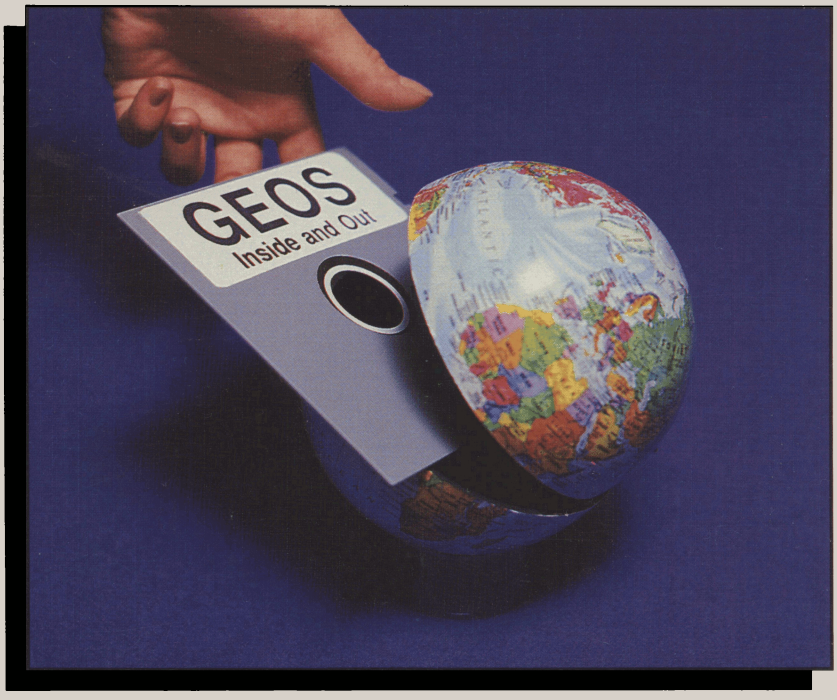
GEOS INSIDE AND OUT

An introduction to GEOS,
its applications and internals



Abacus

A Data Becker book

GEOS INSIDE AND OUT

An introduction to GEOS, its applications and internals

By M. Tornsdorf and R. Kerkloh



A Data Becker Book

Published by

Abacus

Second Edition, Third Printing, May 1988
Printed in U.S.A.
Copyright © 1986, 1987, 1988 Data Becker GmbH
Merowingerstraße 30
4000 Düsseldorf, West Germany
Copyright © 1986, 1987, 1988 Abacus Software, Inc.
5370 52nd Street, SE
Grand Rapids, MI 49508

This book is copyrighted. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written permission of Abacus Software or Data Becker, GmbH.

Commodore 64, Commodore 64C, Commodore 128, Commodore 1541, Commodore 1351 and Commodore 1571 are trademarks or registered trademarks of Commodore Electronics, Ltd.

GEOS, deskTop, geoPaint, and geoWrite are (C) Copyright Berkeley Softworks 1985.

ISBN 0-916439-86-X

Foreword

GEOS (Graphic Environment Operating System) has changed the face of home computing. Until now, home computer users were pretty much on their own once they bought the computer, and either had to shell out the money for ready-to-run programs, or go through the tedious process of learning a computer language and do their own programming. But GEOS comes with the 64C when you buy it—you're ready to run it the moment you get home.

Since we follow Commodore closely, we were wondering just what difference there was between the "new" 64C and the old C-64. When we saw GEOS, we were overjoyed. The 64 now has an advanced user interface similar to GEM, an interface previously available only in much higher-priced computers. Most commands don't need to be typed in. Instead, graphic symbols are used. Now when you want to delete a file, you no longer need to type:

```
OPEN 1,8,15,"S0:NAME":CLOSE1
```

Now you just move an *icon*, or symbol that represents the program, and "deposit" it in to another icon that looks like a waste basket. It's hard to believe what the developers of GEOS have done with the 64. We've spent many hours in front of our 64's learning about the the new GEOS system, and we're constantly amazed by its capabilities. GEOS is the user-friendly interface that beginners have been waiting for.

A few points about our book:

1. If you have bought this book because you are still hesitant about buying a computer and want some information about the 64C and GEOS, to a great extent we wrote this book for you. There is a thorough introduction to working with GEOS, with scores of actual screen illustrations, to help you make your decisions about your computer purchase.

2. If you are among those who already have a 64C (or an older C-64) and GEOS, you will also find the introduction to GEOS valuable. We'll be pointing out all the things that GEOS can do. Give your creativity free rein; we'll try to spark your imagination with the suggestions and hints we have to offer. GEOS will allow the artist in you to run wild. Keep this book handy whenever you work with geoWrite and geoPaint.

Do you have a favorite BASIC program in '64 format—or even a machine language program—that you would like to run under GEOS? Do you want to create your own custom icons for your programs running under GEOS? Or do you want to add your name to the info box to identify yourself as the author of a program? If you answered yes to any of these questions, then Chapter 5 should be of special interest to you. It contains the pertinent information and the **FILEMASTER** program, which enables you to do all of these things and more.

3. Some of you may be programmers who would like to combine your own program routines with GEOS. This book should clarify the steps you need to know. We have saved you the trouble of doing the "grunt work" of finding GEOS file structure, joystick control, etc. Apply your energies to the programming, and use the information and assistance contained in this book as reference.

   We think that it's well worth the effort to write programs for and under GEOS, and we want this book to support later adaptations of the GEOS concept. However, since GEOS is a completely new operatimg system, some memory locations specified in this book may differ by a few bytes from those of your system.

We wish to thank Commodore for its friendly support. In particular, we would like to thank Dr. Kittel, who helped us along throughout the project.

Manfred Tornsdorf
Rüdiger Kerkloh
Muenster, Germany
July 30, 1986

Table of Contents

Foreword		*iii*
Chapter 1	**Using this book**	**1**
Chapter 2	**GEOS for beginners**	**5**
2.1	Backup copies and work diskettes	7
2.2	Preparing the work diskette	11
2.3	The perfect setup: The Preference Manager	17
2.4	Work diskettes: Deleting and adding files	19
2.5	Applications: geoWrite	24
Chapter 3	**GEOS in detail**	**29**
3.1	deskTop: The window	31
3.2	deskTop: Menu and items	33
3.3	deskTop: Functions	39
3.3.1	deskTop functions performed with icons	39
3.3.2	Keyboard	40
3.3.3	Clicking	40
3.3.4	The remaining functions	41
3.4	geoPaint	42
3.4.1	Starting geoPaint	42
3.4.2	The menu	44
3.4.3	The toolkit	49
3.5	geoWrite	57
3.5.1	Starting geoWrite	57
3.5.2	The menu	59
3.5.3	Entering text	64
3.5.4	Formatting text	65
3.6	Accessories	67
3.6.1.	Alarm clock	67
3.6.2	Calculator	70
3.6.3	Notepad	71
3.6.4	The Preference Manager	73
3.6.5	The Photo Manager	76
3.6.6.	The Text Manager	80
Chapter 4	**GEOS applications**	**87**
4.1	Making diagrams using geoPaint	87
4.1.1	Pie chart	88
4.1.2	Bar chart	93
4.1.3	Line graph	98
4.2	Organizing and planning with geoPaint	100

4.2.1	Room layout	100
4.2.2	Laying out a garden	106
4.3	Electronic circuits	108
4.3.1	PC Board	108
4.3.2	Schematics	111
4.4	GEOS and educational applications	120
4.4.1	Diagrams in reports	120
4.4.2	Teacher's forms	121
4.4.3	Seating charts	122
4.4.4	Science class diagrams	122
Chapter 5	**GEOS Tricks and Tips**	**123**
5.1	Tips and Tricks for the GEOS user interface	125
5.1.1	Problems loading GEOS	125
5.1.2	deskTop "Do's and Don'ts"	128
5.1.1.1	GEOS file management and printing	129
5.1.3	Tips and Tricks with geoPaint	131
5.1.4	Tips and Tricks for geoWrite	137
5.1.5	Tips and Tricks for Notepad	144
5.2	Programs in GEOS format	148
5.2.1	FILEMASTER	149
5.2.2	Description of the FILEMASTER program	160
5.2.3	The FILEMASTER menu	171
5.2.4	Using FILEMASTER	175
5.3	The function of the real-time clock	178
5.3.1	Programming the clock	178
5.3.2	The routines in GEOS	180
5.3.3	The time at a glance	187
5.3.4	The foreign aberration in the GEOS KERNAL	197
Chapter 6	**Inside GEOS**	**201**
6.1	The single step simulator	203
6.2	Window techniques	218
6.2.1	Characteristics of windows	220
6.2.2	Custom windows in GEOS	225
6.2.3	Three examples from GEOS	234
6.3	The system interrupt	244
6.4	Job loop/ Job structure	247
6.5	The GEOS file structure	252
6.5.1	File management under Commodore DOS	252
6.5.2	A file entry under GEOS	253
6.5.3	The INFO sector	255
6.5.4	The border and GEOS format V1.0	258
6.5.5	File type / File structure	262

6.6 Writing your own GEOS programs 267
6.6.1 How to load your own programs 267
6.6.2 How to extend GEOS with your own programs 268
6.7 Memory layout and memory locations 274
6.7.1 Memory map 274
6.7.2 Important memory locations 275

Glossary 279
Index 287

List of Figures

Figure 1	The deskTop	11
Figure 2	**disk** sub-menu	12
Figure 3	Rename window	13
Figure 4	info box	15
Figure 5	`Can't delete` dialogue box	16
Figure 6	Preference Manager	17
Figure 7	Format window	19
Figure 8	geoWrite south of the border	20
Figure 9	Ready to copy	22
Figure 10	Copier dialogue box	23
Figure 11	geoWrite dialogue box	25
Figure 12	Letter to Deb	26
Figure 13	`Font` sub-menu	27
Figure 14	Point sizes in GEOS	28
Figure 15	The deskTop window	31
Figure 16	Files arranged by `SIZE`	35
Figure 17	geoPaint	43
Figure 18	`change brush`—32 different brushes	48
Figure 19	Edit box	49
Figure 20	Ruler and lines	51
Figure 21	Repeated effect with brush	52
Figure 22	Text processing with GEOS	53
Figure 23	Pattern box with 32 patterns	56
Figure 24	The geoWrite window	58
Figure 25	Marked geoWrite document section	61
Figure 26	Graphic in geoWrite document	62
Figure 27	Alarm clock	68
Figure 28	Calculator	70
Figure 29	Notepad	72
Figure 30	Preference Manager	73
Figure 31	Photo Manager—starting menu	77
Figure 32	`photo manager`—Graphic for `To Deb`	78
Figure 33	Text and graphics with `photo manager`	79
Figure 34	Marking text for `text scrap`	80
Figure 35	`Empty text album`	81
Figure 36	Marked text in the Text Album	82
Figure 37	Title and circle	89
Figure 38	Chart segments with fill patterns	90
Figure 39	Chart segments with actual percentages	91
Figure 40	Finished pie chart	92
Figure 41	Starting the vertical axis	94

Figure 42	Zeropoint and horizontal axis with divisions	95
Figure 43	Two sides of a bar	96
Figure 44	Section of the finished bar chart	97
Figure 45	Bar chart in `preview` mode	98
Figure 46	Line graph	99
Figure 47	Walls and dimensions	101
Figure 48	The bed	103
Figure 49	Right half of the bedroom	104
Figure 50	Fully furnished room	105
Figure 51	Garden	107
Figure 52	`PC Board`	109
Figure 53	An IC in `pixel edit` mode	110
Figure 54	Multivibrator schematic	112
Figure 55	Ohm symbol (Omega)	113
Figure 56	Starting the schematic for the multivibrator	114
Figure 57	Mμ	115
Figure 58	Capacitor and transistor	116
Figure 59	Copying transistors	117
Figure 60	Mirroring potentiometers	118
Figure 61	`preview` of the schematic	119
Figure 62	Russian economic structure—1913	121
Figure 63	Using patterns	134
Figure 64	Dotted lines 101	135
Figure 65	Dotted lines 102	136
Figure 66	Dotted lines 103 (mixed with patterns)	137
Figure 67	Unformatted letter	138
Figure 68	Left and right margins	139
Figure 69	Highlighting words	141
Figure 70	`Delete page break?`	143
Figure 71	Notepad as daily calendar	144
Figure 72	Notepad as address book	145
Figure 73	Notepad as a recipe file	146
Figure 74	Constant display clock	188
Figure 75	The Stepper	215
Figure 76	An output window: error box	218
Figure 77	Dialogue box	219
Figure 78	Redrawing a window	221
Figure 79	Odd-shaped window	222
Figure 80	Different type styles	222
Figure 81	Patterned shadow	223
Figure 82	Two-button dialogue box	224
Figure 83	Custom window	229
Figure 84	`window` demonstration	243

Chapter One

Using This Book

Using this book

About the different versions of GEOS

At the time of this writing, there are three different versions of GEOS in use throughout the world: V1.2, V1.2+ (V1.2 with a V1.3 deskTop) and V1.3. Berkeley Softworks will upgrade GEOS even further as time goes on.

The programs in this book have been modified to work with GEOS V1.2 and V1.3. Since this book's machine language programs access the GEOS jump table instead of specific addresses (see Section 6.2.3 of this book and Section 5.6 of Abacus' *GEOS Tricks & Tips* for information), these programs should also work with versions of GEOS that will appear on the market later on.

The text references to GEOS throughout this book are intended for users of GEOS V1.3 and beyond.

Presentation

Don't be surprised when you run into similar formats, instructions and formulas throughout this book. We do this to ensure your success with this book. GEOS was systematically designed, and we have taken great efforts to make our book as systematic and thorough as possible. You'll see that the chapters are all arranged in a similar format. You'll notice quite a bit of this overlapping in our book regarding individual program sections. We did this not so much for creative purposes, but to ensure that the reader understands the material completely.

Repetition

There is a second, subtle characteristic you will see as you read this book. Information is often repeated from chapter to chapter, even though stating the same information once in a single chapter would be sufficient. We did this to save you page-turning. We know how difficult it is to work with one hand trying to find the page, and the other trying to handle the computer.

We essentially wanted each chapter to be an independent section of the book, so you wouldn't have to turn a lot of pages.

Illustrations

As you thumb through this book, you may be wondering why we included so many illustrations, and whether all these screen dumps were necessary when you can see for yourself what's on the screen. We decided that this book, or at least a good part of it, would be designed for simply reading about the capabilities of GEOS. The illustrations also help to user to read now, see the results immediately, and try the procedures later on his computer.

Another advantage to all these illustrations is that every time your actual screen appears, you can compare it with the illustration, to make sure that everything is correct.

Remarks

You will see some differences between most of the illustrations in this book and your own screen. This is because hardcopy (a printout of the screen) does not display sprites. Different objects in GEOS—for example, the pointer, the small rectangle in the page pointer, and current color marker in color list—are constructed as sprites, and therefore do not appear on the screen. However, don't be concerned about the differences between our illustrations and your screens. When the icons are in sprite format and don't appear on our illustrations, they will still appear normally on your GEOS screens.

Chapter Two

GEOS For Beginners

GEOS for beginners

You may be wondering why we included this chapter at all, considering you already have a copy of the GEOS User's Guide. This section is for the computer tenderfoot looking into a strange land, viewing this geography from a different perspective than the veteran user. The GEOS User's Guide may not be appropriate for the first-timer. But the corresponding chapters of this book were written to make the learning process easier for those newcomers with a minimal knowledge of computers.

If you still have questions about words or phrases, the Glossary at the end of this book defines the most common terms.

2.1 Backup copies and work diskettes

When you buy GEOS V1.3, you get two diskettes. Each diskette has information on both sides; you access each side by inserting the diskette label up for one side and label down for the other side. This type of diskette is often called a *flippy* instead of a floppy diskette.

The GEOS diskette labeled DISK 1 has the system diskette software on Side A and QUANTUMLINK™ software on Side B. The system diskette software includes the main programs needed to start GEOS, the GEOS applications (geoWrite and geoPaint), and a number of *input drivers* and *printer drivers*. These drivers let you adapt GEOS to your input device (the device with which you control mouse pointer movement) and printer. QUANTUMLINK™ (or QLink for short) is an *online service*, or telecommunications service (more on this in Section 3.2).

The diskette labeled DISK 2 contains the backup system diskette software on Side A and the applications diskette information on Side B. The backup system diskette contains many of the same items as the system diskette. It includes a few accessories, input and printer drivers. The geoWrite and geoPaint applications, as well as font files and accessories, are on the applications diskette.

All the GEOS diskettes are heavily copy protected, and make it very difficult for the user to make a *backup copy*. This is why you get two diskettes when you buy GEOS V1.3; one for the main system and the other for everyday use. Once you are done configuring the backup system diskette for your system, put the system diskette in a safe place, far away from anything magnetic or liquid. You can copy the applications and drivers to work and backup diskettes, one file at a time (more on this in Section 2.4).

There is a generic copying program on the system diskette and applications diskette called `DISK COPY`, which will copy any non-copy protected diskettes. You cannot use `DISK COPY` for copying the two GEOS diskettes, but it is useful for creating copies of document diskettes later on.

1. Plug the joystick or mouse into the port closest to you on the right-hand side of your '64. Turn off the printer, if you have one connected. Turn on your computer and disk drive.

2. Insert the GEOS system diskette and type:

   ```
   LOAD"DISK COPY",8,1 <RETURN>
   ```

   The word `LOADING` appears on the screen. `DISK COPY` loads in about 30 seconds.

3. When the disk drive stops, type the word RUN and press the <RETURN> key.

4. The following appears on the screen:

   ```
   DISK COPY UTILITY
   PLACE DISK TO BE COPIED IN DRIVE
     AND PRESS RETURN TO CONTINUE
   ```

   Remove the system diskette from the disk drive and insert the source diskette (the diskette you want copied). Press the <RETURN> key. The screen displays:

   ```
   READING SOURCE DISK...
   ```

5. The disk drive runs, and the following appears on the screen:

```
DISK COPY UTILITY
PLEASE INSERT DESTINATION DISK
AND PRESS RETURN TO CONTINUE
```

Remove the source diskette from the drive. Insert a blank, unformatted diskette in the drive and press the <RETURN> key. The screen displays:

```
FORMATTING DESTINATION DISK...
```

After a few seconds the screen changes to:

```
WRITING DESTINATION DISK...
```

6. The program may ask you to switch between the source diskette and the destination diskette. When the copy is done, the screen displays:

```
DISK COPY UTILITY
PLEASE INSERT SOURCE DISK
AND PRESS RETURN TO CONTINUE
```

7. Insert the source diskette and press the <RETURN> key. The following message appears:

```
DISK COPY UTILITY
BACKUP COMPLETE!
MAKE ANOTHER COPY?
PRESS 'Y' for YES, 'N' FOR NO
```

Press the <Y> key to make another copy or the <N> key to exit the program. If you press <Y>, the copy procedure repeats. If you press <N>, the following appears on the screen:

```
DISK COPY UTILITY
PRESS 'B' TO RETURN TO BASIC, OR
INSERT GEOS BOOT DISK
AND PRESS RESTORE
```

Press the <B> key to return to Commodore BASIC. If you want to go directly to GEOS, insert the system diskette or the backup system diskette and press the <RESTORE> key. GEOS loads automatically.

Note: If you wish to make your own backup copies of the applications diskette, you might try one of the standard copy programs on the market. However, backing up the system and backup system diskettes is nearly impossible for most beginning users.

2.2 Starting GEOS

You're ready to start learning GEOS.

Plug the joystick or mouse into the port closest to you on the right-hand side of your '64. Turn off the printer, if you have one connected. Turn on your computer and disk drive. Place the GEOS system diskette (or backup system diskette) into the drive and type:

```
LOAD"GEOS",8,1<RETURN>
```

After five seconds, the words `BOOTING GEOS` appear on the screen. GEOS takes about 30 seconds to load. If GEOS has problems loading, one of the following may happen:

- After five seconds, `BOOTING GEOS` appears on the screen; suddenly the screen compresses and returns you to the 64 power-up screen. This means that GEOS thinks the diskette is not copy protected. You must start over and `LOAD "GEOS",8,1` to re-try the operation. If GEOS keeps refusing to load, see Section 5.1.1 for help. If all else fails, take the GEOS diskette to your Commodore dealer—either the GEOS diskette is defective or your disk drive needs alignment.

- The status lamp (red for 1541 drives, green for 1571 drives) lights and goes out, yet the 64 doesn't react. This means a load error has occurred—you'll have to start over. If this happens repeatedly, take the diskettes back to your Commodore dealer for replacement.

After GEOS loads, remove the system diskette or backup system diskette. Use the system diskette only for copying input drivers and printer drivers to your applications diskette. Use your backup system diskette for the initial booting process only. Use the applications diskette and work diskettes for other procedures.

Once GEOS finishes loading, your screen should look like this:

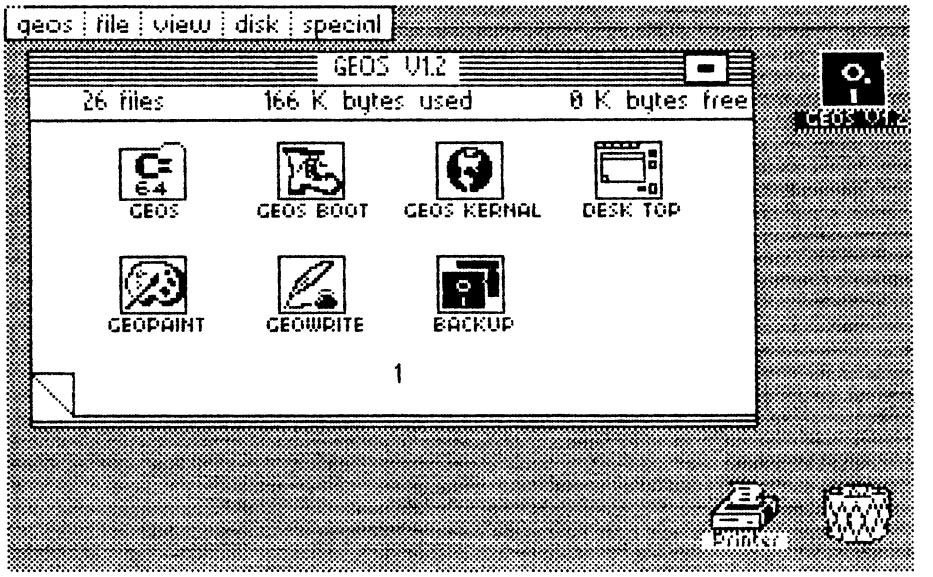


Figure 1: The deskTop

Before you begin working with GEOS, there are four things you must do:

1. Take your input device (you're probably using a mouse or joystick). Move the input device and watch the blue *pointer* (arrow) in the upper left corner of the screen. If the pointer doesn't move, make sure your input device is plugged into the first joystick port (the port closest to you on the right side of the computer). If it's plugged in, you need to change input drivers. Remove the applications diskette from the drive and insert the backup system diskette in the drive. Press and hold the <C=> (<Commodore>) key and press the <I> key. A list of input drivers appears on the screen. Using the cursor keys, move the pointer down to your choice and press the <RETURN> key. Move the cursor keys to the OK button and press the <RETURN> key again. This process lets you select an input driver temporarily (more on permanent set-up below). Try your input device to make sure it works.

2. GEOS must be informed when you have changed diskettes. Remove the backup system diskette and replace it with the applications diskette. Move the pointer to the word **disk** on the *command menu.* Pressing the fire button on this choice opens up a *sub-menu* of items which you choose by moving the pointer to that command and pressing the mouse button or fire button (we'll call this process *clicking* from here on).

   After you click **disk** the sub-menu appears on the screen:

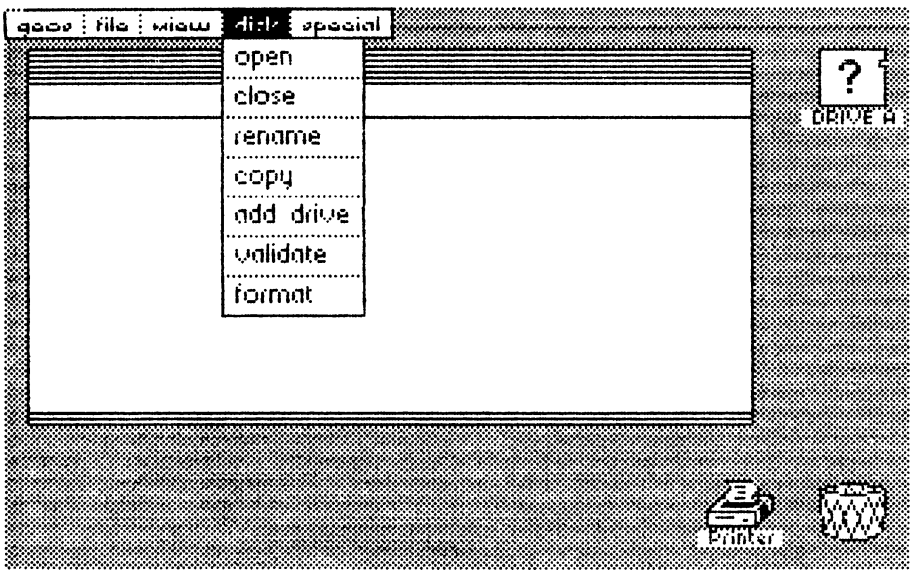


Figure 2: **disk** sub-menu

Place the pointer on the close item and click on it. The icons change—instead of the diskette icon in the upper right corner, an icon with a question mark appears. GEOS is waiting for you to open the diskette that you have inserted.

Click **disk** again, and click open from the sub-menu. The disk drive will run for a few seconds, and the icons of the disk directory will appear. This is your applications diskette directory.

3. For later reference: It is extremely important that the different diskettes you use with GEOS have different names from one another. This is pretty easy to do in GEOS:

   Click the **disk** menu, then select the item `rename`. A *dialogue box* appears on the screen with this prompt:

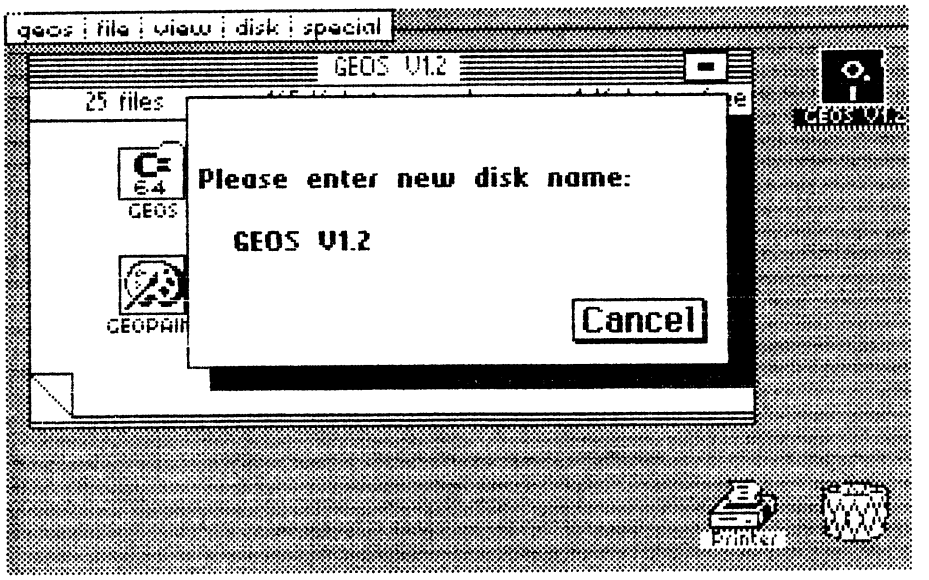


Figure 3: `rename` dialogue box

   Pressing the <DEL> key deletes the old name. You can enter almost any name you want for a new disk name and press the <RETURN> key to end the input. The *window* will disappear.

GEOS has *`info` boxes* for every program. The data in an info box includes the disk name, the filetype, date the file was last changed, and programmer's commentary. You can also find the write protect box data here.

Remove the applications diskette and insert the backup system diskette. Select the **disk** menu and the `open` option. Move the pointer to `GEOS KERNAL` and click it once. The icon changes color. Choose the **file** menu and click `info`. The disk drive runs, and the `info` box appears:

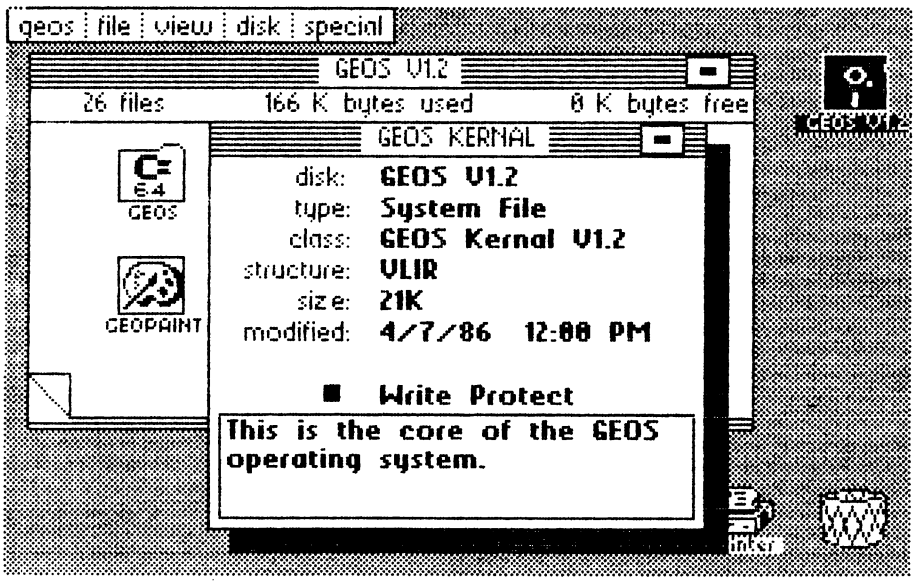


Figure 4: `info` box

This window contains information about the program. We are interested in one line in particular: `Write Protect`. The small filled box in this line signifies active write protection. To deactivate the write protect function, click the *write protect box*. It turns white, and `GEOS KERNAL` can now be deleted. Save this change to diskette and close the window (click the *close icon* on the upper right corner of the window). The window closes, the disk drive runs, and `GEOS KERNAL` is now deletable.

Note: Never delete a program from the system diskette or the backup system diskette.

`GEOS KERNAL` should now be a darker color than the other icons. If this is not the case, click `GEOS KERNAL` once. Pause for one second. Click `GEOS KERNAL` again.When you move the pointer, an image of `GEOS KERNAL` moves instead of the pointer (this is a *ghost icon*). Move the inverted image to the *waste basket* in the lower right corner and press the fire button. GEOS displays a dialogue box telling you that you can't delete this file:

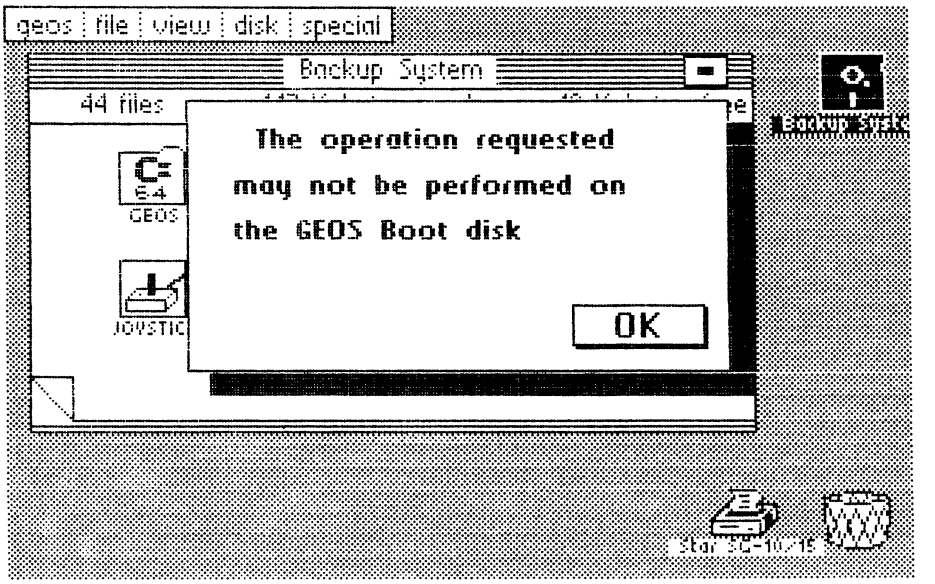


Figure 5: `Can't delete` dialogue box

Click `OK` (you'll get to do plenty of file deleting soon). Remove the backup system diskette and insert the applications diskette. Click on the diskette icon.

If you feel that 24 kilobytes of memory isn't enough for you, see Section 2.4 for instructions on making your own work diskettes for GEOS applications.

2.3 The perfect setup: The Preference Manager

GEOS has several *accessory* programs in addition to the main application programs geoWrite and geoPaint. One such accessory is the Preference Manager, with which you can change and store certain parameters to your own liking (pointer speed and shape, installation data, etc.). This accessory is loaded every time GEOS is loaded. Changing these parameters will not destroy any of your documents, so don't be afraid to experiment with them.

Let's load the Preference Manager. All the accessories appear as icons on the screen and as filenames under the **geos** menu. Go to the **geos** menu and click the desired accessory. For example, to open the Preference Manager, move the pointer to the upper left corner, click on **geos**, then move the pointer to the `preference mgr` item and click on that selection. The disk drive will run for a few seconds, then a new window appears:

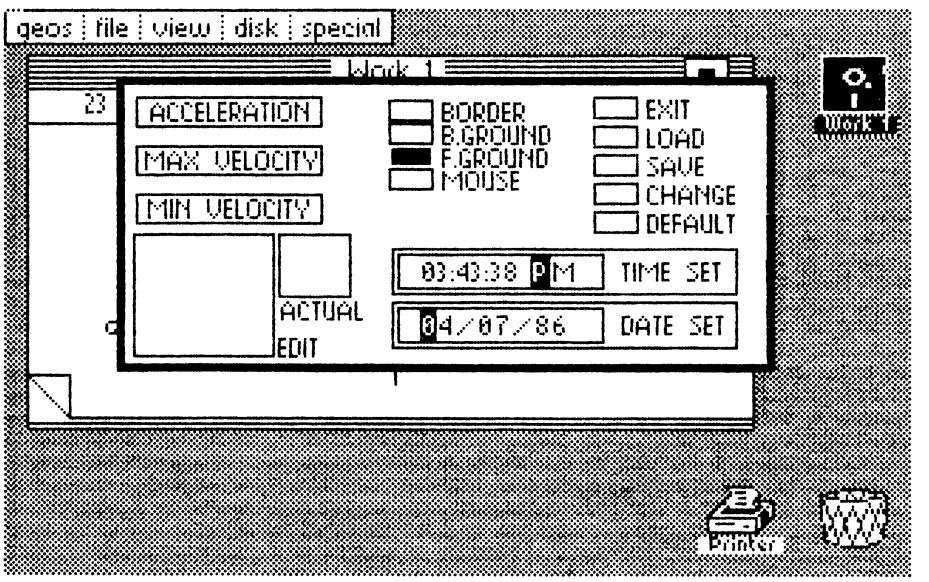


Figure 6: Preference Manager

The first thing you should do is change the time and date. You should do this every time you work with GEOS, since every document file is marked with the date and time. This helps you to immediately determine the last time you edited a document file, for example. Move the pointer to TIME SET. Move the text cursor to the P with the spacebar. The numbers are not erased by the spacebar—only the cursor is moved. GEOS will accept only valid terms. You can change the PM to an AM, and that's about all. GEOS will ignore any other keys at this particular point. Move the text cursor to the first number and change the time.

For example, press 0 9 4 5 3 0 A <RETURN>. The clock now reads 09:45:30 AM. Change the clock to the current time (hours from 0 to 12 only).

Now change the date by setting the pointer to DATE SET and entering the date in the format Month/Day/Year. The cursor also can be moved here by the spacebar. Remember to press <RETURN> when done editing the date.

Many other changes can be made with the Preference Manager. See Section 3.6.4 for complete details on the Preference Manager. For now, though, you'll only use this accessory to set the current time and date.

When working with these accessories, you should know that memory limitations may prevent programs such as the Preference Manager from loading. As long as your work diskette has enough memory, this won't happen. However, when there are less than 5K free on a diskette, GEOS displays an error message:

NOT ENOUGH DISK SPACE

This means that GEOS needs more disk space, otherwise the accessory will not be loaded. GEOS stores a section of itself on disk before loading an accessory. This is called a SWAP file. If there is not enough memory on the diskette, the error message appears. Make absolutely sure that there is enough room on the work diskette.

Now let's return to the deskTop. Move the pointer to the upper right corner (EXIT) and press the fire button. The main GEOS window will appear after a few seconds.

2.4 Work diskettes: Deleting and adding files

We mentioned in the last section how important it is to have enough room on your applications diskette. However, when you would like to use a diskette exclusively for letter files or artwork, for example, it's time-consuming to keep careful track of the amount of memory on diskette. GEOS allows you to copy individual files from diskette to diskette, and therefore let you make your own personalized application diskettes. To learn how this is done, we'll put together a diskette called `Letters` to contain your personal letters and notes.

First you'll need a blank diskette. Insert this blank diskette in the disk drive in place of the backup system diskette. Click the **`disk`** menu and choose the sub-menu item `format`. A dialogue box appears asking that you name the diskette. Type in the name `Letters` and press the <RETURN> key. The diskette will be formatted and given that name.

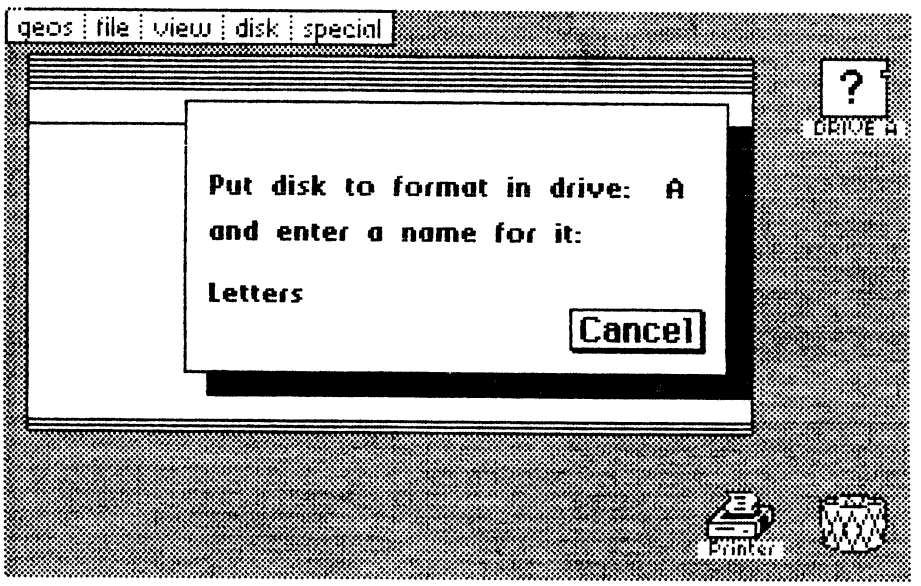


Figure 7: Format window

Remove the newly-formatted `Letters` diskette, insert the applications diskette and click `OK`. To copy files to your new diskette, proceed as follows:

Move the pointer to the `geoWrite` icon and click it. The icon changes into a *ghost icon*. Pause for one or two seconds and click it again. Now you can move the icon around. You'll remember this from Section 1.2, when you dragged icons to the waste basket to delete files. Now move the `geoWrite` icon below the edge of the window, and click it (press the fire button). geoWrite is now below the window, and the pointer is again visible. Your screen should look something like this:

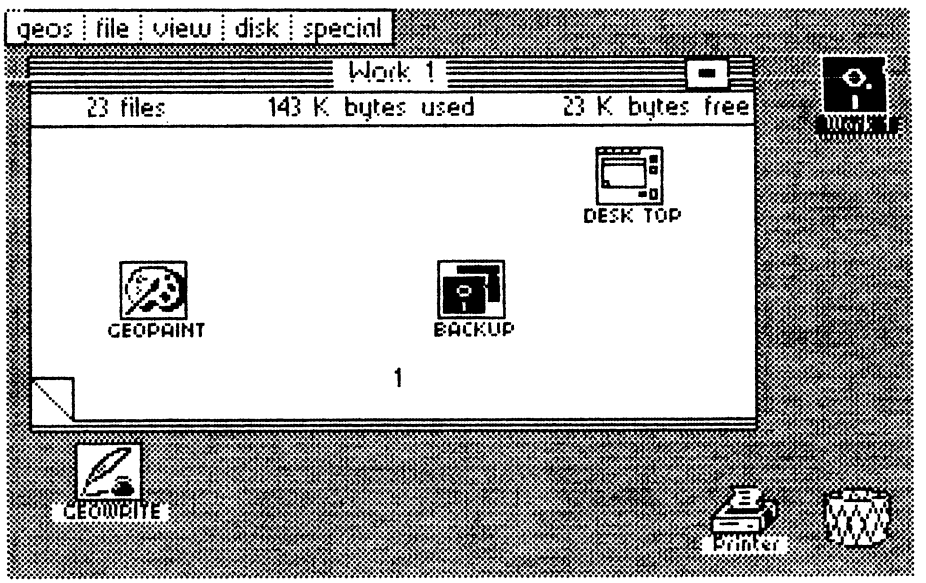


Figure 8: `geoWrite` south of the border

You are now ready to copy geoWrite to the `Letters` diskette. But we're going to want some other files which are important to the program. You have a number of programs in the deskTop window, and geoWrite below the window. However, the diskette contains many more programs which are invisible at the moment. You may already know that each page (section) of the disk directory can contain up to eight file entries. Since GEOS normally displays a sector in icon form, you can see eight programs as icons at a time.

To see the rest of the programs, you must turn to the next page of the directory. You'll notice that the lower left corner of the window has a *dogear*, similar to the folded corners on an old book. When you click the folded part of the corner, the directory flips to the next page. Clicking the unfolded area takes you to the previous directory page.

Before we do some page-flipping, look at the lower edge of the window. There is a 1 in the middle, which obviously refers to page 1. Now click the folded portion of the dogear. The new window is page 2, and there are a set of different program icons on this page. The Preference Manager is located here, which we talked about a few pages ago. There is another program on this directory page, named the Text Manager. This program will later allow us to "cut" sections of text, and then "paste" them into a text album. Click on the `Text Manager` icon and pause for a second or two. Move the icon to the area below the window and click again. The Text Manager should now be below the window and next to geoWrite.

Note: Some versions of GEOS may not allow you to copy some accessories such as the Text Manager. You may have some warning of this when you first copy the program over as instructed below—the icon of the copied program may be corrupted. The definite test is when you try to access that copied program. Either the screen changes to a normal C64 screen or the system crashes completely. If you can't copy an accessory you need, use the diskette that contains the program for your work and copy your text/picture files over to your work diskettes.

Next we'll want to copy some *fonts* for later use. We need to turn a page in the directory, but don't use the dogear this time. There is a simpler method—just press the desired number key. To get to page 3, simply press the <3> key on the upper row of the keyboard. GEOS turns the page, and displays the Font page. This page contains all the fonts available to geoWrite—you may well want to use all of them. Drag all five `Font` icons to the area below the window, just as you've done above. These fonts have the names `California`, `University`, `Cory`, `Roma` and `Dwinelle`.

You may wonder why we want you to copy all the font files; well, this is going to be a diskette for letters, and we want to be as flexible with our text effects as possible. Besides, the next section deals with some of GEOS capabilities, and we'll need those fonts.

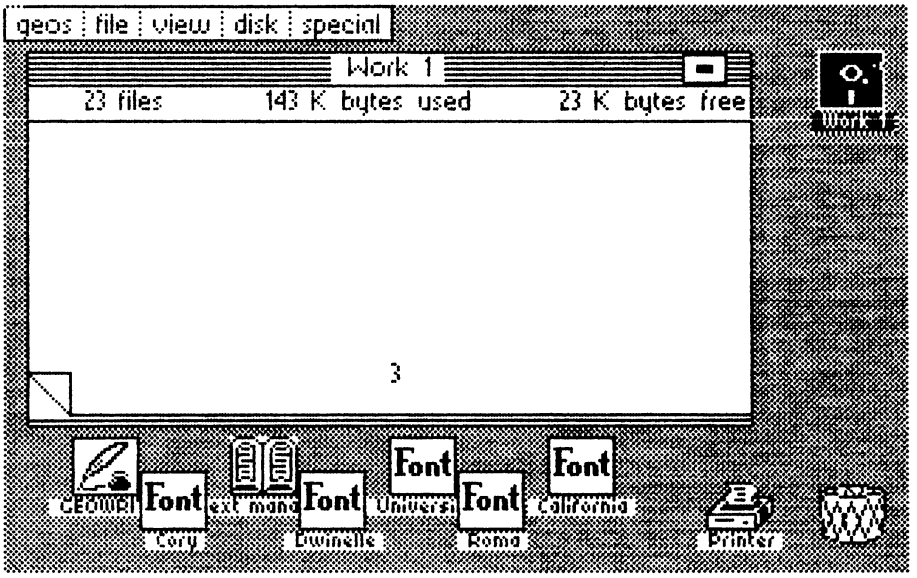


Figure 9: Ready to copy

Now we have everything we need for copying. Put the new diskette `Letters` into the disk drive. Click **`disk`**, then select the item `close`. Open the new diskette with **`disk`** and the `open` item. GEOS displays the directory of `Letters`, which is empty (contains no icons). You'll use the same process you used to move the icons from the work diskette, to move them onto the empty window of `Letters` (click/pause/click). Click anywhere on the empty window. Immediately a new dialogue box appears asking you to insert applications diskette, and click `OK` (we've renamed our applications diskette to `Work 1` here).

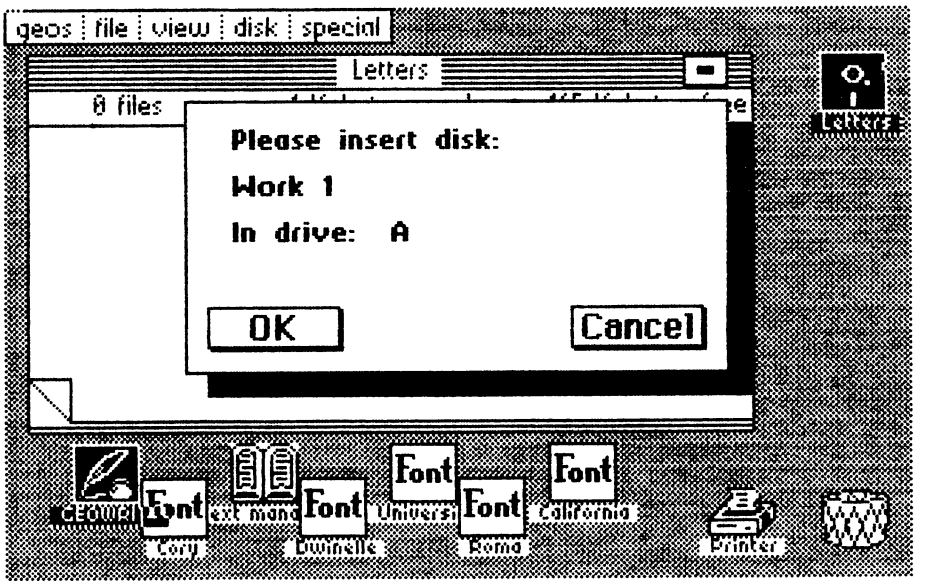


Figure 10: Copier dialogue box

Insert the applications diskette into the drive and click `OK`. After a short while a new dialogue box asks you to insert the `Letters` diskette.

Exchange the diskettes in the drive and click `OK`. Since geoWrite is a fairly large program, it can't be copied in one pass, and you will have to change between `Applications` and `Letters` more than once. When you click `OK`, GEOS copies the rest of the program, and soon a new directory window appears. geoWrite is now part of the `Letters` diskette.

Remove the `Letters` diskette and insert the applications diskette. Click the diskette icon. Click on the `Text Manager` icon. Pause a second and click again. Move the ghost icon onto the GEOS window and click again. GEOS asks you to insert the applications diskette. Change disks when you are asked to do so by GEOS, and soon the Text Manager will be on the `Letters` disk (see the note, page 21, if the copied Text Manager icon looks strange). Copy the rest of the files you want using this procedure.

Before we use a GEOS application for the first time, open the applications diskette again. Click any icon below the border, move the ghost icon back into the window, and click it into place. Do this with the rest of the icons. GEOS will understand that these files are to go onto this disk, and that you won't need to copy them again.

2.5 Applications: geoWrite

In addition to the accessories, GEOS V1.2 also includes two complete application programs. They are geoWrite and geoPaint. For the moment we'll be discussing geoWrite.

You now should have the `Letters` disk in your drive. Let's load geoWrite. Until now, we've had you load files by clicking the icon, then the **`file`** menu, then the `open` item from that menu.

But there is a second, faster method of loading programs. Move the pointer to the `geoWrite` icon and click twice in rapid succession. This action is called *double-clicking*. The program loads. Now you understand why we had you pause between clicks in the earlier chapters. So now we have three methods of clicking an icon:

1. Single click: The icon changes color. This signals GEOS that any commands sent by you now concern this program. To exit this process, just click another icon, or any free space between icons.

2. Double click with a pause between the two clicks: This allows you to move the icon. This movement is used for dragging an icon below the border or to the waste basket.

3. Double-click: Clicking the fire button or left mouse button (depending on the input device you have) twice in rapid succession selects and loads the program.

We've loaded geoWrite with a double-click. After loading, a dialogue box appears, offering you the following options:

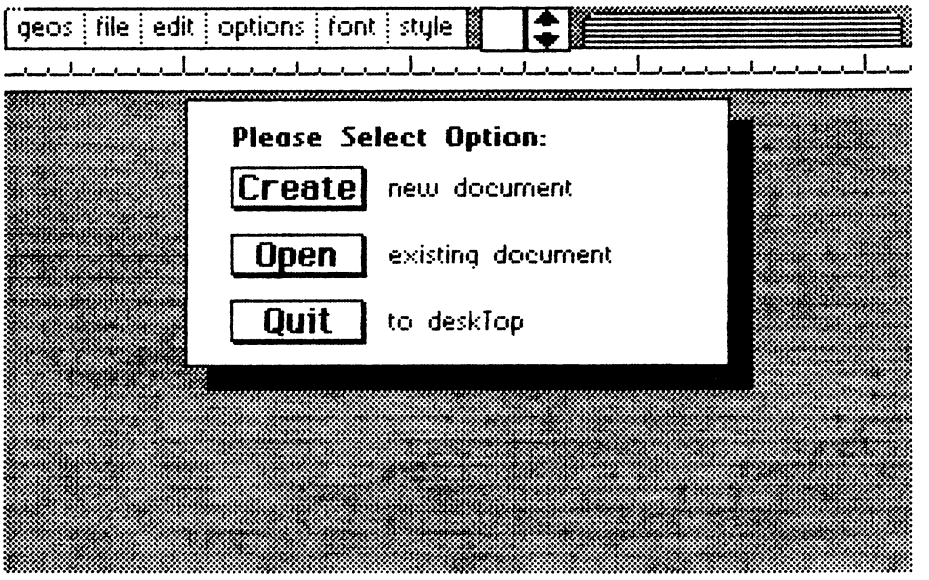


Figure 11: geoWrite dialogue box

Click `create`. This tells GEOS that you want to write a new letter. Another dialogue box appears, asking you to enter the name of your letter. Enter a name—`To Deb`, for example. Then press the <RETURN> key; part of the "typing paper" window will appear.

Now you can begin to write. Here's what our sample letter looks like:

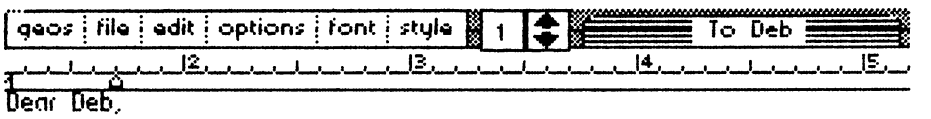


Figure 12: Letter to Deb

You'll note that a *text cursor* appears where the next character is to be typed. Press <RETURN> at the end of every line for now (i.e., the right border of the screen). You can type beyond the right edge of the screen; geoWrite only displays two thirds of the window at a time, and presently shows the left two-thirds. The window scrolls to the right when you type past the right border, but this tends to be confusing. We'll explain later how to work with the "hidden" right section of the paper.

After you've typed a few lines, you can experiment a bit with geoWrite's different fonts. Click **font** and then the font Roma. A second menu appears, showing four different *point sizes*:

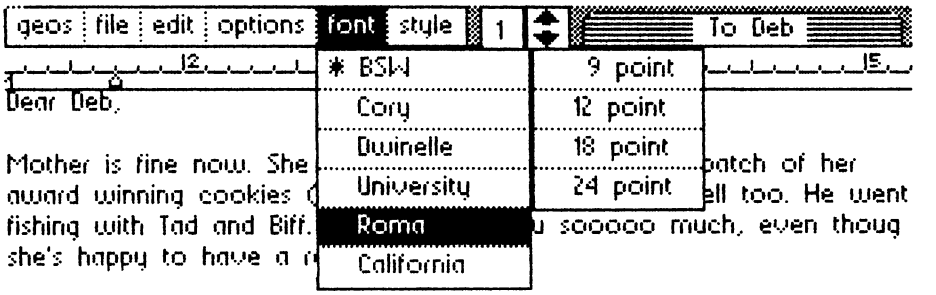


Figure 13: `Font` sub-menu

Try 18-point by clicking that item. The second menu, from which you selected 18-point, controls the size of the characters. Nothing seems to happen when you pick the font size, such as a window change. However, when you press a few keys, the disk drive runs and the characters appear in the selected size.

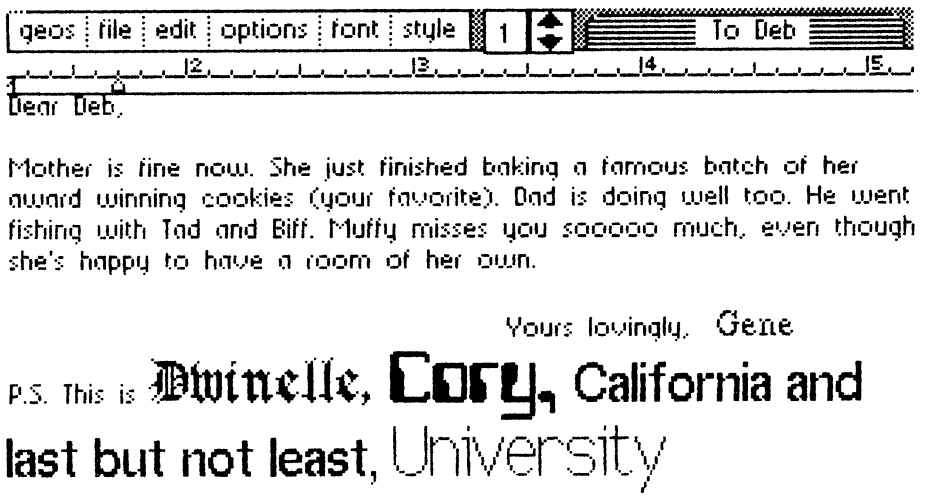


Figure 14: Point sizes in GEOS

We've written `Gene` in Roma 18-point, and displayed a few of the other fonts in the `P.S.`. This is just a hint of the different possibilities GEOS offers for text processing. You'll find that all of these fonts are on the `Letters` diskette, provided you copied them in the previous section. Feel free to experiment with the fonts and point sizes—you'll find you like some fonts better than others.

To exit geoWrite and return to the deskTop, select the **`file`** menu and the `quit` item. A dialogue box appears and says:

```
Please insert a disk containing the deskTop
          Version 1.3 or higher
```

We didn't copy the deskTop over to the `Letters` diskette. We did this to leave as much disk space as possible for text. If you prefer, you may copy this file to the `Letters` diskette later. For now, though, insert the applications diskette and click `OK`. deskTop will soon appear.

Our first encounter with GEOS is now complete. We hope you've enjoyed it. In the coming chapters we will systematically guide you through the rest of GEOS.

Chapter Three

GEOS In Detail

GEOS in detail

You have completed your first steps with GEOS, and have even seen a few of its features. This chapter contains systematic descriptions of the programs which make up GEOS. First we'll describe the window layout, then the menu and sub-menu items (where applicable). Finally, we'll discuss how you use the program.

3.1 deskTop: The window

An important part of the GEOS concept is the deskTop. For a long time, the primary mass-storage device for the C-64 was the DATASETTE. Thus the LOAD and SAVE commands defaulted to device number 1—the cassette drive—rather than the disk drive, device number 8. GEOS uses the disk drive exclusively.

The deskTop simplifies diskette access, saving you time and effort.

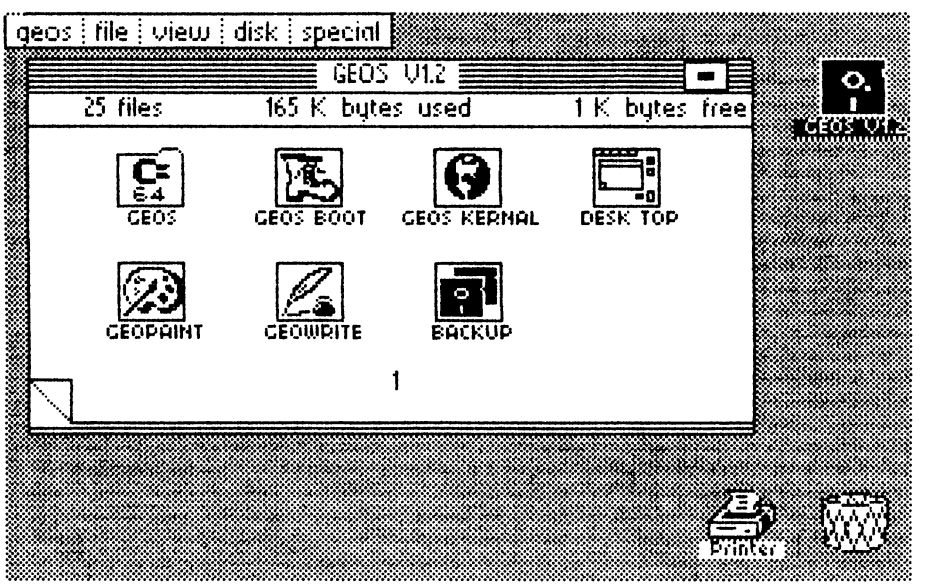


Figure 15: The deskTop window

The topmost line of the deskTop window is the *command menu.* Each of the terms in the command menu represents a *sub-menu,* which is called by clicking its respective word. The individual choices in this sub-menu are known as *items.* The first two lines in the deskTop window comprise the *title line* and the *info box.* The title line contains the diskette name. The info box contains the number of files on the diskette, the amount of space used, and the amount of unused space that remains.

The info box also has a small box containing a smaller black box in the upper right corner. This box is the *close icon.* Clicking this box closes the diskette, just as if you clicked close in the **disk** menu. You'll find this box frequently in GEOS windows. For example, the info box has a close icon, which lets you save information and exit the window. The opposite of the close icon is the *diskette icon,* at the upper right corner of the screen. It is labeled with the name of the diskette. When you click this icon, the diskette opens. The diskette is black when a diskette is opened, and becomes a ghost icon with a question mark when closed.

The rest of the screen beneath the menu are called GEOS *windows,* and display up to eight diskette file entries per page. The pages are flipped by clicking the dogears in their lower left corner. Clicking the "fold" advances to the next page, while clicking the corner pages backwards. Pages can also be turned by typing in the page number from the number keys on the keyboard. There are two icons at the lower right section of the screen: A waste basket and a printer. To print a text file, you don't have to access geoWrite first—just click the icon that represents your text file and the printer icon to print. Drag the icon over to the waste basket and click to delete. The icons indicate a filetype.

GEOS uses the following filetypes (if you won't be using your own programs with GEOS, you can skip this):

1. System file: Important system programs like `GEOS KERNAL` and system data like `photo scraps`.

2. Program file: Applications (geoWrite), accessories (Preference Manager) and self-starting machine language or BASIC programs.

3. Data files: Documents (text/pictures), font files, non-GEOS data.

4. Interface files: Programs/data for various printers or input devices.

5. Non-GEOS files: Not created by GEOS, or altered programs and data in GEOS format. They're identified by the C= symbol.

3.2 deskTop: menu and items

Now we'll explain the functions of the menus and sub-menu items. The menu is listed first, followed by its respective sub-menu items:



geos:

This menu contains commands to display program information, specify input and output devices, and select accessories:

`geos info:`
Displays a window with the names of the authors of GEOS.

`deskTop info:`
Displays a window naming the programmer who designed the deskTop.

`select printer:`
Lists the printer drivers available on the diskette. Choose a printer driver by clicking the desired printer. The name changes color. The printer driver is installed when you click `OK`.

`select input:`
Similar to `select printer`, except this allows you to select input devices. A number of input drivers are available for joystick, Koalapad©, lightpen and mouse.

`preference manager:`
Displays the user-specified parameters (screen colors, pointer speed, time, date, etc.).

`notepad:`
A 127-page notepad onto which you can enter commentary or notes. These notes are saved and reloaded with the GEOS system.

`photo manager:`
An online "photo album". Here you can store documents on album pages, and "cut" and "paste" them to use with other documents or text.

`text manager`:
Similar to `photo manager`, except it works with text. You can create a text album of frequently-used texts and "paste" these into other texts.

`calculator`:
An online pocket calculator with the commonly-used math functions.

`alarm clock`:
An online digital clock with an adjustable audio alarm.

`file`:

This menu contains all the commands applicable to individual files (programs, letters, pictures, etc.).

`open`:
Loads a program. Trying to load a non-program file (e.g., printer drivers) will cause an error. Exceptions to this rule are document files and albums, which load after automatically loading the application program (for instance, opening a text document file first loads geoWrite).

`duplicate`:
Copies the selected program to the same diskette. You are asked to enter a name for the duplicate file, followed by a <RETURN>.

`rename`:
Changes the name of a file. A dialogue box appears containing the old filename. Enter a new filename followed by <RETURN>.

`info`:
All GEOS format files have an `info` box. When this item is selected, a dialogue box is opened. Clicking the write protect box activates (black) or deactivates (white) the write protection. You can enter your own notes or description in the area below the write protect box, and this text is saved as part of that file's information.

print:
Documents created with geoWrite or geoPaint can be output to a graphic printer (usually a dot-matrix printer). Be sure that you have previously selected the correct printer driver using `select printer`.

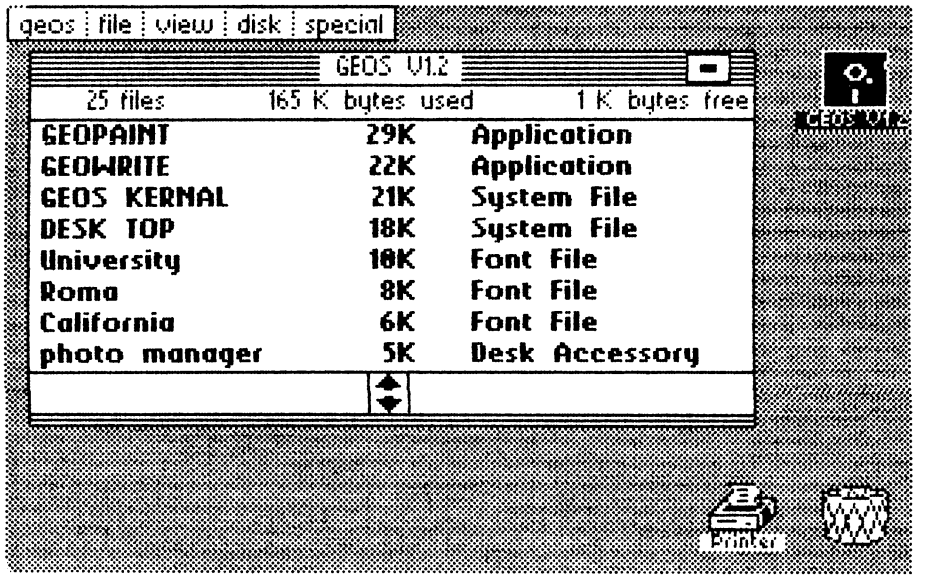


Figure 16: Files arranged by `SIZE`

view:

deskTop normally displays files as icons. This menu allows you to change the format of the display. The files can be displayed by name rather than icon. However, you can move and select files only if they are displayed as icons. After displaying these files in one of these formats, return to icon mode to continue your work.

by `ICON`:
Displays the files as icons from left to right and top to bottom, in the order they are stored on the diskette.

by `SIZE`:
Displays file entries according to file size, starting with the largest to the smallest.

by TYPE:
Displays the files according to GEOS filetype. Filetypes can be found in the info box.

by DATE:
Displays the files according to date. Every GEOS file has a date. The files are displayed starting with the most recently-opened file.

by NAME:
Arranges the files in alphabetical order.

Since GEOS can only display only eight file entries at a time, you can use the *scroll box* to see the other entries. A scroll box has two arrows. Clicking these arrows scrolls the list up or down. Holding down the fire button scrolls the list automatically.

disk:

This menu item contains all commands involving access to an entire diskette.

open:
GEOS "opens" the diskette found in the disk drive. The pertinent diskette data appears in the window (e.g., diskette name, free memory, etc.).

At the same time GEOS tests the diskette for GEOS format. If the diskette is not in GEOS format, the system asks if the diskette should be converted.

This has nothing to do with the FORMAT command, which completely erases and reformats the diskette. In conversion, GEOS writes another sector to this diskette which notes the files outside the window border. This border is used when copying individual files, as you've already seen. Thus, GEOS can have icons outside the lower border only when the diskette is in GEOS format.

close:
Click this command before exchanging diskettes. Once a diskette has been exchanged, then click open.

rename:
Renames the diskette. The current diskette name is displayed. You can then erase this name with the <DEL> key and enter a new name followed by <RETURN>, or cancel the command by clicking Cancel. You can avoid a lot of confusion by giving your diskettes different names.

copy:
Copies the contents of the current diskette to another diskette. You'll be asked to insert the destination diskette. The destination diskette should be either a blank diskette, or a diskette whose contents are no longer needed.

After you've inserted the destination diskette and clicked OK, GEOS reads the name of the diskette. If the diskette is unformatted, GEOS asks if the new diskette should be formatted.

Once GEOS has the name of the destination diskette, it asks you to insert the source diskette (the one from which you wish to copy), and after you click OK, asks you to insert the destination diskette. GEOS will refer to the two diskettes by their names, not simply SOURCE and DESTINATION.

The copy command is meant for the user with two disk drives. If you only have one drive, then this command is not worthwhile to you, since you would have to exchange disks 30 times for a full source diskette. You're best off using the DISK COPY program—it requires only three passes.

add drive:
This command allows the two-drive GEOS user to activate a second disk drive. The individual steps needed are described in a window. Turn off the previously-used drive, and turn on the second drive. Follow the directions in the window. The second disk drive is assigned the device number 9, and is referred to as DRIVE B. Now turn on the original drive, which is DRIVE A with device number of 8.

validate:
This command "cleans up" the diskette. Sectors of the diskette that do not contain valid data are freed for subsequent use.

Warning: A GEOS file takes up more sectors than a non-GEOS file. These added sectors are not recognized outside of GEOS by the VALIDATE command. **Never use the normal C-64 VALIDATE command with a GEOS diskette. If you do this, immediately load GEOS and execute a GEOS validate.**

format:
GEOS asks you to enter a name for the diskette followed by <RETURN>. The diskette is automatically formatted in GEOS format.

special:

This menu contains three commands that bypass GEOS.

BASIC:
This command exits GEOS and returns you to the BASIC interpreter. To reboot, insert the original GEOS diskette and press the <RESTORE> key.

reset:
This item re-initializes GEOS. The important data values are reset to their default values, and the current diskette is opened. You might try to use this item when you've forgotten to perform a diskette change, and you're stuck in an error loop.

Q-link:
This command connects you to QUANTUMLINK™ (QLink), an online communications service. It allows C-64 users with GEOS and a modem to talk to the world through telecommunications. You'll find this QLink software on the opposite side of your GEOS system diskette.

3.3 deskTop: Functions

In addition to the commands in the GEOS command menu, there are also a number of functions to make your work simpler for you. Some of these functions, such as copying, deleting and clicking files, and different click items, are already familiar to you. Let's take a closer look at these functions.

3.3.1 deskTop functions performed with icons

You can close the current diskette by clicking on the close icon in the title line. This icon looks like a tiny cassette. The icons disappear, and the diskette icon on the right side of the screen becomes a ghost icon, with a question mark at its center:

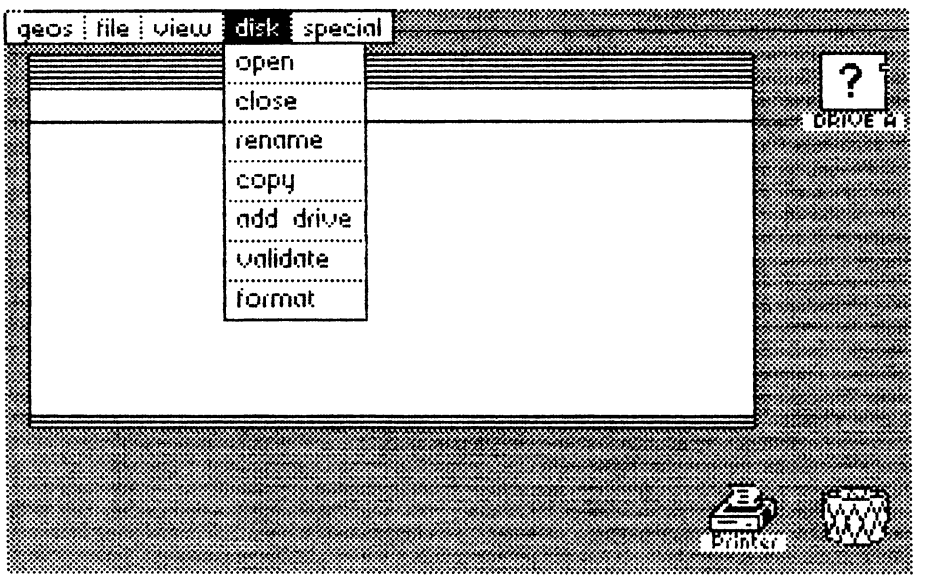


Figure 16.5: Diskette ghost icon

When you click the question mark inside the diskette icon in the right corner of the screen, the diskette is opened. No harm is done if you forgot to `close` the diskette and click `open` again to open another diskette. It's enough to just click the icon directly.

In the lower corner of the screen is the printer icon. If you select a document file icon and then drag its ghost icon to the the printer icon, a printout of the file is produced.

To the right of the printer icon is the waste basket. To delete a file, you simply drag the desired file's icon to the waste basket. If the error message `WRITE PROTECT ON` appears, you must go into the `info` box (click the `info` from **`file`**), click the square to the left of the write protect box, and then retry the delete operation.

The dogear in the lower left corner of the GEOS window is used to view multiple pages of the directory. Clicking it advances the window to the next page. Clicking the "unfolded" corner displays the previous directory page.

3.3.2 Keyboard

Press keys 1 through 9 on the top row of the main keyboard to select the directory pages. To go to page 3 of the disk directory, press the <3> key. GEOS turns to that directory page.

3.3.3 Clicking

You can click GEOS icons with the joystick fire button in the following ways:

Simple click:
The icon changes color. The icon is then ready for different operations such as looking at the `info` box or printing the file. You can exit a simple click by clicking any other icon, or by clicking the blank area outside of the icon.

Two clicks:
Click, then pause briefly (at least a half-second), then click again. This clicking selects the icon, and allows you to move it anywhere around the screen, such as outside of the border. You can get out of this mode by clicking again.

Clicking an icon:
Depending on the situation, a click can have different results (print, delete, move below the border). Clicking again returns the icon to its original state.

Double-click:
Double-clicking means very quickly pressing the fire button twice without a pause. The double-click opens a file, and automatically opens the corresponding application program.

3.3.4 The remaining functions

Copying:
When you click a file twice and drag it below the window border, you can then copy it. Insert the destination diskette and `open` that diskette. Now you can move the icon into the new GEOS window and click it there. GEOS prompts you to change the diskettes as needed until the file is copied.

The border is not only used for copying programs—it can also be used for changing the order of programs on the diskette. You can have up to eight files below the border. If you want to change the order of the icons, move the icons below the border and then drag them back into the GEOS window in the desired order. The order in which the icons appear represent their position on the diskette. If, for example, you want to switch geoPaint and `DISK COPY`, click each file, drag each file below the border, then drag each icon to the desired position (geoPaint where `DISK COPY` was, and vice versa). GEOS fills in free positions from left to right and from top to bottom.

You can move an icon to another page within the GEOS window. Place the file at the space below the border, press a number, and move the icon back up to the page.

3.4 geoPaint

3.4.1 Starting geoPaint

One of the two applications included with GEOS is geoPaint. The geoPaint application has many extras in addition to the normal functions. You can create color documents from the sixteen colors used for characters and background. Foreground and background colors work in 8 x 8-pixel matrices. You can also add text in different *fonts* to the document.

geoPaint documents are set up in a standard (8.5" x 11") page format. Therefore, you see only a part of the whole picture. But you also have the option of viewing the entire document on the screen in a reduced resolution.

You can access geoPaint from the deskTop in one of three ways:

1. Click the `geoPaint` icon and `open` the file from the **`file`** menu.

2. Double-click `geoPaint`.

3. When you have a document already made, click that document's icon. This automatically loads geoPaint, and then the document.

If you click geoPaint and not a pre-existing document, a dialogue box appears offering three choices:

`create:`
Click this item to create a new document. geoPaint asks you to name the document. Enter the name and press <RETURN>.

`open:`
Click this item to display the names of existing geoPaint documents. Since only five are shown at a time, the scroll box can be used to scroll through the listing. **Note**: geoPaint only shows the first 16 documents on the diskette. If you have more than 16 files, you may need to change the order of files (see Section 3.3.4).

`quit:`
Click this item to exit geoPaint and return to the deskTop.

If you select `create` and enter a filename, the window appears as follows:

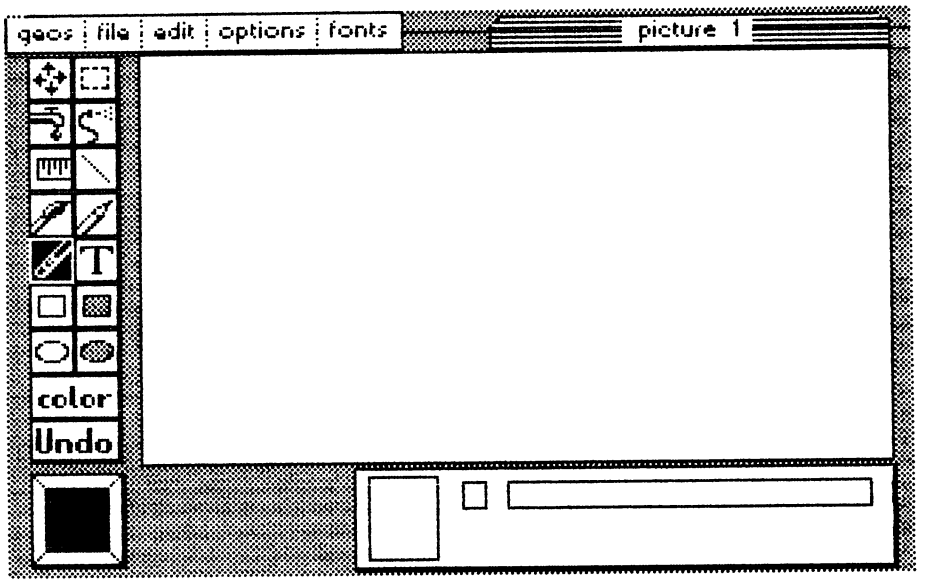


Figure 17: geoPaint

The topmost line to the left is the command menu, and the name of the document is to the right. The geoPaint window takes up most of the screen to the right of the toolkit. This is where part of the document is displayed.

The *toolkit* is on the left side of the screen. It contains the different tools available for creating and modifying geoPaint documents. Click a tool once to activate it. Some tools require a second click to select another function.

The current pattern is indicated in the lower left. The Airbrush tool uses the pattern.

The *status box* is to the right of the pattern. In most cases, it contains information about the tool being used. When you begin working with geoPaint, the pencil is the default tool. You can tell which tool is "active" because its image is darker than the others.

The status box will display information about the active tool. To the left is the *page map*, in which there is yet another small rectangle in the upper left corner of the page display.

Note: As we mentioned at the beginning of this book, sprites do not appear in hardcopy. Therefore the small rectangle does not appear in Figure 17.

This rectangle corresponds to the section of the document visible on the screen. Thus the upper left corner of the document is now displayed. Next to this display is the color indicator (small square), which shows the current color being used. The color listing to the right of the color indicator shows the sixteen available colors, while the arrow above this line shows the current color selected.

You select a tool by pointing to it and clicking. Use the tool in the window page by moving the pointer to the desired tool icon, clicking to activate, using the tool, and clicking to deactivate. The pointer will change color when on or off.

3.4.2 The geoPaint menu

As already seen in deskTop, each *menu* has a *sub-menu* from which you may choose *items*. This sub-menu drops down when you click a menu. What sorts of functions do we have here, and what can they do for you?

geos:

Here you'll find essentially the same items as found in the same area under deskTop. `geoPaint info` replaces `deskTop info`, and displays the names of the authors of geoPaint. After that the accessories for the current diskette follow. Clicking the sub-menu selection is enough to start the accessory.

The sub-items `select printer` and `select input` are not available here. Therefore, remember to select a printer driver before printing a geoPaint picture.

file:

This menu contains the items that handle geoPaint documents:

close:
This command closes your document file. This is used when you're through with your picture. This does not close your workspace (i.e. geoPaint). geoPaint asks if you wish to create a new picture, open another picture, or quit to the deskTop.

update:
Normally, geoPaint does not save the changes you have made to a document. Sometimes you may want to save a picture to diskette. You can use this item to save the most current document and continue work. If you make a mistake later, you can click recover to return to the document picture that was last updated.

preview:
geoPaint documents are set up in a total resolution of 800 x 640 pixels. This gives you more room to work, and a great degree of detail. This command allows you to stand back to get an overall look at the document; the picture shrinks to let you see the entire document on the screen at a reduced resolution. Click OK to exit this mode.

recover:
This command works in conjunction with update. recover reloads the last-saved picture. **Note:** The picture currently on the screen is lost when the previous picture is recovered.

rename:
Lets you change the name of a document. The current name is displayed—delete it with the <DEL> key, enter a new name, and press <RETURN>.

print:
This command works the same as the deskTop version. The entire picture is output to a printer. Remember to choose a printer driver from deskTop before using geoPaint to print.

quit:
This command closes the file, exits geoPaint, and returns you to the deskTop. The last picture is automatically saved.

edit:
This menu contains commands for editing documents. geoPaint lets you to "paste" portions of a picture into an album. You can then copy the album to a new picture, or merge the picture with text.

cut:
This command is similar to using a pair of scissors. cut removes a section of the picture, and once cut, the section disappears. The section is delimited before before cutting by clicking the *edit box* in the upper right corner of the toolkit. Now move the pointer, which appears as crosshairs, to the upper left corner of the desired section. When you click, geoPaint marks this point and changes the color of the crosshair.

Moving the pointer to the lower right creates a dashed rectangle that delimits the box for cutting. Click this corner to fix the size of the box. Now choose **file**, and click cut. The disk drive spins, and geoPaint saves this section to diskette under the name photo scrap. You can use photo manager later to paste this scrap into an album.

copy:
This command is different from cut in that the delimited area is not removed, but simply copied into photo scrap.

paste:
This command lets you paste the current photo scrap to any area of the document. The destination is specified by the edit box. If the area is too small, then only a portion of the document is pasted.

paste lets you copy part of the geoPaint window. Cut or copy the desired area to photo scrap (using the edit box with cut or copy), move the geoPaint window to the desired destination area of the page (with the scroll box), mark the destination and then click paste.

Remember the following when you work with these three commands:

photo scrap can handle only one geoPaint section at a time. When there is already a section in that area and you cut or copy a new section, the old photo scrap is replaced by the new one. Thus, you can copy only one segment at a time to photo album. When you want to paste several window ranges into your album, put a section into photo scrap, call up photo manager and paste it in. After you leave photo manager, you can use photo scrap again.

options:

This menu contains various items that allow you to further edit a document.

pixel edit:
This command allows you to create detailed documents under "magnification." Clicking this item displays a zoom marker with four corners—the zoom area. Set the zoom marker over the area you wish to enlarge. Click once, and the enlarged area appears in a window. Each point is now represented by a small square. Activate the pencil and click an area in the window—a point will turn "on". This is the equivalent of drawing a picture on paper with only dots. Click the pencil again to turn the point "off".

You can use the eraser to turn off (reset) the individual points. Click a point that you want turned off. The eraser removes that point. Click again to turn that point on. You have both eraser and pencil available simultaneously.

With a little practice, you can create extremely detailed drawings with pixel edit. GEOS shows the section being edited in its original size in the lower left corner of the window, so that you can view your work from a distance. The scroll bar performs another function in pixel edit. When you want to move the zoom window, click the scroll box in the upper left corner of the toolkit. The zoom window moves in the specified direction. The following tools are not available in this mode: text, edit box, circle and filled circle.

normal edit:
This command switches back to the normal window.

change brush:
geoPaint offers a total of 32 different brushes with which you can produce some remarkable effects. Clicking change brush allows you to select one of these brushes. Click the desired brush. It is outlined by a small square.

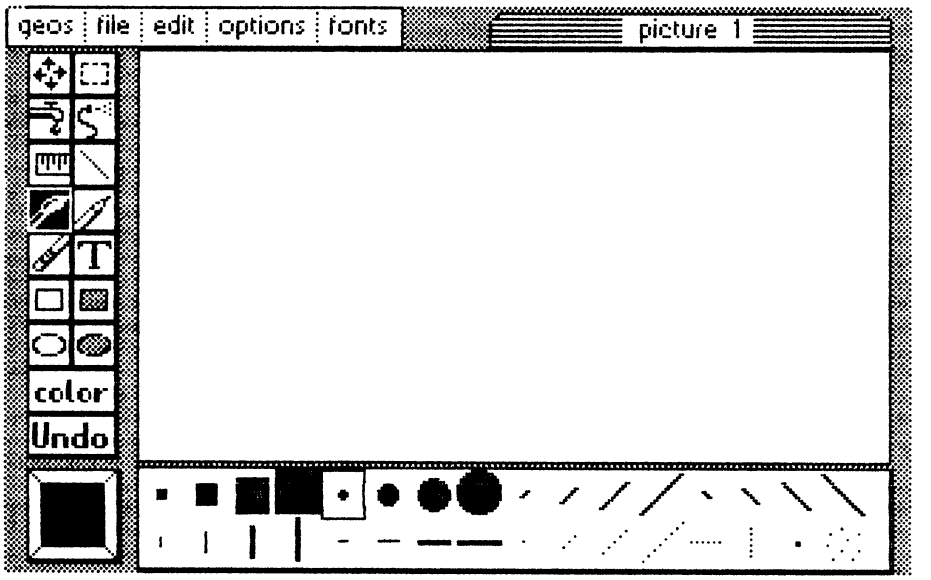


Figure 18: `change brush`—32 different brushes

Click the brush icon in the toolkit to use the brush.

`color off:`
When you have a color picture but no color printer, you can obtain a hardcopy printout of your drawing by clicking `color off`. geoPaint then displays your document in black and white. If you look at the **`options`** menu now, the fourth item will read `color on` instead of `color off`. Clicking this returns you to your color document.

`font:`
One of geoPaint's great strengths is that it allows you to use different fonts in your documents. The different fonts are chosen under **`options`**. Clicking an item displays another window that lists different fonts in several *point sizes*. Click the desired font and size.

3.4.3 The toolkit

The toolkit offers you different desigh tools and functions. We'll explain the toolkit icons and functions in detail now, beginning at the top left corner, moving left to right and top to bottom.

Scroll box:

The four arrows represent the different directions in which you can move the section marker. Click the scroll box, and after a short time the marker appears in the middle of the geoPaint window. Moving the joystick moves the section in the appropriate direction, as long as the border of the picture has not been reached. As you move, the section that becomes "hidden" is saved to diskette, and the new section to which we are scrolling is loaded from diskette. When the desired section is displayed, click it. The scroll box has another purpose in `pixel edit`: clicking it lets you move the zoom window around the screen.

Edit box:

Marked sections are needed for more than `cut`, `copy` and `paste`. geoPaint can perform many functions using an enclosed area. When you click the edit box, the status box shows the available commands:

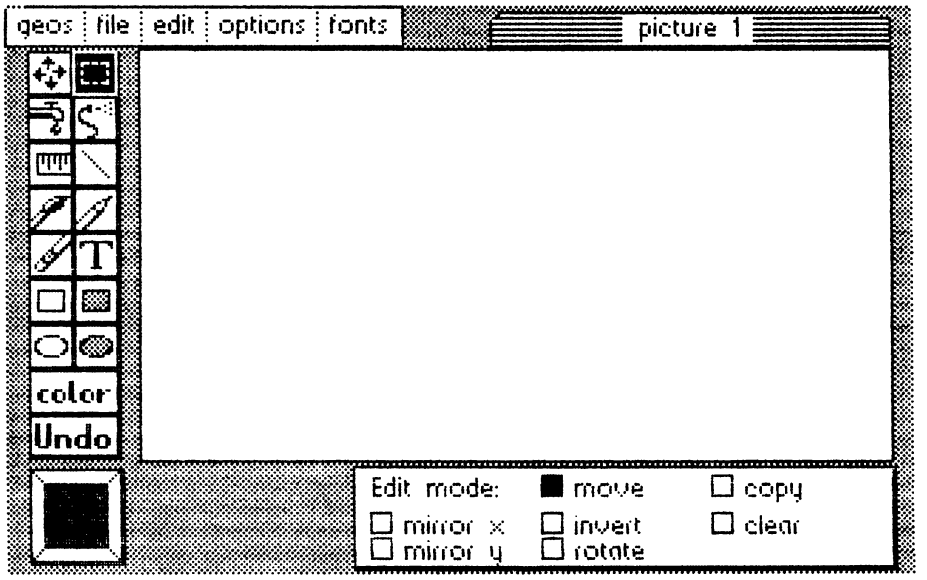


Figure 19: Edit box

Enclose an area by moving the crosshairs to the upper left corner of the desired area and clicking. Next, move the crosshairs down to the lower right corner of the desired area and click again. The enclosed area is set off by a dotted rectangle. Now you can use the **edit** mode to cut, copy, or paste this enclosed area in photo scrap. Or, select one of the commands in the status line:

mirror X	mirrors the image in the Y axis (i.e. left to right).
mirror Y	mirrors the image in the X axis (top to bottom).
invert	inverts the display of the enclosed area—set pixels are turned off, and unset pixels are turned on. This gives you a "reverse video" effect.
rotate	rotates the enclosed area 90 degrees clockwise. rotate an area four times, and you return to your original orientation. A segment just outside the enclosed area can end up disappearing during rotation. The solution is to either make the marked area as square as possible, or mark an area large enough to encompass all sections of the document.
clear	deletes the enclosed area.

The move and copy commands are similar. The main distinction is that move physically moves the enclosed area to a different location, while copy leaves the original enclosed area and duplicates it at a different location. Be sure that the area is enclosed first. Click the appropriate boxes in the status box to activate these commands. To execute the command, click inside the enclosed area with the pointer. When you move the pointer, note that the enclosed area "follows." When you have reached the "new" area, click again, and the enclosed area is moved or copied as ordered.

You can use copy to create duplicates of the enclosed area. Click the center of the enclosed area; the crosshair will change color. Move the crosshair to the new location, and click again. Presto! A copy of the copy. Do this as often as is required.

Pattern Fill:
The tool that looks like a water faucet lets you fill areas with a specified pattern. This pattern is displayed in a box beneath the toolkit. The area to be filled must be completely enclosed—otherwise the entire window will be filled in. Click the faucet icon, move to the area to be filled, and click. The area will fill in with the specified pattern.

Airbrush:
The airbrush works like a can of spray paint. It sprays in the pattern indicated in the pattern box.

Ruler:
The ruler lets you measure your document and determine dimensions and positions. When you click its icon, the status line looks like this:

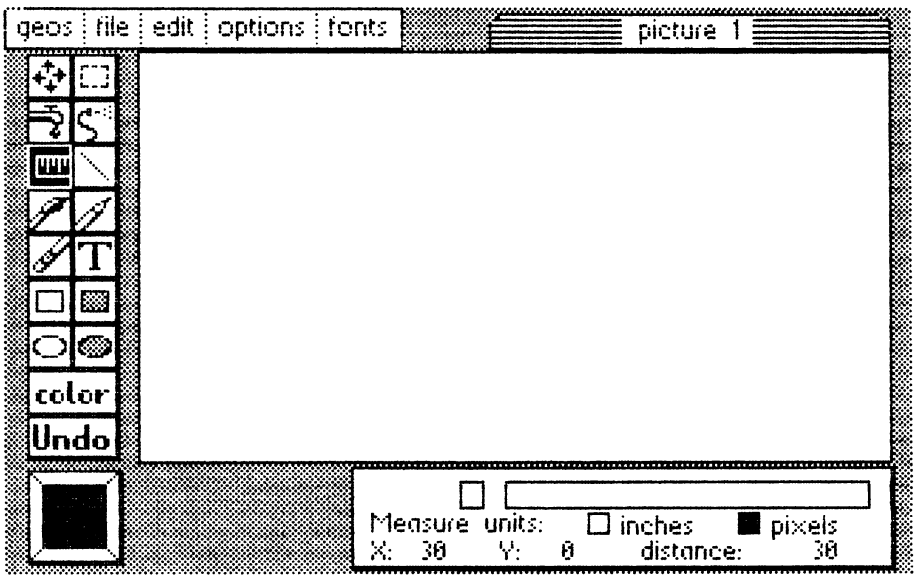


Figure 20: Ruler and lines

You can choose your measurement in pixels or inches. One pixel (picture element) is the smallest point useable in a picture. Click the appropriate box. `X`, `Y` and `distance` are measured from the starting point. Move the crosshairs to the point that you want to start from. When you click here, the message reads:

```
X : 0  Y :  0  distance : 0
```

Measurements are taken from this point. Move the crosshairs and the display reads accordingly. The X measures movement to the right, and Y measures up or down. Distance is the distance between the starting point, or original coordinates, and the current position of the crosshairs.

Lines:

This tool lets you draw straight lines. It functions basically the same as the ruler above. Click a starting point, move the crosshairs to the desired endpoint, and geoPaint draws a line between the two points. The endpoint can then be changed to any degree, which erases the previous line and draws a new one. Once you click again, the line stays as is. Here, too, you can work accurately with coordinates and distance in the status line.

Brush:

The brush icon allows you to specify one of 32 different brushes. Choose another brush by selecting `change brush` in the **`options`** menu. In addition to the different sizes and forms, there are a few brushes used for special effects. Experiment with them. Try out the brush in the bottom row, fifth from the right—you'll get a multiple line:

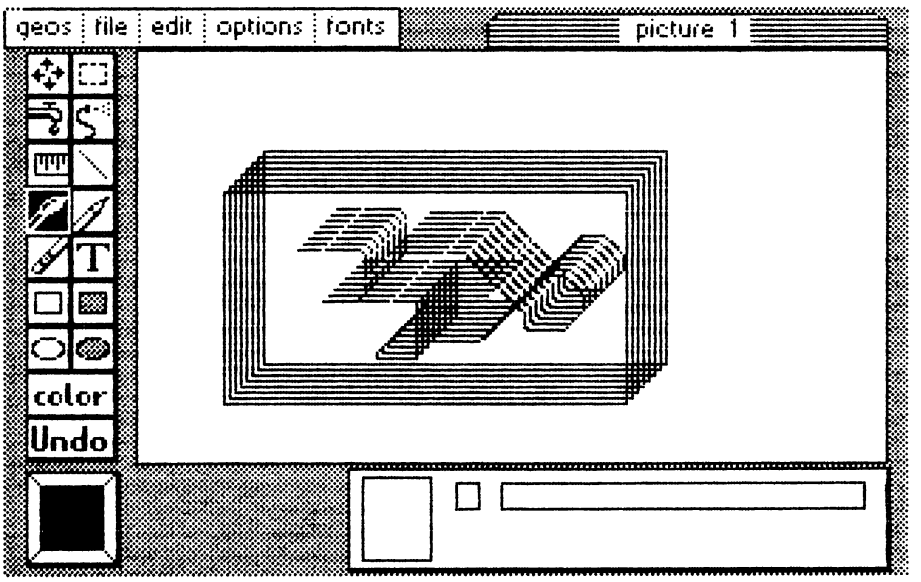


Figure 21: Repeated effect with brush

Pencil:

This tool is the default for geoPaint. You can draw freehand with this tool. Move the pointer to the window and click it. The pencil draw a single dot. Moving the pencil after you click it draws a line. Click again to turn off the pencil.

Double-clicking the pencil switches to `pixel edit` mode. As with all the tools, you can change the color of the pencil.

Eraser:

The eraser removes lines much like a rubber eraser, but you don't have to rub it back and forth as you do with a real eraser. You can erase the entire geoPaint window by double-clicking the eraser icon. We've found that this doesn't occur instantly. Sometimes you have to double-click the eraser several times to wipe out the desired area. It seems that when you double-click geoPaint quickly, the second click doesn't register. Try to double-click a little slower than usual. If you have problems and want an alternative, mark the entire geoPaint window off with the edit box, and click `clear` in the status line. Double-clicking in `pixel edit` erases only the range visible under the zoom window.

Text:

One of geoPaint's great strengths is the number of fonts and type styles you can use within your documents. For example, it's extremely simple to do technical drawings thanks to feature. When you select the font and font size, and you click **T** for Text, the status line looks like this:

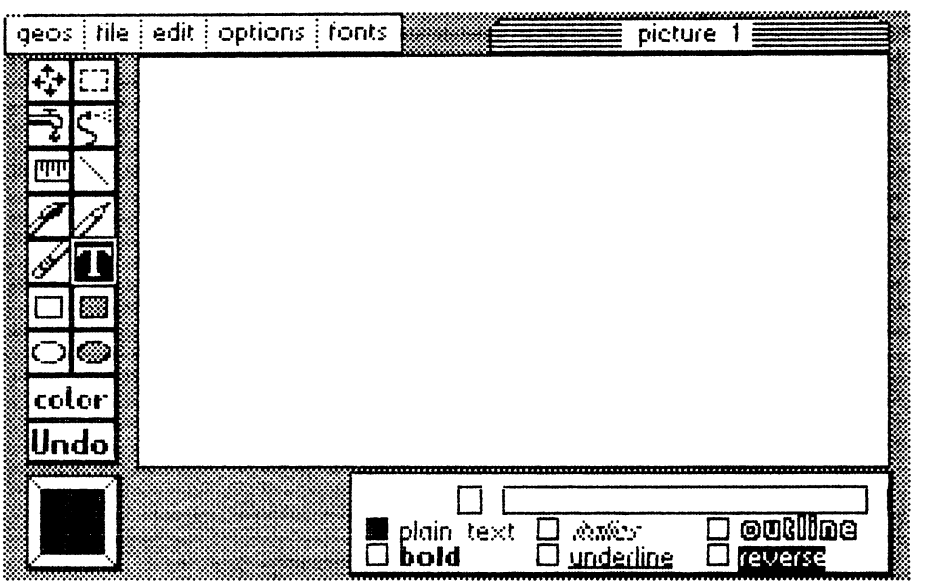


Figure 22: Text processing with GEOS

You have text control almost comparable to geoWrite. Five type styles can be combined: `Bold`, `Italics`, `Underline`, `Outline`, and `Reverse`. Each variation is selected by clicking its box in the status box, and clicking again to deactivate. `plain text` returns you to the default.

When you have the desired combination, you must mark off a text window, i.e. determine where the text will be written. You can write this text in the window as you do in geoWrite with word wrap, etc. Maybe you marked a larger text window than you needed. If you have already clicked **T**, move the pointer to the upper left corner of the desired text area. When you move the pointer to the lower right corner of the marker, the edit box becomes a dashed rectangle surrounding the text. Click when the text window is the desired size. A text cursor appears at the upper left corner, and you can enter text. When the text is ready, you can change the text as follows:

You can make changes to the font as you're typing in text. The text in the window changes accordingly. You can change text to `Reverse`, or change `Roma` to `University`, for example.

You can move the window to another location on the screen and change the size of the text window. Move the pointer to the position of the new upper left corner of the window. Click once. The window moves to that position.

You can change the size and form of the window. geoPaint redraws the text in the new window. If the window ends up being too small, then the entire text may be invisible, and dependent upon the width of the window, may be divided into more lines of differing widths. But you can change the new format of the text. To finish this procedure, close the window by clicking the **T** (text) or any other tool icon. The text region vanishes.

Rectangle: 

This tool lets you quickly and easily draw rectangles of different forms and sizes. Click the tool. Move the crosshairs to the desired upper left corner position, and mark this corner with a click. Now move to the lower right corner. geoPaint will draw rectangles as you move. When you are satisfied with the result, click again, and the rectangle is complete.

Filled rectangle: 

This function works the same as the rectangle described above, except it is filled with the pattern displayed in the pattern box.

Circle:

After choosing this tool, move the pointer to the center point of the circle and click. Then move the pointer to any point on the radius and click. geoPaint draws it immediately.

Filled circle:

Same as above, except the circle is filled with the current pattern.

Color:


geoPaint allows you to choose from sixteen colors for the drawing or background color. When you click `color`, two color lines appear in the status box. The upper line represents the drawing colors. The lower line represents the background colors. The arrow in each line points to the current choices. To change to another color, give a single click.

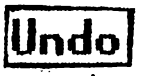
Make the point color different than the background. When you change the foreground color, every newly set point will be that color. The background color must be changed manually. For this reason, there is a square beneath the color as a tool. When you drag it to the drawing surface and click, the background area changes into the desired color. You can remove the square with another click.

There is a problem with color selection, due to a peculiarity of the '64. In high-resolution mode (in which geoPaint uses to draw documents) the foreground and background colors can only be set in blocks of 8 x 8 pixels. Thus the background color depends to a great extent upon the size of the square being changed to the new color. You may be thinking, "Okay, but I can set every new point in the foreground in different color—why not the background?"

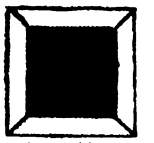
When you change the colors of set points and use different colors for different points within a block, these points may change to one color, since the 8 x 8 block interferes with the color selection.

You should keep in mind this color selection limitation—the fact that you can only change color from block to block—when you are developing colorful graphics. Try to pre-plan different graphics so that multiple color changes don't occur within the same 8 x 8 block. If it does happen, then you can always move the block to another area.

Undo:



This command allows you to recover a previous picture. (Your other option is the `recover` command from the **`file`** menu, discussed earlier). If you don't like the line you've just drawn, or you don't like the range you just marked off, `Undo` reverts to the most recent command. You cannot `Undo` commands from the menu. You must click `Undo` immediately after the erroneous command is performed. When you have drawn a bad circle, and then used the eraser, but change your mind and click `Undo`, the circle doesn't get erased—the eraser just disappears. However, `undo` can be a lifesaver.



Pattern box:

The pattern box is not exactly a tool. Rather, it displays the current pattern selected from `pattern`. To choose another, click the pattern box, and the available patterns appear. Simply click to choose from one of the 32 patterns.

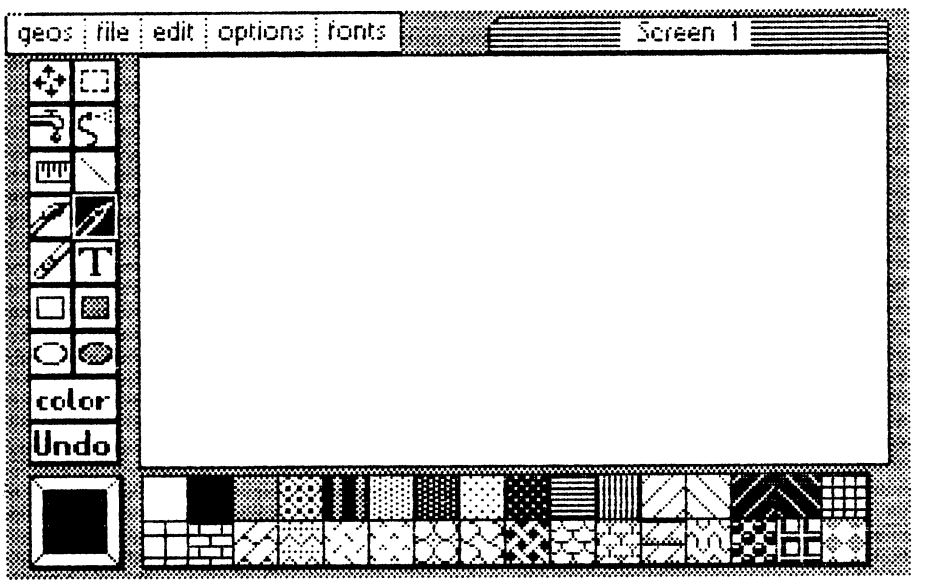


Figure 23: Pattern box with 32 patterns

3.5 geoWrite

3.5.1 Starting geoWrite

The geoWrite word processor is the second application available to you in GEOS. It has a number of features that set it apart from other word processing programs.

First, geoWrite prints text with *proportional spacing*. This means that the letter "I" is narrower than the letter "W". Proportional spacing gives your geoWrite document a more professional look.

geoWrite lets you use several different fonts, point sizes and type styles.

geoPaint documents can also be added to geoWrite text documents.

Like geoPaint, geoWrite can be loaded in one of three ways:

1) Click the `geoWrite` icon once, then click `open` in the **`file`** menu.

2) Double-click geoWrite to load it directly.

3) `open` a text document file created in geoWrite. geoWrite is loaded and then the selected document file is loaded, so you can immediately edit the file.

When you choose one of the first two items for loading, a dialogue box appears offering you three options:

`create`: Create a new geoWrite document.

`open`: Open an existing geoWrite document. geoWrite displays the first five names of a total of sixteen allowable files in a window. Click the desired name, and then click `open`.

`quit`: Returns you to the deskTop.

When you click `create`, the screen looks like this:

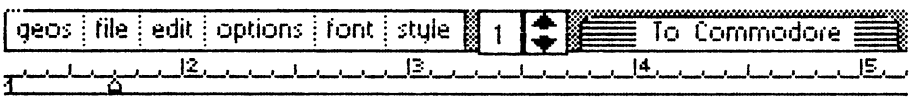


Figure 24: The geoWrite window

The *command menu* is in the upper left half of the topmost line. The page number in approximately the center, which at the moment reads `1`, is displayed. Next to this you'll see the *scroll box* (two arrows) and the title box.

The next line down is the *position line.* This corresponds to the bar on a typewriter that is used to set the left and right margins and tab positions. The rest of the screen is the geoWrite *window.* It shows part of a page which you can edit. The position of the geoWrite window on the page is indicated by the small rectangle within the page display. It is currently at the upper left corner. You'll also see your text cursor at the upper left corner of the geoWrite window. It indicates where the next character will appear when you type it.

3.5.2 The geoWrite menu

As we've seen at the deskTop, selecting an item from the command menu displays a drop-down sub-menu from which you make your selection. What are these items in these sub-menus, and how do they work?

geos:

This menu lists the names of the programmers who wrote geoWrite. The accessories that are currently on the diskette are also listed. Click the desired accessory to load it.

You cannot `select printer` or `select input` in geoWrite. If you plan to do any printing, you must remember to select the necessary printer driver from the deskTop before loading geoWrite.

file:

This menu contains all the commands applicable to files (your texts).

`close`:
This closes a document file. The file is closed, but geoWrite is still active. You can use `create` to start a new document file, `open` to call an existing file, or `quit` to return to the deskTop.

`update`:
geoWrite doesn't store document file changes to diskette immediately. Instead, this happens when you scroll to another section of the window or when you are done working on your document. It's wise to occasionally save the current changes to diskette. For example, when you want to change the contents of a document file, save the current file to diskette with `update` before experimenting. Then if you make a mistake, you can always `recover` the previous version.

`preview`:
geoWrite documents are set up in an 8.5" x 11" page format, even though it is not always visible. Use the `preview` command to view the entire page onscreen—it is displayed as a miniature page at reduced resolution. The letters won't be readable, but `preview` will at least give you an idea of the page layout.

recover:
This command works in conjunction with update. recover restores the last update to the screen. The current document file displayed on the screen is irretrievably lost.

rename:
This command lets you change the name of the document file. The current name is displayed in a dialogue box. Delete the old name with the <DEL> key and enter the new name followed by <RETURN>.

print:
You'll remember this command from the deskTop. Your entire geoWrite document is output to the printer. Remember to first select printer in the deskTop before you using this command.

quit:
This closes both the document and geoWrite, and returns you to the deskTop.

edit:

The following editing commands should be familiar to you from the sections on the photo manager and text manager. geoWrite allows you to keep often-used text passages in a text album, copy them from this album and "paste" them into a different document file. You can also paste geoPaint documents and other items from your photo album in your texts.

cut:
cut allows you to remove a segment from a geoWrite document. This works much like cutting a section (block) from the newspaper with scissors: The section disappears from your text. Unlike geoPaint, however, blocks of text aren't marked with a rectangle. Rather, you mark a block as follows:

Set the pointer on the first character of text. Press and hold your input device's main control button. Drag the pointer to the end of the block and release the button. The block is indicated by a color bar. If the block consists of several lines, then you mark the beginning (holding down the fire button) and pull the text cursor down a line at a time rather than a word at a time. Once you've marked the block, you can proceed to the next command. You can "unmark" the block by clicking any other "unmarked" area of the page.

A marked block looks something like this:

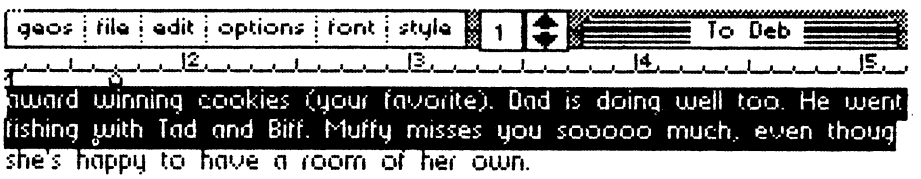


Figure 25: Marked geoWrite document section

Now move the mouse to **file** and click `cut`. The disk drive runs and geoWrite saves this section of text (block) to diskette with the name `text scrap`. You can later paste this into a text album using `text manager`.

`copy`:
This command works very much like `cut`, but the block is not removed from the geoWrite document. It is simply copied to `text scrap`.

`paste`:
This command lets you paste the current `text scrap` to a new location. You can also paste graphic documents from geoPaint. When you click `paste`, another dialogue box appears and asks whether you wish to insert text or graphics. The range is pasted at the current text cursor location. This is what it looks like when we insert a graphic into a document file:

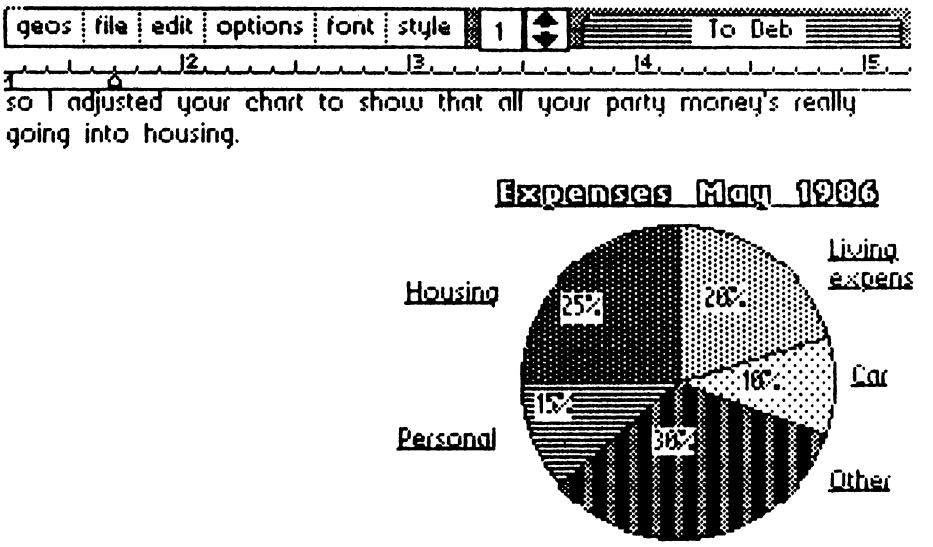


Figure 26: Graphic in geoWrite document

Please note the following when using these three commands:

`text scrap` only has room for one extract at a time. If you `cut` or `copy` a geoWrite document passage, but `text scrap` already contains a passage, the previous passage is replaced by the most recently cut passage. By the same token, you can `paste` only one text passage at a time to the `text album`.

If you wish to `paste` several geoWrite document segments, you'll have to `paste` a segment into `text scrap`, load the `text manager` and paste in the segment. Once you leave `text manager`, you can cut another `text scrap` and repeat the operation.

A geoPaint document which has been pasted in geoWrite document may be deleted like text. Position the text cursor at the first character following the geoPaint document, and press <DEL>. Unfortunately, you can't write anything in geoWrite inside of this inserted geoPaint document.

options:

This menu has essentially the same commands as in the page format section of geoPaint.

previous page:
This command goes back one page in the document (works only if your document has more than one page).

next page:
This command goes forward one page in the document. Again, the command only works when more than one page exists.

go to page:
This command displays a specific page in a document. Click this command and enter the desired page number in the document.

hide pictures:
This command is very important when you include geoPaint documents in your text document. That's because they are loaded from diskette, rather than stored in text memory. When you want to hide them, use this command. A rectangle replaces the geoPaint document. The frequent loading processes stop, and you save a lot of time: The document will be printed without graphics. But remember, this command functions only under geoWrite.

page break:
The final item in the **options** sub-menu lets you mark the end of a page. A horizontal line is drawn at the text cursor location and the text cursor is advanced to the next page. You can remove the page break by moving the cursor to the upper left corner of the page break and pressing <DEL>. geoWrite asks whether you wish to delete the last character of the preceding page (the last character is the page break). Click YES and the page break is gone.

Frequent use of page break can save you a great deal of time. When you must insert text in a multiple-page file, the altered text often spills over to the pages that follow when there is no room on the current page. page break lets you reformat the pages to make room.

font:
One of geoWrite's great strengths is that it allows you to use multiple fonts in your texts. When you click font, a number of fonts appear. Selecting the font calls up a list of point sizes. Click the desired point size. The current font is marked by an asterisk.

style:
This menu item contains the different type styles (variants) which you can select individually. The active type styles are marked by asterisks. plain text turns off all type styles and returns you to normal text.

3.5.3 Entering text

Using geoWrite, text is entered simply by typing it in. The text appears at the text cursor's present position. geoWrite displays the leftmost two-thirds of the text—when you reach the right border of the screen, the document scrolls to the left, allowing you to see its rightmost two-thirds.

You can erase the last character typed by pressing <DEL>. The text cursor deletes the character to the left. When you press the <DEL> key repeatedly until you're at the first character of a line, the text cursor moves to the preceding line. The cursor keys work differently than you'd expect. They delete characters in geoWrite, like the <DEL> key. To move the text cursor within a document, set the joystick pointer to the desired area and click.

geoWrite has *word wrap*. This means that words are not separated at the end of a line, as with a typewriter. If the word is too long to fit at the end of a line, it is "moved" to the beginning of the next line.

When you want to add to or delete from an existing section of text, move the joystick pointer to the desired area and click. Now when you type in characters, they are inserted at the new text cursor position. The existing characters are moved to the right to make room for the new text. Text to the left of the cursor is deleted with <DEL> or a cursor key. All of the keys repeat using geoWrite. Instead of repeatedly pressing a key, just hold it down; the key repeats automatically.

To insert a blank line, move the text cursor to the beginning of where you want the blank line (move the pointer, and then click), then press the <RETURN> key. The text will move one line down from the cursor, and the blank line will appear. To remove a blank line, move the text cursor to

the beginning of that blank line and press <DEL>. It is also possible to move a page break this way (geoWrite draws a line signifying the beginning of the next page). Set the text cursor past the last character before the page break, and press <RETURN>. The break moves down one line, and you now have a new blank line.

When you move the pointer to a point past the end of your text, the cursor is placed at the first blank area following the text. You cannot move the pointer beyond the end of your text. To move the end of text downward, press the <RETURN> key repeatedly. Another method is the tab, which we'll discuss shortly.

3.5.4 Formatting Text

When you want to end a line, press the <RETURN> key. This will end the current line and move the text cursor to the beginning of the next line.

Now we come to the elements of text formatting. The position line at the top of the window is for text formatting. Typewriters have a similar line for setting left and right margins and for setting tabs to jump across a number of columns at a keypress.

The position line has two M markers for setting your margins. The first is on the left side of the position line. Click it, move it to the desired position, and your left margin is set. This means that text lines will start at that position. The other marker is set at the far right of the position line. You can't see it at the moment, because geoWrite only displays two-thirds of the window at a time. Move the pointer to the right edge of the window. The window will scroll to let you see the right portion of the window and the other M marker. Click it, move it to the desired position, and click again. geoWrite will immediately adjust to the new right margin setting.

During text entry, you should set your left margin at 1 and the right margin at 5 (the numbers are on the position line). This makes the entire text line visible onscreen (important in proofreading), and saves you time since you don't have to scroll back and forth between screens. When your text is complete and ready, then you can move the left and right margins to their proper positions for printout.

You can set up to eight tabs in the position line. Point to the spot on the position line at which you want a tabulator set, at the same height as the M markers. When you click this spot, the disk drive will run, and a pointer (the tab marker) appears. A tab can be clicked and moved to another place on the position line. To remove a tab, click it and move the arrow up to the numbers on the line. If you click again, it is deleted.

How do you use tabs? You press and hold down the <CTRL> key at the top left of the keyboard then press the <I> key, or <CTRL> <I>. If the text cursor has not reached the end of a line, it jumps to the next tab column.

The geoWrite window displays only a small part of your entire text. You can move around the page in several ways. For one, when the text is wider than the screen width and you can no longer see both margins, you can overlap between screens with the pointer. Just move the pointer to the right or the left borders. You can also move up or down the window by clicking the scroll box arrows. This moves the text in the geoWrite window up or down one line. Holding down the fire button scrolls geoWrite line by line.

The page display can be used for further position changes. As you know, it displays the current page number, and is located to the left of the scroll box. In addition to the page number, the display also shows the relative position in the document as a small black square. Click this rectangle, slide it to the desired area of the document page and click again. After a few moments, the geoWrite window displays the new text location.

Finally we come to the PAGE commands, which let you move to a specific page in the document. You can mark passages in geoWrite and use different commands in these blocks. Move the pointer to the beginning of the passage (the first character), and move to the last character while holding down the fire button. The marked block will turn dark. You can now do one of the following with this block:

- cut the text and make it a text scrap
- copy and make it a text scrap
- Delete it with the <DEL> key
- Replace this block with new text, just by typing in the new text. Pressing any key automatically deletes the block.
- Select another font, type style or point size by clicking the selection. If the altered text takes up more room than before because of larger point sizes, the text following the block is moved down to make room for the new font.

3.6 Accessories

With the two applications geoWrite and geoPaint, GEOS has a number of accessories that can make your work easier. The advantage of these accessories is that they can be loaded from an application. For instance, you can use the calculator while working in geoWrite, and draw graphics from a photo album while in geoPaint.

You'll need sufficient memory on diskette to have these accessories available. A SWAP file is set up on diskette. It records the current state of your work and stores program sections that might be overwritten by the accessory. When you leave the accessory, the SWAP file is reloaded and the program picks up from where it left off. It's very important that there is enough space on the diskette to maintain this file, which can be up to 15K in size.

All available accessories are listed under the **geos** menu. A single click loads the accessory. If you prefer, you can double-click the accessory name in the **geos** menu, or click the accessory, then **file**, then open. When you want to leave the accessory, click the close icon (that small box in the upper right of the window).

3.6.1 Alarm clock

The alarm clock in GEOS has several advantages to it:

- You have a digital clock available at any time.
- You can set an exact time for the alarm to go off.
- You can save text and graphics with the date and time, to quickly determine which is the most current version of a file.

Be sure you enter the current date and time from the deskTop at the beginning of every GEOS session. The easiest way to do this is to use the preference mgr accessory, which lets you set both the date and the time.

When you load the alarm clock, the screen will look something like this:

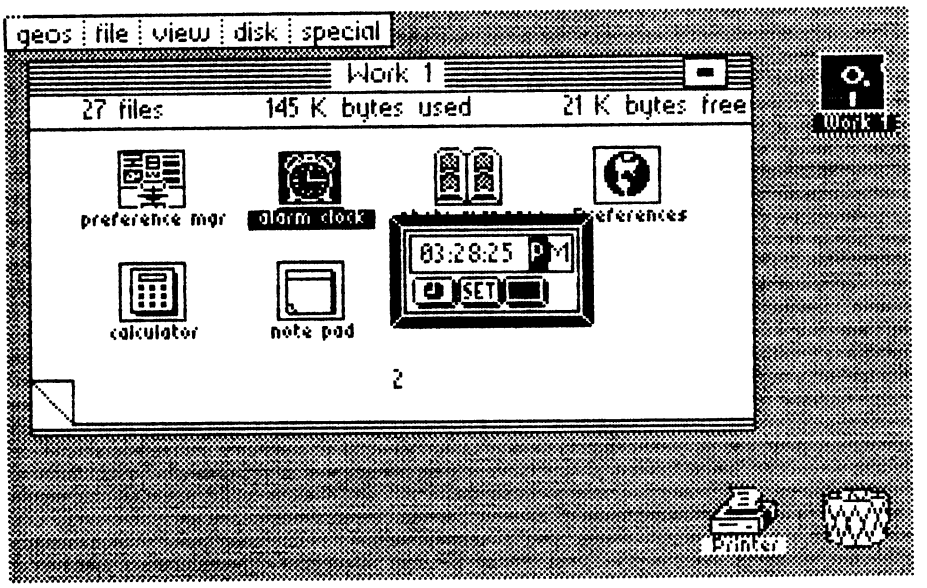


Figure 27:Alarm clock

The center section has an eight-digit time display. The first six digits are the clock time in hours, minutes and seconds; the latter two figures read AM for before noon and PM for after noon. The items beneath the time, looking from left to right, serve the following functions:

Mode button:
This gives the time mode, in which the time is normally displayed. The mode button is normally a white clock on black background. The other mode is alarm mode, indicated by a bell, which appears when the alarm is set.

Set button:
This button lets you set the time.

Close button:
Clicking this icon closes the alarm clock.

The alarm bell will appear to the right of the close icon.

To set the current time, move the text cursor (which should be at the P of PM) to the first digit by pressing the spacebar. The spacebar moves you from digit to digit without erasing or changing numbers. Now you can enter the new time. Remember that GEOS accepts only valid data. That is, you can't enter a 2 for the first digit, since hours can only be between 01 and 12. The last character may be only A or P. When you are finished setting the time, press <RETURN> or click the set button.

In the same manner you can set the desired alarm time. You must be in the alarm time mode. Click the mode button. You can toggle back and forth between modes by pressing the <M> key. Now set the time and click the set button or press the <RETURN> key. An active alarm is signaled by the display of a bell symbol.

To exit the alarm, click the close icon or press <SHIFT> and <Q> at the same time. Although you've left the clock mode, the clock remains running —and the alarm is still active. When the alarm time is reached, a tone will sound for five seconds.

The GEOS clock was designed for 60Hz electrical power. However, some European countries have 50Hz power. Because this book is distributed internationally, for our European readers we've written a routine that corrects the timing (and keeps the clock accurate at 50Hz). This program is included in Chapter 5, **Tips and Tricks**, and on the optional diskette for this book.

An annoying feature of the alarm clock is that you have to load it to get the current time. Another disadvantage is that you can't determine the alarm time without resetting it again. For these reasons, we wrote a GEOS program that displays the current time and alarm time on the screen. This program is also found in Chapter 5 and included on the optional diskette.

3.6.2 Calculator

Another accessory is the calculator. This performs the usual basic math functions. When you load it, the window appears as follows:

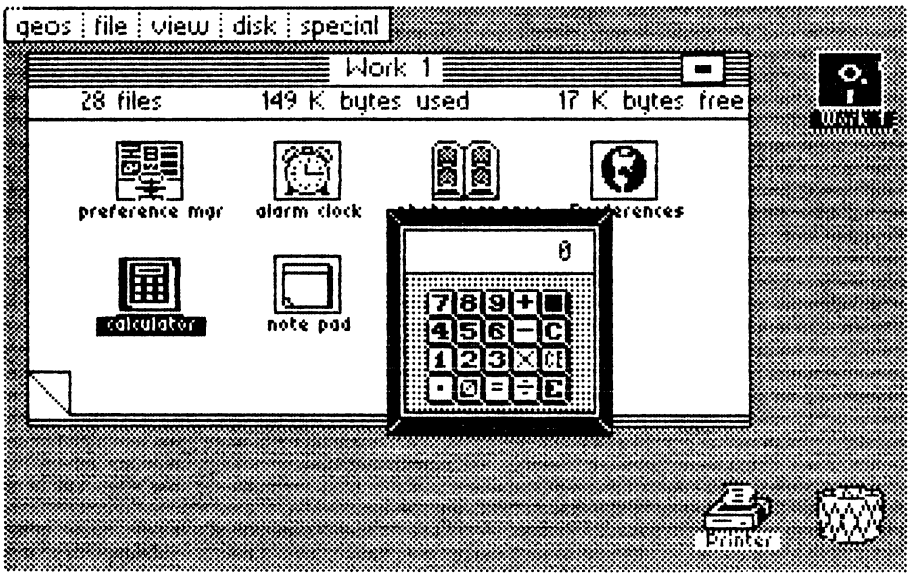


Figure 28: Calculator

There are two ways to enter numbers or perform calculations. You can point to and click the desired keys with the pointer, or you can use the C-64 keyboard to enter the numbers and operations. The keys on the keyboard have the following functions:

Addition :	+
Subtraction:	-
Multiplication:	*
Division	/
Exponentiation:	E
Clear:	C
Clear entry:	CE
Close:	<SHIFT> Q

For example, to compute the equation:

$$3.45 \times 10^8 \times 1.123 \times 10^4$$

you would type:

```
3.45E8*1.123E4=
```

The result is:

```
3.87E12
```

The range of exponents can be no higher than 37 and no less than -37.

The close icon is the dark box at the upper right of the calculator keyboard. Click this to exit the calculator.

3.6.3 Notepad

To use an "old-fashioned" desk effectively, you need a notepad to jot down times, notes, appointments and comments during conversation, just to mention a few examples. GEOS has a notepad too. It allows you to keep up to 127 pages of notes and store them on diskette as a notebook. Thus, the notepad and notebook go together.

After loading the accessory, a window displaying dogear pages appears. Unfortunately, it is impossible to directly access pages with the notepad as with the deskTop. Perhaps this drawback will be eliminated in a later version of GEOS.

You can enter your text into the notepad as usual. The <DEL> key deletes the last character typed and moves the text cursor one character to the left. You've already seen this from geoWrite. We have already typed in a page on our notepad, so you can see how it works:

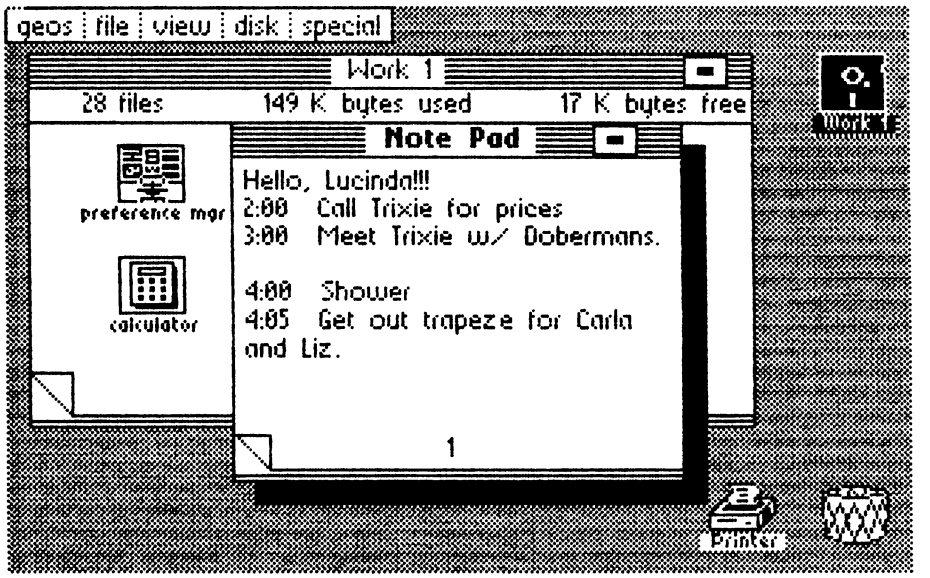


Figure 29: Notepad

When you are finished reading or writing your notes and want to exit the program, click the close icon in the upper right corner of the window. When you want a fresh notepad, you can move the existing pad into the waste basket. This erases its current contents, and you have a fresh notepad.

We've skipped two other notepad functions that would make GEOS life easier. One is that a page can be deleted on a keypress. The other is that you can flip through pages if you so desire.

3.6.4 The Preference Manager

The Preference Manager is an accessory that lets you specify certain program parameters (sizes, values) such as background color. These values can then be saved and recalled at any time.

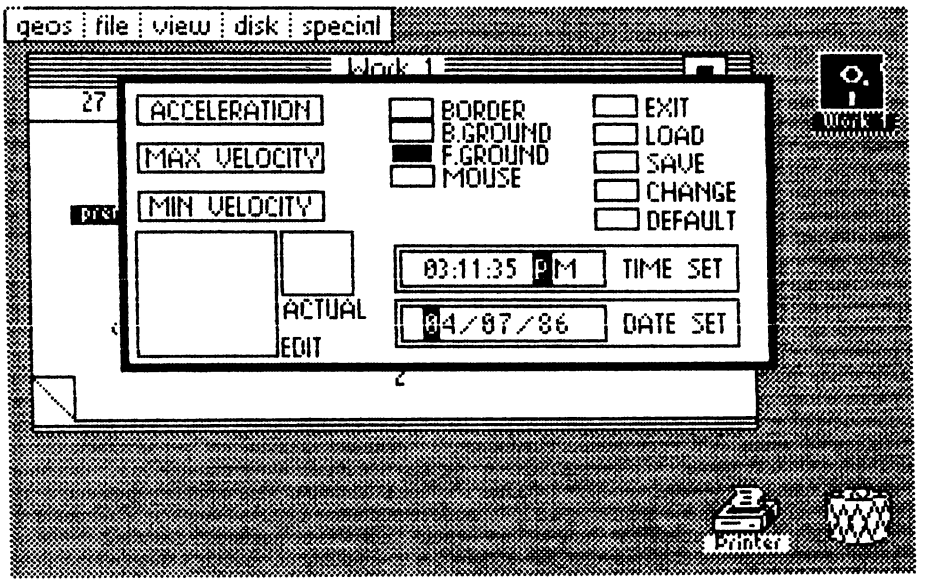


Figure 30: Preference Manager

The Preference Manager window is divided into several sections. The first three lines to the left control the speed of the pointer. Below is a window that lets you change the appearance of the pointer. Later we'll see how it works. The top center specifies the color selection for the border, foreground, etc. Below are the two windows that let you set the time and date. The upper right contains the command list for the Preference Manager.

The pointer speed controls are in the upper left corner. You can change the parameters and experiment until you find the "right" settings.

Pointer speed:

ACCELERATION:
This control lets you set how quickly the pointer reaches maximum speed. When you work a lot with the deskTop, rapid acceleration is important if you move the pointer frequently for clicking OK, etc. Click the control bar knob to a new position and click again to set it. Moving the knob to the left slows the pointer, and moving it to the right speeds up the pointer.

MAX VELOCITY:
Here you set the highest speed at which the pointer travels after accelerating. Again, you'll want high speed if you frequently use the deskTop (shifter all the way to the right).

MIN VELOCITY:
This control bar adjusts the starting speed of the pointer.

If you work a lot with geoPaint you probably want to start with a slow minimum velocity and average acceleration. This makes it simpler to set the center point of a circle, for example.

If you want to see the effects of the change immediately, click the CHANGE button. We'll discuss the other buttons shortly, but we mention CHANGE so that you can see the effect.

Pointer appearance:

The large rectangle in the lower left displays the pointer in bitmap mode, while the small rectangle displays it as a normal sprite. To change the appearance of the pointer, move the pointer into the large rectangle. This window's pointer works as a small pencil and eraser, similar to geoPaint's pixel edit mode. When you click where there is no set point, a point is turned "on". Clicking a point turns it "off". As you turn pixels on and off, you can see the results in the small rectangle. The pointer with which you are working does not change. The reason for this is that if you deleted all the points, you'd have this invisible pointer, and you'd be unable to use the Preference Manager any longer. This is another reason why alterations don't take effect until CHANGE is clicked.

Colors:

Each respective range or area changes color when you click the corresponding button. You can choose from 16 colors. They appear in order as you click the button. You can "cycle" through all the colors by clicking 16 times.

BORDER changes the border color. This is not the deskTop border which we use to copy files. Instead, it's the edge of the entire screen area.

B.GROUND controls the color of the background, as you may remember from geoPaint.

F.GROUND changes the color of the foreground, i.e., set points and colors.

MOUSE controls the color of the mouse pointer.

There is a limit to the choices of colors. You are not allowed to choose the same colors for background, foreground or pointer. There is good reason for this: The wrong choice could result in an invisible pointer. Then you wouldn't be able to read error messages, for example.

Time and Date:

We've already discussed the time and date—you should always set them when working with GEOS. We'll just repeat the important aspects:

You can change time and date by clicking TIME SET and DATE SET with the pointer. Move the text cursor over the characters with the spacebar. They are not erased by the spacebar. Enter numbers from the keyboard. GEOS allows only valid input, e.g., A or P for morning or evening. When you're ready, and the entry is complete, press <RETURN>. If you've forgotten and already exited the area with the pointer, the date and time will assume the default values.

Remember to set the date and time to the current values whenever you start GEOS. The date/time data is saved along with every geoWrite and geoPaint document. This can be extremely useful when trying to determine the most up-to-date version of a graphic document or text document.

Command buttons:

EXIT:
This button exits the Preference Manager. Your parameters are not automatically saved or executed.

LOAD:
This button loads saved values.

SAVE:
Saves the current values. These are not necessarily the displayed values. You must activate the new values by clicking the CHANGE button.

If the values aren't saved, the alterations are current only for the duration of the GEOS session. Returning to the deskTop or re-booting GEOS will reset the previous parameters. To retain the new preferences, be sure to SAVE them. A Preferences file is saved to diskette, and is reloaded when the diskette is later opened.

CHANGE:
Activates the changes. When you have changed parameters by moving the knob for MIN VELOCITY to the left, the pointer will move more slowly. Values altered with CHANGE can then be stored with SAVE.

DEFAULT:
This command can be very useful when you wish to set all parameters back to the default values, i.e., back to the values used before you started changing the preferences with the preference mgr.

3.6.5 The Photo Manager

The Photo Manager lets you organize a collection of your best pictures and sections of graphics in a computerized photo album. You can later copy these photos from the album and paste them into text or graphics. Each photo album can contain up to 127 pictures. This is usually enough room —but make sure there is enough diskette space. The picture sections must be in geoPaint format.

Set a range with the range marker from the toolkit and click either cut or copy from the **edit** menu. This segment is then designated as a photo scrap, which is the name of the file created on diskette to store it. This photo scrap file connects geoPaint and the photo manager, and allows a picture to be transferred.

Note: Only one photo scrap is allowed at a time.

When you want to load the photo manager, and you already have a photo album on diskette with which you would like to work, you can double-click the photo album from the deskTop. The photo manager is then loaded, along with the desired album. The alternative is to first open the photo manager. The following dialogue box appears:

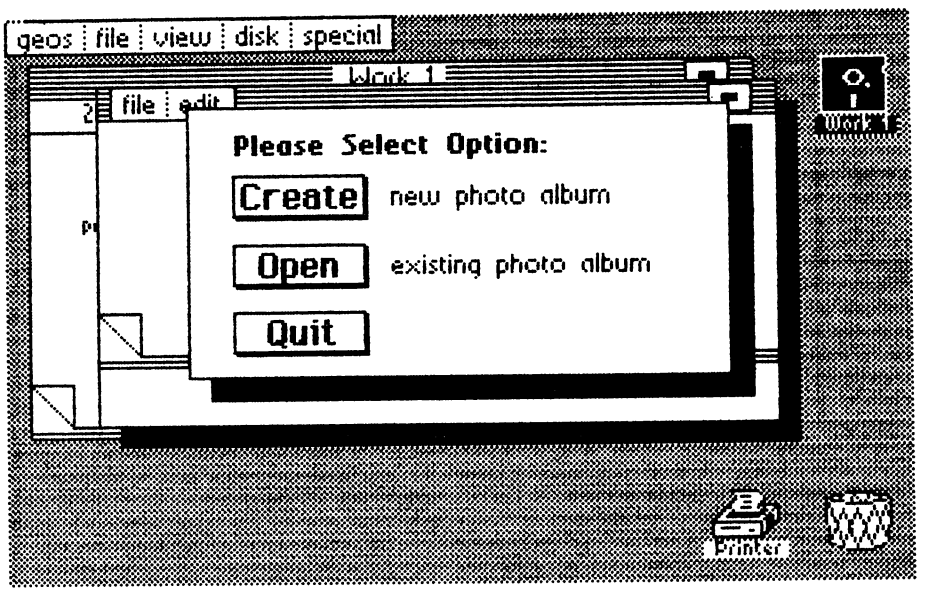


Figure 31: Photo Manager—starting menu

The three choices are already familiar to you from geoWrite and geoPaint. You can choose create to make a new album in the photo manager. open displays the first five albums on the diskette; a total of 16 albums can be displayed by clicking the scroll boxes. Click the desired name (which turns dark) and open to open the file. quit returns you to the deskTop.

When you have created a new album, a window appears and displays `Empty Photo Album`. If the album already exists, the first document is shown. The dogear in the lower left corner lets you flip through the pages of the album, and the close icon at the top of the window closes (exits) the album. Here is a page from our own `Photo Album`, so you can see the layout:

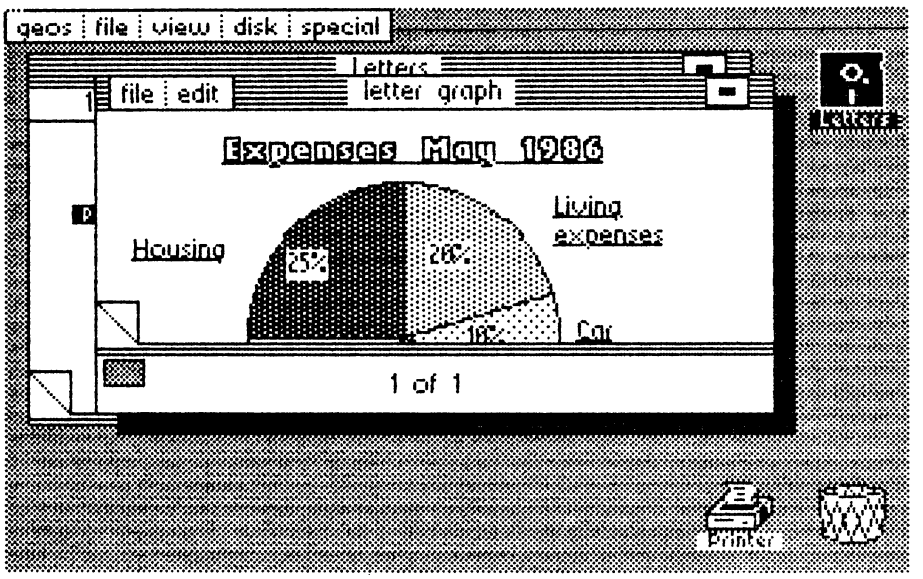


Figure 32: `photo manager`—Graphic for `To Deb`

The upper left corner is a menu with two sub-menus. **`file`** has the two items `close` and `quit`. `quit` exits the `photo manager`, and `close` closes the album and returns you to the `create/open/quit` dialogue box.

`edit` offers the following items:

`cut`:
Cuts the indicated picture from your album and sets it in `photo scrap`. From there you may `paste` it into a text or a graphic.

`copy`:
Copies the indicated picture into `photo scrap`, without removing it from the Photo Album.

`paste:`
Pastes the `photo scrap` picture into your Photo Album. The picture is copied into the album, and also remains as a `photo scrap`.

In the lower left corner of the window (beneath the dogear) is an icon you'll need when the geoPaint segment doesn't completely fit into the `photo manager` window. The black rectangle gives the size of your picture. The rectangular borders represent the window. You can click the border and insert the picture in another range. This appears in the window.

To help you better understand what you can do using the `photo manager` and a little paste-up, we've written a letter in geoWrite. Now we'll include a geoPaint graphic.

Assuming you have a geoPaint document ready, delimit the range using an edit box, and `copy` the range into `photo scrap`. Now we load the `photo manager` and `paste` the `Expenses` document into the album. Then we exit the `photo manager` and open the existing letter file `To Deb`. Move the cursor to the end of the text. Now click the `paste` function from the **`edit`** menu. The result should look something like this:

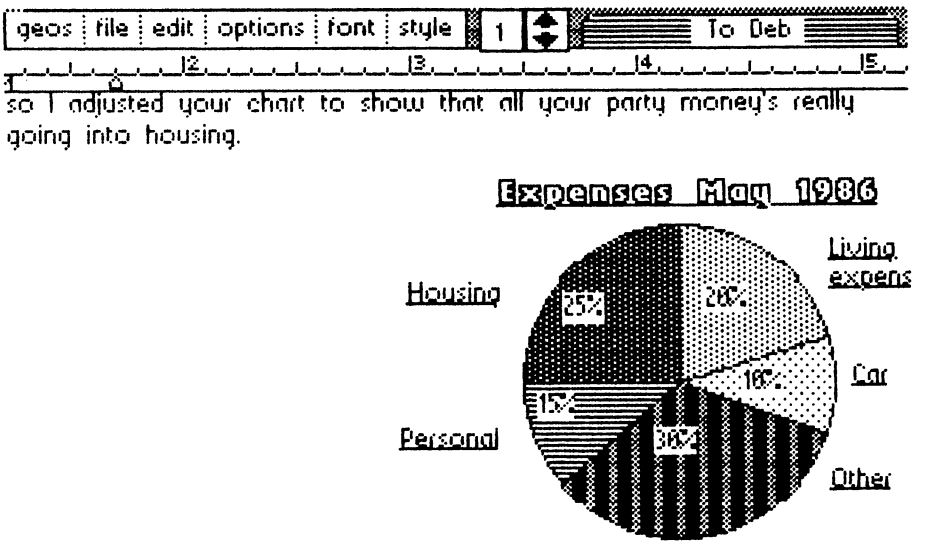


Figure 33: Text and graphics with the `photo manager`

This text/graphic integration was not easily accomplished before GEOS. We should mention that the picture was made in about ten minutes. Maybe you can dream up more applications with GEOS.

3.6.6 The Text Manager

The Text Manager lets you store often-used passages in several Text Albums, and paste these passages in new documents as needed. Each Text Album can have up to 127 pages, the same as Photo Albums and notepads. The text passages must be cut from documents created by geoWrite. Mark the text by setting the pointer to the first character, press and hold the fire button, and move the pointer to the end of the text passage. The text range will turn dark.



Figure 34: Marking text for `text scrap`

Now click **`edit`**, then click either `cut` to cut the segment, or `copy` to make a copy of it. This segment is saved to a file on diskette called `text scrap`. This file may contain only one text segment at a time. It serves to connect geoWrite and the Text Manager, acting as the "go-between".

You can go from the deskTop to the Text Album, if a Text Album is on the diskette. The Text Manager is first loaded and then the album is opened. The other option is to open the Text Manager first. After a few seconds a dialogue box appears with the same items as geoPaint or geoWrite (`create`, `open` or `quit`). `create` is used to create a new album. `open` displays the first five album names on the diskette. To see the rest (maximum of 16 albums per diskette), use the scroll boxes and click down or up. Click the desired name (it will turn dark), then click `open`. `quit` returns you to the deskTop.

When you `create` a new album, the following window appears:

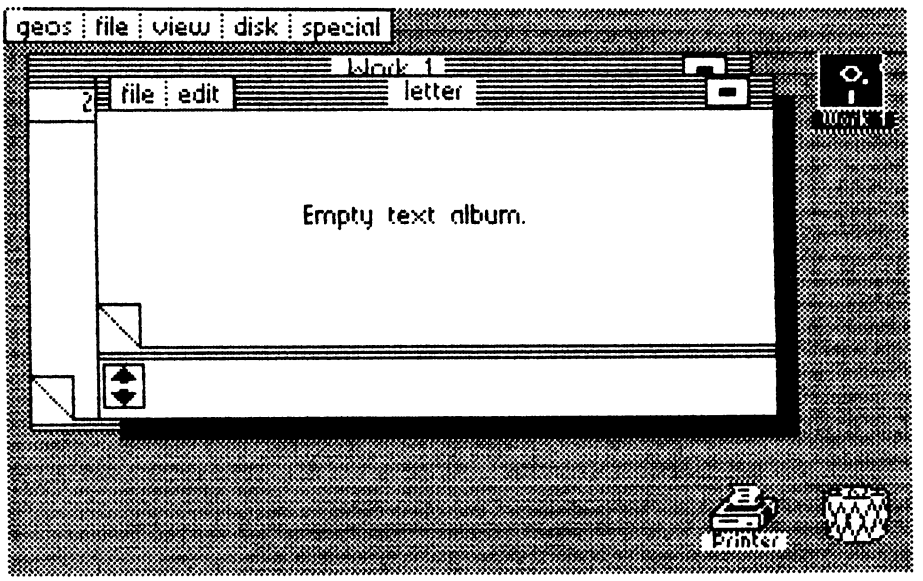


Figure 35: `Empty text album`

If the album already exists, the first page of text is displayed. You can flip through the pages using the dogear at the lower left corner of the window, or close (exit) the album with the close icon at the top right corner of the window. The upper left corner contains a short menu with two sub-menus. **`file`** has the items `close` and `quit`. `close` closes the album currently open and sends you back to the window menu `create`; `open` and `quit` exits the Text Manager and returns you to the deskTop.

edit offers the following items:

cut:
Cuts a marked text area from the album and sets it into text scrap. From there, you can paste it into another document.

copy:
Copies the marked text into text scrap. Unlike cut, the original text is not deleted and remains in the Text Album, and is copied into text scrap.

paste:
Pastes the text scrap into your Text Album. The text is copied over, so you can use text scrap for other purposes (even another Text Album).

Below is an example of marked text pasted into a Text Album, so you can see how it's done:

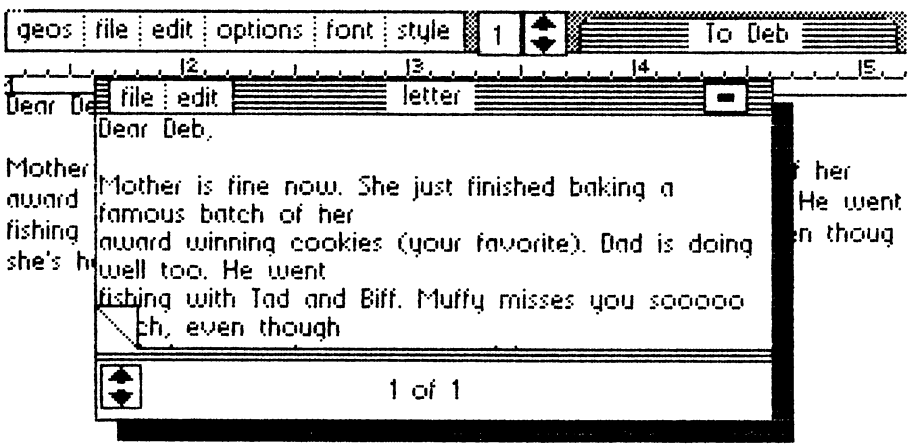


Figure 36: Marked text in the Text Album

Notice that in Figure 36, the text looks somewhat different from the original text. For example, `Love, Gene` was in a different font altogether. Text passages in a Text Album revert to a single font and point size. However, GEOS remembers the original fonts and type styles, and when you paste a text passage back into a document, the passage appears in its original fonts, type styles and point sizes.

The lower left corner of the window, beneath the dogear, contains a scroll box that allows you to see documents that are much larger than the screen allows. Clicking either arrow scrolls through the document one line at a time. You can, therefore, paste larger text passages into the album. You don't need to limit yourself to the onscreen page size.

Chapter Four

GEOS Applications

GEOS Applications

By this time you've probably learned a lot about GEOS' features. You've created your first graphics with geoPaint and written documents with geoWrite. You may have come to the point where you are not quite sure what more you can do with the system—you've written letters with interesting text fonts to all your friends, and you've run out of ideas for new graphics.

This chapter may help you see some of GEOS' capabilities and give you more concepts of practical applications. Not only will we present ideas—we'll go through the ideas step by step. In this way you'll also learn how to produce the effects you want with less work.

4.1 Making diagrams with geoPaint

You need a geoPaint work diskette with at least 30 K of empty space for this section (we'll refer to this work diskette as `Diagram`). Format a diskette using the `format` item in the **`disk`** menu. Copy the Photo Manager, geoPaint, any fonts and printer driver you'll need to this diskette.

Note: Some versions of GEOS may not allow you to copy geoPaint (try copying and running geoPaint on a work diskette). If your geoPaint won't copy, use your applications diskette for the applications in this chapter.

Before GEOS was developed it was difficult to mix graphics and text with the C-64 computer. At best, it looked as though the graphics were added later. It is much easier to mix graphics and text to form a document using GEOS. We'll look at one application of this now: producing diagrams.

You'll remember pasting a pie chart into a document using geoWrite and the Text Manager. In this chapter we'll examine how you can use geoPaint to produced the three most common types of diagrams:

- Pie charts
- Bar charts
- Line graphs

As an example we've chosen a fictional expense breakdown for May 1986. This can be represented well by the first two kinds of charts—but a line graph does not lend itself to this kind of data. However, we created a line graph for the May 1986 expenditures, even though it is a poor representation, to show you how to create the relevant graphics for each type of diagram. Using the methods shown here you can produce useful charts for your own applications.

Before we can represent the data graphically, we must first have it in front of us. Our example is the following:

Household:	May 1986
Housing:	25%
Food:	20%
Car:	10%
Miscellaneous:	30%
Personal:	15%
Total:	**100%**

Now we'll start to graphically represent the data. We'll start with a pie chart.

4.1.1 Pie chart

First open the work diskette `Diagram` and load geoPaint. Choose `create` and enter `Pie May` as the name of the graphic to be created. Now we can begin to draw the chart.

We'll start with the title. We click the `T` tool (Text) and then click the `BOLD`, `UNDERLINE`, and `OUTLINE` type styles. Then we type `Expenses May 1986` in the upper middle section of the window.

Before we draw the circle, we must mark the center. Otherwise it is difficult to draw the lines needed for the pie chart. You'll notice this right away when you try to place the crosshairs exactly where you want the center. If the joystick pointer (in this case the crosshairs) moves too fast for you, load the `preference manager` and set `MIN VELOCITY` as far as possible to the left, and `ACCELERATION` approximately in the middle. Now the joystick pointer moves more slowly, so it's easier to do precise work .

We'll use the pencil icon to draw a small cross to mark the center of the circle. Click the circle in the toolkit and set the marker exactly on this cross. If you have trouble getting the circle and the cross aligned (for example, you're a little too far to the left and then to the right, etc.), it helps to move the cross several points over and try again.

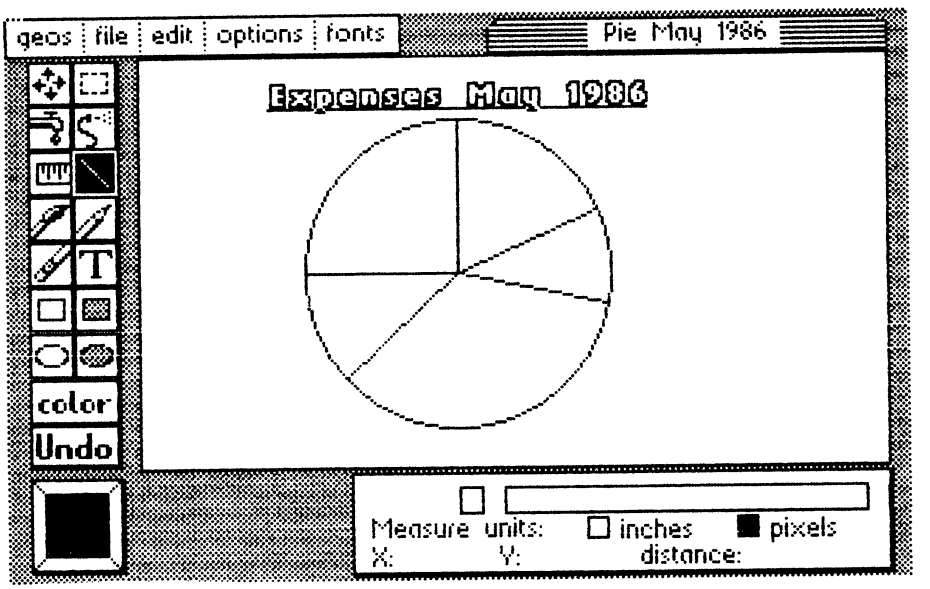


Figure 37: Title and circle

When you've gotten that to work (don't worry—it gets much easier with practice!), click and move the circle so that it is located approximately where ours is in the illustration. It's important that the whole picture fits on the screen. Everything that is drawn outside the geoPaint window is lost.

Once the circle is complete, we need to add several lines to the pie chart. Choose the line tool. Click the center of the circle and then move the line upward. A second click stops the line. Now we need to draw a section of pie that contains 20% of the whole. Since a quarter of the pie is exactly 25%, the piece must be a little smaller than a quarter. By "eye-balling" it, you should be able to do this with no problem.

Now draw the lines for the other percentages. If you draw a line incorrectly, you can erase it using the `Undo` function. If you don't like the way you divided the circle, and just want to get rid of individual lines, you can erase

the lines in `pixel edit` mode. Make sure that your lines and the circle remain enclosed figures—i.e. don't delete a few points from them. Otherwise, when you use the fill patterns, the patterns may spill into other areas.

Once you've drawn the circle and divided it into appropriately sized sections, you're done with the hard part. The rest is easy. Now let's fill the sections with different patterns. Click pattern fill (faucet icon) and then the pattern box. We now have 16 different patterns available. Choose the desired pattern, move the joystick pointer the the desired section of the pie chart, and click. The program takes care of the rest. Once you've filled the five pieces of pie, the result looks something like this:

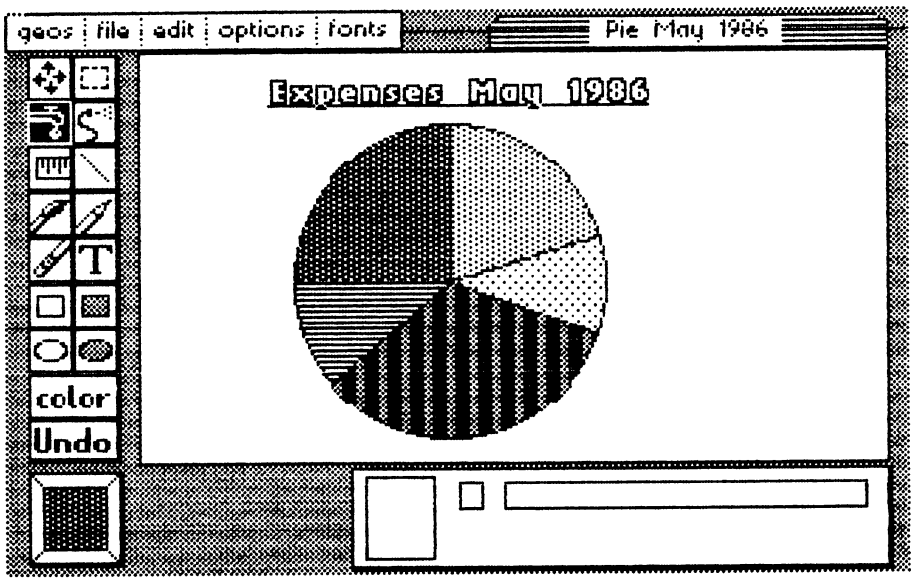


Figure 38: Chart segments with fill patterns

To make the graphic even easier for others to understand, we can label the sections with their respective percentages. Click **T**. Then choose `BSW 9 Point` from `fonts` and `plain text` in the status line below.

Now move the pointer to the first section of pie (upper right section of diagram). You need to create a text window there so that the text can be written in it.

Click the upper left corner of the text window and move it about half an inch down and to the right. It doesn't need to be too big since it only needs to hold three characters: 20%. The result should look like this:

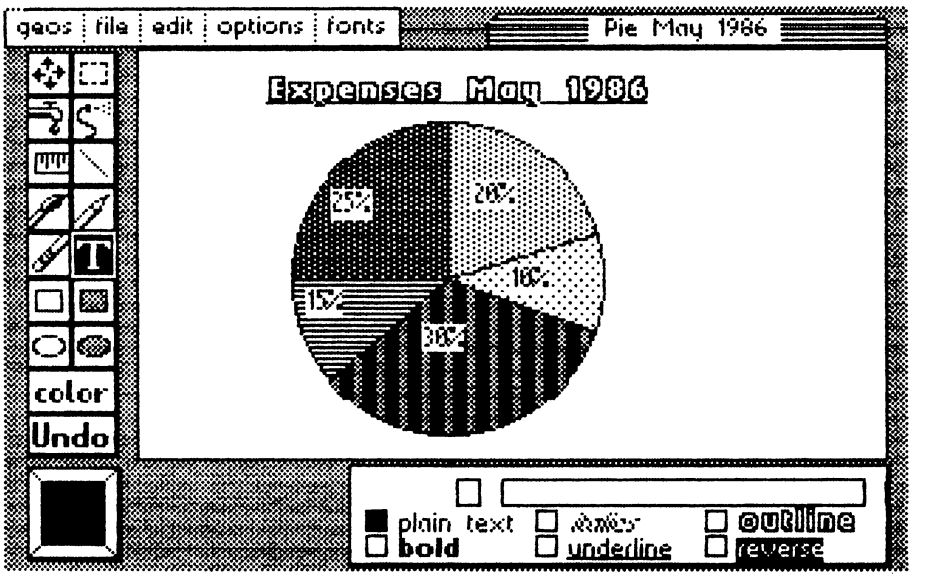


Figure 39: Chart segments with actual percentages

When you like the size of the window (it should be a little bigger than the light area in our pie), click T. A cursor appears in the left part of the window. Now enter the three characters 20%. If you're satisfied, click the T again. The text window disappears and your input is saved. You can change the position of the three characters by simply clicking the new upper left corner of the text window. (Keep in mind that problems may occur if this corner is inside the previous window). Then click a point outside of the circle, move the window, and click again.

geoPaint has created a text window there. Now you can click on the desired point in the piece of pie—it isn't in the window anymore. That's a bit complicated, but it can help us do our work.

When you're satisfied with the layout of the three characters and have closed the text window by clicking T, write the other four percentages in the appropriate pieces of pie.

Now we're almost done. We just need to label the sections so that those who look at the chart will know what each section represents. Click `underline` in the status box. Write the items next to their chart segments. The results should resemble the following:

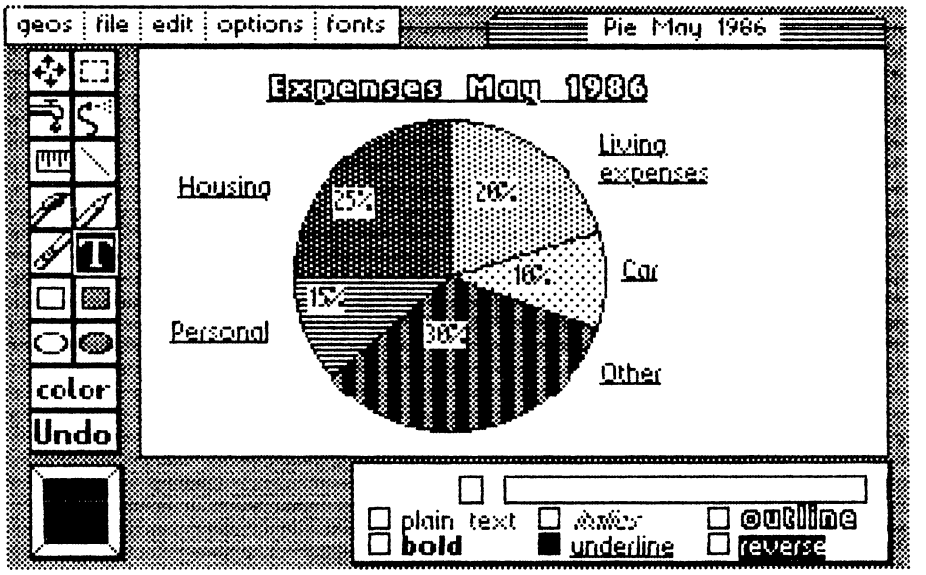


Figure 40: Finished pie chart

You can also add lines from each item to its section of the circle. Now your pie chart is done. It's a good idea to put it in a photo album so that you can paste it in various texts. Delimit the entire figure with the edit box and choose `copy` from the **`edit`** sub-menu. After a few seconds, the section is stored in the `photo scrap` file.

If you want to paste it in a photo album immediately, load `photo manager` from the **geos** menu. If you don't have an album yet, create one with the name `Graphics`. When the `photo manager` shows you the empty album, choose `paste` from **`edit`**, and your pie chart will appear in the window. `quit` returns you to the `geoPaint` program.

If you now want to create a bar chart using these percentages, close the `Expenses May 1986` file (click **`file`** and then `close`). The next section describes how to create a bar chart.

4.1.2 Bar chart

Click `create` and enter `Bar Chart` as the name. With the last example, we made sure that the pie chart fit in the geoPaint window. That makes the work a lot easier, but has the disadvantage that the items must be fairly small. Now we want to make a figure larger than the window.

First we need to decide how large we should draw the chart. Since the largest percentage value from our data table on page 88 is only 30%, we don't need to extend the scale to 100%. We settled on a bar chart that goes from 0% to 50%. We also decided to represent each 10% increment with 40 pixels. Each bar should be 50 pixels wide. We also included another 20 points per bar to create a three-dimensional effect. With these criteria, we see that the vertical axis must be 5 x 40 = 200 pixels long and the horizontal axis must be 5 x (50+20) = 350 points long. The bar chart won't fit in the geoPaint window.

Now we can start drawing. The page marker shows the upper left corner of the geoPaint window. We'll put the vertical axis about an inch from the left boundary so that we'll have enough room to write the percentages. Click the line tool and place the crosshairs in approximately the middle of the screen (but keep it an inch from the left border). Then move the line until you reach the bottom of the window, then click it there.

We want to move the geoPaint window as little as possible, since doing this takes a lot of time. We won't finish drawing the vertical axis now (otherwise we'd have to move the window). Instead, we'll draw the axes' divisions right away. Put the crosshairs on the upper end of the axis and draw a 10-pixel-long dash to the left. We'll mark this position with 50% later.

Now we can mark the position on the chart where 40% is located. It should be 40 pixels below the top division marker. Since we don't want to have to count out 40 pixels, we'll set the coordinate pointer in the status line. We set the cross on the top point of the axis and click. These coordinates are set to zero and we can move to exactly the right spot on the axis. Once you've found it, click there and set these coordinates to 0.

Now click again to draw a line to the left (10 pixels in length). The division marker for the 40% is now finished. Now we want to label the axis in the visible area. Click **T** and choose the `BSW 9 Point` and `bold` items. Now write in the percentages. The visible part of the geoPaint window is now finished.

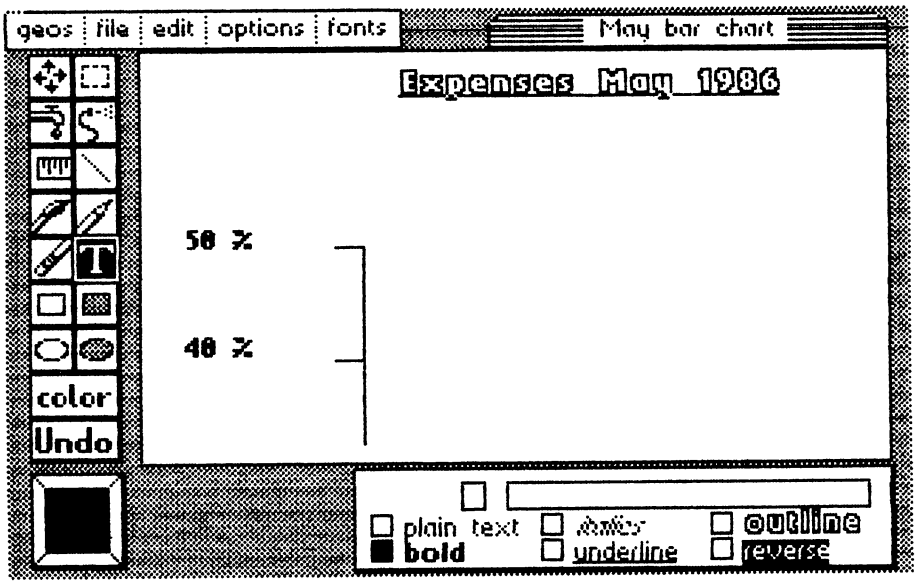


Figure 41: Starting the vertical axis

Now we need to move the window down so that we can draw the rest of the vertical axis. Click the scroll box, and move the window as far down as you can while still being able to see the 40% division marker.

Now draw the rest of the axis with the division marks. It's quickest and easiest if you draw a 40-pixel line down and then a 10-pixel line to the left. Eventually you'll need to move the window a bit further down to enter the zero point. Once this is done, we can start on the horizontal axis.

We need to divide the line into sections of lengths 50, 20, 50, 20, etc. We can use these marks to draw the individual bars later.

Our chart should look something like this now:

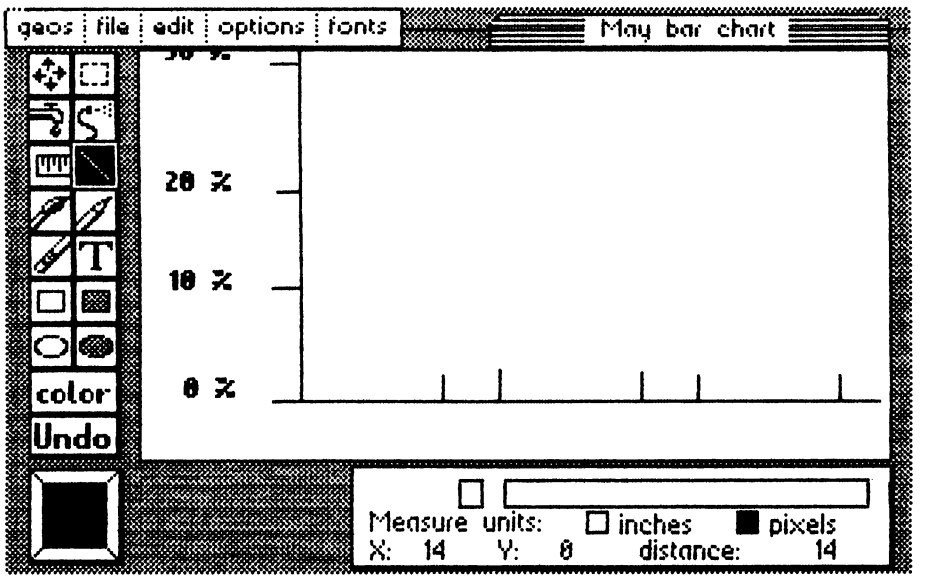


Figure 42: Zeropoint and horizontal axis with divisions

Move the geoPaint window far enough to the right so that you can draw the whole axis.

When you're finished, it might be a good idea to write the descriptions right away. Since you've marked in 50-pixel-long sections, you can easily see where they go. The window is already all the way on the right, you can label the sections from right to left. Write `Housing` for the last section, then `Personal`, `Other`, `Car`, and `Living Expenses`. For the last three, you'll need to scroll far enough back so that you can see the zero point.

Now we can draw the bars. We need to calculate the percentages in pixels. That's very easy: 10% = 40 pixels

The first column for food must be 60 pixels tall. It's best to click the line icon and set the crosshairs exactly on the zero point (0%). If you click here, move 60 pixels up the axis and you have the correct height for the bar.

If you keep the crosshairs on the axis, you can find exactly the correct coordinates. Double-click to set the coordinates displayed in the status box to zero again and move 50 pixels to the right (width of the bar). Then draw a line from there to the horizontal axis.

Now let's add a third dimension to the bars. We can do this as follows:

Set the crosshairs on the upper left corner of the bar. Now draw a line 20 pixels to the right and 10 points up. Click and then draw a line 50 pixels long toward the right side of the screen. Now draw another 20 pixels to the left and 10 down. You've finished the top of the bar.

It should look like this:

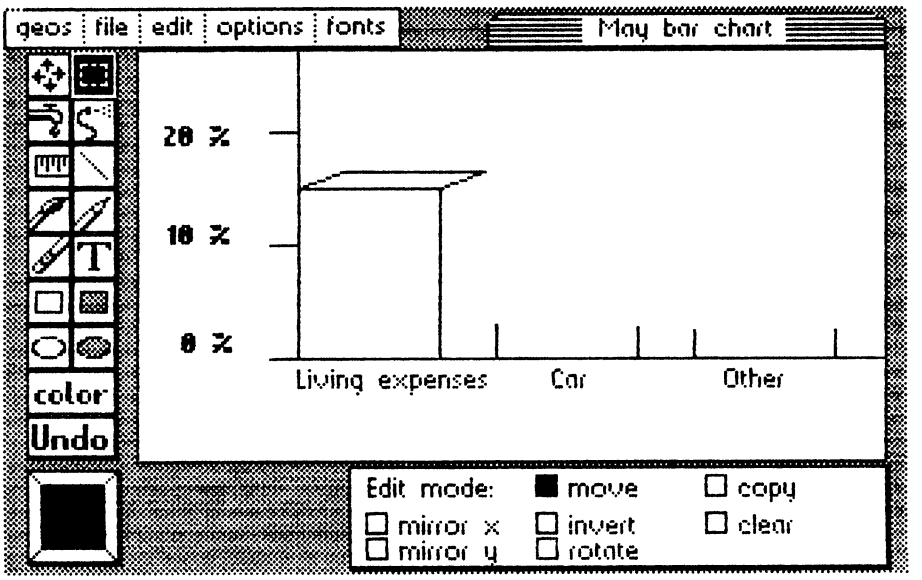


Figure 43: Two sides of a bar

If you draw a vertical line from the right rear corner of the bar to the horizontal axis, the figure is finished.

Now repeat this for the other bars. Once you've finished, the bar chart is almost done. You can improve the appearance even more by using the fill patterns. Click the current pattern icon to display the pattern options in the status box. Then choose the desired pattern by pointing to it and clicking. The new pattern appears in the current pattern icon. Now click the faucet icon, place the crosshairs inside the bar to be filled, and click.

Our chart looks like this:

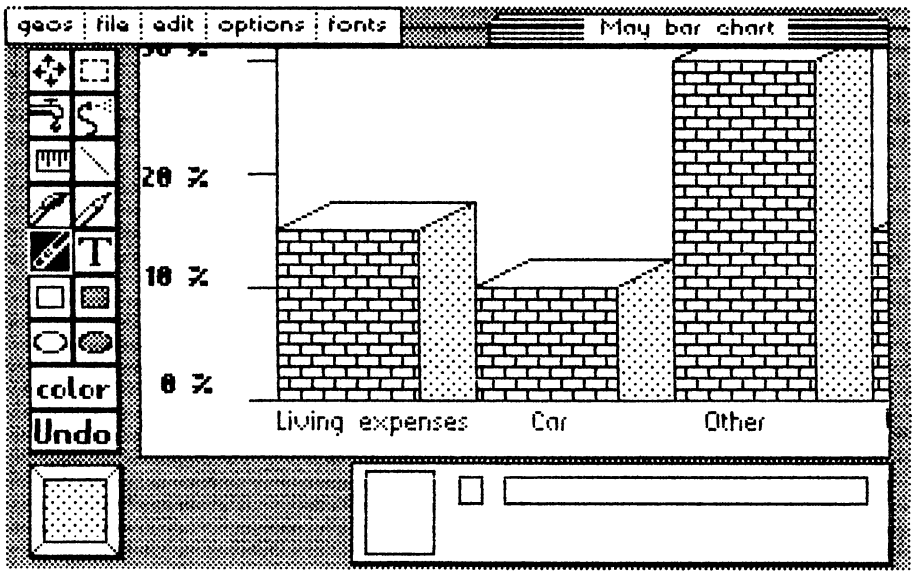


Figure 44: Section of the finished bar chart

Since the graph is bigger than the window, it's useful to look at the whole figure at the same time. Choose `preview` from the **`file`** menu. The page is reduced and displayed. Now you can see how the chart appears on the entire page. The full page in our example looked like this:

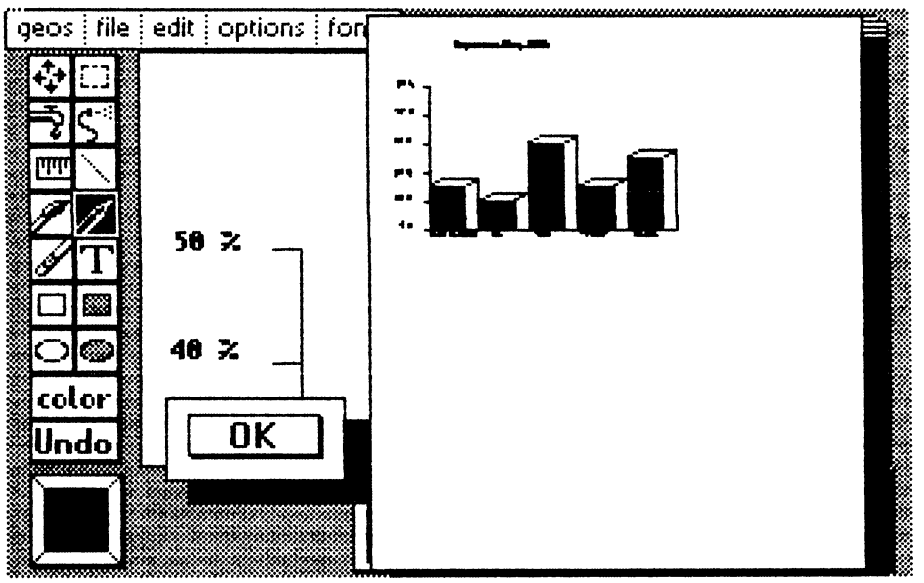


Figure 45: Bar chart in `preview` mode

4.1.3 Line graph

A line graph isn't a very good representation of our `Expenses` data. They're best when you want to display data with a time factor—for example, expenditures over a year. But we'll graph it anyway so you can compare the various kinds of diagrams.

First we need two axes. To make it easy to compare a bar chart and a line graph, we'll divide the axes in the same way as we did for the bar charts. Each 10% division should be represented with 40 pixels. If you have trouble doing this, refer to the previous section where we describe the steps required.

After you've drawn the axes, draw the lines for the chart. Start with the line icon at the zero point and draw a line 50 pixels to the right and 80 up (20%). Then draw a line 50 pixels to the right and 40 down (10%), etc... We changed the labeling of the lower axis to make the diagram more readable. To let you compare the two kinds of charts more easily, here is a section of our line graph:

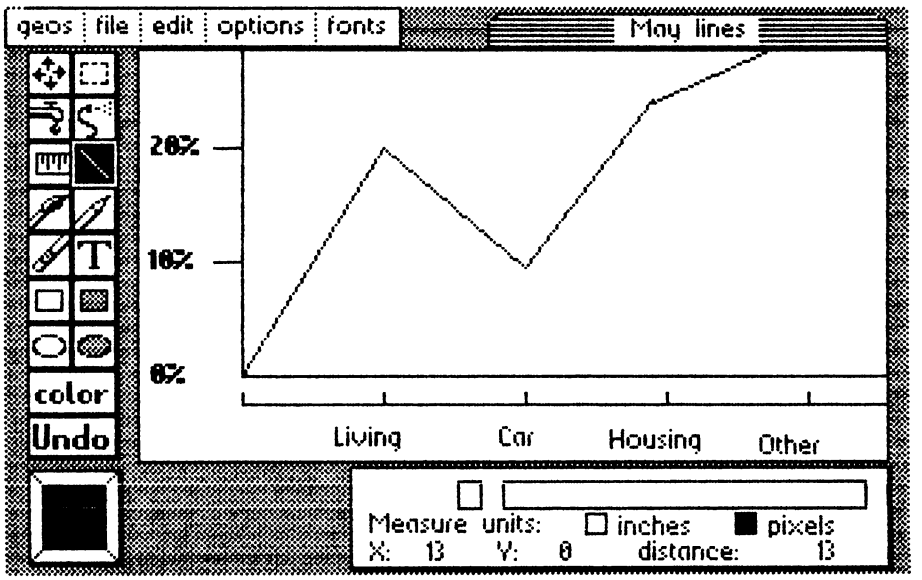


Figure 46: Line graph

We hope that these examples help you produce charts and graphs more easily. Using these three types of diagrams, you can graphically represent most kinds of data.

4.2 Organizing and planning with geoPaint

You'll need a geoPaint work diskette with at least 30 K of empty space for this section. Format a diskette using the `format` item in the **`disk`** menu. Copy the Photo Manager, geoPaint, any fonts and printer driver you'll need to this diskette.

Note: Some versions of GEOS may not allow you to copy geoPaint (try copying and running geoPaint on a work diskette). If your geoPaint won't copy, use your applications diskette for the applications in this chapter.

When drawing objects, you may find that you need to insert identical objects several times. You may need to rearrange the objects to get them the way you want them. Using geoPaint you can add and arrange objects until you've got the form and positioning of the objects exactly the way you like it. We'll look at these capabilities with two examples. You'll see ways to make your work easier by using geoPaint's capabilities fully.

4.2.1 Room layout

Our first example is planning the layout of a room. This is a good example for geoPaint, since you'll probably move the furniture around while rearranging one of your rooms. We made this example fit in the geoPaint window to make it easier—you won't have to spend time scrolling. In this way, you can see the whole picture at once. To simplify our example, we used metric measurements since the dimensions are easily divided by 10. Here is a list of the values that we'll use for the room layout:

Object	Length	Width
Room	6.0 m	3.0 m
Bed	2.0 m	1.0 m
Cupboard	2.0 m	0.5 m
Table	1.0 m	1.0 m
Arm chair (2)	0.5 m	0.5 m
Shelf	1.8 m	0.3 m
Desk	1.4 m	0.7 m
Chair	0.4 m	0.4 m
Table lamp	25 cm	25 cm

Let's start drawing the room. Load geoPaint. Let's call the document `My Room`.

Since we want to see the whole room in the geoPaint window, we'll use the scale of 1.0m = 40 pixels. Since just describing how to draw the figure is kind of dry and hard to follow, we documented it with pictures. The first picture shows you what the first several paragraphs describe. This should make it easier to follow the description.

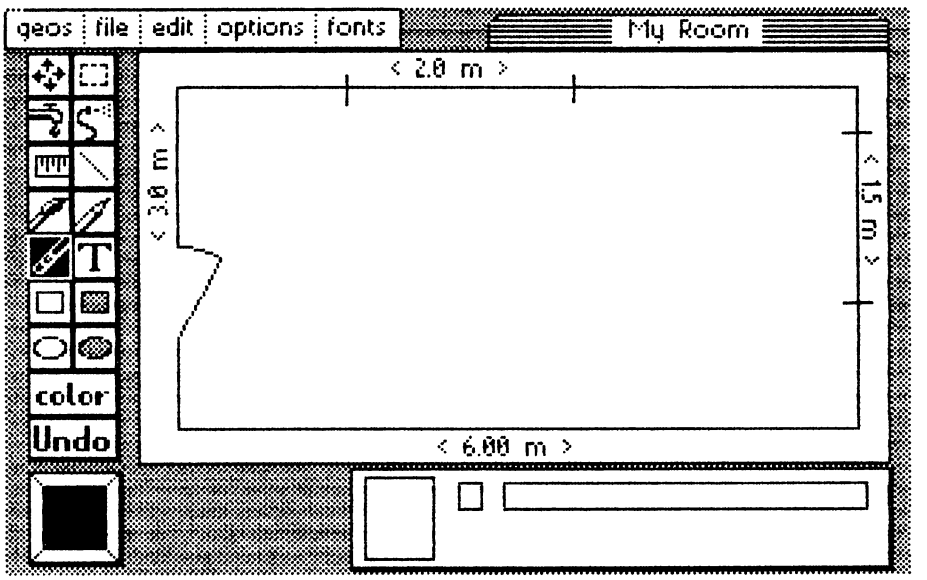


Figure 47: Walls and dimensions

The line tool is perfect for drawing the walls. Draw two horizontal lines 240 pixels in length and two vertical lines 120 pixels in length. Leave some room to write dimensions around the figure. Next mark the window. The top window begins 60 pixels (1.5m) from the left and is 80 pixels wide. To mark its position, draw two lines that are five pixels long. Unfortunately you can't use the ruler to measure the distance. Technically, you can measure the distance, but you can't mark the end point so that you can draw the lines with the pencil tool.

But you can use the trick from the previous chapter. Draw a new line on the wall that's already there. Then you can easily find how far 60 pixels is and then draw a vertical line from that point right away.

The right window is 44 pixels (1.1m) from the top. It is 60 pixels wide. The door starts 32 pixels (0.8 m) from the bottom. Set the line there and draw a line 16 pixels right (0.4 m) and 28 pixels up (0.7 m). To show the arc for the door, use the following trick:

It's difficult to draw a curved line correctly, so click the circle tool and choose the door hinge as the center point. Move the circle until it touches the upper corner of the door. After this, erase most of the circle with the eraser tool, and remove the rest in `pixel edit` mode. Now we have a good curved line.

It is easy to label the upper and lower walls. We used `University 6 point` type. To write on the left to right walls, enter the length in the middle of the room, surround it with with the cut marker, and then use `rotate` to rotate the first label once. Use `rotate` to rotate the other label three times. Then you can `move` the labels to their correct positions. Make the border for the edit box as small as possible so that you won't erase part of the wall when you copy.

Now we can draw the first object and put it in an arbitrary location in the room. We'll start with the bed since it's pretty big and can't fit just anywhere (you do need to open the door!). Draw a rectangle 80 pixels x 40 pixels (2.0m x 1.0m) in the middle of the room. Label it `Bed`. We'll put it next to the window. This example shows how easy it is to move objects around the room. We'll show you the result of this in the following picture before we describe the moving process:

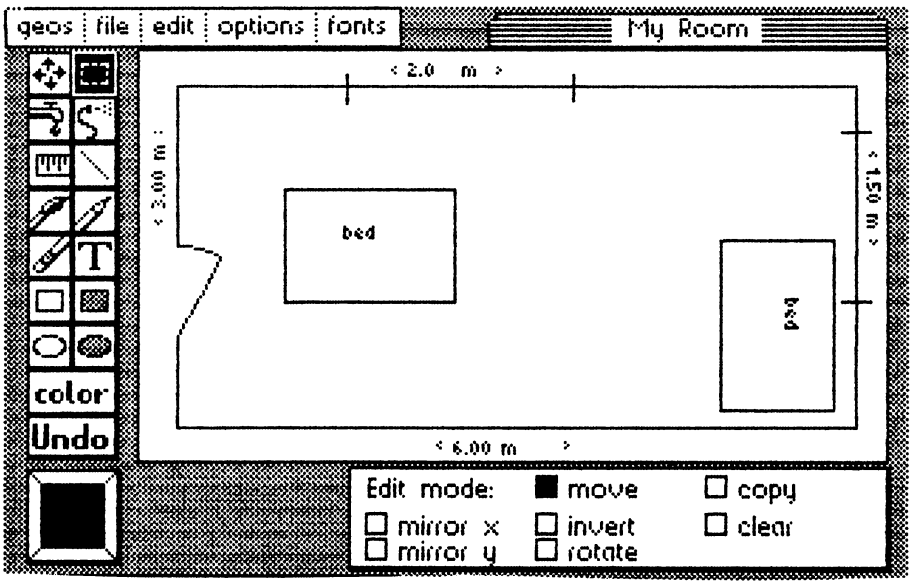


Figure 48: The bed

First rotate the bed 90 degrees. Select the edit box icon and enclose the bed. Then click `rotate` in the status box. Next click `copy` in the status box. Now position the cross-hair on the bed and move it to the window. `copy` and click the lower right corner of the bed. Move the lower right corner of the room. The lower corner of the bed is deposited where the crosshairs is positioned. Using this tip, you can put the object exactly where you want it. Leave a little space between the bed and the right wall so that the window border doesn't disappear.

We didn't `move` the bed. Instead, we used `copy` to put it in the correct position so that we could show the bed rotated and unrotated in our illustration. We got rid of the extra bed in the middle by `clearing` the selected area.

Now we can draw and position the rest of the furniture. Here is the right half of the room once this is completed:

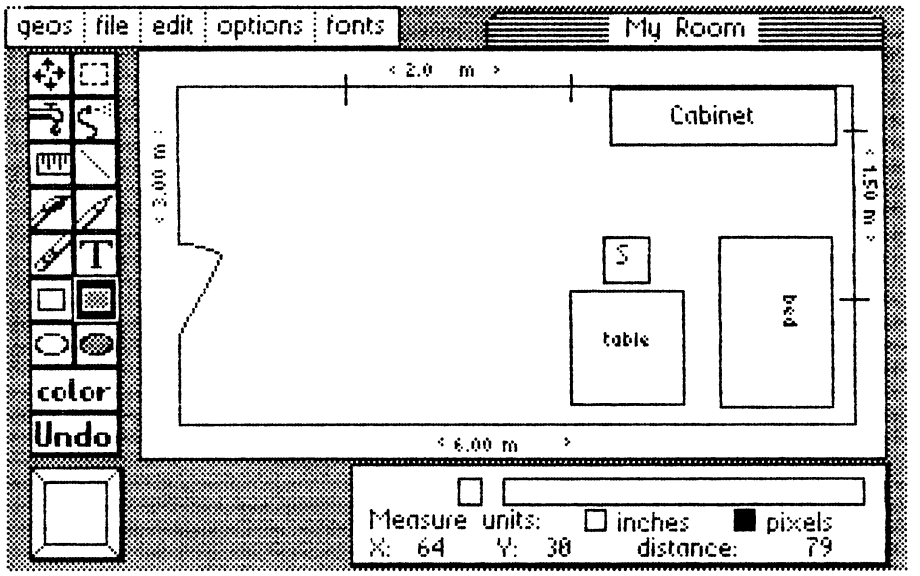


Figure 49: Right half of the bedroom

It is easiest to create the other objects in the same way you made the bed. Draw each object in the center of the rectangle that forms the walls of the room. Add the label using the same font type for the bed. Rotate the object if necessary, using the edit box icon. Once you've rotated the object, move it to the correct location.

Since the seats are so small, we labeled them with `S` in `BSW 9 point` type to make them stand out a little more. Figure 50 shows the finished product:

Here's how the room looks fully furnished:

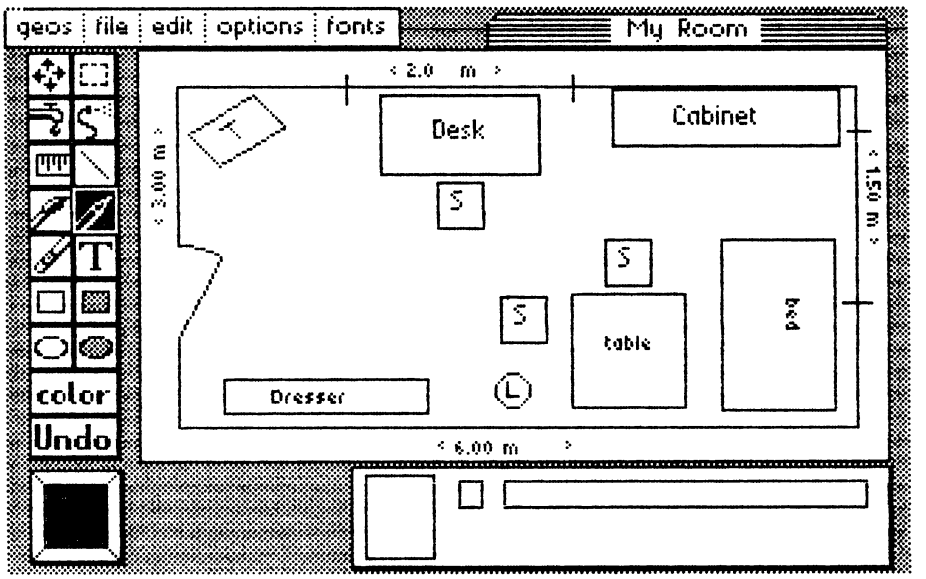


Figure 50: Fully furnished room

We can draw the lamp `L` with the circle tool. We can draw the table `T` in the upper left corner. Since geoPaint rotates items only in 90 degree increments and this table is rotated 45 degrees, we must draw it with the line tool. We labeled it with the `T` in `pixel edit` mode using individual points.

Now we're finished laying out the room. Maybe you'd like to to try moving the furniture around the room. Here's a little tip:

Since the room doesn't have enough free space to move the large objects freely, you can use some **`edit`** functions. For example, to exchange the position of the bed and the desk, enclose the desk and chair with the edit box and use `cut` to put the section in `photo scrap`. Then move the bed to its new place under the window and `rotate` it three times to get it positioned correctly. Then move all the chairs far enough to the left so that you have room for the desk. Position it with `paste` and rotate it to an appropriate place. Then you can move the chairs to the right again. If there's not enough room to rotate, you can move the geoPaint window to display and unused area, put the object there with `paste`, rotate it there, and then use `cut` again to put it in `photo scrap`.

4.2.2 Laying out a garden

We made the drawing fit in the geoPaint window when we did the room layout. However, if we want to lay out an entire house with geoPaint, we will need the whole page. We thought it would be a good idea to go through a different example of how to do this. Try to plan and lay out a garden using geoPaint. In attempting this we discovered:

- you can draw an attractive garden using geoPaint, and
- we're not particularly good at drawing gardens.

While we weren't completely satisfied with the results of our artistic endeavors, we think that using the tricks we have learned, you will meet with greater success.

We designed a 8.0m x 8.6m garden and used a scale of 1.0m = 50 pixels. We named the document `My Garden`.

First draw a border around the garden. The upper line is 430 pixels long. Don't start in the upper left corner. Start about an inch down and to the right of that corner so you'll have room to write labels and title the drawing. We drew a dash every 100 pixels and labeled them so that we could orient ourselves in the drawing more easily.

To draw this line you'll need to move the geoPaint window twice with the scroll box. You can save a lot of time if you move the window as little as possible. When you've finished the line to the right, it's best to draw the 400 pixel long vertical line immediately.

After the border is finished, start with the terrace in the lower right corner. It's 130 pixels wide and 100 pixels long. The three steps on both sides of the terrace are 50, 40, and 30 pixels wide and 10 pixels long. Indent each higher step 5 pixels from where the preceding step starts on both sides. We then can fill the steps with a pattern.

The steps lead to a path that is 20 pixels wide. An oval representing a small pond is in the middle of the area surrounded by the path. Since geoPaint can draw circles, but not ellipses, we had to use a trick to draw it.

First mark the center, and then draw a circle. Draw a vertical line through the center. Then use the edit box to move the left half of the circle about 10 pixels to the left, and the right half 10 pixels to the right. Then connect the tops and bottoms of the two half-circles and erase the vertical line. Now the oval for the pond is finished.

The `preview` option can help you in laying out the garden. We used it a lot. It allows you to see how the size and positioning of the individual items in the garden look.

Here is what our garden looked like:

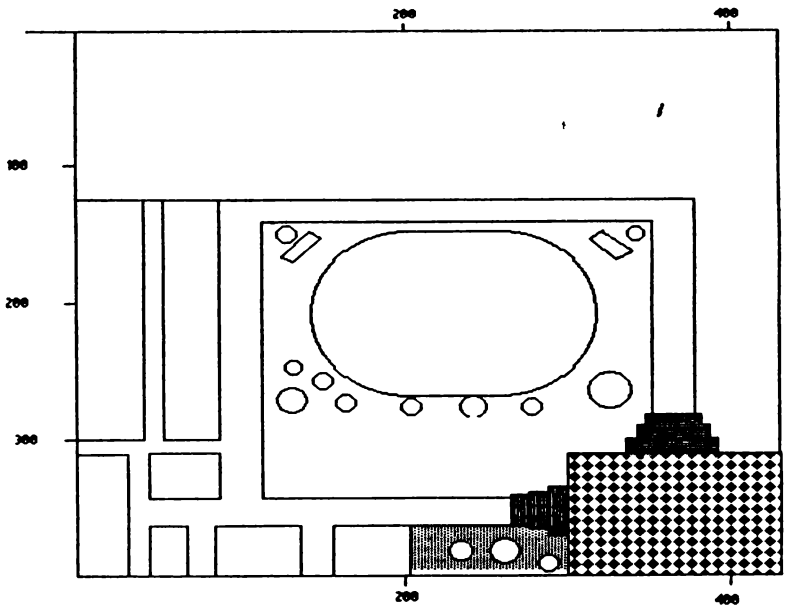


Figure 51: Garden

We're sure you have more landscaping potential than we do...

4.3 Electronic circuits

You need a geoPaint work diskette with at least 35 K of empty space for this section. Format a diskette using the `format` item in the **disk** menu. Copy the Photo Manager, geoPaint, any fonts and printer driver you'll need to this diskette.

Note: Some versions of GEOS may not allow you to copy geoPaint (try copying and running geoPaint on a work diskette). If your geoPaint won't copy, use your applications diskette for the applications in this chapter.

If you looked inside your C-64, you'd see the thin composite board that holds and connects the different components and computer chips. It is actually a large printed circuit board. PC boards are used in the design and production of virtually every electronic device. There are programs available for laying out circuits and designing printed circuit boards with the '64.

We'll show you how to you can perform this intricate work using geoPaint. We were surprised how easy it is to draw electronic circuits and how professional the results looked (in contrast to our work with the garden!). First we'll work with an example that fits in the geoPaint window: a PC board.

4.3.1 PC Board

Load geoPaint and enter `PCBoard` as the name of the document. We'll show the results first so that you can follow the individual steps more easily:

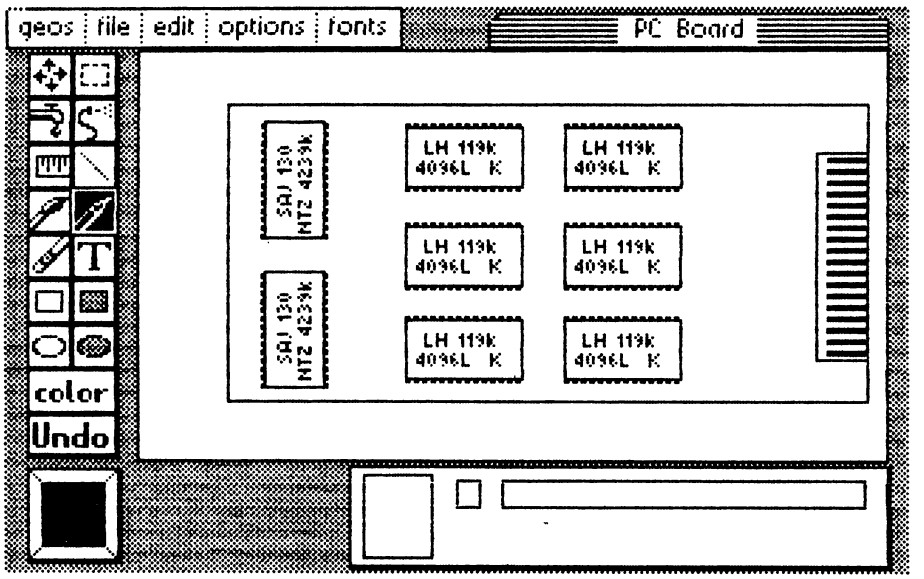


Figure 52: PC Board

First draw the border of the PC board quickly using the line tool.

Next draw a pattern for an integrated circuit (IC) in the middle of the PC board. It's easy to do if you use the line tool in pixel edit mode. The IC is 42 pixels long and 21 pixels wide. This is what an IC looks like under the magnifying glass:

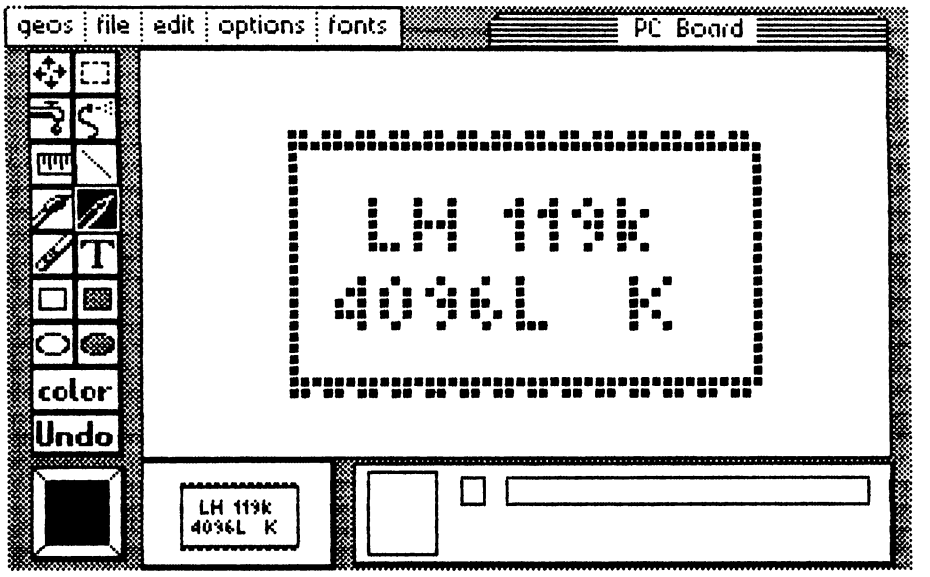


Figure 53: An IC in pixel edit mode

Draw the IC's connection pins now (but not the label, since it varies for different ICs). When you're finished, click the edit box and enclose the IC. Put it in photo scrap using the copy option. Now label the IC in the geoPaint window with SAJ 130 NTZ 4239K. Choose University/ 6 point type from the fonts menu.

Now rotate the IC by 90 degrees, and then move it to the upper left corner of the PC board. Use copy to make a duplicate of it in the lower left corner. You can duplicate the other ICs in the same way. Copy the IC pattern from photo scrap with the paste command. Label it LH 119 K 4096L K. Then move it with move to the empty area in the upper left corner. Position the rest of the ICs using the copy command.

Now you can draw the power connection on the right side of the board. You can easily do this using the pixel edit mode. Now the PC board is finished. If you want to use the IC for future designs, load photo manager and put it in a photo album. In this way, you can gradually develop a collection of components in a photo album. You can then use these components to design other PC boards.

4.3.2 Schematics

Here's another example of how you can use geoPaint. The following is a schematic that is often found in introductory electronic books: a multivibrator with relays.

We won't look at the technical side of this problem (i.e., we're not going to draw a circuit that will really work)—we're just going to show how easy it is to draw a schematic with geoPaint. Most circuits of this type contain just a few types of components (resistors, diodes, transistors, etc.). To design more powerful circuits, create a special `photo album` exclusively for these components. Then just paste the required component in the circuit and connect them with wiring as they're needed.

Another small trick that can save you a lot of work is creating another character set. Almost every technical field has special symbols that it uses. geoPaint doesn't have these special character symbols–we need omega (Ω) and mu (μ) for our circuit.

While you can't alter geoPaint's character set, there is a trick you can use.

After manually drawing a special character, copy it into `photo scrap`, and then put it in the `photo album`. If you need another special character, paste the album's special character page in an area of the screen not being used, draw the new special character in the area pasted in, copy it to the desired area, and paste the expanded special character page back in the `photo album`. In this way, you can quickly build up a set of all the characters you need for these sorts of applications.

Before we start to describe how to draw the circuit, we'll show you the results so you can get an overview of what we're going to do:

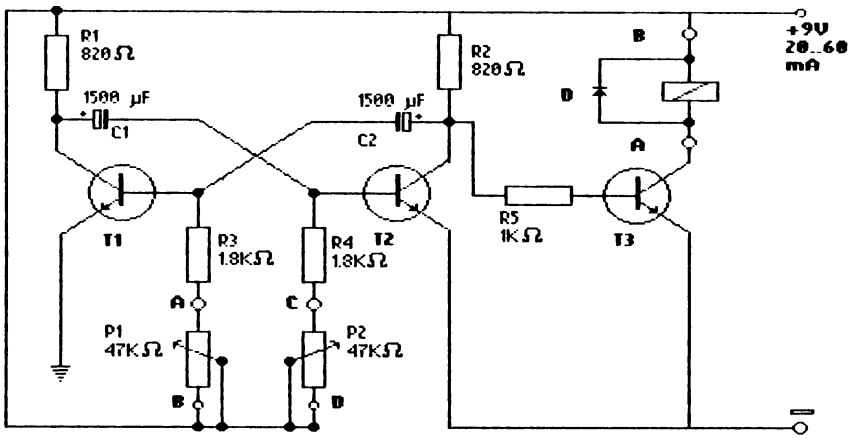


Figure 54: Multivibrator schematic

We'll start in the upper left corner. First draw the part of the power supply wiring (+9V) that fits in the geoPaint window. Then start with the connection line to the resistor `R1`. The resistors are 10 pixels wide and 30 pixels long. We wrote the `R1 820` label with `BSW 9 point` type. The character "Ω" for ohm is the first special character that we'll need. Draw it using `pixel edit`. Here's how it looks in that mode:

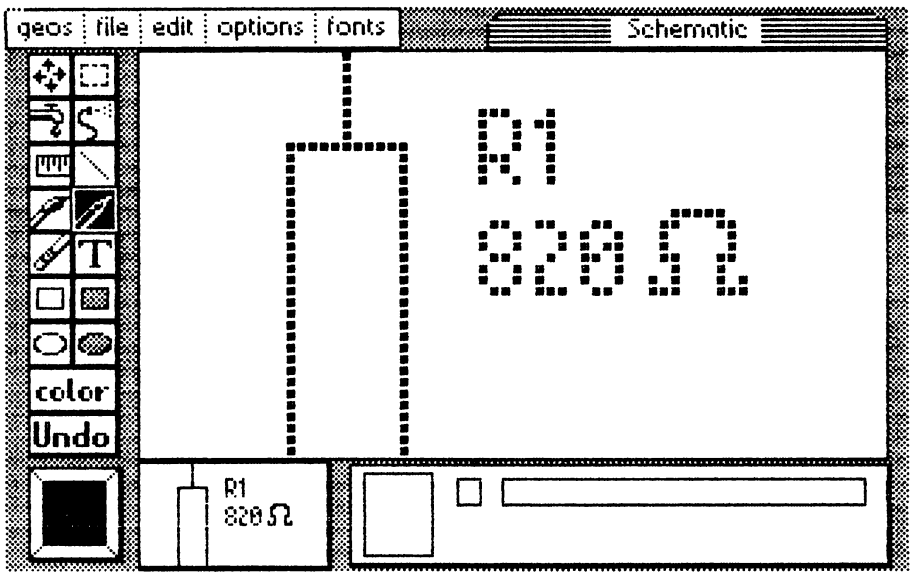


Figure 55: Ohm symbol (Omega)

Leave `pixel edit` mode. Enclose it with the edit box and `copy` it into `photo scrap`, since we'll need it several times. Now we can draw the top connection. We can easily draw it with the filled circle tool and the filled surface as the pattern in the pattern table:

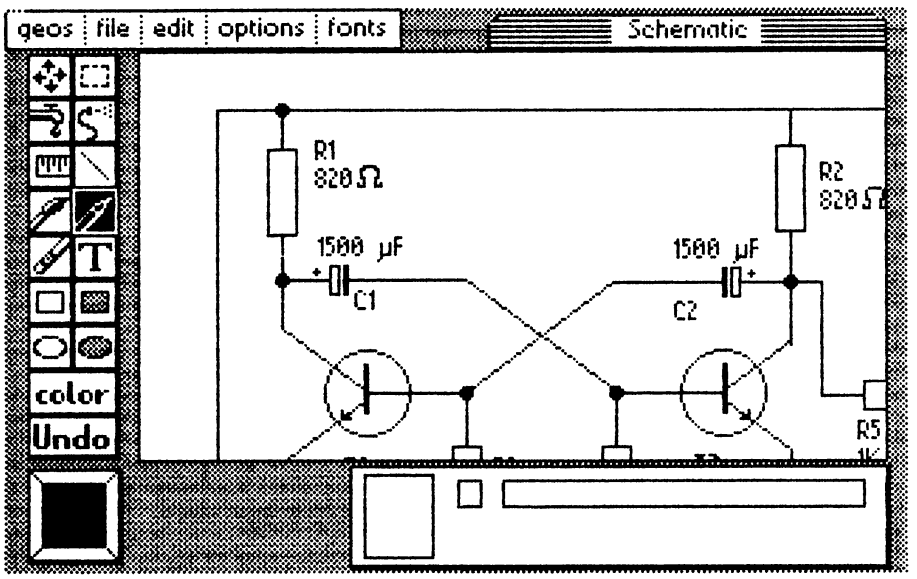


Figure 56: Starting the schematic for the multivibrator

We can draw the capacitor in `pixel edit` mode and also paste it in the `photo album`. We can also draw the special character mu (µ) this way.

We saved a little work by entering `1500 µF` in `BSW 9 point`. We just had to add a couple points to the µ to finish the special character:

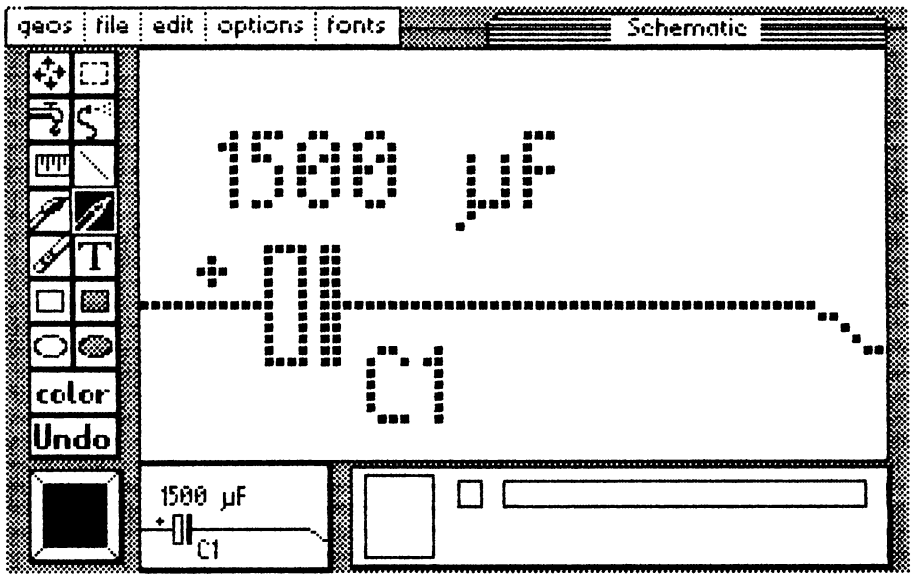


Figure 57: Mµ

The transistor `T1` is placed beneath the capacitor. Move the geoPaint window with the scroll box as far down as you can and still see the capacitor. Then you can draw the wiring between the components. Here's a printout of this section:

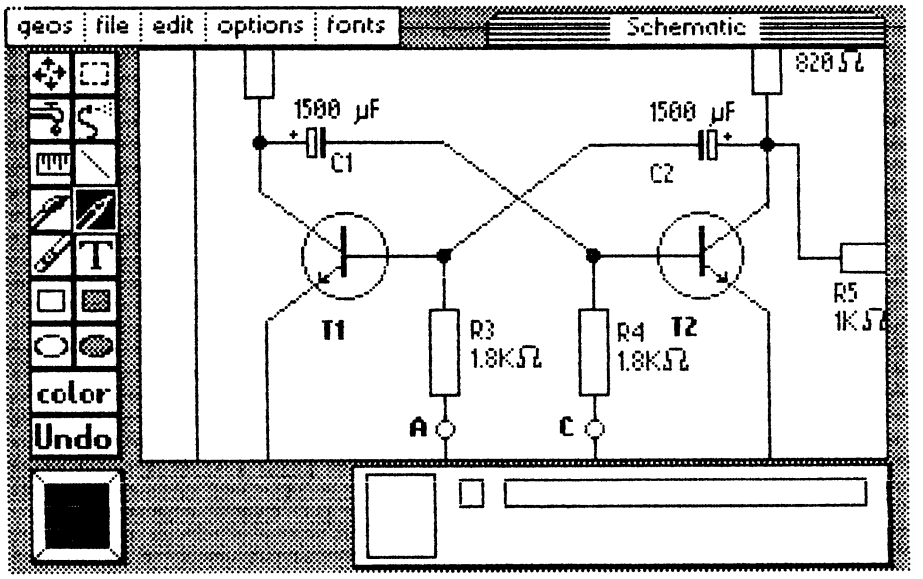


Figure 58: Capacitor and transistor

Draw a straight line from the capacitor to the transistor that goes as far right as the `T1 - R3` connection point. This saves us a lot of work later.

Draw the circle for the transistor symbol with the circle tool. It's easiest to use `pixel edit` to draw the lines for the emitter, base, and collector of the transistor. Then draw the power supply in the far left section of the screen. Remember, you'll save time if you use scroll box as seldom as possible.

Next draw resistor `R3`. If you copy it to the `photo album`, you can paste it in this document and move it to its proper location. To do this, load the `photo manager`. You can paste the special characters Ω and Mµ in the album now also, if you haven't already done so. The potentiometer `P1` located below `R1` is easy to make. Copy `R1` (which already has the small circle below) to the area below it. When you need to place something in the document, check to see if you can copy part of it from another area (using mirrors or rotations). For this circuit we drew only a few components. The rest we drew by copying sections of the page.

Finally, complete the wiring to transistor `T1` (that's the line with the small dash) and the two connections from `P1` to the line below.

You may want to draw transistor `T2` first so that you can find where the second group of components should go. Use the scroll box to move up until `T1` and `C1` are in the geoPaint window. Now enclose both with the edit box:

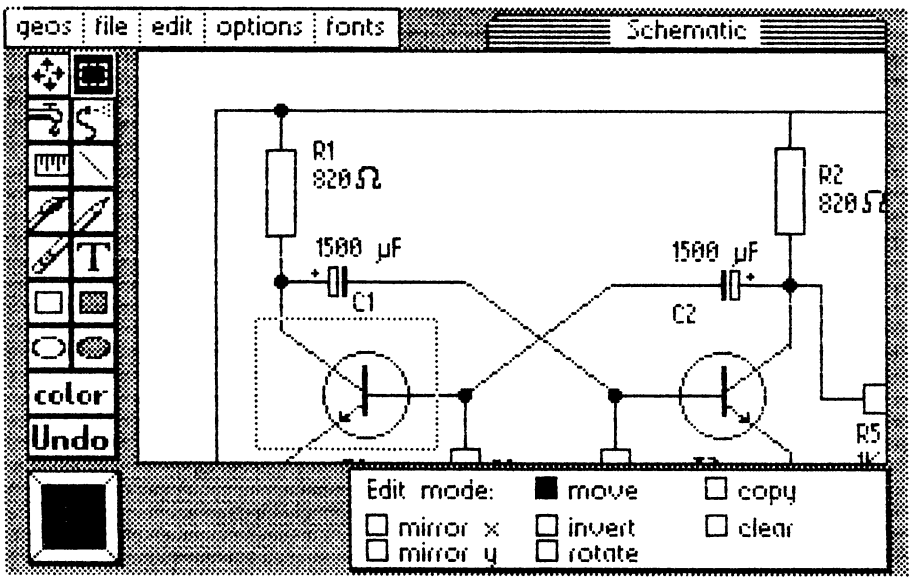


Figure 59: Copying transistors

`move` the enclosed area as far as possible to right. Then reflect it about the Y-axis with `mirror X`, and you've finished the second transistor with capacitor. If necessary, move the geoPaint window a little to the right and then `move` the transistor too. You'll need to mark it again after using the scroll box. Mark the connection point between the capacitor and the wiring with a black connection point, drawn using the circle tool.

Next we'll make resistor `R2` and capacitor `C2`. Copy `R1` (without the description) in such a way that it's located relative to transistor `T2`. Copy the description on the resistor's left side later.

You won't need to draw resistor R4 and potentiometer P2. Enclose R3 and P1 with the edit box. Then copy the section to the right and reflect it using the mirror X command. Then move it so that the connections from R4 and T2 fit. Now just erase the description and type a new one (you can copy the ohm character (Ω) from R3). Now the left half of the circuit is finished:

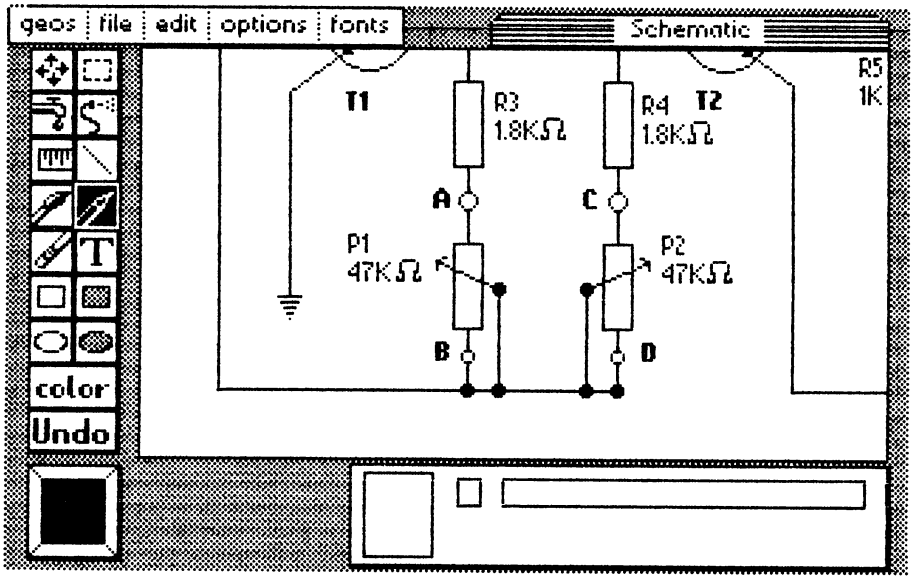


Figure 60: Mirroring potentiometers

You can create R5 by rotating another resistor 90 degrees. Transistor T3 is made by copying T2 and then changing the title. The component group above T3 consists of a diode (D) and a relay. We created the diode in pixel edit mode. It is located 5 points from the corner.

Then we drew the rest of the wiring and labeled the power supply. The dash - and the text +9V 20..60 µA are written in BSW 9 point outline type. We clicked bold for the title Multivibrator with relay.

Here's the way the reduced picture looks using `preview`:

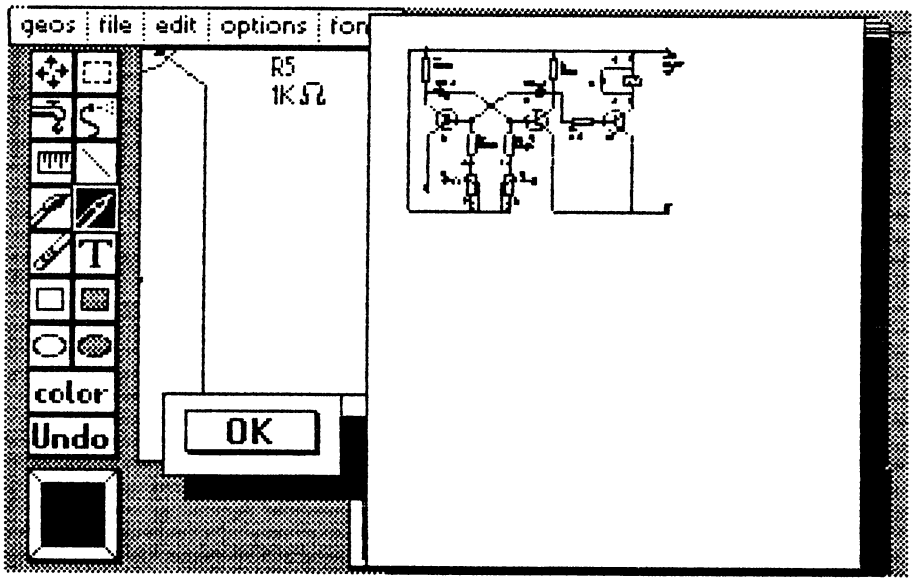


Figure 61: `preview` of the schematic

Looking at the descriptions, it is easy to recognize the components. The `preview` option is very good for orienting yourself to the picture and seeing if your layout is correct, especially when you create documents that are larger than the geoPaint window.

We hope that you paste so many components and special characters in the component album that you'll soon have a complete collection of finished components.

4.4 GEOS and educational applications

Teachers and students alike are constantly required to produce illustrations, drawings, forms, tables, and other kinds of documents.

We'd like to give you a couple of examples of how GEOS can help teachers and students save time and produce higher-quality documents. We want to look at both the advantages of GEOS as a graphic tool—such as its easy editing, and its diverse range of labeling possibilities.

We'll also look at its disadvantages. While working with geoPaint, we tried to produce some figures that are just about impossible with this program. Freehand drawings are very difficult to do with geoPaint. It's best if the documents you create are almost entirely composed of geometric objects that geoPaint offers (i.e. circles, rectangles, and lines). For example, you can make ellipses by drawing a circle and then moveing the two halves of the circle away from each other.

You'll learn to recognize what can and cannot be quickly and easily drawn with the geoPaint program. Now let's look at a few educational applications.

4.4.1 Diagrams in reports

Some illustrations can't be drawn with other '64 graphics programs, because the labeling techniques needed just aren't available. But these labels can be easily created with geoPaint. This first example is a diagram for your History 101 class: the economic structure of Russia in 1913. We'll just show you the final result–since our descriptions earlier in this chapter should suffice–and let you work on your own.

It didn't take us much time to draw it, and the diagram can be used for various projects. You could use it in a book report one year, and then as part of a class paper the next semester (as long as you get different teachers...). We like to think of this as "scholastic multi-tasking".

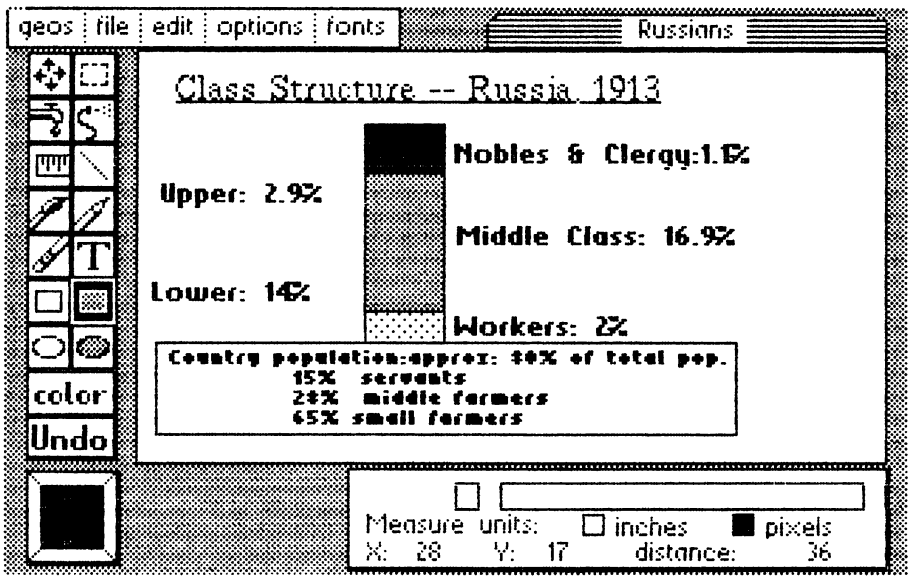


Figure 62: Russian economic structure–1913

4.4.2 Teacher's forms

Every teacher needs to design forms like course outlines, project sheets and assignments to explain and organize material. If you do this with paper and pencil, you quickly end up with a full trash can, because you're bound to divide the page up incorrectly, or discover that one element doesn't fit in the space provided, etc. Also, if you've been teaching any length of time, you've probably already got a stack of old documents for every occasion. All you need to do is copy the format of a few good examples, then fill in new information as needed.

geoPaint is terrific for this kind of application, since it has almost everything you need—circles, lines, rectangles, and many different text styles, fonts, and point sizes. You can even use your forms on an overhead projector. You just print out the document, and then copy it onto a clear sheet of acetate—*voila!*

4.4.3 Seating charts

You might need to make seating charts at the beginning of the year or semester. Not only can you do this using geoPaint, it also offers the following advantages:

- Since seating charts change only when new students arrive or you decide to alter the arrangement of the desks, you simply modify the existing seating chart that you've stored on diskette. You don't need to redraw it from scratch.

- Usually the number and placement of desks is almost identical from room to room. You can make a master that other teachers can use for their classrooms. Then they just need to enter the students' names.

4.4.4 Science class diagrams

If you're a science teacher, your students must often perform experiments and usually need a method to chart the data. It would be nice to be able to produce diagrams quickly, accurately and easily. It's easy to produce scientific charts with the geoPaint program. You can draw a format page with a rectangle divided into sections and have the students plot and label as required.

We have one tip. You may find that you want to make minor modifications to an existing drawing to save some work. But be careful—don't put both the original and the copy on the same diskette. Because they will be so similar, you could lose the original.

Chapter Five

GEOS Tricks and Tips

GEOS Tips and Tricks

Each time we used GEOS, we were always amazed by its capabilities, as well as by the flexibility of its programming. In this chapter we're going to present you with some of these possibilities, and reduce the time you'll need to take to get up to "full speed" with GEOS.

Because this book was written in Germany and will be sold internationally, we've included a section on adjusting the GEOS operating system to operate at the European electrical standard of 50 Hz (the American standard is 60 Hz). Because of this modification, certain programs may not be compatible. However, every program in this book or on the optional diskette is compatible with GEOS both in 60 Hz operation and modified for 50 Hz operation.

5.1 Tips and Tricks for the GEOS user interface

This first section gives you the benefits of our experience with the user interface and the different programs that compose it. In the second section we will expand and improve upon what we discussed in the first section. We believe that this will make GEOS both easier to use and more powerful.

5.1.1 Problems loading GEOS

You'll probably have problems using GEOS before you use the joystick or try to start your first application. This first section deals with the problems you might have loading GEOS.

There are two reasons why GEOS has trouble booting. First, GEOS has a very sophisticated copy protection scheme. If the distribution diskette is copied, the new version will be unable to run because GEOS will not find some important copy protection information. However, occasionally GEOS can't find this information even if you are using the original copy of the program.

There are also problems associated with the fact that GEOS uses fast disk access routines. For one, if you have your printer powered up while you boot GEOS, it may or may not load.

If GEOS does not boot properly, there is no reason to worry. Just restart or reset the computer after an unsuccessful load.

If the `BOOTING GEOS` message disappears from the screen before GEOS is finished loading, turn the computer and disk drive off and then back on again.

If nothing happens, even after a few tries, there is still hope. Read the following troubleshooting list and answer accordingly:

- Did you try to boot with the original diskette? This is the only diskette that GEOS can boot.
- Do you have only one disk drive (with device address 8) turned on? Normally the disk drive has the device number 8.
- Did you turn your printer power off? Sometimes GEOS does not like having a printer on during booting. GEOS will display the printer and waste basket icons, but will then wait until the printer is turned off.

If all these points have been addressed, but you are still having problems booting GEOS, there is still a trick that will take care of most problems.

GEOS is very finicky with disk drives that are poorly adjusted. It's possible that GEOS won't find the copy protection if the read/write head is out of alignment. There are two ways to take care of this problem.

Note: If your disk drive has mechanical problems, neither of the following procedures will work—you will have to take your disk drive to a dealer for readjustment.

Don't use either of these techniques with disk drives that are very warm, because the procedures can cause the mechanical operation of the read/write head to malfunction. If your drive is running very warm, we suggest that you turn it off for 15 minutes or so and read a few more pages in this book.

1) Get a blank diskette, or a diskette containing material you don't need any longer. Format this diskette. This will automatically realign the read/write head (as long as there are no mechanical problems).

   If you don't know how to format a diskette, type the following on the screen in the C-64 direct mode and then press <RETURN>. You can replace NAME with any name up to 15 characters, and ID with any two-character or two-digit combination:

```
OPEN 3,8,15,"N:NAME,ID":CLOSE3
```

   After you have done this, you will hear a whirring sound. After about one and a half minutes the status light on the disk drive will go out. Now try booting the GEOS diskette again.

2) If you don't have any spare diskettes or you are looking for a simpler method, type in the following program:

```
10 REM This program moves the disk head
20 REM into postion
30 OPEN 3,8,15,"I": REM Open disk channel
40 REM Initialize
50 PRINT#3,"M-W"CHR$(0)CHR$(0)CHR$(192)
60 CLOSE3
```

   The REM lines are there for explanation only; you may remove them if you wish. Start the program by typing RUN, you will then hear the whirring. Afterwards, the read/write head is realigned and ready to use.

If none of these work, try another original diskette before running out to your dealer for disk drive repair. Try to borrow a copy of GEOS from a friend, or ask your dealer—this shouldn't be any problem. If it turns out that there is something wrong with the original diskette, ask your Commodore dealer to find out what to do next.

Just remember–do not format or use diskettes in a very warm disk drive.

5.1.2 deskTop "Do's and Don'ts"

When you're working with GEOS, always follow these guidelines:

Never change the diskettes without using `open` to open the new diskette for access. That means that you can't change diskettes unless you are working in deskTop, since the `open` command is available only from the **`file`** menu. There are safeguards in GEOS to protect you from quitting a program without the diskette containing the active program being in the drive.

Even with these safeguards, removing a disk containing geoPaint and using the scroll box to move while the geoPaint window is still active will produce some interesting screen effects—not to mention a system crash.

Never open the drive door when you save or load GEOS. This also means you should never take the diskette out of the drive when saving or loading GEOS.

Never turn off the disk drive power when GEOS is running. The `SWAP` file GEOS uses to store disk input/output will be lost, and you may not be able to recover this data.

Always use different names for each diskette. GEOS cannot distinguish between different diskettes with the same name. Using identical disk names makes copying and saving files very difficult. Therefore, always use different diskette names and ID codes.

Use the original GEOS diskettes for booting only. Once you are finished booting GEOS using the system diskette or backup system diskette, put it in a safe place. If you happen to lose one of the first four programs necessary for loading GEOS, you have only one option for restoring the original diskette:

1) If you are still in GEOS and you notice that one of the programs is missing, you can copy the missing files from your backup system copy of GEOS back to the system diskette, or the other way around. You should then be able to boot up using the original diskette.

Use your GEOS diskettes only with GEOS. Don't use your GEOS diskettes for storing normal BASIC programs, unless these programs are going to be converted to GEOS format.

Never use the normal Commodore DOS commands VALIDATE and SCRATCH on GEOS diskettes. If you do this accidentally, load GEOS immediately and use the GEOS `validate` command under `disk`. Hopefully, GEOS can correct what was done to the diskette.

5.1.1.1 GEOS file management and printing

deskTop allows many different file management options. Take advantage of its flexibility.

One of those options is the order in which the files are stored on the diskette. Let's say that you have a diskette that you usually use for storing geoPaint documents. You can move the documents to the first page of the directory. Now you can double-click this page and go directly to your document, rather than having to search through every one of the pages of a file.

Second, use the `info` box. This allows you to maintain a very ordered and efficient file system:

a) Load the correct time and date with the Preference Manager immediately after booting GEOS. This is useful when you store different versions of a graphic or text document. When you list the files, you can use the date and time to determine which file is the most current, and which is the original.

b) Using the `help` screen, you can store important information about each program. This information could include:

- Information about what the program actually does
- The version of the program
- The starting address for this program (for programs written in machine language)

For instance, you can store your video games' `POKE` address where you can increase the number of "lives" you get for a game. Or you can store answers to riddles to spark your

memory when you play adventure games. Before you can do this, the game must be changed to the GEOS format, so you can access the `info` screen. Use the `FILEMASTER` program discussed later in this chapter.

GEOS lets you print graphic or text documents directly from the deskTop or from an applications program. You can install the printer file only from deskTop. There were many times when we wanted to print a graphic document from an application—only to realize that we never initialized the printer from the deskTop. Therefore, you have to leave the application, return to deskTop, initialize the printer and then return to the application.

We've learned from experience that it is better to print graphic and text documents while running the respective application. At least we found this to be the case with our printer (an FX-85 with interface). It is impossible to return to deskTop if the printer is still turned on. GEOS will still display the printer and waste basket icons—but GEOS will wait until the printer is turned off to continue. This also occurs after using deskTop to print. We decided not to print from deskTop unless it was absolutely necessary.

During printing, a window displaying `Printing...CANCEL` appears. If you click `CANCEL`, you can halt the printing. The joystick cursor appears over the `CANCEL` button. However, you might not be able to see it for a short while. We suggest that you hold the fire button down until the window disappears. GEOS registers this after a few seconds.

To print a file from deskTop, make sure that the desired document is on the same diskette as the application. For instance, to print a letter, the letter must be on the same diskette as geoWrite.

One last comment about printing. Whenever printing from an application, GEOS displays the error message:

```
PLEASE INSERT A DISK CONTAINING THE DESKTOP
```

...even though deskTop is already on the diskette in the drive. After the printer is finished, click `OK` to return to normal.

Occasionally GEOS will not want to print text or graphics. It will only print blank lines. If this happens, we can offer a tip (not a fix) for the problem:

Copy the document to another GEOS diskette. You can now try to print the file using this deskTop, or just use `BACKUP` to make a new working copy.

If you have a lot of files to copy, GEOS has a feature to make life a little easier: `duplicate`. First enter the filename patterns, and then use `duplicate` to make as many copies as you want.

You will have to give a different name for each copy you make. You can simply change a few characters for each name, or add a number to the end of the name to show the version or copy number. If you make three copies of a file (let's say for Meyer, Miller, and Tyrone) you could name the files `MEY`, `MIL` and `TYR`.

5.1.3 Tips and Tricks for geoPaint

geoPaint is a great program for creating graphic documents. The more you use this program, the more ways you'll find to make the job easier, and to create special effects. In this section we're going to look at some of the interesting points of geoPaint.

Graphic formats and scaling

The area with which geoPaint can work is actually larger than what you can see in the geoPaint window. From left to right there are 640 pixels, and from top to bottom there are 720 pixels. Altogether, this is about 460,800 individual points. In comparison, the C-64 without GEOS had only 320*200=64,000 pixels. How does this look on paper with the larger window size? This is very important when trying to visualize the final product.

The size of the image is dependent upon the printer. Here we present the results for the FX-85, which is the main printer we used with GEOS.

If you have a different printer, you can draw a line 200 pixels from left to right, and another line 200 pixels from top to bottom. Now print this to your printer and measure it on the paper. The FX-85 can draw a maximum line width of 24 cm. This corresponds to 530 pixels in geoPaint. If you use the pixels from 530 to 640, these will not be printed on the paper. Since there is no easy way to determine where to stop drawing, it's easier if you draw a boundary line.

To print the whole screen you will need an EPSON FX-185 or compatible printer.

The entire picture can be printed on a standard 8.5" x 11" sheet of paper in the vertical direction. The 720 pixels are printed in 25.4 cm. That means 100 pixels are in 3.5 cm and 1 cm has 28.4 pixels. All of these dimensions are in the following table:

Size and Dimensions

Horizontal max:	640 pixels	= 29 cm	= 10.6
Printing width:	530 pixels	= 24 cm	= 8.8 inches
	100 pixels	= 4.2 cm	= 1.66 inches
	1 cm	= 24 pixels	
	1 inch	= 60.1 pixels	
Vertical max:	720 pixels	= 25.4 cm	= 10 inches
	100 pixels	= 3.5 cm	= 1.39 inches
	1 cm	= 28.3 pixels	
	1 inch	= 72 pixels	

The difference of 4.2 cm (horizontal) and 3.5 cm (vertical) for 100 pixels means that squares on the screen look like rectangles, and circles look like ellipses when printed on paper. If you use a different printer, this may change.

Double-clicking tools

You can activate every tool in the toolbox by simply clicking the icon. There are some tools that have an additional function if you double-click the icon. For example, if you are using the pencil and you want to use the double-click feature of the eraser–erase the entire graphic visible in the window–all you have to do is click twice on the eraser icon. Here is a list of the tools that also have a second function:

Edit box: 
If this item is chosen, the entire geoPaint window is enclosed. However, if you only want part of the window to be saved in a photo scrap, double-click the edit box icon and then choose the copy or cut items under the **edit** menu.

Brush: 
Double-clicking this icon is the same as choosing the change brush item under the **options** menu. You can then choose from the 32 different brushes.

Pencil: 
Double-clicking toggles between normal edit and pixel edit.

Eraser: 
Double-clicking this icon erases the entire picture in the geoPaint window. If you are using pixel edit then only the visible portion is cleared.

Brushes and Patterns

You can do more with the fill patterns than just use them with the faucet or airbrush. You can use the brush tool to actually "paint" the patterns directly onto the screen. When you first run geoPaint, the pattern is a simple color. However, after you choose a different pattern, the paint brush uses this pattern until you switch again. You can create different special effects with different patterns and brushes. Try different widths of paint brushes with different patterns; you'll be surprised what you can do.

Patterns are used wherever you are working. Try using the airbrush with a different pattern and see what happens. You can use different patterns inside of others, as well as, using text. Here are the different combinations in the following hardcopy:

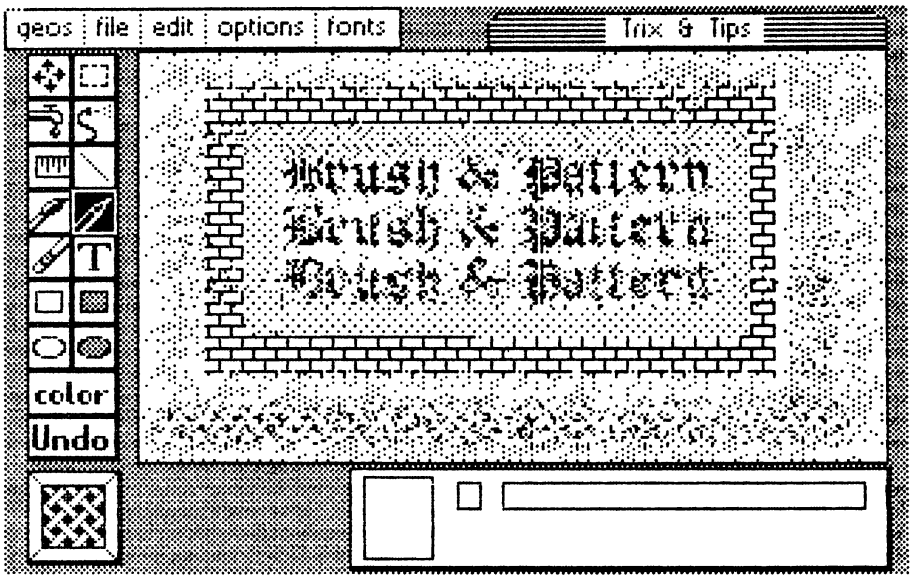


Figure 63: Using patterns

Dotted Lines

Perhaps you've seen technical graphs using different dotted lines to distinguish between different events or topics. You can use a few tricks to make graphs that use dotted lines.

The quickest and easiest way to get dotted lines is to draw a regular solid line and then use the spray can with the empty pattern (upper left of the pattern window) to spray empty spots over the solid line. This will erase parts of the line, resulting in a dotted line. While this is not the best method, it's easy to use and will meet the needs of a lot of applications.

You can make lines with a regular pattern by deciding which individual pixels should be turned on and which should be turned off. Using different patterns and pixel arrangements you can create many different patterns. We have tried different combinations of pixels, which we have listed below. If your application requires a regular pattern to your lines then these might be useful for making your graphs.

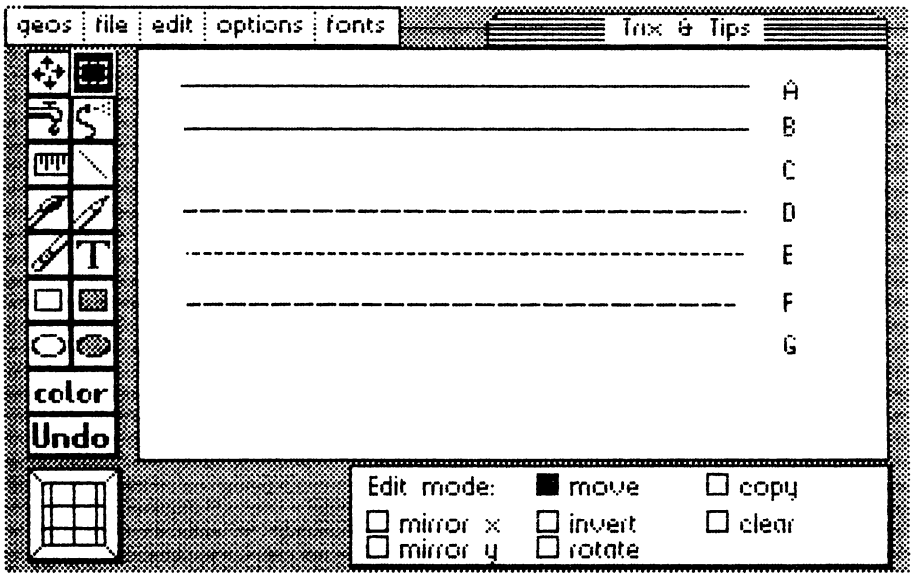


Figure 64: Dotted lines 101

a) Normal straight line
b) Same line with airbrush effect
c) Brush 2.5 (2nd row, 5th from left) and pattern 1.3 (row 1, 3rd from left).
d) Brush 2.5 and pattern 1.5
e) Brush 2.5 and pattern 1.7
f) Brush 2.5 and pattern 1.14
g) Brush 2.5 and pattern 2.1

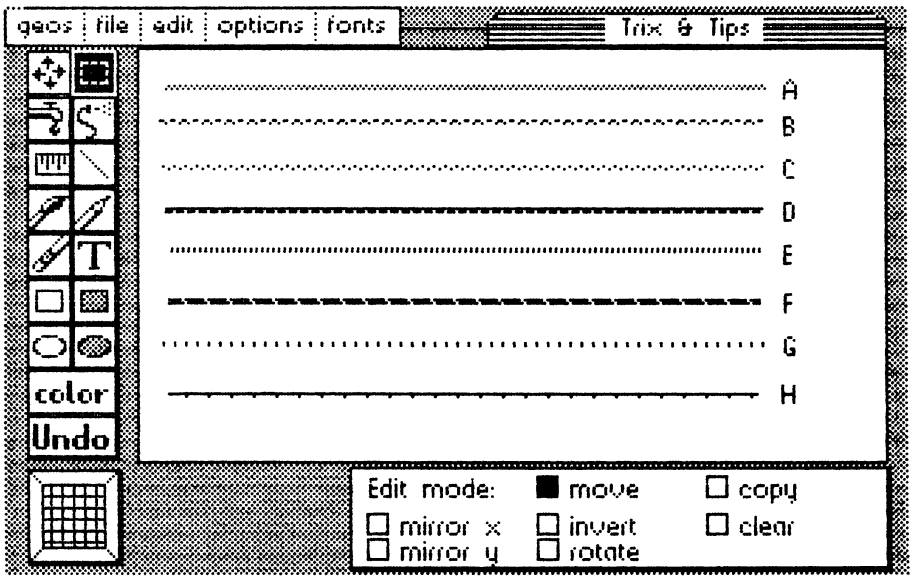


Figure 65: Dotted lines 102

a) Brush 2.7 and pattern 1.3
b) Brush 2.7 and pattern 1.4
c) Brush 2.7 and pattern 1.64
d) Brush 2.7 and pattern 1.9
e) Brush 2.7 and pattern 1.11
f) Brush 2.7 and pattern 1.12
g) Brush 2.7 and pattern 1.16
h) Brush 2.7 and pattern 2.1

What we're trying to show is that there are many different patterns available—experiment and find the best pattern for each application. Not every pattern looks good in every application. An example should show what we mean.

Use brush 2.5 (smaller brush width) and pattern 2.2 (brick wall). The brush paints only a small portion of the pattern. The pattern painted depends on where in the wall you start painting. If you happen to start right on a line between bricks, you could end up with a solid line, or if you start in the middle of a row of bricks you could end up with a very sparse dotted line. We have drawn a document that shows what we are talking about. On the left we painted with a very wide brush using two different patterns.

Adjacent to each wide pattern area is the same pattern painted using a very thin brush, each starting in a different spot in the pattern:

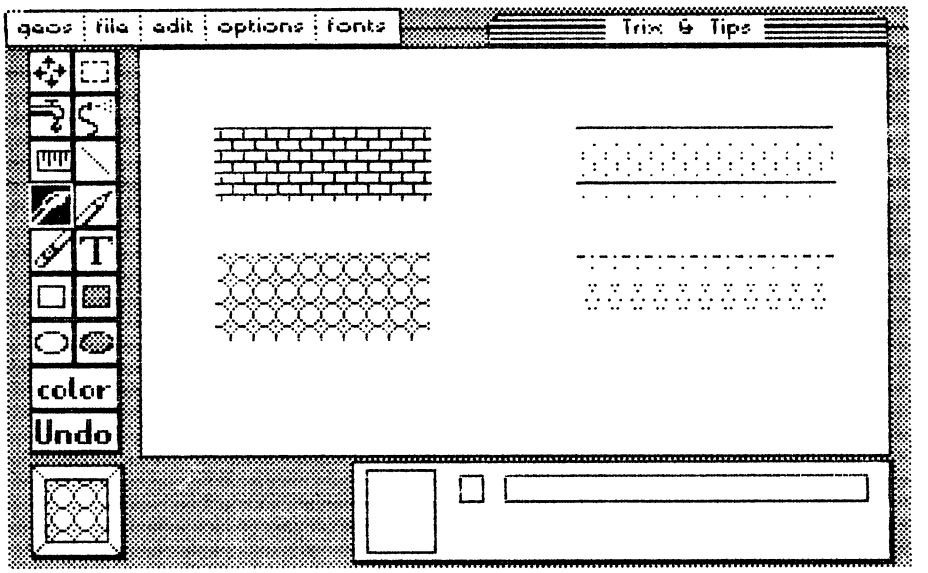


Figure 66: Dotted lines 103 (mixed with patterns)

5.1.4 Tips and Tricks for geoWrite

geoWrite is a WYSIWYG (what-you-see-is-what-you-get) word processor. This means that what you see on the screen will appear when the document is printed. If you underline a word, then it will appear underlined on the screen. This is much nicer than putting a control character before and after the portion you want underlined.

Using this approach, geoWrite has a few limitations. For example, there is a short delay from the time you enter the text until it appears on the screen. But by using a few tricks you can get around many of these inconveniences.

Enter some text using `BSW 9 point` or another font of a similar size. Let's say you later want to print out this text in `Dwinelle` script. It's a good idea to enter the text in the same `BSW 9 point`. That's because you would only be able to see a few characters in `Dwinelle` script in the

geoWrite window, since it's so large. After you have entered and corrected all of the text, then choose the format and style of your text. We will demonstrate what we mean. First, we wrote a letter in `BSW 9 point` script. The lines that appear in the geoWrite window are too wide in `Dwinelle` to fit on a single sheet of standard paper:

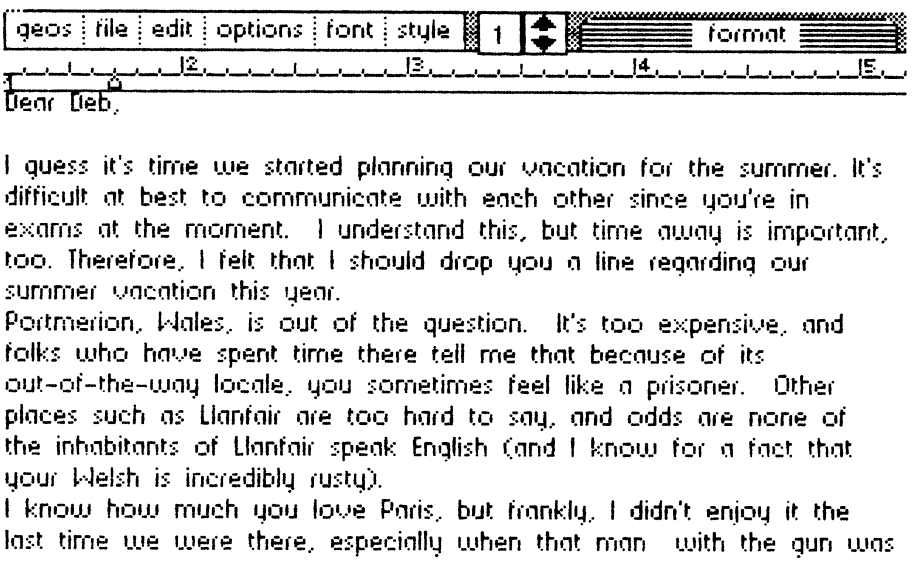


Figure 67: Unformatted letter

It's easy to enter the text—however, you can't see the right margin. In this case, you must use the joystick to set the right margin of the paper. It will take two or three seconds before the screen shows this change. An easier method is to set the left margin to 1 and the right margin to 5. You'll find these numbers on the position bar. The edges are distinguished by an `M` on the left edge; the right side is barely visible. Click the fire button, then point the pointer to the spot you want and click again. This makes the letter easier to read, as shown in the following picture:

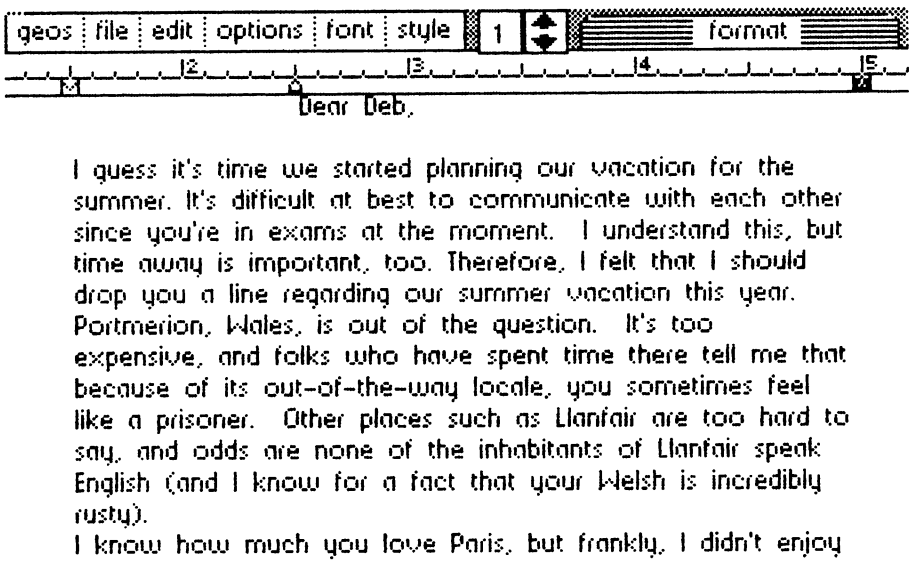


Figure 68: Left and right margins

By setting the margins, the letters will not only be easier to read, but you will also save a lot of time.

When you have completely entered the text, you can choose the desired font. First, you will have to mark the block of text. To do this click the fire button on the first character of the block, then—keeping the fire button key or fire button pressed—move the joystick to the last character in the block. When you release the button, the block is marked, and appears in reverse video. Now, choose the font from the `font` menu, and choose the type style with `style`.

This brings up a problem. You can mark only text that you can see in the marked window. Only the characters in the marked block of text are changed. The text above and below the window cannot be changed. This means that you have to move the window, but to do that you have to unmark the first block.

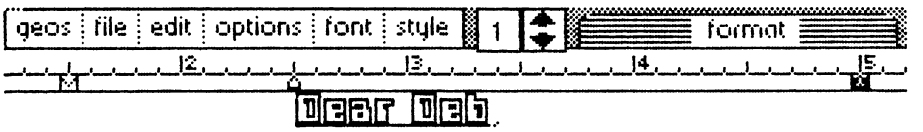
The only possibility is to mark and change the text in sections. This is relatively easy to do (as long as you go from a larger font to a smaller font, the way we did in our example). If you change from a larger point size to a smaller point size, you are actually making more room on the single sheet of

paper. This pushes the extra text in the window down below the current geoWrite window. You'll then have to mark this text, which in turn pushes the other text down, and so forth until the entire file is reformatted. geoWrite automatically moves text to the next page. You can see this if you choose the `next page` option.

Changing the text a portion at a time is slow and annoying. This is because the geoWrite window can display only a few words in, for example, the `Dwinelle` font. If you were to mark the block using `BSW 9 point`, the text would only take up two or three lines on the screen. The rest of the text, written in `Dwinelle`, stays below the window and out of sight. To get it from below the window to where it should be takes a lot of work. Therefore, it is better to start using a smaller point size and font and moving to a larger point size and font if required.

If you have a large document, you can save yourself a lot of time and work during this reformatting stage. All you need to do is set the the left and right margins as close to the edges as possible. This allows geoWrite to change the maximum amount of text at one time.

Next, you can alter the text to stand out from the rest of the document. To do this, simply mark the word or phrase and choose the appropriate style. For example:



I guess it's time we started planning our vacation for the summer. It's difficult at best to communicate with each other since you're in exams at the moment. I understand this, but **time away** is important, too. Therefore, I felt that I should drop you a line regarding our summer vacation this year.

Portmeirion, Wales, is out of the question. It's too expensive, and folks who have spent time there tell me that because of its out-of-the-way locale,

Figure 69: Highlighting words

Before you print the document reset the margins for the desired page width.

You have a much greater degree of control over the design of your document using geoWrite, simply because what is on the screen is what is printed onto paper—true WYSIWYG display. If you want a few lines to start a few more columns to the right, simply enter a few spaces before typing to move the line over a little. This brings up two new problems:

The blank spaces may mean that the characters may not line up exactly. Each blank character moves the cursor to the right a certain number of pixels. The number of pixels is determined by the current font. Therefore, you can only move the cursor in a set number of positions.

For example: You have ten spaces before each line, and each line starts in column two. Later, you change the text to a larger font. Now the blanks are also larger and the text now starts in column 3. Every time you change the size of the font, you must change the number of spaces to make the text nice and even again.

There is an easier way. Use tabs to start a line at a different column. This way, if you change the size of the characters, the first character of each line

always starts in the same location. On the other hand, if you want to change all of the lines to start at a new column, all you need to do is change the tab position. To change the tab settings, just click on the column you want the tab. The cursor will move to the next tab location each time you press the <CONTROL><I> key combination.

Use the <RETURN> key sparingly. As a rule, you should only use the <RETURN> key when you reach the end of a paragraph. In other words, don't end each line with a <RETURN>. geoWrite automatically moves any word that doesn't fit on the current line to the next line. If you do press <RETURN> at the end of each line, you will run into some problems when you try to format the text. Press <RETURN> only at the end of a paragraph.

You can delete characters in the middle of the document simply by clicking the fire button to the right of the characters you want to delete, then using the <DEL> key. You can change individual characters, rather than just deleting them. Let's say you type the word `letter`, but you wanted to type `Letter`. Just put the cursor to the right of the `l`, press the <DEL> key, and then type `L`.

It is simpler to click on the `l`, move the cursor to the right (keep the fire button depressed), and then release the fire button after the `l` is marked. Then all you have to do is type `L`. This sounds difficult, but once you try it a couple of times, it is really very easy. You can also use this method to change entire words or phrases, not just characters.

Use the `preview` option of the **`file`** menu frequently. Whenever you change the margins or the fonts, your centered text, etc., will not look correct. `preview` returns all the text to its proper perspective. This is a lot easier than continuing, and then having to come back to readjust all the text so that it looks "right".

There are other tricks you can use to make the time you save even more significant. If you type a multi-page document, and later go back and add text to the first page, the extra lines of the first page are now moved to the second page, and the some of the text from that page are now on the third, and so on. If you want to break a page, even though the page isn't yet full, you can use `page break`. This option could be a real timesaver if you really want pages to end at specified points.

If you later decide that you want two pages separated by the `page break` to be put back together again, all you need to do is remove the `page break`'s special character. To remove the page break, just place the cursor to the right of the special character and press the <DEL> key. geoWrite asks you to confirm—click `OK`:

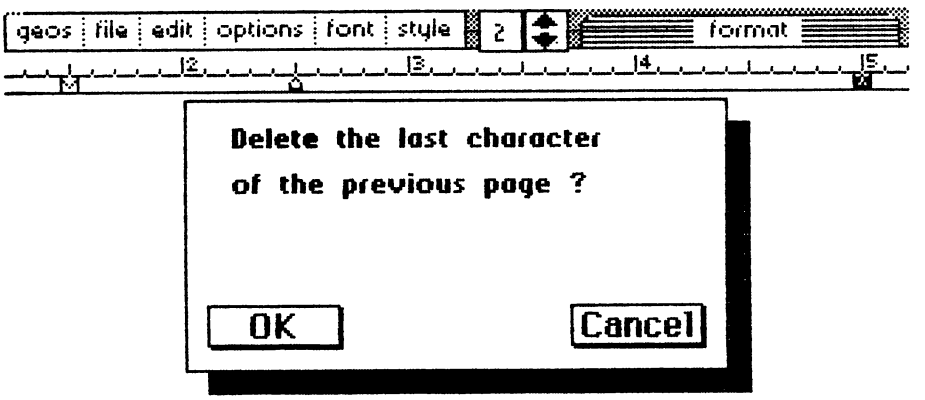


Figure 70: `Delete page break?`

You can do the same thing with graphic document. To get rid of the graphic document, simply place the cursor at the end of the graphic document and press <DEL>.

Some graphic document are as wide as the text. If you mark the document and then change the point size or use the tabs to get a column of smaller text, some of the document is cut off on both the screen and the printout.

You can place a picture in the middle of text. You don't have to worry about leaving enough space for the picture, before the text continues. geoWrite will take care of this for you by inserting the graphic document into the text.

5.1.5 Tips and Tricks for Notepad

Notepad is a collection of 127 note pages. You can use the Notepad from the deskTop or from any application. You can use the Notepad to write yourself notes at any time and look at them later. But you can do even more than this with the Notepad.

You can use the Notepad as a daily calendar/reminder. Reserve a page for each day and enter your appointments on the appropriate page. Each day you can call up the page for that day to review the appointments you have that day. A page in the notebook can look something like the example below:

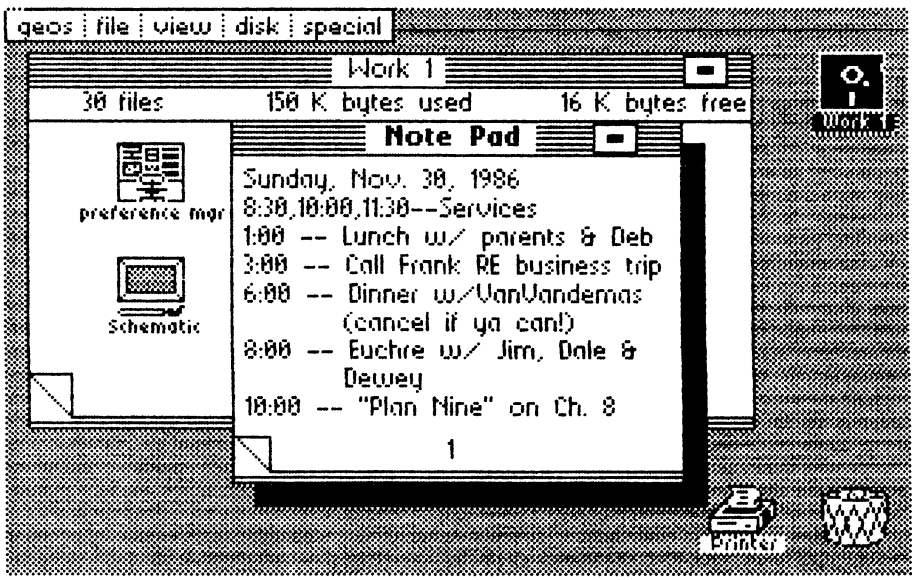


Figure 71: Notepad as daily calendar

You can even use the appointment calendar as a way of checking to see what you did on any given day. If you left yourself notes about calling someone or doing something, you can refer back to the appropriate day and confirm that it was done. In most cases you might want to use the calendar for a single month. Page 1 is always the first to appear when you use Notepad. You have to click the folded corner of the Notepad to turn the page you want. If you have a lot of pages it gets very annoying to have to turn through a lot of pages to get to the current day.

If the notebook becomes full and you feel it's time to start over, you can clear the `Notes` file on the work diskette. The next time you start up the Notepad you will have a clean notebook.

The number of characters that can appear on a single page is dependent on two factors. On one hand, the maximum number of characters per page is 254. On the other hand, you cannot start a line where the dogeared corner is located. If you use proportional text, then the number of characters you can fit on the page of the notebook will depend on the widths of the characters you use. Using narrow characters such as "." you would use all 254 characters in about 2/3rds of the page; however, if you used all W's, you couldn't fit 254 characters on a single page.

You can use the Notepad to store information that you need to get at quickly. What we need is a notebook manager. You can keep addresses or telephone numbers in this convenient Notepad manager. Figure 72 has a sample of one way to keep an address book using Notepad:

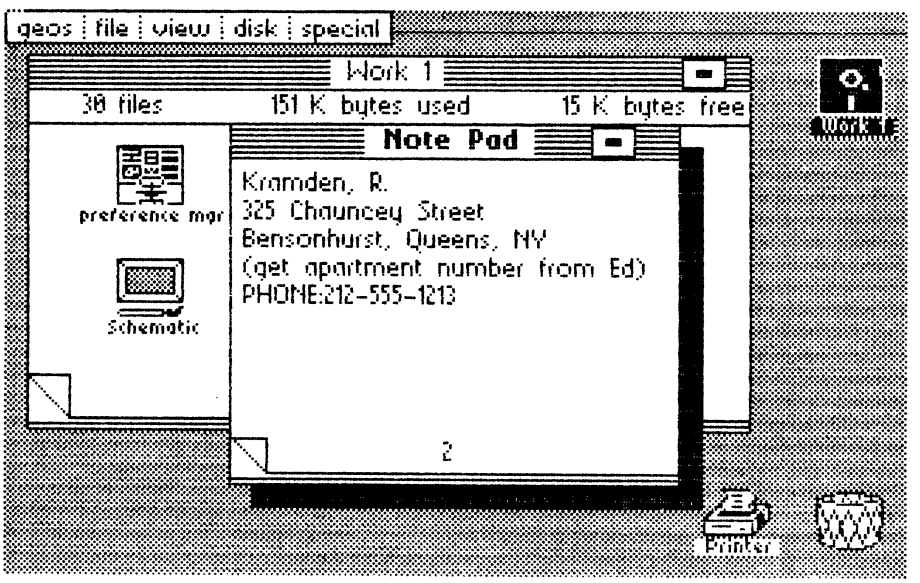


Figure 72: Using Notepad as an address book

You can keep other information you need in the notebook. The catch is, you can only have one notebook on each work diskette. You can get around this by keeping different data in notebooks on separate diskettes. An example would be to keep addresses and phone numbers in one notebook, recipes in another, and your car's fuel consumption records in yet another. Below we have another example of how to use the notebook for a recipe box. *Bon Appetit!*

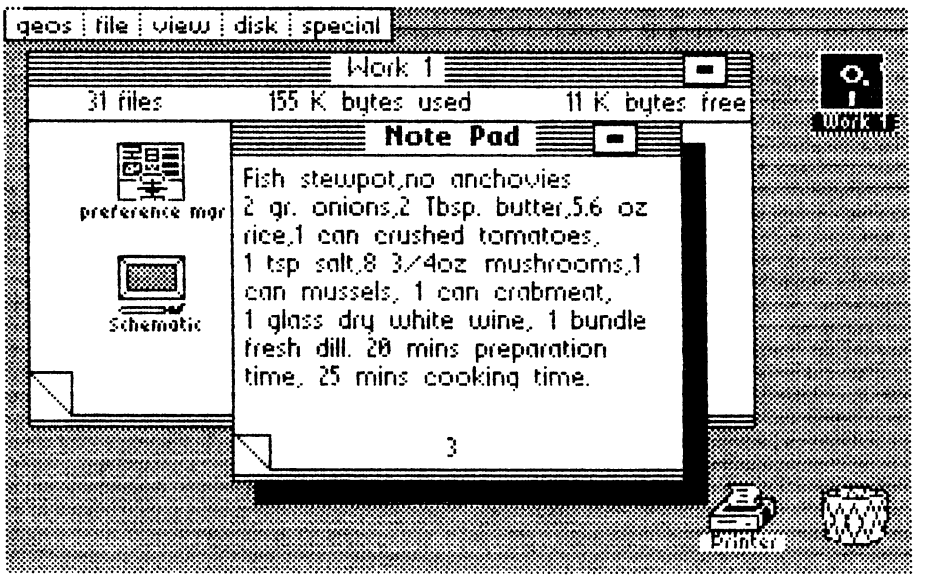


Figure 73: Notepad as a recipe file

In this next section we'll discuss some programming concepts and how to use a diskette monitor, as well as some more tips to manage the notebook. If you want to learn more about the basic information, read the section about the `VLIR` file structure in Chapter 6.

Normally, a file entry in the directory contains a pointer to the starting location of that program on diskette. The sectors that compose the file are then tied together by the first few bytes of each sector. If you were to look at the first sector and file entry for `Notes`, where the data should start, you'll find a group of numbers instead. These numbers are the pointers to the *track, sector* format. Therefore, the first pointer is for the first page, the second is for the second page, and so on.

In other words, this sector contains pointers to each page in the notebook. What would happen if we used a diskette monitor to change the pointers? Later, when Notepad searches for a page, it will display a different page than it would normally. Therefore, you can manipulate Notepad with a BASIC program. Just read in each field of the file (`B-R:` and `B-W` commands).

With a simple program you could:

- Change the page numbering
- Clear the first page
- Clear the last page
- Clear any page you want

When you have changed the page you want, simply save the file and the notebook is changed.

If you use your Notepad as an appointment calendar, and you don't like having to clear out everything at the end of every month, wouldn't it be easier to write a program that clears out the first twenty days and moves the remaining days to the front—i.e. keeping your calendar rotating? Also, this way the file will never get too large.

It's not usually possible to print the contents of the notebook to a printer directly from GEOS. This is a definite disadvantage. However, it's possible to load the ASCII file directly from the `Notes` file on diskette to a reserved area in memory. From there you can send the file either to the screen or to a printer. Using what we learned above, you can print any page you want, or the entire Notepad.

There's one thing you have to remember if you do want to print the contents of the Notepad. GEOS uses a form of security on the file. Therefore, when you read in the file, you will also have to do a few manipulations to print out the file in a legible form.

5.2 Programs in GEOS format

If you've worked with GEOS for a while, you have probably noticed that you couldn't add your name as the programmer in the `info` box. Perhaps a friend of yours has copied one of your programs, but he can't tell it's one of yours. Or maybe you have an interesting idea for an icon that would be perfect for identifying a program you've written, but you can mark only GEOS programs, and not standard C-64 files. If this is the case, then we have a solution to these problems.

Working with GEOS, we have come across many different file format possibilities. We could just send you to Chapter 6, **Inside GEOS**, to read every detail about how the info box is handled under GEOS and how you can access it with a disk monitor. But maybe you don't have a disk monitor, or don't have the slightest idea of what to do with one.

That is why we've done the work for you. The `FILEMASTER` program in this section quickly and easily transforms your standard C-64 files into GEOS format.

Before we start, let's have a quick lesson in how a file is converted to the GEOS format. For a more detailed summary, refer to Chapter 6 and Abacus' *Anatomy of the 1541 Disk Drive* or *1571 Internals* .

Normally, the directory of a diskette has an entry for each file with the following information:

1) File type
2) Track and sector of the first data block
3) Name of the file, ended by the character <SHIFT><SPACE> (ASCII $A0)
4) Number of sectors in the file

Each file entry has 30 bytes reserved for it. However, it does not use the full 30 bytes. The unused bytes are simply set to `0`. GEOS uses these unused bytes for the following purposes:

1) GEOS file type
2) Track and sector of the info box
3) Date and time

5.2.1 FILEMASTER

To convert a file to GEOS format, you will have to free up a sector for the info box, and add the correct information to the file entry in the directory. Finally, the icon and the associated text must be entered and saved.

FILEMASTER takes care of all of this for you. The icon has the same format as a sprite, which you may be familiar with from the C-64 handbook. Therefore, FILEMASTER also has a built-in sprite generator.

Before you go on to the FILEMASTER listing, we would like to mention the control codes in this program listing. When included in quotation marks as part of a BASIC PRINT statement, these control codes perform functions during program execution. These codes are difficult to read when a BASIC listing is printed out on a Commodore-type printer. We have listed FILEMASTER in a form that makes these control codes readable. Here are the codes and their meanings:

(home)	=	<CLR/HOME> key
(clr)	=	<SHIFT>ed <CLR/HOME> key
(down)	=	<CRSR DOWN> key
(rght)	=	<CRSR RIGHT> key
(up)	=	<SHIFT>ed <CRSR DOWN> key
(left)	=	<SHIFT>ed <CRSR RIGHT> key

```
20 rem filemaster manfred tornsdorf
30 printchr$(14)
40 for i=0 to 7:read sp% (i):nexti
60 print chr$(147)
80 v=53248:pokev+21,0
100 print"(clr)(rght)(rght)(down)(down)(down)Pleas
e insert diskette & press a key."
120 geta$:ifa$=""then120
140 open5,8,15,"i"
160 open3,8,3,"#0"
180 print chr$(147)
200 print"(down)(rght)(rght)(rght)(rght)(rght)(rgh
t)(rght)(rght)(rght)(rght)(rght)(rght)(rght)(rght)
FILE-MASTER"
```

```
220 print"(rght)(rght)(rght)(rght)(rght)(rght)(rgh
t)(rght)(rght)(rght)(rght)(rght)(rght)------------
-"
240 print"(down)(rght)(rght)(rght)Convert Files in
to GEOS format."
260 print:print
280 print"(rght)(rght)(rght)Disk directory
 = dir"
300 print"(rght)(rght)(rght)Load sprite
 = spl"
320 print"(rght)(rght)(rght)Create sprite
 = csp"
330 print
340 print"(rght)(rght)(rght)Read file
 = rfl"
380 print"(rght)(rght)(rght)Display file info
 = fdi"
400 print"(rght)(rght)(rght)Create GEOS file
 = geos"
420 print"(rght)(rght)(rght)Input/change date
 = date"
430 print
440 print"(rght)(rght)(rght)Change diskettes
 = disk"
460 print"(rght)(rght)(rght)End program
 = end"
480 print"(rght)(rght)(rght)Read info
 = info"
500 n$="xxxxxxx":print
520 input"(rght)(rght)(rght)Your choice:";n$
540 if n$="dir" then gosub 4060
560 if n$="spl" then gosub 4060
580 if n$="csp" then gosub 2320
600 if n$="rfl" then gosub 800
620 if n$="date" then gosub 3160
640 if n$="fdi" andz1=1 then printchr$(147):gosub
980
660 if n$="info" then gosub  5100
680 if n$="disk" then close3:close5:goto100
700 if n$="end" then 760
720 if n$="geos" then gosub 3540
740 goto 180
760 close3:close5:pokev+21,0:end
780 stop:rem**********************
```

```
800 z1=0:gosub 1380 :if t=0 or z1=0 then return
810 for i=1 to 500:nexti:printchr$(147)
820 at=t: rem dir-track found
840 as=s: rem dir sector found
860 ab=bp:rem buffer pointer filetype
880 tf=ts:sf=ss:rem track file,sector file
900 fi$=na$ :
920 gosub2220:fs=a:gosub2220:gt=a:gosub2220:ja=a:j
a$=str$(a)
930 gosub2220:mo=a:mo$=str$(a)
940 gosub2220:ta=a:ta$=str$(a):gosub2220:ho=a:ho$=
str$(a)
950 gosub2220:mi=a:mi$=str$(a)
960 gosub2220:ll=a:gosub2220:lh=a:
980 printchr$(145)chr$(145)"            ";fi$;chr$(
145);chr$(145)
1000 print"starting  track :";tf
1020 print"starting sector :";sf
1040 print
1060 print"file type       :";ft
1080 print"info track      :";it
1100 print"info sector     :";is
1120 print#5,"b-p:";3;bp+21
1140 print"file structure  :";fs
1160 print"geos f-type     :";gt;
1180 if gt=1 then print" =Basic"
1181 if gt=2 then print" =Assembler"
1182 if gt=3 then print" =Data File"
1183 if gt=4 then print" =System File"
1184 if gt=5 then print" =Accessory  "
1185 if gt=6 then print" =Application"
1186 if gt=7 then print" =Application Data"
1187 if gt=8 then print" =Font File  "
1188 if gt=9 then print" =Printer Driver"
1189 if gt=10then print" =Input Driver"
1190 if gt=0 then print"Non-GEOS file"
1220 if gt< 0 and gt >10 then print
1240 print"Year    :";ja
1260 print"Month   :";mo
1280 print"Day     :";ta
1300 print"Hour    :";ho
1320 print"Minute  :";mi
1340 print"file length :";ll+256*lh
```

```
1350 print
1360 input"(down)(down)(down)(rght)(rght)(rght)<RE
TURN> to go on";x$:return
1380 rem ****************************
1400 printchr$(147):t=18:s=1:
1405 print
1410 print
1420 input"(rght)(rght)(rght)Name to search for:";
su$
1440 su=len(su$)
1460 printchr$(147)
1480 sv$="":fori=1tosu: sv=asc(mid$(su$,i,1 )):if
sv>192 andsv<219 then sv=sv-96
1500 sv$=sv$+chr$(sv):nexti:
1520 print#5,"b-r:"3;0;t ;s
1540 print#5,"b-p:"3,0
1560 bp=2
1580 gosub 2220
1600 tn$=a$:tn=asc(tn$)
1620 gosub 2220
1640 na$=""
1660 sn$=a$:sn=asc(sn$):ifn$<>"dir"then print"t=";
t;"  s=";s
1680 print#5,"b-p:";3;2
1700 for j=1 to 8
1720 print#5,"b-p:";3;bp
1740 gosub 2220:ft$=a$:ft=asc(ft$)and 63
1760 gosub 2220:ts$=a$:ts=asc(ts$)
1780 gosub 2220:ss$=a$:ss=asc(ss$)
1800 for i=1 to 16:gosub2220
1820 if asc(a$)=160 then i=16:goto 1860
1840 na$=na$+a$:
1860 nexti:
1880 print#5,"b-p:";3;bp+19
1900 gosub2220:it$=a$:it=asc(it$):
1920 gosub2220:is$=a$:is=asc(is$):
1940 if ft=0 then 2080
1980 print"  ";ft;"  ";ts;" ";ss;"  ";chr$(34);na$
;chr$(34);
2000 print"  ";it;"  ";is
2020 if n$="dir" then 2080
2030 print
```

```
2040 if su$=mid$(na$,1,su) then print"(rght)(rght)
(rght)(rght)(rght)(rght)(rght)found.":z1=1:return
2060 if sv$=mid$(na$,1,su) then print"(rght)(rght)
(rght)(rght)(rght)(rght)(rght)found.":z1=1:return
2080 na$="":
2100 if peek(203)<>64 then 2180
2120 bp=bp+32
2140 nextj
2160 if tn<>0 then print chr$(17):t=tn:s=sn:goto15
20
2180 return
2200 rem**************
2220 get#3,a$:ifa$=""thena$=chr$(0)
2240 a=asc(a$)
2260 return
2280 rem**********************
2300 fork=0to62:poke832+k,0:nextk
2320 printchr$(147):print"Spritemaster"
2360 x1=0 :y1=0:x=x1:y=y1:ps=43
2380 vx=0:vy=0:vt=0 :
2400 poke2040,13:pokev+21,1:pokev,30:pokev+1,200
2420 for j=x1+14 to x1+23+14
2440 for i=y1+1 to y1+21
2460 p=1024+i*40+j
2480 gosub 3000:va%=vi%andvn%
2500 if va% <>0 then ps=42
2520 if va% = 0 then ps=46
2540 if p>1024 and p<2023 then pokep,ps
2560 y=y+1:next i : y=0
2580 x=x+1:nextj:x=0
2600 ps=43
2620 p=1024+(y+1)*40+x+14
2640 pr=peek(p)
2660 ifp>1023andp<2024 then pokep,ps
2680 geta$:ifa$=""then2680
2700 if a$=chr$(29) andx<23   thenx=x+1
2720 if a$=chr$(157)andx>0    thenx=x-1
2740 if a$=chr$(17) andy<20   theny=y+1
2760 if a$=chr$(145)andy>0    theny=y-1
2780 if a$="*" then pr=42:
2800 if a$=" " then pr=32:
2820 if a$="e" then 3120
2840 if a$=chr$(147) then 2300
2860 pokep,pr
```

```
2880 if a$<>"*"anda$<>" "then2620
2900 gosub 3000
2920 if a$="*" then pokevb%,vi% or vn%
2940 ifa$=" " then pokevb%,vi%andsp%(vt%)
2960 goto 2620
2980 rem ********************
3000 vy%=y :vx%=int((x)/8):vt%=x-vx%*8
3020 vb%=832+vx%+vy%*3 :rem byte number
3040 vi%=peek(vb%)       :rem contents
3060 vn%=2↑(7-vt%)       :rem new bit
3080 return
3100 rem ********************
3120 pokev+21,0:printchr$(147):return
3140 rem **** set date *****
3150 printchr$(147)
3160 printchr$(147):print"        ---------D A T E
----------
3180 print"            ";ta$;".";mo$;".19";ja$;"  "
;ho$;":";mi$
3220 print:print:print
3240 print"   Year     :";ja
3260 print"   Month    :";mo
3280 print"   Day      :";ta
3300 print"   Hour     :";ho
3320 print"   Minute   :";mi
3330 print
3340 print"   Change - type first two characters
    of choice (YE MO,etc.)"
3360 print"   Done   - type end"
3370 print:print
3380 input "   Your choice";x$
3400 if x$="end" then return
3410 print:print
3420 if x$="ye" then input"   Year      :";ja$:ja=v
al(ja$):goto3160
3440 if x$="mo" then input"   Month     :";mo$:mo=v
al(mo$):goto3160
3460 if x$="da" then input"   Day       :";ta$:ta=v
al(ta$):goto3160
3480 if x$="ho" then input"   Hour      :";ho$:ho=v
al(ho$):goto3160
3500 if x$="mi" then input"   Minute    :";mi$:mi=v
al(mi$):goto3160
3520 goto 3160
```

```
3540 printchr$(147) :rem ***** geos file set **
3560 if it<>0 then 3660 :rem should be set
3580 it=tf:is=sf:printit,is
3600 if it=18 then it=19 :rem dir-track not used
3610 print#5,"b-a:"0;it;is
3620 input#5,a,b$,c,d:printa,b$,c,d:
3640 if c<> 0 then it=c:is=d:goto 3600
3650 print:print:print
3660 print"    Please input GEOS filetype:"
3670 print:
3680 input"    Filetype";x$:x=val(x$)
3700 if x< 1 and x>15 then 3660
3720 fs=0:gt=x
3740 print#5,"b-f:"0;it;is
3760 print#5,"b-r:";3;0;at;as
3780 print#5,"b-p:"3,ab+19
3800 print#3,chr$(it);
3820 print#3,chr$(is);
3840 print#3,chr$(fs);
3860 print#3,chr$(gt);
3880 print#3,chr$(ja);
3900 print#3,chr$(mo);
3920 print#3,chr$(ta);
3940 print#3,chr$(ho);
3960 print#3,chr$(mi);
3970 if at<1 or at>35 then print:print"   ERROR:Il
legal track":goto4040
3980 print#5,"m-w"chr$(6)chr$(0)chr$(2)chr$(at)chr
$(as)
4000 print#5,"m-w"chr$(0)chr$(0)chr$(1)chr$(144)
4020 print:print "       Written":
4040 input " <RETURN> to continue";x$:return
4060 rem **search for dir and sprites **
4080 printchr$(147) :t=18:s=1:ifn$="dir"then 4200
4090 print:print:print
4100 input"   Name to search for:";su$
4120 su=len(su$)
4140 printchr$(147)
4160 sv$="":fori=1tosu:sv=asc(mid$(su$,i,1 )):if s
v>192andsv<219then sv=sv-96
4180 sv$=sv$+chr$(sv):nexti:
4200 print#5,"b-r:"3;0;t ;s
4220 print#5,"b-p:"3,0
4240 bp=2
```

```
4260 gosub 2220
4280 tn$=a$:tn=asc(tn$)
4300 gosub 2220
4320 na$=""
4340 sn$=a$:sn=asc(sn$):ifn$<>"i"then print"t=";t;
"  s=";s
4360 print#5,"b-p:";3;2
4380 for j=1 to 8
4400 print#5,"b-p:";3;bp
4420 gosub 2220:f1$=a$:f1=asc(f1$)and 63
4440 gosub 2220:
4460 gosub 2220:
4480 for i=1 to 16:gosub2220
4500 if asc(a$)=160 then i=16:goto 4540
4520 na$=na$+a$:
4540 nexti:
4560 print#5,"b-p:";3;bp+19
4580 gosub2220:i1$=a$:i1=asc(i1$):
4600 gosub2220:i2$=a$:i2=asc(i2$):
4620 if f1=0 then 4740
4640 if n$="i" then print"    ";na$:goto4740
4660 print"  "; chr$(34);na$;chr$(34)
4680 if n$="dir" then 4740
4700 if su$=mid$(na$,1,su) then print"       Found
.":goto 4900
4720 if sv$=mid$(na$,1,su) then print"        Foun
d.":goto 4900
4740 na$="":rem n$=""
4760 if peek(203)<>64 then 4860
4780 bp=bp+32
4800 nextj
4820 if tn<>0 then print" ":t=tn:s=sn:goto4200
4840 if n$<>"i"then print"   Sorry, not found."
4860 input"(down)(rght)<RETURN> to continue.";x$:r
eturn
4900 rem *** read sprite ***
4910 printchr$(147);
4920 print"                Read sprite";chr$(145)
4940 if i1=0 then print"Sprite not yet defined":fo
r i=0 to 1000:nexti:return
4960 print#5,"b-r:"3;0;i1 ;i2
4980 print#5,"b-p:"3,5
5000 for i=0 to 62
```

```
5020 gosub 2220
5040 poke832+i,a:printi,a;chr$(19)
5060 nexti:
5080 return
5100 rem *** read info screen *****
5120 if it=0 then return:rem track=0
5140 print#5,"b-r:"3;0;it;is
5160 print#5,"b-p:"3,0
5180 fori=1to6:a$(i,2)="":nexti
5200 gosub2220:ifa<> 0  then 5600
5205 gosub2220:ifa<>255 then 5600
5210 gosub2220:ifa<> 3 then 5600
5215 gosub2220:ifa<>21 then 5600
5220 print#5,"b-p:"3,71
5240 gosub 2220:x=a
5260 gosub 2220:x=x+256*a
5280 a$(1,2)=str$(x)
5300 gosub 2220:x=a
5320 gosub 2220:x=x+256*a
5340 a$(2,2)=str$(x)
5360 gosub 2220:x=a
5380 gosub 2220:x=x+256*a
5400 a$(3,2)=str$(x)
5420 gosub 2220:ifa=0 then 5460
5440 a$(4,2)=a$(4,2)+a$:goto 5420
5460 print#5,"b-p:"3,97
5480 gosub 2220:ifa=0 then 5520
5500 a$(5,2)=a$(5,2)+a$:goto 5480
5520 print#5,"b-p:"3,160
5540 gosub 2220:ifa=0 then 5580
5560 a$(6,2)=a$(6,2)+a$:goto 5540
5580 rem
5600 rem **** create info screen ******
5620 a$(1,1)=chr$(17)+"  1 Load address    : "
5640 a$(2,1)=chr$(17)+"  2 End address     : "
5660 a$(3,1)=chr$(17)+"  3 Entry point     : "
5680 a$(4,1)=chr$(17)+"  4 Class           : "
5700 a$(5,1)=chr$(17)+"  5 Author's name   : "
5720 a$(6,1)=chr$(17)+"  6 Help screen     : "
5740 rem ****************************
5750 printchr$(147)
```

```
5760 print"(clr)       Create info screen        "
5780 for i =1 to 6:printa$(i,1);a$(i,2):nexti
5790 print:print
5800 input"   Input number or E to end";x$
5820 if x$="E" or x$="e" then printchr$(147) :goto
6000
5840 if val(x$)<1 or val(x$)>6then5760
5850 print:print
5860 print" ";a$(val(x$),1);" ";a$(val(x$),2)
5880 printtab(21);chr$(145);
5900 if len(a$(val(x$),2))>19 thenprintchr$(145);
5920 input a$(val(x$),2)
5940 goto 5760
5960 rem
5980 rem
6000 print#5,"b-p:"3,0
6020 print#3,chr$(0);chr$(255);chr$(3);chr$(21);ch
r$(191);
6040 for i=0 to 62
6060 print#3,chr$(peek(832+i));
6080 nexti
6100 print#3,chr$(ft+128);
6120 print#3,chr$(gt);
6140 print#3,chr$(fs);
6160 if gt=1 then goto 6340
6180 rem x$=a$(1,2):gosub6720
6200 rem print#3,chr$(x2);chr$(x1);
6220 x$=a$(1,2):gosub6720
6240 print#3,chr$(x2);chr$(x1);
6260 x$=a$(2,2):gosub6720
6280 print#3,chr$(x2);chr$(x1);
6300 x$=a$(3,2):gosub6720
6320 print#3,chr$(x2);chr$(x1);
6340 x=len(a$(4,2)):if x>19 then x=19
6360 print#5,"b-p:"3,77
6380 for i=1 to x:x$=mid$(a$(4,2),i,1):print#3,x$;
6400 nexti :print#3,chr$(0);
6420 x=len(a$(5,2)):if x>19 then x=19
6440 print#5,"b-p:"3,97
6460 for i=1 to x:x$=mid$(a$(5,2),i,1):print#3,x$;
6480 nexti :print#3,chr$(0);
6500 x=len(a$(6,2)):if x>40 then x=40
6520 print#5,"b-p:"3,160
```

```
6540 for i=1 to x:x$=mid$(a$(6,2),i,1):print#3,x$;
6560 nexti :print#3,chr$(0);
6570 if it<1 or it>35 then print" False track":got
o6680
6580 print#5,"m-w"chr$(6)chr$(0)chr$(2)chr$(it)chr
$(is)
6600 print#5,"m-w"chr$(0)chr$(0)chr$(1)chr$(144)
6620 print"        Written":
6640 print#5,"b-a:"0;it;is
6660 input#5,a,b$,c,d:printa,b$,c,d:
6680 input"(down)(rght)(rght)(rght)<RETURN> to con
tinue";x$
6700 return
6720 x=val(x$):x1=int(x/256):x2=int(x-x1*256 )
6740 return
6760 data127,191,223,239,247,251,253,254
```

NOTE: Some older C-64 models might have problems with the `Spritemaster` section of the `Filemaster`. Some older '64s would need color memory `POKE`d as well as screen memory, otherwise the `Spritemaster` would be onscreen, but in the background color (invisible). If you get a blank screen when you call the `Spritemaster`, change these lines:

```
2460 p=1024+i*40+j:cm=p+54272
2540 if p>1024 and p<2023 then pokep,ps:poke cm,14
2620 p=1024+(y+1)*40+x+14:cm=p+54272
2660 if p>1024 and p<2024 then pokep,ps:poke cm,14
2860 pokep,pr:pokecm,14
```

5.2.2 Description of the FILEMASTER program

If all you want to do is use FILEMASTER, then you can probably skip this section. On the other hand, if you want to alter the program or use parts of it in your own programs, then you should definitely study this listing for its content. Before you do, you should be familiar with BASIC and all of the commands.

40	Input of 8 values in the SP% field. These values are used by the sprite master to clear specific pixels of the sprites.
60	Clear the screen.
80	V=starting address of the Video controller. Register 21 turns the sprites on or off. 0 turns all the sprites off.
100-120	Wait until a key is pressed. Used at places where a diskette change is needed during the course of the program.
140	Open and initialize the command channel for the diskette. This is where the diskette RAM is written to in the program buffer. For more information see *The Anatomy of the 1541 Disk Drive* or *1571 Internals*.
160	Open the channel to the disk drive. This channel will use buffer 0 ($0300-$03FF).
180-480	Display the menu.
500	The input variable N$ is set to a value, which is not available in the menu. This prohibits any erroneous possibilities when choosing a menu item.
520-720	Input and jump to the appropriate subroutine.
740	After returning from the subroutine, or in the case of bad input, redisplay the menu.
760	Jump here if the input value END is entered. This closes both channels and halt the program. To restart the program, all you would need at this point is a GOTO 100.

800- Subroutines: File searches and data display.

800 Z1=0 means that no file entry has been found. After an entry is found in the subroutine at 1380 the variable Z1 is set to 1. If Z1=0, control is returned to the menu. The same thing happens if T=0, that is, if the actual track is 0. This would mean that a search of the diskette at track 0 would cause an error.

820 This variable stores the track information where the file entry is located. This variable is really unnecessary, since the directory for the diskette is always located on track 18. As a matter of review for later, we store this value anyway.

840 The sector where the file entry was found.

860 Each disk buffer contains a pointer, which points to the next byte to be read. AB contains the position of the first character of the file entry (this is also the file type.)

880 The starting track of the program is stored in TF, and the starting sector is stored in SF.

900 The variable NA$ contains the name of the file entry. The full name is actually found in the variable FI$. If you were looking for the file entry ALARM CLOCK, all you need is ALA. This would point to the full name ALARM CLOCK.

920-960 The information that GEOS uses in read in at this point. The subroutine at 2200 is called, which returns a character and the associated ASCII value, these reside in A$ and A.

980-1340 The portion of the program where the data in the file entry points. All of the information of the file entry is stored here. Those files which are not stored in the GEOS file format will have all zeroes. The variables JA, MO, TA, HO, and MI have the date and time.

1360 Wait until all of the information is read in, then return to the menu.

1380- Subroutine: Search for the file entry in the directory.

1400 Clear the screen, and set the actual track and sector to 18 and 1, respectively.

1420-1500 The name of the needed file entry is in SU$. At the same time the altered form is in SV$. Since GEOS codes every character strangely, all GEOS file entries will not be found.

1520 Put the current track and sector into buffer 0.

1540 Set the buffer pointer to the beginning of the buffer. With this we can read or save the track and sector of the next blocks in the directory.

1560 Set the variables for the buffer pointers to the beginning of the first file entry.

1580-1600 Read in the starting track of the next block and put in TN..

1620-1660 Read in the starting sector and put in SN.. The actual file name is cleared (NA$="").

1680 Sets the buffer pointer, in this case, to the start of the first file entry.

1700 Loop through the eight entries in the sector of the directory.

1720 Set the buffer pointer to the correct value. This value is calculated at 2120.

1740 Get the file type. If the file is protected against deleting, the value FT is compared with 63 using the AND operator. This will not affect either of the first two bits.

1760-1780 Get the track and sector of the first data block. This is also the beginning of the actual program to which the file entry belongs.

1800 Loop through the 16 possible characters of the file entry: Get one character.

1820 If the character is a <SHIFT>ed <SPACE> (ASCII=160), then the name is finished.

1840 Puts the filename together from the individual characters.

1860 End of the loop.

1880 The buffer pointer is set to the first character after the name. This is where GEOS puts its additional information.

1900-1920 Get the track and sector of the info box from the file entry.

1940 If `FT`(File type)=`0`, then the file has been erased. In this program, we don't care about erased files. However, this could be used to your advantage to save erased files in other programs. All you would need to do is add the line: `IF FT=0 THEN FT=2`. Naturally, it is possible for another program to have written over the file since it was erased. Then again, maybe a new file wasn't written over the file, in which case the file can be salvaged.

1980-2000 The name and other important information is displayed on the screen. This allows you to see if the subroutine did actually find the file we're interested in, or if it found something else by mistake.

2040 Make sure that the name of the file found is the same as the file the user originally asked for. To do this we will only use the full name in `SU$`. If the names agree then `Z1=1` and the routine ends.

2060 Check the altered characters in the GEOS file entry.

2080 The actual name is cleared. This is important, since the individual characters were stored earlier. If they weren't cleared, the name would continue to grow.

2100 If you want to break the search, all you need to do is press a key. Then the location 203 is unequal to 64 and the subroutine is ended. This would be important if you mistyped the filename and don't want to wait for the program to realize the file doesn't exist.

2120 Sets the variable for the buffer pointer to the beginning of the next file entry.

2140 End of the loop for all eight file entries.

2160 If the file entry is not found in this block of the directory, then the variables TN and SN are used to move to the next block. This will allow us to continue to search for the file entry.

2180 The file entry is not found. We reached the end of the directory and the subroutine ends.

2200-2280 Get a character from the disk buffer and put it in A$ and the ASCII equivalent in A.

2300- Clear the sprite block.

2360-2380 Set the variables for the coordinates of the beginning values.

2400 The information for sprite 1 is in block 13 (13*64=832). Sprite 1 is turned on and the coordinates are set to 30, 200 (lower left corner).

2420- Also, an enlarged picture of the sprite appears on the screen. This rectangle takes up over half of the screen.

2420 Loop through all columns of the enlarged sprite.

2440 Loop through each line.

2460 P=actual position on the screen.

2480 Get the contents of the actual byte (VI%) in the sprite block and the position of the appropriate pixels (VN%). VA% is 0 if this pixel is set.

2500 If the pixel is set, then the character on the screen is a "*".

2520 If the pixel is not set, then the character is a . (period).

2540 If the position is inside of the screen, the character is saved in the screen memory. This opens possibilities of changing the size of the rectangle, or even restoring the screen if any error occurs.

2560 next row

2580 next column

2600 43 is the value for a "+". This cross appears in the actual location.

2620 Calculate the memory location in the screen memory for the actual position.

2640 The contents of this memory location is saved in PR, and the 43 for the cross is placed in the screen memory.

2660 If the position is inside of the screen memory, then the "+" is placed in this position.

2680 Wait until a key is pressed.

2700 <CRSR RT> : If the right hand edge hasn't been reached, then the position of the cross is moved to the right by one location.

2720 <CRSR LFT>

2740 <CRSR DN>

2760 <CRSR UP>

2780 If a * is pressed, a pixel is set at that location, and the saved location is changed to a *.

2800 The spacebar can clear the location (ASCII=46).

2820 The <E> key will end the work with the sprite master.

2840 Pressing the <SHIFT>ed <CLEAR/HOME> combination clears the sprite. This makes life easier if you want to start over.

2860 Return the saved character to the same position. This can be changed with a <*> or a <SPACE>.

2880 If the last key pressed was a cursor key, then the polling is restarted. Nothing is changed.

2900 Otherwise, the subroutine at 3000 places the actual contents of the bytes into sprite block (VI%) and the actual bit in (VN%) and the position in (VT%).

2920 If a pixel was set, then the content of the sprite is ORed with the new bit.

2940 To erase a pixel you have to use the correct mask in the field SP% (position of the bit), and then clear it with an AND operation. The eight values for SP% are in the DATA statements at the end of the program, and are differentiated by the lack of a one in each of the eight bit locations.

2960 Return to the question loop

2980-3100 Calculate the position of the pixel in the sprite block which is in the enlarged sprite rectangle.

3120 End of work with the Sprite Master. The sprite is saved to memory and we return to the menu.

3140- Subroutine: Set date

3160-3180 Show the title and other data.

3200-3360 The values are shown on the screen. Using the first two characters, the value can be changed (for example: DA for DAY). Enter END to exit the program.

3380-3520 After changing a value, save the value and redisplay the data.

3540- Subroutine: Change the file into the GEOS format.

3560 If the variable IT is not a 0, then the info box is already installed and the change doesn't take place.

3580 IT, IS are track and sector of the info box. The starting value is set and displayed.

3600 This sector is checked to make sure that you are not going to overwrite anything. If you will, then a disk error is announced, and the next free sector is displayed.

3620 Read the disk error report.

3640 If an error does occur, then C=0. Otherwise, C is the track, and D is the sector of the next free blocks.

3660-3700 Your are asked for the GEOS file type. This program will accept all the file type numbers (see Section 6.5.5, page 262 of this book).

3740 The sector for the info box is returned to free space. Otherwise, the diskette must be searched for another free sector. This section looks for the free sector for the same file entry.

3760 The block containing the directory for the file entry must be reread into the disk buffer. Now the changes can be made to the file entry.

3780 Set the buffer pointer to the first byte behind the filename.

3800-3960 Write the changed value to the buffer.

3980-4000 These two lines write the buffer to the diskette. Now, the change has been made and saved on the diskette. In this case we would want to use the command (B-W:)3;0;AT;AS. Of all things, this command is not very reliable. It saves the first character incorrectly. Since we want a very reliable, error-free program (especially at this point), we did not use this command.

Warning: This command is a powerful tool. It can totally erase your diskette if misused. If you want to change FILEMASTER, please use the following guidelines:

- Never switch the order of the two lines.
- Never allow AT to become larger than 35. This command will allow you to write to tracks over 35. The floppy head could reach the end of the diskette and strike the back of the drive, which can damage the drive.

4020-4040 Information about the write operation and return to the menu.

4060-4860 This subroutine is used in conjunction with the file search at line 1400. There are two differences between them:

- If the subroutine is called from the menu, then N$="DIR" and the actual name of the file is in NA$, and no other information is found. At the end of the routine, control is given back to the main program.
- If the subroutine is called from the sprite input, then it will search for the file entry, and if found it jumps to the routine at 4900. Otherwise it will return to the menu.

4900 Subroutine to input sprite.

4940 If the file entry did not have an info box, then IT=0, and the sprite is not read.

4960 Get the sector of the info box in the buffer.

4980 Set the buffer variable to the beginning of the sprite.

5000-5080 Read in the 64 bytes of the sprite from the buffer and write it to block 13. Afterwards return to the main menu.

5100 Subroutine: info box input, change and save.

5120 If the file entry does not have a pointer to the info box, the subroutine ends and returns to the menu.

5140 Read the sector with the info box into the buffer.

5160 Set the buffer pointer to the first byte of the buffer.

5180 Set the text to the value . (period).

5200-5215 Test the first four bytes of the sector. These must have special values—otherwise this sector does not contain the info box. If everything is as it should be, then the text should be 0. Otherwise, if the sector does not have the info box, then there will not be 0. If this is the case, then the remaining text is skipped.

5220 Set the buffer pointer to the beginning of the output text in the info box.

5240-5560 Read the remaining six values. The final value is recognized as a 0. This would be a problem, if the info box was not in the sector.

5600-5720 The text is all in a single field. Therefore you can use a loop to output the values.

5760-5780 Output loop for the title and text.

5800 Input the number to change or `END` to quit.

5820 If the answer is `END`, then change the buffer and rewrite it to diskette.

5840 Check the converted number and check for legality.

5860-5920 Enables text editing. The text is output to the screen and the cursor for the `INPUT` command is placed so that the contents can be changed.

5940 Output the edited text.

5980- Change the values and rewrite the text to the buffer.

6000 Set the buffer pointer to the start of the buffer.

6020 Set the first bytes. The entire story behind the values can be found in Chapter 6, **Inside GEOS**.

6040-6080 Read the values for the ICON in the sprite block into the buffer.

6100-6560 Write the text into the buffer and set the text to 0.

6570 The value `IT` is checked for legality for both of the next commands.

6580-6620 Rewrite the buffer.

6640 Identify the sector with the info box.

6660 Output the error message to the screen.

6680-6700 Return to the menu.

6720-6740 Change the value in `X$` into two numbers in the format HIGH byte/LOW byte.

6760 The value to clear individual pixels in the sprite is in the `SP%` field.

5.2.3 The FILEMASTER menu

After FILEMASTER is started, as well as any time a key is pressed, a menu is displayed on the screen. Here are the options:

DIRECTORY = **DIR**

This menu option displays the directory of the diskette. The name and the diskette ID are displayed. You can halt the output of the directory by pressing any key. The best keys to press are the function keys (F1-F8), since these do not print anything on the screen. FILEMASTER checks the keyboard only after each complete file entry is displayed. This is to make the input go faster. To return to the main menu, press the <RETURN> key.

LOAD A SPRITE = **SPL**

This option loads an existing icon from any GEOS file and stores it in the sprite block. You can store the icon as it is, or change it any way you wish.

First of all, FILEMASTER asks you to enter the name of the program that contains the icon. Enter the name as it appears in the directory. If the file is not found on the working diskette, select the option on the menu to switch diskettes. If you switch diskettes, after FILEMASTER is finished use the menu option DISK to return to the working diskette.

When you enter the name of the file, FILEMASTER searches the file entries on the diskette, and points to the name. If the program is found, then the message READING SPRITE appears, followed by the number 0 - 63. Afterwards, the menu reappears. If the file is not found, the menu automatically reappears.

CREATE A SPRITE = **CSP**

This option allows you to create a new icon, or to edit a previously-loaded sprite. You should know that a sprite is made of 63 values (bytes). Every byte is composed of eight bits; each bit describes a pixel. Therefore 63*8=504 pixels form a single sprite. The sprite generator makes creating a sprite very easy. By starting the sprite generator, you can see what happens as you read along.

The right hand area of the screen displays a matrix (a rectangle) that represents the sprite. If the sprite block at 832 is empty, then you have not loaded a sprite. This rectangle is 24 pixels by 21 pixels (a total of 504 pixels). If you enter a sprite, it appears in the lower left corner. Each pixel in the matrix is represented by a *. A cross appears in the upper left corner. This cross is the pointer to the position in the sprite as well as the cursor to set the pixels.

For example, if you want a * in a position, then move the cross to that location, press the * key and then press the <CRSR RT> key. This is what happens:

1) Cursor keys: Moves the cursor within the matrix. This is the means of changing individual pixels.

2) <*> (asterisk) key: Sets the pixel at a given location. This also sets the appropriate bit in the sprite block. The cursor does not change position.

3) <SPACE> bar: Clears a pixel in the sprite block.

4) <SHIFT><CLR/HOME>: In BASIC this key combination would clear the entire screen. In sprite master this clears the entire sprite. So if you want to start over, this is the quickest way to reset the sprite.

5) <E>: Ends the operation of the sprite master program and returns to the menu. The sprite is no longer pointed to in block 13—however, it is not cleared. If you want to use the sprite master later, then the sprite is still there for you to use.

READ FILE: **RFL**

This command reads a single file entry from the diskette. Enter the filename, and FILEMASTER searches the diskette and displays the contents of the file entry.

To find the name, you need only enter enough characters for the program to recognize the file you want. To find a file named ALARM CLOCK, you would only have to enter the three characters ALA. If the program cannot find the entry with these three characters, then you would have to enter the entire filename for the search.

If the file is not in the directory, then FILEMASTER returns to the menu. Otherwise, all the information in the directory will be displayed. These are:

1) Starting track and sector:
   Track and sector where the file starts on the diskette.

2) File type:
   Normal DOS file type; 0=DELETED, 1=SEQUENTIAL, 2=PROGRAM, 3=USER, 4=RELATIVE. On the original GEOS diskette, all files outside of GEOS and GEOS BOOT are USER files.

3) info track and sector:
   Track and sector of the info box. If both are set to 0, then there is no info box.

4) File structure:
   GEOS recognizes two file structures: SEQUENTIAL=0 and VLIR=1. For your programs you should always use sequential.

5) GEOS filetype:
   0=BASIC, 1=ASSEMBLER. For more file types see the INSIDE GEOS. However, the way the program is written then only use a '0' or a '1'.

6) Date:
   The date is stored in five parts: Year, Month, Day, Hours and Minutes.

7) Length of file:
   The number of sectors is the amount used by the file on the diskette.

After you press the <RETURN> key, FILEMASTER returns to the menu.

DISPLAY FILE INFO: **FDI**

This command shows all of the information in a file entry. This command works only if a file has been read.

CREATE GEOS FILE: **GEOS**

This command converts the data in the file entry on the diskette. The program searches for an unused block to store the info box. All you have to do is enter the date. After this is finished, the word WRITTEN appears on the screen. Pressing <RETURN> will bring back the menu.

INPUT/CHANGE DATE: **DATE**

This displays the date of the file, and allows you to change it. The default is 0. You can change any entry by entering the first two characters of the field name. For example: entering an HO will allow you to change the hour. You can exit this subroutine by entering the command END.

CHANGE DISKETTES: **DISK**

This command allows you to convert a file that is on a different diskette than the work diskette. After the diskettes are changed, control is returned to the menu.

END PROGRAM: **END**

Use this to leave the FILEMASTER program. Don't exit the program by pressing the <RUN/STOP> key. It's possible that something might not be completely stored, and the information could be lost if the program is not exited properly.

READ INFO: **INFO**

This command readies the info box. Before you can use this option, you must have either loaded an existing sprite, or created one using the sprite master routine. The FILEMASTER program checks the sector where the file entry said the info box is supposed to be.

After the info box is finished, then you can change the six variables. If your program is written in BASIC, then don't use the first three. These won't be used by GEOS, and therefore should not be written to diskette.

If you are finished with the input, then type E to end. The saved or altered info box will be written to diskette.

5.2.4 Using FILEMASTER

The FILEMASTER program alters the contents of the file directory on a diskette. We suggest that you copy the programs to be converted to a single work diskette. Use the work diskette for this purpose. You can then copy the changed programs to whatever diskette you wish later.

First, copy the program to the work diskette. Then load the FILEMASTER program, and start it with RUN. Insert the work diskette and press any key; the menu appears. To convert the program to the GEOS format, simply perform the following steps:

Read the file with the RFL command. We will assume that the program is a BASIC program.

Now create or load a sprite for the GEOS file. If you need to change diskettes, then use the CDSK command.

Use the DATE command to set the date of the file. Begin by setting the year with the YE command. FILEMASTER will then prompt you for the year.

Warning: When entering date information, be careful that you enter correct data— FILEMASTER does not check the validity of the date. If you enter something like 123456789 for the month, it will destroy the locations of everything else in the file entry, and the GEOS format is also destroyed.

Now choose the GEOS command to actually perform the conversion. The message 65 NO BLOCK is displayed on the screen. This is normal; it's telling us that FILEMASTER is searching for a free block to store the info box. Finally the message OK 0 0 is displayed. Enter a 1 for the file type for BASIC. The last thing to appear on the screen is the word WRITTEN, which tells us that the file has been converted.

If everything went perfectly, but there is still no info box for this text, then the following message is displayed:

```
0 OK 0 0
```

If the info sector is written, then the message:

65 NO BLOCK (and two more numbers)

is displayed. Then the block is correctly written.

Now, you can either quit FILEMASTER or convert another file if you wish.

The last thing you will do is copy the GEOS file to the work diskette, where you can use your program.

Finally, we want to leave you with some tips for working with FILEMASTER.

As you read through our code, you undoubtedly came across a few opportunities to change our program. To do this you will have to change at least one line of code. We wrote the code so that it is easily understood and very flexible, so that you can make the changes you want.

If you accidentally erase a file on your GEOS diskette, then you can recover it with FILEMASTER. This will only work if you have not saved any files on top of the one you want to recover. When you delete a file, the contents of the sector it occupied are not cleared. Rather, a file type of 0 is stored in the file entry, and the sectors of the file are freed for use. When you change the filetype, then the file is restored and ready to use.

If you want to include these features in FILEMASTER, all you have to do is change the following lines:

```
old:   1940 IF FT=0 THEN 2080
new:   1940 IF FT=0 THEN PRINT "***";
```

Now the cleared files are not overwritten, and can be restored. Doing this marks the file to be restored.

Now change the file type from 0 to another value:

```
new:   3765 IF FT=0 THEN PRINT#5, "B-P:"3,AB:
new:   3770 IF FT=0 THEN PRINT#3, CHR$(130);

old:   3720 FS=0: GT=X
new:   3720 GT=X
```

This will change the file type to a legal value. It is rewritten to the buffer at line 3980 and is changed on the diskette. This will cause one problem:

After we have altered a program with the GEOS command and set the info box with INFO, GEOS loads the command VALIDATE. This command checks the sector of the saved file, and makes sure the file was saved correctly.

We will go through how this takes place with a short example:

Pretend you have a text file you created named CONCEPT with geoWrite and have since erased. Now, quit GEOS and start up BASIC. Load FILEMASTER and make the changes described above. Insert the diskette with the "missing" file in the drive with the missing file. Enter RFL and the name CONCEPT. If the program is found, then the important data would be:

File type: 0
GEOS F-Type: 7
File structure: 1

Now choose the GEOS option from the menu, and change the file type from 0 to 7. Finally, save the change, and leave FILEMASTER. Reboot with GEOS. Insert the altered diskette in the drive. Select the command VALIDATE. Now you can either work on your data again, or print it.

FILEMASTER can handle all the file types available (see Section 6.5.5 for a detailed listing of GEOS file types and their identifying numbers). You are still responsible for making sure that the file type you enter is a correct value. If you give a text file the file type for a BASIC program, GEOS will try to load the file as a program and then try to run it. Odds are that the text will hang up the program and crash the computer.

If you want to change a GEOS file (like the date or your name as the programmer), read the file in first (RFL), then read the sprite in (SPL). Now you can change your file and not worry about losing your sprites.

5.3 The function of the real-time clock

GEOS uses at least one of the two real-time clocks in the '64. This clock's greatest advantage over the KERNAL-operated TI$ clock is its relatively high frequency. This is similar to the way your clock/radio uses the alternating current from the outlet for synchronization.

But clocks run "faster" in America than they do in other parts of the world, like Great Britain, Australia, and Germany. The clocks of these international Commodore owners run two seconds slower in a ten-second span than U.S. machines.

As you know, this discrepancy is due to the fact that European current alternates 10 Hz slower than the rate of U.S. current. Unfortunately, the GEOS developers have not been able to correct this problem for intenational users at this time. There is a program that can be used to evaluate every new GEOS installation. But the results are based on a programming error in the GEOS KERNAL, and the clock must be set to 60 Hz.

Before we start debugging, we'll quickly study how to program the real-time clock, as well as look at some routines in the GEOS KERNAL.

5.3.1 Programming the clock

The '64 has two 6526 microchips known as CIA's (for Complex Interface Adapters). These chips are used in conjunction with input and output to different peripherals (keyboard, disk drive, joysticks, etc.) and, in addition, control a twenty-four hour clock with a programmable alarm.

Of the 16 registers, we'll only discuss those that are relevant to the problem with the clock. The register numbers are related to the first memory locations for the different CIA's.

The time register

There are four registers that can be controlled to work with the actual time, in BCD format:

Register #	Contents
$08	1/10th seconds
$09	Seconds
$0A	Minutes
$0B	Hours

All registers can be read or written to. While reading the data from a clock register, the time cannot be updated. This means that the 1/10th second register is frozen. (The clock itself continues to run).

When writing to the clock registers, the clock stops just long enough for the 1/10th second register to be set, and to check the synchronization.

The control register ($0E)

Only two bits are needed to control the clock. These are the last two bits of the CIA register.

Bit 7 determines if your clock is running properly. If you are running at 50 Hz, then this bit must always be set to 1.

By solving this problem, we have actually taken ourselves further away from solving the second problem. The second control bit, which can be found in the same position in register $0F, determines if a subsequent write operations uses the normal clock or the alarm clock (bit 7 = 1).

The same memory register is used for setting both clock time and alarm time, as well as for reading the clock. This is the reason why you can't reset the alarm clock.

(You could solve this problem by setting the alarm time input in another memory register.)

Alarm output

To keep the actual time in synch with the time in the alarm clock, you must also correctly handle register $0D (the Interrupt Control Register, or ICR).

GEOS automatically sets bit 2 of this register whenever the alarm is set. This bit is inspected every cycle. Then alarm is eventually sounded when the interval has expired. This byte is cleared after every read to ensure that every set alarm will be sounded. For more information on the CIA, see *The Anatomy of the Commodore 64* , Chapter 8.

5.3.2. The routines in GEOS

To adjust the time and alarm there is a program on the GEOS diskette that is called ALARM CLOCK. You can use the program to set important dates. For this to work continually, the KERNAL routines must check the cycles. If an alarm is set for a specific time, then an audio/visual alarm is sounded.

The date calculation routines are related to the real-time clock. If you've ever left your computer running overnight, you know the Preference Manager automatically updates the new calendar date.

If the date is February 28th of a leap year, GEOS also handles this correctly. Both the clock routine and the date calculation routine work together. The date routine is accessed from the KERNAL at $C2C8.

If you use a machine language monitor, then you can see what we are talking about. You'll find the date and alarm routines in the $C2C8 region. Other routines that GEOS uses in normal operations during each cycle are located in $C2D1 through $C2D4. Following these are routines for the mouse.

Both of these subroutines are documented. The first is in RAM at $F390. The second can be found after $F9ED.

Note: These addresses may be different in some versions of GEOS. To help you find the correct routines, GEOS uses a jump table to access memory. The jump address for the job loop pointing to $C2C8 above is jump table address $C1C3.

```
************************ Time and Date Routines

f930 a5 01    lda $01   save old memory configurations
f932 48       pha
f933 a9 35    lda #$35  turn on I/O region
f935 85 01    sta $01
f937 78       sei
f938 ad dc 0f lda $dc0f switch alarm input to time input
f93b 29 7f    and #$7f
f93d 8d dc 0f sta $dc0f
f940 ad 19 85 lda $8519 contents of hours in hex-format
                        (0-24hr)
f943 c9 0c    cmp #$0c  is it after noon?
f945 30 08    bmi $f94f no:continue
f947 2c 0b dc bit $dc0b is the pm flag set?
f94a 30 03    bmi $f94f yes: continue
f94c 20 84 f9 jsr $f984 otherwise: set date and new day
f94f ad 0b dc lda $dc0b get hours
f952 29 1f    and #$1f  clear PM flag
f954 c9 12    cmp #$12  compare with 12 hour clock
f956 d0 02    bne $f95a unequal: continue
f958 a9 00    lda #$00  otherwise: set back to 0
f95a 2c 0b dc bit $dc0b am/pm test
f95d 10 05    bpl $f964 am: continue
f95f f8       sed
f960 18       clc
f961 69 12    adc #$12  add 12 for 24hr clock
f963 d8       cld
f964 20 d9 f9 jsr $f9d9 change hours (BCD) in hexadecimal
f967 8d 19 85 sta $8519 and store
f96a ad 0a dc lda $dc0a get minutes
f96d 20 d9 f9 jsr $f9d9 change to hex
f970 8d 1a 85 sta $851a and store
f973 ad 09 dc lda $dc09 get seconds
f976 20 d9 f9 jsr $f9d9 change to hex
f979 8d 1b 85 sta $851b and store
f97c ad 08 dc lda $dc08 get 1/10th  seconds
f97f 68       pla       restore old memory configuration
f980 85 01    sta $01
f982 58       cli
f983 60       rts
```

```
************************ calculate day
f984 20 b4 f9 jsr $f9b4 put number of days in this month
                        into accumulator
f987 cd 18 85 cmp $8518 and compare with today's date
f98a f0 04    beq $f990 same: add one to month
f98c ee 18 85 inc $8518 otherwise: only increment day
f98f 60       rts

************************calculate month and year
f990 a9 01    lda #$01  first day of the month
f992 8d 18 85 sta $8518
f995 ee 17 85 inc $8517 increment month
f998 ad 17 85 lda $8517
f99b c9 0d    cmp #$0d  end of the year?
f99d d0 14    bne $f9b3 not yet, continue
f99f a9 01    lda #$01  otherwise: january
f9a1 8d 17 85 sta $8517
f9a4 ee 16 85 inc $8516 increment year
f9a7 ad 16 85 lda $8516
f9aa c9 64    cmp #$64  end of the century?
f9ac d0 05    bne $f9b3 no: continue
f9ae a9 00    lda #$00  otherwise: year = 0
f9b0 8d 16 85 sta $8516
f9b3 60       rts

****************************get number of days this
****************************month, so far
f9b4 ac 17 85 ldy $8517    contents of actual month
f9b7 88       dey
f9b8 b9 cd f9 lda $f9cd,y get number of days from table
f9bb 48       pha         save
f9bc c0 01    cpy #$01    february (decrement by 1)
f9be d0 0b    bne $f9cb   no: continue
f9c0 ad 16 85 lda $8516   otherwise: get year
f9c3 29 03    and #$03    test for leap year
f9c5 d0 04    bne $f9cb   not leap year: continue
f9c7 68       pla         otherwise: forget value
f9c8 a9 1d    lda #$1d    new value = 29
f9ca 60       rts
f9cb 68       pla
f9cc 60       rts
f9cd 1f       ???
f9ce 1c       ???
```

```
f9cf 1f       ???
f9d0 1e 1f 1e asl $1e1f,x
f9d3 1f       ???
f9d4 1f       ???
f9d5 1e 1f 1e asl $1e1f,x
f9d8 1f       ???

************************change 2 BCD's to 1 hex
f9d9 48       pha       save BCD value
f9da 29 f0    and #$f0  get leftmost nibble
f9dc 4a       lsr       and rotate to right
f9dd 4a       lsr
f9de 4a       lsr
f9df 4a       lsr
f9e0 aa       tax       mark number
f9e1 68       pla       get BCD number
f9e2 29 0f    and #$ 0f clear left half
f9e4 18       clc       and add X times 10
f9e5 ca       dex
f9e6 30 04    bmi $f9ec
f9e8 69 0a    adc #$0a
f9ea d0 f9    bne $f9e5
f9ec 60       rts
```

Here are some things to look for in the listing:

The real-time clock uses the first CIA port location, located at $DC00. At the same time GEOS keeps its time in normal memory in hexadecimal format, and uses these routines to convert back and forth. This results in the following rules (the 1/10th second location is not used):

$8519	Hours
$851A	Minutes
$851B	Seconds

Since there is no register in the CIA for the date, it is found in RAM, and is taken directly from memory:

$8516	Year
$8517	Month
$8518	Day

```
************************Alarm output
f9ed a5 01     lda $01      Save old memory configuration
f9ef 48        pha
f9f0 a9 35     lda #$35     fade in I/O region
f9f2 85 01     sta $01
f9f4 ad 0d dc  lda $dc0d    Get alarm status
f9f7 85 04     sta $04      and save
f9f9 68        pla          old memory configuration
f9fa 85 01     sta $01
f9fc a5 04     lda $04      Alarm status
f9fe 2c 1c 85  bit $851c    Is an alarm time set? (bit 7=1)
fa01 10 18     bpl $fa1b    NO: Continue
fa03 29 04     and #$04     Is alarm set to go?
fa05 f0 1c     beq $fa23    NO: Finished
fa07 a9 4a     lda #$4a     YES: Get value for 10 beeps
fa09 8d 1c 85  sta $851c    (set bit 7 to 0, the alarm
                            will be handled.)
fa0c a9 00     lda #$00     Test addresses $84ad/$84ae
                            for a job
fa0e cd ad 84  cmp $84ad
fa11 d0 05     bne $fa18
fa13 cd ae 84  cmp $84ae
fa16 f0 03     beq $fa1b    No alarm job, output
                            normal GEOS beep,
fa18 6c ad 84  jmp ($84ad)  otherwise jump to job
fa1b 2c 1c 85  sta $851c    All beeps done?
fa1e 50 03     bvc $fa23    YES:Continue
fa20 20 24 fa  jsr $fa24    NO:Take care of beeps
fa23 60

******************** Output alarm sounds
fa24 ad 1b 88  lda $881b    Wait for system IRQ until
                            decremented to 0
fa27 d0 2a     bne $fa53    not 0:Perhaps next time
fa29 a5 01     lda $01      Save old memory configuration
fa2b 48        pha
fa2c a9 35     lda #$35     Blend in I/O region
fa2e 85 01     sta $01
fa30 a2 18     ldx #$18
fa32 bd 54 fa  lda $fa54,x  Sound device with date
fa35 9d 00 d4  sta $d400,x  is retrieved from table
 ***(sounds now)
fa38 ca        dex
```

```
fa39 10 f7     bpl $fa32
fa3b a2 21     ldx #$21
fa3d ad 1c 85  lda $851c
fa40 29 3f     and #$3f
fa42 d0 01     bne $fa45
fa44 aa        tax          Accumulator = 0:turn off alarm
fa45 8e 04 d4  stx $d404    Control register is set to voice 1
fa48 68        pla          Restore old memory configuration
fa49 85 01     sta $01
fa4b a9 1e     lda #$1e     Set a new value for the interval
                            pause
fa4d 8d 1b 88  sta $881b
fa50 ce 1c 85  dec $851c    Decrement number of beeps
fa53 60        rts
```

Things to look for in the listing:

You'll notice when evaluating the date that two memory locations are switched. However, the function of these locations is not specified in the listing:

1. The status register - $851C

Whenever you set a time for the ALARM CLOCK, you also set the 7-bit at this address. This is the basis for the decision whether or not an alarm has been set in the previous code (see $F9FE). If an alarm has been entered, and bit 2 of the ICR ($DC0D) is set, then the status register is loaded with the value #$4A. As long as bit 6 is set, then $FA20 in the alarm sound generator is started. This subroutine gets the values for the sound generator output from a table, located at $D400, then starts producing sound while any of the bits 0 through 5 are set ($FA3B - $FA45). The data for the sound generator is chosen to produce a short tone that dies off quickly.

Finally, the beep counter in $FA50 is decremented by 1.

2. The interval register - $881B

At first glance, it is difficult to determine the purpose of this location. It is accessed by the IRQ rountine (Interupt Request), which is activated when a signal is detected on one of the lines. The main GEOS loop is halted just long enough to check the keyboard input, read the joystick port, perform input and output.

In the area $E321, this location $881B is decremented by one:

```
$E321 LDX $881B
$E324 BEQ $E32A
$E326 DEX
$E327 STX $881B
$E32A ...
```

After the IRQ is complete, the computer returns to the interrupt routine so that the program can continue to run—at least until the next time an interrupt request comes along.

If you look refer to the sound output subroutine again, you will understand the purpose of the region at $FA24. Location $881B determines the pause between the beeps. If this register is not cleared, a beep is not sounded and the "Beep-Counter" is not decremented. When it is set back to zero by the interrupt routine, the next sound is produced. At $FA4B a new starting value representing the pause interval is placed in the register.

5.3.3 The time at a glance

Isn't it annoying to have to load the Preference Manager or ALARM CLOCK to sneak a quick look at the clock?

If you think so too, here's a program that constantly displays the time (in 24-hour format) in the upper left corner of the screen.

We have also made a change to the alarm output routine. The alarm signal itself is a sound from the computer and a quick flash on the monitor.

There was one problem with implementing this program: there is no room in memory for the program! GEOS does not leave any spare memory.

However, we found an area in the KERNAL that we could use without locking up the system. This means that we can use the routines with which you are already familiar. We can handle all of the hours in the day fairly easily. However, the hour we have to pay attention to is midnight. At midnight we have to change the date, as well as the time. However, we made the asssumption that most people who use GEOS will not use the program at that time of night. Therefore, it was easier to just eliminate this exception.

As you can see, we can handle every hour of the day in the same manner—we don't have to look at every hour to change the date.

The program is loaded into memory starting at $1000. When you type in and run the BASIC loader version of this program, it asks whether or not you want the clock program to load when you boot GEOS. If you press <Y> the program changes so that it loads as GEOS loads. Remember to set the time from the Preference Manager once the deskTop loads.

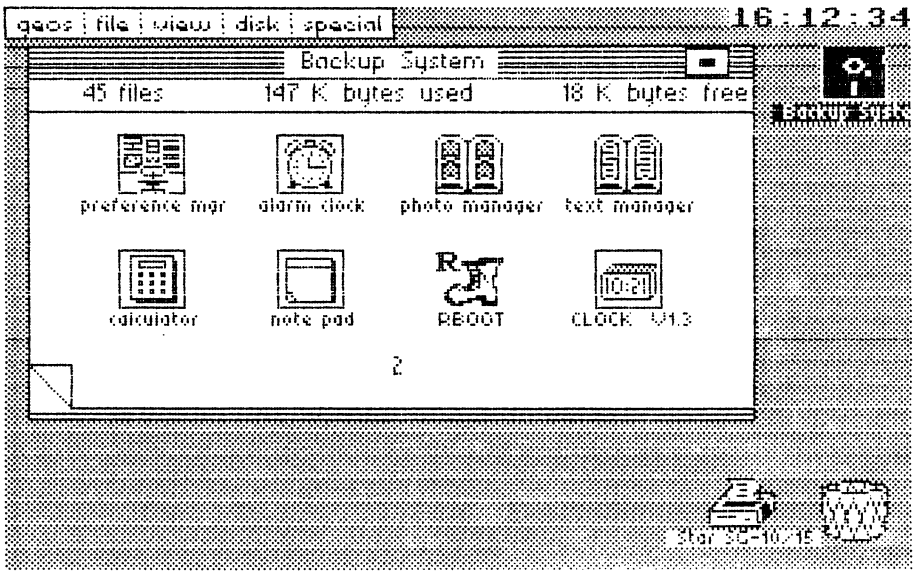


Figure 74: Constant display clock

Press <N> in response to the AUTOSTART? prompt if you'd rather just call the program from its icon. The loader generates the program. Double-click the icon to start the clock from the deskTop.

Our program is divided into three parts:

1) The graphic output area for the time display.
2) The date change from the original program.
3) The time display in the upper right corner of the screen.

There should be enough memory for both time routines to run. However, once you load an application, the application may overwrite the clock.

Here is the machine language source code of the program to show you the program flow. A BASIC loader version of the program follows this listing.

```
;******************************
;* program : clock  v1.3      *
;* author  : ruediger kerkloh *
;* update  : 04.07.88         *
;******************************

*    = $1000       ;start address
```

```
       .offs $c000-*     ;code at $c000

       ;geos-macros

        desktop = $c22c
        getfile = $c208
     cmpfstring = $c26e
       movedata = $c17e
           main = $849b
      irqvector = $fffe
          count = $ff

        ;look for speeder routine (to keep program
        ;compatible with later GEOS versions)


init   ldx #<daten     ;compare bytes w/
       ldy #>daten     ;speeder-routine
       stx $02
       sty $03

       jsr search
       bne clock        ;found.

       ldx #<name       ;compare bytes
       ldy #>name       ;(text:"clock ")
       stx $02
       sty $03

       jsr search       ;not found
       beq leave

       lda irqvector    ;point to the
       sec              ;program
       sbc #<[go-name]  ;identifier
       sta $02
       lda irqvector+1
       sbc #>[go-name]
       sta $03

       ;* enter speeder head  *

       ldy #5
loop   lda daten,y
       sta ($02),y
       dey
       bpl loop


       ;* enter old irq-vector *

       sei
       ldy #<[oirq-name+1]
       lda ($02),y
       sta irqvector    ;entry
       iny
       lda ($02),y
       sta irqvector+1
       cli

       clc
       bcc leave        ;jump


       ;$04/$05 contains start address
```

```
clock ldx irqvector    ;old value
      ldy irqvector+1
      stx oirq+1
      sty oirq+2

      sei
      lda $04          ;turn around
      clc              ;irq vector
      adc #<[go-name]
      sta irqvector
      lda $05
      adc #>[go-name]
      sta irqvector+1

      ldx #<name         ;copy program
      ldy #>name         ;into kernal
      stx $02
      sty $03

      ldx #<[stop-name] ;nr of bytes
      ldy #>[stop-name]
      stx $06
      sty $07

      jsr movedata     ;copy routine

      cli

      ldx #<fname      ;pointer to
      ldy #>fname      ;filenames
      stx $0e
      sty $0f

      lda #0           ;load normal
      sta $02
      sta $16          ;no ram reserved

      jsr getfile      ;load

leave jmp desktop      ;load desktop


search ;* search routine *

      ldx #$ff         ;start at $9000
      ldy #$8f         ;(increment as
      stx $04          ;needed!)
      sty $05

cnt2  inc $04          ;incr.  pointer
      bne cnt1
      inc $05
      beq fail         ;not found
cnt1  lda #6           ;nr. of bytes
      ldx #$02         ;zeropge pointer
      ldy #$04
      jsr cmpfstring   ;compare
      bne cnt2         ;continue search

      lda #$ff         ;found
fail  rts


      ;* filename for accessory *
      ;("Pref. Manager")
```

```
fname  .byte  $50,$72,$65,$66,$2e,$20,$4d
       .byte  $61,$6e,$61,$67,$65,$72,$00

       ;* compare data *

daten  .byte  $48,$a5,$22,$a2,$ff,$38

       ;* main clock program *

name   .text 'clock ' ;6-byte ident.

go     dec count      ;all 16 irq follow
       beq tiout      ;output only

oirq   jmp $0000      ;(see above)

tiout  pha            ;reserve register
       txa
       pha
       tya
       pha

       ldx #$00       ;reserve $01-$0d
rette  lda $01,x      ;from stack
       pha
       inx
       cpx #$0d
       bne rette

       lda #$35       ;add i/o
       sta $01

       lda #58        ;colon
       sta $08
       sta $0b

       lda $dc0b      ;hours
       and #$1f
       cmp #$12
       bne cnt
       lda #$00       ;zero out clock
cnt    bit $dc0b      ;pm?
       bpl am         ;no
       sed
       clc
       adc #$12       ;else +12 hours
       cld

am     tax
       lda irqvector
       clc
       adc #<[cnt3-go-1]
       tay
       lda irqvector+1
       adc #>[cnt3-go-1]
       pha
       tya
       pha
       txa

       clc
       bcc conv1      ;to conv via conv1

cnt3   stx $06
       sta $07
```

```
       lda irqvector
       clc
       adc #<[cnt4-go-1]
       tay
       lda irqvector+1
       adc #>[cnt4-go-1]
       pha
       tya
       pha

       lda $dc0a     ;minutes

       clc
conv1  bcc conv

rel    bcc oirq      ;jump help

cnt4   stx $09
       sta $0a

       lda irqvector
       clc
       adc #<[cnt5-go-1]
       tay
       lda irqvector+1
       adc #>[cnt5-go-1]
       pha
       tya
       pha

       lda $dc09     ;seconds

       clc
       bcc conv

cnt5   stx $0c
       sta $0d

       lda $dc08     ;release

       ;* character output in bitmap *

       lda #$31      ;combine with char
       sta $01       ;generator
       ldx #$38      ;$a138 is the top
       ldy #$a1      ;address of the
       stx $04       ;second block in
       sty $05       ;the bitmap
       ldx #7        ;character counter

next2  lda #0        ;clr hi (address in
       sta $03       ;character rom)

       lda $06,x     ;screen poke

       asl a         ;times 8
       rol $03
       asl a
       rol $03
       asl a
       rol $03

       sta $02       ;lo

       lda #$d0      ;character set
```

```
       clc            ;starting address
       adc $03
       sta $03

       ldy #7         ;line counter
next1  lda ($02),y
       sta ($04),y
       dey
       bpl next1

                      ;next character
       lda $04
       sec
       sbc #8
       sta $04
       lda $05
       sbc #0
       sta $05

       dex            ;all characters?
       bpl next2

       ldx #$0c       ;$0d-$01 from stack
hole   pla            ;get it
       sta $01,x
       dex
       bpl hole

       lda #$10       ;new starting value
       sta count

       pla            ;return register
       tay
       pla
       tax
       pla

       clc
       bcc rel        ;to geos irq as "rel"

       ;* bcd -> 2 convert
       ;            poke values *

 conv  pha            ;reserve
       and #$f0
       lsr a          ;left half
       lsr a
       lsr a
       lsr a
       clc
       adc #48        ;poke value offset
       tax

       pla            ;right half
       and #$0f
       clc
       adc #48

       rts

stop

.end
```

Here is the same program in BASIC loader form. The DATA statements create the program, icon and info box data.

```
0 rem ********************************
1 rem * program: clock                *
2 rem * author:  ruediger kerkloh     *
3 rem * update:  4/7/88               *
4 rem ********************************
12 open 1,8,15,"i:0":rem error channel
14 restore:print chr$(147):rem clr home
16 rem * checksum computation *
18 for i=1 to 539
20 read a:s=s+a
22 print chr$(19);539-i;chr$(157);" ";
24 next:restore
26 if s<>57427 then print "error in data!":stop
28 rem * save file *
30 f$="clock  v1.3"
32 open 2,8,2,f$+",u,w":gosub 112
34 for i=1 to 426:read a
36 print#2,chr$(a);:next:gosub 112
38 close2
40 rem * insert in info sector *
42 a=1:b=0
44 t=a:s=b:print#1,"b-a:0";t;s
46 input#1,n,t$,a,b:if a=18 then a=19:b=0
48 if n>0 then 44
50 open 3,8,3,"#0":print#3
52 print#1,"u1";3;0;t;s:rem sektor lesen
54 for i=1 to 113:read b:print#3,chr$(b);:next
56 for i=1 to 141
58 print#3,chr$(0);:rem delete rest
60 next
62 print#1,"u2";3;0;t;s:rem write entry
64 rem * in directory *
68 print#3
70 a$=chr$(18):b$=chr$(1):rem directory
72 a=asc(a$):b=asc(b$):print#1,"u1";3;0;a;b
74 get#3,a$,b$:rem next sector
76 for i=0 to 7:rem number of entries
78 print#1,"b-p";3;i*32+5
80 for j=1 to len(f$)
82 get#3,w$:if w$<>mid$(f$,j,1) then k=0:goto 86
84 k=k+1
```

```
86 next j
88 if k=len(f$) then 92:rem found
90 next i:gosub 112:goto 72
92 print" autostart (y/n)";ft$
94 get ft$
96 if ft$="y" then ft=14:goto 102
98 if ft$<>"n" then 94
100 ft=6:rem application
102 print#1,"b-p";3;i*32+21
104 print#3,chr$(t)chr$(s)chr$(0)chr$(ft);
106 print#3,chr$(88)chr$(4)chr$(8)chr$(14)chr$(25)
chr$(2);
108 print#1,"u2";3;0;a;b:rem return to  dir
110 close1:print"o.k.":end
112 input#1,n,t$:if n then print"error: ";t$:stop
114 return
116 rem
118 data162,181,160,16,134,2,132,3,32,137,16,208,5
6,162,187
120 data160,16,134,2,132,3,32,137,16,240,108,173,2
54,255,56
122 data233,6,133,2,173,255,255,233,0,133,3,160,5,
185,181
124 data16,145,2,136,16,248,120,160,11,177,2,141,2
54,255,200
126 data177,2,141,255,255,88,24,144,65,174,254,255
,172,255
128 data255,142,198,16,140,199,16,120,165,4,24,105
,6,141,254
130 data255,165,5,105,0,141,255,255,162,187,160,16
,134,2,132
132 data3,162,239,160,0,134,6,132,7,32,126,193,88,
162,167
134 data160,16,134,14,132,15,169,0,133,2,133,22,32
,8,194,76
136 data44,194,162,255,160,143,134,4,132,5,230,4,2
08,4,230
138 data5,240,13,169,6,162,2,160,4,32,110,194,208,
237,169
140 data255,96,80,114,101,102,46,32,77,97,110,97,1
03,101,114
142 data0,72,165,34,162,255,56,67,76,79,67,75,32,1
98,255,240
144 data3,76,0,0,72,138,72,152,72,162,0,181,1,72,2
```

```
32,224,13
146 data208,248,169,53,133,1,169,58,133,8,133,11,1
73,11,220
148 data41,31,201,18,208,2,169,0,44,11,220,16,5,24
8,24,105
150 data18,216,170,173,254,255,24,105,72,168,173,2
55,255,105
152 data0,72,152,72,138,24,144,23,134,6,133,7,173,
254,255
154 data24,105,99,168,173,255,255,105,0,72,152,72,
173,10,220
156 data24,144,117,144,160,134,9,133,10,173,254,25
5,24,105
158 data124,168,173,255,255,105,0,72,152,72,173,9,
220,24,144
160 data90,134,12,133,13,173,8,220,169,49,133,1,16
2,56,160
162 data161,134,4,132,5,162,7,169,0,133,3,181,6,10
,38,3,10
164 data38,3,10,38,3,133,2,169,208,24,101,3,133,3,
160,7,177
166 data2,145,4,136,16,249,165,4,56,233,8,133,4,16
5,5,233
168 data0,133,5,202,16,207,162,12,104,149,1,202,16
,250,169
170 data16,133,255,104,168,104,170,104,24,144,139,
72,41,240
172 data74,74,74,74,24,105,48,170,104,41,15,24,105
,48,96
176 rem **** data for info sector ***
180 data0,255,3,21,191,255,255,255,128,0,1,128,0,1
,128,0,1
182 data128,0,1,131,255,253,133,85,85,138,170,169,
159,255
184 data245,160,0,21,169,134,85,170,65,85,170,81,8
5,170,70
186 data85,170,84,85,169,134,85,160,0,25,159,255,2
41,128,0
188 data1,128,0,1,255,255,255,131,5,0,0,16,170,17,
0,16,195
190 data76,79,67,75,32,32,32,32,32,32,32,86,49,46,
51,0,0,0
192 data0,210,85,69,68,73,71,69,82,32,203,69,82,75
,76,79,72
```

5.3.4 The foreign aberration in the GEOS KERNAL

We've found a problem in using GEOS with 50 Hz electrical current. If you're using the U.S. standard of 60 Hz current, you can skip this section.

We discovered this problem while working on the clock program in the previous sections. We changed the 50/60 Hz control register for the clock with the help of the single step simulator (which we will discuss in the next chapter). With this we have written two subroutines which work during the initialization of GEOS (each time it is started up, for example with RESET). The first routine is located at $CFA1, and gets the correct time using the standard of the video signals (PAL for example: SECAM) and places it in the control register $DC0E.

The clock will work just as well in Europe as it does in the U.S., just as long as the frequency is set properly. This is where the GEOS operating system problem occurs. It constantly uses location $CD6B as the jumping point to routines. Each time this occurs the machine writes the value for 60 Hz from a table into the control register. The problem is that this has simply been ignored in GEOS.

Apparently the developers of GEOS didn't notice this problem, since all of their machines were running at 60 Hz. However, if you want to display the correct time on a European GEOS, you can perform the following steps: After booting GEOS, go into BASIC, then enter the command `POKE 53132,128` and then start over again with the command `SYS 52298`.

This procedure must be followed before every application that uses the clock. However, this can be very time-consuming—especially changing all of the system diskettes. Therefore, we have written a BASIC program that searches the table in the KERNAL and writes the appropriate value for 50 Hz to the control register.

This is not an actual fix of the GEOS KERNAL, since the output of the video standard is always ignored. However, it's a Quick-n-Dirty fix that displays the correct time when you need it.

Warning: Don't try to run the program discussed unless you read the following information. **It's possible that you'll lose all of your data.** Before you start the program, exit GEOS and make an additional backup copy on a completely blank diskette. This program only works with the European version of GEOS.

Enter only the BASIC program. Make sure that your backup copy—**not the original GEOS diskette**—is in the disk drive. Then start the program.

The computer searches for the correct location on the diskette, and notifies you if a change is made to the diskette. During this procedure, the computer displays the number of blocks searched.

If the program cannot find the correct location, there are a few possible reasons:

1) The program was entered incorrectly (check the program again, or even have a second person proof read the program for you).
2) The diskette is not correct.
3) The copy of the KERNAL is corrupted. (Try making another copy on a different diskette.)
4) There is a hardware problem. (The read/write head could be out of alignment.)

If the program did run correctly, you can copy the results back to the original diskette.

Note: Before you perform any of the following operations, try running the altered version of GEOS to make sure that everything is working properly.

Get the original diskette and remove the write-protect tab. Now boot up your machine with the new (changed) copy of GEOS. Now, using `FILE INFO`, remove the file protection from the program `GEOS KERNAL` and move it to the waste basket. (Did you make a backup copy, just in case?).

Finally, copy the altered KERNAL to the original diskette. Once the program is copied, return the file protection to this file.

Here's a listing of the program:

```
10 PRINT CHR$(147);:  REM CLR HOME
15 OPEN 1,8,15,"I": GOSUB 100
20 OPEN 2,8,2,"GEOS KERNAL":GOSUB 100
25 GET #2,A$,A$,T$,S$:GOSUB100:CLOSE 2
30 OPEN 3,8,3,"#1":GOSUB 100
35 PRINT#1,"U1:"3;0;ASC(T$);ASC(S$)
40 PRINT CHR$(19);B:B=B+1:REM BLOCK COUNTER
45 PRINT#1,"B-P:"3;0
50 GET#3,T$,S$:S$=S$+CHR$(0)
55 RESTORE
60 READ D:IF K=254 THEN K=0:GOTO 35
65 GET#3,A$:A=ASC(A$+CHR$(0)):K=K+1
70 IF A<>D THEN I=0:GOTO 55
75 I=I+1:IF I<4 THEN 60
80 PRINT#3,CHR$(128)
85 PRINT#1,"M-W"CHR$(1);CHR$(0);CHR$(1); CHR$(144)
90 CLOSE1
100 INPUT#1,A,B$:IF A THEN PRINTB$:STOP
105 RETURN
110 DATA 13,220,3,127:REM COMPARISON VALUES
```

If this program's operation interests you, here's a brief description:

Line	Function
25	Get first track / sector of the KERNAL (`A$` is ignored).
30	Open data channel.
35	Read a sector into the channel.
45	Set pointer to 0 byte in channel.
50	Get next sector and track numbers.
60	Get comparison byte out of `DATA` statement. Read all bytes from the sector, then get new sector.
65	Get value from channel: increment byte counter.
70	Compare the byte with `DATA`; if unequal, then no bytes are correct : get the next byte from the channel.
75	Otherwise increment the counter. Continue if there are more bytes .
80	Write the value for 50 Hz to the correct location.
85	Call the routine `Block Write`.
100	Read from the error channel if there were any errors. Output any which might have occurred.

Chapter Six

Inside GEOS

Inside GEOS

In this chapter we'll present information you need to write your own GEOS applications. For instance, we'll tell you how to access GEOS routines for your own purposes. We'll begin with an important GEOS tool.

6.1 The single-step simulator

We developed a *single-step simulator* for investigating the GEOS operating system. This simulator makes it possible to follow the operation of the microprocessor very closely.

Within the simulator, a branch is made to a routine after each machine language instruction, which displays the current contents of the processor registers at the bottom of the screen and waits until a key is pressed. Then it returns to the main program. At the same time, the current contents of any four memory locations can also be displayed. We used this tool primarily for tracing important subroutines in the KERNAL (window routines, alarm evaluation, etc.).

Here's the machine language source code for the SST (Single STep simulator), followed by a BASIC loader version of the program.

```
;***    sst seq ***

 ;author: r kerkloh
 ;update: 03/21/88
 ;addresses reside in disk memory


;*      = $c600    ;start ger v1.2

 *      = $9630    ;start us v1.3

 irqv   = $fffe    ;irq start addr

 via1   = $dc00    ;cia

 talo   = via1+4   ;timer lo
 tahi   = via1+5   ;      hi
 icra   = via1+13  ;irq control
 cra    = via1+14  ;timer control

 vic    = $d000    ;vic start
```

```
      ;*    initialization      *
      ;*  singlestep:  cia 1 a  *

start sei

      lda $01
      pha

      lda #$35      ;$e000 -> ram
      sta $01       ;i/o on

      lda #$2c      ;alarm output
      sta $c0e8     ;save room in
      lda #$ea      ;kernal for
      sta $9067     ;reading icr
      sta $9068
      sta $9069     ;(v1.3 usa)

     ;lda #$2c      ;alarm output
     ;sta $c2d4     ;save room in
     ;lda #$ea      ;kernal for
     ;sta $c4e0     ;reading icr
     ;sta $c4e1
     ;sta $c4e2     ;(v1.2 europe)

      lda #$00      ;save line irq
      sta vic+26

      ldx irqv      ;note original irq
      ldy irqv+1
      stx oirq
      sty oirq+1

      ldx #<irq     ;insert new irq
      ldy #>irq
      stx irqv
      sty irqv+1

      lda #$00
      sta cra       ;timer a: stop

      lda icra      ;clr

      lda #$7f
      sta icra      ;clr mask

      lda #$81      ;set mask: timer a
      sta icra

      ldx #27       ;one step time
      ldy #0        ;27 cycles
      stx talo
      sty tahi

      lda #$09      ;start one shot
      sta cra

      pla
      sta $01

      ldx #$3e
      ldy #$c2
      stx $849b
      sty $849c
      rts
```

```
      ;*** new irq start address ***

irq   sta akku      ;reserve

      lda $01
      sta mem

      lda #$35
      sta $01

wait  lda #$fd      ;test shift key
      sta $dc00
      lda $dc01
      and #$80      ;printed?
      eor #$80      ;(turnover)
      ora flag      ;direct from geos-irq?
      beq contj     ;both not:sst

      jmp contm     ;else abbreviate

contj lda #$7f      ;c= key pressed?
      sta $dc00
      lda $dc01
      and #$20
      beq contk     ;yes -> sst

      lda #200      ;else: max wait
      sta rpeat
      bne wait      ;unconditional jump

contk lda rpeat
      beq contl     ;min wait!
      sec
      sbc #5        ;decrement wait time
      sta rpeat

contl stx xreg      ;reserve x/y registers
      sty yreg

      tsx
      inx           ;correct stack
      inx
      inx
      stx stac

      pla           ;status-register
      sta stat

      pla           ;p-counter
      sta pc+1      ;lo
      pla
      pha
      sta pc        ;hi
      lda pc+1
      pha           ;return

      lda stat
      pha

      ldx #0

      ;* convert register into b code
*
```

```
      lda pc
      jsr asc       ;pc 1st & 2nd places
      sta tab
      sty tab+1

      ldy #0
next  lda reg,y     ;pc lo
      sty pc        ;(free)

      jsr asc

      sta tab+2,x
      inx
      tya
      sta tab+2,x

      ldy pc
      inx
      inx
      iny
      cpy #6
      bne next


      ;set status register

      ldx #7
nexts lsr stat
      bcs set

      lda #46       ;point

      .byte $2c

set   lda #30       ;arrow
      sta tab+17,x
      dex
      bpl nexts

      lda $8507     ;current mouse speed
      jsr asc
      sta tab+26
      sty tab+27

      lda $84b6     ;mouse status
      jsr asc
      sta tab+29
      sty tab+30

      lda $8506     ;joystick value
      jsr asc
      sta tab1+26
      sty tab1+27

      lda $877d     ;number of program jobs
      jsr asc
      sta tab1+29
      sty tab1+30


      ;*** output routine ***

      ldx #5        ;$07-$02 -> stack
nexta lda $02,x
      pha           ;$02/$03  char rom
```

```
      dex           ;$04/$05  bitmap
      bpl nexta     ;$06/$07  table

      lda #$bd
      sta $05       ;bitmap
      lda #$b0      ;top line
      sta $04

      lda #<tab1    ;top table
      sta $06
      lda #>tab1
      sta $07

      jsr out

      lda #$be      ;bottom line
      sta $05
      lda #$f0
      sta $04

      lda #<tab     ;bottom table
      sta $06
      lda #>tab
      sta $07

      jsr out

      ldx #0        ;stack -> $02-$07
nextb pla
      sta $02,x
      inx
      cpx #6
      bne nextb

      ldx rpeat    ;wait
      beq leave    ;min wait !
      ldy #$00
wait1 dey
      bne wait1
      dex
      bne wait1

leave ldx xreg
      ldy yreg

contm lda icra      ;clr

      lda #$09      ;start
      sta cra

      lda #$01      ;geos irq?  ; 2
      and vic+25                ; 4
      sta flag      ;mark it    ; 4
      bne girq                  ; 2

      lda mem                   ; 4
      sta $01                   ; 3

      lda akku                  ; 4

      rti                       ; 6
                               ;---
                                ;29 cycles
```

```
      ;*** for geos irq program ***

girq  lda mem
      sta $01

      lda akku

      jmp (oirq)        ;to geos irq


asc   ;*** convert hex ->to->b code ***

      pha           ;reserve
      jsr code      ;left : accumulator
      tay           ;right: y-reg
      pla
      lsr a
      lsr a
      lsr a
      lsr a
      jsr code
      rts

code  and #$0f
      cld
      sec
      sbc #$0a
      bcs hex
      adc #57
hex   clc
      adc #1
      rts

      ;*** character output ***

out   lda #$33      ;add character
      sta $01       ;generator

      ldy #30       ;char counter

next2 lda #0        ;clr hi
      sta $03       ;(address in char rom)

      lda ($06),y   ;b-code

      asl a         ;times 8
      rol $03
      asl a
      rol $03
      asl a
      rol $03

      sta $02       ;lo

      lda #$d0      ;hi
      clc
      adc $03
      sta $03

      tya
      pha
      ldy #7
next1 lda ($02),y
      sta ($04),y
      dey
```

```
      bpl next1
      pla
      tay

      sec           ;next character
      lda $04
      sbc #8
      sta $04
      lda $05
      sbc #0
      sta $05

      dey           ;all characters?
      bpl next2

      lda #$35
      sta $01

      rts


      ; working memory

flag  * = * + 1     ;fr. geos-irq
mem   * = * + 1
rpeat * = * + 1
pc    * = * + 1     ;hi
reg   * = * + 1     ;lo
akku  * = * + 1
xreg  * = * + 1
yreg  * = * + 1
stac  * = * + 1
stat  * = * + 1
oirq  * = * + 2

      ;display area

tab1  .byte 32,16, 3,32,32, 1, 3,32
      .byte 24,18,32,25,18,32,19,16
      .byte 32,14,22,35, 2, 4, 9,26
      .byte  3,32,32,32,32,32,32

tab   .byte 32,32,32,32,32,32,32,32
      .byte 32,32,32,32,32,32,32,32
      .byte 32,32,32,32,32,32,32,32
      .byte 32,32,32,32,32,32,32

      .end
```

Next, the BASIC loader of the SST. Enter, save and run this program from BASIC. The DATA statements in this BASIC loader program generate the SST program, generate the SST program icon and info box data. If you don't like the icon we've given the program, you can change it from the FILEMASTER program.

```
0 rem ********************************
2 rem * program : sst v1.3           *
4 rem * author  : ruediger kerkloh   *
6 rem * update  : 03/30/88           *
8 rem ********************************
10 :
12 open 1,8,15,"i:0":rem error channel
14 restore:print chr$(147):rem clr home
16 rem * compute checksum *
18 for i=1 to 696
20 read a:s=s+a
22 print chr$(19);696-i;chr$(157);" ";
24 next:restore
26 if s<>74062 then print "error in data stat
ements!":stop
28 rem * save file *
30 f$="sst  v1.3"
32 open 2,8,2,f$+",u,w":gosub 104
34 for i=1 to 582:read a
36 print#2,chr$(a);:next:gosub 104
38 close2
40 rem * insert info sector *
42 a=1:b=0
44 t=a:s=b:print#1,"b-a:0";t;s
46 input#1,n,t$,a,b:if a=18 then a=19:b=0
48 if n>0 then 44
50 open 3,8,3,"#0":print#3
52 print#1,"u1";3;0;t;s:rem read sector
54 fori=583 to 696:read b:print#3,chr$(b);:ne
xt
56 fori=1 to 142
58 print#3,chr$(0);:rem delete rest
60 next
62 print#1,"u2";3;0;t;s:rem write
64 rem * to directory *
66
```

```
68 print#3
70 a$=chr$(18):b$=chr$(1):rem directory
72 a=asc(a$):b=asc(b$):print#1,"u1";3;0;a;b
74 get#3,a$,b$:rem next sector
76 for i=0 to 7:rem number of entries
78 print#1,"b-p";3;i*32+5
80 for j=1 to len(f$)
82 get#3,w$:if w$<>mid$(f$,j,1) then k=0:goto
 86
84 k=k+1
86 next j
88 if k=len(f$) then 92:rem found
90 next i:gosub 104:goto 72
92 print#1,"b-p";3;i*32+21
94 print#3,chr$(t)chr$(s)chr$(0)chr$(5);
96 print#3,chr$(88)chr$(4)chr$(2)chr$(13)chr$
(15)chr$(4);
98 print#1,"u2";3;0;a;b:rem dir return
100 close1:print"o.k.":end
102 rem ******************************
104 input#1,n,t$:if n then print"error: ";t$:
stop
106 return
108 rem
110 data120,165,1,72,169,53,133,1,169,44,141,
232
112 data192,169,234,141,103,144,141,104,144,1
41
114 data105,144,169,0,141,26,208,174,254,255,
172
116 data255,255,142,54,152,140,55,152,162,146
118 data160,150,142,254,255,140,255,255,169,0
120 data141,14,220,173,13,220,169,127,141,13,
220
122 data169,129,141,13,220,162,27,160,0,142,4
124 data220,140,5,220,169,9,141,14,220,104,13
3
126 data1,162,62,160,194,142,155,132,140,156,
132
128 data96,141,49,152,165,1,141,45,152,169,53
130 data133,1,169,253,141,0,220,173,1,220,41,
128
```

```
132 data73,128,13,44,152,240,3,76,170,151,169
134 data127,141,0,220,173,1,220,41,32,240,7,1
69
136 data200,141,46,152,208,217,173,46,152,240
138 data6,56,233,5,141,46,152,142,50,152,140,
51
140 data152,186,232,232,232,142,52,152,104,14
1
142 data53,152,104,141,48,152,104,72,141,47,1
52
144 data173,48,152,72,173,53,152,72,162,0,173
146 data47,152,32,208,151,141,87,152,140,88,1
52
148 data160,0,185,48,152,140,47,152,32,208,15
1
150 data157,89,152,232,152,157,89,152,172,47,
152
152 data232,232,200,192,6,208,229,162,7,78,53
154 data152,176,3,169,46,44,169,30,157,104,15
2
156 data202,16,240,173,7,133,32,208,151,141,1
13
158 data152,140,114,152,173,182,132,32,208,15
1
160 data141,116,152,140,117,152,173,6,133,32,
208
162 data151,141,82,152,140,83,152,173,125,135
164 data32,208,151,141,85,152,140,86,152,162,
5
166 data181,2,72,202,16,250,169,189,133,5,169
168 data176,133,4,169,56,133,6,169,152,133,7,
32
170 data236,151,169,190,133,5,169,240,133,4,1
69
172 data87,133,6,169,152,133,7,32,236,151,162
174 data0,104,149,2,232,224,6,208,248,174,46,
152
176 data240,8,160,0,136,208,253,202,208,250,1
74
178 data50,152,172,51,152,173,13,220,169,9,14
1
180 data14,220,169,1,45,25,208,141,44,152,208
182 data9,173,45,152,133,1,173,49,152,64,173,
45
```

```
184 data152,133,1,173,49,152,108,54,152,72,32
186 data222,151,168,104,74,74,74,74,32,222,15
1
188 data96,41,15,216,56,233,10,176,2,105,57,2
4
190 data105,1,96,169,51,133,1,160,30,169,0,13
3
192 data3,177,6,10,38,3,10,38,3,10,38,3,133,2
194 data169,208,24,101,3,133,3,152,72,160,7,1
77
196 data2,145,4,136,16,249,104,168,56,165,4,2
33
198 data8,133,4,165,5,233,0,133,5,136,16,203,
169
200 data53,133,1,96,0,0,0,0,1,0,0,0,0,0,0,0,3
2
202 data16,3,32,32,1,3,32,24,18,32,25,18,32,1
9
204 data16,32,14,22,35,2,4,9,26,3,32,32,32,32
206 data32,32,32,32,32,32,32,32,32,32,32,32,3
2
208 data32,32,32,32,32,32,32,32,32,32,32,32,3
2
210 data32,32,32,32,32,32,32
212 rem * info sector *
214 data0,255,3,21,191,255,255,255,128,0,1,14
3
216 data255,241,208,0,9,84,113,235,84,73,42,2
12
218 data73,42,148,73,235,148,73,41,151,113,41
220 data144,0,9,143,255,241,128,0,1,185,207,2
49
222 data169,72,9,185,203,233,128,8,137,144,13
6
224 data9,185,207,249,128,0,1,255,255,255,130
226 data5,0,48,150,49,150,48,150,83,83,84,32,
86
228 data49,46,51,32,85,83,65,0,32,65,13,32,32
230 data32,32,210,85,69,68,73,71,69,82,32,203
232 data69,82,75,76,79,72,0
```

Now for a few remarks about how the program works.

In the initialization, the SST automatically copies itself to disk memory at $9630. It is not possible to load directly to this address because under GEOS, the area in which a program is loaded is recorded on the diskette. The operating system writes the resulting file (SWAP file) at this location after the end of the program. Since we want to exit our program without having it overwritten by the SWAP file, we copy it to an area that will not be overwritten. The computer can crash when the single-step simulator is loaded if actions are performed which require a large amount of memory (such as drawing a filled circle in geoPaint). Because of the limited memory space, this could not be done differently.

The actual operating principle is based on the interrupt technique. A timer in the Complex Interface Adapter (CIA), located at $DC00, is programmed in the initialization so that the microprocessor can execute only one instruction in the main program before it is interrupted and a branch is made to an SST routine.

Here we have to evaluate the cause of the interrupt because GEOS also works with program interruptions in order to read the keyboard or joystick. This is done by reading the Interrupt Control Register of the video chip. GEOS does not create the system interrupt through a timer (jiffy clock) as in the original '64 operating system, but by evaluating the raster line in which the electron beam of the monitor is scanned. The video chip can generate an IRQ every time a given line of the screen is written. The system IRQ is given priority in the SST so that inputs can be made via the keyboard or input device.

If the Commodore (C=) key is pressed, the current register contents are taken from the stack and converted to codes which can be accessed easily from the old '64 character set (screen codes). The output is displayed using the high-resolution graphics of GEOS, first the top line with the titles, and then the register contents. The timer is started again after the output and control returns to the interrupted program.

To initialize GEOS properly, run another GEOS accessory before running SST. Then under the **geos** menu click the item `single step`. After loading, it may seem that the computer has crashed, because no mouse movements are possible. This is not the case. The computer is just waiting for you to press the Commodore (C=) key.

Your screen will look something like this when you press the Commodore (C=) key:

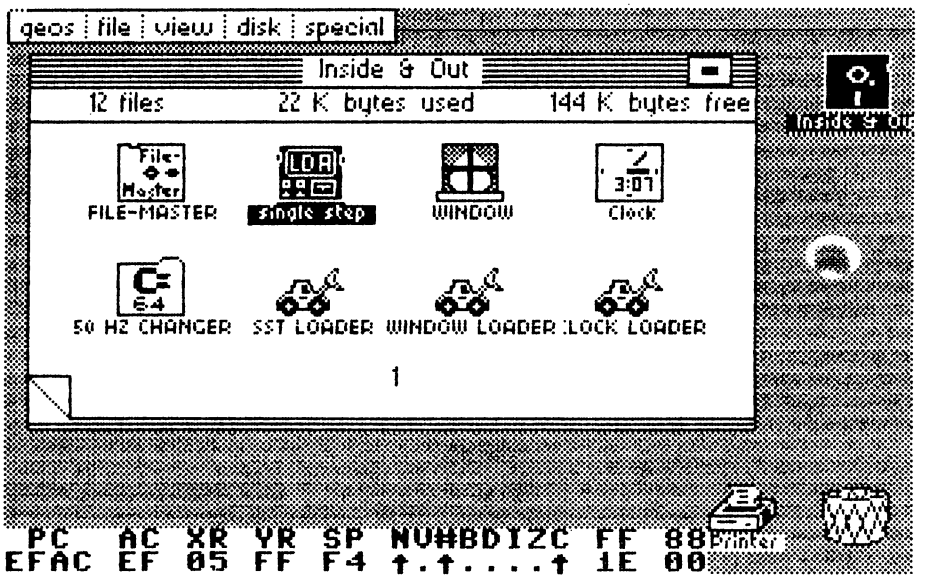


Figure 75: The Stepper

A two-line message is displayed at the bottom of the screen. The top line contains the names of the register of the 6510 microprocessor. PC stands for program counter, AC for accumulator, XR and YR for X and Y registers, SP for the stack pointers. The flags of the status register are designated as follows:

N	negative
V	overflow
B	BCD mode
#	unused (always set)
I	interrupt
Z	zero
C	carry

The bottom line displays the current register contents following the last executed instruction. The output is displayed in hexadecimal, except for the status register. For the status register, an arrow represents a set flag and a period represents a cleared flag.

In addition, the contents of the four memory locations (which you specify before the program started) are displayed at the lower left corner of the screen: $8507 (mouse speed), $84B6 (mouse edge contact), $8506 (input device value) and $877D (the current program job number).

Each time you now press the C= key, the single instruction corresponding to the program counter address is executed. Then the display is refreshed. If you depress the C= key for more than two seconds, multiple instructions are executed, so that you can process more than a single instruction without having to release the key.

You can press the <SHIFT> key to execute the instructions at an even faster rate. The <SHIFT LOCK> key can also be used for extended periods of time.

Using the SST, GEOS operates at a much slower rate than normal.

Press and hold the C= key. You'll see a number of memory locations run past. Many of these locations are part of the GEOS *job loop*. The GEOS job loop controls all basic GEOS actions (e.g., time and date, input device reading).

These memory locations within the job loop vary between versions of GEOS. To keep applications and other programs compatible with every version of GEOS, applications are often written using the GEOS *jump table*. Jump tables are a set of universal addresses which tell a program to "jump" to another address. For example, the GEOS job loop address in the jump table is at memory location $C1C3. GEOS V1.2 transfers a machine language call to this address to memory location $C2C8, for example. GEOS V1.3 uses different jump table with different target addresses, but the main jump table addresses stay the same for all versions.

The job loop controls the reading of the input device used in GEOS. Take your input device. Press and hold the C= key while you move the pointer around the screen. The pointer moves jerkily at first, then the action smooths out as you continue to hold down the C= key. Watch the addresses change (you'll see certain ones repeat themselves). Release the C= key and press it down again. Watch the addresses.

Follow the program flow with the SST without moving the joystick. The routine is processed in large jumps because of branch instructions in the routine named `set` (see page 206 of this book for this routine).

Release the C= key. Hold down the <SHIFT> key and move the pointer onto the close box. Press and hold the C= key and click on the close box.

You'll see the window slowly erase itself (press the <SHIFT> key if you want the routine to move faster without seeing the memory locations displayed). As the window disappears, you'll notice that the same memory locations appear again and again. These locations control the erasing and redrawing of the window.

You'll eventuallly see an empty window and the question-marked disk icon. Click on this diskette icon as you hold down the C= key. The diskette re-opens, and the SST shows the memory locations it uses to draw the window and icons.

The SST is a useful tool for observing what GEOS does and when it executes certain routines.

6.2 Window techniques

You've probably noticed that all of the operating system messages in GEOS, such as disk drive error messages, are displayed in their own windows known as error boxes.

You can easily induce an error box by opening the disk drive door and then trying to load a program from the GEOS diskette. This produces an error box in the center of the screen:

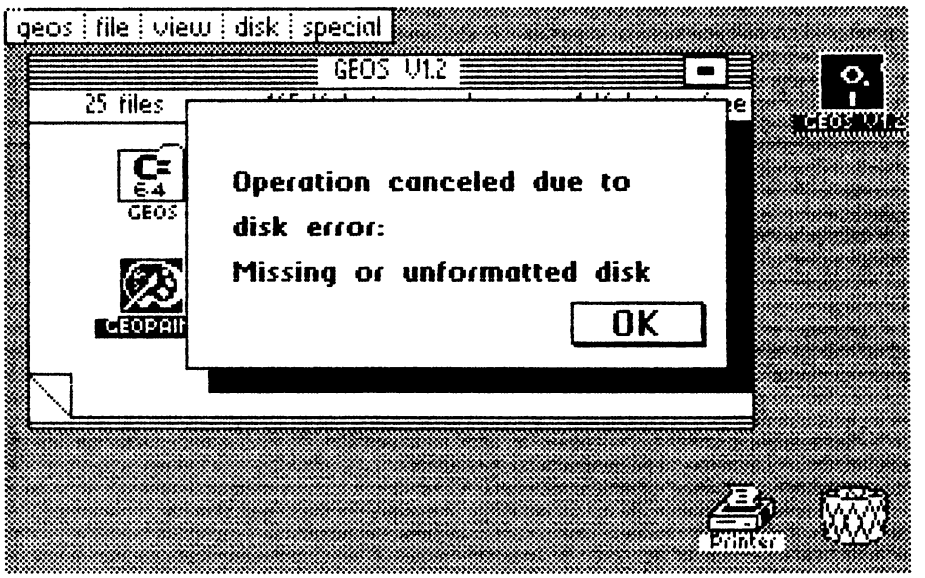


Figure 76: An output window: error box

Correct the error by closing the door, and click OK. The window disappears and the screen contents are the same as before the error box.

GEOS also uses windows known as dialogue boxes, in which a cursor appears, and allows you to enter characters from the keyboard. Select the item `rename` from the **`disk`** menu. This will produce the following window:



Figure 77: Dialogue box

The name of the current diskette is displayed in the lower section. It can be changed from the keyboard. If you change your mind and decide that you want to keep the old name, you can terminate the process with CANCEL, even if you have altered the name already.

You can do this only if you have not pressed the <RETURN> key. This type of dialogue between the user and computer is used throughout GEOS. We have documented this technique so that you can make use of this form of communication in your own programs.

6.2.1 Characteristics of windows

In the introduction you learned what windows are and how they can be used. This section details the general rules that must be followed when you make your own windows.

The first two examples illustrate that the original contents of the screen (the icons of the disk directory) are restored after the window disappears. It should become clear to you that the windowing technique consists of nothing more than overwriting certain areas in screen memory with the data that represent the window.

This technique erases the previous memory contents—and therefore the screen contents as well. To be able to reconstruct the screen after the window disappears, the original contents must be saved elsewhere.

In principle there are two ways for the computer to reconstruct the area of the screen covered by the window:

1. Save a complete copy of the current screen into a protected area of memory.

   When the picture is then regenerated, the entire screen copy is then copied back in to screen memory. The disadvantage of this technique is the excessive memory space required to save the copy.

2. The area that is overwritten by the window in screen memory is copied to a protected area before the window is displayed.

   The original picture is then reconstructed with this "patch", instead of complete window regeneration:

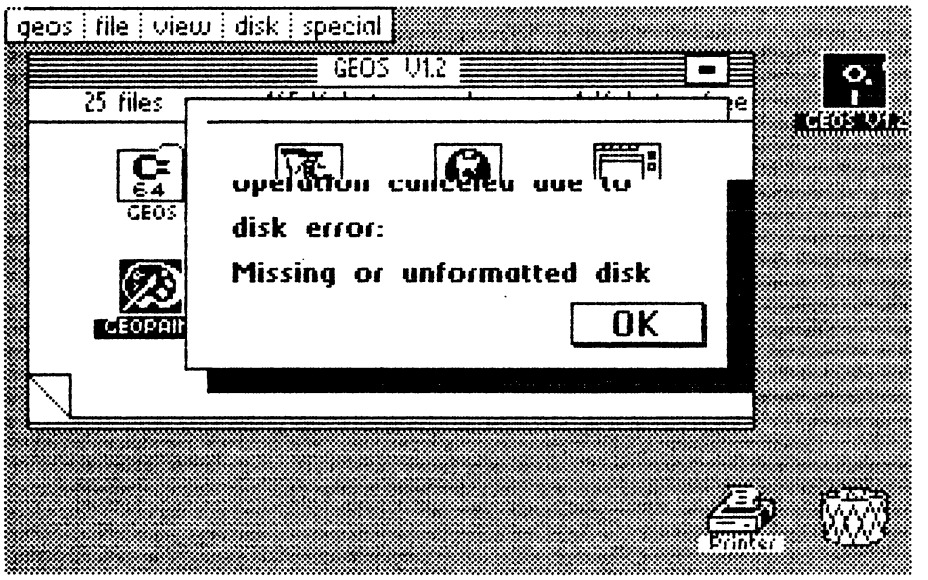


Figure 78: Re-drawing

You must be able to change the size of the window to fit the task. It wouldn't make sense to continuously display the current time in a window that takes up half the screen. Nor would it be appropriate to display important operating system messages in a tiny window where it would hardly be noticed.

GEOS is extremely flexible in this respect. You can use an appropriately sized window for all your specific applications.

In most cases, windows are used to pass information to the user in the form of text. Therefore there must be a way to pass text to the program that creates the window. In addition, we must specify where within the window the text is to appear. This allows you to do things like display very short strings in the center of large windows without having to use blank spaces or carriage returns to center the text.

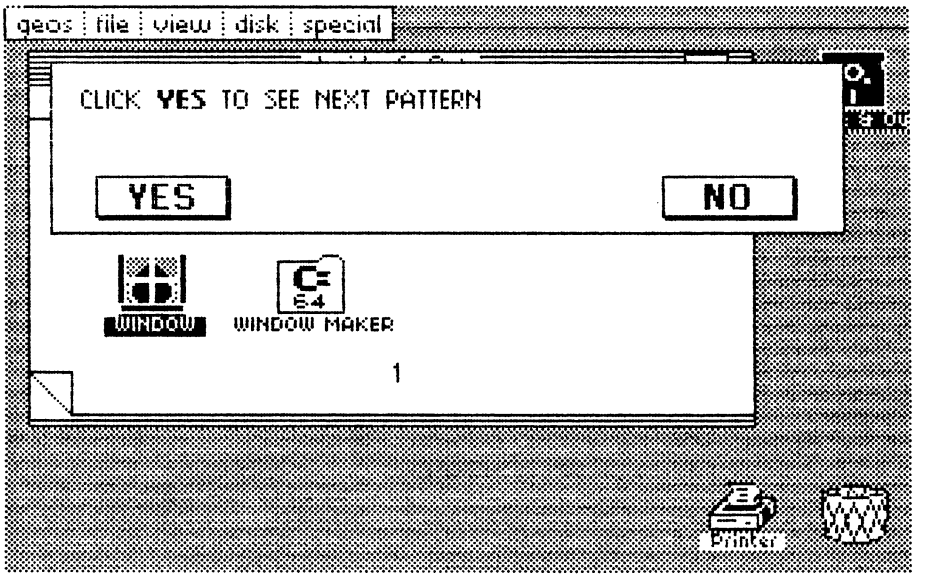


Figure 79: Odd-shaped window

You can also select the style of text. You can use this feature to make the information in your window more concise and clear to the user.



Figure 80: Different type styles

The routine must know where the text is to appear in the window. In addition, the length of the text must be checked to ensure that it doesn't spill out of the window.

Look back at the the first error box that you caused to be displayed (click the disk icon with the drive door open). The window is bordered on the right and the bottom. This border adds depth to the window and sets it out from the rest of the screen. We refer to this as a *shadowed window*.

When making a window, you must specify either a shadowed or shadowless window. You can also select a pattern for the shadow:

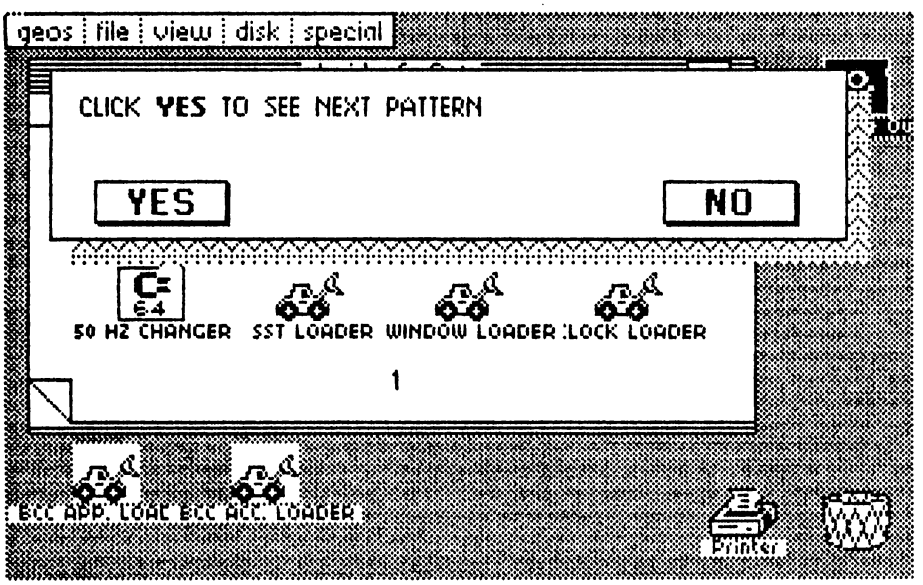


Figure 81: Patterned shadow

Another parameter specifies the allowed responses. In this example, the program waits until you click the OK button.

Some windows do not allow any fields to be clicked with the mouse—for example, GEOS system information boxes. You simply press the mouse button.

Still other windows offer you two or three choices:



Figure 82: Two-button dialogue box

This type of window,the dialogue box, allows us to evaluate one of the two choices. Clicking the OK button executes a specified procedure, while CANCEL closes the window.

So GEOS provides many of the tools that can help you program windows easily.

6.2.2 Custom windows in GEOS

Now we'll actually see how to make custom windows using the GEOS routines and a minimum of programming effort.

1) What routines are available?

Surprisingly, only one GEOS routine is of interest to us for customizing windows. This routine addresses all of the window characteristics mentioned in the previous section. The routine is located in KERNAL RAM from $F1BB to $F23E. When writing your own routines, it is much wiser to use the GEOS KERNAL jump table to address this routine. The routine could change in a newer revision but the KERNAL jump table would still point to the right routine. The KERNAL jump vector used for the window is located at $C256.

With this single routine, you won't have to be concerned with how to save the screen contents before outputting a window. Also, you won't have to be concerned with window management.

The only element you have to furnish to run your program is a pointer to a table that contains the window specifications. We'll take a closer look at this pointer in the next section.

2) The parameters

Let's say that you want to display a message in a window.

To use the window management routine, you have to pass the appropriate parameters to GEOS. The following information is required for windowing:

1. window size (X-direction)
2. window size (Y-direction)
3. window position on the screen
4. parameter for the shadowed window
5. parameter for the output text
6. parameter for the data input
7. exit criteria (OK, CANCEL, etc.)

Before the window routine is called, the starting address of this table is saved in the zero page at $02 and $03.

Example: The table of parameters is at $1234. To display the window, proceed as follows:

```
LDX #$34    low byte of your table
LDY #$12    high byte of your table
STX $02     pass the table pointer
STY $03     to $02 and $03
JSR $C256   call window routine
```

You are responsible for making sure that the table contains valid parameters. Now let's take a look at these parameters in more detail.

3) The job codes

Job codes are commands that direct a GEOS evaluating routine to perform a specific action.

Some of the job codes require additional information. The job codes are stored in a table. Any required data is inserted in the table following the job code. The end of the table is marked with a zero.

4) The format of the window

The first position in the table is a job code that does not require additional data. Therefore it is a one-byte command. This job code specifies the window format.

The individual bits have the following significance:

Byte 1:

Bit 7=1: The window has the same format as the operating system boxes (error messages, etc.). It is displayed on the screen at the same location.

Bit 7=0: The format and position of the window is selected by the user. If this is the case, the following five bytes of the table contain the exact specifications. Otherwise, these five bytes are omitted and the next job code follows immediately.

Bits 0-4: If all of these bits are equal to zero, no shadow is displayed. Other combinations affect the pattern of the shadow (32 possible combinations).

The following bytes are necessary only if the seventh bit of the job code (first byte) is zero. They determine the position (in pixels) of the top and bottom borders of the window on the screen:

Byte 2:	top border
Byte 3:	bottom border

Values may range from 0 to 199. A 0 represents the top edge of the screen, and a 199 represents the bottom edge of the screen. The value of Byte 3 must always be greater than the value of Byte 2.

Byte 4:	left border (low byte)
Byte 5:	left border (high byte)
Byte 6:	right border (low byte)
Byte 7:	right border (high byte)

Two bytes are required to represent each border position. Values may range from 0 to 319. A 0 represents the left edge of the screen. A 319 represents the right edge of the screen The value of Bytes 6 and 7 must be greater than the value of Bytes 4 and 5.

These parameters determine both the size of the window and its position on the screen.

The following table demonstrates this:

```
1st byte: #$01 (non-standard window with solid shadow)
2nd byte: #$14 (top border)
3rd byte: #$50 (bottom border)
4th byte: #$10 (low byte of left border)
5th byte: #$00 (high byte of left border)
6th byte: #$28 (low byte of right border)
6th byte: #$01 (high byte of right border)
7th byte: #$00 (table terminator)
```

This creates a solid shadowed window starting at point (20, 16) extending to (80, 296).

5) Pure text output

Job codes $0B and $0C are used to display text (strings). For job code $0B, four bytes of data follow:

1. # pixels from left border to text
2. # pixels from top window border to text
3. address of text (low byte)
4. address of text (high byte)

Example:

The text is located at $5678. A value of zero terminates the text string. It is to be displayed in a standard window with a shadow.

The table looks as follows:

```
Byte 1: #$81 (standard window with shadow)
Byte 2: #$0B (job code for text output)
Byte 3: #$0A (offset from left border) - 10 pixels
Byte 4: #$10 (offset from top) - 16 pixels
Byte 5: #$78 (low byte of text address)
Byte 6: #$56 (high byte of text address)
Byte 7: #$00 (table terminator)
```

Then place the starting address of the table ($5678) in $02/$03, and call the window routine at $C256. The following picture results:

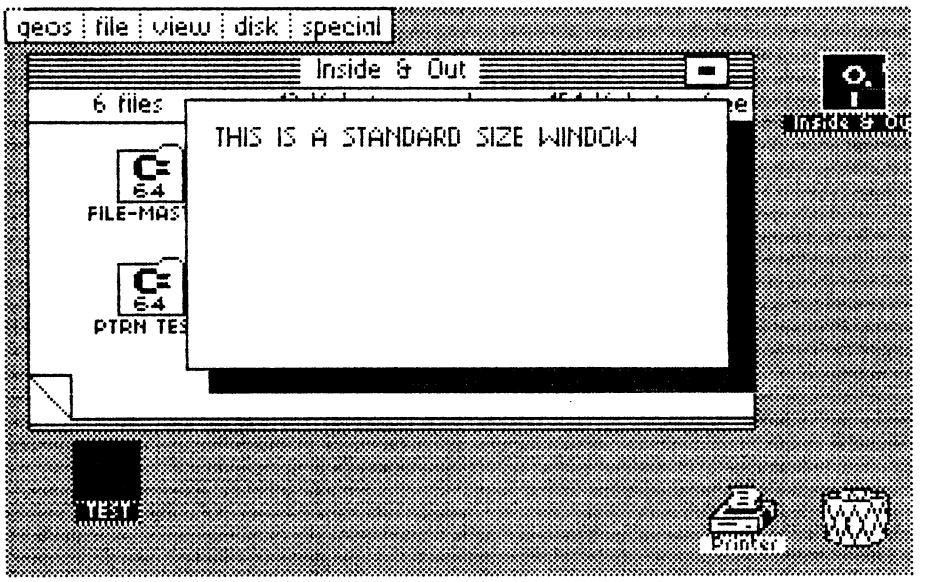


Figure 83: Custom window

The second job code $0C differs in that instead of specifying the absolute address of the text, you specify a pointer in the zero page that contains the actual text address. This involves indirect addressing, only three bytes are required (offset from left edge, offset from top, pointer to zero page).

Example:

Two different text strings are located in memory at address $1234 and $5678, respectively. One of them is to be displayed by means of indirect address via $0C/$0D. The following table must be set up to do this:

```
Byte 1: #$81 (standard window with shadow)
Byte 2: #$0C (job code: indirect addressing)
Byte 3: #$0A (offset from left edge)
Byte 4: #$0A (offset from right edge)
Byte 5: #$0C ($0C/$0D is a pointer to the actual
             address of the text)
Byte 6: #$00 (end of table)
```

Before the window routine is called, the pointer in zero page ($0C/$0D) must contain the starting address of the text string. To output the second text string, just replace the pointer ($0C/$0D) with the location of the new text and call the routine again.

This method allows you to display different texts very quickly using the same table and routines in your program. All you have to do is place the starting address of the text in the zero page addresses specified in the table.

6) Keyboard input

To position a cursor within the window so that the user can enter data from the keyboard, use job code $0D. The evaluating routine then expects four more bytes of data:

Bytes 1 and 2: Distance of the cursor from the left and top edges of the window.

Byte 3: Contains a pointer to an address in the zero page which contains the start address of a buffer in which the data entered will be stored.

Byte 4: Determines the maximum number of characters that can be entered.

Example:

Here we'll create a window that accepts a filename (maximum of eight characters) from the keyboard and stores them at $9876. The table is placed at $1234.

You table could then be constructed similarly to this:

```
Byte 1: #$81 (standard window)
Byte 2: #$0D (job code for input)
Byte 3: #$05 (5 pixels from the left edge)
Byte 4: #$0A (10 pixels from the top)
Byte 5: #$0E (buffer pointer at $0E)
Byte 6: #$08 (maximum input of 8 characters)
Byte 7: #$00 (end of table)
```

The routine is called the same way as for text output. To display the window, proceed as follows:

```
LDX #$76    (low byte of buffer)
LDY #$98    (high byte of buffer)
STX $0E     (pass the buffer pointer)
STY $0F     (to $0E and $0F)
LDX #$34    (low byte of table)
LDY #$12    (high byte of table)
STX $02     (pass the table pointer
STY $03     to $02 and $03)
JSR $C256   (call window routine)
```

This window can also display a default text string which can then be edited. If you don't need this, just set the first value of the buffer to zero before the output.

The routine places the text string you entered into the buffer after you press the <RETURN> key.

Format of text strings

Text strings are composed of the standard Commodore ASCII codes found in the '64 User's Guide. The graphics symbols cannot be used. The important point is that the text strings must always be terminated with a zero byte. With this method, the length of the string doesn't need to be stored.

One special feature is the control codes, which you can use to change the appearance of your text. The following table contains the possible values and the type styles to which they correspond. These styles can also be mixed (such as bold and italics).

Control code	Type style
$18	bold **text**
$19	italics *text*
$1A	outline text
$1B	standard text

(**Note**: Values other than those listed here can lead to a system crash!)

The exit criteria

You have no doubt guessed that there are one or more routines in the GEOS KERNAL which must be called with appropriate parameters to make the window disappear again and restore the previous screen contents. Additionally these routines test the joystick to determine if the mouse is over the OK or CANCEL when the button is pressed.

The nice part about it is that GEOS takes care of almost all of the work for us, and we don't have to know more about these routines.

The GEOS window disappears when a button within that window is clicked or when the <RETURN> key is pressed (only during input).

To add buttons to your own windows (OK, CANCEL, YES, etc.), simply expand your window parameter table. To do this you first enter the job code for the desired button, followed by two bytes for the location within the window where you want it placed.

Here are the job codes:

Job code:	Button:
$01	-OK-
$02	-CANCEL-
$03	-YES-
$04	-NO-
$05	-OPEN-

For example, we want to place a cancel button at point 16,64 ($10,$40) within our window. We would add the following to our window table:

```
Byte 1: #$02 (cancel button)
Byte 2: #$02 (place at 2x8 pixels over and
Byte 3: #$40  #$40 pixels down)
```

Note: When placing buttons in windows, the x direction is multiplied by 8. We want the button 16 pixels over, so we divide 16 by 8. Therefore, the second byte is 2.

Next we'll discuss how to read the button that was clicked.

Evaluation of the buttons

To determine which button was pressed, GEOS uses memory location $02 in the zero page. After the window disappears, the value found in memory location $02 corresponds to the job code of the button pressed.

Example:

You have displayed a window that contains the buttons OK, CANCEL, and NO. The window disappears after the user clicks the NO button and control returns to your program. Memory location $02 now contains the job code for NO ($04), and uses this information to perform the corresponding program action.

If you create a dialogue box (input window) that was terminated by the <RETURN> key, the code returned is $1B (13 decimal).

6.2.3 Three examples from GEOS

Following are three examples from the GEOS deskTop that illustrate the complete process of window operation. We'll use two routines which you have encountered already in the introduction.

The first displays an error message originating from the disk drive operation, and the second changes the name of the diskette. This involves text display and keyboard input. The third example is a program that illustrates some of the window techniques you just learned about.

If you want to follow us "live" in our invasion of the operating system, go back to BASIC using `special` and load a monitor into the area $6000 (we assume that you have a monitor, because you've followed us this far!). This overwrites a copy of the bitmap screen. If you'd rather keep the bitmap screen intact, try the EDMON program which appears in ***GEOS Tricks & Tips*** from Abacus.

GEOS can be reactivated from the monitor or from BASIC by calling the jump table routine at $C22C (decimal 49708). This reloads the deskTop, which shortens the initialization time considerably. You can even make changes to the KERNAL first and then observe their effects.

A `SAVE` operation may not precede the restart, or the disk drive will not operate. One small disadvantage of this restart is that the screen and cursor colors are changed, unless the computer finds the `preferences` file on diskette. This can be corrected with the Preference Manager, of course.

Note: The following listings are from the American GEOS V1.2. Later and European versions will be different. The GEOS jump table should always be correct. The actual locations of the subroutines may be a few bytes off. High quality machine language monitors often have a FIND or HUNT function to help you find routines in different memory areas.

Now on to our first example.

The disk error evaluation routine

When you look at the area starting from $1601 to $1640 with the monitor, you will discover the following routine:

```
$1601 CPX #$0C
$1603 BEQ $1640
$1605 TXA
$1606 BEQ $1640
$1608 LDA $1653,X
$160B STA $0C
$160D LDA $1660,X
$1610 STA $0D
$1612 CPX #$0E
$1614 BCC $1635
$1616 LDA #$38
$1618 STA #$0D
$161A LDA #$D2
$161C STA $0C
$161E TXA
$161F SEC
$1620 SBC #$20
$1622 BMI $1635
$1624 TAX
$1625 CPX #$0E
$1627 BEQ #162B
$1629 BCS $1635
$162B LDA $166E,X
$162E STA $0C
$1630 LDA $167D,X
$1633 STA $0D
$1635 LDA #$16
$1637 STA $03
$1639 LDA #$41
$163B STA $02
$163D JSR $C256
$1640 RTS
```

This is the output routine for the disk drive error messages. Similar to the old C-64 KERNAL, GEOS has a table containing nothing but jump commands. In Version 1.2, $C256 jumps to the routine at $F1BB. Future versions of GEOS may not locate the window management routine at $F1BB, but a jump to $C256 will always have the desired effect—manage a window.

Let's go through the routine step by step.

The first action compares the X register of the processor with decimal 12. If the two match, the routine is exited immediately. As will be shown later, the X register contains the number of the current error message. 12 indicates that no error occurred, as does a zero, which is tested in the fourth line.

For other error numbers, a value is fetched from two tables, one at $1653 and the other at $1660, which is then placed in $0C and $0D. The computer gets the address of the message belonging to the error number in the zero page. We can already guess that the job code $0C will appear in the parameter table for the window. If the error number is less than 14, execution passes directly on to the transmission of the table address for the parameters at $1635 and the window output.

For larger error numbers, the address of the message to output is set to $38D2. This corresponds to the message "I:24". In addition, the error number is compared with 32. If it is smaller, execution continues as above at $1635.

Otherwise a test is made to see if the error number is larger than 45 ($0E plus $20). This would display "I:24".

If the number is between 32 and 45, the addresses in $0C/$0D are taken from another table, which lies at $166E or $167D. Shortly before the window routine is called, the pointer for the parameter table is placed in $02/$03. In this case the parameter table is at $1641 and looks as follows:

```
Byte  1: $1641 #$81 (job code for standard window)

Byte  2: $1642 #$0B (job code for text output)
Byte  3: $1643 #$10 (distance from left edge)
Byte  4: $1644 #$20 (distance from top edge)
Byte  5: $1645 #$1A (low byte of the text address)
Byte  6: $1646 #$38 (high-byte)

Byte  7: $1647 #$0B (job code for another text output)
Byte  8: $1648 #$10 (distance from left edge)
Byte  9: $1649 #$30 (distance from top edge)
Byte 10: $164A #$35 (low byte of the second text)
Byte 11: $164B #$38 (high byte)

Byte 12: $164C #$0C (job code for indirect text output)
Byte 13: $164D #$10 (distance from left edge)
Byte 14: $164E #$40 (distance from top edge)
```

```
Byte 15: $164F #$0C (text address is in $0C/$0D)

Byte 16: $1650 #$01 (display OK button)
Byte 17: $1651 #$11 (distance from left edge)
Byte 18: $1652 #$48 (distance from top edge)

Byte 19: $1653 #$00 (end of table)
```

A total of three different text areas are displayed: The first two are addressed in the absolute mode, while the third uses indirect addressing. Ther error message consists of three lines.

The job codes $0B display a standard text string which always appears (`Operation cancelled due to disk error:`). The current message is displayed with $0C, as we guessed earlier.

The `OK` button is positioned in the window so that the program will continue after the error is acknowledged. It is not necessary to read memory location $02 here because there is only one button can be pressed.

When an error occurs, the error number is placed in the X-register and the routine above is called. It then outputs the corresponding message and returns to the calling routine after the `OK` button is clicked.

We now come to our second example.

The Rename routine

It's possible to rename the current work diskette from the deskTop. We have documented the Rename routine to illustrate inputting text from a window.

The operation of this routine closely resembles the operation of the first example. It simply contains a few more commands to organize text input:

```
$0D43 LDA #$4A   (the current disk name is at $4A90)
$0D45 STA $0B    (address to $0A/$0B)
$0D47 LDA #$90
$0D49 STA $0A

$0D4B LDA #$41   (put a copy of the name in $41DF)
$0D4D STA $0F
$0D4F LDA #$6F
$0D51 STA $0E

$0D53 LDX #$0A   (parameters for the flexible copy
                 routine, which is at $3FA8)
$0D55 LDY #$0E   (the pointers can be selected as
                  desired)
$0D57 LDA #$10   (number of elements to copy)
$0D59 JSR $3FA6  (call the copy routine)
$0D5C LDA #$0D   (parameters for the
$0D5E STA $03     window lie at
$0D60 LDA #$B1    $0DB1)
$0D62 STA $02

$0D64 JSR $C256  (output window)

$0D67 LDA $02    (button evaluation:)
$0D69 CMP #$02   (was CANCEL clicked?)
$0D6B BEQ $0DB0  (yes, terminate)

$0D6D LDA $416F  (is a valid name in the buffer?)
$0D70 BEQ $0D5C  (no, repeat process)

$0D72 JSR $1F21  (else store new name)
```

The subroutine at $3FA8 is called to make a copy of a specific memory area. The pointers (in which the source and destination addresses are stored) are passed in the X and Y registers. In this case the source text address is in $0A/$0B and the destination address is in $0E/$0F. The accumlator contains the number of elements to be copied.

```
$3FA6 STX $3FB4    set source pointer
$3FA9 STY $3FB6    set destination pointer
$3FAC STY $3FC8
$3FAF STA $20      save number of elements
$3FB1 LDY #$00     position to first value
$3FB3 LDA (...),Y  copy
$3FB5 STA (...),Y
$3FB7 CMP #$A0     shift space?
$3FB9 BEQ $3FBD    yes, don't count
$3FBB STY $21      else save number
$3FBD INY          and increment
$3FBE DEC $20      all characters copied?
$3FC0 BNE $3FB3    no, keep copying
$3FC2 LDY $21      increment number of valid characters
$3FC4 INY
$3FC5 LDA #$00     mark the end of the copy
$3FC7 STA (...),Y
$3FC9 RTS
```

A copy of the current diskette name is made. This allows the renaming process to be terminated at any time, even after changes have been made to the name, by clicking CANCEL. The address of the window's parameter table is passed in $02/$03. The table looks as follows:

```
$0DB1 #$81 (job code for standard window)

$0DB2 #$0B (job code for text output)
$0DB3 #$04 (distance from left edge)
$0DB4 #$20 (distance from top)
$0DB5 #$B7 (low byte of the text address)
$0DB6 #$3C (high byte)

$0DB7 #$0D (job code for data input)
$0DB8 #$10 (distance from left edge)
$0DB9 #$30 (distance from top)
$0DBA #$0E (data will be stored indirectly via $0E/$0F)
$0DBB #$10 (maximum of 16 characters)

$0DBC #$02 (display CANCEL field)
$0DBD #$11 (distance from left edge)
$0DBE #$48 (distance from top)

$0DBF #$00 (end of table)
```

The additional job code $0B displays a request to the user to enter a new name for the diskette. The corresponding text string is located at $3CB7 and is preceded by the control code $18, which determines the type style.

Let's go on to our third example–define our own window.

A sample window

We will now write an application to demonstrate how to create your own windows. This application displays a window with the text string CLICK **YES** TO SEE NEXT PATTERN. Then it places YES and NO buttons within the window. If you click YES it closes the window and then redraws it with a new shadow pattern until it displays the last pattern. If you click NO it closes the window and returns to the deskTop.

```
4580  A9 FF     LDA #$FF     Load accumulator with $FF
4582  8D A5 45  STA $45A5    Store it in 1st byte/window table
4585  EE A5 45  INC $45A5    Increment 1st byte/window table
4588  AD A5 45  LDA $45A5    Load acc. with 1st byte of table
458B  C9 20     CMP #$20     Compare to #$20 - end of patterns
458D  F0 11     BEQ $45A0    Branch to $45A0 if comparson
                             is =
458F  A2 A5     LDX #$A5     Load X with lo byte of table
4591  A0 45     LDY #$45     Load Y with hi byte of table
4593  86 02     STX $02      Store X and Y in pointer
4595  84 03     STY $03      accessed by window subroutine
4597  20 56 C2  JSR $C256    Jump to window subroutine
459A  A5 02     LDA $02      Get the "clicked" window button
459C  C9 04     CMP #$04     Compare to #$04 - the NO button
459E  D0 E5     BNE $4585    Branch not = to increment pattern
45A0  4C 2C C2  JMP $C22C    Jump to reload deskTop
```

The following is the window table used by the above program.

```
45A5 01 14 50 10 00 28 01 0B            Open a non-standard sized
45AD 0A 10 B8 45 03 02 28 04            window. upper border-$14,
45B5 1B 28 00 43 4C 49 43 4B...CLICK    lower-$50, left-$0010
45BD 20 18 59 45 53 1B 20 54..YES..T    right-$0128. Output text at
45C5 4F 20 53 45 45 20 4E 45 O.SEE.NE   ($0A,$10) from text table at
45CD 58 54 20 50 41 54 54 45 XT.PATTE   $45B8. Put a YES button at
45D5 52 4E 00                RN         ($02*8, $28) and NO at
                                    ($1B*8,$28). $00 ends the window table.
                                    The text table starts at $45B8,
                                    contains the string "CLICK YES TO SEE
                                    NEXT PATTERN" and is ended with $00.
```

The following is a BASIC program that creates the file `window` on disk. With a monitor, you can load `window` directly into memory and modify it to try some of the other job codes.

```
1000 rem **********************
1010 rem *    from the book    *
1020 rem *'geos inside and out'*
1030 rem * published by abacus *
1040 rem **********************
1050 open 1,8,2,"window,p,w"
1060 print#1,chr$(128);chr$(69);:rem load addr of $
4580
1070 readx$:if x$="xx"then1100
1080 y$=left$(x$,1):gosub 1120:z=v*16
1090 y$=right$(x$,1):gosub 1120:z=z+v:print#1,chr$(
z);:ck=ck+z:goto1070
1100 close1:if ck<>7065thenprint"error in data"
1110 end
1120 v=-(y$="a")*10-(y$="b")*11-(y$="c")*12-(y$="d"
)*13-(y$="e")*14-(y$="f")*15
1130 v=v+val(y$):return
1140 rem machine code
1150 data a9,ff,8d,a5,45,ee,a5,45
1160 data ad,a5,45,c9,20,f0,11,a2
1170 data a5,a0,45,86,02,84,03,20
1180 data 56,c2,a5,02,c9,04,d0,e5
1190 data 4c,2c,c2,00,00
1200 rem table for window parameters
1210 data 01,14,50,10,00,28,01,0b
1220 data 0a,10,b8,45,03,02,28,04
1230 data 1b,28,00
1240 rem text for window
1250 data 43,4c,49,43,4b,20,18,59
1260 data 45,53,1b,20,54,4f,20,53
1270 data 45,45,20,4e,45,58,54,20
1280 data 50,51,54,54,45,52,4e,00
1290 data xx
```

To convert `window` into a GEOS application, load and run `FILEMASTER`.

Insert the disk containing `window` and press any key (as prompted). Type `csp` to create your icon. We created a picture of a window. Next, type `rfl` to read the file `window`. Type `window` for the name to be searched for. Once the name is found, enter the date. Now, type `geos` to convert it to a GEOS file. Enter 6 for application. Type `info` to create the info sector for `window`. Enter as follows:

```
Starting address : 17790
End address      : 17789
Entry point      : 17792
Program class    : example
Author's name    : Russ Taber
Help screen      : this program demonstrates how to
                   create your own windows
```

Once entered and back to the main menu, type `end` to quit. Now, load in GEOS insert your disk containing `window` and then open `window`. You should see the following display:

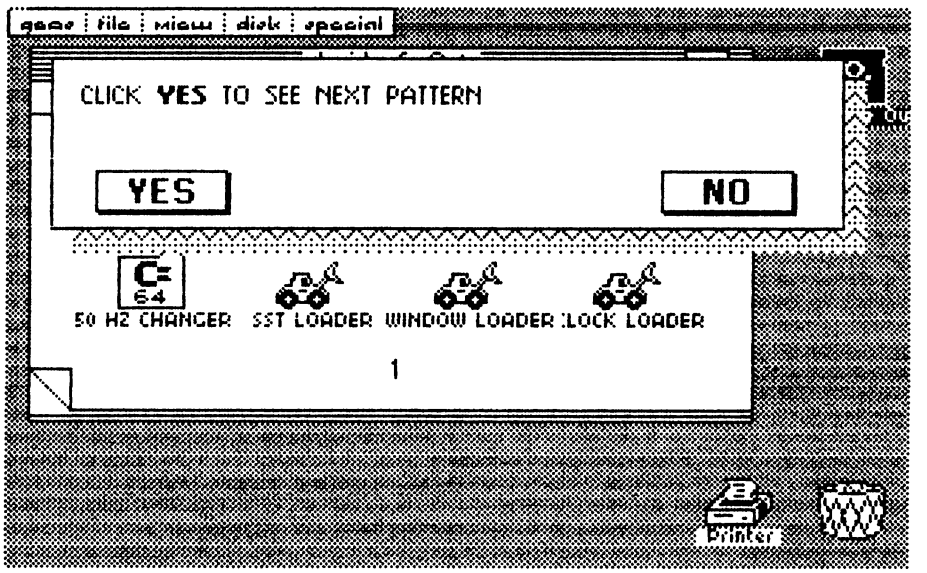


Figure 84: `window` demonstration

6.3 The system interrupt

The system interrupt in GEOS has essentially the same function as in the original C-64 operating system. It relieves the main program of a number of routine tasks. To accomplish this, the main program is interrupted at regular intervals and a special routine is executed. When an interrupt request is made (IRQ), the processor fetches the starting address of the interrupt routine from memory locations $FFFE/$FFFF (Low/High).

Normally the interrupt routine is at $EA31 and performs such tasks as reading the keyboard, flashing the cursor, and actualizing the system clock, TI$. In GEOS, this routine starts at $E2DC. It reads the joystick from the joystick port, reads the keyboard, and decrements certain memory locations on each pass.

An interrupt can be caused by various general events:

- sprite-sprite collision
- sprite-background collision
- raster line of the video controller
- light pen
- timer
- handshake for data transmission
- alarm clock
- BRK command

In the original C-64 operating system, the interrupt is generated by a timer. In GEOS, the video controller raster causes the interrupt. GEOS also allows for a BREAK interrupt.

The raster-line IRQ is generated as follows:

Register 18 of the VIC contains the number of raster line. When this line is being drawn on the screen, the VIC generates an IRQ to the processor. Raster line 0 means that the interrupt will occur at the very top of the screen. If the desired value is greater then $FF, the overflow bit can be placed in bit 7 of register 17. To actually generate an interrupt, bit 0 in register 26 (mask register) must be set.

Since an IRQ can interrupt the processor during any activity, the following rule must be noted: If any of the processor registers are changed during the IRQ routine, the previous values must first be saved. This is normally done by placing them on the stack (PHA, TXA, PHA, TYA, PHA). The registers are then restored before returning to the interrupted main program.

If the processor cannot be interrupted during certain activities, it is possible to disable interrupts requests by executing the SEI instruction in the main program. Interrupts can be enabled again with the CLI instruction.

Let's take a look at the system interrupt in GEOS.

```
$E2DC STA $880B     save acc
$E2DF PLA           get status from stack; placed there
                    automatically on interrupt
$E2E0 PHA           write status back; since the status is
                    the acc, it can be tested
$E2E1 AND #$10      test BREAK flag
$E2E3 BEQ $E2E9     no BREAK, system interrupt
$E2E5 PLA           status no longer needed
$E2E6 JMP ($84AF)   indirect jump to routine: "System error
                    near XXXX"
$E2E9 TXA .......   system interrupt
```

BREAK IRQ

With the branch to a special routine on BREAK IRQ, GEOS has the ability to display a message and memory location on the screen for even serious program errors. This is possible because the return address is placed on the stack when an interrupt request is serviced. This address is fetched from the stack with ($84AF) and printed on the screen.

System IRQ

After the processor registers and important memory locations have been saved on the stack, the interrupt service routine (at $E306) performs the following tasks:

- Enable I/O area
- Decrement counter $8515 by one. This is not done if the counter value is already zero.
- Decrement counter $87EA by one. This is not done if the counter already contains the value $00 or $FF.
- Jump to routine $E36F to read the keyboard.
- Decrement counter $881B, end if $881B=0. This counter creates the interval for the tone creation in case of an alarm.
- Execute job 1: Address of the job in $849D/$849E. This job is normally $E360= joystick movement.
- Execute job 2: Address in $84DF/$84A0. Normally both memory locations are zero, so that this job is not executed.

Following this, the saved memory locations and registers are restored from the stack and the return from the interrupt is prepared.

6.4 Job loop / Job structure

One of the flexible programming features of GEOS is something we've called the *job loop*. This program section checks certain memory locations to see if jobs are supposed to be executed. These can even be placed on stack and processed in succession. In addition, the job loop contains two routines involved with the alarm clock. The job loop starts at $C2C8 and the actual main routine is relatively short:

```
C2C8 JSR $E28F
C2CB JSR $9EA9
C2CE JSR $9F79
C2D1 JSR $F930
C2D4 JSR $F9ED
C2D7 LDA $849B
C2DA LDX $849C
C2DD JSR $C1D8
C2E0 CLI
C2E1 JMP $C2C8
```

If you want to write your own programs to work under GEOS, you can save yourself a great deal work if you are familiar with the basic function of the job loop. We'll briefly explain the operation of the individual routines here.

First we have to talk a little bit about the interrupt, which accomplishes important tasks in GEOS. One of its responsibilities is reading the joystick. Several memory locations play an important role in understanding the interrupt routine. On one hand, the interrupt is informed as to whether the joystick cursor is enabled or disabled, visible or invisible, whether it may go beyond certain boundaries (important so that the cursor cannot leave certain windows), and so on. On the other hand, the interrupt tells the program whether the cursor moved, whether the button was pressed or not, etc.

1. C2C8 JSR $E28F

This subroutine checks memory location $39. In this location the interrupt passes information on whether the joystick was moved or the button was pressed:

```
E28F BIT $39      test joystick status
E291 BVC $E2A2    bit 6=1; new stick position
E293 LDA #$BF     clear bit 6
E295 AND $39
E297 STA $39
E299 LDA $84A5    job address low
E29C LDX $84A6    job address high
E29F JSR $C1D8    job entry
E2A2 LDA $39
E2A4 AND #$20     bit 5=1; change on button status
E2A6 BEQ $E2B7    no, continue
E2A8 LDA #$DF     yes, clear bit
E2AA AND $39
E2AC STA $39
E2AE LDA $84A1    job address low
E2B1 LDX $84A2    job address high
E2B4 JSR $C1D8    job entry
E2B7 BIT $39
E2B9 BPL $E2C7    bit 7=0; no job in FIFO
E2BB JSR $E515    read new job from FIFO
E2BE LDA $84A3    job address low
E2C1 LDX $84A4    job address high
E2C4 JSR $C1D8    job entry
E2C7 LDA $84B6    job flag set?
E2CA BEQ $E2DA    no
E2CC LDA $84A7    job address low
E2CF LDX $84A8    job address high
E2D2 JSR $C1D8    job entry
E2D5 LDA #$00
E2D7 STA $84B6    reset job flag
E2DA RTS
```

This routine makes it possible to execute jobs independently of the value of the joystick port. The big advantage of this routine is that program can wait for a change in the joystick port without being stuck in a loop, simply reading the joystick status. This way GEOS can update the system clock time while geoPaint waits for a joystick movement.

You can write your own values in the jump addresses and execute program fragments independent of changes of the joystick. You could use this to write a hardcopy routine which waits for the user to press the fire button and then stores the contents of the screen.

FIFO (First In, First Out) is a buffer to which two pointers point. The first pointer indicates the next free position within the buffer. The second points to the next element to be processed. The pointers are actualized each time the buffer is accessed. When a new job is written in the buffer, GEOS advances the first pointer one position. When a command has been processed, the second pointer is incremented by one position. This causes the second pointer to lag behind the first. The buffer is actually a "circle", and holds a maximum of 16 entries. When the second pointer catches up to the first (i.e. no more commands need be processed), bit 7 in $39 is cleared. When a new command is added to the buffer, bit 7 of $39 is set to one.

2. C2CB JSR $9EA9

This routine manages a table of jobs which are processed depending on a status word. In addition, a special bit in the status word can be used to disable the processing.

```
9EA9 LDX $878E    job number
9EAC BEQ $9ED6    no job
9EAE DEX
9EAF LDA $872A,X  get job flag
9EB2 BPL $9ED3    bit 7=0; already executed
9EB4 AND #$40     bit 6=1?
9EB6 BNE $9ED3    yes, then skip
9EB8 LDA $872A,X  reset flag
9EBB AND #$7F
9EBD STA $872A,X
9EC0 TXA          save job number
9EC1 PHA
9EC2 ASL          *2
9EC3 TAX
9EC4 LDA $873E,X  job address low
9EC7 STA $02      save
9EC9 LDA $873F,X  job address high
9ECC STA $03      save
9ECE JSR $9ED7    job entry
9ED1 PLA          get job number back
9ED2 TAX
9ED3 DEX          next job
9ED4 BPL $9EAF    another job
9ED6 RTS
9ED7 JMP ($0002)
```

3. C2CE JSR $9F79

This routine manages a job buffer in which a two-byte counter (16-bit) belongs to each job. This counter delays the execution of the job.

```
9F79 LDX $878F    job number
9F7C BEQ $9F9E    no job
9F7E DEX
9F7F LDA $8790,X  counter low
9F82 ORA $87A4,X  counter high
9F85 BNE $9F9B    not 0, skip
9F87 LDA $87CC,X  job address high
9F8A STA $03
9F8C LDA $87B8,X  job address low
9F8F STA $02
9F91 TXA
9F92 PHA          save job number
9F93 JSR $9FA8    advance jobs; current job number
                  overwritten
9F96 JSR $9F9F    job entry + 1 because address is
                  fetched from the stack by JSR
9F99 PLA          get job number
9F9A TAX          and into X
9F9B DEX          next job number
9F9C BPL $9F7F    another job present
9F9E RTS

9F9F INC $02      job address + 1 and execute
9FA1 BNE $9FA5
9FA3 INC $03
9FA5 JMP ($0002)
```

This job table is used by GEOS when you click an OK button. The button is first inverted (black instead of white) and then inverted again after a very brief pause. The counter determines the length of this pause, which in this case contains the value $00A0 as the starting value. The interrupt decrements the counter.

There is a routine at $9FDE to which the starting value is passed in $03 , $02. It fetches the next free location in the table from $877E and writes the starting value for the counter to the corresponding positions in the table. The return address is then fetched from the stack and written into a second table. When the counter reaches zero, the return jump is made.

4. `C2D1 JSR $F930`

Date maintenance.

5. `C2D4 JSR $F9ED`

Alarm clock evaluation.

6.
```
C2D7 LDA $849B   job address low
C2DA LDX $849C   job address high
C2DD JSR $C2D8   job entry
```

7.
```
C2E0 CLI
C2E1 JMP $C2C8
```

Enable interrupts and return to the start of the job loop.

6.5 The GEOS file structure

One of GEOS' many strong suits is its file structure. GEOS' file management makes many options available. When we first started working with GEOS and discovered the changes from normal file structure, we thought that the makers of GEOS were trying to protect development secrets. But we discovered that each file change has an important function in GEOS. This section explains the details of GEOS file structures. To do this, we must first show how normal Commodore DOS manages files.

6.5.1 File management under Commodore DOS

Every file on the diskette has a file entry or directory entry which contains all of the important information about the file. The directory is found on track 18 of the diskette, and starts at sector 1. Here's the format of a directory entry:

Byte	Contents
0	File type † ORed with $80
1,2	Track and sector of the first data block
3-18	Filename, padded with shifted spaces ($A0)
19-21	Used only for relative files
22-25	Not used
26,27	Track and sector of the new file when overwriting
28,29	Number of blocks in the file (low/high)

† File types: 0 = deleted
1 = sequential
2 = program
3 = user
4 = relative

An example:

```
01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
C2 11 00 F  I  L  E  N  A  M  E  A0 A0 A0 A0

16 17 18 19 20 21 22 23 24 25 26 27 28 29
A0 A0 A0 00 00 00 00 00 00 00 01 00 0A 00
```

This file is a program, because the lowest 4 bits in byte 01 equal 2. In addition, it is write-protected, because the 6th bit is set ($C2 instead of $82). The first data block of the program lies on track 17 (=$11), sector 0. The file contains one data block, which is one sector (byte 26=01). From the number of zeroes in the file entry you can see that there is space for additional information. This space is used by GEOS.

6.5.2 A file entry under GEOS

We will now take a look at a file entry under GEOS. A GEOS file entry contains more information than a normal '64 file. This additional information is at the following locations and has the following meanings:

Bytes 19,20 : Sector of the INFO screen

Byte 21 : File structure
0 = SEQUENTIAL
1 = VLIR

Byte 22 : File type
1 = BASIC
2 = machine language
3 = data file
4 = system file
5 = GEOS desk accessory (e.g. the alarm clock)
6 = GEOS application (e.g. geoPaint)
7 = application data
8 = font file
9 = printer driver
10 = input device
11 = disk device

Bytes 23-27 : Date and time, starting with the year

Let's look at such a GEOS file in detail. As an example we'll select the program GEOS BOOT and show the corresponding section of the directory:

A GEOS file entry:

```
0320 ......C2 01 10 47 45 4F ..b..geo
0328 53 20 41 4F 4F 54 A0 A0 s boot..
0330 A0 A0 A0 A0 A0 01 08 00 ........
0338 02 56 03 07 0F 00 06 00 .v......
0340 00 00 ........................
```

The file entry starts at $0322. The first bytes correspond exactly to Commodore DOS. This is important so that the files can be loaded without GEOS. The additional information that GEOS places in the file entry starts at byte $0335 behind the last $A0. The INFO sector lies at track 1, sector 8. The file structure is SEQUENTIAL and the GEOS file type is a machine language program (02).

Following this are the date and time:

Year, month, day, hour, minute

Only the last two digits of the year are included, so $56 = 86 = 1986. The hours are entered in 24-hour format: $0F = 15 = 3 PM.

We have printed a larger section of the directory below so that you can become better acquainted with the GEOS file entry structure and the additional information. Study the file entries until you are comfortable with the pattern. The GEOS information always starts after the last $A0.

Note that GEOS uses a somewhat different coding of the characters than usual. The ASCII values differ from the usual values:

```
0300 12 09 C2 11 00 47 45 4D ..b..geo
0308 53 A0 A0 A0 A0 A0 A0 A0 s.......
0310 A0 A0 A0 A0 A0 00 00 00 ........
0318 00 00 00 00 00 00 01 00 ........
0320 00 00 C2 01 10 47 45 4F ..b..geo
0328 53 20 42 4F 4F 54 A0 A0 s boot..
0330 A0 A0 A0 A0 A0 01 08 00 ........
0338 02 56 03 07 0F 00 06 00 .v......
0340 00 00 C3 01 11 47 45 4F ..c..geo
0348 53 20 4B 45 52 4E 41 4C s kernal
0350 A0 A0 A0 A0 A0 01 09 01 ........
0358 04 56 03 07 0F 00 54 00 .v....t.
0360 00 00 C3 05 07 44 45 53 ..c..des
```

```
0368 4B 20 54 4F 50 A0 A0 A0 k top...
0370 A0 A0 A0 A0 A0 05 14 01 ........
0378 04 56 03 07 0F 00 47 00 .v....g.
0380 00 FF C3 08 0F 47 45 4F ..c..geo
0388 50 41 49 4E 54 A0 A0 A0 paint...
0390 A0 A0 A0 A0 A0 08 06 01 ........
0398 06 56 03 07 0F 00 5B 00 .v......
03A0 00 00 C3 0D 02 47 45 4F ..c..geo
```

6.5.3 The INFO sector

Every GEOS file has an INFO sector which contains information about the file. The INFO sector's address is located behind the last $A0 after the file entry, bytes 19 and 20. Commodore DOS uses these bytes for relative files.

In the INFO sector the bytes have the following meaning:

0-1 00, FF - Normally the first two bytes contain the pointer to the following sector. Byte 0 = track, byte 1 = sector. Since there is only one sector, byte 0 = $00 and byte 1 specifies the number of valid bytes in this sector. The first two bytes thus correspond to the standard Commodore DOS format.

2-3 These two bytes specify the size of the icon that symbolizes this file on the screen. Byte $02 indicates the width and byte $03 the height. A width of 3 means that the icon is three bytes wide and the height of $15 = 21 lines. The icon therefore has the same format as a sprite, which also consists of 63 bytes.

4 This byte represents a flag, the so-called bit-mapping flag. The highest bit is always set ($80) and the lower bits 0-6 specify the number of bytes in the icon ($3F = 63).

5-67 Sprite representation of the icon. Here are the 63 bytes that determine the appearance of the icon. They are in sprite format, so that 3 bytes correspond to one line of the sprite.

68 File type (C-64) ORed with $80. GEOS checks to see if the file is data (=1) or a program (=2). $82=130 must be here for a program. An active write protect sets bit 6 to 1.

69 GEOS file type (1 = BASIC, 2 = machine language, 3 = data file, 4 = system file, 5 = GEOS desk accessory, 6 = GEOS application, 7 = application data, 8 = font file, 9 = printer driver, 10 = input device, 11 = disk device).

70 GEOS file structure type (0 = SEQ, 1 = VLIR). SEQ = SEQUENTIAL here does not mean quite the same thing as in Commodore DOS, but just that the sectors are chained via the pointers in bytes 0,1. SEQ means the normal file structure. VLIR (Variable Length Indexed Record) represents a different format of file. We'll talk about this a little later. Programs that you write yourself = 0.

71,72 Load address of the program. GEOS will load the program into memory at this address. Note: Normally bytes 2,3 in the first data sector of the program specify the start address. This is different in GEOS, and we will explain later how to avoid difficulties that arise because of this.

73,74 These two bytes have significance only for accessories. If you enter a GEOS file type of 2, you don't have to worry about these bytes. However, if you enter 5, for accessory, then these bytes determine the end of the program. This way GEOS knows what area to save out to the disk as a SWAP file when loading the accessory.

75,76 Entry point of the program. GEOS will begin executing your program at this address.

77-159 The strings that will be printed in the info box will be stored here. They must be terminated with 0.

77-96 CLASS = type of program.

97-159 Name of the author.

160-254 Information text. This will be printed only if the deskTop is on the diskette.

255 =0, because the information string is terminated with a zero.

As an example, we will look at the INFO sector of ALARM CLOCK. Viewed with a disk monitor, it looks as follows (note that GEOS uses a different coding system for the characters):

```
0300 00 FF 03 15 BF FF FF FF ........
0308 80 00 01 83 3C C1 8C A5 ........
0310 31 89 7E 91 93 81 C9 96 ........
0318 08 69 8C 49 31 84 10 21 ........
0320 89 10 91 88 10 11 8B 1F ........
0328 D1 88 00 11 89 00 91 84 ........
0330 00 21 84 49 21 86 10 61 ........
0338 87 81 E1 8E 7E 71 80 00 ........
0340 01 FF FF FF 83 05 00 00 ........
0348 54 D8 5F 00 54 41 6C 61 .....aLA
0350 72 6D 20 63 6C 6F 63 6B RM CLOCK
0358 20 56 31 2E 30 00 00 00  v1.0...
0360 00 44 61 76 69 64 20 44 .dAVID d
0368 75 72 72 61 6E 00 10 A9 URRAN...
0370 01 85 11 A9 00 85 17 85 ........
0378 16 20 3B C2 8A D0 0A A5 ........
0380 11 D0 09 20 55 08 B8 50 ........
0388 03 20 20 1A 60 A2 90 A9 ........
0390 00 9D DE 43 CA D0 FA A2 ........
0398 FF A9 00 9D 6E 44 CA D0 ........
03A0 53 65 74 20 74 68 65 20 sET THE
03A8 61 6C 61 72 6D 20 63 6C ALARM CL
03B0 6F 63 6B 20 74 6F 20 6B OCK TO K
03BB 65 65 70 20 79 6F 75 72 EEP YOUR
03C0 73 65 6C 66 20 74 69 6D SELF TIM
03C8 65 2D 63 6F 6E 73 63 69 E-CONSCI
03D0 6F 75 73 2E 00 3B C2 8A OUS.....
03D8 D0 24 A9 08 38 E5 11 18 ........
03E0 69 04 48 09 80 8D C0 05 ........
03E8 68 85 02 A9 0E 85 04 A0 ........
03F8 0C 8D BB 05 A2 00 60 05 ........
```

We printed the character equivalents only for bytes that contain text.

At $0347 we see (in the format low byte/high byte) the load address ($5400), the end address ($5FD8), and the entry point ($5400). When loading the ALARM CLOCK, the area $5400-$5FD8 is first saved as a SWAP file to diskette. When you want to exit ALARM CLOCK, GEOS loads the SWAP file back at this location. With this method, an accessory can be loaded from any application without losing any data.

6.5.4 The border and GEOS format V1.0

There are still two more features of the GEOS file structure we have to discuss. But to do so, we must first look at the BAM of a diskette. It is always located at track 18, sector 0. Normally the Commodore DOS places the following information in the BAM:

Byte	Meaning
0,1	Track and sector of the first block of the directory
2	Format (A = 1541 format)
3	0
4-143	Bit pattern of the occupied and unoccupied blocks (0=occupied). Set up in groups of four bytes, whereby each group characterizes one track. The first byte specifies the number of free blocks on the track and the next three bytes the bit pattern for the sectors 0-7, 8-15, 16-23.
144-161	Name of the diskette padded with Shifted space = $A0
162-163	ID of the diskette
164	Shifted space ($A0)
165-166	2A = format and DOS version
167-170	Shifted space ($A0)
171-255	Not used, filled with zeroes

Like the directory, we also have a good deal of unused space here. GEOS uses this area for its own purposes. Additional information is placed at two locations:

171-172 Track and sector of the border. When you use a disk with GEOS that was not formatted under GEOS, the error message `THIS IS NOT A GEOS-FORMATTED DISK` appears.

You can convert the disk to GEOS format. Another block is reserved, and the pointer in 171,172 points to this block, which represents an extension of the directory. All files which GEOS places on the border receive the character for "scratched" in the normal directory and are entered in this border block. Since only eight file entries fit in a sector, you can place a maximum of eight files on the border.

173-188 `geos FORMAT v1.0`
The GEOS format is placed here at the same time the border block is allocated. From the entry `v1.0` we see that the GEOS file structure has not changed with the new version 1.2. GEOS uses this entry to determine whether or not a disk is in GEOS format when opening.

Now let's take a look at the BAM of a GEOS diskette:

```
0300 12 01 41 00 00 00 00 00 ..A.....
0308 00 00 00 00 00 00 00 00 ........
0310 00 00 00 00 00 00 00 00 ........
0318 00 00 00 00 00 00 00 00 ........
0320 00 00 00 00 00 00 00 00 ........
0328 00 00 00 00 00 00 00 00 ........
0330 00 00 00 00 00 00 00 00 ........
0338 00 00 00 00 00 00 00 00 ........
0340 00 00 00 00 00 00 00 00 ........
0348 0B B4 B5 05 00 00 00 00 ........
0350 00 00 00 00 00 00 00 00 ........
0358 00 00 00 00 00 00 00 00 ........
0360 00 00 00 00 00 00 00 00 ........
0368 00 00 00 00 00 00 00 00 ........
0370 00 00 00 00 00 00 00 00 ........
0378 00 00 00 00 00 00 00 00 ........
0380 00 00 00 00 00 00 00 00 ........
0388 00 00 00 00 02 10 08 00 ........
0390 47 45 4F 53 20 56 31 2E geos v1.
0398 32 A0 A0 A0 A0 A0 A0 A0 2.......
03A0 A0 A0 44 46 A0 32 41 A0 ..df.2a.
03A8 A0 A0 A0 13 08 47 45 4F .....geo
03B0 53 20 66 6F 72 6D 61 74 s FORMAT
03B8 20 56 31 2E 30 00 00 00 .v1.0...
03C0 00 00 00 00 00 00 00 00 ........
03C8 00 00 00 00 00 00 00 00 ........
03D0 00 00 00 00 00 00 00 00 ........
03D8 00 00 00 00 00 00 00 00 ........
03E0 00 00 00 00 00 00 00 00 ........
03E8 00 00 00 00 00 00 00 00 ........
03F0 00 00 00 00 00 00 00 00 ........
03F8 00 00 00 00 00 00 00 00 ........
```

The pointer for the border block points to track $13, sector $08 (19, 8). We also want to look at this block to show that it is constructed just like a directory block:

```
0300 00 FF 83 23 05 4D 50 53 .....mps
0308 2D 31 30 30 30 A0 A0 A0 -1000...
0310 A0 A0 A0 A0 A0 23 0F 00 .....#..
0318 09 56 04 07 0F 06 04 00 .V......
0320 00 00 83 22 03 4D 50 53 ...".MPS
0328 2D 38 30 33 A0 A0 A0 A0 -803....
0330 A0 A0 A0 A0 A0 22 0D 00 .....".. 
0338 09 56 04 07 0F 06 04 00 .V......
0340 00 00 00 20 0D 55 6E 69 ... .Uni
0348 76 65 72 73 69 74 79 A0 versity.
0350 A0 A0 A0 A0 A0 20 06 01 ..... ..
0358 08 56 04 07 0C 00 28 00 .V....(.
0360 00 00 00 1A 00 43 61 6C .....Cal
0368 69 66 6F 72 6E 69 61 A0 ifornia.
0370 A0 A0 A0 A0 A0 1A 0B 01 ........
0378 08 56 04 07 0C 00 1A 00 .V......
0380 00 00 00 1C 01 43 6F 72 .....Cor
0388 79 A0 A0 A0 A0 A0 A0 A0 y.......
0390 A0 A0 A0 A0 A0 1C 0C 01 ........
0398 08 56 04 07 0C 00 17 00 .V......
03A0 00 00 00 1D 04 44 77 69 .....Dwi
03A8 6E 65 6C 6C 65 A0 A0 A0 nelle...
03B0 A0 A0 A0 A0 A0 1D 0F 01 ........
03B8 08 56 04 07 0C 00 0D 00 .V......
03C0 00 00 00 1E 05 52 6F 6D .....Rom
03C8 61 A0 A0 A0 A0 A0 A0 A0 a.......
03D0 A0 A0 A0 A0 A0 1E 10 01 ........
03D8 08 56 04 07 0C 00 22 00 .V....".
03E0 00 00 00 23 05 4D 50 53 ...#.MPS
03E8 2D 31 30 30 30 A0 A0 A0 -1000...
03F0 A0 A0 A0 A0 A0 23 0F 00 .....#..
03F8 09 56 04 07 0F 06 04 00 .V......
```

Here again we have given the character equivalents only for the names and replaced the rest with periods. Note the unusual notation. The first letter appears in lower case and the rest in upper case. This is how the file entries would look if you used a disk monitor to examine them.

The first two bytes of the border block indicate that there is no block following this one and that there are $FF (=255) valid data present. Only the first two file entries (`mps-1000` and `mps-803`) are valid because only they have a valid file type ($83). The following file entries do not appear on the border because their file types are set to 0.

Note: If you place a file on the border and then exit GEOS, you will not be able to load it again without GEOS. GEOS erases the file from the normal directory. If you place GEOS BOOT from the System diskette on the border and then exit GEOS, you won't be able to boot GEOS from that diskette.

6.5.5 File type / File structure

When you want to use programs of your own under GEOS, you'll have to know what file types GEOS uses. You may have seen terms like SYSTEM FILE, DESK ACCESSORY, FONT FILE, and so on appear under TYPE in the INFO screen of various GEOS files. GEOS distinguishes between a total of twelve different file types. We have listed these file types below:

GEOS file type (GT)	File structure (GS)	Designation (INFO window)	Example (program)
1	0	BASIC	BACKUP
2	0	Assembler	GEOS BOOT
3	?	Data File	none present
4	1	System File	GEOS KERNAL
5	0	Desk Accessory	ALARM CLOCK
6	1	Application	GEOPAINT
7	1	Application Data	NOTES
8	1	Font File	UNIVERSITY
9	0	Printer Driver	MPS 1000
10	0	Input Driver	JOYSTICK
11	?	Disk Device	none present
12	0	System Boot	GEOS BOOT

GEOS also recognizes what you can do with this file from the file type. For example, you can't print a desk accessory (GT=5) or copy a system file (GT=4).

Now we'll explain the difference between FS=0 (SEQ) and FS=1 (VLIR).

File structure: SEQ

If you store a program without GEOS, a directory entry will be created. A pointer to the first data block will be placed behind the file type. At the start of the data block there is a pointer that points to the data block following it. The last block of data always has a 0 as the first byte. The second byte specifies the number of valid bytes in the last sector. This combines the individual data blocks into a long chain. That's why GEOS calls this type of file SEQUENTIAL.

For example, if you want to read the last data block, you would have to follow all of the pointers until you got to the last block. All of the previous sectors in the chain must be loaded. In addition, you can only add things by lengthening the chain. For this reason GEOS has a second file structure.

File structure: VLIR (Variable Length Indexed Record)

If you use the disk monitor to look at the file entry of a file that has the VLIR file structure (such as deskTop), you won't see any difference at first. But when you load the first data block (track and sector behind the file type), you'll be surprised to see that there is only one sector of data. The first byte of the first sector is zero, meaning that no additional sectors follow. Furthermore, this sector does not contain the start of the program or data, but just a few numbers and the rest of the block is alternately filled with 00 and FF.

The first numbers in this sector represent pointers to records of the program of data. These records are again joined to each other sequentially, with a pointer at the start of each block to the next one. This has two important advantages:

First, the length of the individual program or data records can be changed. Additional sectors can be added. Second, the last sectors can be accessed faster because the first blocks need not be read.

One of the places GEOS makes use of this is in the note pad, which has a VLIR file structure. There can be up to 127 pointers to pages of the note

pad. Each record corresponds to a page and consists of just one sector. If you fill pages of the pad with `NOTE PAD 3` and then use the disk monitor to look at the sector specified as the first block of the file, six bytes will show up behind the first two bytes $00 and $FF. These are the pointers to the three sectors in which the text is stored. The pointers have the format *track, sector.*

In addition to data, GEOS also recognizes programs with the `VLIR` file structure. Here the construction looks the same—a pointer points to each record—but loading is done differently. Usually the first record represents the entire program. The other pointers point to records that can be loaded using the overlay technique. This involves overwriting parts of the existing program with other sections loaded from disk as they are needed.

For example, geoPaint has seven pointers:

track	sector
$08	$03
$10	$11
$10	$0A
$10	$06
$11	$12
$13	$05
$13	$0F

These pointers can have different values on your disk.

When you click geoPaint on the deskTop, the first record (track $08, sector $03) is loaded. The other records can then be loaded by geoPaint to accomplish specific tasks.

We'll use two examples to illustrate the overlay technique.

1) deskTop consists of two records:

   Record 1: track $05 sector $10

   Record 2: track $08 sector $02

   Normally record 1 is loaded. When you load the INFO sector, however, the text in record 2 is loaded for the output. You can see this if deskTop is not on the current disk and no text will be printed, but this error message appears:

```
DESKTOP NOT ON DISK
```

2) When you select various tools in geoPaint, routines belonging to the given tool are loaded from disk. Record 2 of geoPaint (track $10, sector $11), for example, belongs to the "faucet" used to fill surfaces.

How can you load a record? Using your disk monitor, find the track and sector of the record you want to load. Then create a dummy program, which we'll use to create a file entry.

You can create a dummy program in BASIC by entering the following:

```
NEW
10 GOTO 10

SAVE "dummy",8
```

Now you to know two things:

1. The start address of the record in memory.

Look in the file entry to find out where the INFO sector of the record you want to load is located. The track and sector of the info box are after the last $A0 of the file entry. Load this sector into the disk buffer and the load address of the record will be in memory locations $0347 (low byte) and $0348 (high). geoPaint's start address is $0400.

2. You must change the first two bytes of the record. In our example, put the first block (track $08, sector $03) in the buffer. The first bytes are then:

   $0300 = track of the next block
   $0301 = sector of the next block
   $0302 = $4C = JMP
   $0303 = $04 = address: low
   $0304 = $05 = address: high

To load the record at the correct address, write down byte $0302=$4C and byte $0303=$04. Then change these so that $0302=$02 and $0303=$04. When you later load this record, GEOS will look for the load address in these two bytes. The address must be two bytes higher than the load address of the record, because the first two bytes will be overwritten and you will have to insert them yourself after loading.

Now write the modified sector back to the disk. Read the part of the directory that contains the file entry `dummy` into the disk buffer. In this entry, change the address of the first data block to the address of the record. Behind the $82 (file type in the file entry) write a $08 (start track of the record) and then a $03 (start sector of the record). Your "dummy" file entry now begins with the three bytes:

```
$82  $08  $03
```

After you have written the modified directory sector back to the disk, you can load the record simply by loading `dummy`.

```
LOAD "dummy",8,1
```

Once you insert the two missing bytes with the monitor, you have geoPaint in memory. Change $0400 to $4C and $0401 to $04.

Here we face the difficulty that the screen normally lies at $0400. We can get around this by loading geoPaint with an offset of $1000. To do this, write $02, $14 in the first block of the record instead of $02, $04. This will load geoPaint at $1402, and you can examine the program at your leisure.

6.6 Writing your own GEOS programs

In this section we want to show you various ways of loading your own programs under GEOS and adding your own GEOS applications. Loading your own programs from GEOS is especially interesting because the fast-loading routines of GEOS can be used under certain circumstances.

6.6.1 How to load your own programs

With GEOS you can load programs written in BASIC or assembly language. There are some differences.

a) BASIC programs

You can load any BASIC program from GEOS by double-clicking the program icon or by clicking OPEN. GEOS recognizes a BASIC program by the starting address $0801=2049, which is located in the first data block of the program on disk behind the pointer to the next block. GEOS then checks to make sure that the program does not exceed a certain length. The GEOS fast-loading routines use memory starting at $8000 in the C-64. If the BASIC program does not extend into this area, it will be fast-loaded with these routines and automatically started.

If the program is too large, it will be loaded by the normal '64 operating system, which is slower. The program can still be loaded by double-clicking, however.

b) Assembly language programs

GEOS recognizes programs written in assembly language by a starting address that is not equal to $0801. These programs can be loaded only if the start address is under $0400. Such programs usually use the stack or auto-start start area. We found that the slow C-64 routines were always used for loading assembly language programs.

With a little trick, however, it is possible to use the fast GEOS routines for assembly language programs as well. To do this, simply convert the program into a BASIC program. To do this convert your program to start with a SYS command. If your program isn't in the location you want, you'll have to write a routine to transfer it to where you want it, as follows:

$0801 = 2049: `10 SYS 2080`
($0820 = 2080: move routine; moves the program into the memory area for which it was written.)

$0850 = 2128: Actual program to follow move routine

With this method, assembly language programs can also be loaded with the fast-loading routines and then started automatically.

6.6.2 How to extend GEOS with your own programs

There are two ways to wedge your own programs into GEOS. Both methods require a comprehensive knowledge of GEOS internal operations.

a) Applications

Let's start with the simpler method. We want GEOS to load a user program, start it, and then return to the deskTop when the program is finished. To achieve this, you have to write your own APPLICATION in assembly language, because the BASIC interpreter is not available under GEOS. In addition, the screen output must take place on the hi-res screen at $A000. How is this type of program inserted in GEOS?

1) Place a `JMP $C22C` at the end of your program. First you must set up your program so it can return to GEOS. Branching to this address reloads the deskTop. As long as you did not change the screen colors, the deskTop returns with its previous colors. If you did change the colors, a `PREFERENCES` file must be on the diskette to set the colors back to the correct values.

2) Save the program. Now it must be converted into an application. Use a blank work diskette in case an error occurs. You can later copy the program from this work diskette to the desired disk.

3) Now the actual conversion to the format of an application must take place. You can accomplish this with a disk monitor or more easily with the `FILEMASTER` program (listed in Chapter 5). Applications are GEOS file type 6.

The following changes must be made:

- Your program must contain an INFO sector. The track and sector must be located after the last $A0 of the filename.
- Change the next two bytes to 0 and 6, respectively. The first byte signifies the file structure (SEQ) and the second byte indicates the GEOS file type (application).

You can make this change with the FILEMASTER program by selecting RFL, read file. Select GEOS from the menu and 6 to set the GEOS file type. You can also enter a date (date).

With INFO sector, the following bytes in the buffer must be set to the following values (buffer 0 = $0300-$03FF).

Byte $0344 = C-64 file type (such as $82)

Byte $0345 = GEOS file type (=6)

Byte $0346 = File structure (=0=SEQ)

Byte $0347 = low-byte of load address

Byte $0348 = high-byte of load address

For the load address, you must enter a value of where your program is to load into memory. The regular DOS used by the computer saves two bytes at the start of your program to tell it where to load. GEOS takes care of this in the INFO sector, and therefore doesn't use these two bytes. If you saved your programs using regular DOS, then you have to offset the load address. GEOS will load those two extra bytes in, so you have to move the load address down by two. The following two example programs (application and accessory) do not use the regular DOS load address if loaded using the BASIC loaders. Therefore the two-byte offset is not needed.

Byte $0349 = end address of the program (low)

Byte $034A = end address of the program (high)

We've found that this address is not used for applications. You should still enter a value here, however, since the geoWrite and geoPaint applications also have values here. These two bytes do not contain the actual end

address, they contain the entered load address minus one. For example, if we set the load address to $1000, the end address would be $0FFF.

For accessories, the memory between the load address and the end address entered is saved by GEOS as a SWAP file on the diskette. This is not done if you enter the value just described.

Byte $034B = entry address (low)

Byte $034C = entry address (high)

This is the actual entry point of where your program takes over.

After you have saved the changes on disk, the application is finished. You can now use it from GEOS. We would like to offer a simple example here. The following program changes the screen border color and then loads the deskTop again.

```
4590  A5 01     LDA $01       Store the memory
4592  48        PHA           setup on the stack
4593  09 07     ORA #$07      Switch in ROM and
4595  85 01     STA $01       I/O
4597  EE 20 D0  INC $D020     Increment border color
459A  68        PLA           Return old memory
459B  85 01     STA $01       setup back
459D  4C 2C C2  JMP $C22C     Jump to load desktop
```

```
100 open 1,8,2,"bcc app.,p,w"
1000 fori=17808to17823:readx:ck=ck+x
1010 print#1,chr$(x);
1020 next
1025 ifck<>1418 then print"error in data"
1030 close1
1040 end
9000 data 165,1,72,9,7,133,1,238
9001 data 32,208,104,133,1,76,44,194
```

In this example, set the following values with FILEMASTER under the option info:

Load address = 17808

End address = 17807

Entry point = 17808

b) Accessories

The main advantage of accessories is that they can be loaded from any application as well as the deskTop. When you are finished working with the accessory, you can continue with the application where you left off. This is only possible because GEOS places certain parts of the program on the disk as a SWAP file. How does this work?

Before the accessory is loaded, GEOS reads the start and end addresses. The area between these two addresses in memory is then saved out as a SWAP file on the diskette. The accessory is then loaded and can be started. When you want to leave the accessory again, GEOS loads the SWAP file back in at the correct location in memory and the previous program (the application) is set back in its original condition.

You can test this with the RESET button on your computer (if you have a C-64 and haven't installed a reset button, then you'll have to take our word for it). Boot GEOS and open the alarm clock. After it is loaded, exit GEOS by pressing the RESET button. If you now boot GEOS again, you will discover a new file (perhaps not on the first page) which has the same icon as the alarm clock and is called SWAP file. If you look at the INFO sector of the SWAP file with your disk monitor, you will see that the load and end addresses match those of the alarm clock. After examining it, delete the SWAP file, for it is of no use anymore.

How do you create your own accessories?

Basically, the same steps involved in making an application are necessary for making an accessory. The following changes must be made:

GEOS file type = 5

Set the end address of the program with the `info` option to the end address of the program, not to the load address minus one.

Don't exit from the accessory with `$C22C`. Exiting the program in this way is not possible for two reasons. First, the user should be able to load the accessory from geoPaint as well, and ending the program with `JMP $C22C` would simply reload the deskTop. Naturally, the current geoPaint screen data would not be saved. Second, a `SWAP` file would appear on the disk each time.

There is a specific address provided for ending a desk accessory. When a branch is made to this address, GEOS loads the `SWAP` file from diskette, deletes, and then continues the program execution at a modifiable address.

The entry address for loading the `SWAP` file is $918F. The KERNAL vector pointing to this is $C23E. This address, where the program should continue after loading the `SWAP` file, is passed into $849B and $849C. This address is pushed on the job stack and then branched to with `RTS`. In this case the following bytes must be passed:

$849B = $3E

$849C = $C2

```
4590  A5 01     LDA $01      Store the memory
4592  48        PHA          setup on the stack
4593  09 07     ORA #$07     Switch in ROM and
4595  85 01     STA $01      I/O
4597  EE 20 D0  INC $D020    Increment border color
459A  68        PLA          Return old memory
459B  85 01     STA $01      setup back
459D  A2 3E     LDX #$3E     Put $C23E in job loop
459F  A0 C2     LDY #$C2     (Load Swap file)
45A1  8E 9B 84  STX $849B
45A4  8C 9C 84  STY $849C
45A7  60        RTS
```

```
100 open1,8,2,"bcc acc.,p,w"
1000 fori=17808to17831:readx:ck=ck+x
1010 print#1,chr$(x);
1020 next
1025 ifck<>2635 then print "error in data"
1030 close1
1040 end
9000 data 165,1,72,9,7,133,1,238
9001 data 32,208,104,133,1,162,62,160
9002 data 194,142,155,132,140,156,132,96
```

In this example, set the following values with FILEMASTER:

Load address = 17808

End address = 17832

Entry point = 17808

6.7 Memory layout and memory locations

If you want to wedge your own program into GEOS, you must know two things:

- The memory map—what areas of memory your program can use without destroying GEOS.
- Important memory locations—which memory locations GEOS uses for its purposes, and which locations must be saved before you use them for your own purposes.

Let's start with the memory map of the C-64 in the basic GEOS configuration (GEOS KERNAL + deskTop).

6.7.1 Memory map

```
$0000-$00FF : Zero page, important pointers and system addresses

$0100-$01FF : Processor stack

$0200-$03FF : Essentially 0 up to $0300-$0340, which contains the usual
              pointers. GEOS has set some of these pointers to its own
              values.

$0406-$40FC : Program deskTop

$40FD-$456F : Working memory deskTop

$4570-$496E : Not used (safe for your own use)

$496F-$4AFF : Working memory deskTop

$4B00-$4BFF : Directory page 1 (diskette)

$4C00-$4CFF : Directory page 2

$4D00-$4DFF : Directory page 3

$4E00-$4EFF : Directory page 4

$4F00-$4FFF : Directory page 5
```

```
$5000-$5FFF : Partially used as working memory

$6000-$7F40 : Copy of the bit-mapping screen

$8000-$80FF : Buffer 1 for reading or writing disk sector

$8100-$81FF : Buffer 2

$8200-$82FF : Buffer 3 for disk BAM

$8400-$8BFF : Working area, storage for mouse pattern ($84C0), etc.

$8C00-$8FFF : Color RAM

$9000-$9FFF : GEOS kernal part 1

$A000-$BFFF : Bit-mapping screen

$C000-$FFFF : GEOS kernal part 2. At $C100-$C2C5 there is a jump table
              for a number of important functions.
```

6.7.2 Important memory locations

```
$0030  Mouse flag
       Bit 7=0: no mouse action
       Bit 6=1: tests for marked area

$0039  Joystick flag
       Bit 7=0: no jobs in job buffer
       Bit 6=1: stick position changed
       Bit 5=1: button pressed

$003A  Current mouse X-position low

$003B  Current mouse X-position high

$003C  Current mouse Y-position

$84A1  Job address low: button pressed

$84A2  Job address high: button pressed

$84A5  Job address low: joystick movement
```

```
$84A6  Job address high: joystick movement

$84A7  Job address low: mouse left specific area ($84B6 not equal to 0)

$84A8  Job address high: mouse left specific area

$84B6  Mouse status register
       Bit 7=1: Contact upper border (area 1)
       Bit 6=1: Contact lower borer
       Bit 5=1: Contact left border
       Bit 4=1: Contact right border
       Bit 3=1: Mouse outside second marked area

$84B8  Upper border for mouse

$84B9  Lower border for mouse

$84BA  Left border for mouse LOW

$84BB  Left border for mouse HIGH

$84BC  Right border for mouse LOW

$84BD  Right border for mouse HIGH

$84C1  Upper border for mouse (2nd area)

$84C2  Lower border for mouse (2nd area)

$84C3  Left border for mouse LOW (2nd area)

$84C4  Left border for mouse HIGH (2nd area)

$84C5  Right border for mouse LOW (2nd area)

$84C6  Right border for mouse HIGH (2nd area)

$8501  Maximum mouse speed

$8502  Minimum mouse speed

$8503  Acceleration: increment

$8505  Code for current button position
       Bit 7=0: button pressed
       Bit 7=1: not pressed
```

```
$8506  Code for current stick position
       $00 = right
       $02 = up
       $04 = left
       $06 = down
       $FF = error or neutral position

$8507  Current mouse speed

$8516  Current year

$8517  Current month

$8518  Current day

$8519  Current hour

$851A  Current minute

$851B  Current second

$87DD  Number of jobs in buffer 1

$8719-$872C  Table of flags for the jobs in buffer 1

$872D-$8755  Table of job addresses in buffer 1 (low/high)

$877E        Number of jobs in buffer 2

$877F-$8792  Table of delay counters LOW for buffer 2

$8793-$87A6  Table of counters HIGH for buffer 2

$87A7-$87BA  Job addresses LOW buffer 2

$87BB-$87CE  Job addresses HIGH buffer 2

$87D7  FIFO read pointer

$87D8  FIFO write pointer

$87DA-$87E9  FIFO (buffer 0)

$C2C8   job loop

$CC4A - loads deskTop *
```

```
$E2DC - IRQ entry *

$E36F  read the keyboard

$E5F0  mouse movement within the IRQ *

$FE8B  Absolute acceleration

$FE8C  X-speed (+/-)

$FE8D  Y-speed (+/-)

$FE8E  Value of the joystick port, button only

$FE8F  Value of the joystick port, stick only

$FE90  Value of the joystick port
```

Glossary

Glossary

Accessory: A program that is available at practically any time. For example, you can create a graphic document with geoPaint, and then use the calculator while you're in the middle of editing it. You cannot use multiple accessories at the same time, however.

Application: A program that can be loaded only from the deskTop. An application makes use of the GEOS routines (for example, geoPaint and geoWrite).

Assembly language: A computer language that has a one-to-one relationship to machine language. Assembly language is simply a more readable way of writing machine language.

Backup: Name for a copy of the contents of a diskette. There is a program named `DISK COPY` on the GEOS system diskette. This program can be used to make copies. It can be loaded from the deskTop with a double click.

BASIC: One of the most popular high-level language. The '64 understands BASIC only indirectly. The only language that the computer understands directly is machine language, which is rather difficult for humans to work with. BASIC is translated into this machine language by the BASIC interpreter. BASIC is built into the '64, so it is available to you at any time.

BASIC interpreter: A program that converts BASIC into machine language. If you write a BASIC program and start it with `RUN`, the program's BASIC commands are broken up into the smallest tasks that the computer can process—in machine language code.

Booting: The name of the procedure by which GEOS and other programs are loaded and started. You can boot GEOS only from the original diskette.

Border: Refers to a specific area on the screen when you are in the deskTop. This area lies below the GEOS window at about the level of the printer icon. When icons are placed on the border, their order can be changed, and they can be copied.

Buffer: An area of memory used for short-term data storage. We usually use this term to describe a diskette buffer. This diskette buffer stores the sectors that have been read in from diskette or will be written to the diskette.

Copy protection: Diskettes can usually be copied just like cassettes can be copied. Since software manufacturers often see unrestricted copying of software as contrary to their business interests (as well as a violation of copyright laws), they attempt to prevent the successful copying of their disks. Usually some aspect of the diskette track or sector is changed, and this alteration is tested by the program. If the alteration is not on the inserted diskette, the program will not run. These alterations—the copy protection—cannot normally be copied. GEOS has a very effective form of copy protection, which is checked only when you're booting.

Cursor: Flashing symbol on the screen that tells you the computer is waiting for input. On the '64 under GEOS, the cursor appears as a vertical line; the usual '64 cursor is rectangular. In geoWrite the cursor indicates the location at which the next letter will appear in the text.

<DEL> key: Located on the upper right of the keyboard, this key deletes one character to the left of the cursor when pressed.

deskTop: A program that creates a user interface resembling the surface of a desk. On this "desk" objects can be moved and processed. In addition, the deskTop contains a printer and a waste basket. deskTop tries to make working with the user interface like working on an ordinary desk top.

Destination disk: The diskette to which you want to copy a file.

Directory: The table of contents of a diskette. A file entry is created in the directory when each file is saved so that the disk drive does not have to search through the entire diskette to find a file.

Disk buffer: see *Buffer*.

File: A collection of data. A file is any cohesive block of data. The difference between a file and a program is that a program also contains data items that are recognized by the computer as commands. Under GEOS, each file has a name (filename) and also an icon (symbol). Documents, programs, photo albums, etc. are all files.

File entry: Important data about a file is stored as a file entry in the disk directory. This data includes the filename, size, location on the diskette, type of file, and so on.

Formatting: The process that prepares a diskette for use by the '64. The diskette is completely erased in this process. You should format all of the disks which you need for working with GEOS under GEOS—its formatting differs from the normal '64 format command. GEOS gives the diskette a special format.

GEOS: Graphic Environment Operating System. GEOS creates a user interface with windows, pull-down menus, icons, and a joystick. In addition, it offers fast disk operations and its own file format.

GEOS window: Designates a section of the screen (usually the middle) in which up to eight files can be displayed at one time in the deskTop.

geoPaint window: The section of the screen in which you create and edit your graphic documents. The window displays only a small part of the entire document. You can move this part.

Hardcopy: A paper copy of the information on the '64 screen that is produced by a printer or plotter.

Icon: A graphic symbol. GEOS represents its documents and graphics on the screen as icons when you work with deskTop. The appearance of an icon generally correlates with its purpose. The printer and waste basket are also icons.

Input device: Device used to control the GEOS pointer. Joysticks, mice, lightpens and drawing pads are examples of input devices.

Interrupt: Process used by the computer to interrupt its current task, process another task, and then return to its original work. In GEOS, reading the joystick and keyboard is accomplished through interrupts.

Menu: Line on the screen (usually in the upper left) that lists various sub-menus. deskTop contains the following menus: **`geos`**, **`file`**, **`view`**, **`disk`**, and **`special`**. Each names a menu containing a set of additional commands known as *items*. When you click an item in the menu, its sub-menu is displayed in a window below the menu.

Operating system: Describes a program that performs the most important and fundamental operations of a computer. Without an operating system program, the computer can do nothing. In the '64 the operating system is stored in ROM so that it remains intact when the power is turned off.

Page pointer: Term from geoPaint and geoWrite. In geoPaint it refers to a rectangle in the status line. This rectangle corresponds to the entire page, and the small rectangle that appears in the upper left corner at the beginning indicates the situation of the geoPaint window. In geoWrite it appears at the top of the screen next to the name of the document and also indicates the page number.

Pixel: Smallest representable point on the screen.

Pointer: An important part of the GEOS user interface, the pointer represents your finger on the screen. With this "finger" you can grab icons and move them.

The pointer can assume a variety of shapes in GEOS. Normally it is displayed as an arrow. You may change the arrow to another shape by editing the pointer sprite in the Preference Manager. The pointer might also look like a crosshair or a thin pen in geoPaint. The pointer is moved with the input device (see above). The input device button is used to click at the spot to which the arrow points.

Printer adaptation: Since there are different types of printers, not every printer can be connected to every computer. Usually a program known as a *printer driver* must be loaded to match the computer to the printer. You tell the computer what type of printer you are using before you print something.

Proportional type: A kind of type in which the letters require a variable amount of space. For example, the letter "I" occupies considerably less space than the letter "M".

<RESTORE>key: Located on the upper right of the keyboard in the second row, it's used in GEOS to reboot the system. If you exited GEOS through BASIC or by loading a BASIC program, insert the original diskette and press the <RESTORE> key to boot GEOS.

Screen memory: Part of the computer's internal memory. It stores information that determines the appearance of the screen. Without GEOS, the screen memory usually consists of 1000 memory locations, containing the codes for 40 x 25 possible characters on the screen. It's sufficient to write the correct number in the screen memory to display a character on the screen. In contrast to this there is also the hi-res (high-resolution) screen. You're not limited to the Commodore-defined character sets with this screen. Instead, you can determine the display of the entire screen with individual pixels. GEOS uses this hi-res screen.

Scroll box: A small rectangle that contains two arrows. You can scroll the contents of the window in the direction of the arrows. This causes part of the screen display to disappear and additional lines to appear at the opposite side of the screen. The scroll box is usually used when the window is too small to contain all of the information.

Sector: The diskette is divided into a number of tracks. In turn, the tracks are divided into a number of sectors. The disk drive can locate a sector on the diskette with the help of special markings. A sector contains a maximum of 256 bytes. When you store a program, it is divided up into suitable segments. These segments are then stored one to a sector.

Single-step simulator: A program that causes another program to be processed step by step, allowing the user to examine important information after each step. A single-step simulator program is printed in Chapter 5 of this book. You can use it to examine GEOS programs step by step.

Source disk: The original diskette. This is the diskette you want to copy.

Sprite: Graphic symbols that can be moved on the screen. Sprites on the '64 consist of 24x21 point blocks. Sprites are especially useful for writing games because they can be moved very easily. For example, GEOS uses sprites for displaying the joystick pointer on the screen. Because sprites are created differently than the normal screen elements, they do not appear on a hardcopy.

Track: The diskette is divided into a number of circular tracks, which consist of a number of sectors. There are 35 tracks on a '64 diskette.

User interface: Many computers allow you to communicate only in a programming language (such as BASIC). Before the computer can do any useful work for you, you must first learn the language. With a good user interface you can use a computer without any knowledge of programming languages or how the computer actually works. One user interface involves graphically displaying important elements for working with the computer on the screen itself, which can then be manipulated to perform various tasks. For example, files are displayed as graphic symbols called *icons* in GEOS. You delete a file simply by moving the file's icon over to the waste basket icon.

Window: A section of the screen delimited by a border. This section is then treated as a separate screen for a certain length of time. GEOS uses windows to output information (error messages) or to ask you for information (the name of a document). The special thing about windows is that data can be displayed on the screen and afterward the original screen is restored. To do this, the part of the screen memory that will be changed by the window is saved at another location and later written back.

Working copy: You must copy some GEOS files to another diskette to use this diskette for applications and save wear and tear on your system diskettes. This "incomplete" copy is called a working copy.

Index

A

accessory programs 17, 67-86
add drive 38
adding files 19
airbrush 52, 133-135
alarm clock 35, 70, 180-186
application programs 7, 24, 43, 68, 80, 87, 88, 111, 120, 130, 134, 203, 221, 267, 269
arrows 37, 50, 59, 67
ASCII 147-148, 161-165, 231, 254

B

backup copy diskettes 7-14, 128, 197-198
BAM 258-259, 275
BASIC 39
BASIC 33, 147-175, 267-269, 281-285
blank diskettes 8, 19, 38, 127, 197
booting 7, 14, 125-129, 197-198, 254, 262, 271, 281, 282, 284
boundary 93, 131
BREAK IRQ 245
bricks (geoPaint pattern) 136
brushes 48, 49, 53, 133
BSW (font) 90-93, 104, 112, 114, 118, 137, 140
buffer 160-164, 167-170, 177, 230, 231, 238, 248, 250, 265, 266, 269, 275, 277, 282

C

C-64 31, 71, 87, 108, 235, 244, 256, 267, 269, 271, 274
C= (Commodore key) 33, 214-217, 281-285
calculator 35, 71
calendar (Notepad) 144-147, 180
CANCEL button 130, 219, 224, 225, 232-239
change brush 48
characters 27, 43, 65, 76, 91, 111
charts 87-99
circles (geoPaint) 106, 120-121, 132
clear 51
clicking 12, 24, 27
clock (programming) 178
clock (constant display prg.) 187-196
close 22
close icon 7, 9, 12, 15, 22, 32
color 56
color off 49
command menu 8
<CONTROL> key 141
copy 47, 61
copying (files) 42
Create 25, 43
crosshairs 47, 51-55, 95-97, 103
cursor 18, 26, 55
custom windows 225
cut 47, 61

D

data files 33
DATASETTE 31
<DEL> key 13, 38, 46, 61, 63-65, 67, 72, 142, 143
deleting 14, 40, 142, 162}}
deskTop 11, 16, 18, 20, 28, 31-40, 128-130, 263-265, 270-272
deskTop info 34
deskTop parameters 17, 34, 74-75, 125, 215, 276, 277
destination diskette 8, 38, 42, 47
diagrams 87-99
dialogue box 13, 219, 223-224
dialogue boxes (custom) 225-233
directory 3, 13, 20-23

disk 12, 22, 37
DISK COPY 8-9
disk drive 7-10, 13-15, 38, 126-128, 160, 178, 197, 234
disk drive head alignment 126-127, 198
disk drive error messages 218, 235
diskette icon 32
documents (GEOS) 17, 34, 43, 44, 46-49
dogear page 21, 145
DOS (Disk Operating System) 129, 173, 252-258, 269
dotted lines 134-137
double-clicking 24, 42, 54, 132
duplicate 35
Dwinelle (font) 137, 140

E
edit 47
editing 47-48, 50, 51, 54, 55, 57-66
edit box (geoPaint) 47, 48-55
electronic circuits 108
ellipses (geoPaint) 106, 120, 132
Epson FX-185 132
Epson FX-85 130, 131
eraser (geoPaint) 48, 54, 57, 75, 102, 132
error box 218, 223

F
faucet (geoPaint) 51, 90, 97, 133, 265
FIFO storage 248, 277
Figures (listed) *viii-ix*
file 35
files (GEOS) 8, 14, 16-21, 23,
files (non-GEOS) 33
files (under DOS) 252
file structure 146, 252-259, 263-264, 269
file types 33, 148, 163, 173, 177, 252-256, 262
FILEMASTER 3, 129, 148-150, 160, 167, 171-177, 189, 203, 242, 268-273
FILEMASTER menu 171
FILEMASTER (prg. listing) 149
filenames 17, 35, 44, 131, 162, 163, 167, 172, 194, 230, 269
fire button 12
font 26
font (geoPaint) 49
fonts 21, 26-28, 33, 49, 54, 55, 65, 67, 84, 90, 121, 137-143, 262
format 39
formatting GEOS diskettes 8, 39
formats (geoWrite) 65-66, 128, 137-143

G
geoPaint 3, 17, 24, 36, 42, 44-56, 53-56, 58, 61-64, 131-136
geoPaint info 45
geos 17, 45
geos info 34
GEOS *iii*, 3, 4, 7-25
GEOS BOOT 9, 14, 16, 87, 100, 108, 173, 253, 262
GEOS file structure 252, 256, 258, 259
GEOS KERNAL 14-15
GEOS User's Guide 7
GEOS V1.0 3, 258
GEOS V1.2 3, 24, 58
geoWrite 17, 20-21, 23-26, 28, 32, 33, 35, 57-66, 72-81, 87, 137-143, 177
ghost icon 15, 20, 23
go to page 64
graphics 36, 56, 62, 64, 68, 76-81, 87, 88, 120, 130, 214, 231
graphs 87-99, 134

H

hardcopy 4, 41, 45, 49, 66, 115, 133, 143, 248
hide pictures 64
highlighting text 141

I

IC (integrated circuit) 109-110
icons 4, 7-9, 12-17, 20-24, 32, 36, 40-43, 45, 49, 50-55, 58, 68-70, 72, 73, 79, 80, 82, 89, 90,
icons (custom) 148-178, 189,
indirect addressing 229, 237
info 35
info box 14, 15, 32, 35, 37, 41, 129, 148, 149, 163, 166-170, 173-175, 177, 189, 204, 256, 265
INFO sector 254-256, 265, 269, 271
invert 51
IRQ (Interrupt Request) 184-186, 214, 244, 245, 277, 278
items (menu) 8, 32

J

job code 226, 228-230, 232, 233, 236, 239
job loop 216, 247-251 272, 277
job structure 247-251
job table 250
joystick pointer 24, 65, 88, 90
jump commands 235

K

KERNAL 178-180, 197-199
keyboard 40

L

Letters 19, 20, 22-24, 28
loading GEOS 9, 11, 18, 24, 35, 58, 60, 64, 72, 125, 126, 128, 214, 256, 258, 264, 266, 267, 272
loading prg's in machine language 129, 197, 198, 203, 253-256
load problems (GEOS) 125

M

margins (geoWrite) 59, 66, 67, 138-142
memory locations (GEOS) 274-278
menu 8, 12-14, 17, 19, 24, 26-28
mirror x 51
mirror y 51
modem 39
monitor (machine language) 146-148, 203-204, 214, 234, 241, 256, 261, 263, 265, 266, 268, 271, 273
monitor (screen) 8
move 51

N

normal edit 48
notepad 34
Notepad 71-73, 144-147

O

open 8, 12, 13, 17, 22-24, 35, 43
operating system 125, 197, 203, 214, 217, 218, 221, 226, 234, 244, 267
options 19, 24, 47, 49, 53, 58, 64, 97, 128, 129, 133, 171

P

page break 64
page map 44
paste 47
patterns (graphic) 57, 89, 90-91, 97, 133-136, 224, 240
PC board 108-110
pencil (geoPaint) 44, 48, 53-54,

75, 89, 101, 121, 132
photo manager 34
Photo Manager 76-79
photo scrap 47
pixel edit 48, 50, 54, 75, 89, 102, 105, 109, 110, 112-118, 133
pixels 46, 48, 52, 54
point sizes (font) 26
pointer 4, 8, 12, 14, 15, 17, 18, 20, 74-77, 277
pointer (editing) 75
pointer speed 17
POKEs 129, 197
position line (geoWrite) 59
Preference Manager 17-18, 34, 68, 73-75, 88
preference mgr 34
preferences 77, 234
preview mode 46, 60, 97, 98, 107, 119, 142
print 36, 46, 61, 121, 130-132, 137, 141, 147, 171, 177, 263
printer 7, 32, 34-36, 41, 45, 46, 49, 60, 61, 87, 100, 108, 126, 130-132, 147
printer (initializing) 130, 160, 214
printing 41, 45, 60, 129-130
program file 33
programming the clock 178

Q

Q-LINK (QuantumLink) 39
quit 43, 60, 128

R

recover 46
rectangles (geoPaint) 55, 120, 121, 132
reformatting (geoWrite) 140
registers 130, 178, 179, 203, 215, 238, 245, 246
rename 13, 35, 38, 46, 61, 218
reset 39
RESET 197, 271
restarting GEOS 126-128, 165
<RESTORE> 8-10, 39, 284
<RETURN> key 7-9, 11, 14
Roma (font) 26, 28, 55
rotating (geoPaint) 118, 147
ruler (geoPaint measurements) 52

S

scaling (geoPaint) 131
schematic diagrams (geoPaint) 111
SCRATCH 129
screen 4, 7-9, 11-13, 15-17
scroll box 37, 43, 47, 48, 50, 59, 67, 84, 94, 106, 115-117, 128
security 147
SEQUENTIAL(file type) 173, 253, 254, 256, 263, 269
select printer 34
select input 34
single-step simulator (SST) 203-217
source disk 9
special 39
sprites 4, 45, 160, 177
starting geoPaint 42
starting GEOS 7-23
starting geoWrite 57
status box 8, 44, 50-54, 56, 90-97, 103, 127, 184, 185, 215, 217, 245, 248-249, 276
status lamp (disk drive) 8
strings 221, 228, 229, 231, 256
sub-menu 8, 12, 13, 19
SWAP file 18, 68, 128, 214, 256, 258, 270-272
system file 33
system IRQ 184-186, 214, 244, 245, 277, 278

T

tabs 14, 15, 59, 66, 67, 141, 143
text cursor 18, 26, 55, 59, 61-67
text (geoPaint) 54
`text manager` 35
Text Manager 23, 80-86
`text scrap` 62
`TIME SET` 18, 76
`To Deb` 25, 79, 80
toolkit (geoPaint) 44, 49-56
track (diskette) 146, 161-163, 167, 173, 199, 252-255, 258, 260, 263-266, 269
track/sector (GEOS diskette format) 146, 263
troubleshooting 126

U

underline 92, 137
`undo` 57
University (font) 55, 102, 110
`update` 46, 60, 61, 248

V

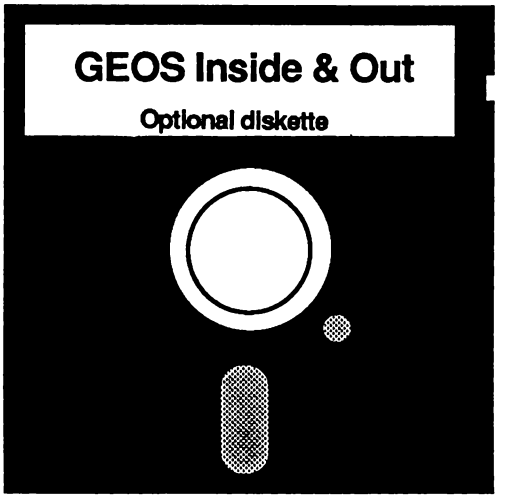
`validate` 38, 129
`VALIDATE` (DOS command) 38, 129
video chip 214, 244
`view` 36
`VLIR` (file type) 146, 173, 253, 256, 263, 264
V1.0 and V1.2 (GEOS versions) 3, 258-261

W

waste basket *iii*, 7, 32
wedging into GEOS 268
windows 8, 10, 13-15, 17-23
window techniques 218-243
windows (custom) 218-243
word wrap 65
`Work 1` 20, 22, 23, 28
work diskette 7, 9-16, 18, 19, 22
write protect tab 14
write protect box 15
WYSIWYG display 137-141

50-60 Hz current 125, 178-179, 197-199
64C *iii*

Optional Diskette



For your convenience, the program listings contained in this book are available on an 1541/1571 formatted floppy disk. You should order the diskette if you want to use the book's programs, but don't want to type them in from the listings in the book.

All programs on the diskette have been fully tested. You can change the programs for your particular needs. The diskette is available for $14.95 plus $2.00 ($5.00 foreign) for postage and handling.

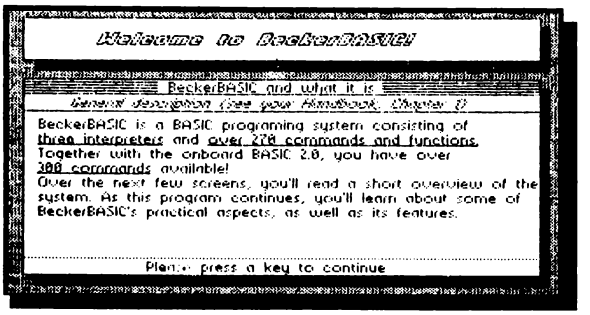
When ordering, please give your name and shipping address. Enclose a check, money order or credit card information. Mail your order to:

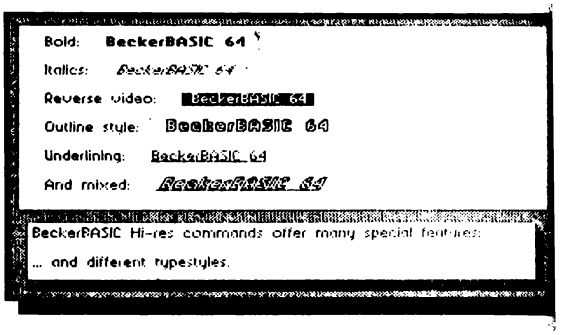
Abacus Software
5370 52nd Street, S.E.
Grand Rapids, MI 49508

Or for *fast* service, call **(800) 451-4319** to order.

BeckerBASIC for GEOS

Now you can write BASIC applications to work with GEOS





Introducing **BeckerBASIC**. If you already know BASIC, you can now write your own GEOS applications in BASIC, easily.

BeckerBASIC gives you the power of over 270 new commands and functions.

Over 20 commands to make your programming easier. For example, TRACE, RENUMBER, DUMP, DIR, etc.

Packed with over 50 commands for easy disk access. Load and save blocks of memory or selected lines of your program. You can even PEEK and POKE into your disk drive's memory.

10 commands can be used for easier cursor control. Turn the cursor on and off. Set how quickly it flashes. Position it at any location on the screen.

20 commands are available for all your hires programming needs. Creat boxes, plot points, and draw lines.

18 additional commands are dedicated to creating sound. Set ring modulation, change the filter, alter the waveform and set the envelope.

Over 35 commands let you create and animate sprites with ease. Load and save sprites directly. Alter their size, change their positions and check for collisions. Use the sprite editor to create sprites and icons.

Use the *Pulldown Menu Construction Set* and *Dialog Box Construction Set* to aid in the creation of you own applications

Royalty-free distribution of your **BeckerBASIC** applications.

Now anyone can create applications in BASIC to run with GEOS. Only $49.95

For credit card orders call 1-616-698-0330

Call today or mail the coupon for your free catalog covering our complete line of software and books for the Commodore 64 and 128. Or ask for the location of the dealer nearest you. You can order direct by phone using your VISA, American Express or MasterCard or detatch and mail your completed coupon. Dealer inquiries welcome—over 2400 nationwide.

Abacus

5370 52nd Street SE
Grand rapids, MI 49508
Telex 709-101 • FAX 616/698-0325

GEOS is a trademark of Berkeley Softworks.
Commodore is a trademark of Commodore Electronics Ltd.

If your Commodore dealer doesn't carry Abacus products, then have him order them for you. Or you can order direct using the following order blank or by calling—(616) 698-0330

Qty.	Product	Price	Total
_____	BeckerBASIC for the Commodore 64	$49.95	_____
In USA add $4.00 for S & H per order. Foreign add $12.00 per item.			_____
Michigan residents include 4% sales tax			_____
Total amount enclosed (US funds)			_____

Payment: ( ) MasterCard ( ) VISA ( ) American Express
( ) Money Order ( ) Check

Card No._____ Exp._____

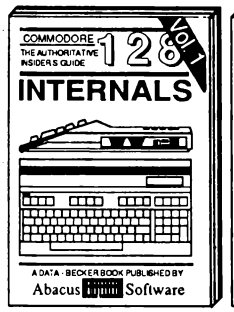
Name_____

Address_____

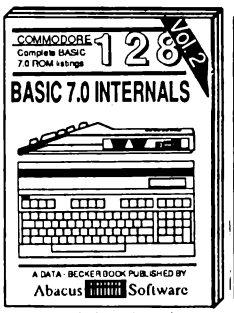
City_____ State_____ Zip_____

Phone No._____

Top shelf books from Abacus



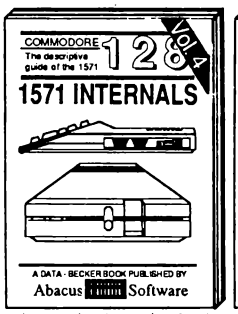
Detailed guide presents the 128's operating system, explains graphic chips, Memory Management Unit, 80 column graphics and commented ROM listings. **500pp $19.95**



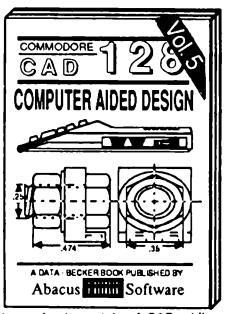
Get all the inside information on BASIC 7.0. This exhaustive handbook is complete with commented BASIC 7.0 ROM listings. Coming Summer '86. **$19.95**



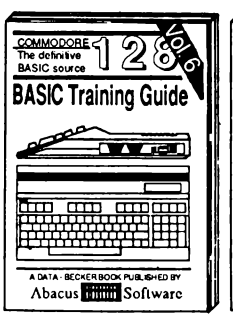
Filled with info for everyone. Covers 80 column hi-res graphics, windowing, memory layout, Kernal routines, sprites, software protection, autostarting. **300pp $19.95**



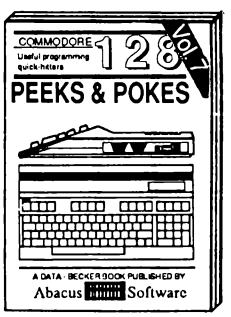
Insiders' guide for novice & advanced users. Covers sequential & relative files, & direct access commands. Describes DOS routines. Commented listings. **$19.95**



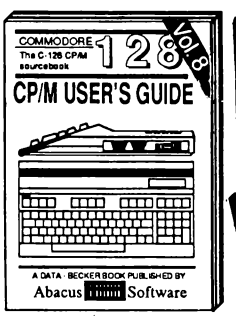
Learn fundamentals of CAD while developing your own system. Design objects on your screen to dump to a printer. Includes listings for '64 with Simon's Basic. **300pp $19.95**



Introduction to programing; problem analysis; thorough description of all BASIC commands with hundreds of examples; monitor commands; utilities; much more. **$16.95**



Presents dozens of programming quick-hitters. Easy and useful techniques on the operating system, stacks, zero-page, pointers, the BASIC interpreter and more. **$16.95**



Essential guide for everyone interested in CP/M on the 128. Simple explanation of the operating system, memory usage, CP/M utility programs, submit files & more. **$19.95**



ANATOMY OF C-64 Insider's guide to the '64 internals. Graphics, sound, I/O, kernal, memory maps, more. Complete commented ROM listings. **300pp $19.95**

ANATOMY OF 1541 & DRIVE Best handbook on floppy explains all. Many examples and fully commented 1541 ROM listings. **500pp $19.95**

MACHINE LANGUAGE C-64 Learn 6510 code write fast programs. Many samples and listings for complete assembler, monitor, & simulator. **200pp $14.95**

GRAPHICS BOOK C-64 - best reference covers basic and advanced graphics. Sprites, animation, Hires, Multicolor, lightpen, 3D-graphics, IRQ, CAD, projections, curves, more. **350pp $19.95**

TRICKS & TIPS FOR C-64 Collection of easy-to-use techniques: advanced graphics, improved data input, enhanced BASIC, CP/M, more. **275pp $19.95**

1541 REPAIR & MAINTENANCE Handbook describes the disk drive hardware. Includes schematics and techniques to keep 1541 running. **200pp $19.95**

ADVANCED MACHINE LANGUAGE Not covered elsewhere: - video controller, interrupts, timers, clocks, I/O, real time, extended BASIC, more. **210pp $14.95**

PRINTER BOOK C-64/VIC-20 Understand Commodore, Epson-compatible printers and 1520 plotter. Packed: utilities; graphics dump; 3D-plot; commented MPS801 ROM listings, more. **330pp $19.95**

SCIENCE/ENGINEERING ON C-64 In depth intro to computers in science. Topics: chemistry, physics, biology, astronomy, electronics, others. **350pp $19.95**

CASSETTE BOOK C-64/VIC-20 Comprehensive guide; many sample programs. High speed operating system fast file loading and saving. **225pp $14.95**

IDEAS FOR USE ON C-64 Themes: auto expenses, calculator, recipe file, stock lists, diet planner, window advertising, others. Includes listings. **200pp $12.95**

COMPILER BOOK C-64/C-128 All you need to know about compilers: how they work; designing and writing your own; generating machine code. With working example compiler. **300pp $19.95**

Adventure Gamewriter's Handbook Step-by-step guide to designing and writing your own adventure games. With automated adventure game generator. **200pp $14.95**

PEEKS & POKES FOR THE C-64 Includes in-depth explanations of PEEK, POKE, USR, and other BASIC commands. Learn the "inside" tricks to get the most out of your '64. **200pp $14.95**

Optional Diskettes for books For your convenience, the programs contained in each of our books are available on diskette to save you time entering them from your keyboard. Specify name of book when ordering. **$14.95 each**

C-128 and C-64 are trademarks of Commodore Business Machines Inc.

Abacus Software

5370 52nd Street SE Grand Rapids, MI 49508 Phone (616) 698-0330

Optional diskettes available for all book titles - $14.95 each. Other books & software also available. Call for the name of your nearest dealer. Or order directly from ABACUS using your MC, Visa or Amex card. Add $4.00 per order for shipping. Foreign orders add $10.00 per book. Call now or write for your **free** catalog. Dealer inquires welcome--over 1400 dealers nationwide.

Expand your vocabulary—Take on a new language

Features of ***Super Pascal***:

- full implementation of Jensen & Wirth Pascal
- C-64 high-speed DOS (3X faster)
- includes many language extensions for systems programming
- integrated assembler for machine code requirements
- built-in editor with renumber, find, auto, change, append, delete
- includes fast graphic library
- large 48K workspace
- works with one or two drives
- advanced error handling
- C-128 version supports 80-column hires graphics
- with 220-page manual

Super Pascal

Your complete system for developing applications in Pascal. ***Super Pascal*** is a complete implementation of standard Pascal (Jensen and Wirth). C-64 version has a high-speed DOS (3X) that makes using it quick and efficient. The extensive editor (source included) contains added features: append files, search and replace. Includes an inline assembler for optionally coding in machine language. ***Super Pascal*** is so capable that it's used in hundreds of schools to teach Pascal. But it can be used for more than just learning Pascal, use it for serious programming. The graphic library (source included) is written in machine language for fast execution. Want to learn Pascal or develop software using the best tool available? ***Super Pascal*** is your first choice.

for the C-64 **$59.95**
for the C-128 **$59.95**



Features of ***Super C***:

- compiles into fast machine code
- built-in editor with search, replace, block commands, more
- supports strings and arrays
- handles object code up to 53K
- supports recursive programming techniques
- Libraries included:
  - standard I/O library
  - 25+ command graphic library
  - math library with trig functions
- C-128 version supports RAM disk and 40/80 column modes
- works with one or more drives
- with 275-page manual

Super C

C is one of today's most popular languages. It's easy to transport C source code from one computer to another. With ***Super C*** you can develop software or just learn C on your Commodore. ***Super C*** takes full advantage of this versatile language. Produces 6502 machine code and is many times faster than BASIC. Includes a full-screen editor (search, replace and block operations), compiler, linker and handbook. You to combine up to seven modules with the linker. Besides the standard I/O library, a graphic library (plot points, draw lines, fill in areas) and a math library (sin, cos, tan, log, arctan, more) are included. Whether you just want to learn C, or program in a serious C environment for your Commodore, ***Super C*** is the one to buy.

for the C-64 **$59.95**
for the C-128 **$59.95**



Abacus

Abacus Software • 5370 52nd Street SE • Grand Rapids, MI 49508 • Phone (616) 698-0330

Call **now** for the name of the dealer nearest you. Or order directly using your MC, Visa or Amex card. Add $4.00 per order for shipping. Foreign orders add $12.00 per item. Call (616) 698-0330 or write for your **free** catalog. 30-day money back software guarantee. Dealers inquires welcome–over 2000 dealers nationwide.

64 or 128

in·no·va·tion ***n.*** **1. the process of making changes 2. a new method, custom, device, etc. 3. Cadpak 4. BASIC Compiler** ***see*** **Abacus**

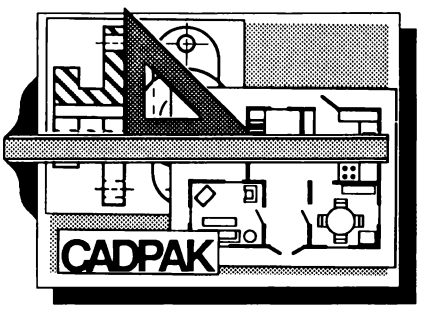
Features of ***Cadpak:***

- dimensioning for exact scaled output
- works with lightpen or keyboard
- copy artwork between two screens
- print in two sizes-full or 1/4 page
- add text in four different sizes—three special fonts included
- includes three tutorials
- create librarys of objects with Object Editor–math and electronic included
- design your own pattern fills
- size, rotate and reflect objects
- work in metric or english sizes
- includes USA map outline
- C-128 added features:
  - draw curves with up to 11 points
  - create templates which you can size or rotate at any degree
  - larger work area (640x360 pixels)

Cadpak

Design pictures and graphics quickly and precisely. Unlike other drawing programs, you can produce exact scaled output on your printer. Design in the units of your drawing problem (feet, miles, meters,etc.) and send hardcopy to most printers. Uses either the keyboard or a lightpen. Two separate work screens—transfer artwork from one screen to the other. Place text in four sizes anywhere in the picture—three extra fonts included: Old English, 3-D and Tech. "Try Again" functions allows you to undo a mistake. Solid or dashed lines, circles, ellipses at any angle, rays and boxes. Save and edit fill patterns, fonts and objects. ***Cadpak*** is the full-featured design package for your Commodore computer.

for the C-64 **$39.95**
for the C-128 **$59.95**



Design an addition to your house...
Remodel your apartment...
Create a layout for your garden...
Draw schematic diagrams...
Engineer a new widget...
all to scale!

Features of ***Basic Compiler:***

- compile to machine language, speedcode or a combination of both for added flexability
- supports overlay structures for large programs
- extensive compiler directives allow parameter changes during compilation
- use BASIC extensions such as Simon's BASIC, Video Basic, BASIC 4.0, VICTREE and others
- compile your programs to speed them up and to protect them from LISTing or altering
- C-128 version added features:
  - works with BASIC 7.0
  - works with FAST mode
  - make use of all 128K

BASIC Compiler

The complete compiler and development package. Make your programs 3 to 35 times faster. Compile you programs into machine language, speedcode (pseudo code) or a combination of both. With the overlay feature you can compile a series of programs that load in consecutively. Even compiles programs written with BASIC extensions (Simon's BASIC, VICTREE, BASIC 4.0, VideoBasic, others). Control memory management to give you total control over your compiled program. When the compiler recognizes an error, it continues to find other errors. C-128 version is completely compatible with BASIC 7.0, works with FAST mode and allows for use of all of 128K of RAM. If your program walks or crawls, give it the speed to RUN!

for the C-64 **$39.95**
for the C-128 **$59.95**



Abacus

Abacus Software • 5370 52nd Street SE • Grand Rapids, MI 49508 • Phone (616) 698-0330

Call **now** for the name of the dealer nearest you. Or order directly using your MC, Visa or Amex card. Add $4.00 per order for shipping. Foreign orders add $12.00 per item. Call (616) 698-0330 or write for your **free** catalog. 30-day money back software guarantee. Dealers inquires welcome–over 2000 dealers nationwide.

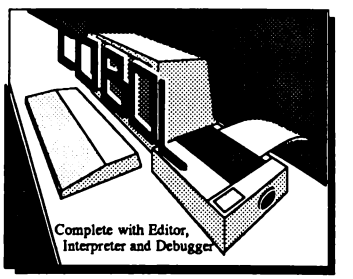
Make Your Choice

C-64 or C-128

BASIC Compiler
Complete BASIC compiler and development package. Speed up your programs 5x to 35x. Compile to machine code, compact p-code or both. '128 version: 40 or 80 col. monitor output and FAST-mode operation. '128 version includes extensive 80-page programer's guide. A great package that no software library should be without. **C-64 $39.95**
C-128 $59.95



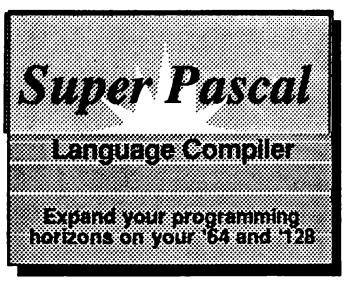
COBOL
Now you can learn COBOL, the most widely used commercial programming language, on your 128 or 64. COBOL Compiler package comes complete with syntax-checking editor, interpreter and symbolic debugging aids. New '128 version works with 40/80 column monitors and is quicker than the '64 version. **C-64 $39.95**
C-128 $59.95



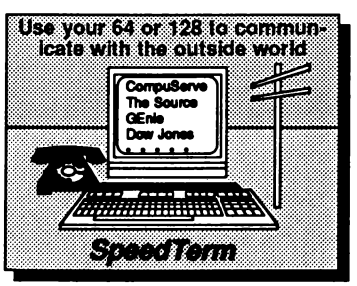
Super C Compiler
For school or software development. Learn the C language on the '64 or '128. Compiles into *fast* machine code. Added '128 features: CP/M-like operating system; 60K RAM disk. Combine M/L & C using CALL; 51K available for object code; Fast loading; Two standard I/O librarys plus math & graphic libraries. **C-64 $59.95**
C-128 $59.95



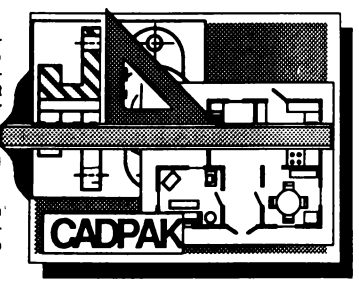
New C-128 Version!
Super Pascal Compiler
Complete system for developing applications in Pascal. Extensive editor. Standard J & W compiler. Graphics library. Added '128 features: RAM disk; 100K source/one drive or 250K/two; 80/40 column. If you want to learn Pascal or develop software using the best tool available, Super Pascal is your first choice. **C-64 $59.95**
New! C-128 $59.95



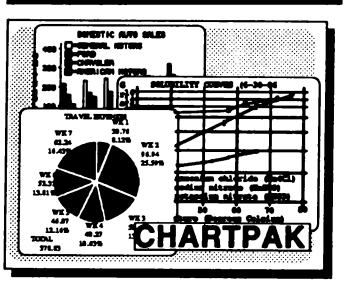
New C-64 Version! ***Speedterm***
Let your 64 or 128 communicate with the outside world. Obtain information from various computer networks. Flexible, command driven terminal software package. Supports most modems. Xmodem and Punter transfer protocol. VT52 terminal emulation with cursor keys, large 45K capture buffer & user definable function keys.
New! C-64 $39.95
C-128 $39.95



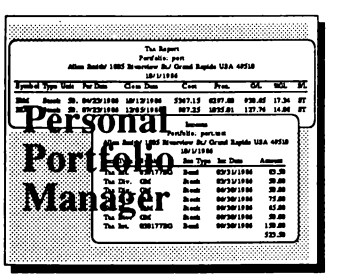
Cadpak
Easy-to-use interactive drawing package for accurate graphic designs. Dimensioning features to create exact scaled output to all major dot-matrix printers. Input via keyboard or lightpen. Two graphic screens for COPYing from one to the other. DRAW, BOX, ARC, ELLIPSE, etc. available. Define your own library of symbols/objects–store up to 104 separate objects. **C-64 $39.95**
C-128 $59.95



Chartpak
Create professional quality charts *fast*— without programming. Enter, edit, save and recall data. Interactively build pie, bar, line or scatter graph. Set scaling, labeling and positioning. Draw charts 8 different formats. Statistical routines for average, standard deviation, least squares and forecasting. Use data from spreadsheets. Output to most printers. **C-64 $39.95**
C-128 $39.95



PPM
Comprehensive portfolio management system for the 64 and 128. Manage stocks, bonds, mutual funds, T-bills; record taxable or non-taxable dividends & interest income; reconcile each brokerage account cash balance with the YTD transaction file; on-line quotes through Dow Jones or Warner. Produces any type of report needed to analyze a portfolio or security. **C-64 $39.95**
C-128 $59.95



Call **now** for the name of the dealer nearest you. Or order directly form Abacus using your MC, Visa or Amex card. Add $4.00 per order for shipping. Foreign orders add $10.00 per item. Call (616) 241-5510 or write for your **free** catalog. 30-day money back software guarantee. Dealers inquires welcome–over 1500 dealers nationwide.

Abacus

Abacus Software • 5370 52nd Street SE
Grand Rapids, MI 49508 • Phone (616) 698-0330

GEOS INFO



Another Abacus Best Seller!

GEOS Inside and Out

If you use GEOS then our new book, ***GEOS Inside and Out***, has the info you need.

A detailed introduction is laid out for the novice–beginning with how to load the GEOS operating system...how to create a backup...how to alter the preference manager...how to format disks...learn geoWrite and geoPaint in detail...use geoPaint for designing floor plans or drawing electronic diagrams. Easy-to-understand examples, diagrams and glossary are included to enlighten the beginner.

The advanced user will find more detailed information on GEOS's internals and useful tricks and tips. Add a constant display clock–includes assembly and BASIC listing...complete listing of our **FileMaster** utility (converts your programs to GEOS format with an icon editor) with a line by line explanation...create a single-step simulator for observing memory and the various system registers...learn about windows and how to use them to your advantage...understand GEOS file structure.

If you're just getting started with GEOS or getting to the point of wanting to add your own applications, then ***GEOS Inside and Out*** will help you on your way. $19.95

Coming Soon!

GEOS Tricks & Tips

Continuing the tradition established by our famous C-64 reference library, ***GEOS Tricks & Tips*** is a collection of helpful techniques for anyone who uses GEOS with their Commodore. It's easy to understand without talking down to the reader, and detailed in the applications of the routines. Includes a font editor to create up to 64 point text and a machine language monitor. A perfect companion volume to ***GEOS Inside and Out***. Available Second Quarter. $19.95

GEOS, geoWrite, geoPaint are tradenames of Berkeley Softworks.

To receive your copy of ***GEOS Inside and Out*** and/or ***GEOS Tricks & Tips***, call now for the name of the dealer or bookstore near you. Or order directly using your Visa, MC or Amex card. Add $4.00 per order for shipping and handling. Foreign orders add $10.00 per book. Call or write today for your free catalog. Dealer inquires welcome—2000 nationwide.

Order both today!

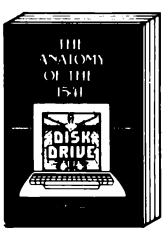
You Can Count On **Abacus**

5370 52nd Street SE
Grand Rapids, MI 49508
Phone (616) 698-0330

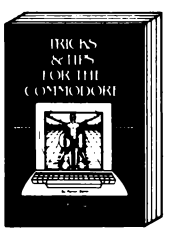
Just a few of our books...



Anatomy of the C-64
Insider's guide to '64 internals. Graphics, sound, I/O, kernal, memory maps, and much more. Complete commented ROM listings. 300pp $19.95



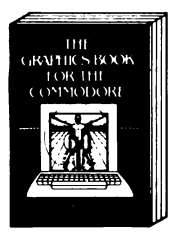
Anatomy of the 1541 Drive
Best handbook on this drive, explains all. Filled with many examples programs, utilities. Fully commented 1541 ROM listings. 500pp $19.95



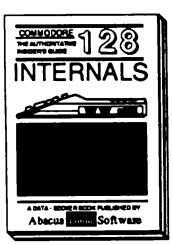
Tricks & Tips for the C-64
Collection of easy-to-use techniques: advanced graphics, improved data input, CP/M, enhanced BASIC, data handling and more. 278pp $19.95



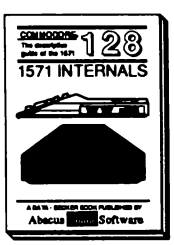
Peeks & Pokes for the 64
Includes in-depth explanations of PEEK, POKE, USR, and other BASIC commands. Learn the "inside" tricks about your '64. 200pp $14.95



Graphics Book for the C-64
Best reference, covers basic and advanced graphics. Sprites, Hires, Multicolor, 3D-graphics, IRQ, CAD, projections, curves. 380pp $19.95



C-128 INTERNALS
Important C-128 information. Covers graphic chips, MMU, I/O, 80 column graphics and fully commented ROM listings, more. 500pp $19.95



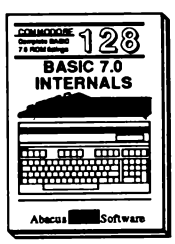
1571 INTERNALS
Essential reference. Internal drive functions. Explains various disk and file formats. Fully-commented ROM listings. 450pp $19.95



C-128 TRICKS & TIPS
Fascinating and practical info on the C-128. 80-col hires graphics, bank switching. 300 pages of useful information for everyone. $19.95



C-128 PEEKS & POKES
Dozens of programming quick-hitters. techniques on the operating system, stacks, zero page, pointers, and BASIC. 240pp $16.95



C-128 BASIC 7.0 Internals
Get all the inside info on BASIC 7.0. This exhaustive handbook is complete with fully commented BASIC 7.0 ROM listings. $24.95

Call now for the name of your nearest dealer. Or order direct with your credit card by calling 616/241-5510. Add $4.00 per order for S&H. Foreign add $12.00 per book.

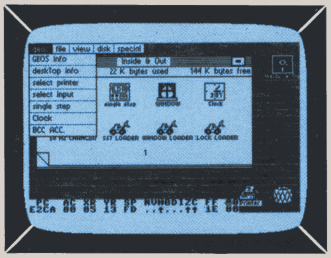
Other books and software also available. Call or write for your free catalog. Dealers inquires welcome—over 2000 nationwide.

Commodore 64 and Commodore 128 are trademarks of Commodore Ltd.

Abacus

Abacus Software
5370 52nd Street SE
Grand Rapids, MI 49508
Phone (616) 698-0330

GEOS INSIDE AND OUT



Single-step through GEOS's internal workings



Use FileMaster to convert your programs to GEOS format (includes icon editor)



Create a clock that you can display constantly in the corner

Whether you're a beginner or a seasoned computer user, if you use GEOS, then ***GEOS Inside and Out*** contains information you need. This book begins with a complete, thorough introduction to GEOS. You'll also learn how to add your own applications to GEOS. It contains the listing for *FileMaster*, so you can convert your programs to GEOS format and create an icon for them. ***Geos Inside and Out*** will help you:

- Master GEOS by following the detailed introduction
- Use your own programs in GEOS (includes icon editor)
- Display memory and the registers with the single-step simulator
- Add windows to your own programs
- Learn about the GEOS file format
- Study memory layout and important memory locations

About the authors:
Manfred Tornsdorf is a physics major who became interested in computers through his hobby—electronics. He is the author of a popular European book *Textomat Plus Tricks and Tips*. Ruediger Kerkloh has owned a Commodore 64 for many years. His specialties are peripheral devices and assembly language programming for peripherals.

ISBN 0-916439-81-X

GEOS is a trademark of Berkeley Softworks

Abacus



This was brought to you

from the archives of

http://retro-commodore.eu