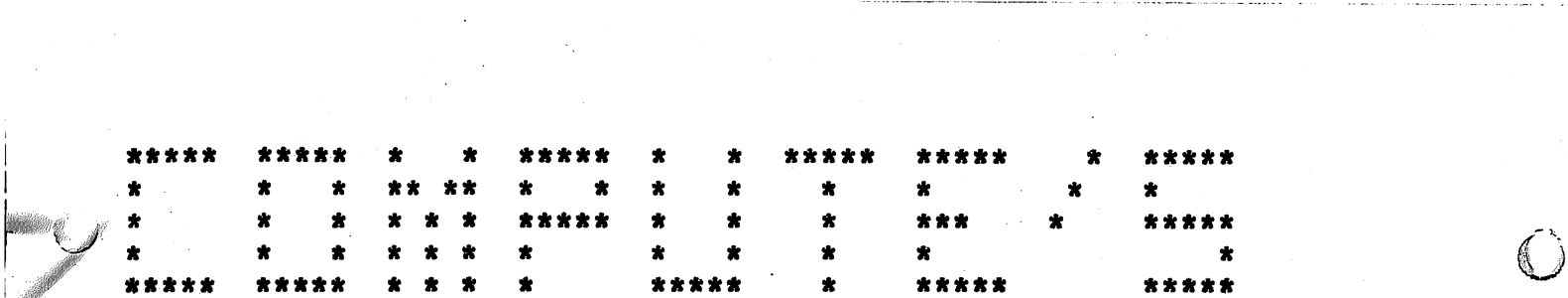
THE COMPUTE!'S GAZETTE DISK

AUGUST 1994



G A Z E T T E O N D I S K
==========================

A U G U S T 1 9 9 4
====================

Disk Magazine

(2) = Table Of Contents

PROGRAMS:

(4) = TurboDOS 2

(8) = Bible Helper

(11) = Wanaka

(12) = Animator

(14) = Mathematics Toolbox

(19) = TenPins (PD)

(19) = Pulsating Pictures (PD)

COLUMNS:

(20) = 64/128 View

(22) = Feedback

(26) = D'Iversions

(29) = Beginner Basic

(32) = Machine Language

(37) = Machine Language 2

(38) = Programmer's Page

(42) = G.E.O.S.

(45) = PD Picks

****

FEATURES:

(48) = Creating G.E.O.S. Fonts

(54) = REVIEW: The Blues Brothers

To Load Disk:

Load "Menu",8,1 and press <Return> --- Commodore 64

****

GAZETTE DISK *** August 1994

Features:

CREATING GEOS FONTS

By Harold Stevens Jr.

Create your own interesting fonts for GEOS by scanning them into your computer with a Handyscanner 64.

Reviews:

The Blues Brothers reviewed by Don Radler.

Columns:

64/128 VIEW by Tom Netsel.
We'd like to hear your suggestions for making Gazette Disk better. Also, have you seen the new Commodore World magazine from CMD?

FEEDBACK.
Comments, questions, and answers.

D'IVERSIONS by Fred D'Ignazio.
Channel Z--A Parking Lot for Couch Potatoes.

MACHINE LANGUAGE (Part 1) by Jim Butterfield.
Reading All Directories.

MACHINE LANGUAGE (Part 2) by Jim Butterfield.
Reader.

BEGINNER BASIC by Larry Cotton.
Variables Won't; Constants Aren't.

PROGRAMMER'S PAGE by David Pankhurst.
BASIC Tools.

GEOS by Steve Vander Ark.
Defining Greatness.

PD PICKS by Steve Vander Ark.
Tenpins and Pulsing Pictures.

128 Programs:

3-D Bar Graphs by S. Bala Gangadher Rao.
A program for creating a variety of graphs for the 128. Requires 80-column mode.

64 Programs:

TurboDOS 2 by Hong Pham.
Use this utility to boost the speed of your 1541 or 1571 by 1400 percent.

Bible Helper by Daniel Lightner.
Create a database of Bible scriptures.

Animator by Shawn Menninga.
Use this program like a flip book to create lo-res moving pictures.

Mathematics Toolbox by Bruce Bowden.
This advanced mathematical utility solves integers, finds roots of functions, and factors integer values.

Wanaka by Shawn Menninga.
Reassemble the map of New Zealand with this block-sliding program.

Tenpins (PD).
A bowling simulation for one to four players.

Pulsing Pictures (PD) by George Trepal.
A graphics utility for creating dazzling animated screens.

Gazette, August 1994

TURBODOS 2

By Hong Pham

Boost the speed of your 1541 or 1571 disk drive by 1400 percent

The 1541 disk drive is notorious for its slow speed. It may come as a surprise to learn that it's not the hardware that's responsible for the slowness; the culprit is the DOS software that's built into ROM. Some DOS routines, such as data transferring and formatting, are inefficient. Fastloaders work because they optimize the inefficient DOS routines.

Last year (June 1993), I wrote a program for Gazette called TurboDOS. It speeds loading, verifying, and saving up to seven times faster than DOS. Because TurboDOS is independent of DOS, it can use all 40 tracks on a 5 1/4-inch double density disk, increasing disk capacity to 749 blocks.

TurboDOS works well, but I've been able to improve it. TurboDOS 2 has all the features of TurboDOS, including the BASIC commands, but it's twice as fast as its predecessor. TurboDOS 2 is 14 times faster than normal DOS, and it is faster than the 1571 in burst mode. TurboDOS 2 achieves its speed without having to resort to an exotic disk format. To put this into perspective, a 200-block program will load in about ten seconds. In addition to faster speed, TurboDOS 2 revises some old commands, and it adds some new ones. TurboDOS 2 will work only with a 1541, a 1571, or a 1541-compatible disk drive.

TurboDOS 2 is written in machine language, but it loads and runs like a BASIC program. After a few seconds, TurboDOS 2 installs itself and then displays a startup message.

TURBODOS 2 COMMANDS

To make disk accessing more convenient, TurboDOS 2 adds 19 new commands to BASIC. If you are already familiar with TurboDOS commands, most TurboDOS 2 commands are similar. For most TurboDOS 2 commands, it is not necessary to denote the device parameter. TurboDOS 2 will assume that you are using drive 8 if no device number is specified. If you are in immediate mode, you can omit the device parameter for LOAD, SAVE, and VERIFY. One thing to remember is that a slash (/) must precede a TurboDOS 2 command if it is used within a BASIC program.

Ten of the 19 new commands are 1541-specific. Consequently, you must use a 1541 or a compatible disk drive to use these commands. Those ten commands are BR, BW, COLLECT, DIR, FORMAT, LOCK, RENAME, RESAVE, SCRATCH, and START. If you try to use the above commands on a non-1541-compatible disk drive, TurboDOS 2 will report a DEVICE NOT SUPPORTED error message.

TurboDOS 2 has added the following commands to BASIC.

BLOAD "filename", device, starting address
BLOAD loads a binary file and puts it at a specified location. If the starting address parameter is omitted, BLOAD will use the location specified by the file.

BSAVE "filename", device, starting address, ending address
Use BSAVE to save a file to a device from a specified memory location.

BR track, sector, buffer, ID, device
BR is like DOS's B-R command. BR reads a sector and puts it at a location specified by the buffer parameter.

Because TurboDOS 2 cannot tell if a new disk has been placed into the drive, an ID MISMATCH error message may appear when you use this command. If a new disk has been placed into the drive, set the ID flag to 1. Otherwise, use 0 for this parameter, or you can omit it altogether. Set this flag to 0 for any consecutive read or write attempts to the same disk.

For example, to read track 18, sector 0, and then put the sector's content at location $4000 (16384), type in BR/18,0,16384,1.

BW
BW is similar to BR, but BW writes from memory to a sector instead of reading from a sector.

COLLECT device
COLLECT is much like DOS's VO command. It validates the disk, updates the BAM, and deletes any splat files from the directory. You should use this command instead of DOS's VO on a TurboDOS formatted disk. It's not only faster, but also DOS will not validate the disk correctly if there are files saved beyond track 35.

COLOR border, background, cursor
You can change the default border, background, and cursor color to your own preference. Use the Commodore color numbers 0-15. Press Run/Stop-Restore to activate.

DEFAULT device
TurboDOS 2 usually defaults to drive 8 when no device number is specified, but you can change the default device with this command.

DIR "filename", device
DIR displays the disk directory on the screen. Entering DIR by itself will display the entire directory. You can specify which files to display with a string parameter. Wildcards are allowed.

To pause the directory display, hit the space bar. To continue, press any key. Pressing the Run/Stop key will stop the display.

DISK "string", device
DISK sends a disk command or displays the drive status. If entered by

itself, the drive status will be displayed.

ERR
To display the last disk error which TurboDOS 2 encountered, use ERR.

FORMAT "ID", "disk name", device
Because normal DOS cannot format tracks beyond 35, TurboDOS 2 has a custom disk formatter. A disk will format in about ten seconds. Use any two ASCII characters for the ID and up to 16 characters for the disk name. Note that the format is reversed from DOS, with the ID first and then the disk name.

KILL
KILL disables TurboDOS 2, and it restores the previous load, save, and BASIC vectors.

LOCK "filename", mode, device
A locked file cannot be scratched. To lock a file, set mode to 0. By using wildcards, you can lock the entire directory.

LOCK can also unlock files. Set mode to 1 to unlock. If you omit the mode parameter, TurboDOS 2 will assume that you want to lock a file.

RENAME "old name" TO "new name", device
Use RENAME to change the name of a file on the disk directory.

RESAVE "filename", device
When updating a BASIC program, you will have to scratch it before you can resave it. RESAVE does all this in one step by scratching the file first and then saving it.

SCRATCH "filename", device, file type
SCRATCH either erases or unscratches a file from the disk directory. If you omit the last parameter, SCRATCH will scratch the specified file. To unscratch a file, set the file type parameter to the file type number of the scratched file. The following is a table of the file type number and its corresponding file type.

0 = DEL deleted files
1 = SEQ sequential
2 = PRG program
3 = USR user
4 = REL relative

The entire directory can be scratched or unscratched with wildcards. If you are scratching files on a TurboDOS formatted disk, do not use the familiar S0: command from DOS.

SKEW sector interleave, directory interleave
Files will load faster if they are saved at an optimum sector interleave. TurboDOS 2 saves files at a 1:4 sector interleave, and it saves the directory at a 1:7 interleave.

START "filename", device, new load address
START displays a program's load address or changes it. If you wish to see the program's load address only, omit the last parameter.

TRACK highest track, directory track
You can set the highest track that TurboDOS 2 will use or which track the directory is on by using TRACK. If entered by itself, the highest track and the directory will be displayed.

If the head always chatters when approaching track 40 on your disk drive, you should lower the highest track number to 39 or 38.

ODDS AND ENDS
To create a protected area for itself, TurboDOS 2 moves the bottom of BASIC from $0801 (2049) to $2701 (9985). Some of you may notice that TurboDOS 2 is larger than its predecessor, but it offers more memory for BASIC programs. The explanation to this contradiction is that TurboDOS 2 stashes some of its codes under BASIC ROM at $B500 to $BFFF (46336 to 49151).

Unlike TurboDOS, TurboDOS 2 will not move the head beyond track 40. TurboDOS 2 will check for illegal tracks, and if one is encountered, it will report an ILLEGAL TRACK OR SECTOR error message.

You should not specify the drive number when using commands which require a filename, such as LOAD, SAVE, LOCK, and SCRATCH. For example, do not use LOAD "0:*", 8 to load from drive 0. TurboDOS 2 will look for a file whose name is 0:*, and you will get a FILE NOT FOUND error message. Instead, you should simply use LOAD "*",8.

If you are using wildcards with TurboDOS, you cannot specify the file type. For example, DIR "*=s" will not display all sequential files on disk.

Because of differences in the circuitry of 128s, TurboDOS 2 may or may not work on your machine. TurboDOS 2 works fine on all 64s.

Hong Pham lives in Antigonish, Nova Scotia, Canada.

Gazette, August 1994

BIBLE HELPER

By Daniel Lightner

Create a database of Bible Scriptures with this program for the 64.

Bible Helper is an information storage utility that allows you to compile huge files full of your favorite Scriptures, a concordance of just the things you need to find on short notice during a Bible study. You can instantly call up any of them using a string Search and Retrieval command.

To use Bible Helper, just load and run as you would any BASIC program. Bible Helper displays a menu of ten different options. You would normally load a previously saved file at this point, but since we're starting, we'll first have to enter some phrases.

Choose the option Enter Phrases by pressing Shift and E simultaneously. (You will be prompted as to whether or not you wish to execute the option. Press Y to continue or N to return to the main menu. Each option will prompt you in a similar manner.)

After you have responded to the prompt with Y to continue, a marker (>) appears several lines below the prompt. This is where you start typing in data. A line of data can be up to 80 characters long. Let's use a few lines from the beginning of the Book of John to illustrate the special functions of Bible Helper.

First, enter partial lines that have the same book, chapter, and verse. Enter, BEGINNING WAS THE WORD JOHN 1:1 and press Return. Now to verify that it is indeed entered, choose the View command by pressing Shift and V. Answer the prompt, and your line should appear at the bottom of the screen. Press the Crsr right key and then Return to exit View Phrases.

If you had entered more phrases, pressing the Crsr right key would have fetched the next phrase. Since there were no other phrases, Bible Helper responded with the PRESS RETURN prompt, indicating the end of the file.

Next, enter the following phrases and, as you do, experiment with the View option. While in the view mode, you can exit anywhere during the listing by pressing the asterisk key (*) or Return.

THE WORD WAS WITH GOD JOHN 1:1

THE WORD WAS GOD JOHN 1:1

IN HIM WAS LIFE JOHN 1:4

LIFE WAS THE LIGHT JOHN 1:4

LIGHT THAT LIGHTETH EVERY MAN JOHN 1:9

POWER TO BECOME SONS OF GOD JOHN 1:12

THE WORD WAS MADE FLESH JOHN 1:14

The reason I chose these particular phrases is that they contain a mixture of similar words that will be ideal for demonstrating the Hunt for Phrase option. This option works exactly like View except you will be prompted for a string to search for. Enter the Hunt for Phrase option with Shift and H.

Use this facility to search for the following and observe the results. Enter GOD, WORD, LIGHT, JOHN 1:1, and JOHN 1.

The concept involved should be thoroughly understood at this point. If you wanted to keep Christ's words separate from all other phrases, you could include CW after each phrase that contains Christ's words, PU for Prophesies Unfulfilled, or PF for Prophesies Fulfilled. Then to recall phrases that contain only unfulfilled prophesy you could do a string search for PU.

A line could be added that simply describes these special codes. For example, (INDEX)=(CW)=CHRIST'S WORDS:(PF)=PROPHESY FILLED:(PU)=. The prospects are limitless.

Next, we'll delete some phrases. Call the delete phrase mode with Shift and X. This acts exactly like view mode except if you hit the Return key on an entry, it will be deleted from the file. The asterisk will still allow you to abort without deleting any phrases should you change your mind. Delete a couple of phrases and then enter view mode to see that they are gone.

The Print Phrase option is accessed by pressing Shift and P. It does a string search, but rather than sending the matching phrases to the screen, it sends them to the printer, device 4 on the serial port. Be sure your printer is set up to receive the data.

The Save option, Shift and S, saves the file under the name that you specify. Files can be as long as 192 disk blocks.

The Load option, Shift and L, does just that. It loads any file you specify.

The Quit option, Shift and Q, sends you back to BASIC.

The menu, accessed by pressing the dollar sign ($), will display the directory listing for drive 8. This option can be called from the Disk Command module. Freeze the listing or restart it using the space bar.

The final option is the Disk Command module. Use Shift and D to enter this mode. Here you are able to send disk commands to the command channel. Disk commands should follow the same syntax that they would when opening the command channel to drive 8 from BASIC.

Copy file command is CO: or C: followed by the new name, an equal sign, and the old name. Press Return after each command. Here's an example.

```
CO:newfile=oldfile
```

The Rename File command uses the same syntax.

```
RO:newname=oldname
```

The Scratch command is SO: followed by the filename to be scratched. Wildcards are allowed.

```
SO:filename
```

Initialize is IO:

Validate is VO:

The Format command is NO: followed by name, comma, and two letters representing the unique ID.

```
NO:disk name,ID
```

The format command will destroy all data on a disk, so be sure that you put the proper disk in the drive before pressing Return.

You can abort Disk Command by entering a single asterisk.

Entering $ will send you to the Menu module.

Bible Helper files can be loaded into SpeedScript and edited, but a few rules of syntax must be applied. If you are adding new lines, you must start at the at sign (@) which is the third to the last character on the screen. Be careful not to enter lines longer than 80 characters. All characters must be lowercase, and lines must end with the high bit of the last character set--in other words, in reverse video. This can be accomplished by using the Control key in conjunction with the number 3 key and then entering the character in lowercase. Do not use any Return characters; all data must be adjoining. The very last line must be followed by the at sign (@) and two other characters of any type.

Daniel Lightner lives in Sidney, Montana.

Gazette, August 1994

WANAKA

By Shawn Menninga

A block-sliding puzzle for the 64

The Wanaka puzzle is an interesting device that can be found all over New Zealand; it consists of a wooden frame containing ten sliding blocks of different shapes.

As geography buffs know, New Zealand consists of two major land groups, the north and south islands. The largest of the Wanaka blocks has a picture on it of New Zealand's north island. There's a similar picture of the south island on the frame. The object of the puzzle is to rearrange the blocks in order to match the two islands, forming a completed map of New Zealand. This program simulates the Wanaka board on the screen of your 64.

SOLVING THE PUZZLE

To work at the puzzle, load and run Wanaka, which is written in BASIC. After a brief pause, the puzzle board appears. In the upper left corner of the board are blue crosshairs. The crosshairs are your cursor. Position the cursor (with a joystick in port 2) on the block you wish to move and hold down the fire button. The crosshairs will change to a set of white arrows. Push the joystick in the direction that you wish to slide the block and then release the fire button. The block will move, if possible, and the arrows will change back to crosshairs. The puzzle is solved when the large purple block is moved from the center top to the center bottom and the islands match up.

For scoring, a counter in the upper left corner of the screen counts your moves. A move consists of moving one block one space. I have not been able to solve the puzzle in fewer than 124 moves, but you may have better luck.

Shawn Menninga lives in Grand Rapids, Michigan.

Gazette, August 1994

ANIMATOR

By Shawn Menninga

A low-resolution animation program for the 64

Remember those flip books you made as a kid where you stapled about 20 sheets of paper together and drew a series of pictures (each a little different from the previous one) on each page? Of course, about the time you finished and ran to show your flip book to your mother, the staple fell out, and the whole thing was ruined. Now you can create simple flip books on the computer, and the best part is--no staples.

CREATING A FLIP BOOK

Animator is written in BASIC, but when it's first run, it pauses briefly to poke in a few machine language routines. The screen will flicker momentarily as the workscreens are cleared and then the title page appears. This page briefly summarizes the editing controls. Press any key to continue to the main screen.

The main screen consists of an input prompt asking for the screen that you wish to edit. Enter a number from 1 to 24. This is the screen number that represents that page of the flip book. If you enter a number greater than 24, the options menu will appear; all other inputs are ignored.

To create an animation sequence, simply use the keyboard to draw the first frame. You can use any of the Commodore's graphic keys, letters, or numbers. After you've drawn the first image on screen 1, continue the sequence by redrawing or copying the screen to screen 2 and so on, making changes in each screen to create the effect of animation.

THE OPTIONS SCREEN

The Options menu consists of six choices, each with the first letter highlighted. To use an option, press its first letter.

COPY SCREEN(S) -- You will be prompted for the numbers of the screen to copy and the screen to copy it to. Any invalid inputs (less than 1 or greater than 24) will return you to the main screen. After the screen is copied, you will be asked if you would like to copy another screen. Press Y to copy another or N to return to the main screen. This feature can save you from having to redraw each screen in order to make a few animation changes.

ANIMATE -- The Animate option assumes that your flip book starts on page 1. It prompts you for the last page of the series. At the next prompt, enter Y to cycle from the first page to the last page and then back to the first or N to jump back to the start from the last page. The final prompt asks for a speed of animation from 1 to 16, with 1 being the slowest and 16 the fastest. I find that speeds 1 to 4 are useful for finding mistakes in the flip book and speeds 12 to 15 are the best for animation. Once the animation is running, press any key

to return to the main screen.

LOAD SCREENS -- Enter the name of a saved animation sequence, and it will be loaded into memory. To see a simple demonstration of Animator in action, load the file BALL and then animate all 24 screens of its screens. Be aware that this demo program is rather large and will take several minutes to load.

MAIN SCREEN -- Returns you to the main screen.

SAVE SCREENS -- First, enter a name for the animation file. If the file already exists, you will be prompted to either replace the existing file or enter a new name. Finally, enter the number of the last screen to save.

QUIT -- Exits Animator.

HOW IT WORKS

Normally, screen data is stored in memory from addresses 1024 to 2023. By changing the high nibble of register 24 of the VIC chip, it is possible to move the screen around. (Actually, all that changes is where the VIC chip looks for the screen.) This gives 16 possible locations for the screen, from address 0 (very dangerous) to address 15360. But, that's only a quarter of the 64's memory.

By changing the least significant two bits of address 56577, the VIC chip can be told to look at a different bank of 16K. This gives a total of 64 possible screen locations. However, many of these locations are unavailable to Animator. Some are hidden under ROM, others have no character sets available, and a few screens are covered by the I/O addresses. Without writing Animator exclusively in machine language, there are at most 25 screens available to it. Animator uses 24 of these for animation and the 25th for prompts.

Shawn Menninga lives in Grand Rapids, Michigan.

Gazette, July 1994

MATHEMATICS TOOLBOX

By Bruce M. Bowden

A mathematical utility for the 64 that solves integrals, finds the roots (zeros) of functions, and factors integer values

Save yourself a lot of numerical grunt work by using this handy collection of mathematical tools for the 64. Mathematics Toolbox numerically solves integrals, finds the roots (zeros) of functions, and factors integer values.

Mathematics Toolbox is written entirely in BASIC. It's very important that the line numbers are used exactly as listed. This program modifies itself to accept formulas from the user; therefore, line numbers must not be altered.

GETTING STARTED:

When Mathematics Toolbox is first run, a title screen will appear. Press any key to get to the first menu page. You may choose the tool you need--integral solution, roots, or factorization--by pressing f1, f3, or f5, respectively.

FACTORIZATION OF INTEGERS

Choose factorization first by pressing the f5 key at the main menu. You're prompted for an integer number to be factored. Try the value 12345678. The factors are displayed as follows.

2 3 3 47 14593

This program will always give prime factors. To find every factor of a number, take various combinations of its prime factors and multiply them. For example, some non-prime factors of our test value of 12345678 are 6 (2 x 3), 9 (3 x 3), 18 (2 x 3 x 3), 94 (2 x 47), and so on.

During the course of our first factorization, we've discovered a large prime: 14593. What if the number we try to factor is already prime? Press any key to move from the factor screen back to the main menu. Choose factorization again, and this time enter 3739. The program requires a moment to search and then reports that 3739 is prime.

ROOTS BY BISECTION

When the curve Y=f(X) is graphed, there may be points where the solution crosses the x-axis. These crossing points are called zeros or roots of the function. They're values of X where f(X) = 0 and are important for solving equations.

Press f3 from the main menu. A new screen will describe what the factorization tool is. Press any key to move on. Now you're prompted for the function. For Y=f(X), you need to enter the f(X) part.

Let's try something simple: the square root of 7. The function whose root gives this value is 7 subtracted from the square of the X value. In BASIC, such an assignment would read Y=xX*X-7. Enter X*X-7 at the prompt and press Return.

Next, you're asked first for the lower boundary of the interval being searched. Enter -10 here. When prompted for the upper boundary, enter 10. You're also asked for the number of partitions. This program finds roots by dividing the range being searched into the quantity of partitions you specify and zeroing in on any roots within each partition. For this reason, care must be taken to insure that there is no more than one root per partition. For now, though, two partitions is enough. Enter 2. In a moment you have the roots: -2.64575131 and 2.64575131, the negative and positive square roots of 7. Press any key to return to the main menu.

MANY ROOTS

For the next example, we'll use a function which yields a great many roots. Return to the function prompt by pressing f3 and any key at the description screen.

Type sin(X) at the function prompt. For the lower boundary, choose 0. For the upper, choose 100. We know that the distance between sine function crossings of the x-axis is about 3.14, so 32 partitions, a value greater than (upper-bound - lower-bound)/3.14, should catch all the roots in the given interval. Enter 32 and the roots will begin to be displayed.

Notice that eight roots (all integral multiples of pi) are found and the prompt for pressing a key comes up. Because of round-off error, the first root is not exactly 0, but its small value should suggest to you that the actual root is indeed 0.

When a key is pressed, the program continues to build up solutions on a new screen. After eight more roots are displayed, you're prompted again, and so on until all of the roots are found and displayed. The final root is actually a little beyond 100, an added bonus of the algorithm used.

This part of the program requires a little forethought before use, for you must always be careful to avoid situations where BASIC will yield an error message because of undefined conditions, such as the logarithm of a negative number or division by 0.

INTEGRAL SOLUTION

Roughly speaking, integrals are used in calculus for performing an infinite sum of infinitesimal parts. To choose integral solutions, press f1 from the main menu. A new screen will describe that this part of the program solves integrals using Simpson's numeric method. Press any key to move on.

Now you have a screen full of different integral solutions to choose from. The choices, selected from the function keys, look like the

following menu.

f1 *** Area under a rectangular curve
f3 *** Area under a polar curve
f5 *** Area between two rectangular curves
f7 *** Area between two polar curves
f2 *** DISPLAY MORE CHOICES

Pressing f2 (shifted-f1) presents a new screen with more choices.

f1 *** Length of a rectangular curve
f3 *** Length of a polar curve
f5 *** Surface of revolution about the y-axis
f7 *** Surface of revolution about the x-axis
f2 *** DISPLAY MORE CHOICES

Pressing f2 again presents a third screen with still more choices.

f1 *** Solid of revolution about y-axis
f3 *** Solid of revolution about x-axis
f5 *** Average for a curve
f7 *** REVIEW CHOICES

Return to the first screen of integration choices by pressing f7 from the last menu.

The choice for finding the area under a rectangular curve is the most basic. It finds the area between the curve Y=f(X) and the x-axis. Press f1, and you'll be prompted for a function. Try entering the function X*X and press Return. When prompted for the lower boundary, choose 0. Enter 4 for the upper boundary.

Like the roots tool, the integration tool uses partitions. The larger the number of partitions chosen, the more accurate the answer. Too many partitions can slow the computer down, however, so choose carefully.

For the partition prompt on this example, enter a value of 10. After only a moment's calculation, a solution of 21.3333333 is returned. The actual solution is 21+1/3, so we're as close to the answer as the 64's accuracy allows. Press any key to get back to the main menu.

Entering data for the other integral solutions is much the same, so I'll conclude here by describing only the special properties of each.

UNDER A POLAR CURVE
Screen 1, selection f3: Area under a polar curve.

A polar curve is one traced out by a the end of a line segment anchored at the origin of the x,y plane. The function used is R=f(X), where X, in this case, is an angle. You can imagine the line segment as having a length R and turned through an angle off of the x-axis by the angle. Why use X to symbolize the angle also?

Because the formula-entry routine in this program only accepts X as the independent variable. The area being calculated is that swept out by the line segment as the angle and length change. When prompted for a lower and upper boundary, remember that these are starting and ending angles.

RECTANGULAR CURVES
Screen 1, selection f5: Area between two rectangular curves.

This selection will solve for the area between the two curves f(X)-g(X). This time you'll be prompted for both functions, but everything else is the same.

BETWEEN TWO CURVES
Screen 1, selection f7: Area between two polar curves.

The same principle applies as in selection f5 on this screen, but with polar curves as described for selection f3.

LENGTH OF RECTANGULAR CURVE
Screen 2, selection f1: Length of a rectangular curve.

This selection solves for the length of the curve described by Y=f(X) for X ranging from some lower to some upper bound.

LENGTH OF POLAR CURVE
Screen 2, selection f3: Length of a polar curve.

This selection solves for the length of the curve described by the polar function f(X) for X ranging from some lower to some upper angular bound.

SURFACE OF REVOLUTION/Y-AXIS
Screen 2, selection f5: Surface of revolution about the y-axis.

This gives the area of the surface defined by the curve of the rectangular function f(X) when rotated about the y-axis. The beginning and end points of the curve are entered as the lower and upper bounds.

SURFACE OF REVOLUTION/X-AXIS
Screen 2, selection f7: Surface of revolution about the x-axis.

This gives the area of the surface defined by the curve of the rectangular function f(X) when given a full rotation about the x-axis. The beginning and end points of the curve are entered as the lower and upper bounds.

SOLID OF REVOLUTION/Y-AXIS.
Screen 3, selection f1: Solid of revolution about the y-axis.

This gives the volume within the surface defined by the curve of the rectangular function f(X) when rotated about the y-axis. The beginning

and end points of the curve are entered as the lower and upper bounds.

SOLID OF REVOLUTION/X-AXIS.
Screen 3, selection f3: Solid of revolution about the x-axis.

This gives the volume within the surface defined by the curve of the rectangular function f(X) when given a full rotation about the x-axis. The beginning and end points of the curve are entered as the lower and upper bounds.

CURVE AVERAGE
Screen 3, selection f5: Average for a curve.

This selection is used to find the average value of f(X) within the range of X values starting at the lower bound and ending at the upper bound.

If you're a student, professional, or amateur enthusiast who uses math frequently, Mathematics Toolbox has been designed to be an invaluable addition to your collection of mathematical tools!

Bruce M. Bowden lives in Greensboro, North Carolina.

Gazette, August 1994

TENPINS and PULSATING PICTURES

Tenpins and Pulsating Pictures are the public domain programs featured in this month's "PD Picks" column. No separate instructions come with these two programs.

Tenpins is a bowling simulation program for one to four players that requires a joystick in port 2. The program automatically keeps score for each player.

Pulsating Pictures, by George Trepal, is a graphics utility that lets you use keyboard graphics to create interesting messages and displays. The program then animates those graphics. You can combine different elements to create some dazzling displays.

For more information about these two programs, be sure to read "PD Picks" in the Columns section of this disk.

If you have an outstanding public domain or shareware program that you think other Gazette Disk readers would enjoy, send it on disk to PD Picks, COMPUTE Publications, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408.

Gazette, August 1994

64/128 VIEW: Towards a Better Product

By Tom Netsel

Last month, I asked you to take a minute and complete the Gazette Readership Survey. This is your chance to tell me what you like and don't like about Gazette Disk. There's also space on the survey for any comments you may care to make. I hope you'll give this area some thought and jot down any ideas or suggestions that you may have for improving Gazette.

In a recent issue of The Users Port, a newsletter from the Rancocas Valley User Group in Cinnaminson, New Jersey, editor Gil Heath had a few words to say about Gazette Disk.

"I have been subscribing to COMPUTE for a few years and even though, as time passed, I was not happy by the reduction in space for the 64/128, I was happy to read everything in the Commodore section. Well! Now with the magazine being on disk and having to load and select each and everything in the magazine, it has become a bore."

He then goes on to say, "I don't think I will continue the subscription after this one ends. I know it's not good for our benefit to cut back on support of a company which has always supported the 64 and 128, but I am not being realistic if I buy a disk/magazine that I do not use."

Tough words, but I respect Mr. Heath's opinion. He said Gazette Disk is a bore. As editor, it's my job to make the disk magazine interesting, and I wonder what I can do to keep Mr. Heath as a subscriber. Since he doesn't read all the articles any more, there's a good chance he'll miss seeing this column as well as the survey that we published last month. If he does though, I hope he'll let me know what I can do to make the magazine more interesting for him. That's why I'm taking this time to ask all of you to do the same thing: pass along any suggestions or ideas for improvement.

If you're a programmer and have already worked out some specific suggestions or improvements, how about sending me a demonstration disk. If you're a writer with some Commodore expertise, I'd also be interested in hearing from you. Gazette Disk has an ongoing need for good, clear articles that deal with technical and nontechnical aspects of the 8-bit Commodores. We pay for those, too.

Speaking of Commodore articles, have you seen Creative Micro Design's new magazine? I've just received the premier issue of Commodore World--and it's on paper! Charles Christianson is general manager and Doug Cotton is the editor. From the looks of the first 44-page issue, they've done an outstanding job.

In case you've been out of touch for a few years and aren't familiar with CMD, it's a company that's long been a supplier of Commodore products. It bought and now sells many of the products once

distributed by RUN magazine; it's the primary GEOS distributor; and it offers hard drives, peripherals, and numerous software titles for the 64 and 128.

As Doug Cotton said in his opening editorial, "Strange as it may seem for us to find ourselves in the magazine business, it's a role which we take seriously."

After reading this first issue of Commodore World, I'll have to agree with Doug. Not that it's a strange idea, but that CMD seems to be putting a serious effort into publishing a magazine that'll be of interest to 64 and 128 users. With columns by Tim Walsh, Steve Vander Ark, Gene Barker, and other talented Commodore experts, CMD is off to a good start--there's even a type-in program. I wish them well in this new endeavor.

Of course, it's a fact that Gazette Disk and other Commodore publications are competing for your subscription and support. We'll all have to do better if we want to keep you happy. So from a purely business point of view, I hope you'll subscribe to our publication first and then consider the competition. But from one Commodore user to another, I'll have to admit that it's good to see a new publication that's dedicated to the 64 and 128.

It's almost like old times!

Gazette, August 1994

FEEDBACK

BUG-SWATTER
Corrections to previously published disks and programs.

I have a problem when adding new words to the Word Search (June 1994) database. I get a NEXT WITHOUT FOR ERROR IN 460 whenever I try to save a large list of words.
RICHARD DEHONEY
BROOKLYN, NY

Since Word Search writes to the disk when it adds words to its database, remember that you can't use it with the original Gazette Disk, which is write protected. You'll have to copy the program to a work disk that can be written to. You can either load the program into memory and save it to another disk or use a program such as Fast File Copier, which is on the May 1994 Gazette Disk. As with all programs on Gazette Disks, we recommend you copy them to a work disk and keep the original in a safe place.

To correct the FOR/NEXT problem, list line 460 and delete the letter P from the command NEXTP. It is the second NEXT on that line. The line should then read as follows.

```
NEXTI: H$=C$(P): C$(P)=S$: C$(S)=H$: NEXT
```

READER TO READER HELP
I have been working on programming in C recently, but I'm having difficulty interfacing with machine language. The directions in the user's manual for Spinnaker's Better Working Power C are not clear in this area. If anyone is familiar with Power C, I would appreciate your getting in touch with me.
EARL WOODMAN
P.O. BOX 85
DILDO, NF
CANADA AOB 1PO

I have a program that I bought in 1986 or 1987 from V.G Data shack in Brossard, Quebec, Canada. It's called the V.G. Data Shack Parallel Copier Superfast File Backup and Utilities, version 1.0.

It has served me well all these years, but I have never been able to back it up. I am looking for another copy if anyone other than me has one. I have written to the company but it is no longer in business. I have even mailed the program away, trying to have it copied, but no luck.
JAMES COTRILL
3119 PIONER RD.
PITTSBURGH, PA 15226

As a dedicated 128 user, I would like to know if a flow chart program is available for that machine. I would also be open to finding one for the 64.
BILL CLARK
8727 WEIDKAMP, RD.
LYNDEN, WA 98264

Perhaps our readers can offer some help with these requests.

MUSIC WANTED
I would like to propose the addition of some music pieces to the Gazette Disk.
G. GRANITZER
VIENNA, AUSTRIA

Thank you for the suggestion. Last month we published a new readership survey that asked for your input on a number of issues relating to Gazette Disk. There was also a space for comments and suggestions for improving the product. Several people have already asked for music files, and we will certainly consider adding them to the disk. If you haven't yet returned your completed survey, please do it as soon as possible. It'll help us make the disk better for you.

BAD DISK
I got the May Gazette Disk and it would not load. Will you please send me a new one right away?
ARTHUR HANAK
MELBOURNE, FL

There are times when a disk is simply defective and will not load. If that's the case, we are happy to offer a replacement, but there are a couple of things you might first want to try.

Gerry Gallinat in Montreal, Quebec, Canada, wrote about a Gazette Disk that apparently was blank on one side. After getting a replacement, he then read on a BBS about a similar problem and a possible solution.

Here's his tip. "Take the disk in both hands, bend it slightly, and pull it a few times over the edge of a table. Reasoning: Somehow the disk gets stuck inside the sleeve; the pulling frees it."

When disks are sent through the postal system, they can become tightly packed with other mail piled on top. This can cause the disk jacket to bind, preventing the disk itself from spinning properly. Check the movement of the disk inside the jacket. Make sure it can move. If it is tight, you may be able to free it by prying up slightly on the corners where the disk jacket is sealed, or trying Mr. Gallinat's tip. (Just be sure you don't crease the disk when pulling it over a table!)

Also, you might check to see if you have any cartridges plugged into

your machine. Sometimes they can interfere with proper loading.

If you do need a replacement, the quickest way to get one is to call (800) 727-6937. Don't return disks to our Greensboro office. That will only slow things. We do not usually get disks here until well after disks have been mailed to all subscribers. If for some reason the folks in Iowa at the 800 number can't help, then contact the Greensboro office.

SPACE FOR GAZETTE INDEX
According to the instructions that come on the 1991 Gazette Index disk, one should be able to enter post-1991 information, but is there enough room remaining on that disk? A listing of the directory shows only a few blocks of disk space remain on the original disk.
JAMES W. HOOD
SALT LAKE CITY, UT

The 1991 Gazette Index is a database of articles and programs that have been published in Gazette from 1983 through 1991. It can be a useful reference for Gazette readers who have a number of back issues of the magazine in their libraries.

To make the index as useful as possible to our readers, we made it so additional information could be added to it as new magazines came out. In other words, if you buy one index, you can update it yourself with later material. By now, most of side 1 and side 2 of the disk are filled with items, but that shouldn't be a problem.

The Gazette Index is not copy protected. You can copy the appropriate files from the original disk and create your own index. You don't have to delete any old material in order to make room for new. Just start a new disk.

With a file copier program, copy the following files to another disk.

BOOT
128BOOT
INDEX
TL
NT
0HELP
1HELP
2HELP
3HELP
4HELP

After you've copied the files, run the program and enter Edit mode by pressing f7. In Edit mode, you can either edit an old file or start a new one. To start a new file, press N on the menu screen.

Rather than going through the entire process here, consult the Gazette Index for complete information about editing. This information can be

found in the Help files. You can create your own custom index, adding as much material as you wish. The index supports up to 256 records in a file (0-255), with a maximum of 40 Index files per disk. Do not exceed this limit.

The 1983 - 1991 Gazette Index is still available for $9.95, plus $2.00 shipping and handling. It can be ordered by writing to 1991 Gazette Index, COMPUTE Publications, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408.

Gazette, August 1994

D'IVERSIONS:
Channel Z--A Parking Lot for Couch Potatoes

By Fred D'Ignazio

Last month we outlined four observable types of computer users whom we have characterized as motorists on their own primitive cyberspace highways, stuck in an insulated world of only one dimension. These are the hobbyist/educator, the business user, the online enthusiast, or the videogamer.

But what is magic about the number 4? Maybe there are just as many other types of computer users who don't neatly fit into these four types. In fact, why limit the motorists and the highways solely to computer users? This, too, reflects a provincial, computer-centric point of view. After all, isn't cyberspace commonly described as the marriage of computer, video, TV, and telephone technologies? These technologies represent multiple industries, multiple points of view, and multiple cultures among their devoted users.

It's likely that the biggest group of motorists who'll be in cyberspace in the future aren't currently out on Highway 101, 202, 303, or even 000. The biggest group is the average American consumer--young, old--you name it. They don't use computers or videogames, but they do watch TV--and lots of it. They are the couch potatoes: the videotape renters, the pay per viewers, the "Wheel of Fortune" fans, the soap opera lovers, and the avid shoppers in the emerging electronic home shopping malls.

These folks have never heard of the other computer-mediated highway, and they don't care about getting their driver's license for it either. Instead they are likely to stay parked in a giant cyberworld parking lot that we might call Channel Z. Cyberspace for them will be like driving their vehicles into a giant drive-in movie, parking, then participating in a collective imagi-dome experience of vicarious thrills: shopping, consuming, fun-housing, snooping, game playing, boredom-denying, and titillating.

What about the rules of the road? They are completely different from the other highways--even Highway 000. In a recent AT&T survey, it was discovered that very few "normal" people want to have anything to do with a computer keyboard. They also don't care to sit or stand close to a display screen. They would rather view their TV from six to ten feet away in a theater-like environment. And they don't want an interactive experience which requires the mastery of complex display screens and game controls. That would be like doing push-ups, sit-ups, and jogging--or reading a textbook on philosophy or physics. Heaven forbid!

Their real-world lives are exhausting and active enough. They want to limit their mental aerobics to passive channel surfing--a no-brainer, low-impact activity performed with one finger by clicking a hand-held

zapper from 70 to 100 times every five minutes.

The rules of the road for this crowd will be alien to many of the motorists on the other highways. The other motorists want stimulation, engagement, productivity, and results. The motorists parked in Channel Z only want escape, distraction, and emotional distancing. They want to tune in and glaze out.

While other motorists want a coherent starting point, middle point, and ending point to their journey, the motorists on Channel Z won't have the attention span to last that long. They crave infinite pictures in a picture (PIP) in which all channels can appear simultaneously on the same display screen. They can watch a cake being baked, a robbery being committed, a touchdown being scored, a city being bombed, a diamond necklace being purchased--all at the same time, all in the same timeless moment. They want to browse, take random detours through cyberspace, experience brief voyeuristic interludes, and then ZAP!--off on a new goalless, purposeless quest.

In the 1980s and early 1990s, travelers in cyberspace were all recognizable and distinct. They all traveled different highways, but once on a particular highway, they invariably drove similar cars, shared the same friends. They dressed alike, talked alike, and thought alike, in good, old-fashioned people-like-us (PLU) style.

Now it's time for nostalgia as those days are about to disappear, and things are soon going to get very, very messy. Suddenly the roadways in cyberspace are starting to converge. They are meeting at roundabouts, intersections, forking together rather than apart.

Suddenly, it's no longer clear that all computing is done with a keyboard, sitting on your bottom, manipulating words and numbers; or that all reality can be squeezed onto a desktop; or that you do your computing two feet from a screen, or six feet, or ten feet; or that you are in an office, a classroom, a bedroom, a living room, or indoors at all.

The early versions of Highway 303--global highways like Internet and smaller highways like America Online, CompuServe, and Prodigy--give us a glimpse into the world of convergent cyberspace. But these highways still have a progress-through-computing flavor and a bias toward text that resembles Highway 101 (educators and hobbyists) and Highway 202 (business users).

In fact, the real convergence is just now taking shape in the drawn-out courtship and prolonged foreplay taking place among cable companies and telephone companies, Madison Avenue ad agencies, entertainment companies, and Silicon Valley wizards.

The new highways where business, entertainment, and education converge will be more like game arcades, movie theaters, sports bars, and coliseums than any of the landscape we associated with traditional computing. The new highways will be group experiences, open-air

theaters in the round, and imagi-dome gatherings like rodeos, stock-car races, and rock concerts in which we interact with a kaleidoscope of real and simulated persons, species, and creatures in casual, ad hoc meetings to do work, play, and learn.

In a very short time the nicely clipped hedges that told us which PLU highway we were on in cyberspace and in real space will mutate into a virtual Br'er Rabbit briar patch.

If our reality is shaped by the road we travel and the company we keep, then we are in for some wild times ahead. I predict a decade or two of profound disorientation for all of us as we venture onto the new electronic highways of cyberspace that even now are being constructed. Businesses that are entering cyberspace through E-mail, LANs, and WANs are experiencing a flattening of their organizational structure. This occurs when night shift clerks start sending E-mail to their daytime superiors and machine operators look into the same electronic crystal ball as the president of their multimillion-dollar corporation.

This flattening and organizational warping is only the beginning. Get ready for some very strange bedfellows as you creep up the entrance ramp to these new highways. The old 3-D space of reality will soon become so convoluted that it will shatter forever. All of us will soon become consciousness packets whisking along multiple simultaneous highways of reality. Our lives will become fragmented into narrowcasting Freudian, Jungian, Mario Brothers, Nintendo, soap opera, toothpaste, Edgar Allen Poe capsules of vicariously shared being.

Maybe highways aren't even the proper metaphor for cyberspace. Perhaps as our lives become less physical and ever more virtual, the Cuisinart will become the metaphor for cyberspace. After all, discrete, real things such as carrots, ice cream, and bananas go into a Cuisinart, but after slicing, dicing, pureeing, and blending, these once-discrete items are transformed into something entirely new.

Cyberspace chefs, alert! Pop the tops of your blenders. Dump in your ingredients of reality. Fire up the cyberspace Cuisinarts!

It's time to begin liquefying each other's consciousness.

Gazette, August 1994

BEGINNER BASIC:
Variables Won't; Constants Aren't

By Larry Cotton

The above phrase is just one of many from a humorous screen saver called After Dark 2.0. This program is for certain non-Commodore PCs.

Even on those sophisticated computers, variables apparently don't vary when they should, and constants change values when they shouldn't. In defiance of laws attributed to Murphy and sages of similar ilk, things don't necessarily have to go wrong when you least expect them to; you only have to exercise care when handling variables and constants in a BASIC program. Let's review.

A variable really is (or should be) something that varies, or changes its value, while a BASIC program is running. A constant, on the other hand, is assigned a value usually near the beginning of a BASIC program and retains that value for the duration. Variables and constants can both have similar names such as X or J2 or FR or LQ.

Let's say we want to print ten blank lines (which effectively moves the cursor down to the middle of the screen). Here's one way to code it.

```
10 PRINT CHR$(147)
20 X=10
30 FOR T=1 TO X
40 PRINT
50 NEXT
```

This doesn't illustrate an efficient use of a constant, but it does show how X's value stays constant throughout the program: It's always 10. In addition to the constant X, there's also a variable in the above short program. T is the variable whose value varies (increments) as the program executes.

X and T are a numeric constant and variable, respectively; that is, they represent numbers. X always represents the number 10, and T sequentially represents the numbers 1 through 10.

Each time T's value increases, a blank line is printed on the screen (the cursor moves down one line). If, before the program runs, you type PRINT T and press Return, you'll see that T=0. All numeric variables are set to 0 before a program runs. In fact, just typing the word RUN zaps all variables to nothingness. After the program ends, if you type PRINT T, you'll see that T is now 11! (In a FOR-NEXT loop, the variable actually increments to 1 higher than the upper limit.)

There are also literal constants and variables known as strings. While numerics can represent 1 or 23 or 4.825 or 12562843, literals can

represent letters, words, phrases, sentences, or symbols (up to 255 characters long). Literal variables and constants are denoted by a following dollar sign. N$ (pronounced N string) can represent "S," "Sammy," or "Sammy is a great kid."

Here's an example of the use of literal constants and variables.

```
10 PRINT CHR$(147)
20 S$="NORTH CAROLINA"
30 PRINT "NAME THE CAPITAL OF "S$".[DOWN]"
40 INPUT C$
50 IF C$<>"RALEIGH" THEN RUN
60 PRINT "[DOWN]CORRECT!"
```

In line 20, "NORTH CAROLINA" is the constant S$. Line 30 prints the challenge using that constant. Then line 40 gets the user's answer, which is assigned to C$. If C$ isn't changed, it's a constant. If it should take on other values later in the program, C$ is a variable.

The 64 and 128 recognize only the first two characters of a constant's or variable's name. For example, CO=10 is the same as COLOR=10. Likewise, CO$="RED" is the same as COLOR$="RED". Numerics can also be named things like R5 or T7. If you code YY34=7654, it's not wrong, and you won't get a syntax error. Just remember that, to the computer, the variable is only YY.

You're probably thinking to yourself, "Can numbers be included in literal variables?". OK, you weren't thinking that, but the answer is yes, they can, although the converse isn't true. For example, A$="12" (note the quotation marks) is a valid statement. But no mathematical operations can be performed on it. Enter and run this program.

```
10 PRINT CHR$(147)
20 A=12
30 PRINT A + A
40 A$="12"
50 PRINT A$ + A$
```

You should see 24 and 1212. What happened? In line 30, 12 and 12 were mathematically added to get 24. In line 50, "12" and "12" were concatenated (think of this as adding strings) to get 1212.

SIMPLE ARRAYS

We'll review arrays more thoroughly in a later column, but you should be aware of this very common type of constant. (Once an array is filled, its contents don't usually vary.) Arrays make it easy to do certain things with numbers and words that you can't do with ordinary constants. Here's a simple example.

```
10 S=5
20 FOR J=1 TO 10
30 A(J)=S
40 S=S + 5
```

```
50 NEXT
```

This short program fills a one-dimensional numeric array with ten numbers in increments of 5. Run it and then type PRINT A(1) and press Return. You should see 5. A(2) is 10, A(3) is 15, and so on. While J varies (increments) from 1 to 10, S increases from 5 to 50.

On this disk is another example, called Password, which illustrates how variables (including one-dimensional string arrays) can be used. Here's the listing.

```
10 DIMA$(20)
20 PRINTCHR$(147)
30 PRINT" TYPE A PASSWORD, THEN PRESS RETURN.[DOWN]
40 N=N+1
50 A$(N)="":GETA$(N):IFA$(N)=""THEN50
60 IFA$(N)=CHR$(13)THEN100
70 PRINT" *";
80 B$=B$+A$(N)
90 GOTO40
100 PRINT"[DOWN] THE PASSWORD IS ";B$
110 PRINT"[DOWN] ITS FIRST LETTER IS "A$(1)
120 PRINT"[DOWN] AND ITS LAST IS "A$(N1)
```

Line 10 tells the computer the maximum possible size of the string array. In this case I have arbitrarily set it to 20. In line 40, N increases by 1 for each letter the user types. This variable N is sort of like a FOR-NEXT loop with an indefinite upper limit. In other words, the user can begin typing any word up to 20 characters in length; it would be impossible to use a FOR-NEXT loop without first asking the user to enter the word's length.

Line 50 waits for the user to type a letter. If he or she presses Return, line 60 sends control to line 100.

Line 70, while keeping the password secret, prints a space and an asterisk to indicate that a letter has been entered.

Line 80 is interesting. B$, which begins life as an empty (null) string variable, grows as each letter of the password is entered. I used an array to store each letter so that the word's first and last letters could be printed at the end of the program.

Make it a practice to sprinkle your programs liberally with constants and variables to define numbers, words, phrases, or a series of symbols that will be used repeatedly in your program. You will save typing, conserve computer memory, and enjoy faster-running programs.

Gazette, August 1994

...y All Directories

By Jim Butterfield

All Commodore 8-bit machines use the same machine language, and the calls to the Kernal operating system are virtually identical. From this, you might think that programs would be compatible across all machines, but there's a hitch. The various machines call for programs to be placed in different memory locations; and machine language programs do not move easily.

Programming relocation doesn't seem too hard--at first. Only those instructions using absolute addressing and its derivatives would need to be changed if a program relocated. That would rule out jump (JMP) instructions; but they could be replaced by branches. Branch instructions use relative addressing, which says, "Jump forward or back a certain number of bytes." Such jumps would still be valid no matter where the program had been moved.

Avoiding absolute addressing would also rule out using subroutine calls within your code, since jump subroutine (JSR) instructions would use this address mode. Perhaps most serious of all, we need absolute addressing to dig out the fixed data held within our program, such as text strings.

Finally, if a machine language program is moved from machine to machine, how will we know the address to which BASIC must issue a SYS call?

Despite these objections, we're going to write a small program that will run equally well on the 64 and the 128. Watch for the tricks.

The program is called Directory, and it's on the flip side. It loads from the Gazette Column Menu in 64 mode, but you can also run it from a 128 if you have one.

THE BASIC PROGRAM

It's not easy for BASIC to read a disk directory. The data comes in a curious format that's better suited to machine language.

Our project is the first step in a sequential-file reader utility. To let a user pick a file for reading, it's desirable to scan the disk and show all available files. That's all this program does.

The machine language code will be directly behind the BASIC program, but BASIC loads to address 2049 in the 64 and to 7169 in the 128 (hexadecimal 0801 and 1C01 respectively). The machine language program might be in either of two places. Things can get worse. Some programs, especially those using 128 graphics, can move the start-of-BASIC location somewhere else entirely!

We'll take care of this by having the BASIC program look for a specific string in memory. I've picked the characters "JIM B." The BASIC program will look for this pattern in the two expected places and then SYS to the appropriate address.

Note that the BASIC program, before calling in the machine language program with SYS, opens the directory file with "$0:*=S". A filename starting with $ calls for a directory; the 0: specifies drive zero; the * specifies any filename; and =S, not a well-known phrase, says that we want to see sequential files only. Note also that we use the unusual secondary address of 0. This forces the directory into the less-complex "fake BASIC load" mode.

THE MACHINE LANGUAGE PROGRAM

We'll specify some locations for our working data. The buffer to hold the directory will be at address hex 2400 and up ("DIRBF"). As we display each directory item, we'll assign to it an identifying letter (A, B, C, and so on). We do this with the aim of allowing the user to select a file. The current letter will be stored at hex 2600, and a work byte, mostly to save the value of the Y register, will be set aside at 2601.

```
dirbf = $2400
lettr = $2600
ysave = $2601
```

We'll assemble the program to address hex 976, although we know that the 128 version will move elsewhere. We'll place the string "JIM B" there.

```
*=$0976
 .byte $4a,$49,$4d,$20,$42
```

Now for the code. The directory was opened in BASIC, before the SYS call took place, as logical file 1. So the machine language program begins by hooking up the input stream to logical file 1. That's done with a call to CHKIN, address FFC6.

```
start ldx #$01
  jsr $ffc6
```

The data will arrive as a fake BASIC program. The first two bytes from a program file are always the load address. We'll throw the address away.

```
; dump the (fake) load address
 jsr $ffe4
 jsr $ffe4
```

Now we're at the first line of our fake BASIC program. You will recall that a directory listing always starts with a title line. We don't need it, so we dump it all: the two-byte link, the two-byte line

number, and the text which ends in a binary 0.

```
; first line.  dump the link
 jsr $ffe4
 jsr $ffe4
; first line.  dump the lnum
 jsr $ffe4
 jsr $ffe4
```

Six calls in a row to GET, address $FFE4--this is getting monotonous! And here comes number seven.

```
; first line.  scan to binary 0
fdump jsr $ffe4
 tax
 bne fdump
```

We're almost ready to read the directory data. But we have to do something else first. We set up the indirect pointer to the directory buffer and set the first identifying letter as A.

```
; set up filename ptr array
 lda #<dirbf
 sta $fd
 ldy #>dirbf
 sty $fe
 lda #$41
 sta lettr
; ready to read data line
```

Here's the loop where each directory line is read. We don't want the two-byte link, and we're not interested in the fake line number (which actually gives the file size in blocks). So we dump the next four incoming bytes.

```
; dump the link
rdirl jsr $ffe4
 jsr $ffe4
; dump the line number
 jsr $ffe4
 jsr $ffe4
```

Now we have reached the text section of the directory line. The name of the file is preceded and followed by quotation marks. Here's a trick. If we don't find quotation marks (hex 22) on the line, we must be at the last line ("... bytes free") of the directory.

```
; scan to either quotes or $0
qscan jsr $ffe4
 tax
 beq wrapd      ; last line!
 cmp #$22
 bne qscan
```

If we find the filename, we first print the selection letter followed by a colon.

```
; found filename.  grab it
 lda lettr
 jsr $ffd2
 inc lettr
 lda #$2e
 jsr $ffd2
```

Now we draw out the letters that make up the name of the file. We print them on the screen and copy them to the filename buffer. The filename ends when we see the second quotation mark.

```
 ldy #0
nmlp sty ysave
 jsr $ffe4
 ldy ysave
 cmp #$22
 beq wrpln  ; end of filename!
 sta ($fd),y
 jsr $ffd2
 iny
 bne nmlp
```

We put a Return character behind the filename, both on the screen and in the buffer. Then we scan the input for a binary 0, which is the end of the fake-BASIC line.

```
wrpln lda #$0d
 jsr $ffd2
 sta ($fd),y
lnswp jsr $ffe4
 tax
 bne lnswp
```

We must bump our indirect directory-buffer address by 17 (16 possible characters in the filename, plus a Return). Then we move back to read the next directory line.

```
 clc
 lda $fd
 adc #$11
 sta $fd
 bcc rddlp
 inc $fe
rddlp bne rdirl
```

When we reach the end of the directory listing, we put the input stream back to its default value with CLRCHN ($FFCC) and return to BASIC, where the file will be closed.

CONCLUSION

We didn't use any absolute addressing modes within our program, so it is indeed relocatable.

The code for Directory is a first step toward a general sequential file-reading program. As a matter of interest, I've include a polished program called Reader, along with its documentation. You'll recognize the directory-reading code, although it has been trimmed back slightly so that no more than 16 directory entries will be displayed. When the user selects a file, Reader will display it one screen at a time with a prompt for moving to the next screen. It's listed on the Columns menu as Machine Language (Part 2).

Gazette, August 1994

MACHINE LANGUAGE (Part 2): Reader

By Jim Butterfield

Reader is a machine language program that will scan a disk for sequential (SEQ) files, list those files on the screen, and then ask you which, if any, yof them ou'd like to list. Reader runs on the 64 or 128, in 40 or 80 columns.

You may optionally list a file to the screen, print it, or abort the process entirely. When a file is displayed on the screen, 23 lines are shown, along with a MORE message. Press any key to access the next group of 23 lines. The 23-line display was chosen to allow continuity from the previous screen.

Reader uses slightly gimmicky coding to allow both the 64 and 128 to run without extra overhead such as boot loads or code relocation.

Binary files may give Reader a little trouble, since this program simply takes from the file and delivers it to the screen or printer.

If you get color changes or screen junk, you are probably trying to look at what Reader considers a crazy file. Note that the 128 honors the TAB character [chr$(9)] and the 64 does not. So documents containing TAB may display differently on the two machines. Reader makes no distinction. Hey, it's just a simple little program!

Reader, on the flip side of this disk, is public domain. It's by Jim Butterfield, and was written in early 1994.

Gazette, August 1994

PROGRAMMER'S PAGE: BASIC Tools

By David Pankhurst

For every craftsman, tools are a necessity. Quality work demands quality tools, and we have a great selection for you here. Whether you're a novice or a veteran programmer, these tools will fit the bill.

NEVER NEW AGAIN!
Ever forgotten what you were doing and typed in NEW with your latest work only in the computer's memory? Oddly enough, this happens to a lot of people. According to Murphy's Law, the chance you'll enter an irreversible command by accident is directly proportional to the length of time since your last backup. The same principal applies to formatting your disk!

Paul Frazier of Lebanon, Pennsylvania, wrote a quick and simple routine that checks with you before doing something this drastic. It's a good idea to load this before you start important work.

```
0 REM  NEW COMMAND CHECK BY PAUL FRAZIER
1 :
10 J=53093:FORI=0TO152:READA:POKE I+J,A:NEXT:SYSJ
20 DATA173,4,3,141,254,207,173,5,3,
141,255,207,169,124,141,4,3,169,207,141,5,3
30 DATA96,173,0,2,201,78,240,3,108,254,
207,173,1,2,201,69,208,246,173,2,2,201
40 DATA87,208,239,169,13,32,210,255,
169,65,32,210,255,169,82,32,210,255,169,69
50 DATA32,210,255,169,32,32,210,255,
169,89,32,210,255,169,79,32,210,255,169,85
60 DATA32,210,255,169,32,32,210,255,
169,83,32,210,255,169,85,32,210,255,169,82
70 DATA32,210,255,169,69,32,210,255,
169,63,32,210,255,169,0,133,198,32,228,255
80 DATA201,0,240,249,24,201,89,208,3,
108,254,207,24,169,32,162,0,157,0,2,232
90 DATA224,81,208,248,24,108,254,207
```

The next time you type NEW, the computer responds with ARE YOU SURE? If you really want to clear whatever's in memory, press Y. If you type anything else, the NEW command is ignored. This program is on the flip side. It's called New Check.

GET THE HEX OUTTA HERE
One of the most useful aids to programming is an understanding of other number bases, such as hexadecimal or binary. Instead of base 10, hexadecimal (hex for short) works on base 16. Digits go from 0 to 9

and then to A, B, C, D, E, and F. From there comes 10, 11, and so on. Although it's confusing at first, this system makes understanding the computer much easier and is essential for working in languages such as Assembler and C.

But not all of us have the time or inclination (or lack of a social life, for that matter) for such intense study. And even if we master hex, individual numbers are still hard to figure out. As an aid, Arthur Moore of Orlando, Florida, wrote Hex-Dec Wedge. It patches into BASIC, ready to assist in conversion. To use, load and run.

The syntax is simple. Use a dollar sign ($) in front of a hex number to convert it to decimal. Use a number sign (#) in front of a decimal number to convert it to hex.

Valid numbers are in the range of 0 to 65535 (0000-FFFF hex) and must be on an entry line by themselves. You'll find this utility great for quick conversions.

This is probably a good point at which to bring up the subject of BASIC patches. They provide tremendous convenience, but at a price. Because many patches use the same vectors in the computer, there can be conflicts.

As an example, BASIC patches often use the error vector. When text unknown to BASIC comes in, it's treated as an error and is moved here. A patched routine can then check to see if BASIC's castoff is in fact a new command. But if two or more routines need the same vector or vectors, problems can ensue.

For that reason, keep the possibility of conflicts in mind when writing and using such routines. Test them together before doing important work on the computer so that nothing is lost if they don't get along.

Look for the Hex conversion program on the flip side. It's called Hex-Dec Wedge, and it runs from the Gazette menu.

TEENY TRACE

Sometimes you may want to trace a routine without loading a complete system. (A corollary of Murphy's Law states that when you need to use a debugging tool, it's almost impossible to load it!).

Teeny Trace, found on the flip side, can be loaded and run with a minimum of fuss. You start it by typing POKE 777,2. It then prints the line number of each statement as it executes.

```
10 REM TEENY TRACE: POKE 777,2 TO START        POKE 777,167 TO STOP
20 FORI=740TO754:READX:POKEI,X:NEXT
30 DATA165,58,166,57,32,205,189,169,32, 32,210,255,76,228,167
```

It will repeat line numbers if there are multiple statements on the

line. Since Teeny Trace prints the line number for each statement rather than for each line, it's handy for checking the number of loops completed or for where a routine quit.

REMOVE THE REMS

Good programming practice dictates that documentation go with every program. One of the best places to put instructions is in the program itself in the form of REMarks or REM statements.

The downside is that REM statements take up space, sometimes a great deal. Once you know how to run the program, that wasted space can be reclaimed with the following routine.

```
1 :REM REPLACE ALL REMS WITH ':'              NOTE: REM MUST BE FIRST ON
LINE
2 P=PEEK(43)+256*PEEK(44)
3 PRINT".";:L=PEEK(P+2)+256*PEEK(P+3): N=PEEK(P)+256*PEEK(P+1):IF N=0
THEN END
4 IFPEEK(P+4)<>143THENP=N:GOTO3
5 PRINT"[CLR][DOWN][DOWN]"L":": PRINT"P="P": G[SHIFT-O] 3 [HOME]";:
POKE631,13: POKE632,13:POKE198,2:END
```

This routine searches for lines beginning with REM and reenters them as a line with a single colon. The first colon on line 5 is the character used to replace the REM. It can be changed to another character or left out entirely to delete the line. As written, the routine changes rather than deletes lines since a GOTO or GOSUB may jump to a line that contained a REM.

If each REM line contains a lot of text, the memory saved can be significant. To check, print FRE(0) before and after running. To be safe, always keep a copy of the original program with the REMs in it, even if it's only a paper copy.

IS EVERYTHING ALL RIGHT?

This last routine addresses a particular need of my own. Whenever I write programs that use machine language, in the course of testing they usually run amok, poking data into places they shouldn't. Later, I'm always worried that my program has been changed, and I won't spot it.

This routine is a SYS call, returning the checksum of a BASIC line or group of lines. A checksum is a specialized count of the values in an area. It uses one number to reflect the state of an area of memory. If that number changes, then the memory area has also changed; if it remains the same, then it's likely the memory is unchanged.

```
10 REM RELOCATABLE BASIC LINE CHECKSUM      SYS X,<LINES>  -  <LINES>
```

```
AS IN LIST
15 FORI=49152TOI+162:READX:POKEI,X:NEXT
20 DATA32,121,0,201,44,208,3,32,115,0,
32,121,0,144,6,240,4,201,171,208,15,32
25 DATA107,169,32,19,166,32,121,0,240,
17,201,171,240,5,162,11,108,0,3,32,115,0
30 DATA32,107,169,208,243,165,20,5,21, 208,6,169,255,133,20,133,21
35 DATA169,0,133,172,133,173,169,5,133,
170,169,128,133,171,160,1,177,95,240
40 DATA75,200,177,95,170,200,177,95, 197,
21,208,4,228,20,240,2,176,58,200
45 DATA132,73,164,73,177,95,240,35,133,
6,162,8,6,6,106,41,128,69,173,6,172
50 DATA42,144,10,168,165,172,69,170,
133,172,152,69,171,133,173,202,208,229
55 DATA230,73,208,215,160,0,177,95,
170,200,177,95,134,95,133,96,208,175,166
60 DATA172,165,173,76,205,189
```

Unlike a typical checksum, this one ignores the line numbers and the hidden line pointers. This means that you can add and delete lines in front of or behind the line numbers or pointers or re-sequence the lines without affecting the checksum. To use this utility, type SYS49152 (or the starting address of wherever you relocate it) and enter the line number range exactly as you would for a LIST. For example:

```
SYS49152,10-50
SYS49152,100-
SYS49152,-9000
SYS49152
```

The last one performs a checksum for the whole program, and you can omit the comma. Note that a group of lines has a checksum different from each individual line. When programming, write down the checksum for each completed section. Then you can check it later to see if there have been any changes made to that section. If so, you can correct them.

This program is on the flip side. It's called Line Checksum.

Gazette, August 1994

GEOS: Defining Greatness

By Steve Vander Ark

A few weeks ago I ran into an old friend on GEnie, a friend of the kind that when you see his or her name onscreen you automatically smile. Her name is Brenda. She used to spend a lot of time on QuantumLink, but now she haunts the Commodore Flagship on GEnie along with a lot of other former Q-Linkers.

Brenda is one of the more intelligent people I've ever met, and she never shrinks from telling it like it is. When we happened to meet in the Flagship chat area, she took me to task right off the bat for something I'd written in a column some time ago. She claimed that I had said then that I considered geoFile to be the best database program around for the Commodore. Brenda asked me to explain how I could think that geoFile was better than Superbase.

You see, the one thing Brenda and I will never agree on is GEOS. I (obviously) love GEOS; Brenda can think of no good reason why anyone would use it. So when Brenda and I get together, we usually end up debating the merits of GEOS. I love those conversations, and I think she does, too. Someone even suggested once that we set up some kind of official online debate on the subject. Anyway, in this case she was certain she'd caught me with my prejudice showing. Superbase, she told me, has far more features and capabilities than geoFile. How could I possibly say that geoFile was better?

She really got me to stop and think. As a matter of fact, she made me think about my high school poetry teacher. Way back in the grungy seventies, when grunge meant something truly scuzzy and not just fashionably messy, I learned about all the great poets o' old from a woman who was old enough to have known some them personally. One day she presented us with a poem to read and then sat back waiting for us to be enlightened by its beauty. I thought it was dumb, and said so.

Granted, my saying so was rude, but the whole point here is what she said in response. She told me that it was a great poem not because it had wonderful word play or elegant rhythm, but simply because it had been written by a great poet. The ridiculousness of that idea, that a poem can be great solely because of its author, made enough of an impression on me that I remembered it when I was speaking with Brenda.

After all, Brenda was accusing me essentially of the same thing. In her opinion, I was calling geoFile a great program simply because it was a member of a great software family: GEOS. I though about my old poetry teacher. This was serious.

One thing about Brenda: she's always willing to listen with an open mind, even though I doubt I'll ever convince her of anything. But as we talked and as I explained my position, I realized that I was not guilty of the same mindless prejudice that I had disliked in my poetry

teacher. I re-read my column, and I could honestly say that I stuck by what I had written. As part of the GEOS integrated system of software, geoFile is the best.

I'd better explain that a little more. Since Superbase does in fact have more features, how can geoFile be better?

An impressive list of features isn't what makes geoFile such a powerful database program--although it's not at all wimpy. What makes it so powerful is the fact that it is part of a larger package of applications. It's designed to be compatible with geoWrite, geoMerge, geoPublish, and so on, and it is serviced nicely by such utilities as Photo Manager and GeoWizard, which allows nearly instantaneous task switching between them all. Superbase, while powerful indeed, is not part of a full range of integrated applications.

So which do you buy? That depends. If all you want is a database--no word processing, no spreadsheet, no graphics--and you need advanced power-user features, you'll probably want to go with Superbase. If that's your definition of great, then Brenda is right, and I'm turning into my old poetry teacher. On the other hand, if you want a suite of integrated applications that work together beautifully, sharing information, then geoFile takes the prize. In other words, it's really a matter of definitions.

I doubt that my logic will convince Brenda to buy a copy of geoFile, but that's OK. If she would convert, I'd really miss the great debates we have on GEnie.

As long as we're on the subject of geoFile, let me clue you in on a couple of excellent alternatives if you need only basic fact storage and search capabilities. The first is geoDex, which is part of the DeskPack Plus package available from Creative Micro Designs.

GeoDex is designed for storing names, addresses, and phone numbers. It looks like a stack of cards, rather like a Rolodex, and all the fields are permanently labeled. It has a few nifty frills, such as telephone dialing capability, but it's not really programmable for any other database uses.

The other alternative also stores information on what looks like a stack of cards. It's called CardFile and is part of the Power Pack Collection originally released by RUN magazine, now distributed by CMD.

This handy little program lets you add your own labels to a preset series of fields. Each CardFile database can be programmed with different labels, but the number and arrangement of fields stay the same. Data entry is a little klutzy because there are no tab keys to jump from field to field. You need to press Return to make the pointer appear and then click where you want to go. But for small database jobs, it's actually less work at times than using geoFile.

CardFile includes a set of basic search functions so you can find a particular piece of information quickly. I use CardFile in my classroom. My third graders are creating a database of the books they've read. I have the fields defined as Title, Author, Genre, and so on. CardFile is simple enough for them to use with little trouble. Of course, since it's a GEOS product, it's part of that integrated operating environment, which makes it an even more powerful choice.

I wonder what Brenda will say when she reads this column?

Gazette, August 1994

PD PICKS:
Tenpins and Pulsing Pictures

by Steve Vander Ark

I just got off the phone with Jim Green. If you read last month's column, you might remember that Jim is the fellow who sent me six of his disks of public domain and shareware programs. I called to thank him for the disks and to find out a little more about him.

As we talked, I realized that Jim is somebody special. He has given his time and effort to the rest of the Commodore community without expecting anything in return. He has spent hours and hours trying various programs from many different sources, selecting the best of the bunch and putting them on disks. He has even touched up many of the programs with additional colors, graphics, and instructions. His disks are a treasure trove of Commodore public domain programming. These are collections every Commodore user should have.

Jim has sent almost every disk he has made to a number of companies that distribute shareware and PD disks. So if you want these fantastic collections, you'll have to write those companies and ask if they have the Jim Green collections. Jim doesn't sell or distribute the collections himself, and I can't tell you where to order one of his disks. In the meantime, I'll keep putting some of the best programs in this column.

Before I unveil the two exciting programs for this month, however, I would like to mention a fear of mine. It really came to a head when I saw a program on one of Jim's disks called Blick. That program comes from the pages of Gazette. I know because I typed it in a good number of years ago and used it quite often. There it was among the public domain files; there was no way to tell that it was from a magazine.

As a matter of fact, it might not have been the same version. Sometimes a programmer will upload a program to a service like QuantumLink or GEnie and then upgrade it and submit it to a magazine later. There was no way for anyone to know.

My point is that no one, certainly not Jim, is trying to break any laws here. A copyrighted program could all too easily sneak into the pile. Before I place any program in this column, I make every effort to figure out whether the file is actually public domain or shareware. I list the code, looking for addresses. I read every screen, including the help screens. Sometimes I find names, occasionally I find addresses, and once in a while I even find a telephone number. It's rare, however, that the information proves to be current. Most often, I find a QuantumLink screen name which has long ago expired. That does me no good at all. I try to contact the author whenever possible.

I call the phone numbers, and I try to E-mail screen names. Along the way I've talked with some interesting people, but seldom have I found

the programmer whose handiwork I'm showcasing. If you ever see a program in this column whose author you know (and by that I mean more than a screen name or handle), drop me a line! I would like very much to contact any author whose work I mention. You can reach me on GEnie, where my screen name is S.VANDERARK, or by means of the Internet, where my address is S.VANDERARK@GENIE.GEIS.COM.

OK, that's enough business for one column. Let's get into the programs!

TENPINS
Here's a fine example of the programs Jim has on his disks. This is one great game. The graphics are a little plain, but it's a lot of fun trying to make the little bowler with the big belly in the brightly colored bowling shirt try to catch that head pin just so.

Getting a strike is not easy. As near as I can figure, the program must put a slightly different spin on the ball here and there, so you can't just sit in the same spot and roll strike after strike. Heck, I've only managed two or three strikes over eight games.

I suppose you could say that's because I bowl on the computer about as well as I bowl in real life--not very well. Graceful hook shots are not my strong suit, either on the monitor or at the alley. I'm even lousy at keeping score, but this program thankfully takes care of that, tallying up the points as you roll along.

As I said, the graphics are fine but nothing spectacular. Another minor flaw is the lack of sound; bowling, after all, is nothing if not loud. All in all, however, this bowling game is entertaining enough to keep you happily rolling for hours.

PULSATING PICTURES II
by George Trepal
What's the most impressive bit of Commodore graphics programming you've seen lately? Is it myriads of sprites (more than eight, that is) zipping around the screen, or is it patterns changing in the border area?

Here's one for you. Pulsing Pictures is a graphics utility which lets you use character graphics to create a picture or display. "Big deal," you might say. We all know that character graphics are simple; you type them right on the screen from the Commodore keyboard.

Pulsing Pictures takes a giant step beyond simple character graphics. This program gives you a few new tools to use. Create a screen with this program, and I guarantee that you'll impress everyone who sees it. Did I say "impress"? No, you'll knock their socks off.

The secret is in the characters Pulsing Pictures gives you to use. Most are the same ones you see on the graphics keys, but there are a few extras. All the number keys and some of the punctuation keys will produce characters that are animated.

You really have to see this to believe it. Instead of presenting a set of horizontal bars, for example, Pulsing Pictures will give you bars that move upward. If you make a block using several of these, you get an effect of a moving series of lines. A black dot animates between small and large, giving it a blinking effect. Other characters seem to glitter or wriggle. The resulting graphics are stunning.

You can, for example, type a series of characters to make a border around some text. Set the program to animate, and the text is suddenly surrounded by a pulsing, shimmering border. All the color commands are in effect, so you can make rippling blue water or twinkling yellow stars. The f1 and f7 keys save and load screens of these graphics, so you can show off your work for everybody. The English pound sign key stops the program.

Look for a demonstration program on the disk that shows off some of Pulsing Pictures' capabilities. Load the sequential file PULSING DEMO.

If you're at all interested in creating exciting graphics on your Commodore, you will find this program irresistible.

Remember, if you have some great public domain or shareware programs that you'd like to share with Gazette readers, send them on disk to COMPUTE's Gazette, PD Picks, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408.

Gazette, August 1994

CREATING GEOS FONTS

By Harold Stevens Jr.

You are reading a book, a magazine, a newspaper, or a brochure, and text in an unusual typeface catches your attention. It's pleasing to your eyes, and you think, "Boy, wouldn't it be great to have this fancy type among my collection of GEOS fonts?"

Then you realize that you are not artistically inclined, nor do you have a whole lot of time to sit down and create a geoPaint file to draw the letters, numbers, and other characters that make up this font. Then you'd have to store each character in a photo album, and then you'd still have to convert the photo scraps into a font by using your favorite font-creating application, such as geoFont. You could wait until someone else comes along to create this font and add it to the downloading files of an electronic bulletin board, but that might never happen.

Don't give up. I've discovered a shortcut that requires no artistic endeavor and very little time. You can create GEOS fonts by scanning them into your 64 or 128 computer with the Handyscanner 64. The Handyscanner 64, distributed by RIO Computers, is a hand-held IBM-type scanner that plugs into a cartridge that connects to the user (modem) port of your Commodore computer.

What the Handyscanner does is take any printed matter, such as a drawing or photograph, and convert it into a digitized image. The Handyscanner can enlarge an image as much as 300 percent of the original or reduce it as much as 33 percent.

The scanner also has three built-in dithering settings, plus the black-and-white mode to copy images. It can scan images up to a maximum of 2.5 inches wide and 5.5 inches long. If you have the Pagefox cartridge, also from RIO, you can scan an entire length of a page.

To start, you first find the typeface that you like. It might be a collection of printer fonts in a book from your local bookstore or library or sheets of rub-on decals that can be found in an art supply store. Another source of fonts are clip art books. These sometimes contain special lettering, numbers, and other characters that are used in advertising layouts.

I have found several excellent books at the local library, including "Type and Typography, The Designer's Type Book," by Ben Rosen and "The Type Specimen Book," by V & M Typographical. Each of the aforementioned books contains as many as 500 different fonts, providing plenty of typefaces to choose from.

Next, I recommend that you buy a proportional scale wheel and a special ruler called a pica pole, both of which can be found in art

supply stores. I would also suggest you buy a small carpenter's square, an L-shaped ruler that can be found in any hardware store.

A proportional scale wheel will help you determine the percentage that you need to enlarge or reduce the font to make it the size you want. If a font on a printed page is 48 points in size, you might want to reduce it to 18 points. A quick twist of the wheel will tell you that reducing 48 points down to 18 points means that you have to downsize the font to 37.5 percent of its original size.

A good pica pole will have a scale for measurements in picas, which is the unit of measurement used by the printing industry, and also a scale for measurements in inches. It should also have scales in metric and points, the other units of measurement used by printers. The point side of the ruler is what is used to measure how large the typeface is. One thing you should remember is that 12 points equal 1 pica and 6 picas (72 points) equal one inch.

You should always allow at least one point of space above and below a letter when measuring your font. This leading ensures that you have room for the tails of letters like "g," "j," "p," "q," and "y." Plus, you should also leave space to the left or right of a letter. This kerning makes sure that letters will not run together. It would be a good idea to keep in mind that letters such as "M" and "W" are usually twice the width of the regular letters. Letters such as "I," "i," and "l"; other characters such as punctuation marks; and spaces are usually half the width of regular-size letters.

The carpenter's square is used as a straightedge to keep the Handyscanner's movement straight when moving down the page. This way, you can make sure the letters that you are scanning will not be crooked along the sides, top, or bottom.

Another helpful device is a point-size gauge. This is usually a chart printed on clear acetate that shows how large a letter or other character is in points. By placing it over a printed page, you can quickly measure the size of the type in points.

SCANNING

To get to work, plug in your Handyscanner cartridge, connect the scanner, and run the scanner program that comes with the package. If you use the Handyscan software on another device, such as a 1581 disk drive or a RAM expansion unit, go ahead and boot it up from either of those devices. I keep Handyscan stored in my RAMLink for instant access.

Next, at the prompt asking if you want to erase and clear the graphic memory before you start, press the Y key for yes and hit Return. You'll notice that the Handyscanner screen looks similar to that found in geoPaint, although the two programs do not function the same way. To start the scanning routine, press the f1 function key.

On the screen you'll see the prompt Zoom-Factor (33-300%). This is

where you decide whether you want to enlarge your type or reduce it. Using our earlier example, let's say that we want to reduce a 48-point original to 18 points. We've determined that the percentage of reduction is 37.5 percent, so we type in 37.5 at the prompt and hit the Return key. Check your settings on the left side of the image scanner, making sure the dithering switch is on the B&W setting and that the bright and contrast settings are set for scanning line art.

Place the image scanner on the printed page, lining the scanner's side against the inside of the carpenter's square; push the button on top; and pull the scanner down the page, scanning the type you want. When you have completed the scan, press the button on top and hit the Run/Stop key to start the digitizing process.

Once the process stops, the Handyscan graphics screen will display the font that you have scanned. It will be an 80-dots-per-inch bitmapped image. Move the cursor down to the Save icon, click on it, and give the file a name to save it to disk. Go back and keep repeating this process until you have scanned and saved all the letters, numbers, and punctuation marks.

CHANGE FONT SIZE

If you want more than one size of fonts, just repeat the process but change the percentage of enlargement or reduction to the desired font sizes. For regular fonts used with geoWrite and geoPaint, you should limit your font sizes to those between 9 and 24 points. For larger type used in gepPublish, you may want to use fonts up to 48 points in size. After scanning and saving all the letters, numbers, and other characters in the various sizes that you need, close down the Handyscan program and return to BASIC.

CREATE FONTS IN GEOS

Next, I am going to tell you about two ways to create fonts while in the GEOS environment, particularly after you've converted the Handyscan graphics file into a geoPaint document. The first method, after booting up GEOS and converting the Handyscan file with the Handyscan Convert application, is to open the geoPaint document and create a photo album with the Photo Manager in which to store photo scraps of each letter, number, or punctuation character. This time-consuming method involves closing the geoPaint document and opening the geoFont application and pasting each letter, number, and other character into it.

A quicker method would be to use gateWay, an alternative GEOS desktop from Creative Micro Design, and its Switcher application. Switcher is a multitasking program that allows a 64 or 128 user to use two separate GEOS applications at the same time.

Although not a true multitasker, Switcher enables you to work on one application and then switch to another one. To successfully use Switcher, you need at least 512 kilobytes of memory in a RAM expansion unit such as the Commodore 1750 REU, geoWork's geoRAM, CMD's RAMLink, or PPI's RAMDrive. With Switcher, you can open the geoPaint document

converted from the Handyscan file; make a photo scrap of each letter, number, or other character; and then, by holding down the Commodore key and hitting the Restore key, switch to geoFont to paste each scrap into a geoFont file.

For the purpose of this article, I created MegaFuture from a 48-point Futura medium font. Futura is a popular sans-serif font used by printers for headlines, book and brochure titles, and on advertising signs.

I am calling this font I created MegaFuture instead of MegaFutura because the names of many fonts are copyrighted although the designs are not. You will find this to be true in many of the fonts used in desktop publications. Roma, MegaRoma, and LW Roma are GEOS names for the Times Roman font, and California, MegaCalifornia, and LW Cal are names for a typeface otherwise known as Helvetica.

I picked 48-point Futura medium as the font to scan because its size is right for what I need in using the geoFont application. Its letters are just tall enough and wide enough to be incorporated into a geoFont application. Taking the pica pole, I measured the letters, which were 35 points from top to bottom of the capital letters and were also 35 points from one side to the other of the wider ones such as "M" and "W."

It's important to measure letters because many fonts, such as Eurostile Extended or an Old English font called Fraktur, are too wide to fit into the geoFont display screen window at 48 points. This means that you will not be able to get wider fonts like these to fit unless you reduce them by as much as 80 percent when you scan them.

Since the scanning window of the Handyscanner 64 is only 2.5 inches wide, I had to scan the page with Futura medium three times at 100 percent, once on the left side, once down the middle, and then on the right to capture all of the letters, numbers, and other characters in this font. After each pass of the scanner, the Handyscan graphic was saved to disk to be converted into a geoPaint document.

Once I was satisfied with the scan, I left Handyscan and booted GEOS.

Once you're in the GEOS environment or, in my case, in the gateWay environment, click on Handy Import to convert the Handyscan image to a geoPaint document. Handy Import is a shareware graphics conversion program created by Joe Buckley, also known as Red Storm on QuantumLink. He also created the Graphic Storm bitmap conversion program. Now that the three Handyscan files that I saved are geoPaint files, I'm ready to complete the font creation process.

Upon opening the geoPaint document that contains the scanned Futura medium fonts, I move the first letter from the upper left corner to another location onscreen to free up that corner for copying. This way when I use the editing feature of geoPaint and copy the image of a letter, number, or other character into Photo Scrap, it will fit

exactly into the geoFont screen when I paste it in. If I don't do this, only part of the image will appear in the geoFont viewing screen.

In using the method I mentioned earlier to create fonts, I clicked on Photo Manager to create a photo album. Since I am starting at the beginning of the alphabet, I paste "A" into the album, close the album, and copy the next letter. I keep copying from geoPaint and pasting to Photo Album until all of the letters are stored in the album. When finished, I close the geoPaint document and open the geoFont application.

Please note that the Photo Manager I am using is version 2.0. With it I can assign a name to each letter, number, or other character that I paste into the album. The name assigned is the letter of each alphabet, number, or punctuation character, such as A for the letter "A," 1 for the number 1 and ? for the question mark.

With geoFont I click on the Create icon to create the new font called MegaFuture. Once inside geoFont I reverse the process from what I did when I worked in the geoPaint file. I open Photo Album and copy each image as a photo scrap and then paste it into the appropriate letter, number, or punctuation character in geoFont. I keep doing this until I come back to the beginning of the Photo Album file.

SWITCH

This, however, can be a tedious process, and that's the reason that I decided to switch to Switcher in gateWay. To activate the Switcher routine, you must hold the Commodore key while hitting the Restore key immediately upon booting gateWay. The Switch is now active and ready to go.

As before, I open the geoPaint file containing the scanned fonts, and I copy the letter into Photo Scrap. Next, I activate the Switcher mechanism by holding the Commodore key and hitting the Restore key, and once again I am in the gateWay menu, where I open the geoFont application and create the MegaFuture font as before. Then I paste the photo scrap of the letter directly into geoFont. I hit the Switcher key again to go back and make a scrap of the next letter in the geoPaint file and then switch back to the font maker to paste it in. This process is repeated until all the images of all the letters, numbers, and other characters are converted into the new font.

Whichever method you use to create a font, you will probably have to do some minor editing because the scanner may leave out minute portions of the letter you are scanning. During the editing process in geoFont, you will also have to determine the kerning or spacing between the letters. In the case of MegaFuture, I left no space on the left side and three pixels on the right of each letter.

One word of warning: Remember when I advised against scanning fonts that were wider than the geoFont display window? I found out the hard way that scanning wider fonts can make the geoFont file much too large

to use. Originally, I scanned the Fraktur font at 80 percent of the size presented in a book of fonts I found at the library.

While the scanning process and pasting process went well, I ran into trouble when I tried to use the fonts in a geoPublish application. The size of the area between the letters "A" through "O" jumped to 4455 bytes. I noticed that anytime an area in a geoFont file exceeds four kilobytes, the file will not show up onscreen, nor will it print.

Of course, you can use either method with another GEOS-type font maker, such as Jim Collette's GEOS Font Maker or the one that comes with CMD's Perfect Print LQ fonts. However, regardless of which font maker you use, the best part of the whole process is using the Handyscanner 64 to do the artistic work of creating a font.

Now, if only somebody could create an optical character reader program for the 64 and the Handyscanner.

Harold Stevens Jr. is an avid GEOS user and is an editor with the Columbus Messenger Newspapers. He is also the former editor of COCUGazette, the official newsletter of the Central Ohio Commodore Users Group. Stevens can be reached under the handle of H.STEVENS5 on GEnie or via Internet at H.STEVENS5@GE.GEIS.COM.

Gazette, August 1994

THE BLUES BROTHERS

Reviewed by Don Radler

If you like rock 'n' roll, think Dan Ackroyd deserves an Oscar, and spend half your waking hours playing Super Mario, you'll just love The Blues Brothers.

This is a joystick-controlled arcade adventure game of the platform variety, with five levels of action on 300 smooth-scrolling and very busy screens. It's for one or two players taking the role(s) of Jake and/or Elwood, the original Blues Brothers. Behind all the action is music from the movie soundtrack.

The premise is that the Brothers' equipment has somehow disappeared. You have to find it, one item at a time, if you want the brothers to be able to play their next concert. (If you don't want them to play, you probably won't be playing this game.) Along the way, you collect records for points, and hearts, and hats and shades--more about these later.

The instructions tell you to remove all cartridges before loading the program, but, being as rebellious as the characters in the movie, I tried loading and running the game with my Epyx FastLoad in place--no help, but no problem either. When you load the program, it takes about two minutes and 40 seconds with or without a cartridge to speed it up.

The directions also tell you that "immediately after the introduction page, you will come to the Character Choice page." In real life, this takes an additional 100 seconds. The Quick Start Card that comes with the program makes several such minor errors in telling you how to run the game on a 64, but you'll spot those right away and ignore the bad advice. If only the documentation writers had the talent of programmers!

With the space bar you highlight one character or the other character and then activate the one you want with the fire button. Next, you move to a map of the town, which is pretty but not particularly edifying, and then on to the game itself.

An item bar at the bottom of your screen keeps score for either one or two players. Shown are a picture of the character in play, the number of lives left, the number of records accumulated, and the number of hearts remaining. With two players, the bar is divided in half and shows the same data for each character. You may have to adjust the brightness on your monitor to read the item bar; it was clearer on my Commodore 1802 than on my Gold Star, although the Gold Star usually provides a better picture.

The screens themselves are something else--they're beautifully rendered and full of action. Behind everything is that soundtrack; you

can turn it up or down, depending on your taste in music. Don't turn it off, however; the sound effects are realistic and fun.

On level 1, you have to find a missing guitar. Joystick action (up plus the fire button) sends you into the appropriate room, where you snag the guitar with a push on the fire button. Stand in the doorway, move the joystick up and press the button again, and you're back on the street.

On level 2, it's a microphone that you need. Level 3 holds an amplifier. (What would a rock concert be without amplifiers?) Level 4 has a concert poster, and level 5 has a concert permit.

To move from one level to the next, you first must find and snare the appropriate item. If you don't snare it, a question mark appears in the items bar, and back you go to look for the missing piece of equipment. The order of play from level to level can't be changed.

Along the way, with disturbing regularity, you encounter folks who want to interfere with your quest, doing such things as shooting at you. Among them are a cop, a jailer, a burnt-out hippie, an angry waitress, and some babies in carriages. You can jump over them, or if you were wise enough to pick up one of the large boxes that litter the landscape, you can knock them out by throwing a box at them. (Just look 'em in the eye and press the fire button.)

There are elevators to take you up and down and people-movers to slide you sideways. There are also hazards such as pools to be swum through. (The animation here is great!) You also come across helpful props such as balloons to loft you upwards and umbrellas to float you gently back down.

On the way, you gather as many records as you can; 100 of them give you an extra heart. Hearts increase your energy level, which is good. Hats and dark glasses, the Blues Brothers' signature wear, add an extra life, which is even better.

Making all this work takes some practice, and being sent back to the character-choice screen means a 30-second wait for a partial load. You can pause the game by hitting P or end it by hitting Q. My teenager usually opted for the P and I for the Q. There seems to be a generation gap between those two letters, and the game's appeal probably depends on which side of that gap you're on.

TITUS SOFTWARE
20432 Corisco St.
Chatsworth, CA 91311
(818)709-3693
$39.95

Gazette August 1994