SCREEN DUMP, ETC.™

IRQ, INC.

P.O. Box 457

St. Charles, MO 63302

"for Innovative, Reliable, and Quality solutions, look to IRQ"

TABLE OF CONTENTS

INTRODUCTION

SCREEN DUMP, ETC is a utility program designed to work alone or in conjunction with other machine language or BASIC programs. It is an interrupt driven program, which means it is always "running" as a background activity. It is transparent (not noticeable) to the user until it is specifically instructed to do something by pressing a function key - then it becomes very noticeable. Each of the eight function keys (f1, f3, f5, f7 and the same keys shifted which give f2, f4, f6, and f8) performs a specific task.

Two identical diskettes are supplied. Since the diskettes are copy protected, the second is supplied as a backup copy for your convenience. If a diskette becomes damaged or unusable for any reason, it can be returned to IRQ, Inc by the original purchaser for replacement. A five dollar (U.S. funds) fee is charged for each diskette replacement.

There are six files stored on the diskette. The first file, BASIC LOADER, is a BASIC program that defines your system configuration for the actual machine language program. To conserve memory usage, the loader program is broken into three parts, and is stored as three separate files on the diskette. The second part of the loader program, PART II, is automatically loaded and run at the appropriate time. The third file, PRINTERS, is a relative file containing the printer information used by BASIC LOADER. There are three different versions of the machine language program; LOWSDE, MIDSDE, and HISDE. These three programs are essentially identical (they all do the same thing), with the main difference being the location in computer memory where they reside. The appropriate machine language program is automatically loaded by the BASIC LOADER program.

SCREEN DUMP, ETC is easy to use. Simplified using instructions are:

1) LOAD "BASIC LOADER",8 (with dual drives, use drive 0)

2) RUN

3) Answer questions.

If you have problems with any of the questions, refer to Chapter 1. Detailed information on what each function key does is contained in Chapter 2. It is recommended that this chapter be read closely and in its entirety since some of the program capabilities are not obvious from just the screen messages. Chapter 3 discusses limitations of SCREEN DUMP, ETC (naturally this is a short chapter), and Chapter 4 explains the program's error handling procedures.

Throughout this manual, and when using SCREEN DUMP, ETC, a hexadecimal number is designated by a leading dollar sign ($). Otherwise, the number is decimal.

CHAPTER 1 GETTING STARTED

(Coming up with the right answers)

BASIC LOADER asks you several questions pertaining to your specific setup before automatically loading the appropriate machine language program. These questions are necessary since printer and disk drive device numbers, printer commands, and printer interface characteristics can differ greatly from one system to the next. The questions will be discussed in the order they appear on the screen. However, depending on the answers given, several questions may be skipped. For example, if the answer to "TYPE OF PRINTER" is "NONE", then the program won't bother asking specific questions about your printer.

Question 1 - Memory Area to be Used: Three different areas of the Commodore 64 memory can be used to load the machine language program. Each area has its advantages and disadvantages. Many of your programs will be compatible with all three memory areas, while others may be compatible with only one or two memory areas. The three memory areas are: LOW (2048 - 14335), MID (18304 - 32767), and HI (40960 - 53247).

The HI area has the advantage that it uses memory not normally used by BASIC programs. The 38911 BASIC bytes all remain untouched. However, this HI area is very popular with machine language programs, and therefore can't be used with some machine language programs. For example, DOS 5.1, the enhanced Disk Operating System, normally resides in this area, so the HI area can't be used for SCREEN DUMP, ETC when DOS 5.1 is being used.

The LOW area is the beginning of the memory normally used by BASIC. When this area is used for loading SCREEN DUMP, ETC, the beginning of the BASIC area is automatically moved to the end of the machine language program. This protects the machine language program from BASIC, but also reduces the number of BASIC bytes available from 38911 to 26623. This normally has no effect on BASIC programs (unless of course they require more than 26623 bytes), and it leaves the HI area available for other machine language programs. Here again, there are some machine language programs that are designed to use this memory area. With such a program, SCREEN DUMP, ETC can not be loaded into the LOW area.

As you would guess, the MID area lies between the LOW and HI areas. When it is selected, the end of the BASIC area is automatically moved in front of the machine language program. This protects the machine language program from BASIC, but now the number of BASIC bytes available has been reduced to 16255. The memory starting at 32768 ($8000) is not used by SCREEN DUMP, ETC. This is left available for program cartridges, which normally use the memory from 32768 ($8000) to 40960 ($A000). In addition to the reserved memory for program cartridges, the MID memory area requires more memory for SCREEN DUMP, ETC than the LOW and HI areas because a copy of the upper case character ROM is included in the reserved memory area. The ROM copy is needed because the Commodore 64 character ROM is not available to screens located in memory banks 1 and 3, and the message screen used by the MID loaded SCREEN DUMP, ETC is located in bank 1.

The choice of which memory area to use is easy when the required memory areas for other programs are known. Unfortunately, most programs don't state what memory area is used. Therefore, some experimentation is usually required. If no information is available, try the MID memory area first. About the only time the MID area won't work is with relatively large programs. If the MID area causes problems, try the LOW memory area next, and finally the HI area if required. If none of the three loading areas seem to work, don't give up yet. Unless the other program uses almost all the available Commodore 64 memory, which is possible but unlikely, there may be other compatibility problems. See Chapter 3 for solutions to these.

Question 2 - Type of Printer: To use the f2 function key capabilities (screen dump to printer), several commands must be transmitted by SCREEN DUMP, ETC to your printer. Since there is little standardization among printer manufacturers for these commands, the answer to this question defines the proper set of commands for your printer to the machine language program. Many major printer manufacturers are listed in the program, so all that is necessary on your part is to enter the number that corresponds to your printer. The information included in the program for each printer was based on the best available information. Since some manufacturers were more cooperative than others in supplying the necessary information, and since new models from a manufacturer may require different commands, it is possible your printer won't dump properly with the supplied information. If your printer is not listed, or if for some reason the supplied commands don't provide proper operation for your listed printer, you will have to enter the number 17 (Other). The program will later ask you to supply the proper command strings for your printer. Numeric entry 16 (Custom) can be used for your specific system after all the appropriate questions regarding your operation have been answered the first time. See the end of this chapter for information on how to set up the proper Custom information.

In all cases the printer must have DOT ADDRESSABLE (sometimes called BIT IMAGE or something similar) capabilities, and must be able to print either seven or eight vertical dots at one time in the DOT ADDRESSABLE mode. Printers that are not DOT ADDRESSABLE, or only print six or less vertical dots, are not compatible with SCREEN DUMP, ETC.

The program assumes the printer receives its information over the normal Commodore 64 serial bus. Printers connected to other computer ports will not work unless corrections are made for routing the information to the proper port in the correct format. If your setup doesn't work properly with SCREEN DUMP, ETC, send us a letter describing your printer interface with the computer, and we will help you "get on the air".

If your system doesn't include a printer, you will have to enter number 18 (None). In this case the f2 function key program will obviously not work, but all other SCREEN DUMP, ETC capabilities will be available to you.

Question 3 - OPEN Secondary Address for Transparent Mode: If the answer to the previous question was any printer other than Commodore or Alphacom, the program assumes your setup includes a printer interface between the Commodore 64 serial output and the printer.

These interfaces typically include transformation routines that change the Commodore 64 "PETASCII" to standard ASCII that the printer understands. However, when printing with DOT ADDRESSABLE graphics, it is important that no changes be made to any byte transmitted to the printer (this is normally called the transparent mode). Any change would result in an incorrect printed dot pattern. Another feature these interfaces usually contain is the ability to automatically add a line feed command immediately following a carriage return command. This is very useful in many instances, but it must not be used for DOT ADDRESSABLE graphics. If it were, each time a dot pattern that corresponded to the carriage return command was printed, the next printed pattern would always be the one that corresponded to the line feed command. These interface characteristics might be controlled either by software (the OPEN secondary address) or by hardware (the proper positioning of one or more switches). If the interface characteristics are controlled by the OPEN secondary address, it should be entered here. The OPEN secondary address is the third (last) number used in the BASIC command OPEN n1,n2,n3. The program will accept any number from 0 to 255. The default value of 5 is the correct secondary address for the interfaces manufactured by CARDCO. If the interface characteristics are controlled by switch position, you must remember to properly position the switches before dumping a screen to the printer.

NOTE TO TYMAC CONNECTION USERS: The interfaces manufactured by Tymac use a secondary address of 6 for the "transparent" mode. However, even in the transparent mode, the interface can insert a carriage return after it thinks it has sent 80 characters to the printer. This can cause all kinds of problems in double size screen dumps. Sending an ESC "L" command toggles this option off. Therefore, unless you need this option for other uses, you could disable it at power up by typing the following:

```
OPEN 4,4:PRINT#4,CHR$(27)"L":CLOSE 4
```

Question 4 - Added Line Feed by Printer: Just as the interface discussed in the previous question can automatically add a line feed after each carriage return, most printers can also be set up to add a line feed each time it executes a carriage return. This does not cause problems with DOT ADDRESSABLE graphics because the printer knows if the incoming byte is to be used as a carriage return, or as a control for printing certain dots. However, SCREEN DUMP, ETC needs to know if the printer is going to add the line feed or not. If the printer adds a line feed, then SCREEN DUMP, ETC does not send a line feed after it sends a carriage return. Otherwise, a line feed command is sent after each carriage return command by the program. The default answer is "no" since this is the recommended way to set up your printer for maximum versatility in other uses.

Question 5 - Printer Device Number: This is the number the Commodore 64 uses to send information to the printer for f2 (screen dump to printer) operation. It is the second (middle) number in the BASIC OPEN n1,n2,n3 command. The program will accept any number from 1 to 255. The default value is 4, which is the normal value for a printer connected to the serial port.

Question 6 - Disk Drive Device Number: This is the number the Commodore 64 uses to communicate with the disk drive, such as for f4 (screen dump to disk) or f6 (screen load from disk) operation. As above, it is the second number in the OPEN command. The program will accept any number from 1 to 255. The default value is 8, which is the normal value for a disk drive connected to the serial port.

If you did not enter "Other" as your type of printer, the specified machine language program is loaded at this time. Turn to the end of this chapter for information on how to make a customized program that incorporates your answers to the preceeding questions. If you answered "Other", the next ten questions will ask for specific information regarding operation of your printer. All ten questions will not apply to any single printer, but some questions will require an input on your part. Please refer to your printer manual for the proper answers.

Question 7 - Number of Printer Dots. The number of dots refer to the number of wires in the print head that form the dots on the paper in the DOT ADDRESSABLE mode of operation. Each byte transmitted from the computer to the printer will print a certain number of vertical dots. The maximum number of dots that can be printed per byte is the number needed here. It must be either seven or eight. If your printer can only print six or less vertical dots at one time, it is not compatible with SCREEN DUMP, ETC.

Question 8 - Value of Top Dot: This is the number that when transmitted to the printer results in printing only the uppermost of the seven or eight vertical dots. The number will be either 1 or 128 for eight dot printers, and will be either 1 or 64 for seven dot printers.

Question 9 - Bit 7 Set High: This question only applies to seven dot printers. Since each byte contains eight bits, but only seven bits are required to control the seven dots, an extra bit is available. Some seven dot printers require that this extra bit (the Most Significant Bit - MSB) be set high (turned on) to signify that the other seven bits are to be used for dot print control. Turning on this bit seven is the same as adding 128 to the seven bits that are used for controlling dot printing.

The last seven questions relate to command strings that instruct your printer to do certain things. A command string is a series of one or more bytes (each byte is a decimal number from 0 to 255) that is interpreted by the printer as a specific instruction. For most printers all seven command strings are not required. For any question that does not apply to your printer, just press the RETURN key without any other input from the keyboard. Printer command strings are normally sent using the BASIC CHR$(n) function, where n is between 0 and 255. For example, the three byte long command string to set the line feed to 8/72 inch on many printers is CHR$(27)CHR$(65)CHR$(8). To indicate this string to SCREEN DUMP, ETC when the question is asked, your response would be 27 65 8. Note that the only input required is the actual bytes to be transmitted. The CHR$( ) command is not typed.

Each number (byte) must be separated by one or more blank spaces. If you prefer to use hexadecimal numbers, the above string could be typed as $1B $41 $8. In fact, decimal and hexadecimal numbers may be mixed on the same line. Just be sure to immediately precede any hexadecimal number by a dollar sign ($). The command string is entered into the program by pressing the RETURN key after all the bytes have been typed. As a double check, the program then prints (in decimal numbers) the command string as it understands it, and asks for your approval. If the string is correct, press "y". If "n" is pressed, you are then required to retype the string.

Question 10 - Graphics Pitch: The first command string question deals with the spacing between the horizontally printed dots (pitch). To get a good representation on paper of what appears on the screen requires that the spacing between the printed dots (vertical and horizontal) be the same ratio as the spacing between the dots on the screen. Unfortunately, most printers are limited as to print dot spacing options, so some distortion is probably unavoidable. The vertical dot spacing is fixed by the physical separation of the print wires in the print head, and can't be changed. Therefore, the horizontal dot spacing must be adjusted to match the vertical spacing. Actually, the vertical and horizontal spacing on the screen are not equal. The vertical spacing is slightly larger than the horizontal spacing. If you were to write a program to draw a circle on the screen, the result would look more like an egg standing on end. If you then dumped this screen to a printer with equal vertical and horizontal dot spacing, the egg would become a circle. If the printer had greater horizontal spacing (less dense) than vertical spacing, the egg would be laying on its side rather than on end. Ideally, the horizontal print density should be about 1.4 times the vertical print density. Typical horizontal dot spacings available on printers are 60 dots per inch (480 dots per eight inch line) or 72 dots per inch (576 dots per line). With a normal 72 dot per inch vertical spacing, neither of these horizontal spacings will give a true representation of the screen, but the 576 dots per line will come closer than the 480 dots per line. If your printer automatically sets the pitch for graphics, you will obviously have to take what it gives you. If you have a selection of horizontal dot densities, you should select the one that is approximately the same as the vertical dot density or slightly greater. If you're fortunate, and have a large selection of horizontal dot densities, you could experiment until you find the one that looks best to you. The graphics pitch command string is sent (if required) at the beginning of each screen dump (when f2 is pressed). There are five bytes reserved in memory for this command.

Question 11 - Graphics Line Feed: In the normal printing mode, printer line feeds are set so a vertical space exists between the printed lines. Otherwise, all the lines would run together making them difficult to read. However, SCREEN DUMP, ETC requires that there be no extra space added between the lines in order to give a true representation of the screen. Some printers automatically adjust the line feed when DOT ADDRESSABLE graphics is selected. If your printer does not do this automatically, then a command string must be sent to change the line feed. The line feed must be set such that the top dot of the second printed line falls just below the bottom dot of the first

line with no extra space between them. For example, if your printer prints eight vertical dots at one time, and the distance from one dot to the next is 1/72 inch, the proper line feed setting would be 8/72 inch. The program sends the graphics line feed command string (if required) to the printer immediately following the graphics pitch command string. Memory space in the program has been reserved for up to seven bytes for this command.

Question 12 - Graphics ON: This command string tells the printer to begin printing in the DOT ADDRESSABLE mode. Some printers include quite a lot of information in the "ON" command. The command string used by EPSON and some others not only turns on graphics, it also specifies the print density of the following characters, and it specifies the number of characters to be printed in DOT ADDRESSABLE mode (after which the DOT ADDRESSABLE mode is automatically turned off). If your printer falls in this catagory, the print density to be used should be the one that gives approximately the same horizontal and vertical dot spacing as discussed in Question 10. The number of characters to be printed (if required) should always be set to 320. This number corresponds to the 320 dots displayed horizontally on the screen by the Commodore 64 (this is true for both LO-RES screens and HI-RES screens). The program sends the graphics ON command string to the printer at the beginning of each printed line. Memory space in the program has been reserved for up to seven bytes for this command.

Question 13 - Characters to be Repeated: A few printers require that certain bytes, which would normally correspond to the first byte of a control sequence, be sent twice in succession for the printer to know that the byte is really a dot pattern to be printed. Although this is not really a command string, up to three bytes can be entered here (in any order). SCREEN DUMP, ETC will check each graphics byte sent to the printer to see if it is one of these "repeat bytes". If it is, it will be sent twice to the printer.

Question 14 - Carriage Return/Line Feed Replacement: When in the DOT ADDRESSABLE mode, some printers require a special command string to execute a carriage return and a line feed of the proper size for graphics printing. This command string (if required) is sent at the end of each printed line instead of the normal carriage return, which is CHR$(13), and line feed, which is CHR$(10). Memory space has been reserved for up to five bytes for this command.

Question 15 - Graphics OFF: The printers that don't automatically turn off graphics after 320 printed characters will require this command string to turn off graphics after the complete screen has been printed. Three bytes of memory space have been reserved for this command string.

Question 16 - Standard Line Feed: After the screen has been printed, the line feed needs to be reset for normal vertical spacing between lines. The printers that automatically adjust the line feed for graphics printing also usually return everything back to normal automatically when graphics is turned off. Others will require a command string to reset the line feed. The typical line feed value used for normal printing is 1/6 inch. The standard line feed command

string is sent to the printer (if required) after the screen has been printed, and the graphics OFF command string has been issued. Memory space in the program has been reserved for up to seven bytes for this command.

After all the appropriate questions have been answered, the machine language program (LOW, MID, or HI version) will automatically be loaded and run. We're sorry to put you through so much just to get the program started, but as you can see by the questions, each printer seems to do things a little (or a lot) differently. Essentially, a customized machine language program must be written for each individual printer. If you have problems with your printer working properly with SCREEN DUMP, ETC, send us a letter summarizing your printer's command strings, and we'll help you over the rough spots.

CUSTOMIZED LOADER PROGRAM - After the questions have been answered, and the appropriate machine language program is loaded and started, you will notice several BASIC line numbers and DATA statements printed at the top of the screen. These are printed for your convenience so you won't have to go through all the questions each time you load the program. The listed line numbers are actually replacements for lines 9950 through 9955 of the BASIC LOADER program. When these lines are substituted into the program, you can answer the Type of Printer question with the numeric response of 16 (Custom). All the answers you just provided will then automatically be loaded into the program. The easiest way to incorporate the new lines is as follows:

STEP 1 - Make a copy of the screen using SCREEN DUMP, ETC. Although not actually required, this step does two things. First, it verifies that all the proper information is in the program, and your printer really does print the screen; and second, it makes a hard copy of the data statements in case you inadvertantly scroll them off the screen later. To make a copy, just press f2 (SHIFT-f1).

STEP 2 - LOAD"BASIC LOADER",8 (don't RUN)

STEP 3 - Move cursor up to the new line 9950, and press RETURN. Repeat for lines 9951 through 9955

STEP 4 - SAVE"CUSTOM LOADER",8

Now, at each power up you can load CUSTOM LOADER instead of BASIC LOADER. When the Type of Printer question is asked, just enter the number 16 (Custom). No additional questions are asked.

After the appropriate machine language program has been loaded, the SCREEN DUMP, ETC diskette should be removed from the drive, and returned to storage. It won't be needed again until the next power up.

CHAPTER 2 FUNCTION KEY DESCRIPTIONS

(Who does what, and how)

Each of the eight function keys performs a different task, but there are also similarities in their operation. One similarity is that the function key task is not performed until the function key is released. If you are concerned about timing constraints (such as trying to dump a certain screen from a program that rapidly changes screens), you should note that all action stops when the function key is depressed, but the actual task is not started until the key is released. While the key is being held down, the computer is in a "wait" mode.

When any function key is pressed, the computer operation is halted, and the status of the computer is saved. When the function key task is completed, the computer status is restored to the same condition as when the function key was first pressed, and computer operation continues from that point as if nothing happened. This means that any function key can be exercised during execution of a program without disrupting the normal program operation.

Another similarity is that while one function key task is being performed, the function keys are momentarily disabled. The first task must be completed before another function key task can be initiated. The only exception to this is the "HELP KEY", which is discussed next.

f1 - HELP KEY: f1 displays useful information regarding SCREEN DUMP, ETC on the screen. In addition to a listing of what each function key does (in case you forget which key to press to dump a screen to disk, or whatever), the message screen reminds you of the memory requirements for the particular version (LOW, MID, or HI) of the program being used. The message also notes that simultaneously pressing the RUN STOP key and the RESTORE key kills SCREEN DUMP, ETC. This action does not erase the program from memory, but merely turns it off. (NOTE: The RUN STOP - RESTORE combination is disabled when a function key task is being performed). With the program turned off, programs that use the function keys for other purposes may then be executed in their normal manner. See Chapter 3 for a way to use SCREEN DUMP, ETC concurrently with other programs that use one or more function keys. When it is desired to turn SCREEN DUMP, ETC back on, all that is required is a simple SYS command. To make it easy to remember, the programs have been arranged so the proper command is SYS10000, SYS30000, or SYS50000 for the LOW, MID, and HI versions respectively.

The HELP KEY can also be used as an "ESCAPE KEY" to break out of a routine that has been started by one of the other function keys. For example, the f6 key (discussed later) loads a previously saved screen from disk. It also allows you to change the computer's memory location where the screen is to be located. Since the Commodore 64 requires that the screen be located at only certain locations, the f6 routine keeps asking for the new memory location (if a change was desired) until a valid location is entered. If you get confused, and can't remember what the restrictions on screen locations are, you could enter locations for hours before finding a valid address that would be accepted (we're exaggerating a little here, but you get the picture).

To get out of this mess, just tap the f1 key, and the f6 routine will be abandoned. The information (HELP) screen will then be displayed, and you can return to the original program in the normal manner by tapping the space bar.

f2 - SCREEN DUMP TO PRINTER: When f2 is pressed (and released), the first thing that happens is a question appears on the screen asking if you want a standard size or double size dump. The standard size dump provides a one to one correspondance between the dots on the screen and the printed dots. The double size dump prints each screen dot as a cluster of four dots (2 by 2), which doubles both the vertical and horizontal dimensions. If your printer fills more than half the paper with the standard size dump, the double size dump can't be used because it would exceed the paper size limitations. After the screen has been printed, control will go back to the original program, which will continue as if f2 had never been pressed. If you press f2 by mistake, or decide you don't really want to dump the screen, you can "ESCAPE" back to the original program by tapping the f1 key.

The left edge of the printed screen will always coincide with the left margin. Therefore, the printer's left margin should be set for desired positioning of the printed screen before the f2 key is pressed. Also, the right margin must be set so the printed screen will fit between the margins.

There are no restrictions on the screen. It can be LO-RES or HI-RES; it can be standard color, extended color, or multi-color; it can use ROM characters or custom characters; and it can include sprites! SCREEN DUMP, ETC will automatically determine where it is, what it is, and make the appropriate transformations before sending the proper information to the printer. All this with just a couple of keystrokes.

As mentioned previously, the limitations of printer dot spacing may cause the printed screen to appear slightly distorted when compared to the video screen. Read the discussion of Question 10 in Chapter 1 for more details regarding printer distortion. Also, when dumping LO-RES screens that use the character ROM, the printed letters look a little strange. This is due to the way the character ROM shapes the characters. Each vertical line is always composed of two dots side-by-side. The Commodore 64 does this to give a better display on television screens. It works good on TV's, but sure looks funny on paper. To get more normal looking printed letters, you can use a custom character set that doesn't use the side-by-side dot technique.

f3 - BASIC MEMORY ALLOCATION: The existing Commodore 64 memory pointers are displayed when the f3 function key is exercised. The memory locations are listed in both decimal and hexadecimal forms. The amount of unused BASIC memory (BASIC BYTES FREE) is also presented. The free bytes are simply the available memory between the End of the Arrays and the Start of the Strings.

Changes can be made by moving the cursor to the line containing the memory pointer you want changed, and pressing the RETURN key. The program will then ask for the new memory location. After the new location (in either decimal or hexadecimal form) has been entered, the revised memory allocations are printed on the screen. This process can then be repeated until all pointer locations are as desired. Note that the BASIC bytes free can't be changed directly, but by changing either the End of Arrays pointer or the Start of Strings pointer, the

BASIC bytes free value is automatically changed. Pressing RETURN without moving the cursor returns control to the original program.

The f3 key allows you to find the actual memory location in which a BASIC program resides. Also, the amount of memory required for variables, arrays, and strings can be monitored while the program is running.

Another useful function of f3 is saving machine language programs. When the BASIC SAVE command is used, BASIC saves everything from the Start of BASIC pointer to the Start of Variables pointer as a program file. By changing these pointers before using the SAVE command, any area in memory can be saved to tape or disk. This includes the upper memory area normally not accessable to BASIC. For example, DOS 5.1 is a machine language program that, when loaded, normally resides in the memory area 52224 ($CC00) through 53080 ($CF58). By changing the Start of BASIC pointer to $CC00, and the Start of Variables pointer to $CF59, the normal command SAVE "DOS 5.1",8 will put DOS 5.1 on any diskette. After the program is saved, you should again use f3 to return the pointers to their original values.

Some other information you should keep in mind when using f3 includes:

1) The memory location just before the Start of BASIC should always contain a 0 for proper BASIC operation. Use the POKE command to insure this if the Start of BASIC pointer is changed (this is not required for the SAVE command).
2) If the Start of BASIC pointer is changed, the variable and array pointers should also be changed to correspond to the new Start of BASIC location. The BASIC NEW command will take care of this (NEW also automatically does a CLR operation).
3) If the End of BASIC pointer is changed, the string pointer should also be changed. The BASIC CLR command will take care of this (CLR will also "zero out" the variables and arrays).
4) Normal pointer locations when the Commodore 64 is first turned on are Start of BASIC = $0801; Start of Variables, Start of Arrays, and End of Arrays (+1) = $0803; Start of Strings and End of BASIC = $A000.

f4 - SCREEN DUMP TO DISK: The f4 key does much more than simply save the existing screen to disk. It also saves the configuration of the Commodore 64 as it relates to the screen (e.g. screen memory location and screen characteristics such as multicolor or number of sprites activated). Any sprite data blocks and custom character sets used to generate the screen are also saved. When the screen is reloaded (using the f6 key) the Commodore 64 is reconfigured as it was when the screen was saved. This eliminates the need to save custom character sets or sprite data blocks as separate files. SCREEN DUMP, ETC does it all automatically.

Before the screen is saved, you must supply the name it is to be saved under. The name must be fifteen or less characters long. SCREEN DUMP, ETC will automatically make the sixteenth character either a "1" or a "2" as described below. If the screen does not use a custom character set, and is not HI-RES, then all the information is stored on the diskette as one program file. The name of this program file will be the name you supplied, with a "1" added as the sixteenth character. If the

screen is either HI-RES or uses a custom character set, then a second file will also be created and saved. This file will have the same name except the "1" will be replaced with a "2". All this is done automatically by SCREEN DUMP, ETC. All you have to do is provide the fifteen (or less) character name.

For those of you interested in what is actually saved (and later restored), it consists of:

1) LO-RES screen (including sprite data pointers),

2) Color RAM

3) Sprite data blocks (if used)

4) HI-RES screen (if used)

5) Custom character set (if used)

6) Memory locations 0, 1, $D000-$D02E, $DD00, and $DD02.

f5 - SCREEN MEMORY ALLOCATION: Function key f5 lists the various memory locations that contain the information being displayed on the screen. If a certain function is not being utilized by the screen, the word OFF will be displayed instead of a memory location. Just as with f3 (BASIC Memory Allocation), the memory location of any listed parameter can be changed. The only thing changed is the pointers that tell the Commodore 64 where to find the appropriate information. The actual values in RAM are not moved from one memory location to another. A parameter can also be turned off or turned on. There are a couple of exceptions to the "turn off" capability. For one, the LO-RES screen can't be turned off - it is always required. Even when using a HI-RES screen, the LO-RES screen still exists, but it is used for HI-RES color information instead of screen characters. The other possible exception is custom character sets. When the Commodore 64 is displaying a LO-RES screen, and the screen information is located in either Bank 1 ($4000-$7FFF) or Bank 3 ($C000-$FFFF), a custom character set must be turned on. This is because the ROM character set is not available in Bank 1 or Bank 3. Since a LO-RES screen needs a character set of some kind, a custom character set must be supplied in these circumstances. If you try to turn off a custom character set when using a LO-RES screen in Bank 1 or Bank 3, the program will display the message "CAN'T TURN OFF - NO ROM", and will then wait for you to supply a memory address for the custom character set. If you try to turn off the LO-RES screen (in any Bank), the program won't display any message, it will just ignore you! (Some programs tend to get a little "uppity" sometimes.)

If a HI-RES screen is turned off, the resulting LO-RES screen is automatically set up to get its character information from the upper case character ROM when using either Bank 0 or Bank 2. In Bank 1 and 3 (where the character ROM is not available), SCREEN DUMP, ETC assumes a character set location. This location can then be changed if in error.

Any time you enter a new memory address that the program can't recognize as either a decimal or hexadecimal number, it will assume you really meant to turn the parameter off, and will react accordingly.

If you enter a new address that would have no meaning to the Commodore 64, the program will display the message "INVALID ADDRESS", and will wait for you to enter a correct address. Remember, the f1 key can always be used as an "ESCAPE" key to jump out of a routine initiated by any function key.

Some things to keep in mind when using f5 include:

1) LO-RES screens must start at memory locations that are a multiple of 1024 ($0400). There are 64* possible memory locations that can be used for LO-RES screens in the Commodore 64.
2) HI-RES screens must start at memory locations that are a multiple of 8192 ($2000). There are eight* possible memory locations that can be used for HI-RES screens in the Commodore 64.
3) Sprite data blocks must start at memory locations that are multiples of 64 ($0040). There are 1024* possible memory locations that can be used for sprite data blocks in the Commodore 64.
4) Custom character sets must start at memory locations that are multiples of 2048 ($0800). There are 32* possible memory locations that can be used for custom character sets in the Commodore 64.
5) All screen information (screen location, sprite data blocks, and character sets) must be located in the same memory bank. Each bank (Bank 0, 1, 2, and 3) consists of 16384 bytes.

*Some memory areas are used by the Commodore 64 operating system, and shouldn't be used for screen information. For example, memory location 0 is a "valid" starting address for a screen, sprite data, or a custom character set. However, the normal Commodore 64 operating system uses the first 1024 memory locations for its own use. Therefore, extreme caution is required when using this memory area for screen information. Also, memory locations $1000-$1FFF and $9000-$9FFF can't contain screen information because the character generator ROM "image" falls in these areas.

f6 - SCREEN LOAD FROM DISK: The f6 function key is the reverse of the f4 key. A screen that has been saved to disk using the f4 key can be loaded into the computer memory using the f6 key. If desired, the screen or other information, such as sprite data blocks, can be relocated in memory from the location it was saved from (see the f5 discussion for screen memory restrictions of the Commodore 64). The first three "pages" of memory ($0000-$02FF) can't be used to locate screen information. The reason for this is that SCREEN DUMP, ETC automatically saves the information on pages 0, 1, and 2 when any function key (including f6) is pressed. This information is then restored just before control is returned to the interrupted program. If f6 would write any of the screen information in these three pages,

the screen information would be overwritten when the original page 0-2 information was restored.

One thing required when using the f6 key is that a character set, either ROM or RAM, be available for the LO-RES screen location in use when f6 is pressed. If this isn't the case, the message displayed after the information is loaded from disk will not be readable. Normally this won't happen, but under certain circumstances, such as pressing f6 when a HI-RES screen is being displayed from Bank 1 or 3 (where the character ROM is not available), the message screen won't have a character set to use. Once more, remember that the f1 function key can be used as an "ESCAPE" key to terminate a routine initiated by any function key.

NOTE: When loading either a HI-RES screen or a screen using custom characters, be sure to use the full name (up to 15 characters) the screen was saved under. Don't use just the first few characters of the name followed by an asterisk (*). Since there are two files stored on the diskette whose names differ only by the sixteenth character (either a 1 or a 2), the asterisk will cause the Disk Operating System (DOS) to load the wrong file.

f7 - DECIMAL/HEXADECIMAL CONVERSION: The f7 key will convert a decimal number to a hexadecimal number or vice versa. The use of a dollar sign ($) immediately preceeding the number (with no space between the dollar sign and first digit) designates a hexadecimal number. If no dollar sign is present, the number is considered to be decimal. The number to be converted must be in the range 0 - 16777215 ($0 - $FFFFFF), and can not include commas or other punctuation marks.

The number to be converted must be displayed on the screen. The cursor is then positioned over any digit of the number, or any number of "non-digit" characters (any character other than 0 thru F) to the right of the number, and f7 is pressed. SCREEN DUMP, ETC will then replace the displayed number with its converted equivalent. Since the converted number may require more characters than the original number, it is good practice to always have one or more spaces following the number. Otherwise, a character that was not part of the original number could be overwritten.

If SCREEN DUMP, ETC gets confused when trying to convert a number, it will print a question mark (?) where it got confused, and then terminate the routine. For example, if you tried to convert the hexadecimal number $23BC6, but forgot to type the leading dollar sign, SCREEN DUMP, ETC would respond with ?3BC6.

f8 - USER SUPPLIED: With the f8 key you can take advantage of the general SCREEN DUMP, ETC capabilities, such as temporarily interrupting a program in progress, for any machine language program you desire to use. The following steps are all that is required to use the f8 key for your program. In fact, only step 4 is really required. The other three steps are optional, and are required only when it is desired to utilize the specific SCREEN DUMP, ETC capabilities.

Step 1) Add a JSR $02B0 command to the beginning of your program if you wish to use the reserved LO-RES screen area "inside" SCREEN DUMP, ETC for displays. The existing screen when f8 is pressed is then automatically saved for later use.

Step 2) Add a JMP $02C0 command at the end (exit) of your program to put everything back the way it was when f8 was pressed (including the original screen if step 1 was used). In many cases, step 2 can be eliminated, and the f1 key can be used as an "ESCAPE" from your program.

Step 3) If you want the BASIC LOADER (or CUSTOM LOADER) program to automatically load your program when SCREEN DUMP, ETC is loaded into the computer, make the following changes:

a. SAVE your program to the SCREEN DUMP, ETC diskette
b. LOAD "BASIC LOADER", and delete the REM at the beginning of line 1080
c. Replace "PROGRAM NAME HERE" with your program name in line 1080

Step 4) Change line number 1 in the BASIC LOADER program to reflect the starting address of your program. (Scratch the original BASIC LOADER, and then save the modified BASIC LOADER if a permanent change is desired.)

An example program will be used to help illustrate the process. The example we will use is SUPERMON64, which is a monitor program by Jim Butterfield that appeared in the January 1983 issue of COMPUTE! magazine. Unless you have a specific machine language program you want to use with the f8 key, we recommend a machine language monitor program for f8. Even if you're not interested in machine language programming, you will find that having a monitor program at your finger tips will come in very handy to investigate (or change) memory locations in the Commodore 64, SAVE and LOAD machine language programs, and other tasks that a monitor program is designed for. Of course, a machine language programmer will find the f8/monitor program even more useful. SUPERMON64 was chosen as the example because it is a well known monitor program, doesn't require much memory, and yet is powerful enough to perform all the tasks normally required. It is also readily available. Your local User Group probably has it in the Group library. If you don't have access to a local Users Group, you should consider joining one of the larger groups such as the Toronto Pet Users Group (TPUG), which has members throughout North America and beyond. You will find the sharing of information in User Groups to be very benificial.

But now, back to the task at hand. Before you begin, LOAD the LOW version of SCREEN DUMP, ETC. Now for the four steps:

Step 1) LOAD"SUPERMON64" and RUN it. SUPERMON64 automatically locates itself into the top part of BASIC RAM (in this case from $97ED to $9FFF), and then waits for a command. Since it is now in operation, additional steps can be added to the beginning of the program using the A (Assemble) command. Type the following, and then press RETURN:

```
A 97EA JSR $02B0
```

SUPERMON64 enters that command into memory, and then displays an A and the next available address for the next program step. Since this is the only command we need to add, the Assemble mode

of operation is terminated by pressing RETURN without typing anything on the next line. A new program step has now been added to the beginning of SUPERMON64 that switches the screen to the reserved screen inside SCREEN DUMP, ETC.

Step 2) Rather than trying to find the SUPERMON64 exit point, we will take the easy way out, and eliminate step 2. When we want to exit SUPERMON64, and return control to the original program, all that is needed is to press the f1 key to "ESCAPE". (Don't use the normal SUPERMON64 command, X, to exit to BASIC when f8 is used.)

Step 3) Now that we have SUPERMON64 fixed the way we want it, we will use its SAVE command to put it on the diskette. Be sure you have the SCREEN DUMP, ETC diskette in the drive, and then type the following, and press RETURN:

```
S "SUPERMON64",08,97EA,A000
```

We're through with SUPERMON64 now, so to get back to BASIC, type X, and then press RETURN. Now LOAD "BASIC LOADER", and then LIST 1080. Modify line 1080 to read:

```
1080 A%=1:LOAD"SUPERMON64",B%(50),1
```

Step 4) Since BASIC LOADER is already in memory, LIST 1, and then modify line 1 to read:

```
1 F8=$97EA:REM F8 START
```

$97EA is the starting address of the "program" we wrote in step 1. Line 1 needs one more change. The BASIC LOADER program requires that F8 be given in decimal form. To change $97EA to its decimal equivalent, just position the cursor over one of the four digits, and press the f7 key. This replaces the hexadecimal number with its decimal equivalent of 38890. Now press RETURN to enter the modified line into memory. To save the modified program, we will first scratch the original (the following assumes that DOS 5.1 is operational),

```
@S0:BASIC LOADER
```

and then save the modified program,

```
←BASIC LOADER
```

That's it. Now SUPERMON64 is automatically loaded along with the SCREEN DUMP, ETC machine language program. NOTE: When using either the LOW or HI version of SCREEN DUMP, ETC, the End of BASIC pointer should be changed to $97EA to protect SUPERMON64 from BASIC. The MID version of SCREEN DUMP, ETC automatically protects SUPERMON64.

CHAPTER 3 COMPATIBILITY

(Now what's wrong?)

Although SCREEN DUMP, ETC is designed to work with other programs, there are times when the other program and SCREEN DUMP, ETC just can't seem to get along together. There are usually ways to correct these compatibility problems, but some programs out there don't want to share the Commodore 64 with anybody. Fortunately, programs written to specifically exclude other programs are a small minority, so you shouldn't come across them very often.

One of the obvious compatibility problems is both programs trying to reside in the same Commodore 64 memory area. Machine language programs are usually written such that they must be located at a specific memory location to operate. Some BASIC programs also fall into this catagory. Since you don't usually have the option of moving these other programs around in memory, SCREEN DUMP, ETC has to be put in a memory area that is not used by the other program. The three versions of the machine language program (LOW, MID, and HI) give you the latitude to move SCREEN DUMP, ETC to a free memory area.

Another common compatibility problem exists when each program is trying to use one of the function keys to do a different task. Usually, the SCREEN DUMP, ETC function key task is performed normally, and the other program is never told the function key was pressed. In other cases, the SCREEN DUMP, ETC function key task is performed, and then the other program's function key task is performed. If this sharing of function keys is unacceptable, the SCREEN DUMP, ETC tasks can be changed to other keys of your choice. If it is desired to modify SCREEN DUMP, ETC such that keys other than the function keys are used to initiate the various tasks, values in four memory locations must be changed. TABLE 1 lists the memory locations that must be modified, and TABLE 2 lists the values to put in these locations for the various keys. To modify the program, all that is required is to POKE the key value of the key you wish to use into the appropriate memory location. For example, to change from the f1/f2 key to the Up Arrow (↑) key when using the MID version of SCREEN DUMP, ETC, a single command; POKE 29236,54 does the trick. Now the ↑ key acts the same way f1/f2 used to, and f1/f2 is "dead" (unless some other program brings it to life).

TABLE 1 FUNCTION KEY MEMORY LOCATIONS

	SCREEN DUMP, ETC VERSION		
	LOW	MID	HI
f1/f2 key replacement	10867	29236	49716
f3/f4 key replacement	10868	29237	49717
f5/f6 key replacement	10869	29238	49718
f7/f8 key replacement	10870	29239	49719

TABLE 2 COMMODORE 64 KEY VALUES

KEY	VALUE	KEY	VALUE	KEY	VALUE
@	46	T	22	3	8
A	10	U	30	4	11
B	28	V	31	5	16
C	20	W	9	6	19
D	18	X	23	7	24
E	14	Y	25	8	27
F	21	Z	12	9	32
G	26	£	48	:	45
H	29	↑	54	;	50
I	33	←	57	=	53
J	34	SPC	60	CUR U/D	7
K	37	*	49	CUR L/R	2
L	42	+	40	INS/DEL	0
M	36	,	47	CLR/HME	51
N	39	-	43	f1/f2	4
O	38	.	44	f3/f4	5
P	41	/	55	f5/f6	6
Q	62	0	35	f7/f8	3
R	17	1	56	RTN	1
S	13	2	59		

There can sometimes be problems in using SCREEN DUMP, ETC with other interrupt driven programs. Interrupt driven programs, such as SCREEN DUMP, ETC, change the normal starting address the Commodore 64 jumps to 60 times a second when it does its "housekeeping". SCREEN DUMP, ETC changes this address to a routine that checks if a function key is pressed. If no key is pressed, the computer is sent to the interrupt address it was originally set up for when SCREEN DUMP, ETC was first started. This second address could be either the normal Commodore 64 housekeeping routine, or it may be a special routine used by another interrupt driven program. In either case, there are no problems. Everything works as it should. The potential problem comes about when an interrupt driven program is started after SCREEN DUMP, ETC is already operating. If this other program sends the computer to the SCREEN DUMP, ETC interrupt routine after its interrupt routine is finished, then everything works as it should. However, some programs send the computer directly to the normal housekeeping routine. In this case, SCREEN DUMP, ETC is completely by-passed, and is effectively turned off. When this happens, the function keys are all dead, just as if the RUN STOP - RESTORE keys had been pressed. If this situation occurs, just do the appropriate SYS command (SYS10000, SYS30000, or SYS50000) to turn SCREEN DUMP, ETC back on, and everything (including the other program) should work properly.

Some interrupt driven programs go one step further than the situation described above. These programs are written to change the interrupt address to their own routine at each interrupt (60 times a second), and then, after their routine is finished, they go directly to the normal housekeeping routine. SCREEN DUMP, ETC (or any other interrupt driven program) is not compatible with programs of this type. As soon as you turn on SCREEN DUMP, ETC with the SYS command, the other program turns it right back off. We don't know if these programs are just poorly written, or if this is purposely used as some kind of protection technique. In any case, the only recourse is to modify the program so it doesn't change the address at each interrupt. This requires a working knowledge of machine language, and each program must be evaluated individually.

A program, when run, may set up the Commodore 64 to a particular configuration. If this initial configuration setup turns off SCREEN DUMP, ETC, you may not be able to regain computer control to issue the appropriate SYS command to turn SCREEN DUMP, ETC back on. In cases like this, the SYS command must be added to a program line so the program itself turns SCREEN DUMP, ETC back on after the initial configuration setup. The Commodore Educational Software programs fall in this catagory. With these programs, a good place to add the SYS command is in one of the lines that print the Title Page. For example, if the MID version of SCREEN DUMP, ETC is being used, adding SYS30000: just before a PRINT command will do the job.

In some cases, the screen you see on the video display isn't what you see on the printer. One reason for this is the loss of color information when the screen is printed. For example, a video display may contain a white, fluffy cloud with a little, red airplane flying in front of the cloud. This looks great on the screen, but when it is printed, the white, fluffy cloud becomes a black, fluffy cloud. This

isn't too bad, but the little, red airplane turns into a little, black airplane. Again, this wouldn't be too bad by itself, but when the black airplane is flying in front of the black cloud, it essentially disappears. The black plane is really there on paper, but you can't see it because it blends in perfectly with the black cloud. Keep this in mind when working with screens that make use of different colors.

A special type of interrupt program is one that uses "raster interrupt" techniques to actually switch between two or more screens, or change screen information such as sprite locations, while the video display is being formed on the TV or video monitor. This technique is typically used to achieve split screens that are part LO-RES and part HI-RES. Since the screen you see is actually a combination of two or more different screens, these split screens can't be saved to disk. However, SCREEN DUMP, ETC is designed to accurately dump these screens to your printer. The only problem you may encounter is that the message screen asking if you want a standard size (S) or double size (D) dump may not be readable because the raster interrupt program is forcing the Commodore 64 to display a screen other than the normal message screen. If this happens, just pretend you can read the message, and type either S or D. SCREEN DUMP, ETC will then dump the split screen to the printer, and will check each eight dot row (total of 25 rows) to see how that particular row is being generated.

CHAPTER 4 ERRORS

(Getting back on track)

SCREEN DUMP, ETC incorporates extensive error checking routines, and allows you to maintain control over the computer when an error is encountered. In general, when an obvious error is encountered, the program tells you about the error, and gives you the opportunity to recover from the error without being reset by the computer.

Some of the error handling capabilities (such as ensuring that a screen's memory location is compatible with the Commodore 64) have been discussed in Chapter 2. In addition, there is a special Input/Output (I/O) error handling routine. If an I/O error is observed, such as Device Not Present, the function key task is abandoned, and a message describing the error is displayed on the screen. You can then return to the interrupted program, correct the I/O problem, and try again. To see how this works, try saving a screen to the disk (f4) with the disk drive door open. As soon as the Commodore 64 tries to talk to the disk drive, an error message will appear telling you the device is not present. You can then close the door, and tap the SPACE BAR to return to the interrupted program. At that point you could press f4 again to save the screen to disk.

Due to the diskette copy protection, you cannot VALIDATE the SCREEN DUMP, ETC diskettes. Therefore, we recommend you only write to the diskette when necessary (such as saving or modifying BASIC LOADER or CUSTOM LOADER, or saving your f8 key machine language program). Don't use the diskettes for general storage of other programs. This will minimize the possibility of ever getting an "asterisked" file on the diskette. If an asterisked file, a file whose directory listing has an asterisk (*) in front of the file type (e.g. *PRG), ever does show up when listing the directory, DON'T SCRATCH THE FILE! Since the VALIDATE command, which is the only safe way to get rid of an asterisked file, can't be used, just leave the file on the diskette. You can still read the other files from the diskette, but you shouldn't try writing (SAVING) to the diskette anymore.

SATISFACTION GUARANTEED

If you are not entirely satisfied with SCREEN DUMP, ETC, return diskettes and manual within fifteen (15) days for full refund of your purchase price. No other warranty, express or implied, exists for this product. It is the purchaser's responsibility to determine the suitability of this product for any purpose.

SCREEN DUMP, ETC

compatibility with

SIMONS' BASIC

SCREEN DUMP, ETC capabilities complement those of SIMONS' BASIC very well. For example, SIMONS' BASIC commands simplify the generation of HI-RES screens, but there is no command available to save your HI-RES creations to diskette. SCREEN DUMP, ETC solves that problem. SIMONS' BASIC commands also simplify the generation of Sprites, but the screen dump routines of SIMONS' BASIC ignore any Sprites displayed, and don't print them. SCREEN DUMP, ETC also solves that problem.

Unfortunately, SIMONS' BASIC is written to "constantly" change the interrupt address of the Commodore 64 to always point to a location in SIMONS' BASIC such that interrupt driven programs, like SCREEN DUMP, ETC, will not work with SIMONS' BASIC. Luckily, the BASIC warm start vector at memory locations 770 and 771 ($0302 and $0303) is used to jump to the routine that changes the interrupt address. A simple POKE command,

```
POKE 770,151
```

is all that is needed to skip the portion of the routine that changes the interrupt address. By using this POKE command before loading SCREEN DUMP, ETC you should have no difficulties in using SCREEN DUMP, ETC with SIMONS' BASIC. (You must use either the MID or LOW version of SCREEN DUMP, ETC. The HI version can't be used because SIMONS' BASIC uses the upper portions of the Commodore 64 memory).

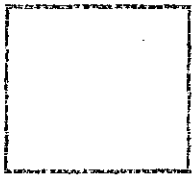
"for Innovative, Reliable, and Quality solutions, look to IRQ"

IRQ, Inc.

P.O. Box 457
St. Charles, Missouri 63302

Thanks for purchasing SCREEN DUMP, ETC. The enclosed printer dump examples will give you some idea of what you can expect when using SCREEN DUMP, ETC with your printer. All examples were made using an Epson LX-80 printer, which has multiple choices of horizontal dot densities in the DOT ADDRESSABLE mode. The example dumps using different horizontal densities include a "square" that illustrates the distortion that occurs with the different densities. The LX-80 has a specified print speed of 100 characters per second (cps). Dump times are included with some of the examples, but your printer dump times could differ because of differences in printer speed, and whether seven or eight vertical dots can be printed simultaneously. The LX-80 used was connected to a Commodore 64 through a Cardco "A" (original model) interface operating in its transparent mode. Any interface that includes a transparent mode (no changes to transmitted data) can be used with non-Commodore printers.

Printer dumps are also included of typical "message screens" displayed when f1 (HELP), f3 (BASIC MEMORY), and f5 (SCREEN MEMORY) keys are pressed.



← THIS IS DISPLAYED AS A SQUARE ON THE VIDEO SCREEN.

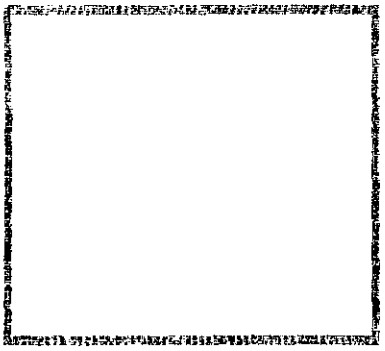
IF THE HORIZONTAL DOT DENSITY IS TOO DENSE, THE ABOVE SQUARE IS "SKINNY" WHEN PRINTED.

FOR DENSITIES TOO SMALL, THE SQUARE IS "FAT" (STRETCHED OUT) WHEN PRINTED.

Horiz. Density = 90 dots per inch (recent Epsons, such as FX, RX, LX, and others)

Standard size dump time = 40 sec.

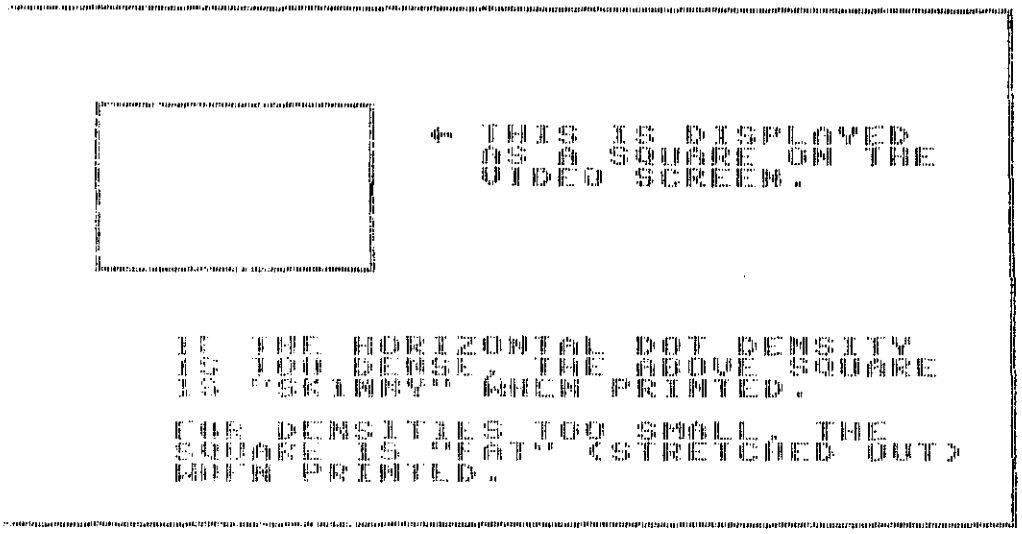
Double size dump time = 165 sec.



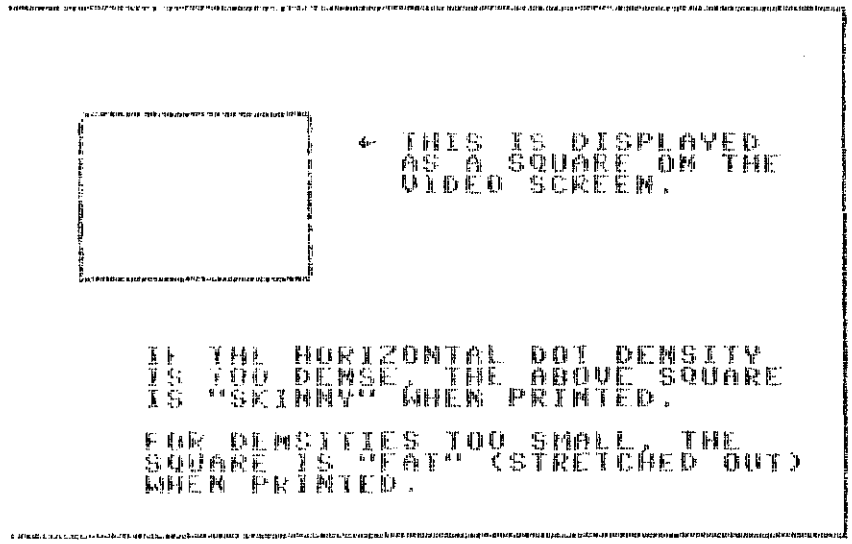
← THIS IS DISPLAYED AS A SQUARE ON THE VIDEO SCREEN.

IF THE HORIZONTAL DOT DENSITY IS TOO DENSE, THE ABOVE SQUARE IS "SKINNY" WHEN PRINTED.

FOR DENSITIES TOO SMALL, THE SQUARE IS "FAT" (STRETCHED OUT) WHEN PRINTED.

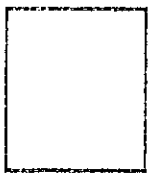


Horiz. Density = 60 dpi (Commodore 1525, Epson MX-70, and others)



Horiz. Density = 72 dpi (Most Okidatas, and others)

With horizontal densities of either 60 or 72 dots per inch, double size dumps can't be used because they would exceed the size of a normal 8 1/2 inch wide sheet of paper.

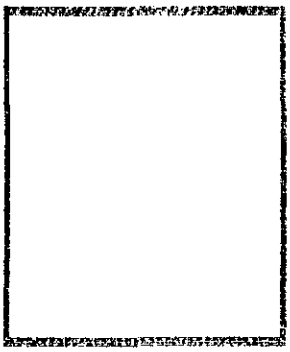


← THIS IS DISPLAYED AS A SQUARE ON THE VIDEO SCREEN.

IF THE HORIZONTAL DOT DENSITY IS TOO DENSE, THE ABOVE SQUARE IS "SKINNY" WHEN PRINTED.

FOR DENSITIES TOO SMALL, THE SQUARE IS "FAT" (STRETCHED OUT) WHEN PRINTED.

Horiz. Density = 120 dpi
(Gemini 10X, Epson MX-80, and others)



← THIS IS DISPLAYED AS A SQUARE ON THE VIDEO SCREEN.

IF THE HORIZONTAL DOT DENSITY IS TOO DENSE, THE ABOVE SQUARE IS "SKINNY" WHEN PRINTED.

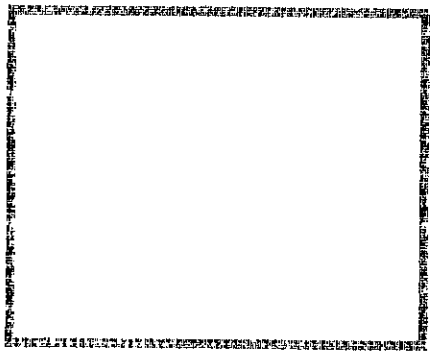
FOR DENSITIES TOO SMALL, THE SQUARE IS "FAT" (STRETCHED OUT) WHEN PRINTED.

← THIS IS DISPLAYED AS A SQUARE ON THE VIDEO SCREEN.

IF THE HORIZONTAL DOT DENSITY IS TOO DENSE, THE ABOVE SQUARE IS "SKINNY" WHEN PRINTED.

FOR DENSITIES TOO SMALL, THE SQUARE IS "FAT" (STRETCHED OUT) WHEN PRINTED.

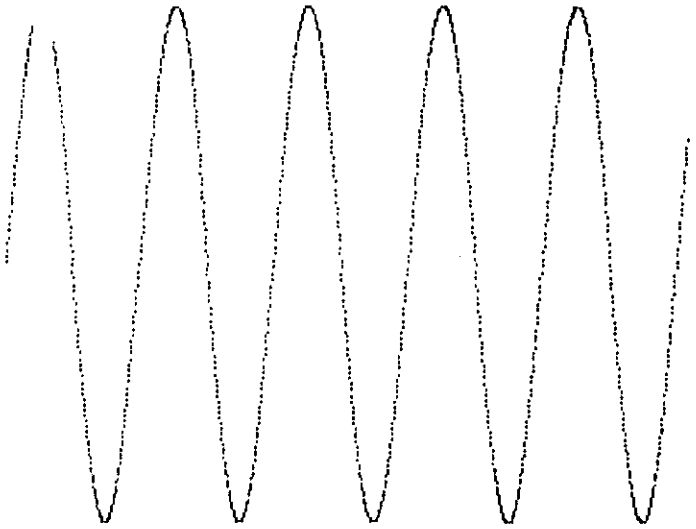
Horiz. Density = 80 dpi
(Prowriter, Everett/Charles, and others)



← THIS IS DISPLAYED AS A SQUARE ON THE VIDEO SCREEN.

IF THE HORIZONTAL DOT DENSITY IS TOO DENSE, THE ABOVE SQUARE IS "SKINNY" WHEN PRINTED.

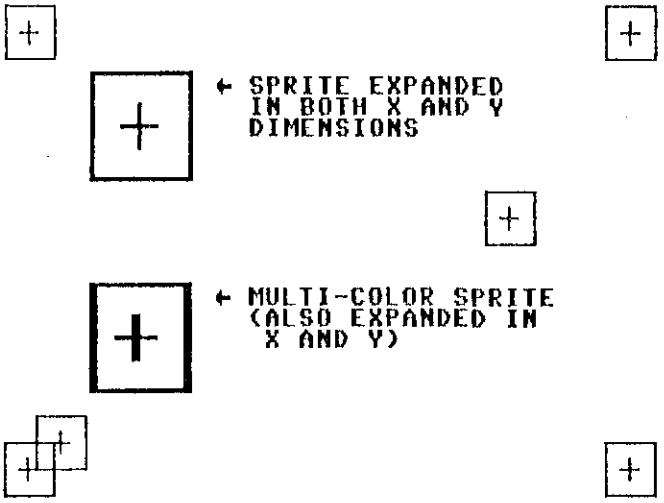
FOR DENSITIES TOO SMALL, THE SQUARE IS "FAT" (STRETCHED OUT) WHEN PRINTED.



Example dump of HI-RES screen

Dump time = 43 sec.

(There is no significant difference in dump times for HI-RES or LO-RES screens. In both cases, there are 320 dots horizontally by 200 dots vertically to be printed.)



Example dump of LO-RES screen with all eight sprites activated

Dump time = 73 sec.

(The number of sprites activated does have an impact on print time. Sprites are pesky little critters that require more time to find them, and then to determine if they are hiding screen characters, being hid by screen characters, or hiding each other.)

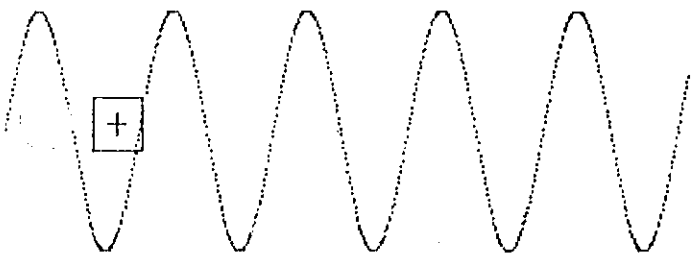
EXAMPLE DUMP OF A "SPLIT SCREEN"
(UPPER PART IS LO-RES; LOWER IS HI-RES)

THE SCREEN IS GENERATED BY A RASTER INTERRUPT PROGRAM.

← A SINGLE SPRITE IS DISPLAYED IN THE LO-RES PORTION, AND IS THEN MOVED TO THE HI-RES PORTION WHERE IT IS DISPLAYED AGAIN (LOOKS LIKE 2 SPRITES).

Example dump of split screen

Dump time = 44 seconds



```
      FUNCTION KEY DEFINITIONS

F1 - HELP KEY (ESCAPE FROM OTHER KEYS)
F2 - SCREEN DUMP TO PRINTER
F3 - BASIC MEMORY ALLOCATION
F4 - SCREEN DUMP TO DISK
F5 - SCREEN MEMORY LOCATIONS
F6 - SCREEN LOAD FROM DISK
F7 - DEC/HEX CONVERSION
F8 - USER DEFINED

   SCREEN DUMP, ETC MEMORY USAGE:
   18304 ($4780) TO 32767 ($7FFF)

  RUN STOP-RESTORE DISABLES KEYS,
  SYS 30000 REENABLES KEYS

       TAP SPACE BAR TO RETURN
         TO ORIGINAL PROGRAM.
```

Typical screen display when f1 is pressed

```
       BASIC MEMORY ALLOCATION

 START OF BASIC TEXT     2049   $0801
 START OF VARIABLES      2375   $0947
 START OF ARRAYS         2389   $0955
    OF ARRAYS (+1)       2551   $09F7

   BASIC BYTES FREE = 34397 ($865D)

 START OF STRINGS       36948   $9054
 TOP OF BASIC           40960   $A000


  TO CHANGE MEMORY LOCATION POINTERS,
MOVE CURSOR TO LINE YOU WANT CHANGED,
AND PRESS "RETURN".  FOR NO CHANGES,
PRESS "RETURN" WITHOUT MOVING CURSOR.
```

Typical screen display when f3 is pressed

```
       SCREEN MEMORY ALLOCATION

 LO-RES SCREEN            1024   $0400
 HI-RES SCREEN          OFF
 CUSTOM CHARACTER SET   OFF

 SPRITE #0 DATA            832   $0340
 SPRITE #1 DATA         OFF
 SPRITE #2 DATA            832   $0340
 SPRITE #3 DATA         OFF
 SPRITE #4 DATA         OFF
 SPRITE #5 DATA         OFF
 SPRITE #6 DATA            896   $0380
 SPRITE #7 DATA            960   $03C0


  TO CHANGE ANY OF THE INFORMATION,
MOVE CURSOR TO LINE YOU WANT CHANGED,
AND PRESS "RETURN".  FOR NO CHANGES,
PRESS "RETURN" WITHOUT MOVING CURSOR.

  IF LO-RES SCREEN LOCATION IS MOVED,
SPRITE DATA POINTERS ARE ALSO MOVED.
```

Typical screen display when f5 is pressed