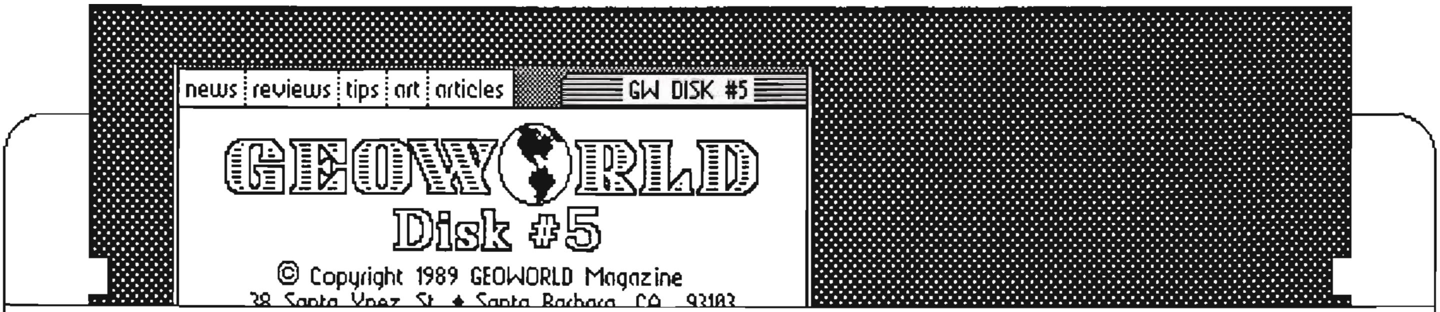
geoWorld

tive az about GEOS

Issue #22


news
reviews
tips
art
articles
GW DISK #5
GEOWORLD
Disk #5
© Copyright 1989 GEOWORLD Magazine
38 Santa Ynez St. ♦ Santa Barbara, CA 93103

The holiday season is fast approaching, and our fifth disk fits right in with the festivities. There is finally a music program for GEOS - *geoSidPlayer* by Roger Lawhorn. We've included some Christmas songs to get you in the spirit.

There's lots more clip art, including autumn and Christmas art for greeting cards, party invitations, and flyers.

A large number of really good fonts have been created by Thomas Dively. Some have unique connecting characters and art elements for more creativity. The first group of his *Starfonts* is on disk 5.

Jean Major's two *geoAlbum* programs provide a simple way to make screen dump photo scraps. One version makes a full-screen scrap for geoPublish, and the other breaks the image into four scraps for geoPaint.

To organize all those photo scraps, even between drives, is Rick Coleman's *Photo Mover*. After using this program, the photo manager seems so slow.

For those of you with Apple GEOS, or with friends that have Apples, we've included Terry Van Camp's conversion programs discussed on page 20. Now it is possible to exchange fonts, graphics and data files between these GEOS versions.

Disk five also has upgrades of several favorites, including the 128, 80 column version of John Howard's *Quiktop*.

To help you learn about color separation and how we did the color covers, the disk includes all the color plates from the Roger Eller cover #21, and of this issue.

Payton Snyder of Arizona has upgraded his *WormDesk* program and has come up with a great graphic viewer/converter called *PicShow*. His *FontSwap* lets you change the available selection of fonts while in the 64 version of geoWrite 2.1

Each GEOWORLD disk is filled with meaningful files on both sides. The graphics, fonts, utilities and applications we've been able to put on the disks are what makes the basic GEOS software so much more practical and enjoyable.

Give yourself a gift of one or all of the GEOWORLD disks with their great variety of programs. Then, when you discover the ones that you will really use over and over, send a gift of money to the programmers whose works you would like to encourage.

ORDER DISK #5 TODAY:

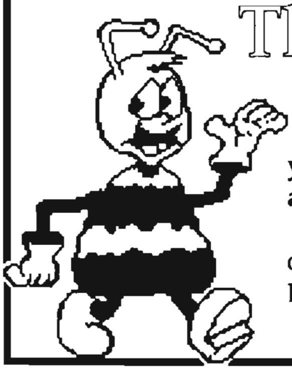
Disks 1 through 4 are still available. Please indicate which numbers you want.

Send $10. check or money order with your request to:

GEOWORLD DISK
38 Santa Ynez Street
Santa Barbara, CA 93103

Canada & Mexico add .50 per disk. Foreign orders,

This is Pippin!



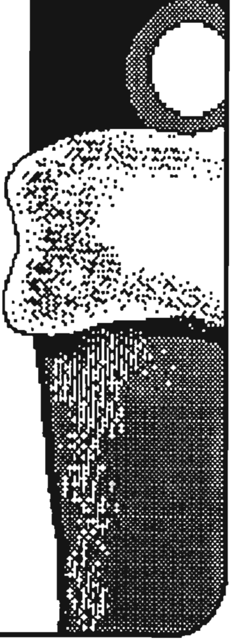
And you can help create him using GEOS 2.0! Have fun putting together your own stories with his **body, faces, arms, legs,** and **backgrounds!**

You can get him by sending a check or money order for **$6.00** to:

Dale Beach/7048 Michigan St.
Elwell, MI 48832

Stuff It!

NOV/DEC 'TOONS!



Here are some cartoons on disk that would make great stocking stuffers! There are 7 disks to choose from, at **$6.00 each.** While any of the disks would make great gifts for anyone who uses a C-64 or -128, and GEOS, the **NOV-DEC Disks** have some holiday art on them!

So, get 'em .... enjoy 'em .... and have a Happy Holiday!

#1)NOV-DEC 88 #2)JAN-FEB 89
#3)MAR-APR88 #4)MAY-JUN89
#5)JUL-AUG 88 #6)SEP-OCT 89
#7)NOV-DEC 88

Send check or M.O. ($6.00 per disk)
to:
Dale Beach/7048 Michigan St
Elwell, MI 48832

GEOWORLD

The definitive magazine about GEOS

Issue Number 22

GEOWORLD is not a monthly publication. Please check the mailing label for when your last issue will be sent.

GEOWORLD
38 Santa Ynez Street
Santa Barbara, CA
93103

Roger Ledbetter
Publisher

Susan Puhn-Lamb
Editor
3575 E. County 18th Street
Yuma, AZ 85365

GEOWORLD is published and printed entirely from the homes of the Editor and Publisher using Commodore 128, GEOS, geoPublish, geoPubLaser, geoTerm, modem and a postscript laser printer.

Articles may be submitted in geoWrite format with artwork in geoPaint pages or photo scraps on GEOS disks. Authors will receive a GEOWORLD PD disk and $15. per page for material published.

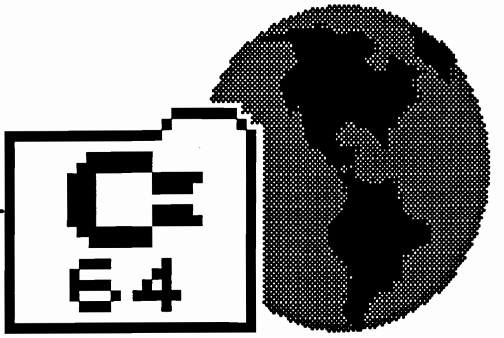
Subscription
(Mail order only)
12 issues - $20.
1 issue - $2.50
Canada - $30. Overseas - $50.

United Kingdom Distributor:
Financial Systems Software LTD.

About The Cover... The Santa Clause on the front cover was found in a Christmas Clip-Art book. I took this and placed Santa in a computer-oriented environment. I then enlarged the design 200%, separating the artwork into four 8x10 panels. Each panel was scanned on a Macintosh scanner into separate MacPaint files. These files were then converted to geoPaint files using MacAttack II. Using Paint-Scrap, I took these panels into geoPublish, where I re-assembled the design. By selecting 300 dpi smoothing & scaling, The resolution was thus improved. From this point on, a time & labor- intensive procedure was engaged to aquire the color-separation for 4-color process printing. This procedure will be outlined in detail in a later issue of GEOWORLD. Until then, Merry Christmas & a Happy geo-New-Year! Roger Eller

Integrating GEOS

Still Further Along the BerkSoft Path:



In which the Author, putting two and two together, comes up with Utopia . . .
By Conrad Tillman

System Integration Revisited

Previously, I discussed the functionality of GEOS v2.0 as system integrating software. Believe it or not, this time I started out to deal with the subjects of software piracy and copy-protection, as they relate to putting together an acceptably consistent system. These subjects are still mentioned somewhere below, but in the space between my brain and the monitor, (while reviewing some personal and common history), the copy shifted toward the province of the crystal ball. The result is the following speculation concerning the future of BSW, GEOS, and Commodore computing.

Recent History

It has been discouraging, these past few months, to open the pages of most any publication catering to the home computer users. Those which are not Commodore specific have already consigned our machines to the DELETE file, mentioning them only in comedic asides, or in the context of games, positioning the C-64 against the likes of *Nintendo*. Those which depend on our particular market promise to continue to provide support, and to provide it for "as long as the sun shall shine and the wind shall blow." This position is not particularly reassuring.

Although everyone seems to agree that GEOS has effectively extended the life of the Commodore eight-bits, few major software publishers appear to be taking advantage of the extension. Most seem to regard GEOS as just another integrated software system and depend on the speed advantage of their own text-based screens to compete. If GEOS was *only* integrated software, this position would make sense. Text-based, wholly memory resident software is, by the nature of the machinery, faster than GEOS' graphic renditions.

GEOS, is much more than merely integrated software. It is even much more than the powerful system integrating software I described in my previous article. GEOS is a full-fledged operating system which only its developers and a few small independents seem to be intent on utilizing. In full-blown applications, (while there is some choice in word processors), only BSW offers such necessities as a database & spreadsheet. For a developer of the demonstrated stature of BerkSoft, these applications seem strangely flawed, lacking features that serious users consider essential.

In the face of this serious lack of features, even as the Q-Link boards sometimes bristled with irritation toward the company, BSW appeared to ignore the concerns of its loyal customers, choosing instead to expand into the ripening orchard of Apple II, and announcing that their next order of business would be the IBM PC.

Sure, they polished up the DeskTop, added a lot of features to Commodore GEOS v2.0, and put a higher sheen on geoWrite 2.1. They did improve certain desk accessories as well, and fixed some flaws in geoPaint. But geoCalc and geoFile are important, and have received little attention in years other than bug exterminations.

People have openly and angrily accused Berkeley Softworks of the kind of greed and ruthless ambition that was rampant a few years ago in the home computer industry, when developers were often perceived to have an attitude, expressed as: "Take the money and run."

This perceived attitude has been blamed for the Commodore world's reputation for widespread software piracy, the prevalence of which puts copy-protection barricades between software and its most efficient use.

The question in my mind is whether it is reasonable to ascribe this attitude to BSW. Besides the aforementioned tweaks to the heart of the Commodore GEOS system, BerkSoft has exhibited a remarkable degree of support for its customers, with a liberal upgrade policy featuring *genuine* upgrades, and free replacement of programs discovered to have bugs without regard to the length of time it may have taken a user to encounter them.

I have a fairly busy schedule, and a new software purchase often goes unused for weeks or even months. If it is a BSW product, no sense of lurking urgency requires an immediate bug-search before the expiration of some arbitrary time limit. It may take a while, but Customer Service responds to its mail and to user demand, as evidenced by recent changes in their Q-Link presence.

Ambition

Berkeley Softworks, like Apple Computer, will probably not be accused of exhibiting any great deal of humility. There is ambition at work, to be sure, an ambition stated by BSW founder Brian Dougherty from the very beginning. The stated goal of Berkeley Softworks is the acceptance of GEOS as a *standard operating system* in the Big Three: Commodore, Apple, and IBM eight-bit computers. The motivation for this goal appears to have many people confused.

Not once have I heard of BerkSoft expressing any desire to dominate the retail software markets for any of these millions and millions of still-in-service computers. Were that their goal, they could have stuck with

Commodore and improved and expanded their line of applications, rather than expanding into other markets. Many have expressed the opinion that BSW ought to have followed precisely this course.

Although the uniting of these relatively outdated computers under a consistent, modern user interface is a worthy social event, the true significance of GEOS is in its consistent, modern *programming* interface to these machines. BSW is a business with a product to be sold. I believe that they intend to stay in business for a very long time to come. Although they fully intend to exploit the fact that millions of owners of these older computers are not ready to abandon them, their strategy is not to unload half-baked software on that market, or to dominate any particular segment of it.

Systems Integration

A basic axiom in the business world is this: "Recognize a need, then fill it." Here is a corollary to that axiom: "If no need exists, first create one, then fill it."

My guess is that, after the software crash of '84 and a general shakeout in the computer industry, BSW had a software development system that they still wanted to sell. They intended to sell their system, designed to work for the production of eight-bit software, to an industry skeptical of the demand for that software. The eight-bit market, being fragmented, scattered, and diminished in size and growth potential, was perceived to be too expensive to develop application software for, since development had to proceed in triplicate for the Big Three's incompatible operating systems.

Programmers may object to this over-simplification, but the GEOS Kernel is essentially a machine specific translator for the implementation of non-machine specific code. In other words, using BerkSoft's development system, a software developer would no longer have to write three complete versions of a program in order to sell it to the eight-bit market. If they are successful in implementing GEOS installations throughout the Big Three, BSW will have integrated these "outdated", incompatible computers into a single, potentially profitable marketing entity.

Risky Business

Anyone who has ever bought a useless piece of software, an incompatible computer gadget, developed a piece of unrewarding shareware, or written a rejected article for a magazine knows something of the risk involved in the home computer game. I leave it to you to imagine the risks of developing, and supporting, a commercial software package. Unless I miss entirely, the contents of the GEOS comment boards on the various 'Links are part of a package that BSW will provide to the major software developers, providing them with years of user-input about the kinds of software that GEOS users want, and the features they will require. The result of this will be less risk, both for the producer and the consumer of potential new packages, with extensive cross-fertilization of ideas between the users of these formerly disparate machines.

If they succeed, BSW will have as a product the entire eight-bit computer market, and as customers BerkSoft would have the major software developers who previously could see no profit in that market. If they succeed, we as users could soon be awash in a flood of GEOS products, immeasurably improved and polished to a higher sheen by the currents of competition that would develop for the profits to be made. As I understand it, success in this regard does not absolutely require that Apple and PC GEOS be immediate, run-away hits in those markets, only that GEOS be present in a sufficient number of installations to make the system viable.

Electric Dreams

For Commodore users, BerkSoft's rising star could result in still more good news in the form of new hardware options. I recently acquired a TurboMaster four megahertz CPU board. In terms of system integration, it was a step backward because it lacks compatibility, at the present time, with the ram expansion unit. Its effect on GEOS, however, must be seen to be believed. The major drawback to C-64 GEOS, of course, is the time required to draw and re-draw its bit-mapped screens. At four MHz, that time is virtually eliminated. I don't presently use it in GEOS, though, because the effect of an REU is more beneficial overall. I have it from a reliable source, that there is high-level interest at BSW in this circuitry. If it were REU compatible, and optimized for GEOS, well, that may just be fantasy. Maybe if interested GEOS users made their interest known . . .

But speaking of fantasy, suppose that empty socket on the REU were put to use. As I understand it, the hierarchy of GEOS design does not rely on brand-specific disk routines and formats. With adequate demand, could someone stuff a hard-drive DOS into an EPROM, enabling the use of low-cost IBM compatible hard-drives interfaced through the parallel port? Alternately, how about an EPROM containing the GEOS Kernel and the DeskTop, now that these appear to be reaching completion? How about, before that step, the implementation of a disk-cache in the REU for the 1581?

Computing By Subscription

When I add up the costs of all the enhancements and expansions to my original C-64 and compare the result to what the sum would buy in today's computer market, it is difficult not to feel some twinges of regret. Then I realize that I would never have spent that sum all in a single lump, and likely would have no appreciation for the progress that's been made since my original purchase. My system has been built over a period of years, during which time it has provided both utility and diversion while never demanding an overly substantial financial plunge. Yes, I suppose it could be viewed as a seduction, but, if so, it is by now an accomplished fact. I am not about to abandon it for the chance to start over with machines that far exceed my needs, and whose available expansions appear to approach infinity.

I, for one, am content to compute within a limited environment, so long as its limitations don't overly restrict my movements. GEOS has extended the borders of those limitations, and might potentially remove them from sight for users like me who don't depend on their computers for the production of income.

Patience and Other Virtues

Many have complained about the lack of copy-protection on Apple GEOS. Some will doubtlessly gripe about how Commodore users had to foot the bill for BSW's ultimate success through their years of devotion. Before you complain too loudly, though, consider this: Apple II and the IBM PC, in 1986, were in no danger of extinction, and we C-64 users still had to tune in channel 15 to accomplish anything with our disk drives.

I personally would not object if IBM GEOS, when available, is distributed *free* in every issue of every PC magazine, if that is what it takes to hasten the advent of the competitive development of productive GEOS software for the C-64/C-128. It surely won't happen overnight. If it does happen, though, the wait will most certainly have been worthwhile.

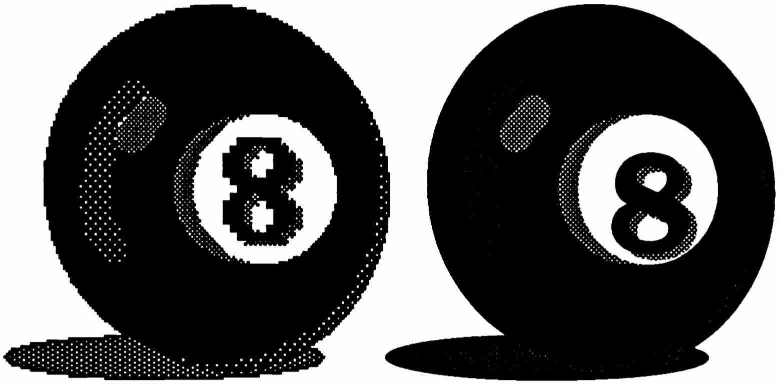
One more thought, on the subject of copy-protection: if I remember correctly, GEOS v1.0 had only minimal protection that any nibbler could duplicate. Why do you suppose that changed? If I were a software developer contemplating selling to this market, I think I'd check with the authors of worthwhile shareware to see how they were doing before I made any decisions on copy-protection.

In Summary

Whether or not my guess concerning BSW's software development system is correct, the purpose of GEOS *is* the integration of the eight-bit computer market into a single entity. This would increase the potential of profit in the production of modern software for that market. Rather than complaining about Berkeley's method of achieving that integration, we all should wish them speed and good luck in their attempt at unification by whatever means it takes. In the meantime, we ought to demonstrate the kind of maturity that will assure software developers that their efforts will be safe from piracy, and in no need of protection. GEOS could be our link to mainstream software developers, but we must present a unified, attractive front for that link to be used to its potential, a front that says collectively, that software worth using is worth paying for.

--Conrad Tillman

Turbo-Master CPU
Schnedler Systems
25 Eastwood Road PO BX 5964
Asheville, NC 28813



Object-Oriented Graphics

By Susan Puhn-Lamb

As an admitted end user, my understanding of how software works is far from technical. I haven't read much about object-oriented graphics and can't tell you for sure what's going on. I do know that a bitmap is made up of pixels like a halftone is made up of dots. It is drawn on the screen, line by line, with every pixel turned on (black) or off (white).

Objects drawn in geoPublish are actually made up of a list of drawing instructions or commands. For example, although you're drawing a line the same as for a geoPaint bitmap, it is stored in the document as a straight line, of a certain thickness, between point x and point y. A rectangle would have four points or coordinates listed for it's description. When you redraw a bitmap, each line of pixels is drawn on the screen. When you redraw a screen of object-oriented art, each element is drawn, one at a time.

Because of the way geoPublish tools draw, they allow for effects that would be difficult or impossible with geoPaint and other graphics programs. A diagonal line can be drawn in various widths and/or patterns, even "white". The ability to draw curved lines (also available in many thicknesses or patterns), is the most significant advance in any graphics software for the C-64 I've seen.

When these object lines are printed on a laser printer, they really look like curves and not little short, straight lines. The 300 dpi LaserWriter prints lines approaching the quality of those drawn with pen and ink. In fact, it is difficult for the untrained eye to detect that the atwork was created by a computer.

You can use also use geoPublish tools to create artwork that will be treated like bitmap graphics by turning the geoPublish page into a geoPaint page with the Paint Drivers. Then, if you print your geoPublish document with an 80 dpi dot matrix printer, the object art looks just like a bitmap geoPaint page (8-ball on the right).

One thing to watch out for when creating graphics for laser printing is the number of objects that can be drawn on a single page. The Master Page holds very few objects, with both special text and bitmaps counting toward the total. The advantage of drawing on the master page is that artwork can be seen in the other modes. Text regions can be plotted around graphics in page layout mode, and tracings may be made in page graphics.

Many more objects can be used in page graphics mode, but there is no library as for the master page. Clip art using a combination of many objects must be saved in individual geoPublish document files rather than the space- saving library files.

One way to increase the number of objects in a document is to break them up among different page modes. Start with the master page, drawing objects until you get the "object list is full" message. Since these items will be seen in other modes, put elements of basic structure and borders here.

Page layout mode can be used to add same/size bitmaps. In this case, regions must first be opened, large enough to contain each bitmap without cutting off a portion.

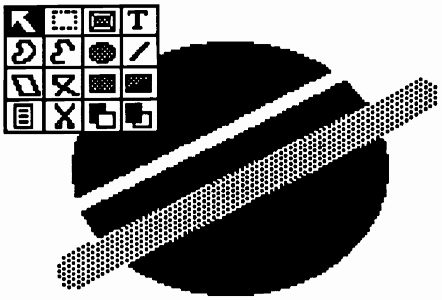
Finally, go to page graphics mode where everything in the other modes is visable and you have another list of objects available.

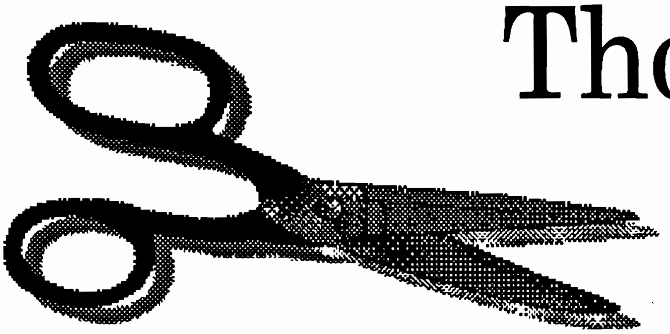
The drawing tools in geoPublish are simple and straightforward. *Line, Circle/Ellipse* and *Rectangle* draw as in geoPaint except that you can set their thickness and pattern with the attributes tool. GeoPaint's circle/ellipse tool has "constrain" to force a round circle (a feature much needed in geoPublish). The *Connected Line* and *Polygon* tools are easy to use, but *Spline* and *Closed Spline* definitely need practice. That is because the line curves between the various points at which you've clicked. You really can't tell if you've drawn a successful line or shape until you double click and the final line is drawn on the screen. When building an illustration of various elements, the *Move to Front or Back* tools will let you combine elements as opaque or transparent.

After creating a graphic made up of several different elements, it can be resized or moved with the *group select* tool. Make sure that you surround all elements or you may end up moving only part of the drawing. If you resize, it may become necessary to change the attributes, such as line thickness. In this case, each element must be individually selected and changed.

It is only after printing with the laser printer that you'll know if the drawing is the way you want it. Some lines not match up and elements might be a little off. With experience you can learn to compensate for the difference of 300 dpi printing and 80 column screen display.

--Susan Lamb





Those Wonderful Photo Scraps

Manipulation of graphics is one area in the GEOS environment that keeps improving.
By Susan Puhn-Lamb

The early lament of geoPaint users was the size limitation of Photo Scraps to that of a drawing window. When GEOS 128 came out, the page-wide scrap was made available, but the height was still limited to 1/5 of a page. During those days, before geoPublish, this limitation didn't really matter.

After Joe Buckley created the *Graphic Storm* and *Import Runner* conversion programs, photo scraps could be made that used the entire screen display. This is the size used by most other graphics programs such as *Doodle*, *Koala* and *Computereyes*, and which can be converted to GEOS format. Once the capability to make the scraps was realized, it was discovered they could be stored in a photo album and used in geoPublish.

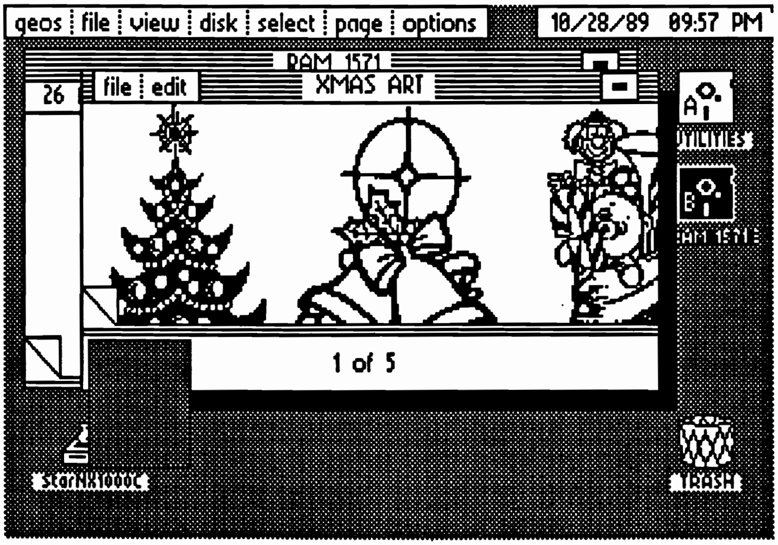
Most computer artists, including myelf, still wished for a way to cut larger portions of a geoPaint page, even the whole page. Early GEOWORLD articles dealt with the painstaking process of moving full page borders and large graphics, scrap by scrap to geoPublish.

Suddenly *BigClipper* by Nick Vrtis appeared on Q-Link. It cuts a photo scrap based on X-Y pixel coordinates of any part of a geoPaint page. His first version (on GEOWORLD Disk #3) will cut a scrap up to 8K. Version 3.0, which can be obtained by a shareware donation will cut oversize graphics or a full page of up to 20K.

Shortly after BigClipper arrived, Dennis Seitz uploaded *Paint-Scrap.* This program copies a full geoPaint page to a photo scrap with one stroke, as long as it is under 28K. It is also available on the GEOWORLD disk with full instructions.

So, after waiting years to make over-size photo scraps, the process is made possible with two different programs, released within a few weeks. The largest scraps can only be used in geoPublish, although one could create a reduction in geoPaint by using the "Stretch and scale to fit" feature before pasting.

Which one of these programs to use depends on the nature of your geoPaint document. If it is a full page border or picture that takes up most of the page, Paint-Scrap is quick and easy. Very complex borders and large pictures can be cut in two or three sections with BigClipper. This is a big improvement over the 15 scraps necessary to move or store a 64 geoPaint page or even five scraps with 128 geoPaint. All of my full page borders can now be efficiently stored in photo albums.



A photo album can hold any size photo scrap, including a full geoPaint page. The shaded box at lower left shows the size of the scrap, although only a portion is displayed.

You can use Paint-Scrap even if the artwork is smaller than the full page. Resizing the box after importing to geoPublish will cut off the unwanted white space. Be sure to always create the artwork as far to the top and left of the page as posible. That is because the area to the right and bottom will be eliminated when resizing.

To use BigClipper you must measure just where the scrap will be cut. If the graphic starts at the very top, your first coordinate will be 0. The bottom of the drawing window is at line 144. The ruler says 143 since the first pixel is called 0, so always add a pixel to each measurement.

After marking the bottom of the window, scroll down and add another 144 lines for each full window length the graphic encompasses. Where the artwork ends within a window, just measure that portion (adding 1 pixel). Therefore, if a graphic starts at the top of the page and is the length of 2 windows and 48 pixels, the top, bottom coordinates would be; 0,336.

To get the horizontal measurement, do the same thing, allowing 264 for each of two window widths and a section 112 pixels wide for a page width of 640 lines.

Once you've learned how to do this accurately, the program is quite easy to use. Keep a log of common coordinates such as 0, 720 x 0, 640 for a full page or 0,200 x 0,320 for a full screen illustration . I always work on a duplicate of the file, marking the window areas to be measured and then clipping from a clean copy.

A new program from Canada has just been released on GEOWORLD Disk #5. It is *geoAlbum* by Jean Major, one of our authors. This program will create screen dumps like the ones used in this article. Version 1.0 puts a large screen-size scrap in an album that can be used in geoPublish. Version 1.1 creates four smaller scraps of the screen that can be used in geoPaint by moving the drawing window around and pasting them next to each other.

There is yet another great new program to create photo scraps on *LOADSTAR* Disk #65, called *geoFetch* by Scott Resh. It can be double-clicked from the DeskTop or opened from the geos menu in most applications. Once the program is activated, the cursor is positioned anywhere on the screen and clicked for the upper left corner of the scrap, dragged to where the lower right corner should be and clicked again.

This program adds a feature to geoPublish that was much needed, that of duplicating something drawn with the tools. Before, this was only possible by converting to geoPaint with the paint drivers. Now, you can draw an object, use geoFetch to make a scrap of it and import it back in without leaving geoPublish. With this procedure, you can create an accurate shadow of any shape drawn.

Retreiver by Ed Flynn gets my vote as one of the most useful GEOS programs written. I receive many geoWrite and geoPublish documents with the photo scraps imbedded and no extra copies. In order to create a new document, I need to retrieve the photo scraps first. The previous procedure was to convert the document to a geoPaint page with the Paint Driver program and re-cut the scraps.

Now, with Retriever, copies are made of the scraps on all pages and placed in a photo album, saving a lot of time and trouble. If any scraps had been resized in geoPublish, they are returned to their original condition.

Another of my favorite new programs, *Photo Mover* by Rick Coleman, remedies a main shortcoming of the Photo Manager -- it's inability to access another drive. This program eliminates the bother and clutter of transferring copies of the albums or the manager from disk to disk. Not only can you move scraps between two drives, but Photo Mover will access the third drive, even if it isn't active on the DeskTop.

Photo Mover makes it possible to leaf through hundreds of album pages and quickly pick the clips you need for a project. Using the photo manager to do this task would require constant opening and closing of the albums, one scrap at a time.

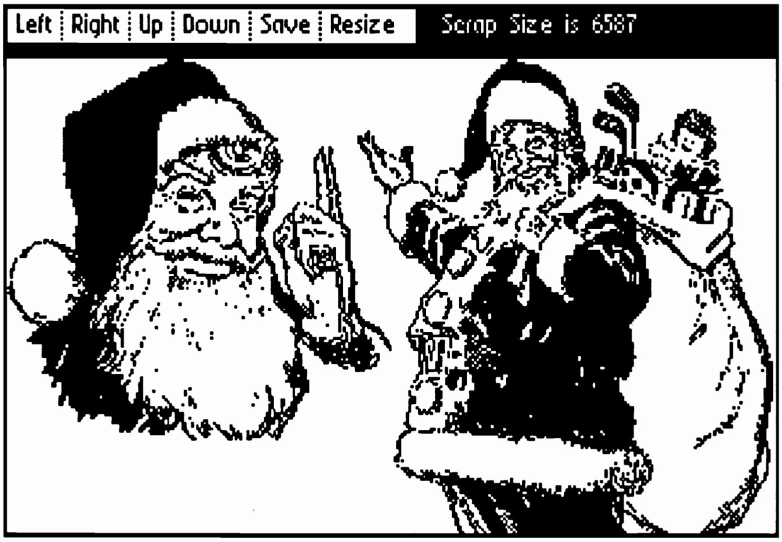
GetGraphic by Nick Vrtis is another time saver. It converts huge collections of Print Shop graphics in one easy operation. After opening or creating an album, the program lists all the graphics on a disk. You can select all the clips on the disk at once, or just highlight those you want. After they are converted and stored, GetGraphic quickly pages through the ones you've chosen.

ScraPeek, another program by Ed Flinn is a handy one to put on all album storage disks. With it, you can view all of the scraps in the albums without the photo manager present. It's also good for looking at the photo scrap on each disk. More than once I have left an important art clip on the temporary scrap and ScraPeek kept me from over-writing with a new one.

Photo Print by Dave Hunt prints a hard copy of all your photo albums, saving time searching through storage disks. The numbering option will help to locate a particular graphic.

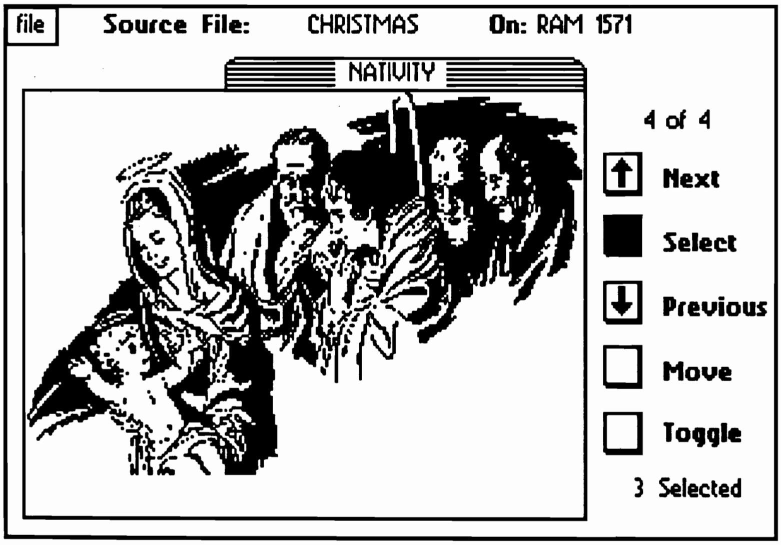
Album Animator by Dennis Seitz flips through the scraps in an album to create a mini motion picture. The albums that come with the program can be printed out in order to study how the action is created. This observation will help you to create your own little movies with geoPaint.

--continued on Page 9



Scrap Size is 6587

BigClipper cuts any part of a geoPaint page into any size photo scrap up to 8K in size. The latest version wll cut a scrap up to 20K.



Source File: CHRISTMAS On: RAM 1571

4 of 4

3 Selected

Photo Mover quickly moves photo scraps between albums. The program will also access other drives.

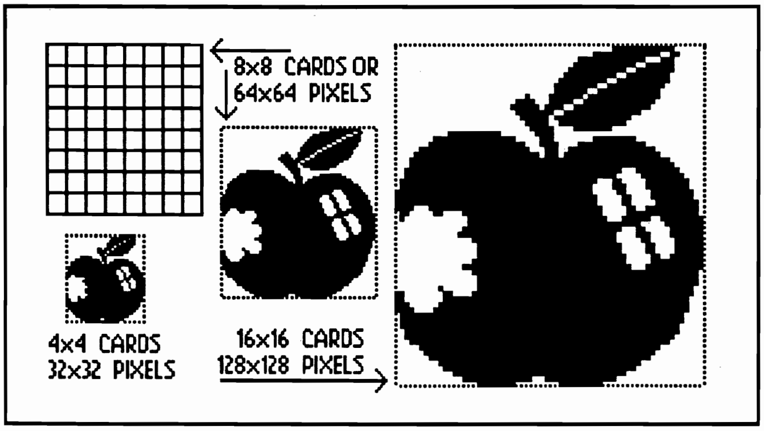


figure 1

Sizing Bitmap Graphics

Learn proportional scaling to enlarge all those little art clips.

By Susan Puhn-Lamb

GeoPaint can be used like most popular graphics software to draw or "paint" pictures presented in screen displays or as printed illustrations. Although drawing window size requires working on a section at a time, an 8 1/2" by 11" page can be produced.

Where the program really outperforms the rest is in creating "clip art" for desktop publishing. This term refers to small black and white pieces of art used to illustrate or break up text. Here, the drawing window size is adequate for most two, three or four column documents. Clip art collections from other text and graphics programs can be converted to GEOS format, but in most cases the art is smaller than desired for professional looking projects.

Before desktop publishing by computer, artwork and photographs were scaled to fit a layout by photographic means. A tool called a *proportional scale* was used to give the cameraman the desired reproduction size as a percentage of the original. Computer bitmap graphics composed of pixels require a whole new approach to adjusting sizes.

Sizing bitmap graphics can be a maddening process as detail is lost when reducing and pixels become pronounced during enlargement. Worse, unless the enlargement is proportional, lines and elements lose their relationship to one another. Smoothing sometimes helps, but it can also wreack havoc with light areas, diagonal lines and patterns.

The only way to enlarge or reduce to exact pixel by pixel increments is to double or halve the image. In other words, a pixel in artwork drawn inside a 1" x 2" box would be doubled when that box is enlarged to 2" x 4". This technique works well in geoPaint which has accurate measuring tools, but is limited to drawing window-size graphics. I have found that using pixel measurement is much more accurate than inches and should be figured in *card* (8 pixel) increments.

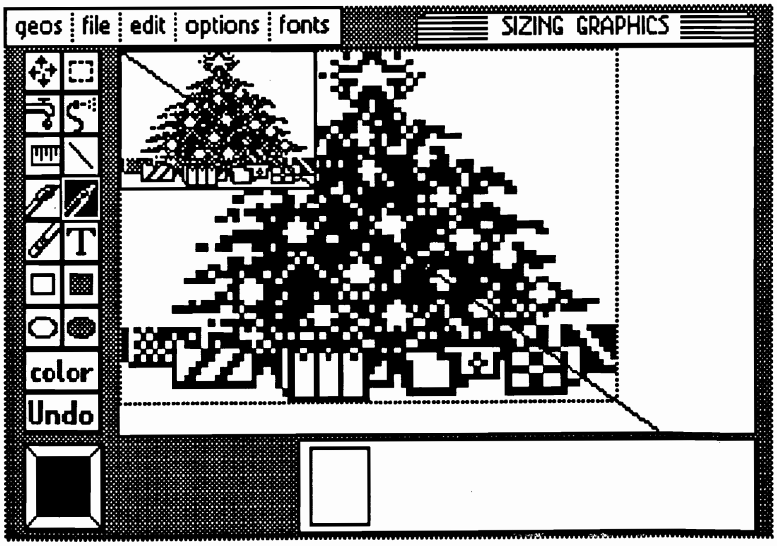
An example of this technique is shown in figure 1, above. A box is drawn around a small piece of clip art measuring 64 x 64 pixels, or 8 x 8 cards. Every card line can be shown by setting down a section of the large grid pattern. (The *grid* that is displayed from the options menu also follows card lines, but is too large for double enlargement). Position the edit box exactly on the card lines around the graphic. A photo scrap is made with *cut* or *copy*.

A double-size box of 128 x 128 pixels is drawn and an edit box is again positioned on card lines for pasting. For attributes, select "Stretch and scale to fit", and probably, "smoothing". The example here is not smoothed to show that every pixel has doubled. A similar piece of art on page 20 has been enlarged, smoothed and touched-up.

The same procedure is used to reduce the graphic by half which does not always turn out satisfactorily because of detail loss. Since you can't actually cut the pixels in half, the computer merely decreases the number of pixels by half and must decide whether the remaining ones will be off or on. The results are not always consistent. If several lines are 5 pixels wide, they may come out 3 pixels wide in one area and 2 pixels wide in another.

The computer randomly makes these pixel-on or pixel-off decisions without regard to aesthetics which is why the human eye is required to decide how reductions of bitmaps should be touched-up to make them look better.

figure 2

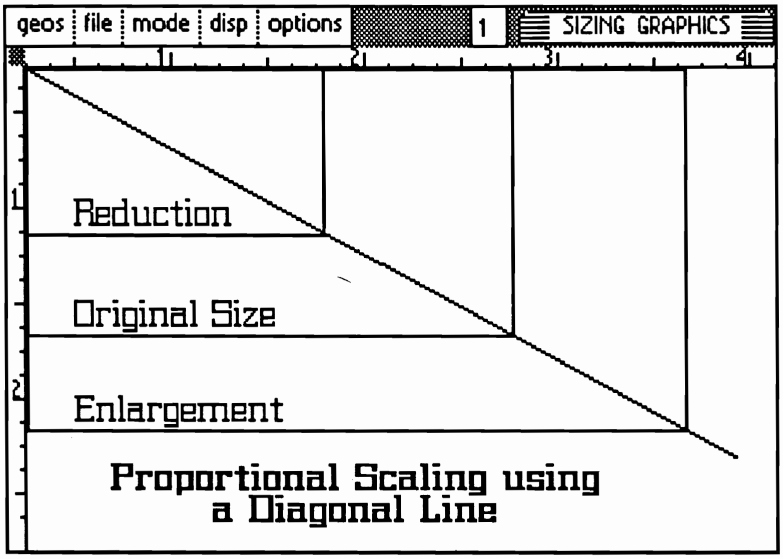


The upper left corners of both boxes start at the same point. The lower right corners lay along the same diagonal line. The artwork here is unsmoothed and would probably be improved with smoothing.

Another way to size (or scale) graphics is to use the traditional diagonal line method. If you know either the length or width of the space that the graphic must fit, a diagonal line, corner to corner through a rectangle containing the artwork will give you the other measurement.

You wont be doubling or halving the pixels, so results will be more random and inconsistent. Therefore smoothing is usually necessary for enlargements, along with touching-up to improve the finished drawing.

There really is no set rule as results are always different depending on variables such as finished size and complexity of the original. A simple bold graphic with rounded lines can be smoothed and need nothing more. A graphic with pattterns could be helped by painting new patterns over those areas. Diagonal lines in the artwork may need to be re-drawn as smoothing makes them wavy.



All rectangles with a common upper left corner and with lower right corners along a diagonal line will be in proportion to one another.

figure 3

In figure two, a box is again drawn around the artwork. With this method, it's not necessary to draw on card lines, but the graphic should be positioned as close to the top and left as possible. Make a photo scrap of the graphic using *copy*. Draw a diagonal line from the upper left to lower right corner of the rectangle. It's all right to draw the line over the drawing since you have a copy on the scrap.

Draw a new box (choosing either the length or width required in your document), with the top left corner in the same upper left point and withthe lower right corner located along the diagonal line. When you paste using "Stretch and scale to fit", your drawing will be the same proportion, larger or smaller, than the original.

For enlargements bigger than the geoPaint drawing window, geoPublish can also be used for this technique. After the bitmap is imported to *page graphics* mode, draw the diagonal line, but not the second box. Click the re-size box on the lower right corner of the bitmap, and drag out the box along the diagonal line. The line can then be selected and cut.

Sizing or scaling works best with simple, bold artwork. Thin black lines drop out and small white areas fill in when reduced. Areas with patterns are unpredictable, often producing new, unattractive patterns or moires. Smoothing lines that are other than straight vertical or horizontal tends to make them lumpy.

It may seem easier to simply draw an edit box the size you need and paste with "Scale to fit" selected. However, the results may be unpredictable and the proportion is often off a little.

For budget-minded desktop publishers using dot matrix printers, who are interested in more professional looking documents (without resorting to cut-and-paste methods), learning to size graphics can make a big difference in how your documents look.

--Susan Lamb
YumaLamb on Q-Link

--continued from page 7

Album Reverter by Joe Buckley changes 2.1 photo albums back to 1.x status. Having this ability may not seem necessary until you try using a program that hasn't been upgraded for use with the newer albums. Albums can always be upgraded to 2.1 format again with the Photo Manager.

There are other programs that create and use photo scraps as part of their operation. *Icon Edit* by Terry Mullett scrolls around a photo scrap to grab whatever is drawn, turning it into an icon.

Icon Grabber by John Paul Young cuts tiny little icon-size photo scraps to paste in geoPaint. They can then be changed or touched up in pixel edit mode, recut and transferred back with the icon grabber as a new icon.

Jim Collete's *Font Editors* (2.4 and 2.5) create small photo scraps of individual characters. They can be pasted in other fonts, or in other point sizes of the same font, as well as geoPaint.

There are more programs that create and use Photo Scraps, even major applications such as geoChart and geoFile. I have mainly discussed those used in desktop publishing and by third party GEOS programmers.

We will continue to make these programs and their upgrades available as we can. Addresses are on the programs for suggestions or shareware donations.

The programs available on GEOWORLD disks are as follows:

Disk #1 -- Graphic Storm, Photo Print, Icon Edit, Album Reverter and ScraPeek 2.2.

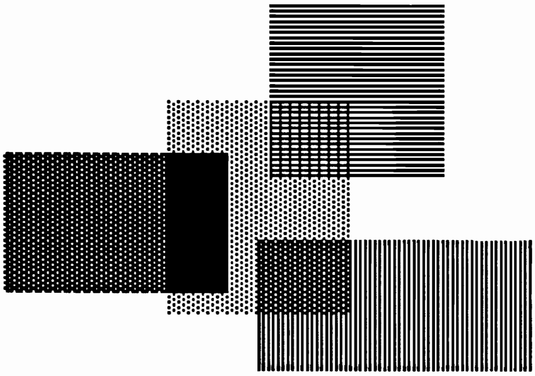
Disk #2 -- ScraPeek 3 and an album of icons for Icon Edit.

Disk #3 -- Retriever, GetGraphic, BigClipper, Paint-Scrap, and ScraPeek 3.4.

Disk #4 --Album Animator and three animated albums.

Disk #5 -- Photo Mover, geoAlbum & the clip art in this article.

We've come a long way from those days when the main purpose of a photo scrap was to stick a picture in the middle of a geoWrite page.



Color Separation with geoPaint

Experimenting with patterns in the four-color printing process

By Susan Puhn-Lamb

Last issue's cover was produced by flat color printing, a procedure where the printer uses a specific pre-mixed color or mixes the ink to match a desired shade.

The headline shadow and issue #21 bar are solid (100%) bright blue. The background of the headline box, monitor screen and ocean are three shades of the same blue, produced by printing in various *screens* to give the illusion of lighter color. The continents are printed solid blue with a black screen, for a very dark blue. There are also areas of gray that are screened. With only two colors, printed on white, quite a variety of colors have been achieved on this cover.

If a page is designed with a large number of different colors, like our font issue (#20), it would be far too expensive and time consuming to use a different ink for each color. Full color, as seen in newspapers and magazines is a level of printing called four-color process printing.

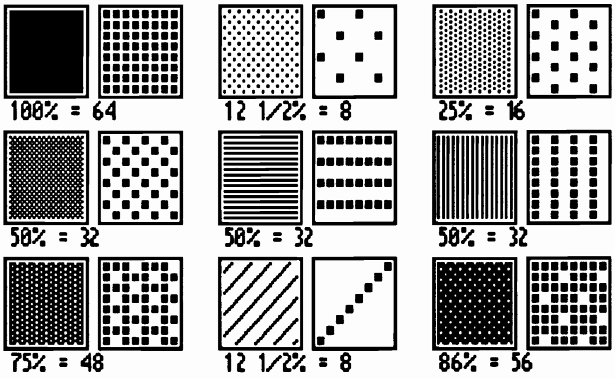
A color photograph or full-color artwork is separated into four basic colors with a photo-mechanical process. Special filters are used to make four different halftone (screened) negatives of each color. The colors used are black, cyan (process blue), magenta (process red) and process yellow. They are over-printed in a combination of solids and percentage screens of these colors to give the *illusion* of full color.

You can observe this technique by looking at the sunday color comics through a powerful magnifying glass or loupe. Notice that every varied color is made up of different-sized dots of only the four process colors.

With these traditional printing methods, a graphic artist can use a color chart and specify percentages of each process color to make an object a certain color. For example, on the Font issue, the green in "Chop Suey" was made by printing a 75% process blue over solid yellow. "Yellowstone" has a 25% process red over solid yellow and so on.

Now, all of this is pretty complicated and to my thinking, may defeat the primary purpose of desktop publishing. I want to devise a technique of color separation that can be done with GEOS so I can present the print shop with four registered printouts, to make the four color plates from. Although we use a laser printer to produce GEOWORLD, perfectly good printouts can be gotten from a dot matrix printer. and a multi-pass printer driver.

figure one



One other consideration of the traditional and complicated method, is what happens when you print two or more screens over one another. An undesirable pattern, called a moire occurs and the screens have to be made at different angles so the dots of each color are in a different position.

To avoid the worry of moires, I decided to try something different that would still use the four-color process. Unfortunately, I wont know what it will look like until after the magazine is printed.

What I'm hoping to produce on the cover of this issue is boxes of many different colors using the process inks. Each different color will be the result of combining dots, lines and patterns of the process colors, printed next to and over each other.

To accomplish "screened" areas, I am using patterns in geoPaint along with those I created with a pattern editor. In order to use a color chart, I need to use patterns that approximate screen percentages. First, I looked at the 8 x 8 pattern modules in an editor to figure a percentage of each pattern. Some of these percentages aren't exact, but they're as close as we can get in matching a color chart.

Since this is the first time I've done a separation this way, I'm using a chart that is designed for the traditional method and color is achieved with 10, 25, 50 and 75 percent screens. Once this cover is printed, it can be a chart for another job using the same equipment and materials. If you want to get into doing color this way, the obvious solution for accurately predicting finished colors is to make your own color charts.

Figure one shows some of the patterns and their 8 x 8 modules. Solid black (or 100% color) is a 64 x 64 pixel module. The 50% screen has half the pixels turned off for a total of 32. Any pattern canbe analyzed this way. By using these patterns instead of regular halftone screens. some interesting effects can be achieved.

To make the cover of this issue, I started with a geoPaint page of the headline and a series of 25 boxes. After making three more copies, I used the fill tool and simply poured patterns into the boxes according to which color plate I'm working on. For example, the top left box was made with 75% blue, 50% red and 12 1/2% yellow. The second box is 100% yellow and 75% red. The patterns of these two boxes are shown at the upper right of this page.

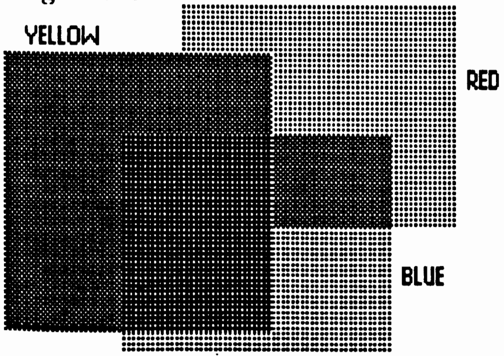
I added an extra step when using 50% screen patterns such as the dots (to the right of solid black), horizontal lines and vertical lines. If the same pattern is used on two different plates, and with good registration, the colors will be printed on top of each other with the white paper showing through. The white will dilute the color and make it lighter.

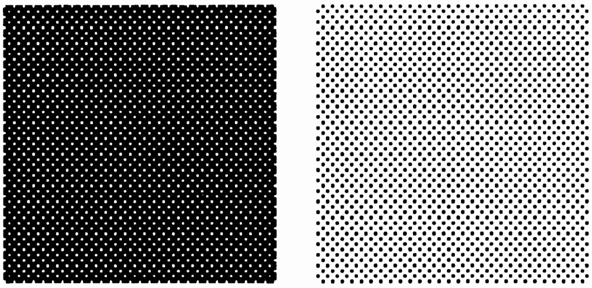
To avoid this situation, use the regular pattern on one color and reverse it for the other. Just position an edit box exactly on the box and reverse. If your object is other than rectangular, you can make a reverse of the pattern with a pattern editor. This reversing trick is also handy when you combine a light and dark color. Use a dark pattern for one color and the light reverse will print between the dots or lines on the other.

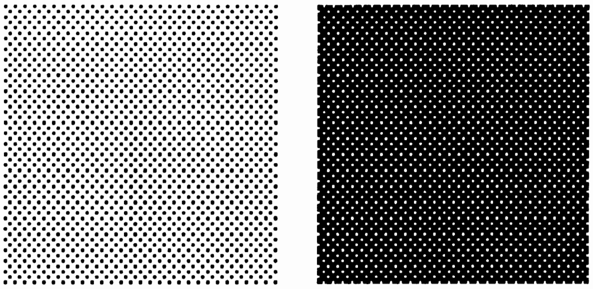
When patterns are printed over one another, a different pattern can be created. This is especially true of red, blue and black. The yellow is too light to form a noticable pattern. Figure two shows how three different patterns combine when printed.

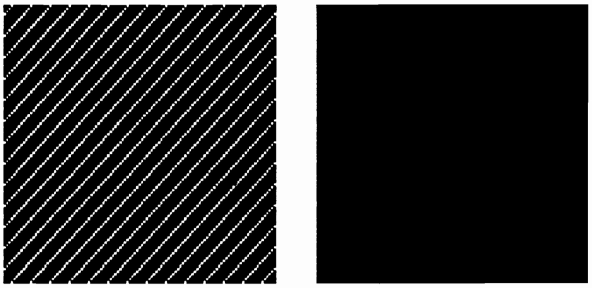
In order to check on how all the patterns on a page will print, a combination page can be made with the *Paint OVERLAY* paint driver. Just name a copy of one of the color pages "OVERLAY" and print any or all of the other pages with Paint OVERLAY selected as the printer driver.

figure two




blue plate


red plate


yellow plate

The resulting geoPaint page will have all the pages combined, transparently. You can view the page or print it to see how the patterns will be printed by the print shop. If a section is solid black, the patterns will completely cover the paper. Where there is still a pattern, it will show what parts will remain white.

For this experiment in color, I didn't use any screens on the black plate. Light black screens can be used over any color to tone it down. Darker black screens (such as 50%) will darken or deepen the color.

Learning the way the colors interact and working with black screens lets you create shading for more dimension. I will be experimenting much more with this technique.

It is a good idea to put crosshairs for registration on the plate before making duplicates. The printer can remove them, but make sure you let him know they aren't part of the design.

Believe it or not, it is also possible to use this four color process with a dot matrix printer. I have used it with color ribbons to produce very professional-looking greeting cards. The color ribbons I use with my Star SG10 are very close to process colors, so the color chart can be used.

Obviously, the most important consideration is registration and can be very frustrating. You'll have to be able to figure out how to roll your paper back for each color. Using tractor feed paper, I usually set up to print ten copies. I tear off a length of 13 sheets of tractor feed paper.

The first sheet is to feed in the stack and the second for registering the colors. The 13th sheet is an extra to keep the feed straight.

Once I've printed the first color on the registration page and following ten sheets, I roll them back or feed the stack back from the top. I always position the paper in the same place, lining up the tear line with the rollers that hold down the paper. I watch how the colors line up on that first sheet, and if they're close enough, let it go on printing. If it's way off, I can stop on this sheet and start over, rather than ruin the other ten.

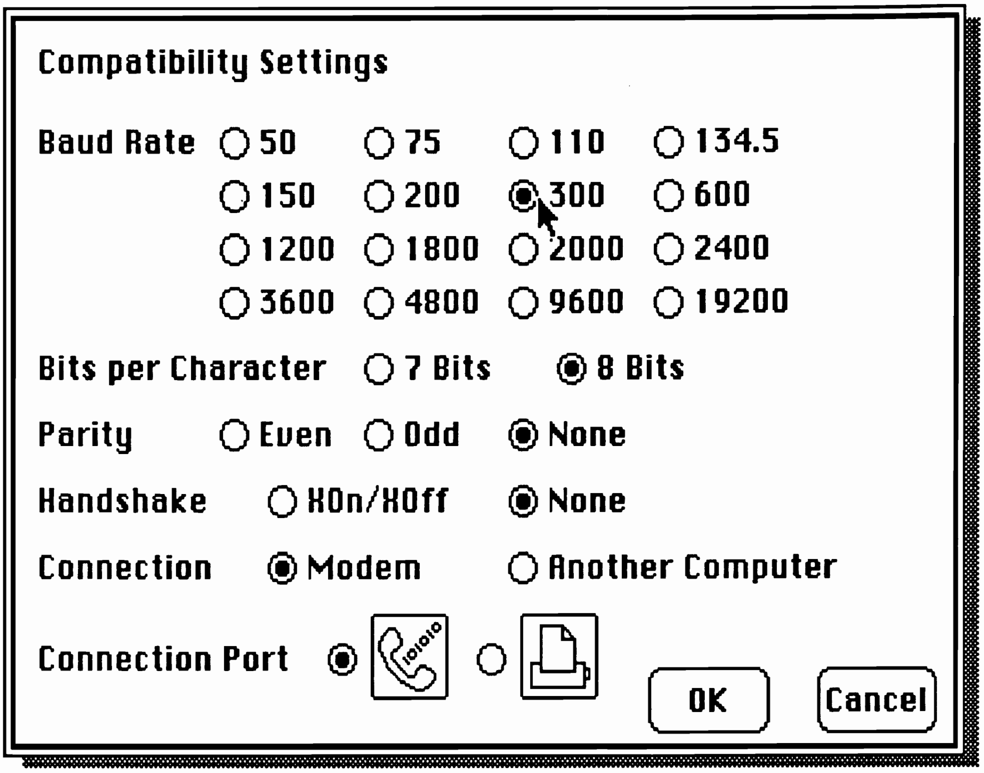
To keep from contaminating the color on the ribbons, always start with yellow, then red, blue and end with black. For better cleaning, just print something without a ribbon until nothing shows on the paper.

I have also printed multiple color greeting cards on heavier paper, feeding single sheets through. Again, if you can put the paper in the same position for each color, fairly good registration can be achieved. This method is not for something that needs very close registration or for a large number of pages. It is meant to be for limited edition graphics--more work, but more special.

The techniques I write about are made up as I go along through the GEOS environment and mostly experimental. To analyze this particular procedure, look at the four plates on GEOWORLD Disk #5. --Susan Lamb

The MAC Connection

Trading information between the Commodore 64 and Apple Macintosh
By Saul Cohen



Compatibility Settings

Baud Rate ○ 50 ○ 75 ○ 110 ○ 134.5 ○ 150 ○ 200 ● 300 ○ 600 ○ 1200 ○ 1800 ○ 2000 ○ 2400 ○ 3600 ○ 4800 ○ 9600 ○ 19200

Bits per Character ○ 7 Bits ● 8 Bits

Parity ○ Even ○ Odd ● None

Handshake ○ XOn/XOff ● None

Connection ● Modem ○ Another Computer

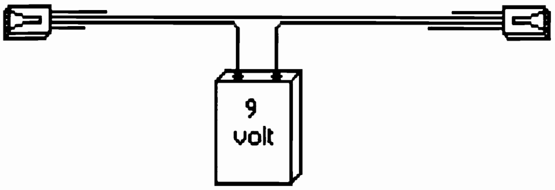
Connection Port ● ☎ ○

MacTerm Dialog Box saved on MAC and edited with SuperPaint. Transferred to Commodore using an RS232 interface. Converted to geoPaint using MacAttack; photo scrap made with BigClipper

There are a variety of ways of trading information between computers, including connecting through established telephone lines and uploading and downloading through a bulletin board. This article is about two less traditional ways. Both methods involve having the two computers side by side and interfacing them through a cable. In this case, the two computers are the Commodore 64 and the Macintosh, but other systems should work equally as well. I've selected the Commodore and MAC because of my familiarity with both machines. Most of my original work is done on my Commodore but needs to be processed on the MAC. Also, there is a vast library of MacPaint files which could be used with geoPublish.

It seemed like a good investment of time to learn how to transfer files in both directions. The first system of transfer involves the use of a modem on each computer with a direct connect phone wire. The second makes use of an RS232 interface to a null modem cable and a direct connection.

Here is a brief description of the modem system. I used a *1670*, 1200 baud modem with *Common Sense* software on the Commodore and an Apple 300 baud modem and *MacTerm* software on the MAC. I connected a two wire cable with phone clips between the two modems. I placed a 9 volt battery in series on either one of the wires (see drawing).



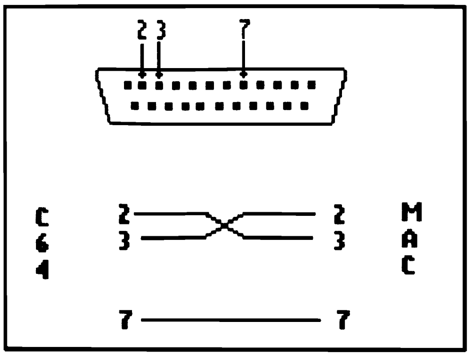
Connect two modems with phone clips

The purpose of the battery is to fool the modems into thinking that they are attached to a signal line. Set the Mac terminal to TTY; set Compatiblity to 300 baud, 8 bits, no parity or handshake, modem and phone port; File transfer X-Modem and delay 1. The phone option also had to be set to 300 baud. Enter any phone number under the phone option.

I selected similar settings for baud, bits, parity, handshaking and 3 for the delay on the Commodore. Next I selected Phone and Dial on the MAC. When the dialing was complete, I typed +++, return and ATZ (using caps) on the Commodore. The connection was made immediately. For file transfer, set the MAC to send and the Commodore to receive. You can also send with the Commodore and receive with the Mac by reversing the process. After exiting the link, use the appropriate software to pull in the data files you have sent. I used MacAttack and Font Monster by Joe Buckley, on the Commodore and PageMaker and Word 3.01 on the Macintosh.

The second system of data transfer makes use of an RS232 connection between the two computers. There are three components that you will need. For the Commodore you need to purchase an RS232 interface. The Mac will need an SCSI (Small Computer System Interface) cable. It has an 8 pin MAC plug at one end and a DB25 connector at the other. The third part is a three wire cable which connects the RS232 interface with the DB25 end of the SCSI cable while switching pins 2 and 3.

To make this null modem cable you need three wires and two more DB25 connectors. You can buy the connectors at any Radio Shack store. If your RS232 interface ends with a female plug, buy a male for that end. Since the MAC cable ends with a male end, buy a female DB25 for that side. Solder a wire to pin 2 of one DB25 and the other end to pin 3 of the other DB25. Repeat with another wire from pin 3 on one end to pin 2 on the other.



NULL MODEM INTERFACE CABLE

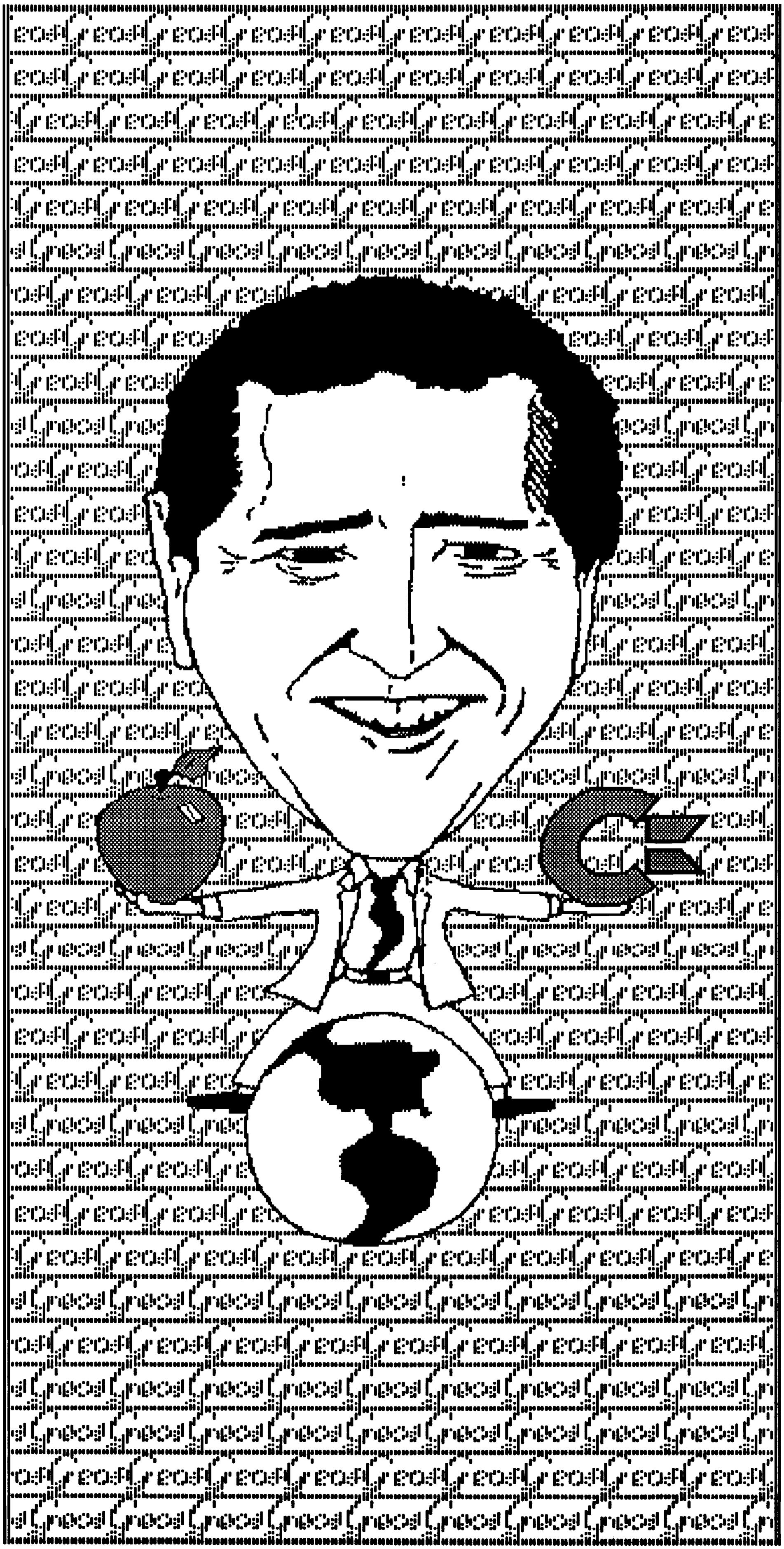
Connect a wire to pin 7 on both DB25 connectors. (Someone at a user group or physics class at a local college, will probably be glad to make you a cable).

Once the cable is built, insert the RS232 interface into the Commodore user port (back left) before turning on the computer. Plug the MAC cable into the phone port, and connect the three wire cable between the Commodore RS232 and the MAC cable. Turn on the computers and load up the terminal software on both. There are only a few setting changes which need to be made. Set both computers to 1200 baud.

Change the MAC Compatibility setting to other computer instead of modem. All the rest of the settings remain the same as with the two modem system. No need to dial - when the settings are correct, typing on one screen will appear on the other and visa versa. File transfers work exactly the same, except at 1200 baud you can transfer data faster. Using the same three wire cable, you can easily transfer files to other computers. I have successfully tested the Radio Shack Portable 100 with the Commodore and the Macintosh.

In summary, there are several simple ways to transfer data between computers of different brands. Two methods were described: one which uses two modems and a simple phone cable, and the other an RS232 interface, null modem and MAC cable. With these items in hand and a little patience with the software settings, you should be able to get your computers communicating.

--Saul Cohen
Qtutor SEC on Q-Link

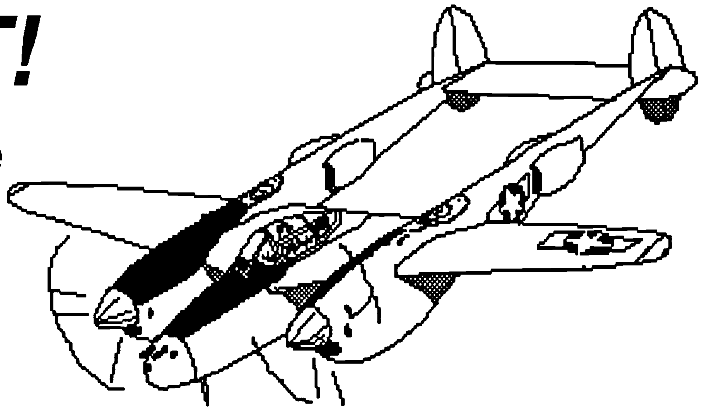


GeoPaint drawing of Brian Dougherty by Dale Beach (CartoonKid on Q-Link).

Make Your Apple GEOS Documents Fly With DISKART!

Apple GEOS users can now have great graphics at a low cost!

Requirements: GEOS by Berkeley Softworks
Apple IIC/Plus, IIE w/128K & 80 col card
Apple IIGS or Laser 128/128EX
Joystick or mouse



DISKART 1A
Vehicles 1
Vehicles 2
Porsche 959
Tin Lizzies
Warbirds 1
DC-3 Airliner
Nieuport 17

DISKART 2A
Graphic Goodies 1
Graphic Goodies 2
Graphic Goodies 3
Graphic Goodies 4
Weather Goodies
U.S.Maps

DISKART 3A
GEOS tips 1
GEOS tips 2
Little Guys 1
Little Guys 2

DISKART 4A
Food Stuff
Tools 1
Vehicles 3
Drafting Equipment
Make-A-Face
Gardening Stuff

DISKART 5A
Little Women 1
Little Women 2
Banners 1
Houses 1

DISKART 6A
Farm Animals
Dogs 1
Cats 1
Fish 1
Zoo Animals
Space Creatures

DISKART 7A
Critters 1
Tropics
Holidays 2
Baby Items
Big Boats
Small Boats

MUSI-KIT
MUSI-KIT
Musi-Kit Info
Single Title
Single Staff
Piano Title
Piano Staff
Music Sample

Order TODAY! **Only $15.95 per double-sided disk**

Please circle your choice(s) below

DISKART 1A DISKART 2A DISKART 3A DISKART 4A
DISKART 5A DISKART 6A DISKART 7A MUSI-KIT A

Name________________________________

Address______________________________

City________________ State/Prov ____________

ZIP/Postal Code_________ Country______________

Date of Order _________ Telephone ______________

California Res. Add 6.5% Sales Tax - Check or Money Order ONLY (U.S.Funds)
For Orders Outside U.S. Add $2 per disk for shipping - Defective merchandise will be replaced - no returns

Total No. of disks___
Sales tax ___
Shipping ___

Total Due _____

USE YOUR DISCOVER CARD!

Card No.

Name (as on Card)

Exp. date

Those Designers

Makers of DISKART, The Original GEOS-ready graphics.
3330 Lewis Avenue, Signal Hill, California 90807 U.S.A.

the GEOS Report

THE WORK OF THE KERNEL (Kernel Knowledge)

Part 5 of a special comprehensive report on the GEOS Environment
By Mike Ross

It is the kernel that puts it all together. It has all the routines to build and handle windows, menus, icons, and fonts, not to mention the Disk Turbo which is 50 percent of the kernel. I will first discuss the design considerations that structure the kernel in which it is fashioned.

The 6502 is often compared to risc-based processors. One of the outstanding features of this chip is the 3-cycle execution on most instructions. It appears as if there are only 3 registers: the Accumulator, and the X and Y registers. However, there is also zero page (locations 0000h to 00FFh). Zero page accesses are phenomenally fast as they do not require a word-length address. In fact, Berkeley uses zero page as if they had 256 other registers-- as an area to pass parameters, perform internal calculations, and return values. Labeling zero page addresses as if they were 16 bit registers makes it easy to keep track of these"pseudoregisters". Registers r0 to r15 occupy addresses 0002h to 0020h. These are word registers but the high byte and the low bytes are used independently. There are 10 more registers that are used only for applications. The kernel does not pass values to them and they are off limits to desk accessories. a0 through a10 occupy addresses in 2 separate parts of zero page -- 00FBh to 00FDh for a0 and a1; 0070h to 007Fh for the other eight registers.

Sometimes A, X, Y and even the Carry flag are used for speed. Another fast method used is the in-line call. This is simply calling a subroutine, then getting the necessary parameters from the bytes following the subroutine.

An example using GEOS's standard macro assembly code would look like this for a standard call (not inline) to build a rectangle:

```
    LoadB    r2L, 0         ;top of rectangle.
    LoadB    r2H, 199       ;bottom of rectangle
    LoadW    r3, 0          ;left side
    LoadW    r4, 319        :right side
    jsr      Rectangle      ;draw it
```

Compare this to an in-line routine:

```
    jsr      i_Rectangle    ;draw a rectangle in the
                             current systempattern
    .byte    0              ;top of rectangle
    .byte    199            ;bottom of rectangle
    .word    0              ;left side
    .word    319            ;right side
```

When the in-line routine is called, it immediately pops a word off the stack. Instead of the return address, this word points to the parameters passed in-line after thejsr. The in-line routine picks up its parameters, loads the pseudoregisters, then puts the return address back on the stack and executes.

This scheme makes sense when a routine is called a number of times with fixed values. **LoadW r3, 0** takes 8 bytes; **.word 0** takes only 2.

To what degree are the pseudoregisters consistent in regards to the values that they hold? Mr. Loveless told me that they mix and match registers to optimize routines, but the consistency of which values are passed where often depends on who wrote a particular routine. Routines such as Rectangle (C124h) will use the same registers as FrameRectangle (C127), while HorizontalLine (C118h) does not clobber those registers at all.

Optimization, for Berkeley Softworks, means eliminating redundancy in the registers, but not to the extreme that the registers are hardcoded. Another optimization routine is cacheing overlay modules. Utilizing a cache of 6K, only 2 or 3 modules can be in main memory at a time. Deciding what module to swap out of the cache is based on frequency of use.

It becomes apparent that the kernel has a lot to do and not a lot of memory to do it in. Using coding techniques from the video game days, GEOS is the tightest code around. There is no wasted space anywhere.

There are tradeoffs -- subroutines mean loops and that means longer execution; unpacking the loops would only increase the memory requirements. What is most fascinating about GEOS is how so little memory can do so much. There are places where speed has to be sacrificed in favor of byte conservation. But, on the other hand, BSW has pulled out a number of coding tricks to gain speed in the system calls. Curiously, error checking (other than disk error and handshaking) is left up to the application.

GEOS Kernel Structure -- there are two levels of code running within the GEOS kernel: **MainLoop** and **InterruptLevel**.

MainLoop is code just waiting for something to happen. When something does happen (user input), it dispatches control to the proper application service routine. The sort of things that MainLoop is looking for are: mouse button clicks on an icon, menu, sub-menu or an activation of **otherPressVector**, which senses a rapid double-clicking. **otherPressVector** is activated when a single click is detected. A constant value **Click_Count** is loaded into **dblClickCount** (8515h), a counter that is decremented at every interrupt. If a second click is encountered before dblClickCount is 0, then MainLoop knows this is a double-click action and dispatches the routine.

In addition, MainLoop looks for keyboard input for an application (through **keyVector**), or for text input into a dialog box. MainLoop may also encounter a process time-out indicating an application service routine should run. The types of action MainLoop does, include pulling down sub-menus or acting upon the selection of a menu item.

InterruptLevel code manages the 6510 IRQ interrupt which is triggering 60 times a second via raster interrupt on the 64. MainLoop halts and interrupt code is run in less than a sixtieth of a second. InterruptLevel saves the machine's state and interacts with the hardware, buffering input, decrementing process timers, sprite moves and mouse clicks.

InterruptLevel sends flags out to MainLoop. MainLoop determines which actions to execute, based on the flags. InterruptLevel looks for action and MainLoop exploits the action. Programmers designing applications for GEOS realize that they need only to define which events they want to occur and let the kernel do the work for them

In addition, the kernel has an extensive library of text, menu and graphic handling routines. To make handling these routines easier, the 8000 byte hi-res screen has an additional 8000 byte buffer holding a copy of the data. When a menu is pulled down or a dialog box is invoked, the information "covered up" is nicely stashed in the buffer where it can be resurrected immediately. This technique is used for the **UNDO** command in geoPaint.

Turbo Disk -- The GEOS kernel loads the disk turbo code into the drive's RAM where it stays resident, ready to receive instructions from the kernel. This is much more intensive than standard 64 DOS. Simple functions like **delete** and **validate** are now controlled through the computer. The standard DOS was slow as information is sent to the drive as single bytes, requiring excessive handshaking. The kernel's turbo routine moves blocks at a time.

Here are the higher level disk turbo routines:

NewDisk (C1E1h) -- Sets up a new disk, reading the BAM into the drive's memory.

SetDevice (C2B0h) -- Opens a disk. On two drive systems, the new drive's turbo is activated and the old drive's turbo is suspended.

OpenDisk (C2A1h) -- Initializes the current disk in a drive.

GetFiles (C208h) -- Very high level routine to load anything that is executable. A pointer to the file name is in r6.

SaveFile (C1EDh) -- Very high save routine. r9 needs to hold a pointer to the header block and r10L needs to hold the number of the directory page to start looking for a hole to place the file's entry in.

Here are kernel locations specific to disk routines:

diskBlkBuf (8000h) -- 256 byte general purpose disk buffer

fileHeader (8100h) -- 256 bytes. Holds header block

curDirHead (8200h) -- 256 bytes. Holds the Directory header and BAM

fileTrScTab (8300h) -- 256 bytes. Holds the VLIR index table.

dirEntBuf (8400) -- 30 bytes. Holds the directory entry.

drACurDkNum (841Eh) -- Name of disk in Drive A

drBCurDkNum (8430h) -- Name of disk in Drive B

curDrive (8489h) -- Currently active drive #

diskOpenFlag (848Ah) -- FFh if drive is open

isGeos (848Bh) -- FFh if disk is GEOS type

interleave (848Ch) -- Sector interleave

numDrives (848D) -- Number of drives available

driveTypes (848Eh) -- 4 bytes -- Type of drive. Identifies RAM drive or shadowed drive configurations

turboDrive (8492h) -- 4 bytes -- Bit 7 indicates Turbo resident; bit 6 indicates Turbo is running

driveData (88BFh) -- 4 bytes -- One byte reserved for each drive's driver.

curType (88C6h) -- 4 bytes -- A copy of driveTypes

drCCurDkNm (88DCh) -- 18 bytes -- Name of Disk in Drive C (Not used at present)

drDCurDkNm (88EEh) -- 18 bytes -- Name of Disk in Drive D (Not used at present)

dir2Head (8900h) -- 256 bytes -- holds the second half of a header block for larger capacity drives.

Here are some intermediate file handling routines:

FindFtypes (C23Bh) -- used by Dialog box function **DBGetFiles**. This routine returns a list of files of a particular type (like a listing of Photo Albums in the Photo Manager)

FindFile (C20Bh) -- Loads a directory entry directly into the computer's memory

GetHdrInfo (C229h) -- retrieve's a file header block

LdApplic (C21Dh) -- Load and executes an application. Similar to GetFile but uses a pointer rather than a filename

LdFile (C211h) -- The actual loading for LdApplic occurs here

LdDeskAcc (C217h) -- GetFile calls this routine to load a desk acessory. The area of memory to be occupied by the Desk Accessory (determined by start and end addresses) is saved to disk temporarily while the desk accessory is active.

ReadFile (C1FFh) -- low level file loading routine.

ReadByte (C2B6) -- simulates reading a byte at a time, but actually reads in a block at a time. ReadByte is used for decrypting files. Routine **BitOtherClip** will use ReadByte to read in a graphic file too big to fit into memory.

That covers the Turbo Disk routines for loading of files. For the saving of files there are an equal number of routines. Rather than being redundant, I will discuss the operation of sector interleaving briefly. The **Interleave** at 848Ch is used for searching the disk for the most optimal location of storing data. The number stored here, usually 8, is how far apart to place the sectors. If at sector 1, the interleave directs that the next block go to sector 9. Setting this value lower can cause sectors to be too close together. Turbo must finish processing a block before getting the next one or risk causing the disk to spin an additional revolution.

One other file handling property should be noted. The X register is used consistently to hold any disk error byte. This error byte will remain unaffected as GEOS steps through a heirarchy of disk routines so that the user will be sure to get the error message.

This concludes the discussion of disk and file handlings. I now want to take a look at how GEOS builds a menu through the routine **DoMenu** at C151h.

DoMenu draws and sets all pull down menus and submenus. This powerful routine does everything for menu processing. Once DoMenu is initialized by the call, menu processing is handled by MainLoop. The accumulator will hold the number of the menu selection to place the pointer on. r0 will hold the address to the menu table. In the process of handling doMenu, r0 to r13, A, X, Y will all have their values destroyed. At the area of memory pointed to by r0, the following bytes are used:

# of bytes	Description
1	Top margin of entire menu
1	Bottom margin of entire menu
2	Left Margin of entire menu
2	Right Margin of entire menu
1	Code byte: bit 7 -- vertical menu bit 6 -- set secondary box descriptor to full screen (allows the mouse to be moved outside of a menu without causing it to be closed) bits 0 - 4 -- Number of entries in menu.

This is all followed by sets of 5 bytes, as many as there are entries

2	Address for the text for this option
1	Code byte which describes what to do with the address that follows: bit 7 -- operand is the address of a submenu descriptor bit 6 -- call subroutine. Must return a result in r0 which is either 0 or the address of the next submenu If neither bit is selected, it will flash before the routine is executed and control will not return to DoMenu
2	Address of either a submenu descriptor or a routine to be executed

Here is how this code would be handled if part of a menu-building routine in a typical application:

```
 LoadW      r0, #MenuTable   ; LoadW is a macro to load the address of MenuTable into r0, one of
                             ; GEOS' pseudoregisters in zero-page. MenuTable is defined below
 lda        #0               ; places pointer on first menu item when done
 jsr        DoMenu           ; have GEOS draw the menus on the screen
```

When DoMenu is called through a **jsr** (jump to subroutine), it looks for the address in memory pointed to by r0. In this case, r0 points to another section of code containing the menu descriptors, MenuTable. That section of code will typically look like this:

```
MenuTable:                      ; menu definition table for main horizontal menu
    .byte      0, 14            ; top and bottom y-coordinates
    .word      0, 49            ; left and right x-coordinates
    .byte      2 | HORIZONTAL;  "|" = logical or ;  2 = # of menu items;  HORIZONTAL = menu type

    .word      GeosText         ; pointer to text for left horizontal menu item
    .byte      SUB_MENU         ; type of menu underneath GeosText
    .word      GeosSubMenu      ; pointer to submenu structure underneath GeosText

    .word      FileText         ; pointer to text for right horizontal menu item
    .byte      SUB_MENU         ; type of menu underneath FileText
    .word      FileSubMenu      ; pointer to submenu structure underneath FileText
```

Methodically, DoMenu has looked at MenuTable getting coordinates, the number of menu items in the main menu structure, and defining the menu as a horizontal type. Afterwards it gets the text to be put into the menu followed by the type of action the menu is to carry out when clicked on. In this case both menu items (GeosText and FileText) will create submenus.

Underneath FileText is FileSubMenu. Here is the code for this sub-menu:

```
FileSubMenu:                           ; menu definition table for FILE vertical menu
    .byte       15, 44                 ; top and bottom y-coordinates
    .word       29, 64                 ; left and right x-coordinates
    .byte       2 | VERTICAL           ; number of menu items, type of menu

    .word       CloseText              ; pointer to the text for menu item
    .byte       MENU_ACTION            ; type of action
    .word       DoClose                ; pointer to handler routine

    .word       QuitText               ; pointer to text for menu item
    .byte       MENU_ACTION            ; type of action
    .word       DoQuit                 ; pointer to handler routine
```

MENU_ACTION, like SUB_MENU, defines the kind of action the menu item is to perform. In this case, rather than pointing to further submenus, the submenu items (CloseText and QuitText) are to execute the routines specified by DoClose or DoQuit.

Here are DoClose, DoQuit, and the various xxxxText code:

```
DoClose:
    jsr         GotoFirstMenu          ; kernal routine at C1BDh that rolls up the submenu
    ; whatever code is needed to close this file without going to DESKTOP goes here
    rts                                ; return -- all done

DoQuit:
    jsr         GotoFirstMenu          ; roll menu back up
    jmp         EnterDeskTop           ; kernal routine at C22Ch -- a direct jump to the DESKTOP
;
;
GeosText:
    .byte       "Geos", 0              ; the word "Geos" is placed in the menu location
FileText:
    .byte       "File", 0              ; the word "File" placed in the menu location
CloseText:
    .byte       "Close", 0             ; the word "Close" placed in the submenu location
QuitText:
    .byte       "Quit", 0              ; the word "Quit" placed in the submenu location
```

In addition to MENU_ACTION and SUB_MENU, a more complex menu selection type can be activated. **DYNAMIC_SUBMENU** builds a menu dynamically by checking the state of the system and altering the menu table before the menu is displayed. In simplest terms, this menu type calls a user-defined routine before the menu is unfolded. The word following **DYNAMIC_SUBMENU** is a pointer to the routine to call before the submenu is opened. That routine will in turn load a pointer in r0 to the sub-menu structure that needs to be opened.

In this way, Desk Accessories can be dynamically allocated on a submenu. Likewise a font submenu with its point-size sub-submenu can be created.

The last topic I will cover is **Process Support**. A GEOS process is a time-based subroutine triggered to run over a number of interrupts. InterruptLevel will set the flag for a process and MainLoop will dispatch the process. GEOS can manage many processes and a process can be run, blocked, or frozen. The application supplies a process definition table. The byte preceeding the process name is a word value **n** that acts as a timer that is decremented every sixtieth of a second.

Runnable processes have actively decrementing timers and will dispatch their routines when the timer reaches zero. A frozen process has its timer frozen and thus is prevented from carrying out its routine. The timer cannot reach zero. A blocked process will not dispatch its routine even though its timer continues to decrement to zero. GEOS also supports "sleeping." A sleeping process is prevented from executing for a specified period of time. When this time is up, the process awakens.

It is apparent that GEOS stretches the limits of what was considered possible for an eight bit machine to do. The fascination of GEOS is that it can do so much with so little memory. Memory is considered a cheap commodity today, but, on the old eight bits, memory is a precious resource.

It may well be that managing and budgeting limited memory is a skill that is being overlooked in today's programming environments. I feel that there is great value in learning tight programming. I suspect that Microsoft Windows and Wordperfect, both weighing in at 512K, need not be so fat. The consumer may well be taking a ride to buy more memory at his/her expense simply because of wasteful programming techniques. As the price of RAM has done a turnaround and started to skyrocket in price, budget-minded programming may come into vogue again, but don't hold your breath.

In the meantime, my "toy" computer takes it share of insults while producing output that continues to silence its detractors. Thanks to GEOS.

Bibliography

GEOS User's Guide, Berkeley Softworks. *The manual that comes with the boot disk*

The Official GEOS Programmer's Reference Guide, Berkeley Softworks, Michael Farr, published by Bantam Computer Books, 1987. *This reference manual preceded the release of geoProgrammer and versions 1.3, so some of the material is slightly dated.*

GEOS Programmer's Reference Guide, by Alexander Boyce. *This is a shareware manual that preceeded the release of geoProgrammer. His labels for kernel routines differ from Berkeley's, but it is a good concise manual and a good companion publication.*

GeoProgrammer User's Manual, Berkeley Softworks, written by Matthew Loveless. *This user manual has no how-to's in it, expecting that you know 6502 Assembly Language and macro Assembly. The manual is amazingly comprehensive in covering the development of programs using their development tools.*

GeoWorld magazine, specifically the **Inside GEOS** series by William C. "Master Blaster" Coleman. *Most of the disk turbo material was capsulized there.* **GeoWorld** is put together solely with GEOS.

GEOS News, Berkeley Softworks' quarterly newsletter. *The description of geoCalc and geoFile were taken from here.*

Other material was taken from a variety of GEOS applications user's manuals: **geoWrite, geoPublish**, and **DeskPack.**

Some information was gleaned by prowling the GEOS message boards of **Quantumlink**, where there is a strong GEOS sub-culture

Lastly, I wish to personally thank Mr. Matt Loveless of Berkeley Softworks who spent a good hour and a half with me talking over technical issues and GEOS history. All this on his dime!

--Mike Ross

Letters to geoWorld

GeoPublish document to geoPaint page

In your geoWorld issue #21 you stated that a geoPublish drawing could be transferred to geoPaint using one of the paint drivers. After reading your article, I tried to transfer a drawing and had no luck.

The GEOS 2.0 manual does not mention converting a geoPublish document, it refers to converting a geoWrite document into a geoPaint version. Can you tell me how to do it?

James Vorrasco
Clearwater, Florida

The 2.0 manual covers the operation of the paint drivers on pages 244-251. GeoPublish is mentioned on page 248 as taking more time to convert. This can especially be the case if your document has a lot of graphics and/or text. With the Paint PAGES driver, the document and geoPublish on the disk, use "select printer" from the "geos" menu and choose the paint driver. Then simply print the document. The screen will stay in the printing mode for a long time, with the pointer occasionally flickering. Another consideration is to have plenty of room on the disk. Some of these geoPaint pages end up being over 40K. --Ed.

Desktop publishing on the C-128

On the contents page you list that you use a Commodore 128 and geoPublish to produce this publication. Does geoPublish work with GEOS 128 v2.0 in at least 40-column mode? I am very interested in doing my own desktop publishing and feel that geoPublish is the most powerful and versatile publisher for Commodore computers (from what I have read). I also want a program that works in the native C-128 mode.

Lyle C. Seplowitz
New Brunswick, N.J.

I started using GEOS with only a 64, so didn't question buying geoPublish. Now, I find a wide-spread notion that it is a 64-only program. Actually, the program is almost too much for the 64 unless you can kick up your memory to 512 with a Ram Expansion Unit. Since the 1750 REU gives the 128 this much memory, it is the best computer to use with geoPublish in either mode.

GeoPublish does work with GEOS 128 in 40 column mode. There seems to be an opinion that a 128 version should be made to provide an 80 column display. It is absolutely not necessary as the preview display lets you work on the whole page at once for layout and drawing. We will cover this subject more in our next issue. –Ed.

Commodore Files On An Apple

Convert Commodore GEOS Files To Apple GEOS Files
By Terry Van Camp

Maybe you have a friend with an Apple computer with whom you'd like to share your Commodore geoWrite files with. Or, you could be an Apple user who has been wishing there was some way to get those Commodore geoPaint files you've seen into your computer.

With the same operating system (GEOS) now available on both Commodore and Apple computers, not only can both machines run virtually the same applications, but the files that are created by the applications can be shared between them.

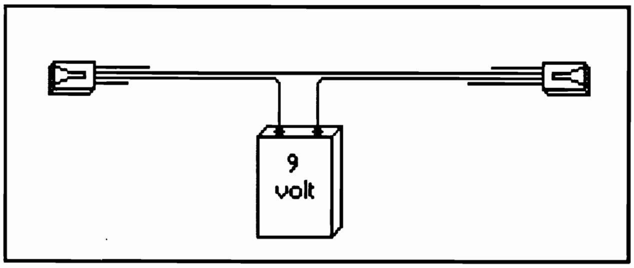
To demonstrate this possibility of sharing files, I have written a program for the Apple that converts Commodore GEOS files into Apple GEOS files. This program works with geoWrite, geoPaint, Font, and Photo Album files. (Programs to convert in the other direction might be available by the time you read this.)

The conversion of files is a complete one -- right down to the icon on the deskTop. A converted file is indistinguishable from its original. There is no way to tell which computer it was created on!

Making the Connection

First, there has to be some way to connect the Apple and Commodore computers together so that you can move the files from one computer to the other. Probably the simplest way to do that is with the use of modems. Modems seem to be becoming more common these days as they become cheaper, simpler to use and more standardized. You could make the connection over the telephone lines in much the same way you do to a telecommunications service, or you could bring the computers together, and make the connection directly. Either way, the process of transferring the files from one computer to the other is the same.

Unless you already feel comfortable using your terminal software, I recommend bringing the two computers together physically so that you can work out your technique. If both computers have modems, the connection pictured below will work. Only the inner-two wires do anything and the battery makes the modems "think" they are on a phone line:



The above figure was taken from a geoPaint file by GeorgeT17 on Q-Link.

Making the Transfer

The Commodore GEOS files to be transferred must be converted to sequential form using the CONVERT utility that allows GEOS file transfers to and from Q-Link. This is the standard form for transferring Commodore GEOS files. GEOS files cannot be transferred "as is". Now you are ready to upload the files to the Apple.

Boot-up the terminal programs on both computers and make the connection. Instruct the Apple to receive a BIN type file and the Commodore to send a sequential type. On the Commodore end, make certain that any transmit translation is turned off. (The file is already in true ASCII) Don't forget to agree on a transfer protocol ahead of time. The most common seems to be XMODEM (sometimes called "Christensen"). From here, the computers will take over and let you know when the transfer is complete.

The Conversion Program

With the Commodore GEOS files (in CONVERTed form) in the Apple, it's time for the fun part--converting the received BIN files to Apple GEOS files.

The program to do the conversion is called "BIN.TO.GEOS". It is written in Applesoft BASIC and has been uploaded to both Q-Link and Apple Link.

BIN.TO.GEOS is in the public domain as Shareware. Shareware is a fairly new idea that I hope will work out in the long run. The idea is that a program is made available for anyone to try out. Only if you decide you want to use the program are you obligated to compensate the author for his/her efforts and expenses. With Shareware, you get to actually try out a piece of software before you buy it! The Commodore GEOS community has been enriched by a number of unique programs written by independent programmers and released as Shareware. (Hopefully this will happen with Apple GEOS also.) These authors certainly will not get rich this way, but with support from their users, they can continue to do what they enjoy and the GEOS community as a whole will benefit.

If BIN.TO.GEOS was downloaded from Q-Link, you will have to transfer it to the Apple in much the same fashion as you did with the

GEOS files. It will already be in sequential form so there is no need to CONVERT it first. Again, don't use any translation features of your terminal software when transmitting. On the Apple end, BIN.TO.GEOS must be received as a TXT type file.

With BIN.TO.GEOS as a TXT file received from a Commodore computer it can be changed to an actual BASIC program file. Boot up PRODOS with BASIC.SYSTEM and enter "EXEC (filename of TXT file)". The computer will now "type" the program for you. (A SYNTAX ERROR at the end doesn't seem to matter.) When it's done, you can LIST the program to verify it's there and SAVE it.

The Conversion Process

Each file to be converted must be put, by itself, on a blank, PRODOS formatted disk. BIN.TO.GEOS expects to see only one file on the disk.

Boot-up PRODOS with BASIC.SYSTEM and RUN BIN.TO.GEOS. Insert the disk, with the file to be converted, into the current drive, ie, the drive you just used. Hit any key to start the process. If the file is very long, expect your drive to get a workout. When "DONE!" appears and the cursor returns, you can insert another disk with a file to convert (One file per disk!) and RUN the program again.

The conversion is much quicker if you can get the file to be converted, onto a RAMdisk. On the IIGS I've been using, I use the GEOS deskTop to move the file to be converted onto the blank RAMdisk. I then reboot the machine with PRODOS and BASIC.SYSTEM. After LOADing BIN.TO.GEOS, I enter "PREFIX/RAM5" before I RUN the program. This selects the RAMdisk as the current drive.

What started as a Commodore GEOS file is now an Apple GEOS file!

Inner Workings

BIN.TO.GEOS takes the standard PRODOS file that contains the Commodore GEOS file information and restructures it into the Apple GEOS file type called Variable Length Indexed Record (VLIR). This is the same file structure GEOS uses on the Commodore.

What BIN.TO.GEOS has to contend with is the difference in the disk operating systems upon which GEOS puts itself in the two machines. In order to maintain compatibility with regular files on the two machines, Berkeley Softworks (BSW) put their disk operations on top of Commodore DOS and PRODOS on the Apple. The GEOS disk operations and file structures are the same on the two computers. It is the underlying disk operating systems that account for the differences. Just as Commodore GEOS files appear as modified Commodore DOS files, Apple GEOS files appear as modified PRODOS files.

VLIR files in Apple GEOS are modified PRODOS "tree" files. PRODOS normally decides for itself when to build a "tree" file, but GEOS VLIR files are always modified "tree" files. BIN.TO.GEOS builds up its own "tree" file out of standard PRODOS "sapling" files. For each GEOS record, a "sapling" file is created. A short machine language routine is used to directly modify blocks on the disk in order to put the "sapling" files together to form a "tree" file.

Thanks to BSW

BSW intentionally maintained the same data structure for the files this conversion works on. BSW intended the two versions of GEOS to be able to share files. I became convinced of this when I found that both versions of GEOS use the same geoPaint file format. This stands out because this format is based on the Commodore hardware configuration!

In speaking with a BSW representative, I learned that this intentional file structure compatibility was abandoned after the original Apple GEOS products. Starting with the Apple version of geoPublish, BSW did not force Commodore structures on Apple files. This does not eliminate the possibility of sharing data files from these latter applications! What it means is that BSW made it "easy" to share the files from the original GEOS products.

The geoWorld

I consider BIN.TO.GEOS to be a very small step in the process of exploring the connections between the two versions of GEOS. The original purpose of the program was to save myself some typing in moving some geoWrite files over to an Apple. It was only after I started delving deeper into the project that the possibilities started to suggest themselves. To tell the truth, I was surprised by what I found. I had started out assuming separate geoWorlds on the two machines. My explorations led to another viewpoint.

The Apple and Commodore communities live in the same geoWorld. GEOS makes it possible for the two communities to come together. Sharing and communicating between the two communities has exciting possibilities that can only benefit both.

--Terry Van Camp

BIN.TO.GEOS and GEOS.TO.BIN by Terry Van Camp are both available on GEOWORLD Disk #5. I highly recommend shareware compensation for these programs that will expand our GEOS world so much more. --Ed.

A Commodore user in Fresno, California has announced the availability of a Font Resource Directory for GEOS. Dick Estel, editor of the Fresno Commodore User Group newsletter, said that the directory displays approximately 380 GEOS fonts.

All point sizes and all supported characters are shown for each font. In addition, "picture" or graphic fonts, such as Dingbats and Evans, are displayed with a "translation." The key pressed is shown next to the character that actually prints.

Production of the directory is a non-commercial venture; however, Estel is requesting $8 to cover printing, mailing and other costs. The directory is a photocopied, dot matrix-printed booklet of over 200 pages. Several supplements have been added to the directory since it was published in July, 1989, and registered buyers will receive one additional supplement displaying at least 25 more fonts.

In addition to the display of fonts, the directory includes an alphabetical index, an alphanumeric listing of font ID numbers, and sources for most of the fonts shown.

To order the Font Resource Directory, send $8 check or money order to:

Dick Estel
3487 E. Terrace
Fresno CA 93703 (209) 224-4163

Inside Geos

A programmer's eye view into the world of Geos

By William Coleman

Well here we are again. Last month we began our odessy into our first application - Convert V2.5. This month we will continue on from where we left off. We'll finish up with the upper-level routines and then continue on to the Geos to Commodore translation routines.

Let's jump right in - take a look at Listing 1. *GetIt* is used to decide what type of file the user selected and which way the conversion needs to go. It will also locate the directory entry from the disk and store it to deBuf and workBuf. If you will remember this routine is called by ConvertOne.

A pointer to the filename is passed to the routine in R6, there is a good reason for this: FindFile is used to locate the directory entry for the file. Once the file has been located delink and deoffs are loaded with the directory sector link and the entry's offset into the sector respectively.

Next the routine fills workBuf and deBuf with the directory entry itself. workBuf will be massaged into a regular commie directory entry; deBuf will be saved for insertion into the convert sector.

Now it's time to check to see what kind of file we are dealing with so that the proper conversion routine can be chosen. If the filetype is DEL or REL conversion isn't possible; if it's a USR file then it must be checked to see if there is a header link in the directory entry, if not then we are dealing with a Commodore USR and conversion is again not possible.

If all of the tests so far have failed then we know that we are dealing with a PRG or SEQ file. So now we need to retrieve the first sector of the file from the disk and check for the Convert ID string. If the string is there then we need to convert back to

Listing 1

```
GetIt:
;Locates file on disk and loads delink and deoffs
;pass: r6 - filename
;ret:  workBuf,deBuf - directory entry
;      flag - 0 = Geos file, 1 = commie file, 2 = unconvertable
;      X - disk error
        jsr     FindFile
        jsr     CheckError
        MoveW   r1,delink
        MoveW   r5,deoffs
        ldy     #ENTSIZE
10$:    lda     dirEntryBuf,y       ;Move dir. entry...
        sta     workBuf,y           ;...to workbuf and debuf
        sta     deBuf,y
        dey
        bpl     10$
        lda     dirEntryBuf
        and     #%11                ;mask out lock and valid bits
        beq     40$                 ;must be a DEL file
        cmp     #4
        beq     40$                 ;is a REL file
        cmp     #3
        bne     15$                 ;must be a PRG or SEQ
        lda     dirEntryBuf+HLOFFS
        beq     40$                 ;no header - C= USR
        bne     50$                 ;Geos file
; at this point we know we have a commie file - check if it's converted
15$:    MoveW   dirEntryBuf+1,r1    ;r1=starting track/sector
        LoadW   r4,diskBlkBuf
        jsr     GetBlock
        jsr     CheckError
        ldy     #ENTSIZE-5
20$:    lda     diskBlkBuf+$23,y
        cmp     keyText+3,y
        bne     40$
        dey
        bpl     20$
30$:    lda     #1                  ;convert to Geos
        .byte   $2c
40$:    lda     #2                  ;can't de-convert
        .byte   $2c
50$:    lda     #0                  ;convert to commie
        sta     flag
        rts
```

Inside Geos

Listing 2 (continued on next page)

```
ToCom:                  ;Converts from geos to C= format.
;pass debuf, workbuf - directory entry
        jsr     GetDirHead
        InitW   r3,1              ;start @ trk1 sec1
        jsr     SetNextFree       ;Find free sector
        jsr     CheckError
        MoveW   r3,chedlink       ;Save it's link
        jsr     PutDirHead        ;Re-write BAM
        MoveB   comType,workBuf   ;Change filetype
        MoveW   chedlink,workBuf+1 ;Pnt at new blk
        ldy     #3                ;name to PETSCII
10$:    lda     workBuf,y
        cmp     #$a0
        beq     30$
        cmp     #'a'
        bcc     20$
        cmp     #'z'+1
        bcs     20$
        sub     #'a'-'A'
        sta     workBuf,y
20$:    iny
        cpy     #19
        bne     10$
30$:    ldx     #HLOFFS
        lda     #0
TC1:    sta     workBuf,x         ;Zero tail end of entry
        inx
        cpx     #ENTSIZE-2
        bne     TC1
        AddVW   1,workBuf+SOFFS
        MoveW   delink,r1
        LoadW   r4,diskBlkBuf
        jsr     GetBlock          ;Get dir. block of file
        jsr     CheckError
        MoveW   deoffs,r5         ;Offset in entry sector
        ldy     #ENTSIZE
TC2:    lda     workBuf,y         ;Move new dir. entry
        sta     (r5),y
        dey
        bpl     TC2
        jsr     PutBlock
        jsr     CheckError
        MoveW   deBuf+HLOFFS,diskBlkBuf
        MoveS   deBuf,diskBlkBuf+2,ENTSIZE
        MoveS   keyText,diskBlkBuf+$20,$1c
        lda     comType           ;If PRG then overwrite
        cmp     #$82              ;'SEQ' with 'PRG'
        bne     TC3
        MoveS   prgSeqText+1,diskBlkBuf+$20,3
TC3:    MoveS   version,diskBlkBuf+$42,4
        MoveS   driveType,diskBlkBuf+$47,3
        MoveS   PrntFilename,diskBlkBuf+$4b,16
```

Geos, if it's not then conversion isn't possible.

When GetIt exits *flag* will contain 0, 1, or 2. This number will be used as an index into a table of functions. If you recall ConvertOne will use this index to execute the appropriate conversion subroutine (or an error box if the file can't be converted).

Converting to Commodore

Now it's time to get into the heart of Convert - the conversion routines. This month we will look at the Geos to Commodore routines. Take a look at Listing 2.By the time we've reached this point debuf and workbuf contain a copy of the file's directory entry.

One point I should bring up: there are two macros in this routine that you probably haven't seen before. Here's what you must add to your geosMac file:

```
.macro  InitW   dest,value
   lda  #value
   sta  dest
   sta  dest+1
.endm

.macro  MoveS  source,dest,number
   ldy  #number
 loop:
   lda  source,y
   sta  dest,y
   dey
  .if (number > 127)
   cpy  #$ff
   bne  loop
  .else
   bpl  loop
  .endif
.endm
```

The first thing that ToCom needs to do is to allocate a block for the new convert sector and update the BAM on the disk

Next it modifies workbuf into a regular directory entry. First the Commodore filetype and the link to the new sector are inserted.Then the filename is converted to PETSCII.

Inside Geos

Now the header link, time, etc. are erased (cleared to zero). The last thing it does it to increment the block count to reflect the addition of the convert block.

Now that we have a modified directory entry it must be written to the disk. GetIt saved the track, sector, and offset for us so it's simply a matter of reading in the sector, inserting the modified entry and writing the sector back out to the disk.

Now it's time to create the convert sector.take a look at the code: it's really nothing more then moving a bunch of strings into diskBlkBuf. Note that a link to the file's header block is inserted also.I'm not going to go through each one, they are rather self-explainatory.

You may be wondering why some of the strings are there, the answer is simple: who knows? That's the way the BSworks' Convert does it so that's the way we have to do it to maintain compatibility.

Once the sector is completed it is written out to the disk.

Now the header block must be modified so that it will point to the body of the file (or the index sector if the file is a VLIR). ToCom reads it in, adjusts the link, and then writes it back to the disk.

By the way, an astute reader may be wondering why some of the labels are not local. Unfortunatly geoAssembler will barf ontoo many local labels (MoveS, AddVW, etc. all add some) so I broke things up a bit.

The last thing that ToCom will do is check to see if the file is a VLIR file. If it is the routine *CrunchVLIR* will be called. If not then the Dialog Box rountine *CSuccess* is called. We'll look at the DB routines in a future article.

Error Handling

Before we look at the VLIR routine let's take a quick detour through the error handling routines, they are really quite simple. All they do is check X for an error and if there is one then abort processing and call DiskError. Note the use of PLA:PLA which pulls the return address of the caller off of the stack.

There are two routines: one is used by ToCom and one is used by the VLIR routines. The only difference is that the VLIR routines calls InitForIO so the error handler must call DoneWithIO.

Listing 2 (continued from preceding page)

```
        MoveS   $c9ef,diskBlkBuf+$5c,16
        MoveS   mbText,diskBlkBuf+$a0,23
        LoadW   r4,diskBlkBuf
        MoveW   chedlink,r1
        jsr     PutBlock           ;Write the new sector
        jsr     CheckError
        MoveW   deBuf+HLOFFS,r1
        jsr     GetBlock           ;Get file's header
        jsr     CheckError
        MoveW   deBuf+1,diskBlkBuf ;Link it to main file
        jsr     PutBlock
        jsr     CheckError
        lda     deBuf+STOFFS       ;Check structure
        beq     10$
        jsr     CrunchVlir         ;It's a VLIR, link it
10$:    jmp     CSuccess           ;we're done!
```

Listing 3

```
ChkError:
;Checks X for disk error. Will NOT return if error
        cpx     #0
        bne     10$
        rts
10$:    jsr     DoneWithIO
        jmp     Abort

CheckError:
;Checks X for disk error. Will NOT return if error
        cpx     #0
        bne     Abort
        rts
Abort:  pla                        ;yank return address
        pla
        jsr     DiskError
        jmp     Convert1
```

VLIR Conversion

If ToCom discovered that the file was a VLIR file then CrunchVLIR is called to collate the VLIR records into a single file. Following how the routine works can be a bit confusing so you need to follow closely.

First the Index Sector is read into diskBlkBuf. What is going to happen is that each link in the table will be converted to a block count and a last

Inside Geos

Listing 4

```
SetIndex:
;Sets up r1 and r4
        MoveW   deBuf+1,r1
        LoadW   r4,workBuf
        rts

CrunchVlir:
;Collates the records from a Geos VLIR file.
        jsr     EnterTurbo
        jsr     InitForIO
        jsr     SetIndex
        jsr     ReadBlock           ;Get index sector
        jsr     ChkError
        LoadB   flag,0
        LoadB   offset,2            ;Aim past link
01$:
        tay
        lda     workBuf,y           ;Skip if track link is 0
        beq     05$
        sta     diskBlkBuf
        lda     workBuf+1,y
        sta     diskBlkBuf+1        ;Put link in buffer
        lda     flag
        bne     02$
        MoveW   diskBlkBuf,workBuf
        bra     03$
02$:
        jsr     WriteBlock
        jsr     ChkError
03$:
        LoadB   blkcount,0
        LoadW   r4,diskBlkBuf
04$:
        MoveW   diskBlkBuf,r1       ;how long is record?
        jsr     ReadBlock
        jsr     ChkError
        inc     blkcount
        lda     diskBlkBuf          ;Loop if a valid track
        bne     04$
        ldy     offset
        MoveB   blkcount,"workBuf,y"
        MoveB   diskBlkBuf+1,"workBuf+1,y"
        LoadB   flag,$ff            ;Flag set after first record
05$:
        inc     offset
        inc     offset              ;Offset=offset+2
        lda     offset
        bne     01$                 ;Done when offset wraps
        jsr     SetIndex
        jsr     WriteBlock
        jsr     ChkError
        jmp     DoneWithIO
```

sector byte count.The index sector will also need to be linked to the first record and the last sector of each record will need to be linked to the following record.

Once the Index Sector has been read in each record will be checked for validity. If the track number is zero the the record is empty and it can be skipped.

If the track is valid then CrunchVLIR will step through the sectors to make a count. Note that ReadBlock is used for maximum speed. Once the count is obtained it will be stored in workBuf along with the number of bytes in the last sector. These bytes are written over the track and sector link for that record.

Note the use of *flag* . The first time through the loop it will be zero. This will cause the track and sector of the first record to be moved to workBuf's link. All of the remaining iterations will store the track and sector to the last sector of the previous record and the write the record back to the disk.

Once the last record has been processed workBuf will be written back to the disk. This completes the conversion process.

That's all for this month. Next time we will look at the Commodore to Geos routines.

I have put together a brand new disk containing the complete source for Convert plus several other goodies including color handling, and a Desk Accessory manager.To order a copy simply send a check or money order for $10.95 to:

William Coleman
BlasterPak II
1431 Pacetti Rd
Green Cove Spgs, FL 32043

As always if you have any problems, questions or suggestions about Geos, feel free to leave me EMail on Genie (my address is WC.COLEMAN) or drop me a line to the above address. Happy 'puting.

geoWorld Disk #5

To get the most out of the public domain programs on GEOWORLD disks, it is best to contact the authors for help or user input. Shareware donations not only allow these programmers to continue their important work, but give them a list of serious users for upgrades and offers.

Listed here are the names (and/or Q-Link screen names) and addresses of those responsible for the files on GEOWORLD Disk #5:

Qwik Top -- John F. Howard
(ILLINI70)
4433 Clemsford Drive
Virginia Beach, VA 23456

WormDesk -- FontSwap -- PicShow
Payton Snider (GeoWorm)
7427 West Coolidge Street
Phoenix, AZ 85033

Selector 64 -- John L. Brown
(JohnyBoy)

geoAlbum 1.0 & 1.1 -- Jean F. Major
119 Terrasse Eardley
Aylmer, Quebec, Canada J9H 6B5

PHOTO MOVER -- Rick Coleman
P.O. Box 44
Sheridan, WV 82801

geoSidPlayer --Roger Lawhorn
(ROGER LU)
3632 Gray Fox Drive
New Albany, IN 47150

GEOS.TO.BIN and BIN.TO.GEOS
-- Terry Van Camp
(Terry V7) 16604 Cypress
Strongsville, Ohio 44136

ABC Picture Font -- (Melvena)

Cockroach pointer -- (TomCat69)

Sword pointer -- (TED 406 W)

Starfonts -- Thomas L. Dively
(Starman35)
8583 Greenbelt Rd. Apt T3
Greenbelt, MD 20770

GEOS
Hardware News

geoRam

The biggest improvement to enhance GEOS on the Commodore is adding the Ram Expansion Unit. Until now the only choice was the 17XX REU. The best choice is the 1750 with 512k which plugs directly into the C=128. Those with C=64 must use a heavy duty power supply to run the 1750 or get the 1764 and add an additional chips for 512k. Those who sought the Commodore REU know that they are scarce and expensive.

Berkeley Softworks recently anounced that they are marketing the "geoRam" which is a 512k REU designed to work specificaly with GEOS. The new geoRam uses CMOS chips which draws less power and can be plugged directly into the C=64 without the need for a larger power supply.

There were rumors about GEOS on an EPROM and possible 1 meg of RAM but sources at BSW have reported that these will not be included. The geoRam acts like the 1750 with the exception that since it lacks a controller chip the DMA options are lost which means that in certain applications such as screen refresh in geoPaint will be slower than the 17XX using DMA to move data.

The geoRam is only compatable with GEOS and comes with an upgraded version of GEOS (2.0 R) which contains a new kernal, configure and RAM driver. The price for geoRam is $124.95 and is available by direct order from Berkeley Softworks. Units will be shipping in mid December please allow 2-3 weeks for delivery.

Hard disk and more

CMD has announced version 6.0 of JiffyDos which replaces the ROM inside the Commodore 64 and 128. The enhanced ROM will speed up GEOS 128 operation of the 1571 and 1581 drives GEOS 64 uses its own built in speed up system that bypasses JiffyDos.

Ramlink is a pass-through card for the 17XX REU which allows connection of its own independent power supply which eliminates loss of data upon powering down the computer. It also includes its own operating system that allows most software to run in the REU as a high performance RAM Disk. Connection is available to RAMCard which will allow up 1 meg to be used under GEOS and a parallel port to connect with the CMD HD series GEOS compatable Hard Drive for ultra fast access.

Prices for the CMD HD20 will be $599 and should be available towards the end of December. RamLink will be approximatly $79.99 and available sometime in March 1990. These products are still under development and the price / release date may vary due to the complexity of the project.

For more information contact:

Creative Micro Designs
50 Industrial Drive, PO Box 646
East Longmeadow, MA 01028

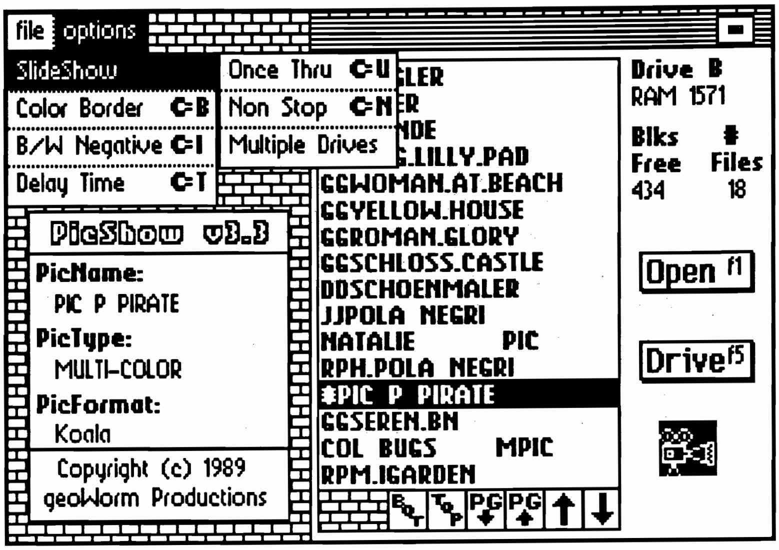
GEOS GEMS - Invaluable Utilities for GEOS

By Peter T. Hughes

PIC and SID Utilities

PicShow

Picshow 3.3 by Payton W. Snider II (GeoWorm on QLINK) is a utility for viewing many standard Commodore 64 Graphic Picture formats in GEOS. This program is a graphic display, slideshow and conversion program all rolled into one. Picshow will display hi-res formats such as JJ, Doodle, Art Studio, B/W Hi-Res, RUN Paint and multi-color formats such as GG, Koala, Advanced Art Studio and RUN Paint. You can show a continuous slideshow on three drives, set delay between pictures, have a forced black border or have the picture select the border color. Hi-res pictures can be saved in the above hi-res formats and as GeoPaints and multi-color pictures can be saved in any of the above multi-color formats and as Blazing Paddles and Artist 64. You can display up to 164 files - great for 1581 users. This program has many keyboard and mouse shortcuts. Picshow is on QLINK and GEOWORLD Disk #5. Also contact the author with comments at this address:
Payton W. Snider II, 7427 W. Coolidge St., Phoenix, AZ 85033.



PicShow v3.3

PicName:
PIC P PIRATE
PicType:
MULTI-COLOR
PicFormat:
Koala

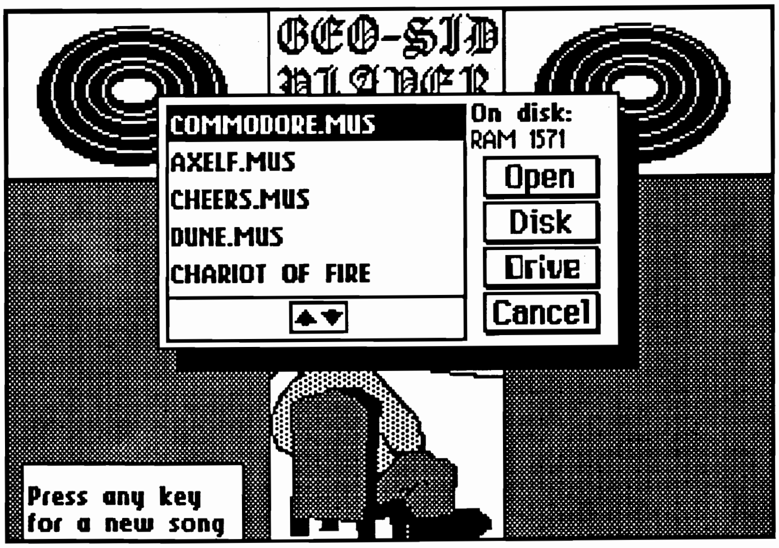
Copyright (c) 1989
geoWorm Productions

Drive B
RAM 1571
Blks Free 434
Files 18

LER
R
NDE
.LILLY.PAD
GGWOMAN.AT.BEACH
GGYELLOW.HOUSE
GGROMAN.GLORY
GGSCHLOSS.CASTLE
DDSCHOENMALER
JJPOLA NEGRI
NATALIE PIC
RPH.POLA NEGRI
*PIC P PIRATE
GGSEREN.BN
COL BUGS MPIC
RPM.IGARDEN

GeoSidPlayer

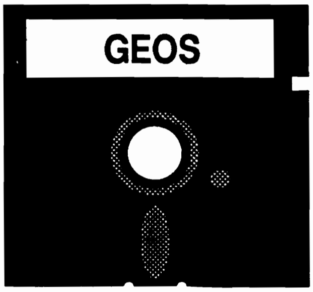
GeoSidPlayer by Roger Lawhorn (ROGER LU on QLINK) is a utility for playing Sid Player songs in the GEOS environment. Another reason for not leaving GEOS. Sid Player songs are those music files that have their file names end in .mus. QLINK has a vast selection of Sid Player songs in the Music Room Software Library. This program works with two drives but the file selection box only holds 15 filenames. Maybe a newer version will hold more files for 1581 users who have a large collection of sid songs on a 3.5 inch disk. While a song is playing a man wearing head phones and tapping his toe is sitting in a chair between two large speakers. The speakers show a spiral design also. GeoSidPlayer is on QLINK and GEOWORLD disk #5. Send $3.00 shareware donation and comments to the author at this address:
Roger Lawhorn, 3632 Gray Fox Dr., New Albany, IN 47150.



COMMODORE.MUS
AXELF.MUS
CHEERS.MUS
DUNE.MUS
CHARIOT OF FIRE

On disk:
RAM 1571

Press any key
for a new song

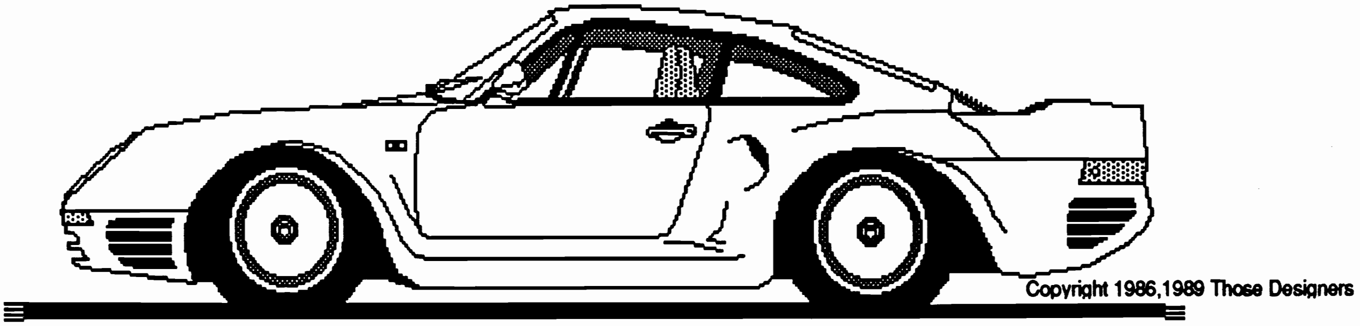


WANTED: GEOS GEMS

Any Programmers or users of GEOS who find other useful programs for GEOS, please let me know and send them on disk to me. I am always looking for new utilities. My address is: Peter T. Hughes, 151 Randolph St., Canton, MA 02021. With your permission these programs may appear in a future GEOS GEMS column in GEOWORLD. You can contact me also on QLINK as GeoLib PH - the GEOS ARENA Software Librarian.

Get in the FAST LANE!

DISKART will make your GEOS documents real winners!



Copyright 1986,1989 Those Designers

DISKART 1
Graphic Goodies 1
Graphic Goodies 2
Weather Stuff
GEOpaint Tips 1
GEOpaint Tips 2

DISKART 2
Graphic Goodies 3
Little Guys 1
Holidays 2
Workdisk Labels
Musical Stuff 1
GEOpaint Tips 1

DISKART 3
Vehicles 1
Vehicles 2
Porsche 959
Tin Lizzies
Warbirds 1
DC-3 Airliner
F4 Phantom

DISKART 4
Little Guys 2
Foodstuff 1
Ovals, Blocks, etc
C64 & Periphs
Tools 1
Gardening Stuff

DISKART 5
Make-A-Face
Drafting Equipment
Vehicles 3
Banners 1
Flowchart Symbols
Little Women 1
Little Women 2

DISKART 6
Farm Animals
Critters 1
Zoo Animals
Dogs 1
Cats 1
Fish 1
Space Creatures

DISKART 7
Big Boats
Small Boats
Tropical Stuff
Baby Items
Graphic Goodies 4
Man/Woman Heads

DISKART 8
Graphic Goodies 5
Graphic Goodies 6
Silly Pictures
Holidays 3
Computer Humor

DISKART 9
Warbirds 2
Trains 1
U.S.Jets
Vehicles 4
Old Vehicles

DISKART 10
Church Pix 1
Church Pix 2
Dinosaurs
Cut 'n Fold House
Cut 'n Fold Boat

DISKFORMS 1
Instructions
Blankforms 1 thru 5
Lined Paper
Delivery Receipt
Inventory form
Bank Deposit form

MUSI-KIT
MUSI-KIT
Musi-Kit Info
Single Title
Single Staff
Piano Title
Piano Staff
Music Sample
Large Instruments

Order TODAY! **Still Only $8.50 per disk** - Please circle your choice(s) below

DISKART 1	DISKART 2	DISKART 3	DISKART 4
DISKART 5	DISKART 6	DISKART 7	DISKART 8
DISKART 9	DISKART 10	DISKFORMS	MUSI-KIT

Name____

Address______________________________________

City____________________ State/Prov ______________

ZIP/Postal Code__________ Country____________________

Date of Order __________ Telephone __________________

California Res. Add 6.5% Sales Tax - Check or Money Order ONLY (U.S.Funds)
For Orders Outside U.S. Add $2 per disk for shipping - Defective merchandise will be replaced - no returns

Total No. of disks___
Sales tax ___
Shipping ___

Total Due _____

USE YOUR DISCOVER CARD!

Card No.

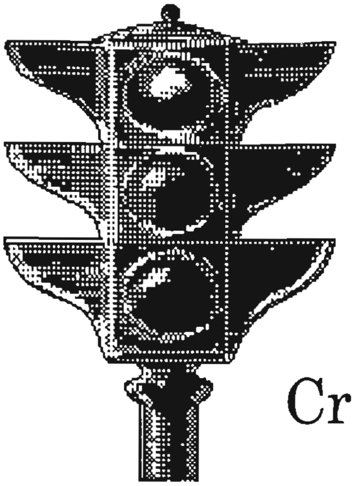
Name (as on Card)

Exp. date

Those Designers

Makers of DISKART, The Original GEOS-ready graphics.

3330 Lewis Avenue, Signal Hill, California 90807 U.S.A.



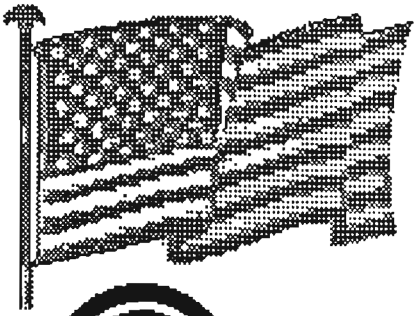
ANNOUNCEMENT

Lamb Art & Design Now Has 5 More Disks!

Creative Graphics, Fonts & Ideas in GEOS Format. High Quality at a Low Price.

Volume 4 -
FLYER STARTERS
Special full page graphics and megaFonts for quick and easy flyers with geoPublish.

Volume 5 - CARD MAKER
Special graphics and fonts designed for cards and stationery. GeoPaint and geoPublish templates in stock sizes makes creating professional-looking cards easy. This disk includes files of my cards and stationery which won 3rd place (open category) in the BSW desktop publishing contest..



Volume 6 - CATEGORY CLIPS ONE
This volume contains albums of clips in specific categories. Holidays, Religion, Seasons, Pointers, Sports, Ethnic, Tools and Food. More category clip disks will be released soon.

HUNDREDS OF BIG, BOLD GRAPHICS IN A VARIETY OF STYLES

Volume 7 - FONTS TWO - ALL SIZES
Many great fonts in many sizes. If you like my fonts on the RUN Magazine disk, here's more.

Volume 8 - OBJECT-ORIENTED CLIP ART
As seen in geoWorld and on GW Disk 3. Use with a laser printer for a smooth look or as regular clip art with a dot matrix printer

STILL AVAILABLE:
Volume 1- GRAPHIC ELEMENTS
Volume 2 - FULL PAGE BORDERS
Volume 3 - FONTS ONE - HEADLINE FONTS

GRAPHIC IDEA FILES
ALL DISKS - $10.00 EACH.

Send check or money order to:
LAMB ART & DESIGN
3575 EAST CO. 18TH STREET
YUMA, AZ 85364

BACK ISSUES

Issue Number 1-5 sold out
Issue Number 6-20 $2.50 each postpaid

Send check or money order to:

GEOWORLD
38 Santa Ynez St.
Santa Barbara, CA 93103

12 Issues - $20.00
Canada $30.00 - Overseas Airmail $50.00
US funds
send check or Money order to:
GEOWORLD
38 Santa Ynez St.
Santa Barbara, CA 93103

Subscribe to GEOWORLD!

Name ______________________________

Address ______________________________

City, State, Zip Code ______________________________

□ New Subscription

□ Renewal