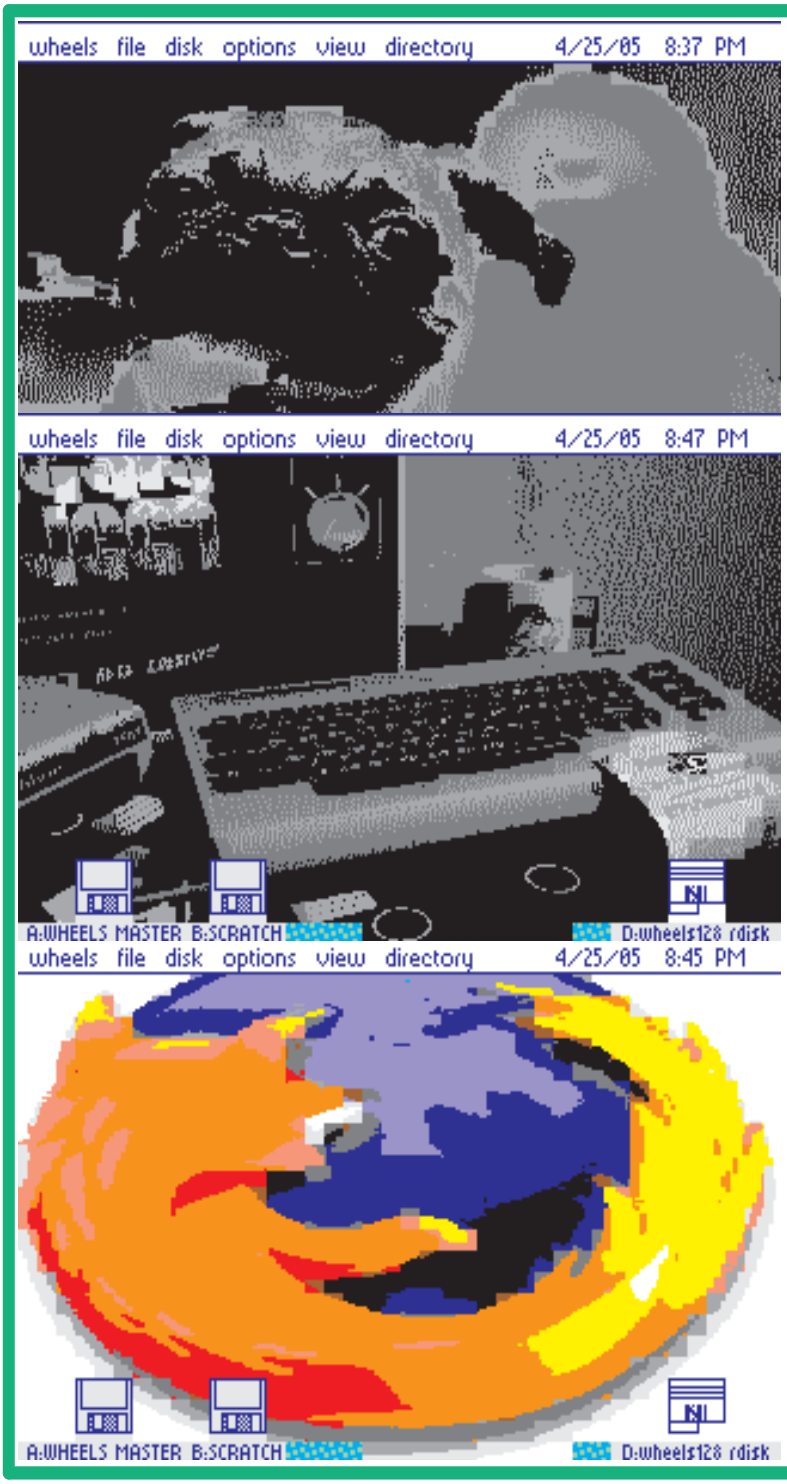
COMMODORE CURRENTS



VICE and Wheels OS

A How-To By Todd S. Elliott

Top: Pug Dog.
Middle: Old Skool, by WiDDY of Genesis*Project
Bottom: Firefox (tm) Logo

Plus- A Wheels Wallery!

Welcome to the third installment of ***Commodore Currents***! My apologies for yet another late issue! Can you believe **19 months** have passed in between issues?!? A lot of the delay can be attributed to general procrastinating, but I also needed to work on my **geoPublish v1.1** upgrade if I am to continue with my Commodore Currents publishing endeavors.

This issue is all about **VICE**, the **Versatile Commodore Emulator**, a comprehensive emulation suite covering nearly all **Commodore** 8-bit computers, from the original PET 2001 to the Commodore 128. VICE can run nearly all OS'es, most notably **Windows**, **Linux** and **MacOSX** platforms. More specifically, this issue is more about how to get the most out of your **Commodore GEOS** experience in using the VICE **x64** and **x128** emulators.

Oh, there's a **Wallery** of sorts, giving you bountiful eye candy for you to enjoy! In a way, this issue also introduces a pet project of mine, **WheelWizard**. I have fully disassembled **Jim Collette's** original **geoWizard** program and have made the necessary modifications for it to work under **Wheels** 64 and 128 systems. I have also made some minor changes to the **geoWizDump** program just to get it working under Wheels 128. I will do further work on this screen dump program so it can work under Wheels 64 and on any kind of disk drives. Currently WheelWizard is undergoing beta-testing and seems to have fallen victim to my procrastinating side for the moment.

I've recently discovered **PERL** on my **FC3** Linux setup. I know PERL has been around for ages, but for me, it's been a pleasant discovery! PERL is somewhat like **BASIC** for the C64/128, but a whole lot better and more powerful. Where is all of this leading...? To **DriveGhost**, naturally! :) I'm a user of **Nick Coplin's** long overdue program and have used it to image my **CMD HD** contents to my **64HDD** partition of my mainstream computer setup via the **XE1541** cable. But there was one big problem left; I have all of these CMD HD images now in my 64HDD partition and no means of accessing its contents.

Enter PERL to the rescue! Ok, I had to provide some programming elbow grease entirely on my own, but the end result is the first preliminary version of **CMD DOS Imaging Utility** or **CMDIMG**. In this initial foray into PERL programming, the cmdimg utility can now list the directories and files that are residing in a CMD (native partition) DOS image that was imaged by DriveGhost.

Hopefully this utility can be expanded to handle all disk image formats that DriveGhost generates in addition to regular CBM DOS disk images (i.e., d64's). I expect to program more features in cmdimg to allow extraction/writing of Commodore files that are housed in CMD/CBM DOS disk images, and as well as GEOS support. And speaking of GEOS support, I hope to also add in 17xx REU ramdisk (VICE's .BIN files, not the real thing!) support into cmdimg, supporting Wheels OS ramdisks. This way, I can better support my investment in DriveGhost and allow for better management of my CMD HD/RL images and its contents.

As for the crystal ball's contents for the next issue of Commodore Currents, this one will prove to be a blockbuster! I hope to publish a single special **C64 DEMOS** Commodore Currents issue, which will reprint a year's worth of C64 demo-related articles and contains tons of eye candy, erm, C64 demo screenshots galore! Hopefully this special issue will be released sometime this year and not in 19 months.

This issue also marks the first time that Commodore Currents can be printed in a magazine format as well as being published in a PDF newsletter format. I have access to a 11"x17" printer and can print the issue in a full-color magazine format. The tentative cost is $5 USD and S&H costs are extra. I know that $5+ is a bit much for a 12 page issue, but color printing on 11"x17" printer is not cheap. At any rate, the PDF version is freely downloadable, and is in full color.

In the meantime, *enjoy* this issue!

This publication, while not perfect, strives to publish material that recognizes all relevant intellectual properties as belonging to their respective authors. This publication recognizes all valid trademarks as belonging to their respective holders and no infringement is intended.

To the extent practical, I have secured some permissions from the author(s) to reprint the relevant works here. If you, the reader, possess an intellectual property interest in any material presented in this publication, please let me know of any objection(s) you may have.

I have included a short bibliography here to give due credit(s) to various authors and other program(s) that have contributed to the successful creation of this publication.

Wheels OS
All by **Maurice Randall** at:
www.cmdrkey.com
geoWrite v2.2
geoPublish v1.1
All original code by **Berkeley Softworks** and licensed to **CMDRKEY**. All code modifications done by **Todd S. Elliott**.

(*Continued on Page 9.*)

Never the Twain Shall Meet,
Wheels and the VICE Emulator Suite?

The Emulation Endgame

Whoever once said that 'imitation is the best form of flattery' probably knew about the **Commodore 64**! The Commodore 64 and 128 computers have become timeless classics, and as a result, programmers have undertaken projects to recreate the C64/128 experience on modern computers via emulation. The first few endeavors were rudimentary in nature, mostly focusing on the C64 itself. It was not until recently that more emulators began to support peripherals associated with CBM computers, where a complete C64/128 user experience could be recreated via emulation.

The **Windows** and **Linux** computing platforms have had the luxury of running one of the finest computer emulators, called **VICE**. I consider this endeavor a luxury because it is an open source project undertaken by hobbyist programmers for the past few years, without any monetary reward. VICE's emulation of the C64 and C128 has approached near perfection and contains many features that makes it a nice platform for a Commodore GUI enthusiast to emulate **Wheels 64** and **128**. I have nothing but praise for the VICE programming team and will support them in their endeavors in creating the perfect CBM emulator experience.

As many Commodore hobbyists know, Wheels 64 and 128 is the next generation of **Commodore GEOS** and requires horsepower in form of memory expansion, and maximizes the users' investment in **CMD** peripherals. Many would not consider running it under an emulator.

VICE nicely fills the bill! For example, VICE has the ability to emulate a **17xx REU** from 128Kb to 16Mb. VICE also retains the REU's contents, saving it for the next emulator session. This concept emulates a battery-backed REU. VICE can go faster than 1MHz, thanks to its warp mode, making the Wheels GUI more responsive. VICE can emulate a **1351 mouse**, and a GUI simply cannot do without one. VICE can emulate a **1581 disk** drive in addition to nearly every CBM drive in the universe, even those PET IEEE disk drives. Wheels will greatly benefit from 1581 usage.

The list goes on...It seems like Wheels 64 and 128 is in a perfect marriage with VICE. Yet, there are pitfalls and shortcomings, which may lead to an undesirable GEOS experience while using it under an emulator. I hope to prevent such disappointments and to illustrate an ideal emulator experience for anyone wishing to run Wheels 64/128 under VICE. Before I go any further, it doesn't really make any difference if the Windows or LINUX version of x64/x128 is used; Both versions are pretty much nearly identical, save for a few differences.

It seems like Wheels is in a perfect marriage with VICE. Yet, there are pitfalls, which may lead to an undesirable GEOS experience while using it under an emulator.

First Things First

In order to emulate Wheels OS, you need to obtain a Wheels OS disk image in a .D81 format. I simply fired up my c128d setup, ran Wheels 128 and used the **MakeSysDisk** utility to create a bootable 1581 disk. Also, you cannot *install* Wheels OS for the first time under the emulator; you will need to do it on a real Commodore 8-bit setup. Next, please go to **Maurice's** Wheels Upgrades site (**www.cmdrkey.com**) and download the latest patches for your Wheels OS version. Following Maurice's relatively painless upgrade procedure will upgrade your entire Wheels OS system to version 4.4. Do this upgrade on the boot disk and not to your original Wheels OS diskette.

MakeSysDisk

Then you can use **Disk Dumper** (**dd**) under LINUX or **1581COPY** v0.54 by **Wolfgang Moser** for Windows 9x/**MS-DOS** systems to extract the contents from this 1581 disk onto a .D81 disk image. The docs are included with 1581COPY. As for dd, there's no docs covering the 1581, beyond what is contained in the **fdutils** docs. Here's a quick summary:

(enter SuperUser mode via **su**.)

$ **mknod /dev/fd0cbm1581**

$ **setfdprm /dev/fd0cbm1581** ***1599*** **10 2 80 2 0x2A 0x02 0xDF 0x2E**

(If you have floppycontrol, type:

$ **floppycontrol --autodetect /dev/fd0 31,7,8,4,25,28,22,21**)

(You'll notice that in the setfdprm command, I used 1599 instead of 1600 as suggested in the fdutils docs. This way, only 1,600 blocks are extracted.)

(Exit SuperUser mode.)

$ **dd if=/dev/fd0cbm1581 of=/home/name/imagename.d81**

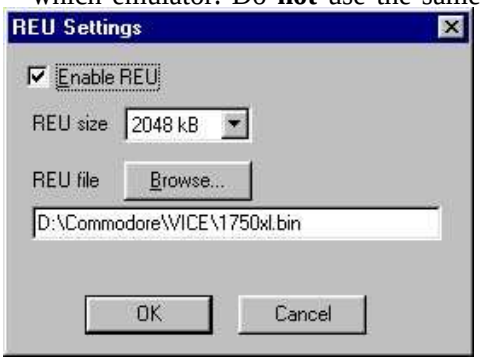
(Where *name* refers to your home directory and *imagename* refers to the name of imagefile you want.)

Disk Dumper (dd) should do its job and create a .D81 image file for you to use under the VICE's x64/x128 emulators. On my setup, dd took a long time in imaging a 1581 disk. My recommendation is to go with 1581COPY under MS DOS mode. *Please do not use these utilities on your original Wheels 64/128 master disk!*

At this point, you should have a .D81 disk image containing a bootable Wheels OS setup. Please try to take some precautions and safeguards in protecting this disk image. You should put this disk image in a system that isn't online to the Internet all the time, or is protected by a firewall. If you sell or remove the hard disk drive in your system, please 'shred' the disk image; This procedure will repeatedly write 1's & 0's over and over and is better than merely deleting the file. **Maurice Randall** has truly put in a lot of effort in producing a Wheels OS upgrade and I'd hate to see his GEOS endeavors vanish suddenly, because of a careless mistake made by an individual exposing his works to indiscriminate copying/piracy. In short, Wheels OS is simply too valuable and deserves some thought and concern over its integrity on a users' system.

Configuring VICE

Let's focus on the key peripheral component powering Wheels OS; The REU. I strongly recommend that a 2Mb REU with the image name, '*1750xl.bin*' be created for the x128 emulator, and/or '*1750xl64.bin*' be created for the x64 emulator. This way, you'll know which .BIN file belongs to which emulator. Do **not** use the same REU image file for both x64 and x128 emulators. Do **not** exceed 2Mb for each REU image file you create, even though you could create up to 16Mb. (It requires a Dashboard hack.) You may want to backup this REU image .BIN file on a regular basis, just in case if something happens to it.



Next, you can set VICE to **PAL** or **NTSC** mode. It really doesn't matter as far as Wheels OS is concerned. However, I have noticed lately that the development team behind VICE are from PAL countries and have generally optimized their code for PAL usage. So, setting VICE to PAL mode may result in better improvements in speed and visual quality, even if you are a North American NTSC user. Hence, my suggestion is to set your VICE configuration for PAL mode in running a GEOS/Wheels program.

> Wheels is a graphical environment and requires tons of disk access & space. Only the 1581 drive will do, adequately meeting Wheels' requirements

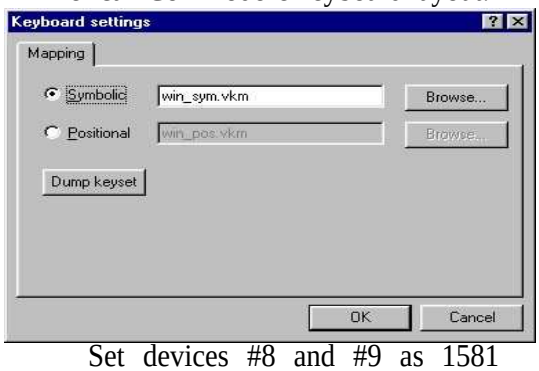
Lastly, turn off **PAL Emulation** feature as it will mimic the 'fuzziness' of an actual PAL TV or 1702 monitor output. This feature is separate from PAL 'mode' and is great for demos, but is poor for GEOS usage.

With respect to the **VDC** 80 column mode, I strongly recommend you to set it to 64Kb of RAM under VICE. Wheels 128 will use a color screen at 80x25 (640x200) resolution if 64Kb VDC is enabled. If 16Kb is on, Wheels 128 will **truncate** the screen to 80x22 (640x176) resolution to display color. You can easily switch between the 40 column mode (VIC-II) and 80 column mode (VDC) under x128 (Windows version) by simply pressing the F3 key. This mimics the **ALT** key on the real machine.

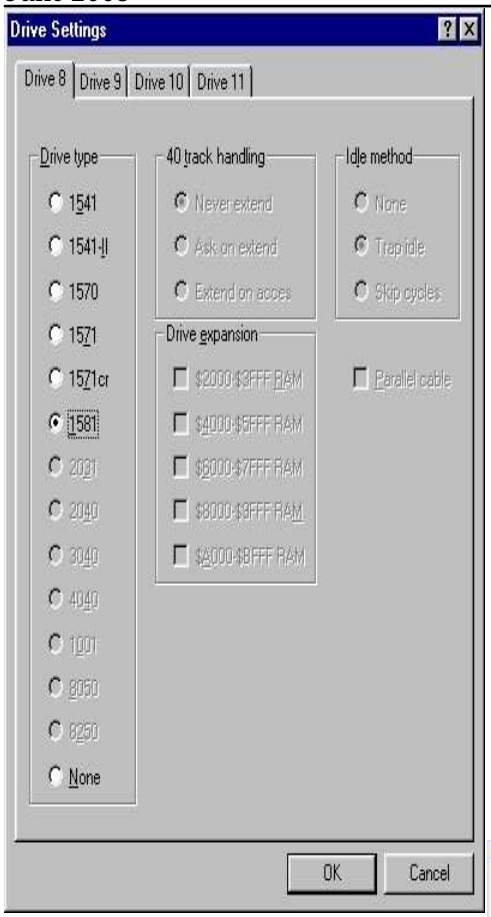
For both **VIC**/VDC screens, you can double-size and double-scan them. That way, the screens are larger and more crisp. This is largely a matter of preference and, on fast computers, the emulation speed penalty switching from a small screen to a larger one is negligible. Do turn on the **Scale2x** feature, as it smooths out the 40 column screen, making for a nicer looking GEOS screen.

Turn on 1351 mouse emulation. Wheels OS is pure joy to use with a mouse. Right-clicking on the mouse acts as a **double-click** under Wheels OS. Left-clicking on the mouse acts like a regular click instead. This feature saves me wear and tear on my fingers, as I don't have to double-click all the time under the Wheels OS environment.

Go to the Keyboard Settings menu entry to activate its dialog box. You will see three choices. I recommend you choose the '*Symbolic US*' setting, so you will have the familiar North American Commodore keyboard layout. If you want to use the actual keyboard layout as shown on your computer keyboard you are using now with VICE, choose the '*Positional*' setting. Choose the 'Positional' setting if you are not intimately familiar with the North American Commodore keyboard layout.



Set devices #8 and #9 as 1581 devices. Do not treat those devices as **1541/1571** drives unless there's a compelling reason to do so, usually a GEOS utility that simply won't work on a 1581 drive. Wheels OS is a graphical environment and requires tons of disk access and space. Only the 1581 drive will do, adequately meeting Wheels OS's requirements.



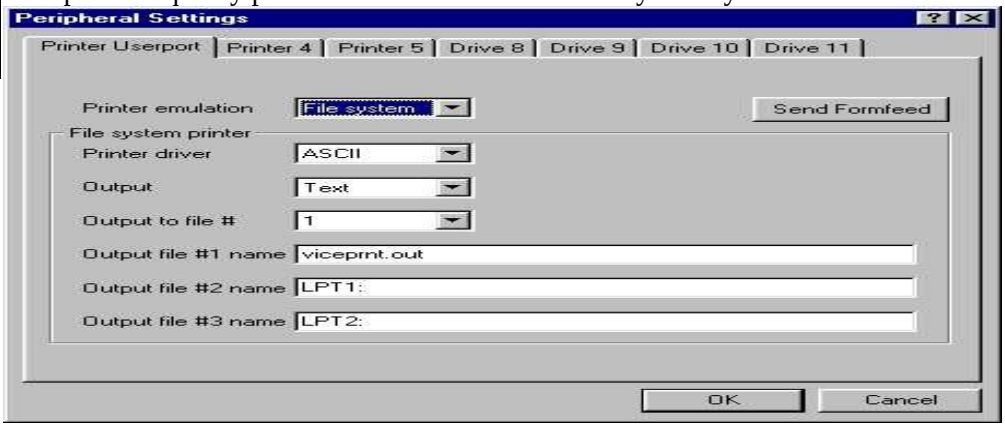
Recently, VICE allowed two more disk drives to be emulated; If you have a fast computer, go ahead and add a third 1581 as device #10, but leave device #11 **blank**.

VICE does have the ability to use Kernal and disk drive enhancements such as **JIFFYDOS**, but I will not go into this, as I doubt the legality of using them in an emulator. Turn on true drive emulation or Wheels OS *will crash*. There is a way to turn true drive emulation off while using the Wheels OS, and it will be described later.

As for **SID**, sound and music playback, you can turn this feature off in VICE. Very few GEOS programs will make use of sounds and the SID chip. Turning this feature off will improve VICE's emulation speed and quality. You can always turn on the sound playback anytime under VICE when you need to run a GEOS/Wheels program that uses the SID chip for sound/music. But, if you have a fast machine that can handle it, then go ahead and leave it on continuously.

When you click on the **Peripheral Settings** menu entry, a dialog box will pop up, allowing you to fine-tune printer settings. First, go to **Printer 4** and **Printer 5** tabs and uncheck *Use IEC Device* setting, to turn printer (serial device) emulation off. Go to the **Printer Userport** tab and select *file system* next to the printer emulation option. The defaults should appear as *ASCII* for the printer driver, *Text* for printer output, *1* for the output to file #, and *viceprnt.out* for the output file name. (Or 'viceprnt.ps' to alert the OS that this is a **PostScript** file.)

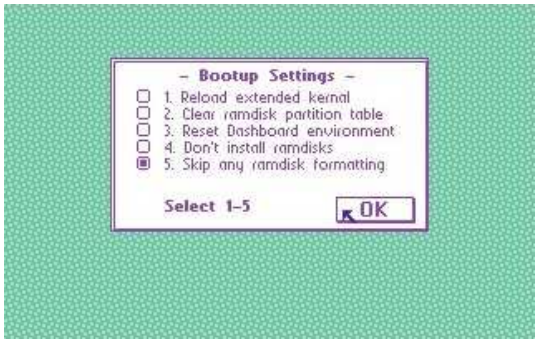
The settings will mimic the behavior of **geoCable**. When you print under Wheels OS to the user port, VICE will redirect the printer output to 'viceprnt.out'. Do use Wheels/GEOS PostScript capabilities as it will result in best possible quality printouts.



Save all the settings you have made so far under x64 and x128 emulators before proceeding! All the changes and settings you have saved will go to a text file called **vice.ini**. You can examine this file later and read the VICE documentation on how to fine-tune these settings to your liking.

On your Mark, Get Set, Fire up your set of Wheels!

If you haven't encountered initial bootup settings that the Wheels OS uses, press and hold the [**CTRL**] and [**SHIFT**] keys together when it is booting. This will force a dialog box to pop up with the following options: [*Refer to the following screenshot*]

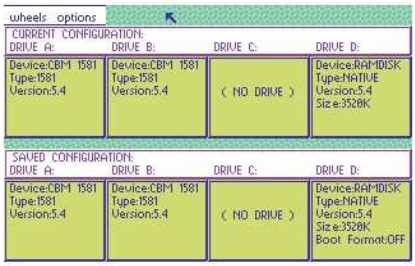


Use the screenshot in this example as a guide in selecting/ deselecting the options you need to configure Wheels OS for the VICE emulation program. This way, your Wheels OS running in VICE will treat the REU as a battery-backed device and act *nondestructively* towards it. Wheels OS will not have to reload its extended Kernal at each and every time it boots, shaving off some seconds in loading times. You can preserve your Dashboard session every time you use Wheels OS under VICE. Also, ramdisks will automagically appear each and every time you boot Wheels OS.

Next, use the **C1351D** mouse driver at Port #1 for using it under Wheels OS for VICE. You're pretty much half-way set now at this point for booting up Wheels OS. The next step is to enter the **Toolbox** and configure drives C & D for Wheels OS usage.

The VICE emulator has true drive emulation support for devices #8 thru #11. But, under the Wheels OS, you can add drives B, C & D by using a ramdisk. You have a 2Mb REU to configure and while it is tempting to configure a 2Mb native ramdisk and assign it as drive D, I'd advise against it. My strongest

Toolbox



recommendation is to simply create one 1581 ramdisk and assign it as drive D. The major reason is compatibility.

If you create a native ramdisk, how can you do a whole disk copy under the Wheels OS to a 1581 disk image? The advantage that a native ramdisk provides over a 1581 ramdisk is subdirectory support. However, if you need to copy the ramdisk contents back onto a 1581 disk image, you won't be able to copy subdirectories. Under a scenario where a native ramdisk is used and backup of these contents is needed onto a 1581 disk image, *file copying* under Wheels OS is required and is cumbersome.

A second reason for installing a single 1581 ramdisk in a 2Mb REU image under VICE for Wheels OS usage is that nearly 1,200Kb of free RAM is unused. Increasingly, Wheels programs like **dotView**, **geoWrite 128 v2.2**, **geoCanvas**, etc. will make use of extra RAM that the Wheels OS provides. If you fill the REU with a 2Mb native ramdisk, you will not be able to use any of these newer programs geared specifically for the Wheels OS. You will find yourself going back into the Toolbox and making a smaller native ramdisk and undo your REU configuration choices.

Let's do the Time Warp.

Finally, the Wheels OS is configured for VICE emulation use. There are a couple of advantages in using the VICE emulation suite as opposed to using the real machines for GEOS use. Let's take a look at one major advantage; speed gains via the **Warp** mode of x64 and x128 emulators. VICE has the ability to execute the 64 and 128 modes at 100% speed, meaning it will run at exactly 1MHz, even if you have a 3GHz Pentium under the hood. But, with Warp mode, VICE will run the entire emulation to the fastest possible speed, utilizing the speed of the host CPU like a 3GHz Pentium in this example. This could translate to an excess of 1000% or more in speed gains.

With x64 and x128 running at speeds in excess of 500% or more, Wheels OS is very responsive and a true pleasure to use. Files open rapidly, windowing occurs more naturally, screen redraws complete in the blink of an eye, scrolling occurs smoothly, and more. Also, drive D is a ramdisk and it does not require true drive emulation. If I need a speed boost, I turn off true drive emulation for drives A, B (& C, if necessary), *and run everything from drive D*. I run **Concept**, a Wheels OS assembler suite, in this fashion, from drive D, assembling and linking files entirely on the ramdisk, finishing its tasks in impressive speeds. However, it is still very important to turn true drive emulation back on for drives A, B & C, if you are going to use them again, or Wheels OS *will crash*.

With Warp mode, VICE will run the entire emulation to the fastest possible speed, utilizing the speed of the host CPU like a 3GHz Pentium in this example.

But, with such gains in speed, there are definite costs in using the Warp mode, unfortunately. The major impact areas are the keyboard and time-keeping. If x64/x128 is running at 800% of its original speed, and you are using geoWrite, you could be typing up your document like this: 'TTThhhhee QQQuuuiiiccckk BBrroowwnnn..." One solution is to use the **DetailShop** program to slow down the keyboard polling rate for Wheels OS to a suitable typing response level.

DetailShop

However, I find it preferable to simply disabling/enabling Warp mode by **toggling** the [**ALT-W**] key combination when needed. If I enter geoWrite 128, I hit [**ALT-W**] key combo to disable Warp and type away. If I need to scroll the document or go to a next page, I hit [**ALT-W**] key combo to enable Warp, and continue.

The other major problem relates to time-keeping. Let's say you set the Wheels OS clock to 8:00 p.m. and you enable Warp mode for around 30 minutes in using GEOS operations under the VICE emulator. By the time you are done, the time clock should read as 8:30 p.m., but it will never reveal the correct time in this case. Most often, depending on how fast the VICE emulation goes, the time clock will be at least 1-2 hours off! I have asked the VICE dev-team for **SmartMouse** support and they seemed receptive to the idea, but a solution to the time issue under Warp mode needs to be addressed first. It seems pointless to implement SmartMouse emulation into the x64/x128 emulators when the user would use the Warp mode and defeat its purpose.

A Modest Proposal.

There may be one possible and elegant solution to the Warp mode and make the x64/x128 emulators even more powerful in the process; I have asked for an **asynchronous CPU** mode for these emulators. This is how the **SuperCPU** operates; in asynchronous mode. The

SuperCPU would run independently of the host CBM computer and only synchronize whenever necessary. The same concept would apply to the 6510/8510 CPU emulation core under VICE. If asynchronous mode is enabled, the CPU could theoretically run at 1GHz+ speeds! With CPU asynchronous mode, only the CPU is sped up and the rest of the core emulation components such as the **CIA**, SID, VIC, disk drives, etc. are at their 1MHz mode or in synchronous operation a la Warp mode.

With the latest release of VICE v1.16, a new cartridge add-on has been emulated; the **IDE64** cartridge. Doing a cursory examination of the IDE64 emulation code, it appears to mimic the qualities of **DMA** access for the host CBM being emulated.

This means that quite possibly, a savvy programmer could adapt/reuse this IDE64 emulation code and add a '**CPU Cartridge**' level of emulation which would basically accomplish the asynchronous CPU mode as outlined previously. Over time, this 'CPU Cartridge' emulation concept, while initially supporting only the 6502 CPU, could be expanded to a full-blown SuperCPU Cartridge being emulated under VICE.

As for time-keeping problem under Warp Mode, perhaps a menu option in VICE could be created to reflect *GEOS mode*. If this mode is selected, the VICE emulator will assume that GEOS is used and that the CIA clocks need to be synchronized with the Windows/LINUX OS clock at every 3-5 seconds or so. An internal 'watch writes only' breakpoint can be put into force by VICE to monitor the TOD clocks and when these location(s) are written to, VICE will automatically insert the host OS clock values onto the TOD clocks, all without user intervention. With VICE forcing the issue of updating the CIA's at regular periods to reflect near-accurate time, users can use Warp mode to speed up their GEOS operations w/o sacrificing accurate time.

I would like to see some GEOS kernal commands be *trapped* under VICE's x64/x128 emulators. For example, **MoveData** & **EnterTurbo** GEOS Kernal routines could be trapped for a speed boost. These two routines are heavily used under the GEOS environment and having them trapped and executed at Pentium-class speeds would be very nice, indeed! Ninety-nine percent of the time, a high resolution screen is used in both VIC and VDC modes. The VICE emulators could turn off cycle-exact timing for the VIC chip in favor of simple VIC emulation (line-based emulation) for snappier graphics redrawing onscreen.

With this asynchronous solution, I could speed up the x64/x128 emulation and still be able to type normally. To date, the VICE development team has been very lukewarm to the idea of a CPU asynchronous mode or a CPU cartridge emulation concept. In fact, the x128 emulator runs at 1MHz mode; *There is no 2MHz mode at all*. Hopefully, these proposed addition(s) will be feasible.

GEOS/Wheels printing has never been so easy under the VICE emulator, PostPrint and GhostScript! geoWrite, geoPublish and geoPaint files can be printed in high (laser) quality.

The Power of PostScript Printing Comes Alive!

There are a lot of things you can do with your GEOS hobbyist endeavors under the VICE emulator. One major example would be printing under GEOS/Wheels OS. It can use PostScript printers and output files in PostScript for superior print quality. With Maurice Randall's **PostPrint** program, I am able to 'print' geoWrite, geoPaint and geoPublish files from my Wheels 128 environment under the x128 emulator. My print job will be redirected to a file and when I open the file in the Windows environment, I see PostScript commands and the like. I simply fire up **GhostScript** and actually see the printer output onscreen in near laser quality. From that point on, I can preview it, go back to x128 emulator and use Wheels 128 to make changes, etc., and then I print the file to my inkjet printer.

GEOS/Wheels printing has never been so easy under the VICE emulator in conjunction with GhostScript. I should note that it is very easily possible to do quality printing under Wheels OS on the real machines. A user would craft his/her masterpiece, and use PostPrint to print it out to a file onto disk media. Then this user could dial into a LINUX box and transfer the file there for later viewing, manipulation/printing under GhostScript.

Alternatively, the user could use **geoDOS** to save it on a MS-DOS 3.5" disk and transfer it to his/her Windows PC for later work under GhostScript. This way, a user need not to purchase an expensive laser printer and can use inkjet printers for GEOS/Wheels OS usage at economical prices w/o sacrificing quality. Using VICE to do GEOS/Wheels OS printing is a matter of convenience as a user can easily switch between the emulator and GhostScript on a Windows machine in creating superb printouts.

Secondly, we are working with 1581 partitions here. While that is plenty of room for small to medium sized geoPublish and PostPrint II projects, it becomes *limiting* when large projects are

factored in. For example, if you wanted to use tons of fonts & graphics for a huge project, you would be hard pressed to do it under VICE emulation. Goes to show you that using the real machines will still outshine their emulated counterparts.

Enter The Beast

Using the real machine is truly an escapist pleasure compared to the beast, a pseudonym for the modern computing platform whose complexities commonly vex and frustrate its users. However, the beast can also be pretty powerful to use. Using the VICE emulator suite brings convenience to the Commodore hobbyist.

Imagine having access to a C64 or a C128 emulation on your mainstream computer, all just a click away from activity. To replicate this setup, the same user would need to combine a C128 computer, a CBM monitor, three disk drives, a mouse, a userport printer interface, a cartridge expander, a stereo SID cartridge, and a REU. Some people simply do not have the desk real estate needed for this purpose. Under VICE, you could open x64 and x128 at the same time. It's like having two full CBM setups running simultaneously!

Running x128 will give you two full screens, the VIC and VDC screens. You can switch between the two and can even watch both at the same time. No more pressing the 40/80 column switch on your 1084 monitor. You can easily make screenshots by simply capturing the relevant VIC/VDC screen and save it as a graphics file for later handling. The screenshot function, while not complete, is very powerful. It captures sprites, background and border activity, character graphics, and bitmapped graphics with ease. You can even make **movies** of your emulator sessions with the **ffpeg** module! The only failing I could observe is that the screenshot function will not fully grab interlaced screens, currently capturing only one half of such screens.

The x64/x128 emulators are readily accessible. I usually browse the Internet for Commodore info and news. I usually come across an item, download it, and immediately view it under the VICE emulators. If it's interesting enough, I save it for viewing on the real Commodore later. (My viewing queue is high!)

I'm always working under geoWrite 128 and geoPublish when I'm under the x128 emulator running Wheels 128. I use the Internet a lot into my Commodore research, and I can easily switch between the x128 emulator and the Internet browser showing **Google**, for instance. I can duplicate this convenience by putting the CBM setup as closely as possible to the mainstream computer, and literally multitask myself between two computer setups, but some people do not have this kind of luxury.

Using the real machine is truly an escapist pleasure compared to the beast, a pseudonym for the modern computing platform whose complexities commonly vex and frustrate its users.

While the VICE emulators are convenient, I like them for their power it gives me in controlling the emulation environment of the 64 and 128 emulators. I talk specifically about its debugging and ML monitor capabilities. (Under LINUX, run x64 and x128 from the shell so you can have access to logging information and the ML monitor.)

Let's imagine a scenario where a programmer would watch the memory area of $a000-$bf7f for any writes under **geoPublish**. Why geoPublish? It's a 40 column only program and by monitoring the writes to the 40 column screen, a skilled programmer could capture these, examine the code and possibly fix it for 80 column duty instead. This is just an example of how powerful VICE really is, as a skilled person can really examine a program, fix it, and much more...

This is how I was able to spot some bugs in Wheels 64 kernal while debugging **geoZIP**. I was able to fix geoCanvas 64 and 128 for their Wheels OS usage. I fixed some GEOS games in this fashion. VICE gives you the following abilities at your disposal:

* Single-stepping through code
* Allows you to 'next' (execute) JSR's so you do not have to follow the subroutine's instructions
* Monitor read's, write's and execute's on any kind of memory location or I/O
* Examine & write memory locations
* Set conditional breakpoints
* Apply descriptive labels to aid disassembly
* Powerful CPU trace functions

And much more! In fact, many of the ML monitor's capabilities are available for disk drive debugging if one wanted to examine drive code and data transfer. With these kind of capabilities, a skilled programmer could defuse copy protection schemes with ease.

And the Beauty!

And therein lies a paradox: Maurice Randall originally designed the Wheels OS to overcome the limitations of CBM's stock equipment and fully take advantage of the latest 8-bit innovations coming out of CMD. All VICE can offer to the Commodore hobbyist is a stock CBM setup consisting of a C64/128 emulator, 15xx disk drives, 1351 mouse, userport, 17xx REU and stereo SID emulations. It might seem counterintuitive and counter-productive to run the Wheels OS in an environment which is so *underwhelming*.

Running Wheels 64 and 128 in combination with CMD devices is a thing of true beauty and testament to Maurice Randall's vision and programming skills. Using the Wheels OS with a full complement of CMD goodies is a far *superior* way of doing things for the Commodore enthusiast, as opposed to the VICE emulation alternative.

With the Wheels OS somewhat at odds with VICE, a Commodore enthusiast will simply adopt a pragmatic approach in using it under VICE's x64 and x128 emulators. Just know well that using Wheels OS under VICE means "*under-utilizing*" it, and that there are definite limitations.

That all said, VICE's stock configuration of a 17xx (2Mb) REU, three 1581 drives, 1351 mouse is a pleasing environment that adequately meets the power of the Wheels OS, resulting in a satisfactory computing endeavor for the Commodore hobbyist. Properly configured, the Wheels OS will shine through the emulated environment, all to benefit the Commodore GEOS diehard.

An Intellectual 800-Pound Gorilla

This phenomenon isn't isolated to VICE. I talk about intellectual property issues surrounding C64 and other CBM emulators. All of the emulator authors, to the best of my knowledge, have not secured the necessary permissions to distribute their respective emulators with ROM's. Compounding this problem is the fact that all relevant intellectual property holders apparently are not exercising their property rights against the emulator authors for potential infringement. A recent development, which could potentially alter the landscape for emulator authors, is **Tulip Computers** granting an exclusive license to **Ironstone** for the Commodore trademark.

Tulip also announced that Ironstone will create (or brand) an 'official' C64 emulator. Their press release implies that they have secured the necessary copyrights to have an official C64 emulator and that unofficial ones may be targeted for legal action.

Recently, **ClickGamer**, under license with **Ironstone**, released their **Windows** (32-bit) version of their **PocketPC** Commodore 64 emulator. This emulator appears to **lack** 1581 & 17xx REU support, both needed for an enjoyable Wheels OS experience. The URL for this official C64 emulator is: http://www.clickgamer.com/moreinfo.htm?pid=4

In the future, the VICE emulation suite may finally bear the **imprimatur** of an 'official' CBM emulator and be a legitimate torch bearer standard for the worldwide Commodore community.

That all said, VICE's stock configuration is a pleasing environment which meets the power of the Wheels OS, resulting in a satisfactory computing endeavor for the Commodore hobbyist.

Is It In Your Future?

Despite its now-cloudy future, the VICE emulation suite is still evolving in many ways and is far from being a finished product. The authors are now working at adding **Plus/4** emulation, for instance. But, for all intents and purposes, the C64/128 emulators are pretty much finished. Armed with the knowledge contained in this article, you should be able to enjoy using the Wheels OS in x64/x128 emulators and not be frustrated by the experience.

While the vast majority of CBM users would not consider running Wheels OS in an emulator, you may be pleasantly surprised as to how much you can do to further your Commodore hobbyist endeavors within a top-notch emulator like VICE's x64/x128. ✲

(*Continued From Page 2.*)

WheelWizard
geoWizDump
Original code by **Jim Collette**, and all code modifications by **Todd S. Elliott**.
GhostScript/GhostView
www.cs.wisc.edu/~ghost/
Laser Lovers
Wrong Is Write 81
Code by **Maurice Randall** and **Joe Buckley**, respectively. Promoted by **K. Dale Sidebottom**. If you want the finest in *Commodore GEOS printing*, check it out!
www.luckyclub.net
ConGo
Created and maintained by **Matthias Matting**.
www.editorix.org/congo/
GoDot
Created/Maintained by **Arndt Dettke**.
www.godot64.de
VICE
The ultimate C64/128 emulator.
www.viceteam.org
Star Commander
Maintained by **Joe Forster**. A cool interface for maintaining CBM/GEOS files on modern platforms.
sta.c64.org
CMDIMG
Maintained by **Todd S. Elliott**. A rudimentary interface for maintaining CMD DOS disk images archived by **DriveGhost**. Coded in **PERL**.
www.geocities.com/eyethian2000/cmdimg.htm
DriveGhost
64HDD
Maintained by **Nick Coplin**. The software combo, in conjunction with an **XE1541/Pwr** cable, can image, backup and restore CMD peripherals.
www.64hdd.com
PERL
A cross-platform programming language.
www.perl.com
CC65 Programming Suite
Maintained by **Ullrich Bassewitz**.
GEOS Support by **Maciej Witkowiak**.
www.cc65.org ✲

Wheels Wallpaper Wallery!

Ever dream of wallpapering your **Wheels**? That is, putting up a nice backdrop for your **Dashboard 128** (40 column) environment! As you can see from the included Photo Scraps in this issue, it is definitely possible, but has mixed results. This is just one example of how I used **VICE's** ML monitor and a hex editor to directly modify Dashboard 128 to allow it to display wallpapers.

The story involves VICE, the Versatile Commodore Emulation Suite and its powerful ML monitor. I set up a '*watch store $a000*' breakpoint and exited. VICE will watch any stores to $a000, which happens to be the start of the hi-res screen used by Wheels OS. First, there were some false alarms, as some routines were used to wipe the screen, to place the top menu bar, etc. I finally hit the jackpot when I noticed Dashboard 128's routines starting at $0ca1 were accessing the screen. First, it will clear videoram @ $8c00 with user's color choices. Then, it will clear the hi-res screen with user's pattern.

I created the ML routines (see sidebar for complete source code listing.) needed for Dashboard 128 in creating a wallpaper. Next, I set up a breakpoint at $0ca1 under VICE's ML monitor running Dashboard 128. When the ML monitor popped up, I typed '*load "wallery.bin" 0*' to load it into $0334. The neat trick involves the area between $0334-$03ff, which is largely unused in Commodore GEOS, and is a near-perfect area for testing out small, homebrewed routines under whatever Commodore GEOS application you're tinkering with.

Next, I type the following at the ML prompt: '*g 0334*'. It's that simple! For the astute Commodore diehard, the 'DD' part of the file name instantly refers to the graphics file as a **Doodle!** file. I suppose I could have used a Photo Scrap instead, but I wanted to mimimize the intrusiveness into pseudoregisters r0..r15 and I needed to conserve code space. Plus, bitmap routines in **GEOS** Kernal didn't support color, and more code is needed to make it work under Wheels Kernal. There is no provision for 80 column mode. Again, it can be done, but would require more code.

After finally getting the routines to work with Dashboard 128, now I needed to find a place to wedge it into that application. Unfortunately, it appears that Dashboard 128 will use the entire memory range and completely runs off from the REU. Considerably more disassembly work, memory mapping and layout would be needed in order to wedge the wallpaper routine into Dashboard 128. The area $0334-$03ff is not a good area for a permanent routine, as **geoPublish** will destroy it. Thus, I felt I needed to *cannibalize* some existing code space in Dashboard 128 just to allow it to have wallpaper abilities.

I looked hard at the **geoSHELL** menu option, but it only offered roughly 50 bytes, and I needed to get around 192 bytes. Finally, my gaze turned upon the icon bitmaps held in VLIR #$02 of Dashboard 128, starting at $6000-onwards. The truly neat thing about VICE's ML monitor is the 'ms' command. It shows sprite data as ASCII representations! Go ahead and type '*ms 6001*', '*ms 6041*', '*ms 6081*' and every 64 bytes thereafter to view all icon bitmaps as used by Dashboard 128. I finally settled on the area between $6280-$633f as a good choice for the wallpaper routine. Three icons, [REU/1541 ramdisk], [REU/1571 ramdisk], and [reu/1581 ramdisk] are overwritten. I'm using a 4Mb REU under winVICE for my Wheels 128 usage and have configured a 3.5Mb native ramdisk, so I really didn't need the REU/15xx ramdisk icons.

I used a hex editor and loaded in Dashboard 128. I found the icon space at offset $4aee and 'pasted' in my 'wallery.bin' addition, making it a permanent part of Dashboard 128. I also redirected the JSR $5007 at $0ca1 to JMP $6281 to point the initial color/pattern screen wipe routine to the custom wallpaper routines. Again, when making the changes in the hex editor, I used its search function to find the hex string, 20 07 50 20 01 50 and was able to find and make the changes instantly.

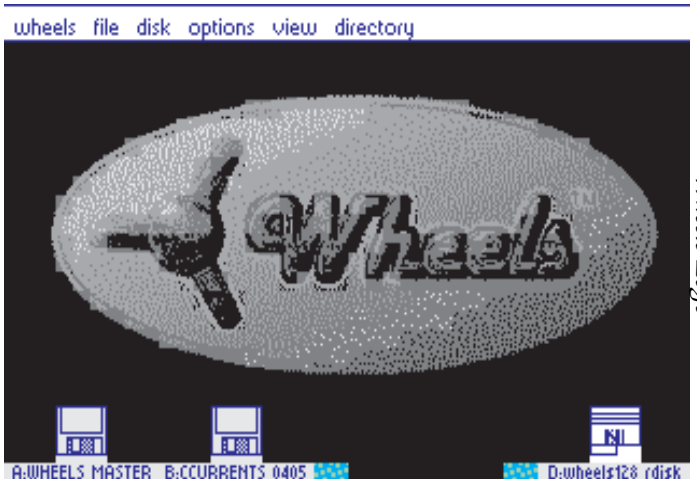
However, this interesting sojourn into Dashboard 128 has brought forth mixed results. Whenever you move disk icons around, they will overwrite the bitmap. Hence, I put the drive icons at the bottom of the screen. Since **ReadByte** is used with **diskBlkBuf** to read in the Doodle! file, Dashboard 128 may not refresh its file window display. For example, I double-click on drive D: and a file window pops up, listing various files. I fire up **geoPaint**, and exit. When exiting, Dashboard 128 will execute the wallpaper routine and then display this file window, but it's empty! I close the file window and double-click on drive D: again to get the file window and all files are displayed intact. Whew. :)

In creating the various wallpapers, I used mainstream PC software to reduce most images to 320x200 size, and used **ConGo** to create resulting 4-bit files. Next, I used **GoDot** to load them in, tweaked the color settings, and saved them as 'Doodle!'s'. These Doodle's are renamed as DDWALLPAPER as necessary. Next, I used **WheelWizard** & **geoWizDump** to create screenshots of these wallpapers and saved them as Photo Scraps. Next, they were pasted into this geoPublish document. Thanks to the wizardry of **PostPrint II/III**, the Photo Scraps came out in *full color*!

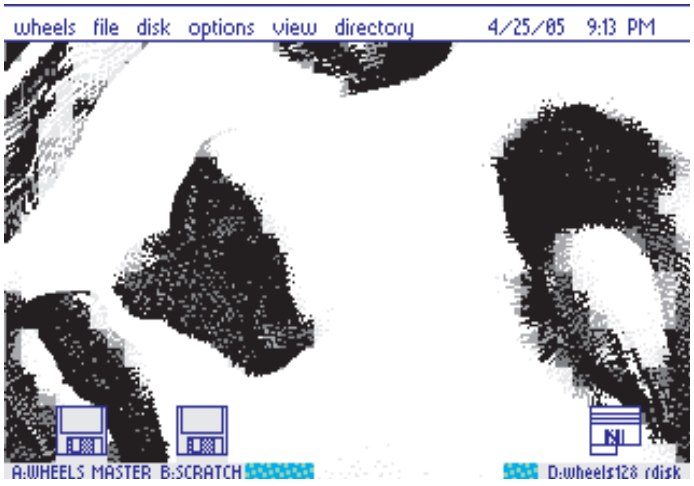
Now, I'm not suggesting that you try wallpapering your Wheels 128 40 column screen under VICE. I don't know for sure if the $6280-$633f is truly a safe area, and that since disk routines (i.e., ReadByte) are used, disk buffers could be overwritten, affecting Dashboard 128's performance. In fact, when I tried to switch 40/80 modes via the [ALT] key, Dashboard 128 overwrote the $6280 area and caused a crash. This is only just to illustrate how you can truly do powerful things with VICE's ML monitor in enhancing your Commodore GEOS enjoyment. Someday, perhaps **Maurice Randall** will finally integrate wallpaper's into a future version of Wheels OS that works in any video and/or computer modes and is in full color.

In the meantime, enjoy the Photo Scraps in this Wheels Wallpaper Wallery! (I wanted a catchy and alliterative title, hence 'Wallery', not 'Gallery' was used.) May they whet your appetite for all things Commodore GEOS and... Wheels OS! ✲

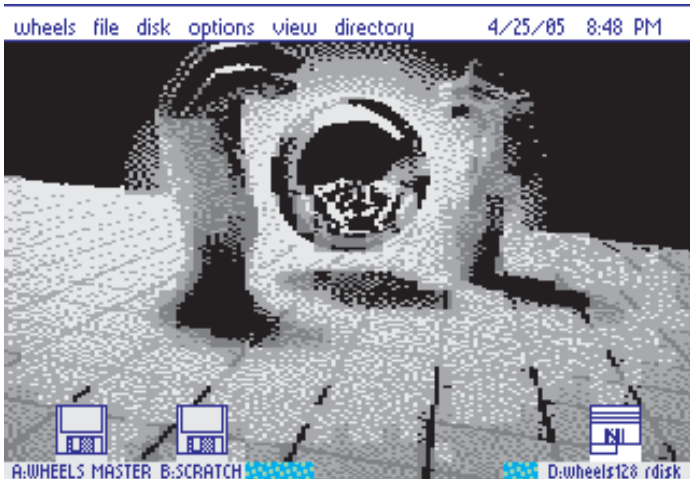
```
; Dashboard 128 WallPaper Add-On
; Copyright (c) 2005 by Todd S. Elliott
.org $6280; $6280-$633f (VLIR #$02)
.byte $bf; icon space in Dashboard 128
WedgeWallpaper: ; start here.
      bit   graphMode     ; check
      bmi   @4            ; 80 cols.
      lda   #$08          ; initial
      sta   r0H           ; dev #.
@2:   jsr   SetDevice
      txa   ; check error
      bne   @3
      LoadW r6, fname
      jsr   FindFile
      txa   ; check error
      beq   @1; find the file?
@3:   inc   r0H; go to next dev #
      lda   r0H
      cmp   #$0c; are we done?
      bne   @2; and repeat the search
      ; for "DDWALLPAPER"
@5:   jsr   $16e2; calls SetDevice if
      ; wallpaper isn't found.
@4:   jsr   $5807; and paints the 40/80 column screen.
      jmp   $0ca4

@1:   lda   dirEntryBuf
      cmp   #$82          ; is it C= PRG filetype?
      bne   @5            ; branch if it isn't.
      lda   #>diskBlkBuf
      sta   r4H
      lda   #$00
      sta   r4L
      sta   r5L
      ldx   #$bf          ; align icon byte - $bf is GEOS compacted bitmap
      sta   r5H           ; datatype every 64 bytes
      lda   dirEntryBuf+1
      sta   r1L
      lda   dirEntryBuf+2
      sta   r1H           ; set t/s
      jsr   ReadByte
      jsr   ReadByte      ; get rid of load address
      LoadW r0, $8c00     ; do video matrix memory
      LoadW r6, $03e8     ; do only 1000 read's
@6:   jsr   ReadByte      ; get video value
      ldy   #$00
      sta   (r0),y
      inc   r0L
      bne   @7
      inc   r0H
@7:   ldx   #r6
      jsr   Ddec          ; go through video matrix memory
      bne   @6
      lda   #$18          ; get rid of superflous data
      sta   r6L
@8:   jsr   ReadByte      ; and discard
      ldx   $bf00         ; align icon byte - $bf is GEOS compacted bitmap
      ldx   #r6           ; datatype every 64 bytes.
      jsr   Ddec
      bne   @8
      LoadW r6, $1f40     ; go through bitmap data
      LoadW r0, $a000
@9:   jsr   ReadByte
      ldy   #$00
      sta   (r0),y
      inc   r0L
      bne   @0
      inc   r0H
@0:   ldx   #r6
      jsr   Ddec          ; go through bitmap data
      bne   @9
      jmp   $16e2         ; calls SetDevice and exits the wallpapering routine.
fname:
.byte "DDWALLPAPER", 0
```

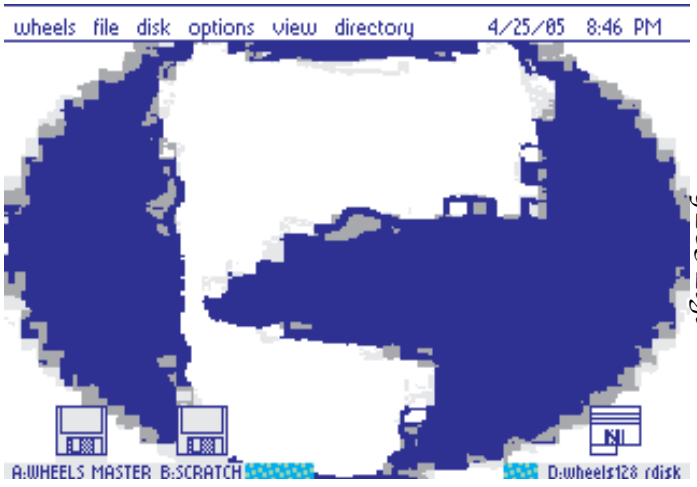


TimeMaster Wheels Logo

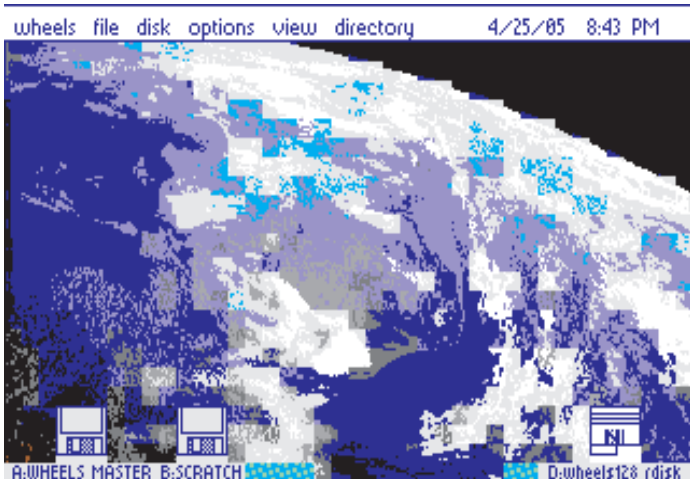


Pinball Construction Set Spacewalk





Earth From Space GEOS Logo



C=Currents Eyes