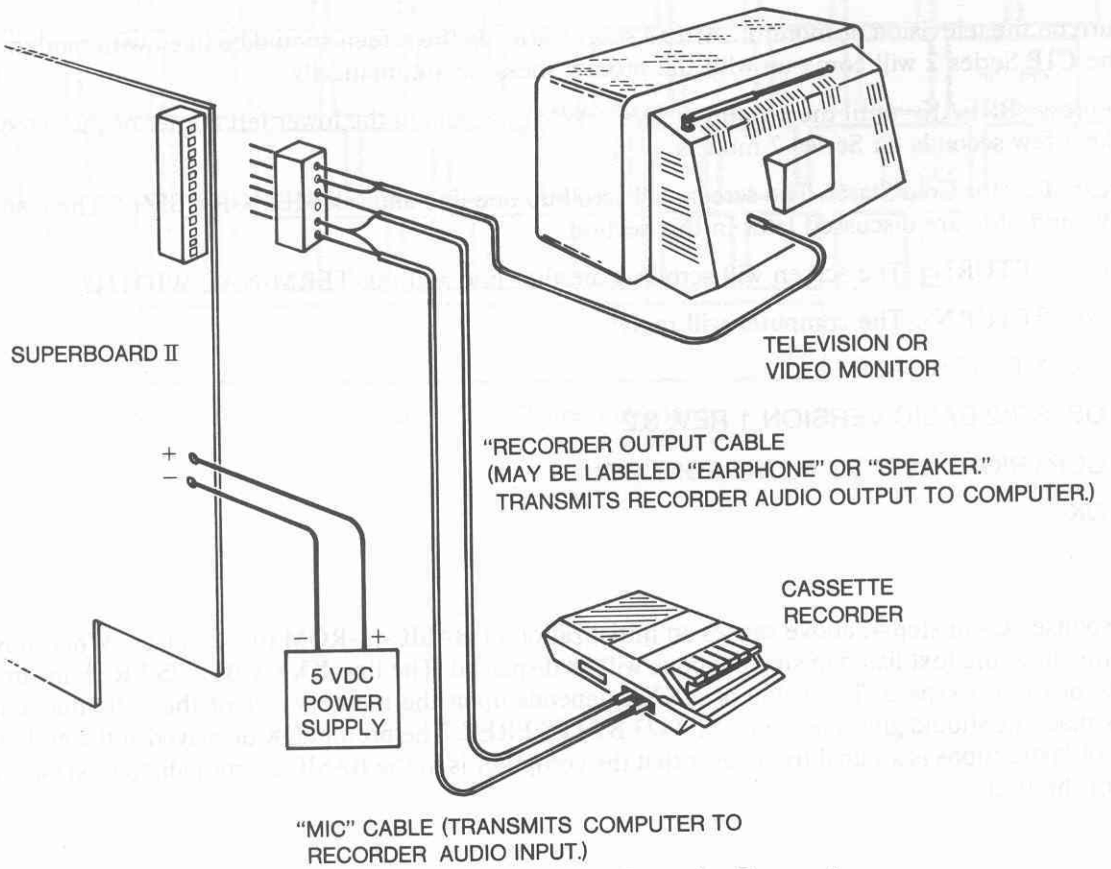

TELEVISION OR
VIDEO MONITOR
SUPERBOARD II
+
−
"RECORDER OUTPUT CABLE
(MAY BE LABELED "EARPHONE" OR "SPEAKER."
TRANSMITS RECORDER AUDIO OUTPUT TO COMPUTER.)
CASSETTE
RECORDER
−
+
5 VDC
POWER
SUPPLY
"MIC" CABLE (TRANSMITS COMPUTER TO
RECORDER AUDIO INPUT.)

Figure 5: Superboard II Cassette Recorder Connections

SECTION 6

STARTING THE MACHINE

Once the power supply and video display connections have been made, the computer should be ready to run. If, after you have connected everything and turned on the computer and monitor or television, you do not see a screen filled with random graphics characters, then you should turn off the power and review all hook-up procedures and check your connections carefully.

As has been indicated, the Challenger 1P has the programming language BASIC permanently stored in ROM memory. The remainder of this section consists of an introduction to the use of ROM BASIC. If, at any time, the computer or television/monitor does not respond as indicated, turn off the power to both and carefully repeat each of the preceding steps.

Figure 6 is a sketch of the keyboard on the Challenger 1P. Several keys or key combinations have special uses on the Challenger 1P system. These special keys are summarized in this figire.

> NOTE: THE SHIFT LOCK KEY MUST BE LATCHED IN THE "DOWN" POSITION BEFORE THE MACHINE CAN BE RESET AND BASIC CAN BE ENTERED.

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

The following sequence of instructions will bring up BASIC:

1. Turn on the computer.
2. Turn on the television or monitor. After a short warm-up the screen should be filled with random characters. The C1P Series 2 will come up with the prompt message automatically.
3. Depress <BREAK> until the prompt or D/C/W/M? appears in the lower left corner of the screen. This will take a few seconds on Series 2 models.
4. Press <C> (for Cold Start). The screen will scroll up one line and ask MEMORY SIZE? The responses <D>, <W> and <M> are discussed later in this section.
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 65Ø2 BASIC VERSION 1 REV. 3.2

COPYRIGHT 1977 BY MICROSOFT CO.

OK
```

The response <C> in step 4, above causes an initialization of BASIC-in-ROM to take place. When this initialization is completed, the text listed in step 6, above will be displayed. The line XXXX BYTES FREE informs the user of the size of the workspace. The value of XXXX depends upon the memory size of the individual computer. A typical 8K machine should give a response of 7423 BYTES FREE. The prompt OK displayed at the end of the above sequence of instructions is a signal to the user that the computer is in the BASIC or immediate mode and is awaiting input from the user.

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. Hold for several seconds.
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. See 65V Primer for detailed information.
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 from memory, last character typed.
10. <SHIFT P>—Press <SHIFT> first, add <P>—erases from memory, 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.
12. <SHIFT N>—Press <SHIFT> then <N>, yields Λ—used for exponential notation.
13. RUBOUT—is not used.

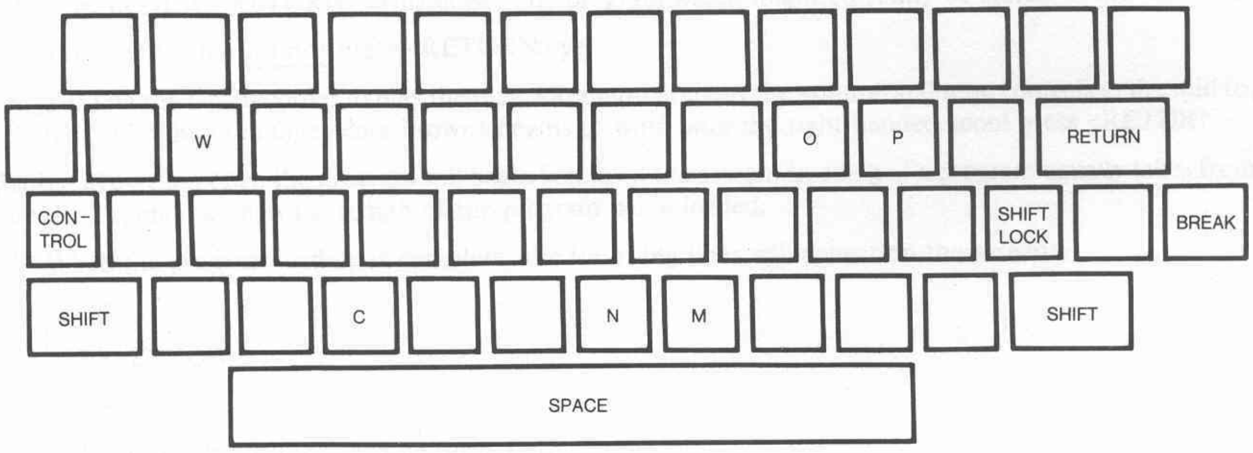


Figure 6: OSI Polled Keyboard

One of the automatic steps in the initialization procedure is the clearing of a region of memory designated as the workspace. This region is used by the computer to store programs written in BASIC.

Programs can be entered into the workspace in several different ways. The user can enter a BASIC program directly through the keyboard or from an external storage device such as a cassette tape. The method of entering a program into the workspace is strictly a matter of convenience. Once a program has been entered into the workspace by any means, the user can list the program, make corrections or additions, run the program or store the program on an external storage device if desired.

A program entered into the workspace remains there until it is removed. The command NEW can be used to clear the workspace, or erase the program, and allow the entry of a new program. If the power to the computer is turned off, then the program is lost. If the user depresses the <BREAK> key for a few seconds then the prompt or D/C/W/M? will be displayed on the screen again. If the user depresses <W> (for warm start) then the computer re-enters BASIC and the contents of workspace are retained. On the other hand, if the user depresses <C> then a cold start is performed and the workspace is cleared.

Section seven will describe how to LOAD and RUN "canned" programs (programs previously written and stored on an external storage device). Section eight will describe the entry of programs directly through the keyboard. Section eleven will describe the techniques involved in the storage of programs on cassette or diskette.

There are two possible responses in step 4) above which we have not discussed yet. These are <D> and <M>. The response <D> is used with mini-floppy disk based versions of the Challenger 1P, such as the C1P MF, to select the disk. Section eleven will discuss this option in detail. The option <M> allows the user to enter the Monitor. This feature allows the user to examine and modify the contents of memory. This capability is primarily used in machine code applications. The 65V Primer, available through your OSI dealer, is an introduction to machine code programming using the Monitor.

SECTION 7

RUNNING A "CANNED" PROGRAM

Ohio Scientific maintains an extensive library of software on both cassette and diskette to meet a wide variety of needs. With these packaged programs a user can make extensive use of the capabilities of the Challenger 1P without the need to know how to program. This section describes how to utilize these "canned" or "ready-to-run" programs.

LOADING CASSETTE PROGRAMS

The standard cassette based Challenger 1P and the Superboard II are supplied with a C1P Sampler cassette, which contains a selection of programs illustrating various capabilities of the Challenger 1P system. The following instructions describe how to load and run programs stored on cassette.

With the cassete recorder attached to the C1P as described in section five and the selector switch on the rear panel set to the left position follow the instructions given in section six to enter BASIC-in-ROM. The BASIC prompt OK should be displayed in the lower left corner of the screen. Place the cassette containing the program to be loaded in the recorder and go through the following sequence of instructions:

1. Rewind the cassette until the tape leader is visible.
2. Type in NEW <RETURN>. This erases any program which might currently be stored in the workspace.
3. Type LOAD but do not press <RETURN> yet.
4. Turn on the tape recorder to play the tape. (Remember to set the volume and tone controls at the mid to high ranges.) When the tape (dark brown) begins to wind onto the right-handed spool press <RETURN>.

Within a few moments, the program will begin listing on the screen. Loading of a program usually takes from 1 to 5 minutes depending upon the length of the program being loaded.

5. When the program loading is complete, the following lines will appear on the screen

```
OK
?S┘ ERROR
OK
```

and the cassette recorder can be turned off.

6. To complete the loading of the program press <SPACE> followed by <RETURN>.

The program is now stored in the workspace and can be executed by entering the command RUN or inspected by entering the command LIST.

The above instructions assume that the program to be loaded is the first program on the cassette tape. When more than one program is stored on a cassette, the tape should be advanced to a point just preceding the program to be loaded rather than being rewound. With the Sampler cassette, load the first program and do not rewind the cassette recorder. Once you have run the first program, the tape will be in place to LOAD and RUN the next program on the cassette. The following is a brief description of the programs on the Sampler cassette.

SIDE ONE:

Basic Math—An educational quiz program that gives addition, subtraction, multiplication and division problems.

Checking Account—This program helps you balance your checkbook. Just give the computer the initial balance and check amounts and let the computer do the work.

Trig Tutor—This program explains and diagrams three trigonometric functions—sine, cosine and tangent. The computer then tests your comprehension with a quiz.

Star Wars—An arcade-type computer game. You move the crosshairs around the screen trying to draw a bead on the target ship.

SIDE TWO:

Counter—This is a combination of an educational game and a cartoon for children learning to count from one to ten.

Presidents Quiz—This program asks you 2Ø historical questions about various presidents.

If your cassette recorder has a counter, it is recommended that you reset the counter at the beginning of the tape and make note of the start of each new program. The use of a cassette recorder for saving programs will be discussed in Section 11.

LOADING DISK PROGRAMS

The Challenger 1P MF Series 2 is a mini-floppy disk based version of the C1P. A large number of applications diskettes are available from Ohio Scientific through your dealer. Diskettes for the Challenger 1P should be labeled with the designation PICO DOS or 08-65D C1P. Many of these diskettes display a "menu" when describing the programs available on the diskette when they are loaded into the computer. In order to use these diskettes, first make sure there are no diskettes in the drive, then turn on the power to the computer, the video monitor and the floppy disk unit (in that order). Then depress the <BREAK> key until the prompt D/C/W/M? is displayed in the lower left corner of the screen. Again, verify that the Shift Lock key is down. Insert the diskette (label side up, label toward you) into the mini-floppy drive (the "A" drive if you have a dual disk drive system), carefully close the drive door, and press <D> (for disk).

If the disk inserted is labeled PICO DOS, then the following text will appear on the screen when <D> is depressed

```
PICO DOS V1.1

MEMORY SIZE? 8955

TERMINAL WIDTH?
```

For now just enter a <RETURN> in response to the query TERMINAL WIDTH? Each PICO DOS disk provides storage for eight programs. These programs can be loaded into workspace by typing the command

```
LOAD n <RETURN>
```

where n is the number (between 1 and 8) of the program you wish to load. The ROM BASIC cassette commands, LOAD and SAVE, still work without the numeric extension. When the program is loaded the prompt OK will be displayed on the screen. The program can then be executed by entering the command RUN or inspected by entering the command LIST.

If the disk inserted is labeled OS-65D C1P a menu will be displayed on the screen. For example, when the standard OS-65D development disk is loaded, the following text is displayed on the screen

```
BASIC EXECUTIVE FOR

OS-65D V3.N

MO,DAY,YR RELEASE

FUNCTIONS AVAILABLE:

  CHANGE-ALTER WORK-SPACE LIMITS

  DIR-PRINTS DIRECTORY

  UNLOCK-UNLOCKS SYSTEM FOR END USER MODIFICATIONS
FUNCTION?
```

This menu offers us three choices. We can enter the response CHANGE and the computer will automatically LOAD and RUN a program by the name of CHANGE. If we enter the response DIR, then the computer will LOAD and RUN a program named DIR. If we respond UNLOCK, then the system is unlocked. This allows the user to assume control of the system with the capability to enter new programs and list programs in the workspace. The response UNLOCK places the system in the BASIC immediate mode with the display of the prompt OK.

For now, we will focus on the program DIR. This program prints a directory of the files present on the diskette. If we respond DIR to the query FUNCTION? then the computer will ask us

```
LIST ON LINEPRINTER INSTEAD OF DEVICE # 2 ?
```

Responding NO will cause the following output to appear on the screen.

```
              OS-65D VERSION 3. N
                 —DIRECTORY—
        FILE NAME          TRACK RANGE
        OS-65D3                 Ø-12
        BEXEC*                 14-14
        CHANGE                 15-16
        CREATE                 17-19
        DELETE                 2Ø-2Ø
        DIR                    21-21
        DIRSRT                 22-22
        RANLST                 23-24
        RENAME                 25-25
        SECDIR                 26-26
        SEQLST                 27-28
        TRACE                  29-29
        ZERO                   3Ø-31
        ASAMPL                 32-32
           5Ø ENTRIES FREE OUT OF 64
```

Some of the contents of this directory listing will be discussed in detail in section eleven. The files listed contain utility programs written in BASIC. Note that we were introduced to two of these programs, CHANGE and DIR, in the menu. In addition to listing the names of the programs on the diskette, the directory tells where they are loated on the diskette. For example, the program DIR is located on track 21 and is one track long while CHANGE is a 2 track program starting on track 15. (Each diskette has 4Ø tracks, numbered Ø through 39.)

Any of the BASIC programs on this disk can be run by responding UNLOCK to the query FUNCTION? and then entering the command ''RUN followed by either the name of the program or the number of the first track where it is stored. For example, either of the commands RUN''DIR'' or RUN''21'' would run the program DIR. The closing quotes are optional, ie, RUN''DIR.

Most of the applications diskettes do not offer the user the option of unlocking the system. On these diskettes programs are run by entering the appropriate response when the menu is displayed.

The use of mini-floppy diskettes for storing programs will be discussed in detail in section eleven.

SECTION 8
RUNNING BASIC

There are a large number of publications available which give detailed descriptions of the commands and functions of BASIC. While the material presented in this manual can in no way duplicate such excellent manuals as Basic and the Personal Computer by Dwyer and Critchfield or the Ohio Scientific BASIC Reference Manual, it at least gives some insight into the fundamentals of the BASIC language.

In order to enter and run the programs listed in this section the computer should be placed in the BASIC mode with a cold start as described in section six. Recall that the Shift Lock key must be latched in the down position before the machine can be reset and BASIC can be entered. The BASIC prompt OK will signify that the computer is prepared for input.

A program is a series of instructions to the computer. These instructions are stored within the memory of the computer and describe a procedure for accomplishing a specific task. Every statement in a BASIC program begins with a line number which the computer uses to sequence the statements in the program. These line numbers make it easy to modify or EDIT a program. For example, a statement can be deleted or erased by typing in its line number following immediately by <RETURN>. To insert a statement just assign it a line number which will place it in the desired location in the program. Any statement can be corrected by retyping the entire line, including the line number. An optional editor is available on disk for the C1P which simplifies these editing procedures.

There are two standard techniques for correcting mistakes that occur as you enter a BASIC program.

1. As indicated in FIGURE 6 in section 6, typing a <SHIFT O> key combination deletes the last character typed. (The <SHIFT O> notation denotes pressing the <SHIFT> and the <O> simultaneously.) Multiple deletions can be made by repeating the <SHIFT O> combination. On the Challenger 1P the character is not actually removed from the input line, but an underscore character is printed for each character deleted.
2. As indicated in FIGURE 6, typing a <SHIFT P> key combination erases the line currently being typed. A "@" character is printed at the end of the line eliminated.

The Challenger 1P is ready to accept input once the computer replies OK as indicated above. Before entering any program, it is good programming practice to first type in NEW <RETURN>. Do this and then enter the following program exactly as it appears, including all punctuation.

PROGRAM ONE

The following programming example demonstrates three types of statements in the BASIC language. Statement 1Ø is a REM statement which is used to include remarks in a BASIC program. Any text included after the keyword REM is considered to be a remark and does not affect the execution of the program. Statements 2Ø-12Ø are PRINT statements. A PRINT statement causes output to be displayed on the next line of the screen. Lines 2Ø, 3Ø, 5Ø and 6Ø just cause a blank line to appear on the screen. In lines 4Ø, 7Ø-1ØØ and 12Ø the actual text enclosed within quotation marks is displayed on the screen. The meaning of line 13Ø is obvious, it ends the program. Each of the following BASIC statements are followed by <RETURN>. This notation symbolizes to the user that the <RETURN> key is to be pressed. The <RETURN> will not be included in future program listings but must be included at the end of each line entered into the computer. The statements in this program are numbered by multiples of 1Ø. This type of numbering routine simplifies the addition of statements at a later time.

```
1Ø REM-PROGRAM #1 <RETURN>

2Ø PRINT <RETURN>

3Ø PRINT <RETURN>

4Ø PRINT "*** HELLO ***" <RETURN>
```

```
50 PRINT <RETURN>

60 PRINT <RETURN>

70 PRINT "========================" <RETURN>

80 PRINT "THIS PROGRAM USES THE" <RETURN>

90 PRINT "BASIC PRINT STATEMENT" <RETURN>

100 PRINT "TO DISPLAY TEXT ON" <RETURN>

110 PRINT "THE SCREEN" <RETURN>

120 PRINT "========================" <RETURN>

130 END <RETURN>
```

The command LIST can be used to instruct the computer to print out the program on the screen as it is currently stored within the computer's memory.

After you have listed your program and made any necessary corrections, the command RUN will instruct the computer to execute your program. If the computer detects any errors in your program it will respond with a message such as

```
?S ERROR IN 10
```

which would indicate an error in statement number 10.

Appendix 7 contains a complete list of error codes. After correcting the indicated error, you can RUN the program again. The Ohio Scientific Basic Reference Manual also contains a list of the BASIC error displays. The program will remain in the workspace until you enter the command NEW or turn off the computer. Once the program is correctly entered and executing properly, you should experiment with the program by adding, deleting, or modifying statements to gain experience with the capabilities of the system. For example you might make the computer say hello to you by including your name in quotes in one of the PRINT statements.

PROGRAM TWO

One of the key features of a programming language such as BASIC is the use of variables. Through the use of variables, such as X, Y, S and A in the following sample program, the computer assigns names to certain locations or addresses in memory. The contents of these memory locations can then be easily modified by a variety of BASIC statements. The following simple BASIC program illustrates the use of the INPUT statement (statement 90) and the assignment statement (statements 100 and 110).

```
10 REM—PROGRAM #2

20 REM—PROGRAM READS

30 REM—TWO NUMBERS

40 REM—CALLED X AND Y

50 REM—THEN PRINTS THE

60 REM—NUMBERS AND

70 REM—THEIR AVERAGE

80 PRINT "ENTER X AND Y"

90 INPUT X, Y

100 LET S = X+Y

120 PRINT "X = "; X
```

```
130 PRINT "Y = "; Y
140 PRINT "AVERAGE = "; A
150 END
```

Statements 10-70 in this program are remarks (REM statements) and can be deleted without affecting the operation at the discretion of the user. It is generally considered good programming practice to include brief comments such as this to document the purpose of a program.

When this program is run, statement 80 will cause the message

```
ENTER X AND Y
```

to be displayed on the screen. Statement 90 will then cause a ? to be printed on the next line. This serves as a prompt which indicates that the computer is expecting input from the keyboard. Technically, statement 80 is unnecessary and could be deleted, but it is included to remind you what the computer expects you to enter. The computer has set aside locations named X and Y in memory to receive the values you enter (see below). You should type in two values separated by a comma, say for example 8, 19 and press <RETURN>. The first value is deposited in the location named X and the second value is deposited in the location named Y.

The next statement executed is statement 100. The expression on the right hand side of the equal sign, X+Y, is evaluated using the current values stored in X and Y and the result is stored in or assigned to, the location named S. Statement 110 then calculates S/2, wich means S divided by 2, and stores the result in the location named A. With the values above, S will contain 27 and A contain 13.5. Statements 120-140 illustrate a slightly different form of the PRINT that was used in Program One. Statement 120 will cause the output

```
X = 8
```

to appear on the screen. As pointed out in Program One, anything enclosed within quotation marks is displayed on the screen. On the other hand, when a variable name, such as X, is listed in a PRINT statement and is not enclosed in quotes, the computer prints the <u>contents</u> of the location X and not the letter X.

The following remarks describe several special features of BASIC on the Challenger 1P.

1. The keyword LET is optional in an assignment statement. For example, statement 100 could be replaced by
   ```
   100 S =X+Y
   ```
2. Statements 80 and 90 could be combined into the single statement
   ```
   90 INPUT "ENTER X AND Y"; X, Y
   ```
3. Statements 90, 110 and 140 could be replaced by
   ```
   140 PRINT "AVERAGE = "; (X+Y)/2
   ```
   An expression appearing in a PRINT statement is evaluated and the result is printed.
4. The symbol "?" can be used as a short-hand abbreviation for PRINT. Thus statement 120 can be replaced by
   ```
   120 ? "X = "; X
   ```
5. More than one BASIC statement can be positioned on one line with the use of the ":" symbol between the separate statements.

PROGRAM THREE

Normally each statement in a BASIC program is executed in sequential order. The BASIC language provides several ways of modifying the normal order of execution. If the statement

```
145 GO TO 80
```

is added to Program Two, then we have constructed a logical loop within our program. The program will now compute the averages of pairs of numbers indefinitely. Statement 145 provides an unconditional branch. Each time the execution of the program reaches statement 145, control will be looped back to statement 80. This new version of Program Two has one major problem—there is no convenient way to terminate execution. Because of the possibility of forming endless loops of this type in a BASIC program, the Challenger 1P provides two methods of interrupting a program (short of pulling the plug.) First, any program can be terminated by holding the <BREAK> key down for several seconds. This resets the computer and displays the D/C/W/M? prompt on the screen. Another way to interrupt the execution of a BASIC program is to press <CONTROL C>. When this is entered, the computer terminates execution and prints the message

```
BREAK IN LINE XXXX
```

The user can then list and modify his program as desired.

Rather than force the user to resort to entering <BREAK> or <CONTROL C> to terminate the execution of our new version of Program Two, we can replace statement 145 by

```
145 IF A<>Ø THEN GO TO 8Ø
```

This statement checks the condition A<>Ø (A not equal to zero). As long as the average is not zero, control is transferred back to statement 8Ø and another pair of numbers is processed. The user can now terminate the execution of the program by entering any two numbers X and Y whose average is Ø. This type of a conditional branch is an extremely useful feature of BASIC.

Refer to Ohio Scientific's BASIC Reference Manual and BASIC and The Personal Computer (available from your local OSI dealer) or any other BASIC language text to continue developing your programming skills.

SECTION 9

GRAPHICS

The Challenger 1P features the same set of 256 graphics characters offered on the more expensive C4P and C8P series of computers. A complete list of these characters may be found in Appendix 9. The normal display mode for the C1P is 24 rows × 24 columns in black and white. The Series 2 provides an alternate 12 row × 48 column text display mode. With the 63Ø I/O expander board the C1P Series 2 user can display any of the 256 graphics characters in up to 16 colors on a standard color television set or color monitor. The color option is discussed in section nineteen where several special features of the 63Ø I/O expander board are described in detail.

For display purposes the screen is divided into a grid of rectangular blocks. Each of these blocks is associated with a specific address in memory. The display within each cell of the grid is determined by the numeric content of the memory location associated with the cell.

Figures 7 and 8 illustrate the video memory maps for both the standard 24 × 24 video display and the alternate 12 × 48 video display on the Challenger 1P. These memory maps indicate the memory address of each cell on the screen. In the standard 24 × 24 display mode, for example, the cell in the upper left hand corner of the screen has address 53381 while the cell in the lower right hand corner of the screen has address 54141. The memory maps actually give the address of each cell in two different number systems—decimal and hexadecimal. The use of the hexadecimal number system for addressing on the Challenger 1P is discussed in Appendix 2. Although your display may have 1 or 2 additional visible lines at the top and/or bottom of the screen due to variations between video monitors, it is recommended that graphic displays be restricted to the regions of the screen prescribed by the video maps.

HEX	DEC.	DEC.	HEX
$DØ85	53381	534Ø4	DØ9C
DØA5	53413	53436	DØ6C
DØC5	53445	53468	DØDC
DØE5	53477	535ØØ	DØFC
D1Ø5	53509	53532	D11C
D125	53541	53564	D13C
D145	53573	53596	D15C
D165	535Ø5	53628	D17C
D185	53637	5366Ø	D19C
D1A5	53669	53692	D1BC
D1C5	537Ø1	53724	D1DC
D1E5	53733	53756	D1FC
D2Ø5	53765	53788	D21C
D225	53797	5382Ø	D23C
D245	53829	53852	D25C
D265	53861	53884	D27C
D285	53893	53916	D29C
D2A5	53925	53948	D2BC
D2C5	53957	5398Ø	D2DC
D2E5	53989	54Ø12	D2FC
D3Ø5	54Ø21	54Ø44	D31C
D325	54Ø53	54Ø76	D33C
D345	54Ø85	541Ø8	D35C
D365	54117	5414Ø	D37C

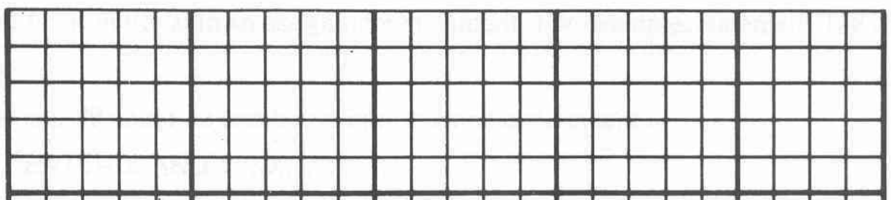


Figure 7: Video Memory Map (24 × 24 Format)

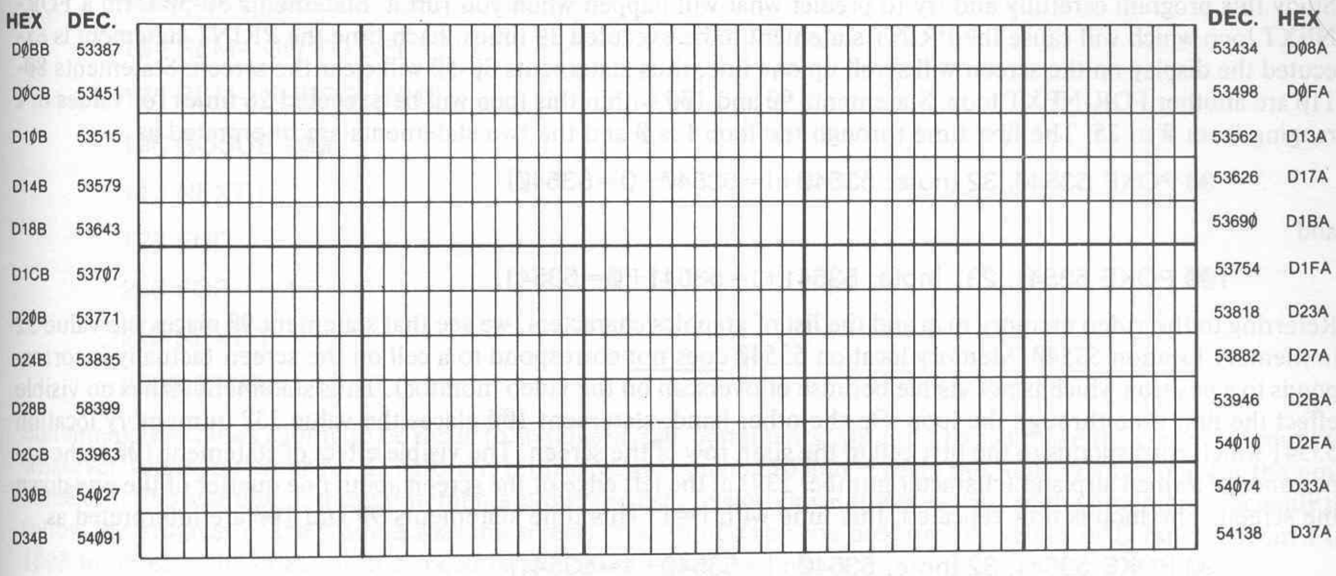


Figure 8: Video Memory Map (12 × 48 Format)

Appendix 9 shows the diagrams and numeric codes for each of the 256 graphics characters available on the Challenger 1P. The Challenger 1P uses the BASIC statement POKE to display a character at a specified location on the screen.

The BASIC statement POKE is an extremely useful statement. It can be used to store any numeric value (in the range Ø-255) at any address in RAM memory. THE POKE STATEMENT MUST BE USED WITH CAUTION. The ability it gives the user to modify the contents of memory can lead to disastrous effects if POKEs are made to random areas of memory. Since the memory associated with the screen is RAM memory, the POKE statement allows us to place the numeric value of any figure we wish to display on the screen in the memory location associated with the cell in which we wish to display the figure. The syntax of the POKE statement is as follows

```
POKE address, value
```

Follow the cold start procedure described in section six to enter BASIC-in-ROM. When the BASIC prompt OK appears, hold down the <RETURN> key until the screen is cleared and then directly enter the command POKE 53776, 239. A small airplane will be displayed in approximately the center of the screen.

Now enter the following sample program.

```
 1Ø REM—GRAPHICS DEMO
 2Ø REM—CLEAR THE SCREEN
 3Ø FOR J=1 TO 3Ø
 4Ø PRINT
 5Ø NEXT J
 6Ø REM—MOVE FIGURE ACROSS
 7Ø REM—THE SCREEN
 8Ø FOR I=Ø TO 25
 9Ø POKE 5354Ø+I, 32
1ØØ POKE 53541+I, 237
11Ø NEXT I
12Ø END
```

Study this program carefully and try to predict what will happen when you run it. Statements 3Ø-5Ø form a FOR-NEXT loop which will cause the PRINT statement to be executed 3Ø times. Each time the PRINT statement is executed the display on the screen will scroll up one line, thus statements 3Ø-5Ø will clear the screen. Statements 8Ø-11Ø are another FOR-NEXT loop. Statements 9Ø and 1ØØ within this loop will be executed 26 times for values of I ranging from Ø to 25. The first time through the loop I is Ø and the two statements are interpreted as

```
9Ø POKE 5354Ø, 32 [note: 5354Ø+I=5354Ø+Ø=5354Ø]
```

and

```
1ØØ POKE 53541, 237 [note: 53541+I=53541+Ø=53541]
```

Referring to the video memory map and the list of graphics characters, we see that statement 9Ø places the value 32 in memory location 5354Ø. Memory location 5354Ø does not correspond to a cell on the screen (actually it corresponds to a position which is not visible because of overscan on the video monitor). Thus statement 9Ø has no visible effect the first time through the loop. On the other hand, statement 1ØØ places the value 237 in memory location 53541 which corresponds to the first cell in the sixth row of the screen. The visible effect of statement 1ØØ is the appearance of a small airplane (character number 237) at the left edge of the screen about one quarter of the way down the screen. The loop is now repeated, this time with I=1. This time statements 9Ø and 1ØØ are interpreted as

```
9Ø POKE 53541, 32 [note: 5354Ø+I=5354Ø+1=53541]
```

and

```
1ØØ POKE 52542, 237 [note: 53541+I=53541+1=53542]
```

This time the effect of statement 9Ø is to place a blank in the first cell in the sixth row (thereby erasing the airplane placed there the first time through the loop) and statement 1ØØ then redraws the airplane in the second cell of the sixth row. As the loop is repeated for subsequent values of I ranging from 2 to 25 the airplane is moved cell-by-cell across the screen. Due to the speed of the microprocessor, the program executes so quickly that it is difficult, if not impossible, to distinguish each step as the plane moves across the screen. This difficulty can be remedied by adding the following lines of code to your program (remember that BASIC will use the line numbers to automatically insert these statements in their appropriate location within the program).

```
25 INPUT "ENTER DELAY"; D
1Ø4 REM—GO TO DELAY
1Ø5 REM—SUBROUTINE
1Ø6 GOSUB 2ØØ
2ØØ FOR T=1 TO D
21Ø NEXT T
22Ø RETURN
```

A complete listing of the modified version of the program is now

```
1Ø REM—GRAPHICS DEMO
2Ø REM—CLEAR THE SCREEN
25 INPUT "ENTER DELAY"; D
3Ø FOR J=1 TO 3Ø
4Ø PRINT
5Ø NEXT J
6Ø REM—MOVE FIGURES ACROSS
7Ø REM—THE SCREEN
8Ø FOR I=Ø TO 25
9Ø POKE 5354Ø+I, 32
```

```
100 POKE 53541+I, 237
104 REM—GO TO DELAY
105 REM—SUBROUTINE
106 GOSUB 200
110 NEXT I
120 END
200 FOR T=1 TO D
210 NEXT T
220 RETURN
```

Statement 106 causes a jump to be made to statement 200. Statements 200 and 210 just make the computer count to whatever value you enter for D before it returns to statement 110 and repeats the FOR-NEXT loop for the next value of I. The addition of these statements allows the user to slow down the execution of the program sufficiently to follow the progress of the plane across the screen. You should run the program for values of D ranging from 1 to 1000 to observe the effect on the speed of execution.

This program illustrates the general concepts involved in displaying graphics characters on the screen. An important fact to remember in moving a figure on the screen is that the figure must be erased from its old location as well as redrawn at its new location. Many of the graphics characters are designed to be used in pairs or groups to produce larger figures. Displaying characters 9 and 10 in adjacent horizontal cells displays a space ship. Characters 179-182 can be combined to display ships. Characters 229-232 depict the four playing card suits—hearts, clubs, spades and diamonds.

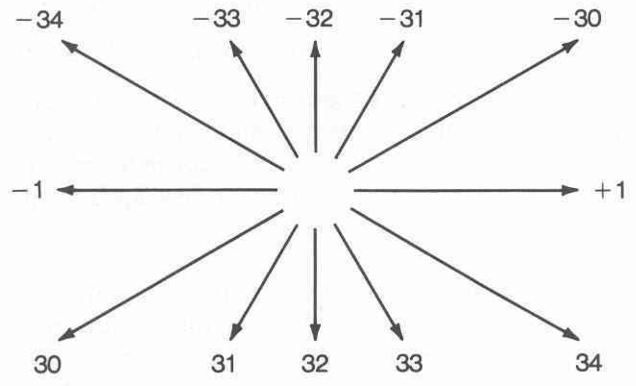


Figure 9: FOR-NEXT Loop Directional Increments

Figure 9, shown above, shows the values to increment a screen location to produce movement in the associated direction. The twelve directions shown in Figure 9 are demonstrated in the following program.

```
10 FOR SC=1 TO 25: PRINT: NEXT
20 X=53711: Y=161: POKEX, Y
30 FOR R=1 TO 12: READ D
40 FORI=1TO10: POKEX+D, Y: X=X+D: NEXT
45 X=53711
50 NEXT R
60 DATA -32, -31, -30,1, 34, 33
70 DATA 32, 31, 30, -1, -34, -33
80 GOTO80: REM PREVENT SCROLL
90 REM PRESS <CTRL C> TO END
```

In section twelve the graphics capabilities of the Challenger 1P will be further illustrated by a program which allows the user to control the movement of a figure on the screen by depressing various keys on the keyboard.

The Series 2 models of the Challenger 1P offer an alternate 12 row × 48 column display mode. This display mode provides 12 lines of text (with intervening blank lines) of 48 characters each. The 12 × 48 mode is primarily intended for the display of text (the intervening blank lines are not compatible with most graphics displays).

In order to use the 12 × 48 display mode, the standard software which controls the video display must be modified. This is accomplished by running a special program which swaps a new video driver for the old. This program named SWAP is supplied in an autorun form on cassette with the standard C1P and on diskette with the C1P MF. The screen size option is controlled by bit Ø of the control register at 55296 (address D8ØØ in hexadecimal). For example, once the program SWAP has been run

```
POKE 55296, Ø selects the 24 × 24 display mode

POKE 55296, 1 selects the 12 × 48 display mode.
```

(Appendix 5 gives a complete listing of the values to poke at 55296 to obtain various combinations of options such as screen width and DAC sound.)

On standard cassette versions of the Challenger 1P, memory location 251 is used in place of 55296 to control screen width, DAC sound and the color option, if the program SWAP has been run. Thus, on the cassette based version of the C1P, the screen size is selected as follows

```
POKE 251, Ø selects the 24 × 24 display mode

POKE 251, 1 selects the 12 × 48 display mode.
```

SECTION 10

SOUND

All Series 2 models of the Challenger 1P have a built-in 8 bit digital to analog converter (DAC) which is capable of generating sound output. The signal from the DAC is available at the DAC output port on the rear panel of the C1P (see Figure 1). The signal from this output port can be fed into the auxiliary input of an audio amplifier or the audio input jack on the rear of a video monitor. Software is available from Ohio Scientific through your local dealer for the Challenger 1P MF Series 2 which allows the user to enter, play and store songs with multiple parts.

The programming techniques required to generate sound through the DAC are relatively sophisticated. The output at the DAC output port must be updated at least 5ØØ-1ØØØ times per second even for the simplest tones. A high level language such as BASIC does not provide sufficiently fast execution speed to be suitable for such applications. Routines to generate musical tones must be written in assembler or machine code (the "native language" of the microprocessor) to attain the execution speed necessary.

Memory location 55296 (address D8ØØ in hexadecimal) is reserved as a control register on the Challenger 1P. Just as the user can use this register to choose between the 24 × 24 and the 12 × 48 display mode, he can also enable (turn on) or disable (mute) the output from the DAC with different POKEs to the location 55296. For the purposes of this section we will restrict our attention to the following two possibilities:

```
POKE 55296, Ø mutes the DAC output

POKE 55296, 16 turns on the DAC output
```

These two POKEs both select the standard 24 × 24 display mode. (Appendix 5 gives a complete listing of the values to POKE at 55296 to obtain various combinations of options such as DAC sound and screen format.)

Random output from the DAC can be heard quite easily by the following method. Turn on the computer and do a cold start. When the OK prompt is displayed, type in

```
POKE 55296, 16
```

(with no line number) and depress one or two keys in the top row of the keyboard. If the DAC is hooked up correctly and the volume is turned up, you should hear various high pitch tones. These tones are being generated since the DAC and the keyboard share the same I/O (Input/Output) port at 57Ø88 (address DFØØ in hexadecimal). These tones can be turned off by entering

```
POKE 55296, Ø
```

which will mute the output to the DAC.

Programming to produce sound output with the DAC generally involves the following simple scheme:

1. Turn on the DAC.
2. Send a constantly varying sequence of values to the DAC output port.
3. Turn off the DAC.

The following sample program alternately stores the values Ø and 255 at address 57Ø88 (the DAC output port). In the beginning the alternating values generate a square wave with a relatively low frequency of approximately 75 cycles per second. As execution proceeds, the frequency increases until it reaches a relatively high frequency of approximately 12ØØØ cycles per second. The audible effect is similar to a slide whistle sliding from a low note to a high note. This routine is written for ROM BASIC.

```
1Ø REM—TURN ON DAC
2Ø POKE 55296, 16
3Ø REM—READ MACHINE CODE
```

```
 4Ø REM—ROUTINE AND STORE
 5Ø REM—BEGINNING AT 3Ø72
 6Ø FOR I=1 TO 38
 7Ø READ V
 8Ø POKE 3Ø71+I, V
 9Ø NEXT
1ØØ REM—READ MACHINE CODE
11Ø REM—SUBROUTINE AND
12Ø REM—STORE BEGINNING AT
13Ø REM—3328
14Ø FOR I=1 TO 12
15Ø READ V
16Ø POKE 3327+I, V
17Ø NEXT
18Ø REM—STORE STARTING
19Ø REM—HEX ADDRESS
2ØØ REM—OF MACHINE CODE
21Ø REM—ROUTINE FOR USR (X)
22Ø POKE 11,Ø : POKE 12,12
23Ø REM—JUMP TO MACHINE CODE
24Ø Y=USR(X)
25Ø REM—TURN OFF DAC
26Ø POKE 55296, Ø
27Ø END
28Ø DATA 169, 8, 141, 12, 13
29Ø DATA 169, Ø, 141, Ø, 223
3ØØ DATA 32, Ø13, 169, 255
31Ø DATA 141, Ø, 223, 32, Ø, 13
32Ø DATA 2Ø6, 12, 13, 2Ø8, 235
33Ø DATA 2Ø6, 13, 13, 173, 13, 13
34Ø DATA 141, 14, 13, 2Ø8, 219
35Ø DATA 96
36Ø DATA 174, 14, 13, 16Ø, 4
37Ø DATA 136, 2Ø8, 253, 2Ø2
38Ø DATA 2Ø8, 248, 96
```

The data statements at the end of this BASIC program comprise a sequence of instructions for the microprocessor written in the "native language" of the 65Ø2 microprocessor. Notice that we are able to turn the DAC on and off within the BASIC program, but we have to resort to machine code to obtain the speed of execution necessary to generate sound. The USR(X) function referenced in statement 24Ø provides a convenient means of interfacing machine code routines with BASIC programs. This feature is discussed in Ohio Scientific's BASIC Reference Manual. For a detailed description of Assembler Programming on the 65Ø2 see the MOS Programming Manual by MOS Technology, Inc. and Ohio Scientific Assembler/Editor and Extended Monitor Manual.

The 63Ø I/O Expander Board provides an alternative means of generating sound with a programmable tone generator. This feature is discussed in section nineteen where the several features of the 63Ø I/O Expander Board are discussed in detail.

SECTION 11

EXTERNAL STORAGE OF PROGRAMS

All models of the Challenger 1P line of computers, including the Superboard II, include an audio cassette interface. This interface allows a standard audio cassette recorder to be used for program storage and playback. Although cassette I/O is not as convenient as disk I/O, it provides an inexpensive means of building a permanent library of programs. Moreover, a large library of applications software is available on cassette from Ohio Scientific through your local Ohio Scientific dealer.

CASSETTE STORAGE

In section seven the user learned how to attach a cassette recorder to the Challenger 1P and was introduced to the procedure for loading and running prerecorded or "canned" programs. This section describes the use of both cassettes and diskettes for saving programs.

The following instructions describe how to record a program onto a cassette tape. These instructions can be used to record any BASIC program contained in the workspace whether the program was entered line-by-line through the keyboard or was itself initially loaded from cassette. Recall that the selector switch on the rear panel of the C1P must be set to the left (cassette) postion in order to do SAVEs and LOADs with cassettes.

These instructions can, for example, be used to create a backup of the Sampler tape provided with your cassette based Challenger 1P by loading each program from the Sampler tape and then recording it onto a blank tape.

It is recommended that you use new or thoroughly erased cassettes of good quality for recording programs to avoid noise and other problems associated with old cassettes.

When your program is in the form you wish to save, place a cassette in the recorder and rewind the cassette so that the tape leader is visible on the right-hand spool (or to the point at which you wish to store the program if you are storing more than one program on a cassette). The following sequence of instructons will then store the program on the cassette.

1. Type SAVE <RETURN>.
2. Type NULL8 <RETURN>.
3. Type LIST but do not press <RETURN> yet.
4. Now turn on the tape recorder in the RECORD mode. When the tape (dark brown) begins to wind onto the right-hand spool, wait 5 seconds and press <RETURN>.

The program will begin listing on the screen and to the cassette port. When the last line of the program is listed, wait a few seconds and turn off the recorder. To reset the computer to keyboard input

5. Type in LOAD <RETURN>.
6. Press <SPACE> followed by <RETURN>.

Each cassette should be labeled to identify the contents. If you wish to protect the contents from accidental erasure, break out the appropriate "record protect" tab from the rear edge of the cassette. The sample programs in Section Nine and Ten can be used to practice saving and loading programs.

Programs stored on cassette using the above procedure can be loaded using the technique described in section seven. This procedure can be modified slightly to store programs on cassette in an autorun format. These programs automatically run themselves once they are loaded from cassette. The procedure described above must be modified in the following manner to make an autorun cassette:

1. The first line of the program to be saved must be

```
POKE 515, Ø
```

2. Follow the SAVE prodecure described above only to step 5. Between steps 4 and 5 type in RUN before you turn off the tape recorder, then type LOAD <RETURN>.

Although a cassette recorder provides an inexpensive means of storing programs, the LOAD and SAVE procedures are slow, and keeping track of the location of multiple programs on a cassette can be cumbersome. A mini-floppy disk unit provides a much faster and more convenient method of saving and loading files. The Challenger 1P MF Series 2 is a mini-floppy disk based version of the C1P. In addition to all the features of the standard C1P, it incorporates a single mini-floppy disk drive and 2ØK of RAM. The C1P MF Series 2 comes complete with two disk operating systems—PICO DOS and OS-65D. The extra RAM memory is necessary to use these disk operating systems since these operating systems are themselves stored in RAM each time the disk is loaded.

The PICO DOS or disk operating system uses ROM BASIC. It allows the use of cassette originated programs on diskettes. PICO DOS occupies approximately 4K of RAM and operates with a fixed 8K workspace. Thus PICO DOS can actually be utilized on a C1P system with a 61Ø expander board and 12K of RAM. This is an intermediate growth step between the C1P Series 2 and the C1P MF Series 2.

The OS-65D operating system is a more powerful disk operating system. This disk operating system occupies somewhat over 12K of RAM and uses 9-digit BASIC by Microsoft rather than the built-in ROM BASIC. With 2ØK of RAM, the C1P MF Series 2 has an 8K workspace under the OS-65D disk operating system. With added memory the workspace under OS-65D can be expanded to 2ØK (or a total of 32K RAM).

Mini-floppy diskettes and disk drives are precision pieces of hardware and require reasonable care to insure continued satisfactory performance. Appendix 8 includes some guidelines on the handling of floppy diskettes ad disk drives.

THE PICO DISK OPERATING SYSTEM

The PICO DOS system provides an extension of the BASIC-in-ROM LOAD and SAVE commands to permit files to be saved on mini-floppy diskettes as well as on cassettes. This system allows for the storage of 8 programs on a single mini-floppy diskette.

In order to use the PICO disk operating system, first turn on the power to the computer, video monitor and floppy disk unit and depress <BREAK> until the prompt "D/C/W/M?" appears in the lower left corner of the screen. Insert a PICO DOS diskette, label side up, into the mini-floppy drive (the "A" drive if you have a dual disk drive system) and press <D>. The PICO disk operating system will respond with the following message

```
PICO DOS V1. 1

MEMORY SIZE? 8955

TERMINAL WIDTH?
```

The memory size is automatically set at 8955 by the PICO disk operating system. Unless the terminal width needs to be changed from the default value of 132 to meet the needs of a specific output device, just enter a <RETURN> in response to the query TERMINAL WIDTH?

The new commands available under the PICO disk operating system are

```
LOAD n
```

and

```
SAVE n
```

where n is program number from 1 to 8. These supplement the normal cassette LOAD and SAVE commands, which still function as before.

To save a program, simply enter it into the computer either through the keyboard, from cassette or perhaps from another PICO DOS diskette and type SAVE n where n is any number between 1 and 8. For example, the command SAVE 5 will save the contents of workspace on the fifth file on the disk. This will erase any program previously stored there. This program can be recalled at a later time with the command LOAD 5. Once the program is loaded into workspace from a diskette, it can be listed, modified and executed in exactly the same manner as programs entered through the keyboard or from cassette.

THE OS-65D DISK OPERATING SYSTEM

The OS-65D disk operating system is a convenient to use disk operating system which fully supports Microsoft's 9-Digit Extended BASIC, an optional 65Ø2 resident Assember/Editor, an optional 65Ø2 Extended Machine Code Monitor and various I/O devices. It supports writing programs in BASIC, storing programs on disk by name or track number, recalling programs and reading and writing sequential and random access data files in BASIC. The system is also well suited to utilize machine code subroutines in conjunction with BASIC programs.

In order to use the OS-65D disk operating system, first turn on the power to the computer, video monitor and floppy disk unit and press <BREAK> until the prompt D/C/W/M? appears in the lower left corner of the screen, check that the Shift Lock key is down. Insert an OS-65D diskette into the mini-floppy drive (the "A" drive if you have a dual disk drive system), remember to check the shift lock key and press <D>. When <D> is depressed the OS-65D disk operating system is loaded into memory and a BASIC program called BEXEC* is automatically loaded and executed. The program BEXEC* on the standard OS-65D development disk causes the following text to be displayed on the screen

```
BASIC EXECUTIVE FOR

OS-65D V3. N

MO, DAY, YR RELEASE

FUNCTIONS AVAILABLE:

  CHANGE—ALTER WORK-SPACE LIMITS

  DIR—PRINTS DIRECTORY

  UNLOCK—UNLOCK SYSTEM FOR END USER MODIFICATIONS

FUNCTION?
```

(On some special applications diskettes the program BEXEC* has been modified. When these diskettes are loaded the response may differ from that listed above. With these disks the user should just respond as directed by the displayed message.)

If the user responds CHANGE or DIR (for a directory of the diskette), then these programs are loaded and executed. When these programs finish executing the OK prompt is displayed, but the system is in a LOCKED mode and will not allow the user to enter new BASIC programs.

If the user responds UNLOCK to the query FUNCTION? then the system is placed in the BASIC immediate mode with display of the prompt OK. This prompt serves the same function as the OK in BASIC-in-ROM. It indicates that the system is prepared to respond to the standard BASIC commands, such as RUN. Unlocking the system does not remove the program BEXEC* from the workspace. If the command LIST is entered after the response UNLOCK, the program BEXEC* will be listed.

If the user depresses <CONTROL S> while a program is being listed or while a program is running, the listing or the execution will be interrupted until <CONTROL Q> is depressed. Before beginning to enter a new program the user should type NEW to clear the workspace.

The commands NEW and LIST are not acknowledged when BASIC is in the LOCKED mode. In order to UNLOCK the system, the user must run BEXEC* and respond UNLOCK to the query FUNCTION?. The program BEXEC* can be run either by entering the command RUN"BEXEC*" or by pressing <BREAK> and reloading or rebooting the OS-65D disk operating system. If the user enters the response UNLOCK (followed as usual by <RETURN>), then the system is unlocked. This allows the user to assume control of the system with the capability to erase old programs, enter new programs and list programs in the workspace.

BASIC programs can be entered through the keyboard in essentially the same manner as when BASIC-in-ROM was used. One slight difference is that when the <SHIFT O> is used as a backspace the character to be deleted is actually erased and the cursor moved one space to the left.

Before discussing the techniques for storing programs on diskette under the OS-65D disk operating system it will be helpful to describe some of the utility programs supplied with each OS-65D development disk. Each OS-65D development disk is shipped with either a black or white write protect tape attached. This tape is located near the upper right hand corner of the disk and covers a notch in the diskette cover. With this tape attached, it is possible to read from the disk but impossible to modify, or write to, the disk. This tape provides protection against inadvertent writes to the disk. The tape must be removed before anything can be stored on this disk. In particular, the utility programs CREATE and DELETE will not execute on a write protected disk (as indicated by an ERR #4 message).

Each OS-65D development disk contains a BASIC program named DIR. This program prints a director of the named files present on the diskette. Please note that programs stored without names will not be listed. This program can be run once the system has been unlocked by entering the command RUN"DIR." This program can also be run by responding DIR to the query FUNCTION? when the diskette is first loaded. A third way to run this program is to enter RUN"21." When the program DIR is run it first asks

```
LIST ON LINEPRINTER INSTEAD OF DEVICE # 2 ?
```

Depending upon your response the following output will appear either on the screen or on a printer (if one is attached and you respond YES).

```
             OS-65D VERSION 3. N
               —DIRECTORY—
FILE NAME                  TRACK RANGE
OS-65D3                           Ø-12
BEXEC*                           14-14
CHANGE                           15-16
CREATE                           17-19
DELETE                           2Ø-2Ø
DIR                              21-21
DIRSRT                           22-22
RANLST                           23-24
RENAME                           25-25
SECDIR                           26-26
SEQLST                           27-28
TRACE                            29-29
ZERO                             3Ø-31
ASAMPL                           32-32
     5Ø ENTRIES FREE OUT OF 64

OK
```

Each mini-floppy diskette has 4Ø tracks, numbered Ø-39. As the above listing shows, tracks Ø-12 are reserved for the OS-65D disk operating system. Note that the program BEXEC* is located on track 14. With the exception of the file ASAMPL which contains a sample assembler routine, each of the other files contains a utility program written in BASIC. These programs can be used without any knowledge of how they are implemented, but the interested user may find it useful to study them as sample programs since they demonstrate a wide variety of programming and file accessing techniques.

The directory listed above indicates that tracks 33-39 are currently not in use. These tracks can be used to store programs written by the user.

The OS-65D disk operating system allows the user to store programs either by track number or by name. The command

```
DISK!"PUT 33"
```

will store the program in workspace on the disk starting at track 33. This method of storing programs must be used with caution since there are no safeguards to prevent overwriting of existing files, as there are with the named file procedures.

Before a BASIC program can be stored by name, it is necessary to create a file to receive it. This will require an estimation of how many tracks your program will use at 2K bytes per track (see page 35). The utility program CREATE provides a means of adding new named files to the directory. To create a file, type

```
RUN"CREATE"
```

(You must be in the BASIC immediate mode as indicated by the prompt OK in order to enter this command.) This command will cause the BASIC utility program CREATE to be loaded and executed. The program output and the expected responses are shown below. Any unacceptable response will result in termination of the program or a repeat of the request for input.

```
FILE CREATION UTILITY

PASSWORD?
```

Unless you modify the code for the program CREATE, the password for this and all other OSI utility programs is just the word PASS. After you enter this password (and press <RETURN>) the program continues with an explanation of its operation:

```
CREATES AN ENTRY IN DIRECTORY FOR A NEW
FILE AND INITIALIZES THE TRACKS THAT THE
NEW FILE WILL RESIDE ON. THE TRACKS WILL
CONTAIN NULLS WITH A RETURN AT THE END
OF THE TRACK.
FILE NAME?
```

Enter a one to six character file name that is not a duplicate of an existing file on the disk. The file name must begin with a letter. The program will then respond.

```
FIRST TRACK OF FILE?
```

Enter the number of the first track the file is to reside on. Note that a named file always begins on a track boundary and resides on a whole number of tracks. The next response is

```
NUMBER OF TRACKS IN FILE?
```

Enter the number of tracks on which the file is to reside. You will have to estimate how large your program will be. Each track of the disk will hold 2K of material. The program will perform a check to verify that the tracks you have specified are not currently occupied by any other named files in the directory. If the tracks you have specified are available, the program continues with

```
8 PAGES PER TRACK. IS THIS OK?
```

Each track on a mini-floppy has a maximum capacity of 8 pages with each page capable of storing 256 BYTES. When a file is being created to store a BASIC program the response to this question should be YES since this will make maximum use of the space available on the diskette.

The file will now be created and its name and track location will be entered into the directory. When the CREATE utility program is finished, the prompt OK will again appear on the screen.

The OS-65D approach to files requires that the user know how large his file needs to be when it is created. To be safe, the user can simply specify a disk file size as large or slightly larger than the available RAM for BASIC programs. For example, with a Challenger 1P MF with 2ØK of RAM, slightly less than 8K is available for programs. Since each track can store 2K BYTES (8 pages at 256 BYTES per page), a four track file will hold any BASIC program that can be entered into the machine.

The user should always maintain a scratch file, usually with the name SCRTCH, which is at least as large as the memory size of the computer. This would mean a 4-track (8K) SCRTCH file for a computer with 2ØK of RAM. This file can serve as temporary storage in several situations. If, for example, the user types in a program and then remembers that he did not create a file for it, then he can simply store the program temporarily on the file SCRTCH, run CREATE to create a new file to hold the program, reload the program from SCRTCH and then store it under its proper name.

It is clear from looking at the directory listing that the utility programs occupy a major portion of the disk and leave little room for the storage of user generated programs. A common solution to this problem is to maintain multiple copies of the OS-65D disk. At least one of these should be left intact with all the utilities present. On the other disks, the utility program DELETE can be used to remove the majority of the utility programs. A reasonable choice might be to delete all the utilities except DIR, CREATE and DELETE since these are the most commonly used utility programs. If the other utility programs are needed, they can be loaded from the OS-65D disk containing them.

The DELETE utility program can be run by typing

```
RUN"DELETE
```

The program output and the kind of input you may enter in response are shown below. Any unacceptable response will result in termination of the program or a repeat of the request for input.

```
DELETE UTILITY
```

```
REMOVES AN ENTRY FROM THE DIRECTORY
PASSWORD? (Enter PASS)
FILE NAME?
```

Enter the name of the file to be deleted and its name will be removed from the directory. The file is still physically present on the disk and can be run by track number. The DELETE utility merely removes its name from the directory.

The other utility programs present on the OS-65D disk will not be discussed in this manual. Their operation is completely described in the OS-65D USER'S MANUAL.

The OS-65D disk operating system contains its own command interpreter. This interpreter handles commands for such tasks as initializing diskettes, loading and saving files, loading the 9-Digit Extended Basic interpreter and loading the Assembler and Extended Monitor, if your disk has these. A summary of the commands in the OS-65D disk operating system is provided in Appendix 6.

For the purpose of loading and saving BASIC programs, the commands of primary interest are the two commands

```
LOAD FILNAM Loads named source file
FILNAM into memory
```

and

```
PUT FILNAM Saves source file in memory on
the named disk file FILNAM
```

These commands, as well as the other commands in the OS-65D disk operating system, are not recognized in this form when in the BASIC immediate mode. To enter these commands when in the BASIC immediate mode, they must be prefixed by DISK!" This prefix identifies the commands as part of the OS-65D disk operating system. The command interpreter only uses the first two characters in each OS-65D disk operating system command, so each command can be abbreviated to two letters.

Suppose now that you have created a file named PROG1 and have entered a BASIC program into the workspace and wish to save it in the file PROG1. The command

```
DISK!"PUT PROG1"   or   DISK!"PU PROG1"
```

will cause the source file to be stored in the file PROG1 on the diskette. There are four common user errors that can arise in connection with this command.

1. The diskette is write protected (ERR #4 message)
2. The disk drive is not turned on (no message on screen).
3. No file has been created to receive the program or the file name is entered incorrectly (EER #C message).
4. The program is too long to fit in the file (ERR #D message).

Appendix 7 contains a list of error codes which are associated with these and other disk I/O errors. If the disk drive is not turned on or the drive door is not shut, the system will just "hang" until the drive is ready and then an error condition will normally be issued. Remember to avoid turning the disk drive on or off when it contains a disk.

Once a BASIC program has been stored on a file, it can be loaded later from BASIC with the command

```
DISK!"LOAD PROG1   or   DISK!"LO PROG1
```

This command will cause the program to be loaded into the workspace. It can then be modified or executed with the standard BASIC commands (e.g. LIST, RUN, etc). OS-65D allows the user to combine the LOAD and RUN command into one command

```
RUN"PROG1
```

This feature has already been illustrated in our discussion of the utility programs.

As was pointed out earlier, the OS-65D disk operating system requires the user to specify the size of a file at the time it is created. If the user follows the preceding recommendations and creates a scratch file SCRTCH of sufficient size to store a program of maximum size, then the disk I/O errors #C (can't find that name in the directory) and #D (read/write attempted past the end of named file) can usually be easily handled by temporarily placing the program in the scratch file. Obviously, it is desirable to store a program in as small a disk file as possible to conserve disk space. The following discussion describes a simple procedure which allows the user to determine the number of tracks needed to store a program.

The OS-65D disk operating system allows the user to leave the BASIC immediate mode by entering the command EXIT. When this command is issued, two lines appear on the screen. These two lines will have the following form

```
ØN TRACK
A*
```

In the first line N is an integer which indicates the minimum number of tracks required to store the current contents of workspace on disk. If the user has just entered a BASIC program or is in the process of entering a BASIC program, this informs him how many tracks would be required to store the program in its present form. The second line, the A*, is the OS-65D disk operating system kernel prompter. It indicates that the user is no longer in the BASIC immediate mode, but rather is in the kernel of the disk operating system. Any of the OS-65D disk operating system commands can be issued when in the kernel (without the DISK!" prefix). If the user has entered the kernel from BASIC, the command RETURN BASIC or RE BA will return the user to BASIC and leave the contents of the workspace unchanged. The command BASIC or BA will also return the user to BASIC but the contents of workspace are lost. Thus, the user who is entering a BASIC program, can get an estimate of its size at any time by entering the command EXIT, noting the track requirements and returning with the command RETURN BASIC or simply RE BA.

Programs can be stored on cassette under OS-65D in the following manner:

1. Type DISK!"IO, Ø3 <RETURN>
2. Type NULL8 <RETURN>
3. Now turn on the tape recorder in the RECORD mode.
4. When the tape (dark brown) begins to wind onto the right-hand spool, type LIST <RETURN>
5. When the LIST is completed, turn off the tape recorder.

Programs can be read into the workspace from cassette under OS-65D using the following technique. The tape must be positioned immediately preceding the program to be loaded or extraneous noise will disrupt the load. More than one attempt may be necessary before the program will be successfully loaded.

1. Type DISK!"IO Ø1 but do not type <RETURN>
2. Start the cassette recorder and immediately press <RETURN>.

If the procedure is successful the program will begin listing on the screen. If not repeat the procedure.

This section has provided an introduction to the OS-65D disk operating system. The system includes a large number of features which we do not have room to properly cover here. The OS-65D USER'S MANUAL covers all of these features in detail and is required reading for the serious user who wishes to take full advantage of the capabilities of the system.

SECTION 12

ADVANCED FEATURES-LOWER CASE, KEYBOARD PROGRAMMABILITY

LOWER CASE

The Ohio Scientific Challenger 1P is capable of generating lower case characters as well as numerous graphics characters. Under normal operation, the shift lock key is in the depressed or locked condition. It must be in this position for normal systems level software to operate, this is because all BASIC commands must be given in capital letters. 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 sign (@) which is recognized in BASIC as being the line delete code (compare figure 6 in section six).

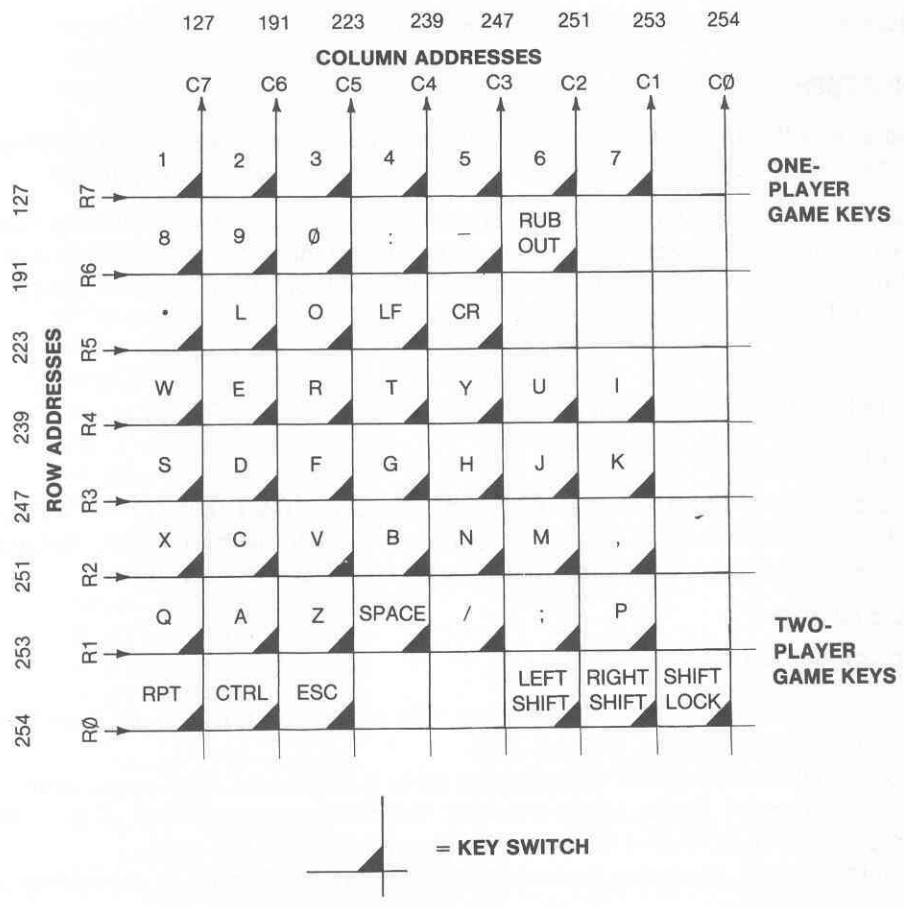
The shift lock key can be released for certain special applications. Specifically, to generate lower case characters as part of literal strings such as "This is a string" in BASIC and for use in conjunction with word processing, the keyboard will act very differently when the <SHIFT> key is not in the locked position. With the shift lock key up, only standard alphabetic keys 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 the numeral 5. Depressing the 5 key in conjunction with the right shift key generates the per cent (%) key. 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.

KEYBOARD PROGRAMMABILITY

The Challenger 1P 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.

The internal design of the polled keyboard on the Challenger 1P views the keyboard as an array of 8 rows and 8 columns (see Figure 10). Normally, when a program is not executing, a polling routine constantly scans each of the eight rows in succession to determine the column of any depressed key. If a key closure is detected, the polling routine supplies the ASCII code corresponding to the face of the key depressed. During the execution of a program this polling routine is disabled and replaced by a second routine which monitors the <CONTROL> and <C> keys. If these keys are simultaneously depressed, then execution of the program is terminated (see Figure 6 in section six).

In many applications it is useful to program keys for special purposes such as controlling the movement of figures on the screen. In order to program special key functions, the CONTROL-C polling routine must be temporarily disabled. The method for accomplishing this depends upon whether the program will be run with BASIC-in-ROM or 9-



NOTES: 1. Standard 53-key layout except:
"HERE IS" deleted, "RUB OUT" at "HERE IS" location,
"SHIFT LOCK" at "RUB OUT" location.
2. "LEFT SHIFT" and "RIGHT SHIFT" separately decoded.

Figure 10: Switch Matrix—CIP Polled Keyboard.

Digit Extended BASIC (with OS-65D). The following table summarizes the commands needed to enable (turn on) and disable (turn off) the CONTROL-C routine in each of these contexts.

BASIC-in-ROM

POKE 53Ø,1 — turns CONTROL-C off

POKE 53Ø, Ø — turns CONTROL-C on

9-Digit Extended BASIC

POKE 2Ø73, 96 — turns CONTROL-C off

POKE 2Ø73, 173 — turns CONTROL-C on

All keyboard polling is accomplished through the I/O port for the polled keyboard. This port is located at memory location 57Ø88 (address DFØØ in hexadecimal). The technique of polling the keyboard consists of two steps (a POKE and a PEEK):

1. address a row

This is accomplished by the statement

POKE 57Ø88, row address

For example, the statement POKE 57Ø88, 247 addresses R3 (see Fig. 9).

2. determine key closures within the column.

If, after addressing a specific row, we enter the statement

```
K = PEEK(57Ø88)
```

then the value of K will summarize the column addresses corresponding to key closures within that row. The value of K is the logical AND of the column addresses in which keys are depressed.

For two integers N1 and N2 the value of N1 AND N2 can be determined in the following manner. Express both N1 and N2 in binary notation. If necessary add leading zeros to the binary representation of one or the other numbers so that both numbers are the same length. The binary representation of N1 AND N2 has a 1 in any position that both N1 and N2 have a 1 and has Ø's elsewhere.

Example

```
147 (decimal) = 1ØØ1ØØ11 (binary)

89 (decimal) = Ø1Ø11ØØ1 (binary)
```

Thus the binary representation of 147 AND 89 = ØØØ1ØØØ1. Since ØØØ1ØØØ1 (binary) = 17 (decimal), the value of 147 AND 89 is 17. For more information on logical operations refer to the OSI BASIC Reference Manual.

For example, suppose a program contains the following two consecutive statements

```
1ØØ POKE 57Ø88, 247

11Ø K = PEEK(57Ø88)
```

K = 127 indicates that <S> is depressed. If K = 191 then <D> is depressed. If K = 63 then both <S> and <D> are depressed (since 63 is the logical AND of 127 and 191).

If K is expressed as an 8 bit binary number (with leading zeros if necessary), then zeros occur in exactly those columns in which keys are depressed. In the above example, the binary representation of 63 is ØØ11111. There are zeros in the first two columns-the columns in which the <S> and the <D> are located.

The following sample program illustrates these keyboard polling techniques in controlling movement on the screen.

PROBLEM: Write a BASIC program which will allow the user to control the movement of a small tank by depressing specified keys. Depressing one of the keys <1>, <2>, <3> and <4> should cause the tank to move to the right, up, left and down respectively. Depressing <S> should terminate the program.

SOLUTION: There are eight different tank figures available in the graphics character set (characters 248-255) printed in Appendix 8. Since we shall not allow diagonal movement in this program, we will only use characters 248, 25Ø, 252 and 254. The following is a list of the most important variables used in the following program.

T	determines the tank figure displayed
L	specifies the location of the tank on the screen
D1	contains the distance of the tank from the right hand edge of the screen
D2	contains the distance of the tank from the top of the screen
D3	contains the distance of the tank from the left hand edge of the screen
D4	contains the distance of the tank from the bottom of the screen
K3	summarizes the key closures in row R3
K7	summarizes the key closures in row R7

A number of comments have been included to the right of the following listing to explain the logic of the program. These comments should not be included when you type the program.

```
1Ø          REM—TANK MOVER                Replace line 3Ø by
2Ø          REM—DISABLE CNTL-C            POKE 2Ø73, 96 for
3Ø          POKE 53Ø, 1                   9-Digit BASIC
4Ø          REM—CLEAR THE SCREEN
5Ø          FOR I=1 TO 3Ø
6Ø          PRINT
7Ø          NEXT
8Ø          REM—SELECT TANK 252
```

```
  9Ø    T = 252
 1ØØ    REM—LOCATE AT CENTER
 11Ø    REM—OF SCREEN
 12Ø    L = 53776
 14Ø    REM—SET DISTANCES           Cell 53776 is 12 cells
 15Ø    REM—TO EDGES                from right and top,
 16Ø    D1=12: D2=12                11 cells from left and
 17Ø    D3=11:D4=11                 bottom.
 18Ø    REM—DISPLAY TANK
 19Ø    POKE L, T
 2ØØ    REM—POLL R3
 21Ø    POKE 57Ø88, 247
 22Ø    K3 = PEEK(57Ø88)
 23Ø    REM—STOP ON <S>
 24Ø    IF K3=127 then 9ØØØ
 25Ø    REM—POLL R7
 26Ø    POKE 57Ø88, 127
 27Ø    K7 = PEEK(57Ø88)
 28Ø    REM—CHECK DIRECTION
 29Ø    IF K7=127 THEN 1ØØØ
 3ØØ    IF K7=191 THEN 2ØØØ
 31Ø    IF K7=223 THEN 3ØØØ
 32Ø    IF K7=239 THEN 4ØØØ
 33Ø    REM—GO TRY AGAIN            No match was found.
 34Ø    GOTO 2ØØ
1ØØØ    REM—1 PRESSED               Key <1> was pressed.
1Ø1Ø    REM—SELECT TANK
1Ø2Ø    T = 25Ø                     Tank 25Ø points right.
1Ø3Ø    REM—ERASE TANK
1Ø4Ø    POKE L, 32
1Ø5Ø    REM-UPDATE L                Increment L by 1 to move
1Ø6Ø    IF D1>Ø THEN L=L+1          to right if not at edge.
1Ø7Ø    REM—UPDATE D3 AND D1        Increment D3 by 1 and
1Ø8Ø    IF D1>Ø THEN D3=D3+1        decrement D1 by 1 if
1Ø9Ø    IF D1>Ø THEN D1=D1−1        not at edge.
11ØØ    REM—REDRAW TANK
111Ø    GOTO 18Ø
2ØØØ    REM—2 PRESSED               Key<2> was pressed.
2Ø1Ø    REM—SELECT TANK
2Ø2Ø    T = 248                     Tank 248 points up.
2Ø3Ø    REM—ERASE TANK
2Ø4Ø    POKE L, 32
2Ø5Ø    REM—UPDATE L                Decrement L by 32 to move
2Ø6Ø    IF D2>Ø THEN L=L−32         up if not at edge.
2Ø7Ø    REM—UPDATE D4 AND D2        Increment D4 by 1 and
2Ø8Ø    IF D2>Ø THEN D4=D4+1        decrement D2 by 1 if
2Ø9Ø    IF D2>Ø THEN D2=D2−1        not at edge.
21ØØ    REM—REDRAW TANK
211Ø    GOTO 18Ø
3ØØØ    REM—3 PRESSED               Key <3> was pressed.
3Ø1Ø    REM—SELECT TANK
3Ø2Ø    T = 254                     Tank 254 points left.
3Ø3Ø    REM—ERASE TANK
3Ø4Ø    POKE L, 32
3Ø5Ø    REM—UPDATE L                Decrement L by 1 to move
3Ø6Ø    IF D3>Ø THEN L=L−1          left if not at edge.
```

```
3Ø7Ø    REM—UPDATE D1 AND D3           Increment D1 by 1 and
3Ø8Ø    IF D3>Ø THEN D1=D1+1           decrement D3 by 1 if
3Ø9Ø    IF D3>Ø THEN D3=D3-1           not at edge.
31ØØ    REM—REDRAW TANK
31ØØ    GOTO 18Ø
4ØØØ    REM—4 PRESSED                  Key <4> was pressed.
4Ø1Ø    REM—SELECT TANK
4Ø2Ø    T = 252                        Tank 252 points down.
4Ø3Ø    REM—ERASE TANK
4Ø4Ø    POKE L, 32
4Ø5Ø    REM—UPDATE L                   Increment L by 32 to move
4Ø6Ø    IF D4>Ø THEN L=L+32            down if not at edge.
4Ø7Ø    REM—UPDATE D2 and D4           Increment D2 by 1 and
4Ø8Ø    IF D4>Ø THEN D2=D2+1           decrement D4 by 1 if
4Ø9Ø    IF D4>Ø THEN D4=D4-1           not at edge.
41ØØ    REM—REDRAW TANK
411Ø    GOTO 18Ø
9ØØØ    REM—TIME TO QUIT               Replace line 9Ø2Ø by
9Ø1Ø    REM—RESTORE CNTL-C             POKE 2Ø73, 173 for
9Ø2Ø    POKE 53Ø, Ø                    9-Digit BASIC.
9Ø3Ø    END
```

To facilitate entering this program you might skip all lines beginning with REM, since these lines do not affect the operation of the program. It is still good programming practice to use REM's throughout your programs to document them.

Once you have successfully entered the preceding program and feel that you understand the logic, there are several variations you might attempt.

1. Eliminate the variables D3 and D4. D1 and D2 can be used to check all four edges.
2. Modify the program to allow diagonal moves.
3. Convert the program to a two player game with each player controlling a different figure and one of the players trying to catch the other.

This program gives an introduction to the potential uses of the polled keyboard and graphics display capabilities of the Challenger 1P. Very elaborate arcade games can be written to run on the Challenger 1P. Many such games are available from Ohio Scientific on cassette and diskette for use on your C1P.

SECTION 13

PRINTER COMMUNICATIONS

The Challenger 1P Series 2 computer is provided with a switch selectable audio cassette, 3ØØ baud modem and printer interface. Figure 1 in section three gives two views of the rear panel of the Challenger 1P.

In order to interface a serial printer with the Challenger 1P, the printer cable should be connected to the printer output port located on the rear panel of the C1P and the selector switch should be rotated to the center (printer) position. The method used for output to the printer depends upon whether BASIC-in-ROM or 9-Digit Extended BASIC under OS-65D is used.

BASIC-IN-ROM—PRINTER USE

When BASIC-in-ROM is being used with the Challenger 1P, output to the printer is handled in the same manner as output to cassette. If the command SAVE is entered, then all subsequent output which would normally appear on the screen is routed to both the screen and the printer. Set the rotary switch to printer, see page 7. Output will continue to be routed to the printer as well as the screen until the user enters the following sequence of commands:

```
LOAD <RETURN>
<SPACE> <RETURN>
```

These two commands terminate output to the printer in the same way that they terminate output to the cassette recorder when the switch is set for cassette input/output.

For example, a program in the workspace can be listed on the printer by the following series of commands:

```
SAVE
LIST
LOAD
<SPACE>
```

As usual, each of these commands should be followed by <RETURN>. The program will begin listing after the command LIST is entered. The command LOAD should be entered after the LISTING is complete. If the printer is not on line or is connected incorrectly (or if the selector switch is turned to printer when no printer is connected) then the computer will stall when the command LIST is entered until the problem is corrected, the switch is reset or <BREAK> is depressed. The results of any PRINT statements are displayed on both the screen and the printer. Note the printer output is 3ØØ baud, like the cassette output.

9-DIGIT EXTENDED BASIC UNDER OS-65D—PRINTER USE

When OS-65D is being used with the C1P, output can be directed to the printer by changing the output flag. This is accomplished by a disk operating system command. The following illustrates the method of accomplishing this:

```
DISK!"IO ,Ø1"— this directs subsequent output to the printer only
DISK!"IO ,Ø2"— this directs subsequent output to the screen only
DISK!"IO ,Ø3"— this directs subsequent output to both the printer and the screen
```

The default mode sets the output flag to send output to the screen. The output flag is automatically reset to "Ø2" (the screen) whenever the computer encounters a syntax error or an abnormal condition such as a CONTROL-C halt to a listing or run of a program.

For the purposes of printer output, setting the output flag to "Ø3" has very much the same effect as entering SAVE when using BASIC-in-ROM. The output to the printer can be terminated by resetting the output flag to "Ø2" with the command DISK!"IO,Ø2."

Under OS-65D the command LIST#1 can be used to list the contents of the workspace on the printer without the necessity of changing the output flag with a DISK!"IO command. The program is listed only on the printer (not on the screen) when this command is entered. Printer output is also accomplished by PRINT#1,"STATEMENT."

Another method to output to the printer is to use a POKE8994,1 for output to the printer only and POKE8994,3 for output to the screen and the printer simultaneously.

A complete discussion of the I/O capabilities of the C1P under OS-65D is beyond the scope of this manual. The interested user is referred to the OS-65D USER's MANUAL for a complete treatment of this topic.

If the user adds the 63Ø I/O Expander to his Challenger 1P, the choice between modem and high speed printer ports is under program control rather than manual control by a switch.

SECTION 14

MODEM AND TERMINAL COMMUNICATIONS

The Challenger 1P can be used as a terminal to communicate with another computer over a telephone. In order to use the Challenger 1P in this manner requires a modem (short for "modulator-demodulator"). This is a hardware item used to connect a telephone to your computer. The computer signals the modem to generate or receive tones which are carried over the telephone lines. Ohio Scientific offers a competitively priced modem suitable for use with the Challenger 1P, catalog item AC-11P.

An RS-232 port is provided for connecting a modem to the rear of the Challenger 1P. In order for the Challenger 1P to communicate with the modem, the selector switch on the rear panel of the computer must be set to the right (modem) position.

The following is a general summary of the sequence of steps necessary to use the C1P as a terminal:

1. Connect a modem to the modem port and set the selector switch on the back of the C1P to the right hand position. The modem should be set to full duplex and originate mode.
2. Load the modem program provided by Ohio Scientific into the C1P. When it is loaded the computer will respond READY. Phone numbers of local modem services are available from your local OSI dealer.
3. Dial the number of the remote computer. When the number dialed answers you should hear a high pitched tone. Insert the phone in the modem and follow the instructions displayed on the screen. The computer called will probably require that you enter a user code and password.

When the Challenger 1P is equipped with the 63Ø I/O Expander the selector switch on the rear panel of the computer is removed. For these models of the C1P, a bit of the special control register located at memory location 63456 (address F7EØ in hexadecimal) is used to select between printer and modem I/O. For example,

POKE 63456,Ø selects the printer

and

POKE 63456,4 selects the modem.

NOTE: See page 77 for more information:

SECTION 15

JOYSTICKS AND KEYBOARDS

JOYSTICKS

The joysticks provide realistic and convenient input devices for games and control. When the joysticks are connected (as shown in Figure 1) and enabled, they generate a digital signal which may be read by the computer.

Only one joystick may be enabled at a time, this is accomplished via a POKE statement. The joysticks are designated A and B, and the POKEs to enable them are:

POKE 6344Ø,127 enables Joystick A

POKE 6344Ø,224 enables Joystick B

As seen in Figure 12, each joystick offers nine possible unique physical positions, labeled A-I. In addition to these positions, each joystick has an "action key," which effectively doubles the possible combinations for each joystick. Detection of the various positions, for a particular joystick, is accomplished by the following procedure;

First, POKE on the joystick that you wish to "read," for example POKE 6344Ø,127 to detect Joystick A outputs.

Next, PEEK location 6344Ø with one catch. In order to avoid unintentional interaction between the joysticks, it is necessary to use the logical function OR on the value PEEKed at 6344Ø, for example, PEEK(6344Ø) OR 224 for joystick A [PEEK(6344Ø) OR 7, for joystick B]. The effect of this command is that the computer will ignore any bit patterns generated by inadvertent use of joystick B while monitoring the output of joystick A.

Figure 11 lists all possible decimal output values from the joysticks, including values with and without the "action key" depressed.

	JOYSTICK A		JOYSTICK B	
	ACTION KEY NOT DEPRESSED	ACTION KEY DEPRESSED	ACTION KEY NOT DEPRESSED	ACTION KEY DEPRESSED
A	239	238	223	95
B	235	234	207	79
C	251	250	239	111
D	243	242	175	47
E	247	246	191	63
F	245	244	183	55
G	253	252	247	119
H	237	236	215	87
I	255	254	255	127

Figure 11: Decimal Values Returned by Joysticks

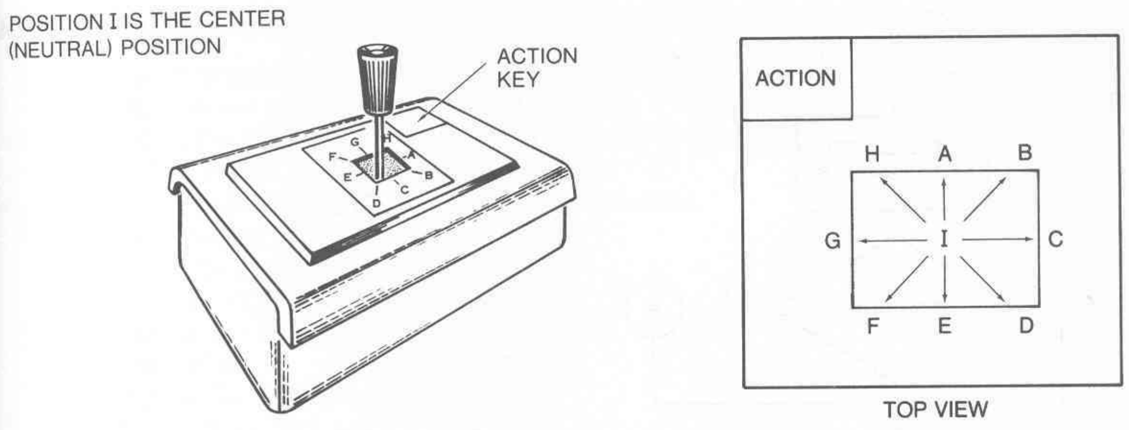


Figure 12: Joystick

The following program, called Joystick Arrows, serves two purposes. First, it is a concise example of "how-to" program in order to use the joysticks. Second, it can serve as the basis of your own game programs.

Figure 13 depicts the various characters used by Joystick Arrows. Figure 14 is the flowchart for this program. Clean out the workspace of your computer (use a NEW), and enter the program exactly as it appears on page 47. After you have it working, store it on tape or disk for future reference and possible modification.

While experimenting with Joystick Arrows, you may find that the arrow occasionally disappears. This happens because the program does not check to see whether the location (variable P) actually corresponds to a screen location. As an exercise, try to modify Joystick Arrows to prevent the arrow from going off the screen.

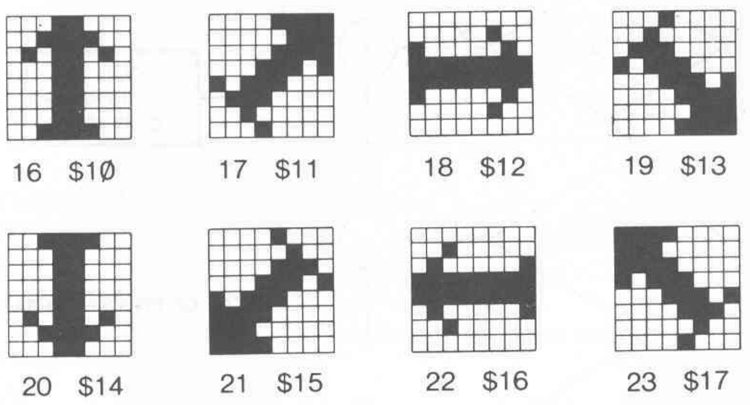


Figure 13: Characters Used in Joysticks Arrows Program

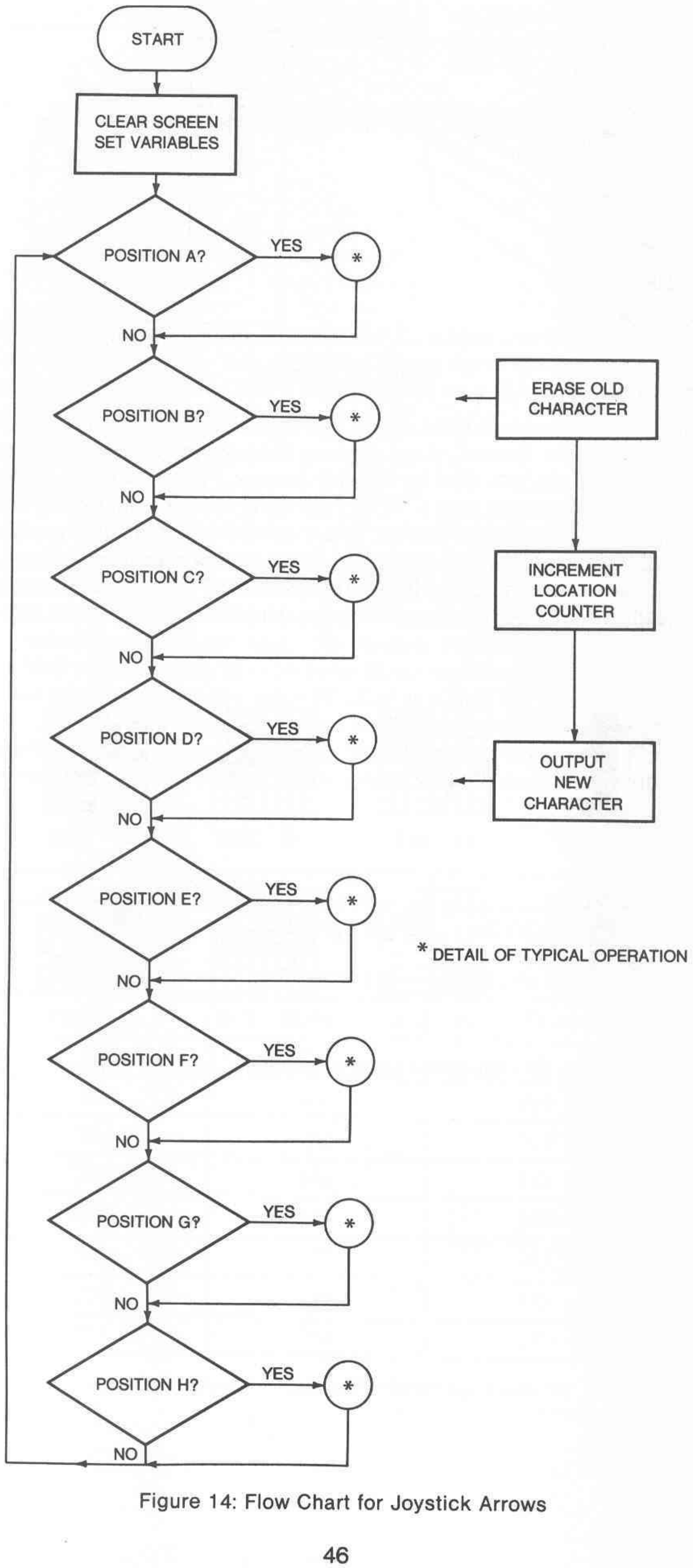


Figure 14: Flow Chart for Joystick Arrows

JOYSTICK ARROWS PROGRAM

```
  10  FORSC=1 TO 25: PRINT: NEXT                          :REM CLEAR THE SCREEN
  20  AP=-32: BP=-31: CP=1: DP=33                         :REM CONSTANTS FOR
  30  EP=32: FP=31: GP=-1: HP=-33: IP=0                   :REM MOVEMENT
  40  A=239: B=235: C=251: D=243                          :REM JOYSTICK A
  50  E=247: F=245: G=253: H=237: I=255                   :REM POSITIONS
  60  P=53711: BLANK=96
  70  POKE63440, 127: POKEP,16
 110  R=PEEK(63440) OR 224                                :REM CHECK JOYSTICK A
 120  IF R=IP THEN 110
 200  REM             POSITION A?
 210  IF R=A THEN 230
 220  GOTO 300
 230  POKEP,BLANK
 250  P=P+AP: POKEP,16
 300  REM             POSITION B?
 310  IF R=B THEN 330
 320  GOTO 400
 330  POKEP,BLANK
 350  P=P+BP: POKEP,17
 400  REM             POSITION C?
 410  IF R=C THEN 430
 420  GOTO 500
 430  POKEP,BLANK
 450  P=P+CB: POKEP,18
 500  REM             POSITION D?
 510  IF R=D THEN 530
 520  GOTO 600
 530  POKEP,BLANK
 550  P=P+DP: POKEP,19
 600  REM             POSITION E?
 610  IF R=E THEN 630
 620  GOTO 700
 630  POKEP,BLANK
 650  P=P+EP: POKEP,20
 700  REM             POSITION F?
 710  IF R=F THEN 730
 720  GOTO 800
 730  POKEP,BLANK
 750  P=P+FP: POKEP,21
 800  REM             POSITION G?
 810  IF R=G THEN 830
 820  GOTO 900
 830  POKEP,BLANK
 850  P=P+GP: POKEP,22
 900  REM             POSITION H?
 910  IF R=H THEN 930
 920  GOTO 110
 930  POKEP,BLANK
 950  P=P+HP: POKEP,23
 999  GOTO 110
1000  END
```

KEYPADS

The keypads function on all C1P's equipped with the optional 63Ø IO expander board. Two keypads, designated A and B, plug into the connectors shown in Figure 1.

Several steps are involved in using the keypads. The keypad memory location is 6344Ø. Figures 15 and 16 detail the POKEs and Peeks used with the keypads. The following short program is presented as an example of keypad programming. Suppose that you want to determine if the "2" key on keypad A has been depressed. The following routine will do just that.

```
1Ø POKE 6344Ø,239

2Ø IF PEEK(6344Ø)=191 THEN PRINT "2 PRESSED"

3Ø GOTO 2Ø
```

The phrase "2 PRESSED" will be printed on the screen whenever the "2" key on keypad A is pressed. You may ask, "why 239 in line 1Ø and 191 in line 2Ø?" For the answers, examine Figure 15. Note that the row containing "2" must be "turned on" (by a POKE6344Ø,239) and a "2" output is then indicated by the value at the top of the column containing "2", or C6=191). That's why 239 and 191.

The following program illustrates one method to recognize all twelve keys of keypad A under software control. This type of routine would be useful in arithmetic quiz and drill type programs or as a numeric input routine for an accounting package and many others.

```
 1Ø       X=6344Ø
1ØØ       POKEX,239
1Ø5       Y=PEEK(X)                      :REM CHECK ROW 4
11Ø       IF Y=127     THEN PRINT "1":GOTO 1Ø
12Ø       IF Y=191     THEN PRINT "2":GOTO 1Ø
13Ø       IF Y=223     THEN PRINT "3":GOTO 1Ø
2ØØ       POKEX,247
2Ø5       Y=PEEK(X)                      :REM CHECK ROW 3
21Ø       IF Y=127     THEN PRINT "4":GOTO 1Ø
22Ø       IF Y=191     THEN PRINT "5":GOTO 1Ø
23Ø       IF Y=223     THEN PRINT "6":GOTO 1Ø
3ØØ       POKEX,251
3Ø5       Y=PEEK(X)                      :REM CHECK ROW 2
31Ø       IF Y=127     THEN PRINT "7":GOTO 1Ø
32Ø       IF Y=191     THEN PRINT "8":GOTO 1Ø
33Ø       IF Y=223     THEN PRINT "9":GOTO 1Ø
4ØØ       POKEX,253
4Ø5       Y=PEEK(X)                      :REM CHECK ROW 1
41Ø       IF Y=127     THEN PRINT "*":GOTO 1Ø
42Ø       IF Y=191     THEN PRINT "Ø":GOTO 1Ø
43Ø       IF Y=223     THEN PRINT "#":GOTO 1Ø
5ØØ       GOTO 1Ø
```

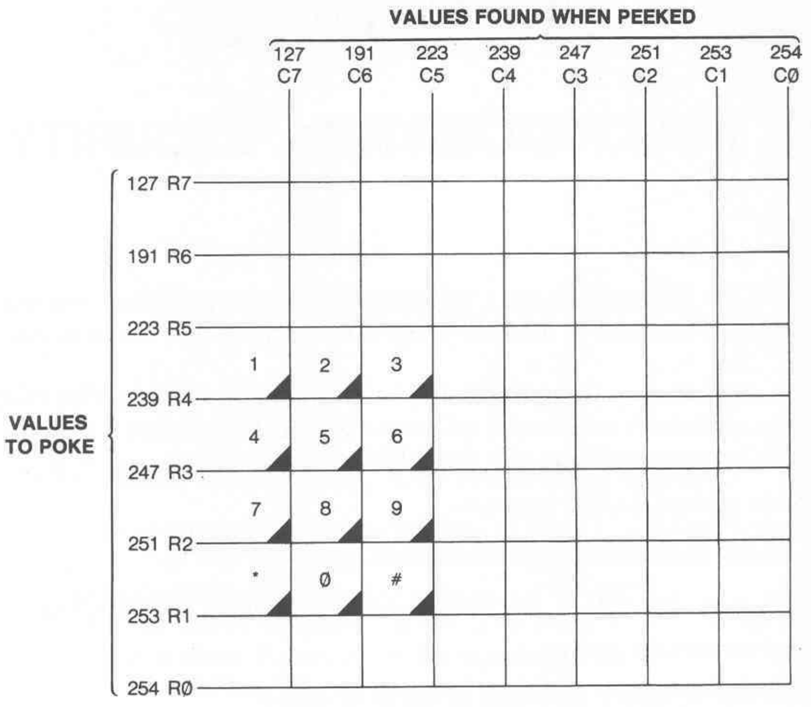


Figure 15: Keypad A

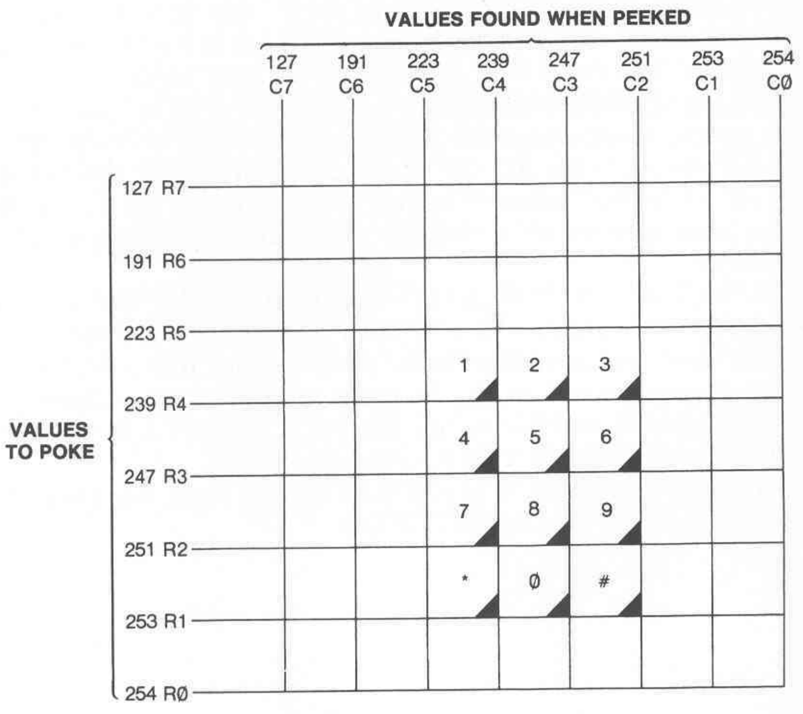


Figure 16: Keypad B

SECTION 16

AC REMOTE CONTROL SECURITY

The installation of the 63Ø I/O Expander on the Challenger 1P makes it possible to use the C1P to control lamps and appliances throughout the home and to monitor a home security system. These popular applications are described in this section.

Ohio Scientific offers the AC-12P remote control starter set including an OSI modified BSR X-1Ø command console, two lamp modules, two appliance modules and software. Ohio Scientific offers a special home control OS-65D, designated HC1, floppy disk operating system with the following capabilities:

1. Compatible with most normal BASIC programs.
2. Supports the time of day and count down event timer.
3. Supports up to 16 separate channels of AC remote control (requires the CA-21 option).
4. includes proportional control of lighting.
5. Constantly maintains on/off sensor detection of up to 48 inputs.
6. Disk event logging by time or event.

The C1P MF with the 63Ø incorporates an internal real time clock. Thus the home control operating system may always know what time it is and maintains a count down timer which can be used to cause a user specified action to occur at a specified time (such as "two hours from now turn on the front porch light").

The AC-12P remote system can be simply used to turn a few things on or off or it can be expanded to a full blown computerized home control system. Installation of the AC-12P on Challenger 1P is accomplished by connecting the modified BSR X-1Ø command console to the AC-12P interface jack on the rear panel of the Challenger 1P (see Figure 1 in section 3). Signals sent to the command module by the computer are transmitted over existing home wiring to special light and appliance modules which plug into wall sockets. For further details refer to the documentation included with the AC-12P.

The Challenger 1P can be interfaced with a Fyrnetics Lifesaver Home Security System. This system, and supporting software, is available from Ohio Scientific as AC-17P. The Fyrnetics unit scans various security monitor inputs and audibly registers any fault condition. In addition to the audible signal, four conditions are registered at screw connections at the rear of the Fyrnetics unit. The Ohio Scientific software monitors these four connectors via a 16-pin DIP cable. When a fault is detected by software, any number of different actions may be taken. The demonstration diskette included with the system reflect the simplest approaches.

The AC-17P is connected to your Challenger 1P at port J3 on the rear panel (see Figure 1 in section three). For more detail see the instructions included with the AC-17P unit.

SECTION 17

PARALLEL I/O

EXTERNAL SWITCHES, ALARMS, OR INDICATORS

In AC control and home security systems, there is often need to sense switch openings or closings. Relay contacts might indicate an air-conditioner "on" for an energy management system; an open window might be read as a set of open contacts to a home security system. Individual imagination is the limit.

The C1P system provides (in the CA-21 package) the ability to sense 48 separate remote contact-pairs. Each of these contact-pairs (lines) is to be at either Ø volts or 5 volts (standard TTL levels). When these lines are computer driven (used for output), a maximum of two TTL devices can be driven at a time. If devices other than OSI peripheral devices are used, be cautioned to use good circuit practices in interfacing circuits.

The input lines are grouped as 6 sets of 8 lines (6x8=48), or 6 input registers. Associated with each input register (group of 8 lines) is a mask register (tells which of the 8 lines to ignore) and an active state register (tells whether a 5 volt or Ø volt signal is to be the chosen active state). The state of each line can be sensed by examining the register bit which reflects the state of the connected line. In the case of windows, for example, it might be desired to identify the active state as an open window in one program but in a different program to have the active state reflect a closed window. Which one is desired will depend on the program.

The associated registers, i.e., the mask register and active state register, are used by the real time monitor, RTMON, to systematically scan the input lines. When an input line becomes active, RTMON's services are requested (in the same manner as the count down timer requested service). Once again, discussion of how RTMON uses these associated registers will be put off until after examination of the hardware which is used to support it.

The associated registers are memory locations which are examined to determine how to interpret switch positions. In contrast, the hardware registers directly indicate line status, 5 volts or Ø volts. The hardware registers also indicate whether a set of lines is to receive signals (be read) or whether output signals should be sent to turn on/off devices (to be written to).

External switches which can be used to provide 5 volts or Ø volts are connected (through back panel connectors, Figure 1) to a *P*eripheral *I*nterface *A*dapter (PIA). The PIA presents groups of input lines for input or output of signals. These input or output lines are addressed in groups of 8 lines. The PIA is a single integrated circuit. Its organization and use are best explained in terms of its addressing, i.e., where the computer looks to input or output data. For this purpose, a map is created.

PIA REGISTERS

Map of the hardware registers used for input and output.

DATA REGISTER				CONTROL REGISTER	
HEX LOCATION	DECIMAL LOCATION	7	Ø BIT	DECIMAL LOCATION	HEX LOCATION
C7Ø4	5Ø948	Port 1A			
			CTRL Register For Port 1A	5Ø949	C7Ø5
C7Ø6	5Ø95Ø	Port 1B			
			CTRL Register For Port 1B	5Ø951	C7Ø7
C7Ø8	5Ø952	Port 2A			
			CTRL Register For Port 2A	5Ø953	C7Ø9
C7ØA	5Ø954	Port 2B			
			CTRL Register For Port 2B	5Ø955	C7ØB
C7ØC	5Ø956	Port 3A			
			CTRL Register For Port 3A	5Ø957	C7ØD
C7ØE	5Ø958	Port 3B			
			CTRL Register For Port 3B	5Ø959	C7ØF

Each port A, port B pair is called a Peripheral Interface Adapter or PIA. These ports provide a way to enter data from the outside world into the computer and to respond with computer-generated signals to the outside. The PIA also holds or latches these input and output signals until the computer is ready to receive them (for input) or until the outside devices can utilize them (for output). Each of the two ports on a PIA (port A and port B) contains 8 lines which may be individually used for input or output.

The CA-21 option contains three PIA's. It is connected to the C8P computer by a 16 pin connector, J2, shown in Fig. 1. External devices are connected to the three sets of input port pairs. Since three sets of port A-port B pairs are accommodated (each port 8 bits wide), there are 3*2*8=48 lines available for external connection.

The operating system will initialize the scan of PIA's to include a complete CA-21 option group of PIA's as a default. Scanning fewer PIA's or scanning the PIA at 63232 decimal (F7ØØ hex) will require making the changes (POKEs) just illustrated.

For example, to scan all 48 lines starting at 5Ø948 decimal (C7Ø4 hex), all six data registers (ports 1A, 1B, 2A, 2B, 3A, 3B) must be scanned along with six control registers. Therefore, location 89Ø2 decimal must be loaded with 12−1=11 (the number of scanned registers minus one). These POKEs can be accomplished as

```
POKE 89Ø2,11 : REM LOOK AT ALL 6 DATA AND 6 CONTROL REGISTERS

POKE 89Ø9,4 : REM LOWER HALF OF C7Ø4 PIA PORT ADDRESS

POKE 891Ø,199 : REM SINCE C7 hex=199 decimal
```

(Only decimal values may be used with POKEs.)
With these POKEs, RTMON will check for an active state.

The foregoing has been a review of the connections to the PIA. Now look at the operation of the PIA. The ports (port A and port B) serve two purposes. Each port accomodates input or output signals. Additionally, these port A and port B pairs serve as data direction registers. When serving as a data direction register, the port specifies which bits serve as input and which serve as output bits. The action of the port, whether it serves as an input/output port or as a data direction register, is set by yet another register, called the control register. A control register is associated with each port. If the control register is POKEd with zeros, then the port serves as a data direction register.

When the control register is POKEd with a 4, the port reverts to its data handling function. By using a data port to serve as a data direction register, the number of hardware connections is reduced. But to understand its increased complexity of function requires paying the price of additional work. To illustrate, for example, the use of the PIA to read port 1A at location 5Ø948 (C7Ø4 hex), the steps are

1. POKE 5Ø949,Ø
   This address, one beyond the PIA port 1A address, is the control register for port 1A. A zero in the control register will allow the use of the PIA port 1A address for its alternate use, designating which bits are input or output (called a data direction register). A one indicates output, a zero an input. At the completion of this POKE, the control register contains

   5Ø949 $\boxed{\text{ØØØØ ØØØØ}}$

   and the port 1A will serve as a data direction register. Therefore, the command
2. POKE 5Ø948,127
   will place the bit pattern Ø111 1111 into the data direction register. The data direction register will now be

   5Ø948 $\boxed{\text{Ø111 1111}}$

   Bit 7, the leftmost bit of the data direction register contains a Ø indicating that its corresponding line will be an input line. The other register bits (bits Ø to 6) are 1's, indicating that their corresponding data lines will serve as output lines.
3. The PIA port 1A is now ready to revert to its data handling function. This is achieved by

   POKE 5Ø949,4

   which commands the control register for port 1A to perform its I/O function.
4. Bit 7, the leftmost bit, was previously set as an output bit in step 2. This output can be set to a high value by

   POKE 5Ø948,64

   This is a bit pattern 1ØØØ ØØØØ. The data register (the alternate function of the port) will now contain

   5Ø948 $\boxed{\text{1ØØØ ØØØØ}}$

   Likewise bit 7 could have been set to a zero by

   POKE 5Ø948,Ø
5. If it were desired to read bit 6, which was designated as an input bit, the result could be

   BIT6=PEEK (5Ø948) AND 64

   where 64 has a bit pattern Ø1ØØ ØØØØ. The 1 in the bit pattern corresponds to the desired line. To the user, location 5Ø948 appears as

	7	6	5	4	3	2	1	Ø	bit
5Ø948	X	1 or Ø	X	X	X	X	X	X	

   where X indicates that A doesn't care about the value. By ANDing the contents of 5Ø948 with the value

   Ø 1 Ø Ø Ø Ø Ø Ø

   only the value of bit 6 will be examined. If bit 6 of 5Ø948 is a zero, then BIT6=Ø; if bit 6 is 1, then BIT6=64. Testing for zero or non-zero value of BIT6 provides a convenient programming test to determine the bit 6 input line state.

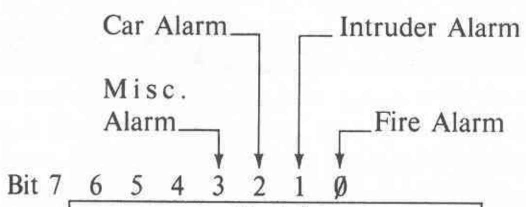
   The socket pin connections are shown in appendix B; socket mating information is also provided.

A short program to make all lines for port 1A into read (input) lines and all lines for port 1B into write (output) lines follows:

```
   5 REM PIA INITIALIZATION SUBROUTINE AT 1ØØØ
  1Ø GOSUB 1ØØØ
  2Ø INPUT "SIDE (A OR B)",C$
  3Ø IF C$="A"GOTO 1ØØ
  4Ø IF C$="B"GOTO 2ØØ
  5Ø GOTO 2Ø
 1ØØ IF A$="I"GOTO 15Ø
 11Ø INPUT "OUTPUT TO A";K
 12Ø POKE X,K
 13Ø GOTO 2Ø
 15Ø PRINT"INPUT TO A IS";PEEK (X)
 16Ø GOTO 2Ø
 2ØØ IF B$="I"GOTO 25Ø
 21Ø INPUT "OUTPUT TO B";K
 22Ø POKE X+2,K
 23Ø GOTO 2Ø
 25Ø PRINT "INPUT TO B IS";PEEK (X+2)
 26Ø GOTO 2Ø
1ØØØ INPUT "STARTING ADDRESS OF PIA";X
1Ø1Ø INPUT "A SIDE I OR O";A$
1Ø2Ø INPUT "B SIDE I OR O";B$
1Ø3Ø POKE X+1,Ø:POKE X+3,Ø : REM SETTING CTRL REGISTER TO ZERO
1Ø4Ø IF A$="I" THEN POKE X,Ø : REM PERMITS SETTING DATA DIRECTION REGISTER
1Ø42 IF A$="I" THEN GOTO 1Ø5Ø
1Ø45 POKE X,255 : REM IF NOT INPUT, THEN SET AS OUTPUT
1Ø5Ø IF B$="I" THEN POKE X+2,Ø
1Ø52 IF B$="I" THEN GOTO 1Ø6Ø
1Ø55 POKE X+2, 255
1Ø6Ø POKE X+1,4:POKE X+3,4 : REM CTRL REGISTER TO FORCE I/O
1Ø7Ø RETURN
```

Multiple lines may be checked at one time.

The home security system addressed at 63232 (F7ØØ hex) is also a PIA port. It is one of two ports. Two ports (of 8 bits each) are available, with the first 4 bits being reserved as:



Location	(Hex)	
63232	F7ØØ	Port A
63233	F7Ø1	CTRL A
63234	F7Ø2	Port B
63235	F7Ø3	CTRL B

A program to handle this device is similar to the previous programs. For example, to check for a fire alarm

```
1Ø REM SET PORT A AS INPUT, LOOK AT BIT Ø, THE FIRE ALARM BIT

2Ø POKE 63233,Ø : POKE 63232,1 : POKE 63233,4

3Ø IF PEEK (63232) = Ø THEN GOTO 1ØØ

4Ø GOTO 2Ø
```

This program segment will continually look at the input port and check for the bit assigned by *OSI* to fire alarm checks.

SECTION 18

CONNECTION OF SIXTEEN PIN BUS DEVICES

Ohio Scientific is pleased to introduce a unique new product line—The 16 Pin I/O BUS. With this system, it is possible to add up to eight special function boards while occupying only one backplane slot.

This is made possible by a novel BUS extension method which allows decoding and timing signals plus eight bits of data to be carried on standard, inexpensive 16 pin ribbon cables.

Up to eight inexpensive 16 pin cables with standard DIP connectors may be attached to a single CA-2Ø board which in turn occupies one slot of the standard Challenger backplane. Alternately, one 16 pin I/O BUS cable may be attached to the A-15 board at the rear of all C1P products equipped with a 63Ø board.

Currently, five HEAD END CARDS are available for interconnection to the system via the CA-2Ø or CA-15 boards.

COMPUTER INTERFACE TO SIXTEEN PIN I/O BUS

The 16 pin I/O BUS may be attached to the computer via two different boards—the CA-15 or the CA-2Ø. The descriptions of these boards are as follows:

CA-15 BOARD

The CA-15 board is a standard accessory interface installed on the following Ohio Scientific systems: C4P-MF, C4P-DMF, and C8P-DF. This is also installed on C1P's equipped with a 63Ø option.

The CA-15 is mounted at the rear of the computer and contains the following interface connection:

Joystick and numeric keypad
Modem and serial printer
Sixteen PIA lines (normally used for the Home Security system—AC-17P)
Sixteen Pin I/O BUS

The interconnect for the Sixteen Pin I/O BUS is simply a 16 pin DIP socket. To use the BUS, the only thing necessary is to attach one end of the 16 pin ribbon cable to the CA-15 board and the other end of the cable to one of the HEAD END CARDS.

Please note that some of the HEAD END CARDS require more power than may be practically carried via the ribbon cable alone. Therefore, some of the cards require auxiliary power supplies.

CA-20 BOARD

The CA-2Ø board contains all the necessary logic to decode eight distinct HEAD END CARD interfaces. The actual interconnect, as with the CA-15, is via simple 16 pin DIP sockets and standard 16 pin ribbon cables.

The CA-2Ø board also requires one slot of the computer's backplane. But remember, from this one slot access is gained to a maximum of eight accessory boards.

The CA-2Ø is recommended for use in the Ohio Scientific C2 series and C3 series computers. It can also be installed in C4P and C8P series systems with some modification to the CA-15 interface. The CA-2Ø can be used in conjunction with C1P computers equipped with the OSI bus expander and 62Ø option.

Since the logic required for the I/O BUS interface is simple, an additional feature was added to the CA-2Ø board—a crystal controlled "time-of-day" clock (hardware) subsystem. The operation of the clock, excepting reading time and setting time, is totally independent of the host computer. As a matter of fact, with the included on-board, auto-recharging, battery back-up, the computer may actually be turned off for several months without losing time.

The features of the clock subsystem are as follows:

Hours, minutes, seconds and 1/1Ø seconds
Day of week

Day of month
Month of year
Four Year calendar

In the C2 and C3 series computers, the CA-2Ø board can actually control the power cycling of the entire computer when equipped with an optional power sequencer package. This means a time (month, day, hour, etc.) may be preset within the clock subsystem and when that preset time agrees with the actual time, A.C. power is applied to the entire computer system through the power sequencer. At a later time, the system's A.C. power may also be removed and the system shut down under software/clock subsystem control.

For applications where the clock subsystem is not required, the CA-2ØA will perform all the Sixteen Pin I/O BUS functions associated with full-feature CA-2Ø.

HEAD END CARDS

HEAD END CARDS is a general name used to describe any or all of the special function boards which attach to the Ohio Scientific Sixteen Pin I/O BUS. There are currently five such boards and, with the exception of the CA-22, they will only interface with the computer via the Sixteen Pin I/O BUS.

Please note, as detailed earlier, a CA-15 or CA-2Ø board must be used at the "computer end" of the Sixteen Pin I/O BUS to complete the interface.

In the following pages a brief product and application description of the currently available HEAD END CARDS will be presented.

THE CA-21 BOARD—BIT SWITCHING AND SENSING

The CA-21 is a 48 line parallel I/O board featuring three 6821 PIAs (peripheral interface adapters) and prototyping/interconnect areas.

The use of PIAs in the design allows for maximum interface versatility as any one of the 48 I/O lines may be configured as either an input or an output. As outputs, each line is capable of driving a minimum of one standard TTL load.

Additional versatility is added because 24 of the lines, when configured as outputs, may simultaneously function as inputs. This feature, although somewhat confusing, is extremely useful for applications such as switch matrix decoding.

Each of the 48 lines is brought out to two foil pads (suitable for wire wrap stakes) as well as a location on one of four 12 pin Molex-type female edge connectors. There are also eight 16 pin DIP socket locations which are intended for use as prototyping areas. Additionally, the 12 PIA "hand-shaking" lines are brought to 12 single foil pads.

The CA-21, with proper buffering, may be used for virtually any computer controlled bit switching or bit sensing application imaginable. With a full complement of eight CA-21s interfaced via the CA-2Ø, a total of 384 individually controllable I/O lines are possible!

An interesting application using one CA-21 board would be a complete, if somewhat slow, emulation of the standard Ohio Scientific BUS.

A more practical application might be augmenting the standard Home Security System (AC-17P) with "hardwired" sensors.

One type of sensor easily added is a standard window "perimeter detector." This could be done with commercially available adhesive foil tape. A break-in (through a broken window) could then be detected by sensing a break in the foil tape.

Another useful application that could be set up in concert with the AC-12P wireless A.C. Remote Control, is sensing when a room is entered. This could be accomplished with pressure-switch door mats or door switches. When room entry is detected, the lights could be turned on or turned off on exit.

For designing any sort of dedicated control system, the CA-21 is an ideal choice. It is possible to easily sense many types of input (pressure transducers, flow sensors, switches, etc.) while controlling outputs from a simple single LED display to a network of solid state relays controlling A.C. power.

THE CA-22 BOARD—ANALOG I/O

The CA-22 is a high speed analog I/O module. Although the CA-22 is classified as a HEAD END CARD, it differs from the rest of the family in that it may also be plugged directly into the computer's standard internal BUS. This allows for maximum flexibility in the use of the CA-22.

The analog input section of the CA-22 consists of a 16 channel analog multiplexer. This means that up to 16 separate signals may be connected directly to the CA-22. Also included is a sample and hold circuit followed by the analog to digital converter circuitry.

The A to D converter is capable of either 8 bit or 12 bit operation. These options are selectable under software control.

The accuracy of the converter is plus or minus one in the least significant bit. The stability of the circuit is rated at one millivolt drift per degree Celsius.

The A to D conversion is extremely fast. It is capable of digitizing up to 66,ØØØ samples per second in the 8 bit conversion mode and 28,ØØØ samples per second in the 12 bit mode. Shannon Sampling Theory states that signals should be sampled at twice the highest frequency present. Therefore, it is possible to convert signals with a frequency greater than 3ØK Hz. Clearly, high fidelity audio is well within the spectrum of the CA-22.

The multiplexer has very high impedance inputs and is capable of accepting inputs in the range of −1Ø volts through +1Ø volts. The input is jumper selectable for other settings including a single sided range of Ø through +1Ø volts.

Due to the indeterminable nature of the actual inputs that may actually be applied to the CA-22, only the multiplexer inputs are brought out. However, a quad op-amp is laid out in foil which may be populated in several different modes to handle some of the more "common" input configurations.

The analog output section of the CA-22 consists of two identical high speed digital to analog converters. Each DAC can convert either 8 bits or 12 bits of data. Data input to the DACs is latched in such a manner that, when in the 8 bit conversion mode, the other four (of the total of twelve) bits are continuously output at a predefined value which may, of course, be defined under software control.

The output of each DAC is buffered with a high speed op-amp capable of changing output voltage at the rate of 2Ø volts per microsecond. The standard configuration of each output is bi-polar with a voltage swing of −1Ø volts through +1Ø volts. This is jumper selectable to allow a uni-polar output of Ø through +1Ø volts.

Some additional I/O capacity is provided on the CA-22. There are three TTL level inputs and six open collector logic outputs. These are strappable to be either standard TTL level outputs or high-voltage outputs.

The CA-22 can be used for a multitude of analog sensing and/or analog controlling applications.

Using the proper transducers and the 16 input channels, it is possible to monitor the temperature in several zones of a home or office. By extending this system with a CA-21, precise temperatures can be maintained by switching the proper controls on and off.

Another interesting, if somewhat obvious application, is in audio processing. Reverberation, phase shifting and echoing are just a few of the uses implementable.

If blocks of RAM were used for data storage, other experiments such as frequency doubling, etc., could be performed.

If more sophisticated software techniques, such as fast Fourier transforms, are applied to store input data, very elaborate signal processing becomes realizable. Projects such as audio spectrum analyzers and speech recognition experiments are certainly practical. Note, in these types of applications, it is likely that some signal pre-processing in hardware is certainly beneficial—if not totally necessary.

Employing both DAC outputs and the on-board unblanking circuit, X-Y oscilloscope plotting is an interesting application. By using these techniques and one or more of the analog inputs, a digital storage scope can be constructed. Note, both of these applications require access to an oscilloscope capable of X-Y input as well as blanking.

THE CA-23 BOARD—EPROM PROGRAMMER

The CA-23 is an EPROM programmer designed for use with the growing families of 5 volt only EPROMS. With the CA-23 you can program and verify all 1K through 8K byte EPROMs of this type. Note that these parts are often identified as 8K—64K bit EPROMS.

The CA-23 can program (or verify) data in two basic modes—EPROM to/from EPROM or EPROM to/from computer RAM memory. Additionally, EPROM data may be read directly into the computer's RAM memory.

There are four LED indicators on the CA-23. The first is "SOCKET UNSAFE." This means that a programming voltage is present at the socket and if an EPROM is removed or inserted it is likely to be damaged.

The second indicator is "PROGRAMMING." This means that the EPROM is currently being programmed.

The third indicator is "ERROR." This means that somewhere along the line a programming attempt was unsuccessful.

The final indicator is "PROGRAM COMPLETE." This means that the program and verification were successful.

The most intriguing application for this product is the creation of "custom" parts for the computer or peripherals. This could range from a new system monitor to a new high level language. It could even include a new character generator for the CRT or printer. Note, however, tinkering around with the internals of computers and peripherals requires a fairly high degree of technical expertise. Also, most manufacturer's warranties are voided by these types of modifications.

Several OEM (original equipment manufacture) and Research/Development applications will be immediately obvious to those involved in that work.

The CA-23, as previously mentioned, is designed for use with 1K through 8K byte EPROMS. These parts come in various package styles and have various product names. For example, Intel's 2K × 8 part is the 2716, Texas Instruments' part is known as the 2516.

The CA-23 has both 24 pin and 28 pin zero insertion force sockets for reading, programming and verifying the EPROMS.

THE CA-24 BOARD—PROTOTYPING

The CA-24 is a solderless bread-board designed for prototyping, experimental and educational applications.

The bread-boarding is made up of seven solderless plug-strips of the type manufactured by AP Products. Two of the plug-strips contain a connection matrix of 5 by 54 connections and are used as signal distribution points. Another pair of 96 location plug-strips are for powering the bread-board area. The actual experimenter area is comprised of three plug-strips, each with a 1Ø by 64 location connection matrix. Additionally, sixteen LED indicators and sixteen DIP switch positions are provided for signal observation and control functions.

Board I/O is via TTL latches and bi-directional PIA ports as well as direct (buffered) data, signal and control lines from the computer BUS. This method allows you direct interconnection of devices such as 685Ø ACIAs in addition to doing more "isolated" and/or independent circuits.

The CA-24 also contains a "clock" generator which is continuously variable from approximately 25,ØØØ Hz. through 7Ø,ØØØ Hz. It is also possible to connect the clock to an on-board 16 stage divider chain. This allows division of the fundamental frequency by as little as 21 (2) to as much as 2^{16}(65,536).

The applications for the CA-24 are primarily prototyping and experimenting. Parts may be inserted and removed from the terminal strip blocks over and over. Interconnection of parts is accomplished simply through the use of solid, narrow gauge wire jumpers. Errors in design or connection are extremely easy to correct.

The CA-24 lends itself very well to structured experiments that are common in the educational environment. It is an ideal tool to aid in the teaching of computer and computer interface fundamentals.

THE CA-25 BOARD—ACCESSORY INTERFACE

The CA-25 is designed to implement some of the functions normally associated with the CA-15 interface board.

It allows direct connection of the Home Security System (AC-17P) and/or the Wireless A.C. Remote Control System (AC-12P) to C2 and C3 series computers. Additionally, those who own an older Ohio Scientific computer can now easily connect these systems to it.

An extremely useful application of the CA-25 is associated with small business systems. Using the CA-25 with the Home Security System, and perhaps a CA-15V (Universal Telephone Interface with speech synthesizer output), the computer could do payroll, inventory, etc. by day and "guard" the shop by night.

SUMMARY

With the introduction of the 16 pin I/O BUS, Ohio Scientific has opened a new world of interfacing capabilities for both the large and the small computer user.

Systems ranging from totally automated sampling and control stations to complete R/D setups to educational lab stations are now available via standard building blocks and standard computer systems.

For pricing and availability, contact the nearest Ohio Scientific dealer.

SECTION 19

ADVANCED FEATURES

With the addition of the 63Ø I/O Expander, the C1P user can generate color graphics on a color monitor or standard color television set. The color monitor or color television is attached to the Challenger 1P in the manner described in section four.

The color option is controlled by one bit of the control register at 55296 (address D8ØØ in hexadecimal). For example

POKE 55296,Ø —disables color, defaults to black and white display

POKE 55296,2 —enables color display.

Appendix 5 gives a complete listing of the values to POKE at 55296 to obtain various combination of options such as DAC sound and color.

Color display is handled in the same manner as the graphics display. The color displayed within a cell on the screen can be set with a POKE to an associated color memory location in the same way that the character displayed within a cell can be set with a POKE to the associated graphics memory location. Figure 17 is a color memory map for the Challenger 1P. The map is for the 24 × 24 display mode. As stated earlier in the manual the 12 × 48 display mode is intended for the display of text. Note that the color address of each cell is offset from the graphics address by 1Ø24. Each color memory location on the Challenger 1P is 4 bits in length. Other memory on the Challenger 1P is 8 bits in length. Thus each color memory location can store a number in the range Ø-15 (decimal). The values correspond to the following 16 different colors:

HEX	DEC	DEC	HEX
D485	544Ø5	54429	D49D
D4A5	54437	54461	D4BD
D4C5	54469	54493	D4DD
D4E5	545Ø1	54525	D4FD
D5Ø5	54533	54557	D51D
D525	54565	54589	D51D
D545	54597	54621	D55D
D565	54629	54653	D57D
D585	54661	54685	D59D
D5A5	54693	54717	D5BD
D5C5	54725	54749	D5DD
D5E5	54757	54781	D5FD
D6Ø5	54789	54813	D61D
D625	54821	54845	D63D
D645	54853	57877	D65D
D665	54885	549Ø9	D67D
D685	54917	54941	D69D
D6A5	54949	54973	D6BD
D6C5	54981	55ØØ5	D6DD
D6E5	55Ø13	55Ø37	D6FD
D7Ø5	55Ø45	55Ø69	D71D
D725	55Ø77	551Ø1	D73D
D745	551Ø9	55133	D75D
D765	55141	55165	D77D

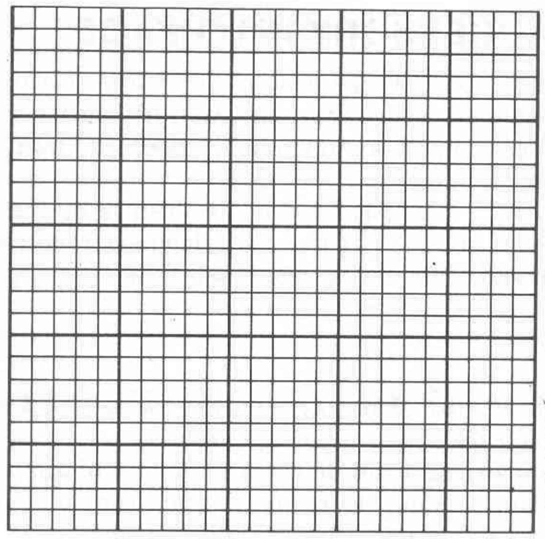


Figure 17: CIP Color Memory Map

DECIMAL VALUE	COLOR
Ø	yellow
1	inverted yellow
2	red
3	inverted red

4	green
5	inverted green
6	olive green
7	inverted olive green
8	blue
9	inverted blue
1Ø	purple
11	inverted purple
12	sky blue
13	inverted sky blue
14	black
15	inverted black (no color).

For instance, to clear the color memory, do the following

```
1Ø POKE 55296,9
2Ø INPUT "NEW COLOR(Ø-15)"; C
3Ø FOR J=543Ø9 TO 55261 : POKE J,C : NEXT
```

The character and the color displayed at a cell on the screen can be controlled with two POKEs. For example, the short program

```
1Ø POKE 53776,239
2Ø POKE 548ØØ,8
3Ø FOR T=1 TO 1ØØØ
4Ø NEXT T
5Ø END
```

will place a small airplane in a blue square near the center of the screen. Statements 3Ø-4Ø were included in the above program as a time delay loop. While the program is executing, the airplane will appear within the colored cell. Once the program is finished executing, the computer will scroll the screen and respond OK. When the screen is scrolled all graphics characters move up on the screen, but the colors remain fixed. On disk based versions of the Challenger 1P operating under OS-65D it is possible to selectively enable and disable scrolling with the following two POKEs:

```
POKE 98ØØ,Ø —disable scrolling
POKE 98ØØ,32 —enable scrolling
```

This capability can be extremely useful for "holding" a graphics display on the screen. There is no convenient way to disable scrolling when BASIC-in-ROM is used. An alternate approach, based upon the keyboard polling techniques of section twelve to hold the display in place, is illustrated by the following program

```
1Ø REM—ENABLE COLOR
2Ø POKE 55296,2
3Ø REM—DISPLAY CHARACTER
4Ø POKE 53776,239
5Ø REM—ASSIGN COLOR
6Ø POKE 548ØØ,8
7Ø REM—WAIT UNTIL
```

```
 8Ø REM—USER PRESSES
 9Ø REM—CARRIAGE RETURN
1ØØ REM—DISABLE CNTL-C
11Ø POKE 53Ø,1
12Ø REM—POLL R5
13Ø POKE 57Ø88,223
14Ø REM—CHECK FOR CR
15Ø K5=PEEK(57Ø88)
16Ø REM—CONTINUE UNTIL
17Ø REM—CR PRESSED
18Ø IF K5<>247 THEN 12Ø
19Ø REM—WHEN PRESSED
2ØØ REM—RESTORE CNTL-C
21Ø POKE 53Ø,Ø
22Ø END
```

This program displays the character and color in the cell and holds the display on the screen without scrolling until the <RETURN> key is depressed.

The use of the built-in DAC to generate sound with the Challenger 1P was discussed in section ten. Although the DAC is capable of generating high quality sound, using the DAC requires sophisticated programming techniques. Moreover, the DAC demands the total attention of the computer when it is used. The 63Ø I/O Expander includes a programmable tone generator. This tone generator allows the C1P user to produce simple tones with a minimal amount of programming in BASIC. The signal from the programmable tone generator is available at the programmable sound output port of the rear panel of the C1P (see figure 1 in section three). The signal from this output port should be fed into the auxiliary input of an audio amplifier or the audio input jack of the video monitor if it has one.

The programmable tone generator is controlled by bit 1 of a special control register located at memory address 63456 (address F7EØ in hexadecimal). For example,

```
POKE 63456,Ø disables the programmable tone generator
POKE 53456,2 enables the programmable tone generator
```

Appendix 5 gives a complete listing of the values to POKE at 63456 to obtain various combinations of options such as simultaneous programmable sound output and AC control. The frequency of the tone generated by the programmable sound generator is determined by the value POKEd into memory location 63424 (address F7CØ in hexadecimal). If the programmable tone generator is enabled and the user enters the statement

```
POKE 63424,N
```

then the frequency of the sound generated is determined by the following formula:

frequency = 49152/N cycles per second.

The value of N should not be set to zero since this will result in division by zero. This equation can be solved to determine N as a function of the desired frequency

N = 49152/frequency.

In order to generate a tone of frequency 44Ø cycles per second, N should be 49152/44Ø = 111.7. This value should be rounded to the nearest integer value (112) before it is POKEd at location 63424 since the POKE statement can only be used to store integer values in the range Ø—255 (decimal).

There is continuous output from the programmable sound generator whenever it is enabled. The tone is constant, changing only when the value stored at 63424 is changed.

The programmable tone generator provides an extremely easy means of generating sound with C1P. Although it does not have the capability of generating the wide variety of sounds possible with the DAC, the sound it produces are suitable for many applications such as sound effects in games.

The following program utilizes the programmable tone generator to play a short tune.

```
 1Ø REM—TWINKLE TWINKLE TUNE
 2Ø REM—TURN ON SOUND GENERATOR
 3Ø POKE 63424,1 : POKE 63456,2
 4Ø REM—READ AND PLAY NOTES
 5Ø FOR T=1 TO 7
 6Ø READ FRQ,COUNT
 7Ø N = INT (49152/FRQ)
 8Ø POKE 63424,N
 9Ø FOR A =1 TO 25Ø*COUNT
1ØØ NEXT A
11Ø FOR D=1 TO 25
12Ø POKE 63424,1
13Ø NEXT D
14Ø NEXT T
15Ø DATA 261. 6,1,261. 6,1
16Ø DATA 392. Ø,1,392. Ø,1
17Ø DATA 44Ø. Ø,1,44Ø. Ø,1
18Ø DATA 392. Ø,2
19Ø POKE 63456,Ø
2ØØ END
```

APPENDIX 1

COMPUTER GLOSSARY

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.

BASIC (Beginners All-Purpose Symbolic Instruction Code) A popular computer language ideally suited for use with Ohio Scientific computers. One of the simplest languages to learn, it can be used for a wide variety of applications.

BAUD A measure of the speed with which information can be communicated between two devices, e.g., if the information is in the form of alphabetic characters, then 300 baud usually corresponds to about 30 characters per second.

BIT (Binary InTeger) The smallest amount of information that can be known. (One or zero.) Eight bits equal one byte.

BUS The means used to transfer information from one part of the computer to another. OSI uses a 48-pin BUS.

BYTE A unit of information composed of 8 bits, which is treated by the computer as a single unit. A byte is usually used to represent an alphanumeric character or a number in the range of Ø to 255.

CASSETTE A medium for the electronic storage of data. Similar to magnetic tape. Most personal computers use ordinary audio-cassette tape recorders and tape.

CENTRAL PROCESSING UNIT (CPU) The part of computer hardware responsible for interpreting data and executing instructions.

COMPUTER An electronic device which is programmable and which processes, operates on and outputs information according to its stored program upon receipt of signals through an I/O device.

COMPUTER LANGUAGE A language that is used for programming a computer, e.g., BASIC.

DAC (Digital-to-Analog Converter) A device that changes digital signals into one continuous analog signal (voltage output).

DATA The information, or set of signals, that is processed by a computer.

DIGITAL Word used to describe information that can be represented by a collection of bits. Modern computers store information in digital form.

DISK A circular piece of rigid material that resembles a record, which has a magnetic coating similar to that found on ordinary recording tape. Digital information can be stored magnetically on a disk.

DISK DRIVE A peripheral which can store information on, and retrieve information from a disk. A "floppy disk drive" can store and retrieve information from a floppy disk.

EPROM (Erasable Programmable Read Only Memory) Information stored in an EPROM IC (Integrated Circuit) 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.

FLOPPY DISK A thin, pliable 8" or 5-1/4" plastic square for storing data. 8" disks store 3, or more, times as much information as 5-1/4" floppies and access the information much faster.

FOREGROUND/BACKGROUND Operation term used to describe the ability of a computer to function with normal programs at the same time it monitors external devices, e.g., home appliances, security, etc.

HARD COPY Information printed on paper or any durable surface, as opposed to information temporarily presented on the CRT screen.

HARDWARE The physical equipment that makes up the computer system.

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.

INPUT Signals given to a computer for processing.

INTERFACE The connection between two systems. A printer interface, for example, connects the printer to the computer.

JOYSTICK Peripheral, accessory equipment that permits the user to move the figures on the monitor. For example, when you and another person play a joystick computer game, you operate joysticks to perform the functions of the game.

K The initial "K" stands for "kilo," meaning 1,000. In computer language, K means 1,024 bytes of information that can be stored in a computer system. A computer with 16K memory, for example, means that the computer has 16 times 1,024, which is 16,384 bytes of memory.

MEMORY The area in the computer for storage of data and instructions.

MICROCOMPUTER A computer based on a microprocessor.

MICROPROCESSOR The "brains" or CPU of a modern personal computer. All Ohio Scientific personal computers use the 65Ø2 microprocessor, generally recognized as the fastest microprocessor available.

MINI-FLOPPY DISK A small 5-1/4" floppy disk that stores about 1/4 the information of an 8" floppy disk.

MODEM Word derived from MOdulator-DEModulator. A device that allows the computer to communicate over telephone lines and other communications media by changing digital information into audio tones (modulating) and from audio tones into digital information (demodulating).

MONITOR A CRT or television screen. You can purchase an Ohio Scientific monitor to hook up to your computer or else simply use an ordinary TV set and attach it with an RF converter.

OS Operating system.

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

PERIPHERAL Any device that can send information to and/or receive information from a computer, e.g., printer, modem, etc.

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

PRINTER A peripheral device which makes hard copy of letters and numerals.

PROGRAM A set of instructions, arranged in a specific sequence, for directing the execution of a specific task, or the solution of a problem, by a computer.

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

RAM (Random Access Memory) A storage device and main memory of any computer, which can be read from and written into. Information and programs are stored in RAM, and they can be retrieved or changed by a program.

ROM (Read Only Memory) A memory storage device in which the information is stored once, usually by the manufacturer, and cannot be changed.

SOFTWARE Programs and operating systems used by the computer; may be on cassette or on disk and in ROM.

APPENDIX 2

BINARY AND HEXADECIMAL NUMBER SYSTEM

THE 6502 ADDRESSING SYSTEM

Numbers in the traditional decimal (base 1Ø) number system are represented as strings of digits selected from the set of ten "decimal digits"

Ø, 1, 2, 3, 4, 5, 6, 7, 8, 9.

The position of each digit within a decimal number is associated with a place value. Thus, the decimal number 2987 can be realized as 2*1Ø.ØØ + 9*1ØØ + 8*1Ø + 7*1. In the decimal number system, the place values (reading from the right to the left) are the consecutive powers of 10 (the base).

$1 = 1Ø^{Ø}$ (this is a mathematical convention)

$1Ø = 1Ø^{1} = 1Ø$

$1ØØ = 1Ø^{2} = 1Ø*1Ø$

$1ØØØ = 1Ø^{3} = 1Ø*1Ø*1Ø$

$1ØØØØ = 1Ø^{4} = 1Ø*1Ø*1Ø*1Ø$ and so forth.

Within a computer, data is more conveniently represented as strings of Ø's and 1's (the "binary digits"), i.e., as numbers in the binary (base 2) number system. In the binary number system place values are the consecutive powers of 2 (the base). Thus, in the binary number system, the place values (reading from the right to the left) are

$1_2 = 1_{1Ø} = 2^{Ø} = 1$

$1Ø_2 = 2_{1Ø} = 2^{1} = 2$

$1ØØ_2 = 4_{1Ø} = 2^{2} = 2*2$

$1ØØØ_2 = 8_{1Ø} = 2^{3} = 2*2*2$

$1ØØØØ_2 = 16_{1Ø} = 2^{4} = 2*2*2*2$ and so forth.

Conversion of a number from the binary number system to the decimal number system is straightforward. Just add up the place values corresponding to the locations of the digit 1 in the number. The binary number 11Ø1Ø1 (binary) can be rewritten in decimal notation as 32 + 16 + 4 + 1 = 53 (decimal).

The MCS65Ø2 microprocessor on the Challenger 1P is designed to process 8 bit (binary digit) data. Each memory location in the C1P is capable of storing 1 BYTE or 8 bits of data. Each BYTE of data can be interpreted as an 8 bit binary number in the range

ØØØØØØØØ – 11111111 (binary) or equivalently

Ø–255 (decimal).

It is easily checked that 11111111 (binary) = 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255 (decimal). In general significantly more binary digits than decimal digits are required to represent a number. The decimal number 1ØØØØØØ requires 2Ø binary digits.

To overcome this difficulty, the hexadecimal (base 16) number system is commonly used instead of the binary number system to describe the contents of memory within a computer. The hexadecimal number system expresses each number as a string of digits selected from a set of 16 "hexadecimal digits." Since the standard set of decimal digits only includes 10 symbols, the characters A, B, C, D, E and F are included to yield a set of 16 hexademical digits. These hexadecimal digits and their binary and decimal equivalents are listed below:

HEX	BINARY	DECIMAL
Ø	ØØØØ	Ø
1	ØØØ1	1
2	ØØ1Ø	2
3	ØØ11	3
4	Ø1ØØ	4
5	Ø1Ø1	5
6	Ø11Ø	6
7	Ø111	7
8	1ØØØ	8
9	1ØØ1	9
A	1Ø1Ø	1Ø
B	1Ø11	11
C	11ØØ	12
D	11Ø1	13
E	111Ø	14
F	1111	15

A single hexadecimal digit is capable of representing any four digit binary number. An 8 digit binary number (BYTE) can be easily converted to a 2 digit hexadecimal number simply by writing down the hexadecimal equivalent for the last of the first four bits of the 8 digit binary number. For example,

```
11Ø11ØØ1 (binary) = D9 (hexadecimal) (or $D9)
   since 1ØØ1 (binary) = $9
     and 11Ø1 (binary) = $D.
```

The conversion of larger binary numbers to hexadecimal is handled in the same manner working from *right* to *left* converting each group of 4 binary digits to its hexadecimal equivalent. For example,

```
Ø1Ø1111Ø1Ø11ØØØ1 (binary) = $5EB1
       since ØØØ1 (binary) = $1
             1Ø11 (binary) = $B
             111Ø (binary) = $E
         and Ø1Ø1 (binary) = $5.
```

Conversely, a hexadecimal number can easily be converted to binary simply by replacing each hexadecimal digit by its binary equivalent. For example,

```
      $9E = 1ØØ1111Ø (binary)
since $E = 111Ø (binary)
  and $9 = 1ØØ1 (binary).
```

The memory addressing scheme on the Challenger 1P is based on the hexadecimal number system. The MCS65Ø2 microprocessor on the C1P addresses memory via a 4 digit hexadecimal address. Thus, the allowable addresses for memory on the Challenger 1P range from

In the hexadecimal number system, the place values (reading from right to left) are the consecutive powers of 16 (the base).

$$\$1 = 1_{1Ø} = 16^{Ø}$$
$$\$1Ø = 16_{1Ø} = 16^1 = 16$$
$$\$1ØØ = 256_{1Ø} = 16^2 = 16*16$$
$$\$1ØØØ = 4Ø96_{1Ø} = 16^3 = 16*16*16$$
$$\$1ØØØØ = 65536_{1Ø} = 16*16*16*16* \text{ and so forth.}$$

Based upon these place values, conversion from hexadecimal to decimal mode is relatively straightforward. For example,

$2A7B = 2*4Ø96 + 1Ø*256 + 7*16 + 11*1 = 1Ø875 (decimal).

Note that we have used the fact that $A = 1Ø (decimal) and $B = 11 (decimal).
The following Appendix Sections include:

1. A BASIC program which will perform hexadecimal to decimal and decimal to hexadecimal conversions for numbers in the range Ø—65535 (decimal).
2. A look up table for quick hexadecimal-decimal conversions.
3. Memory maps for the Challenger 1P in the standard BASIC-in-ROM configuration and for the two disk based configurations: PICO DOS and OS-65D.

The memory maps describe the manner in which the memory is partitioned for different purposes within each configuration.
Each of these memory maps show that BASIC-in-ROM is stored at memory locations $AØØØ—$BFFF or 4Ø96Ø—49151 (decimal). The video display is assigned memory in the region labeled video RAM located at addresses $DØØØ—$D3FF or 53248—54271 (decimal). Compare the video memory maps on pages 44 and 45 in Section Nine.

```
   5 REM THIS PROGRAM CONVERTS NUMBERS (DEC> HEX and HEX>DEC)
  1Ø FORSC=1 TO 3Ø: PRINT: NEXT
  2Ø PRINT"1) CONVERT HEX TO DECIMAL": PRINT
  3Ø PRINT"2) CONVERT DECIMAL TO HEX": PRINT
  4Ø INPUT "WHAT IS YOUR CHOICE (1 OR 2)"; CHOICE
  45 FORSC=1 TO 3Ø: PRINT: NEXT
  5Ø IF CHOICE=1 THEN GOSUB 1ØØØ
  6Ø IF CHOICE=2 THEN GOSUB 2ØØØ
  7Ø GOSUB 3ØØØ
1ØØØ REM HEX TO DECIMAL, CONVERT EACH CHAR. TO ASCII FIRST
1Ø1Ø INPUT "YOUR HEX NUMBER IS"; A$
1Ø2Ø L=LEN(A$) : SUM=Ø
1Ø6Ø FOR K=1TOL
1Ø7Ø M=L+1-K
1Ø8Ø T2=ASC(MID$(A$,M,1))
11ØØ S1=SUM+16↑(K-1)*(T2-55)
111Ø S2=SUM+16↑(K-1)*(T2-48) : REM SHIFT N IS A ↑
113Ø IF T2> 64 THEN SUM=S1 : REM CHECK IF HEX CHAR> 9
114Ø IF T2 <64 THEN SUM=S2 : REM                    OR <9
```

```
115Ø NEXT K
116Ø PRINT "DECIMAL VALUE IS "; SUM
117Ø PRINT: INPUT "DO YOU WANT TO CONVERT ANOTHER HEX NUMBER (Y/N)"; B$
118Ø PRINT: IFLEFT$(B$,1)= "Y" THEN 1ØØØ
119Ø GOTO 5
2ØØØ REM DECIMAL INPUT WITH HEX OUTPUT
2Ø1Ø INPUT"YOUR DECIMAL NUMBER IS "; D
2Ø3Ø T(Ø)=D
2Ø4Ø FORI=1 TO 8
2Ø5Ø T(I)=INT(T(I-1)/16) : CI(I)=T(I-1)-T(I)*16 : K=I
2Ø8Ø IF INT(T(I))=Ø THEN GOTO 22ØØ
2Ø9Ø NEXT I
22ØØ FOR I=1 TO K
221Ø REM REVERSE ORDER OF DIGITS FOR PRINTING
222Ø CH$(K+1-I)=CHR$(48+CI(I))
223Ø IF CI(I)> 9 THEN CH$(K+1-I)=CHR$(55+CI(I))
224Ø NEXT I
225Ø ZIP$=" "
226Ø FORI=1 TO K: ZIP$=ZIP$+CH$(I): NEXTI
229Ø PRINT "THE HEX EQUIVALENT IS "; ZIP$: PRINT
23ØØ INPUT "DO YOU WANT TO CONVERT ANOTHER DECIMAL NUMBER (Y/N)"; C$
231Ø PRINT: IFLEFT$(C$,1)= "Y" THEN 2ØØØ
233Ø GOTO 5
3ØØØ PRINT "YOUR CHOICE SHOULD BE 1 OR 2"
3Ø1Ø PRINT "PLEASE TRY AGAIN": GOTO 5
3Ø3Ø END
```

APPENDIX 3

HEXADECIMAL-DECIMAL CONVERSION

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
010	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
020	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
030	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
040	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
050	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
060	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
070	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
080	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
090	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
0A0	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
0B0	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
0C0	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
0D0	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
0E0	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
0F0	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
100	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271
110	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287
120	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303
130	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319
140	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335
150	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351
160	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367
170	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383
180	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399
190	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415
1A0	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431
1B0	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447
1C0	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463
1D0	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479
1E0	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495
1F0	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511
200	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527
210	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543
220	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559
230	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575
240	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591
250	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607
260	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623
270	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639
280	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655
290	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671
2A0	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687
2B0	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703
2C0	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719
2D0	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735
2E0	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751
2F0	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767
300	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783
310	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799
320	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815
330	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831
340	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847
350	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863
360	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879
370	880	881	882	883	884	885	886	887	888	889	890	891	892	893	894	895
380	896	897	898	899	900	901	902	903	904	905	906	907	908	909	910	911
390	912	913	914	915	916	917	918	919	920	921	922	923	924	925	926	927
3A0	928	929	930	931	932	933	934	935	936	937	938	939	940	941	942	943
3B0	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959
3C0	960	961	962	963	964	965	966	967	968	969	970	971	972	973	974	975
3D0	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991
3E0	992	993	994	995	996	997	998	999	1000	1001	1002	1003	1004	1005	1006	1007
3F0	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023

HEXADECIMAL-DECIMAL CONVERSION

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
400	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
410	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
420	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
430	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
440	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
450	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
460	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
470	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
480	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167
490	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A0	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B0	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C0	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D0	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E0	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263
4F0	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279
500	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
510	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
520	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
530	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
540	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
550	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375
560	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
570	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
580	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
590	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A0	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B0	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C0	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D0	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E0	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F0	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535
600	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
610	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
620	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
630	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
640	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
650	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
660	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
670	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
680	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
690	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A0	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B0	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C0	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D0	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E0	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F0	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791
700	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
710	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
720	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
730	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
740	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
750	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
760	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
770	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
780	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
790	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A0	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B0	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C0	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D0	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E0	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F0	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047
800	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
810	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
820	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
830	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111
840	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
850	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
860	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
870	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
880	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
890	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A0	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B0	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C0	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D0	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E0	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F0	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303

HEXADECIMAL-DECIMAL CONVERSION

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
900	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
910	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
920	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
930	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
940	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
950	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
960	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
970	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
980	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
990	2448	2449	2450	2451	2452	2543	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A0	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B0	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C0	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D0	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E0	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F0	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559
A00	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
A10	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A20	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A30	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A40	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A50	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A60	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671
A70	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687
A80	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A90	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA0	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734	2735
AB0	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC0	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD0	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE0	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF0	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815
B00	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B10	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B20	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B30	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B40	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B50	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B60	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
B70	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
B80	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B90	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA0	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB0	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC0	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD0	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039
BE0	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF0	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071
C00	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C10	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C20	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C30	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C40	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C50	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C60	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C70	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C80	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C90	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA0	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB0	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
CC0	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
CD0	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE0	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF0	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327
D00	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D10	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D20	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D30	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D40	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D50	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D60	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D70	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D80	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D90	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA0	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB0	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519
DC0	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
DD0	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE0	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF0	3568	3569	3570	3571	3572	3753	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583

HEXADECIMAL-DECIMAL CONVERSION

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
E00	3584	3585	3586	3587	3588	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
E10	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E20	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3656	3657	3628	3629	3630	3631
E30	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E40	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E50	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E60	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E70	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
E80	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E90	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA0	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB0	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC0	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED0	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE0	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF0	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837	3838	3839
F00	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F10	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
F20	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F30	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F40	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F50	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F60	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F70	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F80	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F90	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA0	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB0	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC0	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD0	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE0	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	7075	4076	4077	4078	4079
FF0	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095

HEXADECIMAL-DECIMAL INTEGER CONVERSION TABLE

HEXADECIMAL	DECIMAL
01000	4096
02000	8192
03000	12288
04000	16384
05000	20480
06000	24576
07000	28672
08000	32768
09000	36864
0A000	40960
0B000	45056
0C000	49152
0D000	53248
0E000	57344
0F000	61440
10000	65536
11000	69632
12000	73728
13000	77824
14000	81920
15000	86016
16000	90112
17000	94208
18000	98304
19000	102400
1A000	106496
1B000	110592
1C000	114688
1D000	118784
1E000	122880
1F000	126976
20000	131072

APPENDIX 4

MEMORY MAPS

CHALLENGER 1P MEMORY MAP (BASIC-IN-ROM CONFIGURATION)

ØØØØ—ØØFF	Page Zero
Ø1ØØ—Ø1FF	Stack
*Ø13Ø	NMI Vector
*Ø1CØ	IRQ Vector
Ø2ØØ—Ø221	BASIC Flags and Vectors
*Ø2Ø3	LOAD Flag
*Ø2Ø5	SAVE Flag
*Ø218	Input Vector
*Ø21A	Output Vector
*Ø21C	Control C Check Vector
*Ø21E	LOAD Vector
*Ø22Ø	SAVE Vector
Ø222—Ø2FA	Unused
Ø3ØØ end of RAM	BASIC Workspace
AØØØ—BFFF	BASIC-in-ROM
DØØØ—D3FF	Video RAM
DFØØ	Polled Keyboard
FØØØ—FØØ1	ACIA Serial Cassette Port
F8ØØ—FBFF	ROM
FCØØ—FCFF	ROM—Floppy Bootstrap
FDØØ—FDFF	ROM—Polled Keyboard Input Routine
FEØØ—FEFF	ROM—65V Monitor
FFØØ—FFFF	ROM—BASIC Support
*FFFA	NMI Vector
*FFFC	Reset Vector
*FFFE	IRQ Vector

CHALLENGER 1P MEMORY MAP UNDER P-DOS

ØØØØ—ØØFF	Page Zero
Ø1ØØ—Ø1FF	Stack
*Ø13Ø	NMI Vector
*Ø1CØ	IRQ Vector

Ø2ØØ—Ø221	BASIC Flags and Vectors
*Ø2Ø3	LOAD Flag
*Ø2Ø5	SAVE Flag
*Ø218	Input Vector
*Ø21A	Output Vector
*Ø21C	Control C Check Vector
*Ø21E	LOAD Vector
*Ø22Ø	SAVE Vector
Ø222—Ø2FA	Unused
Ø2FB—2ØFF	P-DOS Workspace Pointers
—22FA	BASIC Workspace Under P-DOS (8K)
23ØØ—317D	P-DOS
317E—3FFF	Free
	End of 16K
4ØØØ—7FFF	Free
	End of 32K
AØØØ—BFFF	BASIC-in-ROM
CØØØ—CØØ3	Floppy PIA
CØ1Ø—CØ11	Floppy ACIA
DØØØ—D3FF	Video RAM
DFØØ	Polled Keyboard
FØØØ—FØØ1	ACIA Serial Cassette Port
F8ØØ—FBFF	ROM
FCØØ—FCFF	ROM—Floppy Bootstrap
FDØØ—FEFF	ROM—65V Monitor
FFØØ—FFFF	ROM—BASIC Support
*FFFA	NMI Vector
*FFFC	Reset Vector
*FFFE	IRQ Vector

CHALLENGER 1P MEMORY MAP UNDER 65DV3

ØØØØ—ØØFF	Page Zero
Ø1ØØ—Ø1FF	Stack
*Ø13Ø	NMI Vector
*Ø1CØ	IRQ Vector
Ø2ØØ—Ø221	BASIC Flags and Vectors
Ø2ØØ—22FA	Transient Processor Area Under 65DV3 for 9 Digit BASIC Assembler/ Editor
23ØØ—317D	65DV3 Drivers

317E—3FFF	Free
	End of 16K 65DV3
4ØØØ—7FFF	Free Object Code Workspace
	End of 32K
AØØØ—BFFF	BASIC-in-ROM
CØØØ—CØØ3	Floppy PIA
CØ1Ø—CØ11	Floppy ACIA
DØØØ—D3FF	Video RAM
DFØØ	Polled Keyboard
FØØØ—FØØ1	ACIA Serial Cassette Port
F8ØØ—FBFF	ROM
FCØØ—FCFF	ROM—Floppy Bootstrap
FDØØ—FDFF	ROM—Polled Keyboard Input Routine
FEØØ—FEFF	ROM—65V Monitor
FFØØ—FFFF	ROM—BASIC Support
*FFFA	NMI Vector
*FFFC	Reset Vector
*FFFE	IRQ Vector

APPENDIX 5

CONTROL REGISTERS

Memory location 55296 (address D8ØØ in hexadecimal) is reserved as a control register. This location is "write only," that is the user can POKE values into this location but cannot PEEK to determine the last value stored. This register is used to control the output from the DAC, the display mode on the screen and to enable color (on units equipped with the 63Ø I/O Expander). The following table lists the allowable POKEs at location 55296 and their effects:

VALUE	SCREEN	COLOR	DAC SOUND
Ø	24 × 24	DISABLED	DISABLED
1	12 × 48	DISABLED	DISABLED
2	24 × 24	ENABLED	DISABLED
3	12 × 48	ENABLED	DISABLED
16	24 × 24	DISABLED	ENABLED
17	12 × 48	DISABLED	ENABLED
18	24 × 24	ENABLED	ENABLED
19	12 × 48	ENABLED	ENABLED

When the SWAP program is loaded from cassette in BASIC-in-ROM to allow the use of the 12 × 48 display mode, the above POKEs should be made to memory location 251 instead of 55296.

For models of the C1P equipped with the 63Ø I/O Expander, memory location 63456 (address F7EØ in hexadecimal) is reserved as a control register. The value stored at this location controls the AC control interface, the programmable divider and selects between the printer and modem port. The following table lists the allowable POKEs at location 63456 and their effects:

VALUE	AC CONTROL	TONE GENERATOR	PRINTER/MODEM
Ø	DISABLED	DISABLED	PRINTER
1	ENABLED	DISABLED	PRINTER
2	DISABLED	ENABLED	PRINTER
3	ENABLED	ENABLED	PRINTER
4	DISABLED	DISABLED	MODEM
5	ENABLED	DISABLED	MODEM
6	DISABLED	ENABLED	MODEM
7	ENABLED	ENABLED	MODEM

APPENDIX 6

OS-65D USER'S GUIDE

This section is intended to be used as a quick reference guide only for complete details on OS-65D please refer to the OS-65D User's Manual.

COMMANDS

ASM	Load the assembler and extended monitor. Transfer control to the assembler. (Not present on all disks)
BASIC	Load BASIC and transfer control to it.
CALL NNNN=TT,S	Load contents of track, "TT" sector, "S" to memory location "NNNN".
DIR NN	Print sector map directory of track "NN".
EM	Load the assembler and extended monitor. Transfer control to the extended monitor. (Not present on all disks)
EXAM NNNN=TT	Examine track. Load entire track contents, including formatting information, into location "NNNN".
GO NNNN	Transfer Control <GO> to location "NNNN".
HOME	Reset track count to zero and home the current drive's head to track zero.
INIT	Initialize the entire disk, i.e., erase the entire diskette (except track Ø) and write new formatting information on each track.
INIT TT	Same as "INIT", but only operates on track "TT".
IO NN,MM	Changes the input I/O distributor flag to "NN", and the output flag to "MM".
IO ,MM	Changes only the output flag.
IO NN	Changes only the input flag.
LOAD FILNAM	Loads named source file, "FILNAM" into memory.
LOAD TT	Loads source file into memory given starting track number "TT".
MEM NNNN,MMMM	Sets the memory I/O device input pointer to "NNNN", and the output pointer to "MMMM".
PUT FILNAM	Saves source file in memory on the named disk file "FILNAM."
PUT TT	Saves source file in memory on track "TT", and following tracks.
RET ASM	Restart the assembler. (Not present on all disks)
RET BAS	Restart BASIC.
RET EM	Restart the Extended Monitor. (Not present on all disks)
RET MON	Restart the Prom Monitor (via RST vector).
SAVE TT,S=NNNN/P	Save memory from location "NNNN" on track "TT" sector "S" for "P" pages.

SELECT X	Select disk drive, "X" where "X" can be, A, B, C, or D. Select enables the requested drive and homes the head to track Ø.
XQT FILNAM	Load the file, "FILNAM" as if it were a source file, and transfer control to location $327E.

NOTE:

—Only the first 2 characters are used in recognizing a command. The rest up to the blank are ignored.

—The line input buffer can only hold 18 characters including the return.

—The DOS can be reentered at 9543 ($2547).

—File names must start with an "A" to "Z" and can be only 6 characters long.

—The dictionary is always maintained on disk. This permits the interchange of diskettes.

—The following control keys are valid:

CONTROL—Q	continue output from a CONTROL - S
CONTROL—S	Stop output to the console
CONTROL—U	delete entire line as input
SHIFT—O	delete the last character (polled keyboards)
SHIFT—P	delete entire line as input (polled keyboards)

MEMORY ALLOCATION

ØØØØ—22FF	BASIC or Assembler/Extended Monitor
22ØØ—22FE	Cold start initialization on boot
23ØØ—265B	Input/Output handlers
265C—2A4A	Floppy disk drivers
2A4B—2E78	OS-65D V3.Ø Operating system kernel
2E79—2F78	Directory buffer
2F79—3178	Page Ø/1 swap buffer
3179—3278	DOS extensions
3279—327D	Source file header
327E—	Source File

DISKETTE ALLOCATION

Ø—1	OS-65D V3.(N bootstrap-loads to $22ØØ for 8 pages).
2—6	9-1/2 Digit Microsoft BASIC.
7—9	Assembler-Editor (if present)
10—11	Extended Monitor (if present)
12	Sector 1—Directory, page 1.
	Sector 2—Directory, page 2.
	Sector 3—BASIC overlays.
	Sector 4—GET/PUT overlays.

13	Track∅/Copier utility (loads to $∅2∅∅ for 5 pages).
14—38	User programs and OS-65D utility BASIC programs.
39	Compare routine, on some disks only.

I/O FLAG BIT SETTINGS

INPUT:

Bit ∅—ACIA on CPU board (terminal).

Bit 1—Keyboard on 54∅ board.

Bit 2—UART on 55∅ board.

Bit 3—NULL.

Bit 4—Memory input (auto incrementing).

Bit 5—Memory buffered disk input.

Bit 6—Memory buffered disk input.

Bit 7—55∅ board ACIA input. As selected by index at location $2323 (8995 decimal).

OUTPUT:

Bit ∅—ACIA on CPU board (terminal).

Bit 1—Video output on 54∅ board.

Bit 2—UART on 55∅ board.

Bit 3—Line printer interface.

Bit 4—Memory output (auto incrementing).

Bit 5—Memory buffered disk output.

Bit 6—Memory buffered disk output.

Bit 7—55∅ board ACIA output. As selected by index.

9 DIGIT BASIC EXTENSIONS

INPUT # (DEVICE NUMBER)	(input is set to new device, output is set to null device. If device number > 3, null inputs are ignored.
INPUT "TEXT";# (DEVICE NUMBER)	(print "TEXT" at current output device, then function as above).
PRINT # (DEVICE NUMBER):	(print output for this command at new device).
LIST # (DEVICE NUMBER)	(list program or segments of program to new device).

WHERE (DEVICE NUMBER) FOR OUTPUT IS:

1—ACIA terminal

2—54∅ video terminal

3—55∅ ACIA UART port

4—Line printer

5—Memory output

6—Memory buffered disk output (bit 5).

7—Memory buffered disk output (bit 6).

8—55Ø ACIA output

9—Null output

(DEVICE NUMBER) FOR INPUT IS:

1—ACIA terminal

2—54Ø keyboard

3—55Ø ACIA UART port

4—Null device

5—Memory input

6—Memory buffered disk input (bit 5).

7—Memory buffered disk input (bit 6).

8—55Ø ACIA input

9—Null Input

EXIT	Exit to OS-65D V3. N
RUN (STRING)	Load and run file with name in (STRING).
DISK ! (STRING)	Send (STRING) to OS-65D V3. N as a command line.
DISK OPEN, (DEVICE), (STRING)	Open sequential access disk file with file name, (STRING) using memory buffered disk I/O distributor device number 6 or 7. Reads first track of file to memory and sets up the memory pointers to start of buffer.
DISK CLOSE, (DEVICE)	Forces a disk write of the current buffer contents to current track.
DISK GET, (RECORD NUMBER)	Using last file opened on the LUN (logical unit number) 6 device, a calculated track is read into memory. Where that track is: INT (REC.NUM)/24+base track given in last open command.
DISK PUT	It also sets both memory pointers to: 128*(REC. NUM.) −INT(REC. NUM.)/24))+base buffer address for LUN 6 device. Write device 6 buffer out to disk. The effect is the same as a "DISK CLOSE,6".

EXTENSIONS TO ASSEMBLER (Available As An Option)

For more details refer to the OSI Assembler Editor and Extended Monitor Reference Manual.

E	Exit to OS-65D V3.N
H(HEX NUM)	Set high memory limit to (HEX NUM).
M(HEX NUM)	Set memory offset for A3 assembly to (HEX NUM).
!(CMD LINE)	Send (CMD LINE) to OS-65D V3 as a command to be executed and then return to assembler.

CONTROL-I	Tab 8 spaces. Also: CONTROL-U 7 spaces. CONTROL-Y 6 spaces. CONTROL-T 5 spaces. CONTROL-R 4 spaces. CONTROL-E 3 spaces.
CONTROL-C	Abort current operation.

EXTENDED MONITOR (Available As An Option)

For more details refer to the OSI Assembler Editor and Extended Monitor Reference Manual.

!TEXT	Send "TEXT" to OS-65D V3 as a command.
@NNNN	Open memory location "NNNN" for examination. Subcommands: LF—Open next location. CR—Close location. DD—Place "DD" into location. "—Print ASCII value of location. /—Reopen location. Uparrow—Open previous location.
A	Print AC from breakpoint.
BN,LLLL	Place breakpoint "N" (1-8) at location, "LLLL".
C	Continue from last breakpoint.
DNNNN,MMMM	Dump memory from "NNNN" to "MMMM".
EN	Eliminate breakpoint "N".
EXIT	Exit to OS-65D V3. N
FNNNN,MMMM=DD	Fill memory from "NNNN" to "MMMM"−1 with "DD".
GNNNN	Transfer control to location "NNNN".
HNNNN,MMMM(OP)	Hexadecimal calculator prints result of "NNNN"(OP)"MMMM" where (OP) is + − * / .
I	Print break information for last breakpoint.
K	Print stack pointer from breakpoint.
L	Load memory from cassette.
MNNNN=MMMM,LLLL	Move memory block "MMMM" to "LLLL" −1 to location "NNNN" and up in memory.
NHEX)NNNN,MMMM	Search for string of bytes "HEX" (1-4) between memory location 'NNNN" and "MMMM"-1.
O	Print overflow/remainder from hex calculator.
P	Print processor status word from breakpoint.

QNNNN	Disassemble 23 lines from location "NNNN". A linefeed continues disassembly for 23 more.
RMMMM=NNNN,LLLL	Relocate "NNNN" to "LLLL"−1 to location "MMMM"
SMMMM,NNNN	Save memory block, "MMMM" to "NNNN"−1 on cassette.
T	Print breakpoint table.
V	View contents of cassette.
WTEXT)MMMM,NNNN	Search for ASCII string "TEXT" between "MMMM" and "NNNN"−1
X	Print X index register from last break.
Y	Print Y index register from last break.

NOTE: All commands are line buffered by OS-65D. Thus only 18 characters per line are allowed and CONTROL-U and BACKARROW apply.

SOURCE FILE FORMAT

RELATIVE DISK ADDRESS	MEMORY ADDRESS	USAGE
∅	$3279	Source start (low)
1	$327A	Source start (high)
2	$327B	Source end (low)
3	$327C	Source end (high)
4	$327D	Number of tracks req.
5 and on . . .	$327 and on . .	Source text

DIRECTORY FORMAT

Two sectors (1 and 2) on track 12 hold the directory information. Each entry requires 8 bytes. Thus there are a total of 64 entries between the two sectors. The entries are formatted as follows:

∅-5	ASCII 6 character name of file
6	BCD first track of file
7	BCD last track of file (included in file).

TRACK FORMATTING

The remaining tracks are formatted as follows:

— 1∅ millisecond delay after the index hole

— a 2 byte track start code, $43 $57

— BCD track number

— a track type code, always a $58

There can be any mixture of various length sectors hereafter. The total page count cannot exceed 8 pages if more than one sector is on any given track.

—Each sector is written in the following format:

—previous sector length (4 if none before) times 8∅∅ microseconds of delay

—sector start code, $76

—sector number in binary

—sector length in binary

—sector data

DISKETTE COPIER

The diskette copy utility is found on track 13, sector 1. It should be loaded into location 2ØØ with a "CA Ø2ØØ = 13,1. To start it, type "GØØ2ØØ." To select the copier type a "1." Destination disks must be initialized prior to copying. This is normally used only on computers with two disk drives.

TRACK Ø READ/WRITE UTILITY

This utility permits the reading of data on track Ø anywhere into memory. Also the capability is available to write any block of memory to track Ø specifying a load address and page count. The track zero format is as follows:

—1Ø millisecond delay after the index hole

—the load address of the track in high-low form

—the page count of how much data is on track zero

APPENDIX 7

DOS ERROR MESSAGES

CODE	MEANING
1	Cannot read sector (parity error)
2	Cannot write sector (reread error)
3	Track zero write protected against that operation
4	Disk is write protected
5	See error (track header does not match track)
6	Drive not ready
7	Syntax error in command line
8	Bad track number
9	Cannot find track header within one rev of disk
A	Cannot find sector before one requested
B	Bad sector length value
C	Cannot find name in directory
D	Read/Write attempted past end of named file

BASIC-IN-ROM ERROR CODES

	CODE	DEFINITION
DD	D	Double Dimension: Variable dimensioned twice. Remember subscripted variables default to dimension 10.
FC	F	Function Call error: Parameter passed to function out of range.
ID	I	Illegal Direct: Input or DEFIN statements can not be used in direct mode.
NF	N	NEXT without FOR:
OD	O	Out of Data: More reads than DATA
OM	O	Out of Memory: Program too big or too many GOSUBs, FOR NEXT loops or variables
OV	O	Overflow: Result of calculation too large for BASIC.
SN	S	Snytax error: Typo, etc.
RG	R	RETURN without GOSUB
US	U	Undefined Statement: Attempt to jump to non-existent line number
/Ø	/	Division by Zero
CN	C	Continue errors: attempt to inappropriately continue from BREAK or STOP
LS	L	Long String: String longer than 255 characters
OS	O	Out of String Space: Same as OM
ST	S	String Temporaries: String expression too complex.
TM	T	Type Mismatch: String variable mismatched to numeric variable
UF	U	Undefined Function

DISK BASIC ERROR CODE TABLE

BS	Bad Subscript: Matrix outside DIM statement range, etc.
CN	Continue Errors: Attempt to inappropriately continue from BREAK or STOP.
DD	Double Dimension: Variable dimensioned twice. Remember subscripted variables default to dimension 1Ø.
FC	Function Call Error: Parameter passed to function out of range.
ID	Illegal Direct: INPUT and DEFIN statements cannot be used in direct mode.
LS	Long String: String longer than 255 characters.
NF	NEXT without FOR.
OD	Out of Data: More reads than data.
OM	Out of Memory: Program too big or too many GOSUBs, FOR-NEXT loops or variables.

OS	Out of String Space: Same as OM.
OV	Overflow: Result of calculation too large.
RG	RETURN without GOSUB.
SN	Syntax Error: Typo, etc.
ST	String Temporaries: String expression too complex.
TM	Type Mismatch: String variable mismatched to numeric variable.
UF	Undefined Function.
US	Undefined Statement: Attempt to jump to nonexistent line number.
/Ø	Division by Zero.

APPENDIX 8

FLOPPY DISK CARE

The floppy diskettes and disk drives are delicate pieces of hardware, and should be treated as such. The following rules are strongly recommended to maintain their good condition.

HANDLING FLOPPY DISKETTES

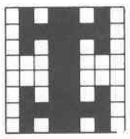
1. Do not touch the surface of the diskette or allow any dirt or dust to come into contact with the surfaces.
2. Be very careful in labeling diskettes, so as not to damage them.
3. Do not bend or fold the diskette.
4. Store the diskette only at temperatures from 1Ø° to 125° F. (−18° to 51° C.) and only use a diskette in a drive if both are at the same temperature.
5. Do not allow magnets to come near the diskette.
6. Always place the diskette in its jacket and store it upright in its box when not in use.
7. If you must lay a diskette on a table, place it with the label side down, to avoid damaging the recording side.
8. When inserting a diskette in a drive, insert it carefully with both hands and an even pressure, until you hear a click. Make sure that it *has not* come back out slightly before you close the drive.
9. Do not try to clean the surface of the diskette.
10. Turn on the power to your computer before you insert the diskette, and turn power off only after you remove the diskette. *Never turn the power on or off while the diskette is in the drive.*
11. Insert the diskette in the disk drive with the label side up.
12. Use only 1ØØ% certified single index hole diskettes, such as the ones which OSI offers.

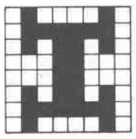
HANDLING DISK DRIVES

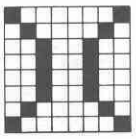
1. The disk drive should only be turned on or off when the computer is already on.
2. Diskettes should be inserted in the drive after the drive has been turned on, and removed before it is turned off.
3. Do not obstruct the air flow in the rear of the disk drive.
4. Disk drives and diskettes will not operate in very high or very low humidity environments. Air conditioning is generally not required unless the unit is operated in a basement, or other area where condensed moisture is likely to occur. RUGS AND CARPETING IN THE VICINITY OF THE COMPUTER SHOULD BE TREATED FOR ANTI-STATIC.
5. The disk drive, being a mechanical rotational device, is susceptible to line voltage and line frequency variations. The unit must be operated at 6Ø Hz for write operations to work.
6. The floppy disk system is mechanical, and thus subject to wear on pulleys, belts, bearings, etc. It is a good practice to remove diskettes from disk drives when disk operations are not anticipated during the next hour or so. Also, turn off disk drives when not in use for prolonged periods of time.

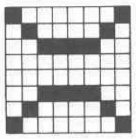
APPENDIX 9

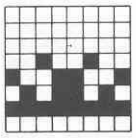
CHARACTER GRAPHICS AND VIDEO SCREEN LAYOUT

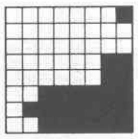

Ø $Ø

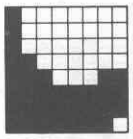

1 $1

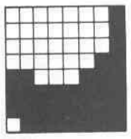

2 $2

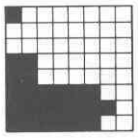

3 $3

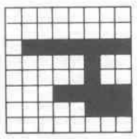

4 $4

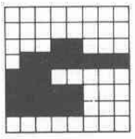

5 $5

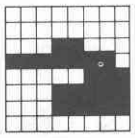

6 $6

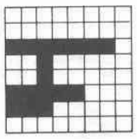

7 $7

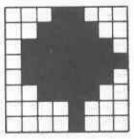

8 $8

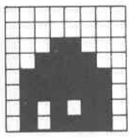

9 $9

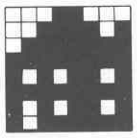

1Ø $A

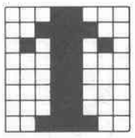

11 $B

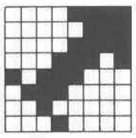

12 $C

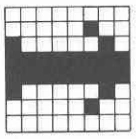

13 $D

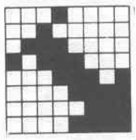

14 $E

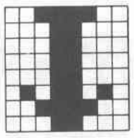

15 $F

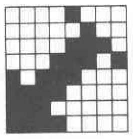

16 $1Ø

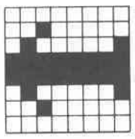

17 $11

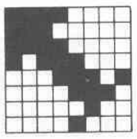

18 $12

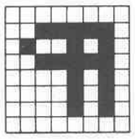

19 $13

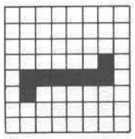

20 $14

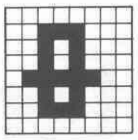

21 $15

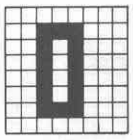

22 $16

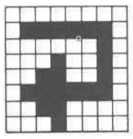

23 $17

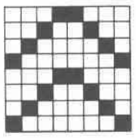

24 $18

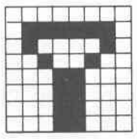

25 $19

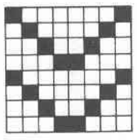

26 $1A

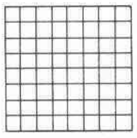

27 $1B

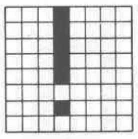

28 $1C

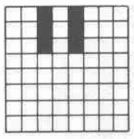

29 $1D

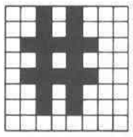

3Ø $1E

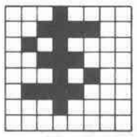

31 $1F

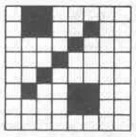

32 $2Ø

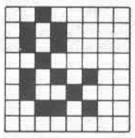

33 $21

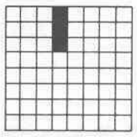

34 $22

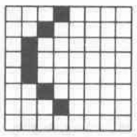

35 $23

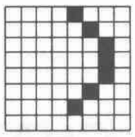

36 $24

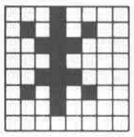

37 $25

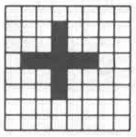

38 $26

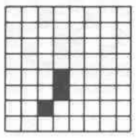

39 $27

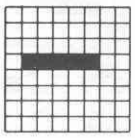

4Ø $28

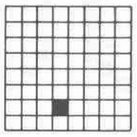

41 $29

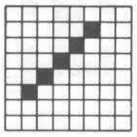

42 $2A

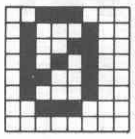

43 $2B

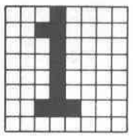

44 $2C

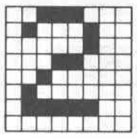

45 $2D

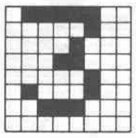

46 $2E

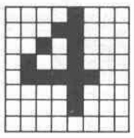

47 $2F

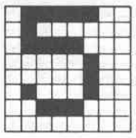

48 $3Ø

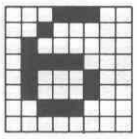

49 $31

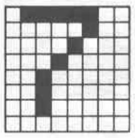

5Ø $32

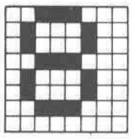

51 $33

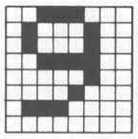

52 $34

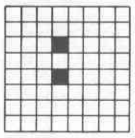

53 $35

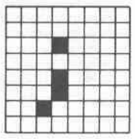

54 $36

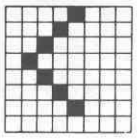

55 $37

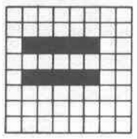

56 $38

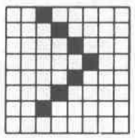

57 $39

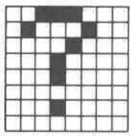

58 $3A

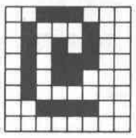

59 $3B

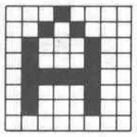

6Ø $3C

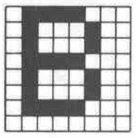

61 $3D

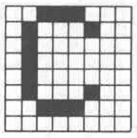

62 $3E

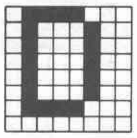

63 $3F

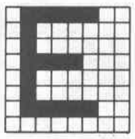

64 $4Ø

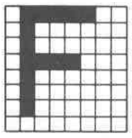

65 $41

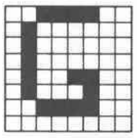

66 $42

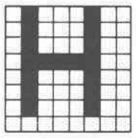

67 $43

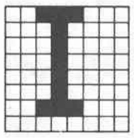

68 $44

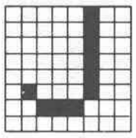

69 $45

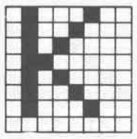

7Ø $46

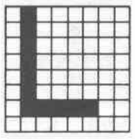

71 $47

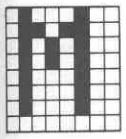
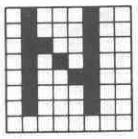
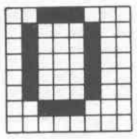
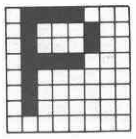
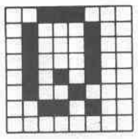
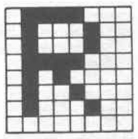
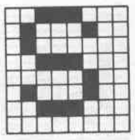
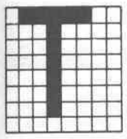
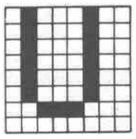
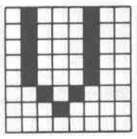
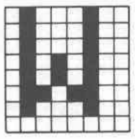
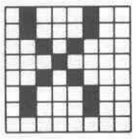
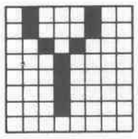
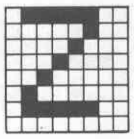
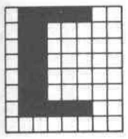
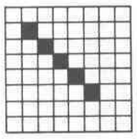
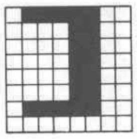
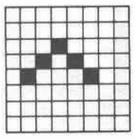
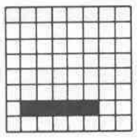
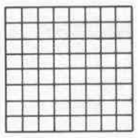
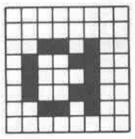
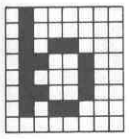
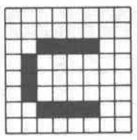
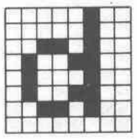
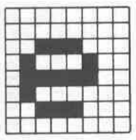
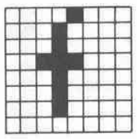
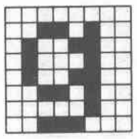
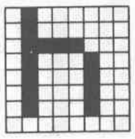
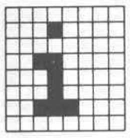
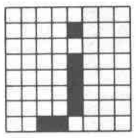
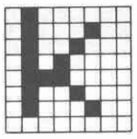
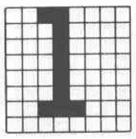
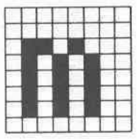
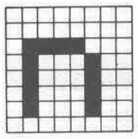
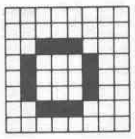
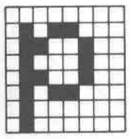
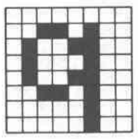
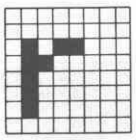
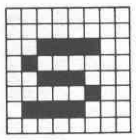
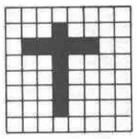
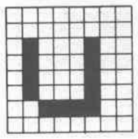
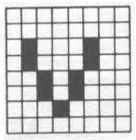

72 $48

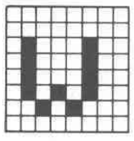

73 $49

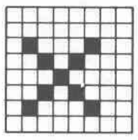

74 $4A

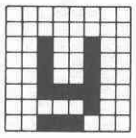

75 $4B

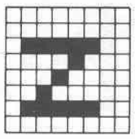

76 $4C

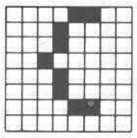

77 $4D

78 $4E

79 $4F

8Ø $5Ø

81 $51

82 $52

83 $53

84 $54

85 $55

86 $56

87 $57

88 $58

89 $59

9Ø $5A

91 $5B

92 $5C

93 $5D

94 $5E

95 $5F

96 $6Ø

97 $61

98 $62

99 $63

1ØØ $64

1Ø1 $65

1Ø2 $66

1Ø3 $67

1Ø4 $68

1Ø5 $69

1Ø6 $6A

1Ø7 $6B

1Ø8 $6C

1Ø9 $6D

11Ø $6E

111 $6F

112 $7Ø

113 $71

114 $72

115 $73

116 $74

117 $75

118 $76

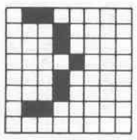

119 $77

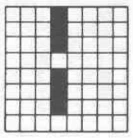

12Ø $78

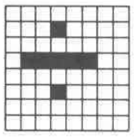

121 $79

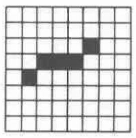

122 $7A

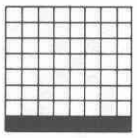

123 $7B

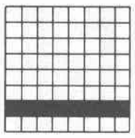

124 $7C

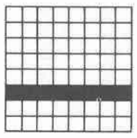

125 $7D

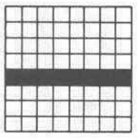

126 $7E

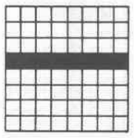

127 $7F

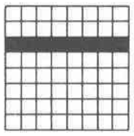

128 $8Ø

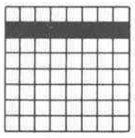

129 $81

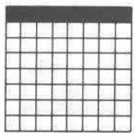

13Ø $82

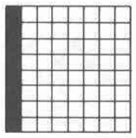

131 $83

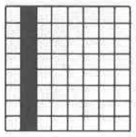

132 $84

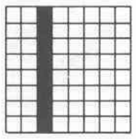

133 $85

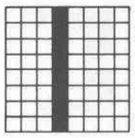

134 $86

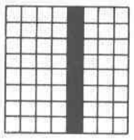

135 $87

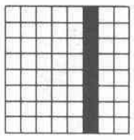

136 $88

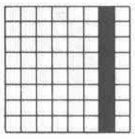

137 $89

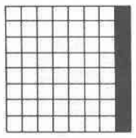

138 $8A

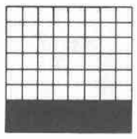

139 $8B

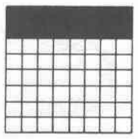

14Ø $8C

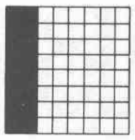

141 $8D

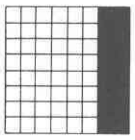

142 $8E

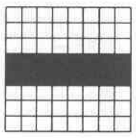

143 $8F

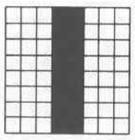

144 $9Ø

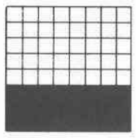

145 $91

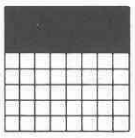

146 $92

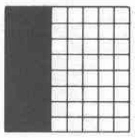

147 $93

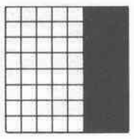

148 $94

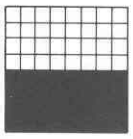

149 · $95

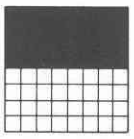

15Ø $96

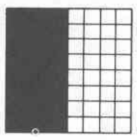

151 $97

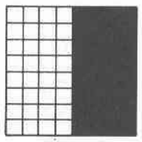

152 $98

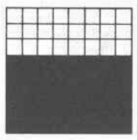

153 $99

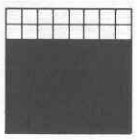

154 $9A

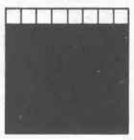

155 $9B

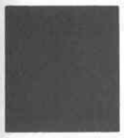

156 $9C

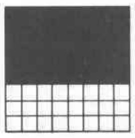

157 $9D

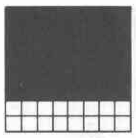

158 $9E

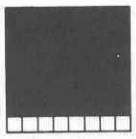

159 $9F

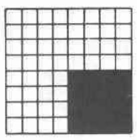

160 $AØ

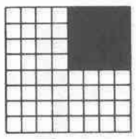

161 $A1

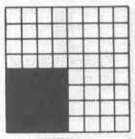

162 $A2

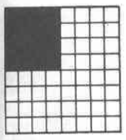

163 $A3

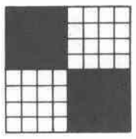

164 $A4

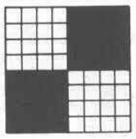

165 $A5

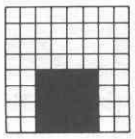

166 $A6

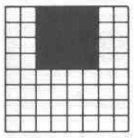

167 $A7

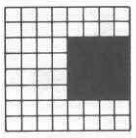

168 $A8

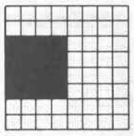

169 $A9

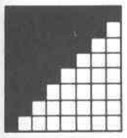

17Ø $AA

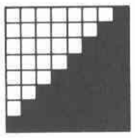

171 $AB

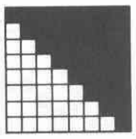

172 $AC

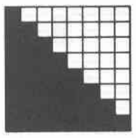

173 $AD

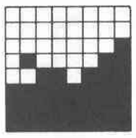

174 $AE

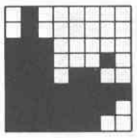

175 $AF

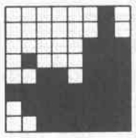

176 $BØ

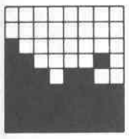

177 $B1

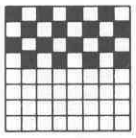

178 $B2

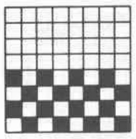

179 $B3

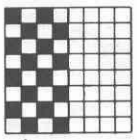

18Ø $B4

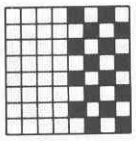

181 $B5

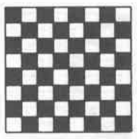

182 $B6

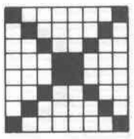

183 $B7

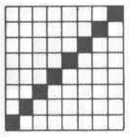

184 $B8

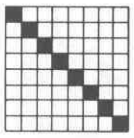

185 $B9

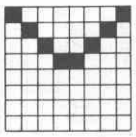

186 $BA

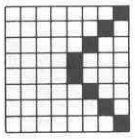

187 $BB

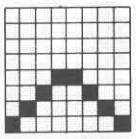

188 $BC

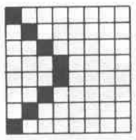

189 $BD

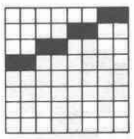

19Ø $BE

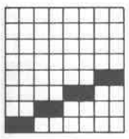

191 $BF

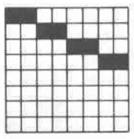

192 $CØ

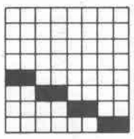

193 $C1

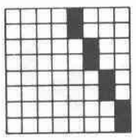

194 $C2

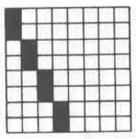

195 $C3

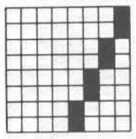

196 $C4

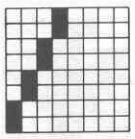

197 $C5

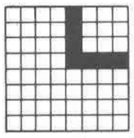

198 $C6

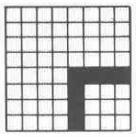

199 $C7

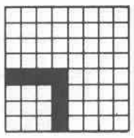

2ØØ $C8

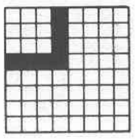

2Ø1 $C9

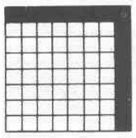

2Ø2 $CA

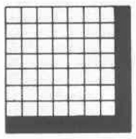

2Ø3 $CB

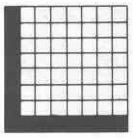

2Ø4 $CC

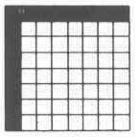

2Ø5 $CD

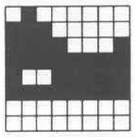

2Ø6 $CE

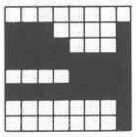

2Ø7 $CF

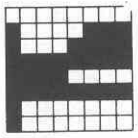

2Ø8 $DØ

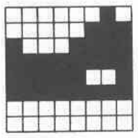

2Ø9 $D1

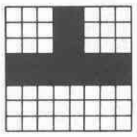

21Ø $D2

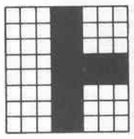

211 $D3

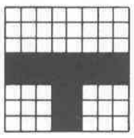

212 $D4

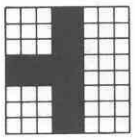

213 $D5

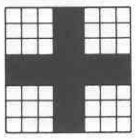

214 $D6

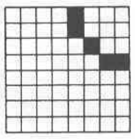

215 $D7

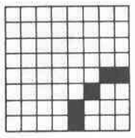

216 $D8

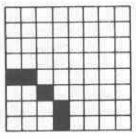

217 $D9

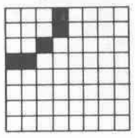

218 $DA

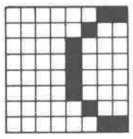

219 $DB

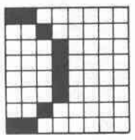

22Ø $DC

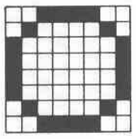

221 $DD

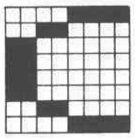

222 $DE

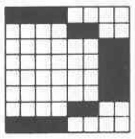

223 $DF

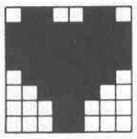

224 $EØ

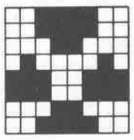

225 $E1

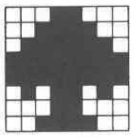

226 $E2

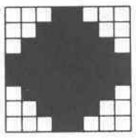

227 $E3

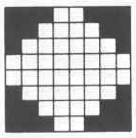

228 $E4

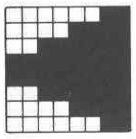

229 $E5

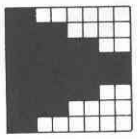

23Ø $E6

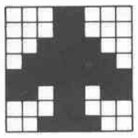

231 $E7

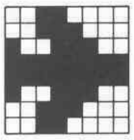

232 $E8

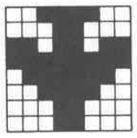

233 $E9

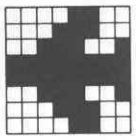

234 $EA

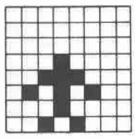

235 $EB

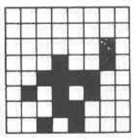

236 $EC

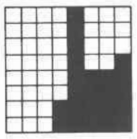

237 $ED

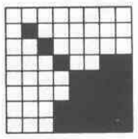

238 $EE

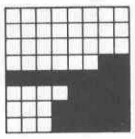

239 $EF

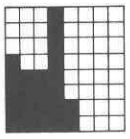

24Ø $FØ

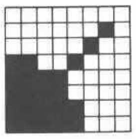

241 $F1

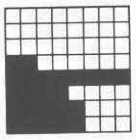

242 $F2

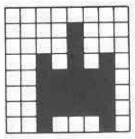

243 $F3

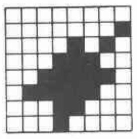

244 $F4

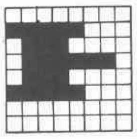

245 $F5

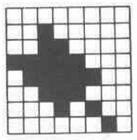

246 $F6

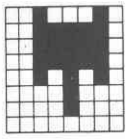

247 $F7

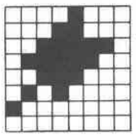

248 $F8

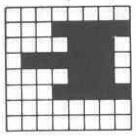

249 $F9

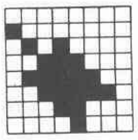

25Ø $FA


251 $FB


252 $FC


253 $FD


254 $FE


255 $FF

APPENDIX 10

POKE LIST-CIP DISK BASIC

As systems develop, different locations are committed to hold parameters. Many of these parameters have been mentioned in the text material. These parameters are collected here, along with some other useful parameters which may be needed by an advanced programmer. Some parameters appear several times, since they are relabeled by other utility programs.

Caution, care must be taken when POKEing any of these locations to avoid system errors and subsequent software losses.

LOCATION		NORMAL	
DECIMAL	HEX	CONTENTS	USE
23	17	132	Terminal width (number of printer characters per line). The default value is 132. Note, this is not to be confused with the video display width (64 characters).
24	18	112	Number of characters in BASIC's 14 character fields (112 characters = 8 fields) when outputting variables separated by commas.
12Ø	78	127	Lo-Hi byte address of the beginning of BASIC work space (note
121	79	50	127=$7F, 5Ø=$32).
132	84	*	Lo-Hi byte address of the end of the BASIC work space. (*con-
133	85	*	tents vary according to memory size such as 255($FF) and 95 ($5F) for $5FFF=24575 for 24K)
222	DE	Ø	Location to enable or disable RTMON (real time monitor). 1 enables and Ø disables RTMON.
223	DF	Ø	Location to start count down timer. A 1 starts the timer, and a Ø stops it.
224	EØ	Ø	Contains the number of hours for timer to count down.
225	E1	Ø	Contains the number of minutes to count down.
226	E2	Ø	Contains the number of seconds to count down.
23Ø-241	E6-F1	Ø	Identifies the I/O masks used for external polling of AC events, i.e. determines which PIA lines are scanned.
249	F9	Ø	Should contain the latest value at 56832 ($DEØØ) which is a "write only" register. This location does not change the display format but acts to maintain the format during ACTL use.
548	224	—	Hi-Lo byte address for AC driver; with no buffers these loca-
549	225	—	tions (with AC enabled) will contain $327F
741	2E5	1Ø	Control location for "LIST". Enable with a 76, disable with a 1Ø.
75Ø	2EE	1Ø	Control location for "NEW". Enable with a 78, disable with a 1Ø.
1797	7Ø5	32	Controls line number listing of BASIC programs, enable with a 32, defeat with a 44.
2Ø73	819	173	"CONTROL C" termination of BASIC programs. Enable with 173, disable with 96.
22ØØ	898	—	The monitor ROM directs Track Ø to load here at $22ØØ.
2888	B48	27	A 27 present here allows any null input (carriage return only) to force into immediate jumping out of the program. Disable this with a Ø. Location 8722 must also be set to Ø.
2893	B4D	55	Alternate "break on null input" enable/disable location. A null
2894	B4E	08	input will produce a "REDO FROM START" message when 2893 and 2894 are POKEd with 28 and 11 respectively.

LOCATION DECIMAL	HEX	NORMAL CONTENTS	USE
2972	B9C	58	Normally a comma is a string input termination. This may be disabled with a 13 (see 2976).
2976	BAØ	44	A colon is also a string input terminator, this is disabled with a 13 (see 2972).
87Ø8	22Ø4	41	Output flag for peripheral devices
8722	2212	27	Null input if=ØØ, normal input if = 27
89Ø2	22C6	ØØ	Determines which registers (less 1) RTMON scans (see the AC control section).
8917	22D5	—	USR(X) Disk Operation Code: Ø—write to Drive A; 3—read from Drive A; 6—write to Drive B; 9—read from Drive B
8954	22FA	—	Location of JSR to a USR function. Preset to JSR $22D4, i.e., set up for USR(X) Disk Operation.
896Ø	23ØØ	—	Has page number of highest RAM location found on OS-65D's cold start boot in. This is the default high memory address for the assembler and BASIC.
8993	2321	—	I/O Distributor INPUT flag
8994	2322	—	I/O Distributor OUTPUT flag
8995	2323	—	Index to current ACIA on 55Ø board. If numbered from 1 to 15 the value POKEd here is 2 times the ACIA number.
8996	2324	—	Location of a random number seed. This location is constantly incremented during keyboard polling.
9ØØØ	2328	7D	Pointer to Disk Buffer
9ØØ1	2329	3E	(Usually $3E7D)
9ØØ2	232A	—	First Track Disk 1
9ØØ3	232B	—	Last Track Disk 1
9ØØ4	232C	—	Current Track in Buffer 1
9ØØ5	232D	—	Buffer 1 Dirty Flag (Clear=Ø)
		Locations 9ØØ6	to 9Ø13 Pertain To Disk 2
9ØØ6	232E	7E	Pointer to Disk 2 Buffer Start.
9ØØ7	232F	3A	This area used for Disk 2 data transfer operations. (Usually $3a7E)
9ØØ8	233Ø	7E	Pointer to Disk 2 Buffer End
9ØØ9	2331	42	(Usually $427E)
9Ø1Ø	2332	—	First Track Disk 2
9Ø11	2333	—	Last Track Disk 2
9Ø12	2334	—	Current Track in Buffer 2
9Ø13	2335	—	Buffer 2 Dirty Flag (Clean=Ø)
9Ø98	238A	—	Pointer to Memory Storage Input (Lo and Hi Byte). Memory is
9Ø99	238B	—	dedicated for use as file.
91Ø5	2391	—	Pointer to Memory Storage Output (Lo and Hi Byte). Use of
91Ø6	2392	—	memory as a file.
9132	23AC	7E	Disk Buffer 1 Input Current Address (Lo and Hi Byte). Default
9133	23AD	32	value is $327E.
9155	23C3	7E	Disk Buffer 1 Output Current Address (Lo and Hi Byte). Default
9156	23C4	32	value is $327E.
9213	23FD	7E	Disk Buffer 2 Input Current Address (Lo and Hi Byte). Default
9214	23FE	3E	value is $3E7E.
9238	2416	7E	Disk Buffer 2 Output Current Address (Lo and Hi Byte). Default
9239	2417	3E	value is $3E7E.
9368	2498	—	Indirect File Input Address (Hi Byte) (Lo=ØØ)

LOCATION DECIMAL	HEX	NORMAL CONTENTS	USE
9392	24BØ	—	I/O Status used by ACTRL.
94Ø3	24BB	—	See AC control section.
948Ø	25Ø8	—	Real Time Clock, Hours
9481	25Ø9	—	Real Time Clock, Minutes
9482	25ØA	—	Real Time Clock, Seconds
9483	25ØB	—	Real Time Clock, Days
9543	2547	—	Contents is hex DOS Entry Point. Under Machine Monitor Load 2547, then "GO".
9554	2552	—	Pointer to Indirect File (Hi Byte only) for output (Lo=ØØ)
9667	25C3	215	When POKEd with N (2Ø7-215) and a LIST command is given, this will move the scroll up 4*(215-N) lines.
9682	25D2	95	Cursor symbol character designation, for video screen.
988Ø	2646	32	Display control parameters. Single Space=64; Double Space=128; Quad Space=255; Two columns=32.
9822	265D	—	Sector for USR(X) on disk
9823	265F	—	Page Count for USR(X) Disk. Read or Write.
9824	266Ø	—	Pointer to memory for USR(X). (Lo and Hi Bytes) USR(X) will re-
9825	2661	—	side in location pointed to.
9826	2662	—	Contains track number for USR(X) on disk
9976	26F8	—	Disable ":" Terminator. See Location 2976 comments.
1Ø95Ø	2AC6	Ø2	Console terminal number. Video terminal is 2.
11511	2CF7	—	Used by Disk Page Ø/1 Swap Used by Random Access File
12Ø42	2FØA	—	Calculation routines to set record size.
12921	3279		Start of work space header.
12922	327A		If contains 32, then have no buffers If contains 3A, then have 1 buffer: If contains 42, then have 2 buffers
12925	327D		Number of tracks to load from disk.
15997	3E7D		Disk 1 Buffer End
15998	3E7E		Disk 2 Buffer Start
19Ø69	4A7D		Disk 2 Buffer End
5Ø944	C7ØØ		OSI BUS PIA
5Ø948	C7Ø4		PIA register's location. See PIA section for use.
to		to	
5Ø959	C7ØE		
53381	DØ85		Video screen memory storage. Video screen memory is 8 bit (1 byte) storage locations (24 × 24 format)
to		to	
54141	D37D		
544Ø5	D485		Video color image storage. Only 4 bits are available for use.
to		to	
55165	D77D		
56832	DEØØ		Screen Format (64 × 32 characters, or 32 × 32), sound, color selected. See video section for POKEs.
57Ø88	DFØØ		Joystick A,B; Also Tone; Also Polled Keyboard location.
57Ø89	DFØ1		D/A Converter Port. (Also frequency divider rate) This location can only be POKEd. See tone generation section.

LOCATION		NORMAL	
DECIMAL	HEX	CONTENTS	USE
63232	F7ØØ		PIA Port address. Home security devices share this location with normal PIA lines.
64512	FCØØ		ACIA Port address. Printer and modem share this location.

INDEX

OHIO SCIENTIFIC

1333 S. Chillicothe Road - Aurora, OH 44202

Phone: (216) 831-5177