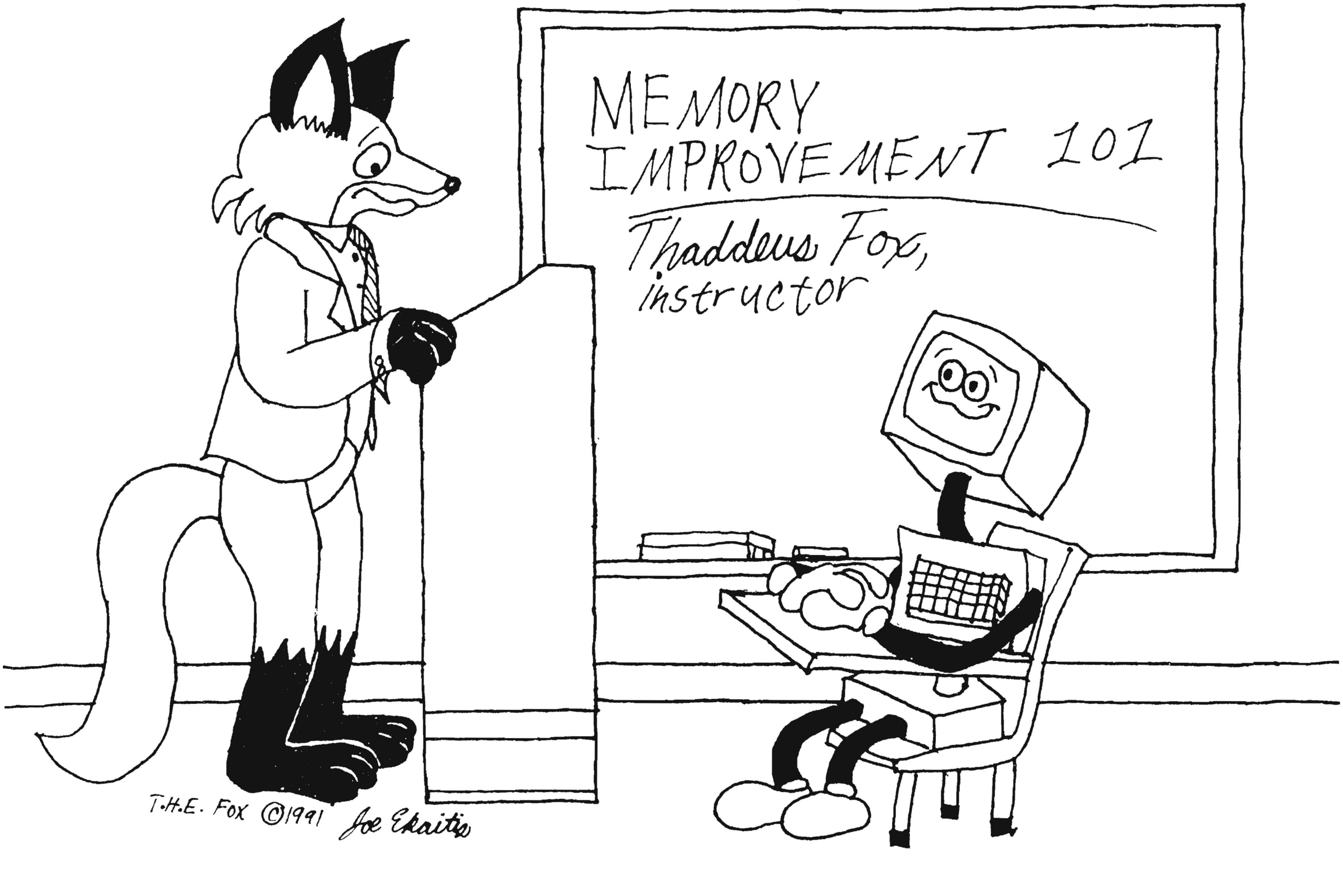
Twin Cities 128 the COMMODORE C-128 Computer Journal



Issue #30
$3.95 U.S.
$4.95 Canada

Parsec, Inc.

POB 111 Salem, MA 01970-0111
U.S.A.

Table of Contents:

Editorial

Hardware

Software

General Interest

Customer Order and Support Number 1-508-745-9125 9:00 - 5:00 EST Monday - Friday
This number is not for subscription changes. These still must be sent by U.S. mail.
COD orders are $4.00 extra and are only accepted from established customers
or current TC128 subscribers.

We now own and produce Compendium Book #1 (the best of issues 1-18) $16.95 and

C-128 500 Q&A Book (all you wanted to know but were afraid to ask about your C-128) $16.95

This magazine is dedicated to my mother to whom I greatfully owe all my successes and failures.

Changing of the guard

Editorial by JBEE

For many of you reading this column, our taking over this magazine is your first contact with Parsec, Inc. I suppose a "who are we" explanation is in order. My name is John Brown, the editor and president of Parsec, Inc. Though while I hide in the shadows of some hulking BBS system I go by the moniker "JBEE". Parsec, Inc. is a small corporation consisting now of two full time people and two part time people. We were founded in 1986 because of the lack of Commodore software and support in my area and I felt I was not getting the full value from my dollar from mail order firms. I thought I could do better, for the most part I have. This company was also founded because I dreamed that I could make video games instead of playing them. Someday I would like to make that dream a reality (I have been at it long enough). We progressed slowly but surely from selling public domain software, to selling commercial software and hardware, to publishing this magazine. During this time we have provided service and support to Commodore users.

In many ways this is both a very sad occasion and very happy occasion. It is sad in one way because it marks the failure of Voyager Mindtools Inc. with Loren handing over the controls of Twin Cities 128 to Parsec, Inc. As I write this editorial Voyager Mindtools, Inc. is dead and out of business and Loren has probably sold everything he owns but a single C-128 and has vowed never to have anything to do with again with computers. I know it would be very easy to lambaste him for all the things he hasn't done or for all the things he HAS done in the last few months. But, instead of that, I think I would rather remember all the things Loren did accomplish when no one else would give C-128 owners the time of day. Through him TC128 supplied us with both the tools and knowledge to better utilize our C-128s. Some of the things covered between the front and back pages have helped produce some of the best software we are using today. I have always felt that Transactor was the best magazine Commodore magazine *I ever read* with TC128 trailing in a solid second place. It was not unusual for either magazine to have single issues so helpful or good that one issue was worth the purchase price of the whole subscription.

How I feel about VMI and Loren, he made some bad mistakes, fell off the path and did not know how to get back up. Ever see that ad for that medical help line with the old lady "I've fallen and I can't get up". Well, put Loren there and realize how frustrated he must have been because no matter what he did he could not get up. I myself have fallen a few times because of some very bad errors in judgement. I was able to get up, shake my head a few times, brush myself off, get reorientated, and go. Loren wasn't able to. I do think VMI and Loren failed because of bad errors in judgment and mismanagement of our subscription money (that is right, until you receive all your issues it is YOUR money, no matter what info tells you). Along with a bit of luck, success in todays market means knowing how to market, how to handle money, how to bring the product to market on time, and most importantly providing timely and effective service to your customers. That last one is both the most crucial and important of them all. It is also the most elusive and hardest to reach objective. I know from time to time it has been for us.

There have been some hard feelings towards Loren, for some of his posts, and his handling of orders and inquires. Maybe you should put some of those hard feelings aside like I did. Put yourself in Loren's shoes for a moment and imagine how you would feel if you lost the magazine you founded. A magazine into which you put both yourself, your time, and your money for over four years. I do not think you would judge him too harshly if you were in his shoes. I would rather have someone try and fail then never to try. I would rather remember his pass work and achievements. Without the past rewarding issues and accomplishments of TC128 I believe the C-128 market would have been greatly lessened and I know I would have missed countless enjoyable hours using my C-128. That is how I prefer for Loren to be judged, on the whole and not just for a few bad months. I salute him, wish him luck, and hope I can produce the same high quality issues into the 90s for you worthy of the TC128 heritage.

The TC128 changing of hands also marked a very sad moment in my life because it marked the death of my mother this July. She was my best friend, tutor, anchor, and the other half of Parsec. Though we both have helped and supported each other in various ways over the years, it was her belief in me that help bring Parsec from a dream and wish to reality over the past five years. It was a learning experience for both of us. She was very happy and proud to finally see me own my own magazine before she died. But I can't help but feel cheated because I can no longer share any of my future successes and failures with her. So, this is why I am dedicating this magazine to her.

How did this takeover come about? VMI/Loren put a message up on Q-link that they were looking for someone to buy TC128 as they were broke and would not be able to produce any more issues. When we contacted VMI we were only one week away from getting our new catalog printed and buying a mailing list. I had also put out "feelers" about writing a C-64 and C-128 newsletter for the "experienced" user that would be a monthly publication. I had no plans for something on this scale and expense. After talking and cementing the deal over the phone with Loren I decided to delay everything I had planned for the next nine months and put everything I had into TC128.

Why did Loren finally decide to sell it to Parsec instead of someone else? Loren said the main reason was he wanted someone to take over where he left off and that he would rather kill the magazine completely than give it to someone who would do a poor job. After all, it is his "baby" and he was put a lot of thought and pride into the magazine. Loren also thought I had what it would take to pull it off. It is a big investment, I figure VMI owed at least 18,000-22,000 issues to thousands of subscribers. Add this on top of the cost of producing the issues due, paying authors, and mailing costs and you have a pretty steep tab.

Why did Parsec take over the magazine? Some of the reasons were I thought the death of this magazine would be the kiss of death for anyone else in the C-128 market that introduced a new product or in my case a new magazine. Especially after Transactor, Ahoy, Commodore, info, and others magazines left tens of thousands of subscribers beached high and dry with their subscriptions. I plan on making my living the next few years selling to C-128 owners. I also felt that if I came out with a competing newsletter after TC128's posted death it would make it that much harder for Loren to sell TC128. But, one of the biggest reasons for buying it was also the most important reason for Loren when selling it to me, I couldn't stand the thought of someone taking it over and doing a poor or slap happy job. I am also working on a project (the accelerator board for the C-128) that needs your trust to see the light of day and a vehicle by which to reach the most interested people. By taking over the operation and publication of TC128 and honoring all those due subscriptions I hope to instill a sense of trust between you the reader and Parsec, Inc.

What is the game plan for the next year? Right now getting TC128 on a regular publishing schedule, getting it on Commodore dealer shelves, mass market shelves, local book stores racks, and software store shelves. Also, a monthly publishing schedule.

Part of this crusade is bringing into the spotlight software and hardware for C-128 owners that you might never have heard about simply because people do not know how to go about getting the limelight. Or for those products with a very limited market that no one cares to touch but are products worthy of notice, however limited.

I wish to educate and encourage people to produce new software and hardware by giving them easy access to both knowledge, Parsec, and a market. It has taken me a few years to realize I can not conquer and shape the whole world and I can not do everything myself. The best way to insure I have new hardware and software for you and my machine into the 90s is by helping and encouraging other people to produce it. Does this mean slanted or less than honest reviews? NO WAY. I priced this magazine when I took it over so I

1.) could pay the authors and myself a small but decent fee for our work,

2.) cover the printing costs,

3.) cover the mailing costs,

4.) put a little away towards future expansion once all the due subscriptions are honored.

Any advertising is just icing on the cake and I will not slant any review for monetary gain. One thing you can expect though is my own biased opinion as a diehard C-128 user with experience using all sorts of software and hardware brands. I refuse to accept average as being good enough.

Okay, so what will TC128 provide over the next year? How about a warranty, guarantee, and hopefully at least six timely issues? If for some reason we do not produce six issues for your $20 and you want a refund what happens? You get it back. That's right! No matter what info tells C-64 and C-128 owners (who feel that they were shafted), when you send in a subscription it is a contract to provide a service and product. Not to do so is a breach of contract in most, if not all, states. So, until you meet those provisions you still owe the customer either the money or the product. Not to do so and not to provide a refund would mean being a spineless worm. Hey, I am not fish bait ;) I like to hold up my head up high and would work 100 hours a week scrubbing toilets if that was the only way to pay back my debts.

What will your money buy for the next six to twelve issues? Increased coverage with a Geos and Geos programming column, how to use existing C-128 applications for non programmers, reviews of everything C-128 we hunt down including hardware projects, a free For Sale/Wanted section, a C programming column, how to modify and use your word processors with the newest and best in printers, reviews of general interest items that you can use with your C-128. Don't worry, you will not find the 1541ii or 1902 covered here twice a year or ads for other computers on our back page. We are also willing to cover any other topic which we think might interest you. Do not worry, there will still be programs and articles for the beginner, in fact issue #31 and #32 will be kicking off a tutorial on a couple of subjects designed to take people from the ground floor up to the top. I will not say what they are but I

(continued on page 25)

Looking back into the future

by: Loren Lovhaug

Entry Initiated: Thursday, July 5, 2021 21:46
Personal Journal, Graham Lovhaug

I had the day off following the fourth of July holiday, so on a whim my son and I decided to visit my dad, who lives in Minneapolis. We boarded the 8:30 Mag-Lev from Chicago to Minneapolis and arrived at Dad's house at about 10:30. As usual mom and dad weren't home, despite the fact that I had called ahead of time. Neither of them were answering their personal phones, so I surmised that my parents had left them in the car again. Fortunately, Basil identified me and let me into the house. It informed me that my parents were visiting friends and expected to be home by noon. I walked over to Basil's main display, recorded a short message and instructed the system to continue to try to reach my parents.

In the meantime I took Michael downstairs to the storage room. For month's Dad had been begging me to clean out some of my old stuff. Michael immediately seized this opportunity to start exploring Dad's old computer equipment. I could hardly blame him. When I was Michael's age, a day did not pass when I did not beg to go upstairs with my Dad so I could play a video game or play with a graphics package. Of course Michael's video book was many times more powerful than anything he would find in the basement, but that fact did not seem to dampen his curiosity.

As soon as we reached the bottom stair Michael darted over to the corner storage lockers his grandfather had showed him last time we visited. He immediately opened the lockers and began his quest for yesterday's technology. His eyes danced over the shelves littered with old style video monitors, keyboards, circuit boards, and the numerous other museum pieces my Dad had acquired over the years. When I came over to see what he was into, Michael was holding one of the machines. For several moments he turned the machine about studying its intricacy. Finally he looked up at me and asked, "Do you think it still works?" I replied somewhat hopefully, "I don't know, but we can try if all the pieces are here." Michael's face lit up as the question shot out of his mouth, "Dad! YOU know how THIS works?" I could not help but laugh. After all, in Michael's eyes this was real horse and buggy stuff, but I remembered fondly the days and nights I spent basking in the glow of that Commodore 128's screen.

I dug around in the storage locker and found the unit's power supply, and one old style floppy disk drive with its power cord. Obviously, since Michael had never seen a floppy disk, he was totally baffled by the drive's purpose. After picking up the disk drive and staring into its thin slot for quite a while before commenting innocently, "Dad, this slot doesn't seem wide enough for a piece of paper." I smiled and replied, "That's right, Mike, but it is wide enough for a five and a quarter inch floppy disk, that is if we can find one somewhere among this pile of memorabilia." Mike frowned. He seemed somewhat disappointed at the prospect of not finding the missing mystic item and my lack of interest. Sensing my affront, I returned to the locker in search of a disk. Michael joined the quest. While we hunted for the "thin black squares with the hole in the middle" I explained how they were used. I also told Michael about the special role my Dad had played in the development and usage of this particular early personal computer. Michael seemed glued on every word as I went on to explain how I inherited this Commodore 128.

I told him how I used it for playing games, writing school papers, telecommunicating, and putting together little programming projects. He seemed particularly intrigued as I detailed my programming ventures. Despite the fact that the BASIC 7.0 programs I created as a child on the 128 bore little resemblance to the "idea-link" applications Michael was familiar with, he seemed able to appreciate my youthful fascination with controlling the computer. It seems that three decades of geometric growth in processing power and mass storage did not remove one of the primary attractions of computers, namely their ability to act as the perfect slave. Somewhere deep in the human psyche is the need to be the master of something, to be in control. Computers are the perfect outlet for this need because they are literally obedient and within the bounds of morality. While in the past this need for domination has driven humanity into committing horrible acts of destruction and cruelty, conquering the machine may curb our desire to enslave our families and foreign populations, while enhancing our creativity and productivity.

Finally after searching nearly every shelf and emptying most of the boxes Michael found a plastic box of old floppy disks hand labeled "Commodore 128 software". Next to this disk box Michael also discovered a book entitled: "The Commodore 128 System Guide". Now we had just about everything we needed to attempt breathing life into this extinct digital breed, so I scooped up the antique components and brought them over to the Basil's basement terminal. Pressing the activate key caused the picture-window sized display hanging on the wall to flash to life. In the upper right corner was the familiar system status display, featuring the obligatory current time, interior and exterior temperature, active task summary and a notification that the entire library of Star Trek: The Next Generation Feature Films, all eleven of them, had been downloaded per my mother's request.

I pressed the new-task key, and Basil responded by uttering the usual model 1270 verbiage, "New task, requested at eleven twenty seven a.m., audio, visual, tactile, and digital input ready." For a moment I stood silently trying to compile in my head the list of the terse instructions which I hoped would allow me to set up a window on the wall display which would emulate a Commodore 128 display. Michael looked puzzled as I barked out, "configure...display...new window...video input...port 4." Now I was stuck. After the requisite ten second pause, Basil chimed back, "Please select your video input port specification from the choices displayed in window 9 or describe a new specification." I studied the pictures and descriptions of the forty seven connectors displayed in window 9. Four of the choices featured the nine pin squarish video connector which matched the connector on the back of the 128. While examining the text describing the possible candidates one stood out as familiar. "Item CGA: Popular old style microcomputer video output standard circa 1985 consisting of separated signals for red, green, blue, and intensity, sometimes referred to as digital RGB", seemed to strike a chord in my head. So I selected the item, and the universal interface adaptor immediately created an array of connections which could be mated to the back of the 128.

I pressed the 128 into the universal adaptor port and then I connected the power supply and the disk drives into their respective ports. Finally the moment of truth had arrived. I let Michael do the honors. He flicked the switch, and the red power light on the left side of the C-128 glowed brightly. On the wall display appeared a tiny black window only a few centimeters wide. At the top the small window appeared to be a set of almost inperceptible blue characters. Immediately I realized that Basil had chosen a 1:1 ratio of its pixels to the 128's pixels. The 128,000 dots which formed the 128's display were dwarfed by the over five billion pixels which surrounded the margins of the tiny display shown on the wall. I ordered Basil to scale the display by a factor of ten and the old Commodore copyright immediately became visible. Michael's face lit up, and we spent the next hour exploring the box of disks my father had stored away.

After my parents came home, my Dad joined in the fun. I was shocked at how familiar Dad's fingers seemed at the 36 year old keyboard. So I left my son in his able company and went upstairs to visit with mom. Five hours later they were still at it. The room was filled with discussions of MOVSPR this, DO..WHILE that, until finally we had to leave for home.

As we disembarked from the mag-lev in Chicago and headed back towards the car, my arms wrapped uncomfortably around a large box of my Dad's things, I looked down at Michael and laughed. I could not help it...he looked just like that silly picture in the 128 system guide entitled,

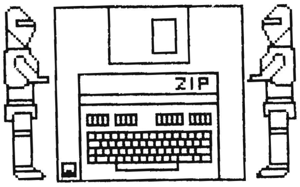
"A Commodore 128 and Student Headed to Class"

Expanding The C128
by Richard Curcio

Expanding the C-128 to 256K

(Part one: Banks 2 and 3)

Copyright 1990,1991 by Richard Curcio
and Parsec, Inc.



Editor's note: Before attempting this project you may want to wait until issue number thirty-one for the additional articles to upgrade your C-128 to 512K !

Perhaps my disdain for "memory gluttons" is a hold-over from my days with the Vic-20 and (gasp!) Timex-Sinclair. However, when I saw that the designers of the C128 had allowed for the possibility of that machine having RAM BANKs 2 and 3, even though the Memory Management Unit (MMU) and other hardware make no such provision, I couldn't resist thinking about ways to make that possibility a reality.

I set down certain specifications for this project:

1) No additional I/O registers should be required to select the new banks. Since the operating system already "thinks" it has BANKs 2 and 3, they should be accessible through existing MMU registers using the normal Kernal and Basic commands.

2) The circuitry should be as simple as possible. It would not be worthwhile if a dozen or more ICs (not counting DRAMs) were needed to merely double the amount of memory.

3) Minimal changes should be made to the existing C128 circuitry, including retaining the existing 128K of RAM.

I feel I have met all these specifications. However, because of dimensional constraints in the low-profile 128, item two should be amended to "...as simple as possible ELECTRONICALLY." It is not so simple to make everything fit into the flat case. On the other hand, the 128D has ample room and the wiring of the extra DRAMs in that machine is much simpler.

This article assumes you are familiar with proper soldering techniques and have some knowledge of electronics and expertise in the building and modification of same. If the foregoing does not describe you, it's best to leave this project to someone who does have those qualifications.

The heart of this modification is the installation of a second Commodore 8722 MMU. It is suggested that before beginning this modification, if possible, you test your extra MMU for a few days by removing the old one and plugging the new one into the MMU socket. Removing such large ICs often results in many pins becoming bent. Attempting to straighten damaged pins can cause them to break. I have found a right-angle screwdriver useful in chip removal. The trick is to pry the chip up a little at a time, first at one end then the other. Don't rush it. It is further suggested that you read these instructions thoroughly before you begin. Do not skip information just because it does not apply to your model of the C-128.

A list of the other parts required for this project, and possible sources for the MMU, are given at the end of this article. It is possible to arrange things so that this modification can later be up-graded to 512K.

Bear in mind that presently no available software makes use of the currently non-existent BANKs 2 and 3. Well written programs, such as the Power Assembler (Buddy) can easily access the new banks as it is written. At least for now any other software will have to be modified or written by users who perform this modification. As it will be shown, the new banks can even be utilized in Basic.

Disclaimer

This modification will render any warranties on your equipment null and void. Neither Parsec, Inc. or the Author assume any liability for any reader's or purchaser's implementation of these instructions. All information is believed to be accurate.

Theory

Run this short program

```
10 FOR I= 0 TO 3:BANK I :POKE49152,I :NEXT
20 FOR I= 0 TO 3:BANK I :PRINTPEEK(49152), :NEXT
```

on a stock, un-modified C128 and you will get:

```
2  3  2  3
```

because when it POKEs and PEEKs Banks 2 and 3 it is really accessing Banks 0 and 1. And, since the POKEs to Banks 2 and 3 replace the values in Banks 0 and 1, the values PEEKed from Banks 0 and 1 are the same as 2 and 3.

Once the modification described herein is completed, the result of running the above lines will be:

```
 0   1   2   3
```

indicating that Banks 2 and 3 are real. They can be accessed with the approriate BANK statement and the VIC can display them by altering bits 6 and 7 of the RAM Configuration Register at $D506.

Refer to figure 1. Above the dotted line is the relevant portion of the C128 circuitry. Pin numbers in parentheses are for the 128D. Below the dotted line is the added circuitry. (The additional DRAMs are not shown.) The original MMU (MMU-1,) generates one of two Column Address Strobes, depending upon the currently selected Bank, the range of memory being addressed, and several other factors, determined by the contents of the MMU's internal registers. The MMU also sends several signals to the PLA, telling it to generate ROM selects, I/O or /CASENB, which selects RAM. The VIC chip provides a "master" /CAS. Note that when the MMU registers at $FF00-$FF04 are being addressed, both /CASs from the MMU remain at logic 1 (inactive). That is how those registers are made to appear across all configurations.

Once the additional circuitry is installed, the original /CASs from MMU-1, combined with /CAS0 from the second MMU (MMU-2,) form a 3-bit "CAS code". This 3-bit code is applied to the A, B, and C inputs of IC Z2, a 74LS138 1-of-8 decoder. We get four 64K RAM blocks because four of the codes are invalid, so four of Z2's outputs are unused. ICs Z3 and Z4 detect and condition a number of signals, and let the two MMUs work together without interfering with each other. Adding a second MMU is much simpler than attempting to reconstruct the required logic with separate ICs.

There are two limitations to this arrangement: None of the additional memory is available in 64 mode, (For this modification, at least) and the MMUs cannot relocate zero-page and the stack to the new blocks. I believe that, with more logic, this last should be possible, but I don't think it's worth it. When common memory is at the low end, as is usually the case, the block pointers have no effect, and zp and stk are always in RAM 0. When common memory is only at the high end, or switched out altogether, each bank has its own zp and stk. Since zero page and stack relocation is a tricky feature under normal circumstances, I've chosen to keep the hardware simple and accept the block 0 and 1 limitations.

For the more technical: The 128's designers used a 74F32 for U9 because they were concerned about the speed with which the various signals occur. (The "F" stands for Fast.) The MMU takes time to determine which (if any,) /CAS to generate. The PLA, based on the signals it receives from the MMU and elsewhere, takes time to generate its signals. Although this amounts to a few dozen nanoseconds (billionths), /RAMCASn must occur within a rather brief "window". In figure 1, Z2 replaces the actions of U9 and the delays through Z2 are not much greater than those through two 74F OR gates. Adding decoders and such before or after U9 would have increased the delays and would not have worked. (However, a similar strategy DOES work when expanding a RAM Expansion Unit, but that is another story, and will not be covered here.)

As for expanded system RAM itself, using eight 256K x1 DRAMs would have required the removal of the existing DRAMs, something I was unwilling to do. And, since the MMU selects RAM in 64K chunks, it also would have meant somehow translating the /CASs from the MMUs into extra address bits and multiplexing them into a single bit. This is not feasible, because by the time the MMUs figure out which /CAS to generate, /RAS has already occurred and by then it's too late for the new address bit to have had a valid ROW value. For the same reason, two 256K x4 DRAMs could not be used, either. The way the VIC chip refreshes memory further precludes the use of 256K x4s.

Disassembly

When your modification is completed, to test it, you will need your power supply, disk drive and monitor. Remove your printer and all accessories if they occupy too much space in the area you will be working in. Protect any finished surfaces with layers of old newspaper.

KEEP YOUR DISKS FAR AWAY FROM YOUR SOLDERING IRON!

Low-profile C128 disassembly is as follows: Begin by disconnecting all peripherals and the power supply from your computer. Remove six screws from the bottom of the case, noting the different size (a "torx" driver may be needed for some C-128 computers). Partially lift the top cover and remove the three wire LED power indicator on the left side (when you reconnect it. the polarity won't matter). At this point, you might want to discharge any static charges from yourself by touching a cold water or radiator pipe. Unscrew the keyboard grounding strap on the right side of the case. If you have enough room to work in, you may keep your keyboard connected to the main board and attached to the top cover. If not, carefully remove the keyboard connector. Do this by gently rocking it from side to side while maintaining an upward pull. This connector is "keyed" and will go back on only one way. If any of the pins bend, you can gently bend them back to the vertical position. If any pins break, you will lose use of your keyboard. When the time comes to test your modification, I do not recommend that you repeatedly connect and disconnect the keyboard. Therefore, you might want to remove it from the top cover. Don't lose any of the little plastic "braces." If you think it will help during re-assembly, make a little sketch of how everything fits together before you start removing screws.

Remove the screws holding the RF shield in place, again noting any difference in sizes. The metal shield is soldered to the system board at one point. This will have to be unsoldered, using a solder vacuum and/or solder braid. Gently unbend the small metal tabs holding the top and bottom metal covers together and expose your computer's innards. The white gooey stuff on the larger chips and the underside of the top metal cover is heat-sink compound. Be careful not to get any on clothing, carpets, etc. You will not have to remove the circuit board from the bottom case.

To disassemble the 128D: Begin by disconnecting all peripherals and the power supply cord from the outlet and from your computer. Remove the six screws from the cabinet top by removing two screws from the bottom front and three from the rear. Slide the top back 1/2 inch and lift. Label all connectors so you can put them back from where they came. Make note of polarities! (Again, a sketch of how things WERE before disassembly may be helpful.) To remove the disk drive, disconnect connectors CN12, CN14, CN15 and CN17. Pull the lever knob from the front of the drive. Remove one screw from the left and two from the right sides of the drive. Slide back and remove. To remove the power supply, remove the screw holding the LED to the front panel. Disconnect connector CN7. Remove two screws from left, two from rear right, and one from front right. Lift. To remove the circuit board from the case, remove the screw holding the drive LED to the front panel. Remove one screw from right side and one from rear bottom. Remove eight screws from connectors CN2, 3, and 4. Unscrew the spacer the power supply rested on. Remove seven screws from circuit board and lift (Phew!).

Support Circuitry

C-128D owners should have no trouble finding room for a small perforation board to hold ICs Z2, Z3, and Z4. Owners of low-profile 128s will have to use the upside-down chip technique: the ICs are stuck to the main board upside-down and the necessary connections made by wire-wrapping directly to the pins. C-128D owners can use this method as well.

Before sticking any chip down, place a small adhesive label to the bottom and write its designator on it, and a dot to indicate the location of pin 1. Use a small piece of double-stick foam tape to hold each chip in place. This is considerably stickier that double-stick masking tape. Adhesives such as epoxy or cyanoacrylate (Krazy Glue,) SHOULD NOT be put directly in contact with the main board. Rubber and contact cements are OK, but they require "coaxing" to stick to the ICs. I positioned Z2 near U9 (near the memory chips,) Z3 and Z4 near the MMU(U7). Figure 2 illustrates my low-profile installation. This is only a suggested layout. See "Mechanical Alternatives" for other possibilities. The objective is to keep the leads as short as possible. "Ground" is available at pin 10 of U62 or the heavy trace by pin 34 of the MMU. Z2, not shown, can get +5v and Gnd from pins 14 and 7 respectively of U9. When completed, the longest wire should be the one connecting pin 12 of MMU-2 to pin 3 of Z2. This wire should not pass over the top of the MMUs.

When wire-wrapping to bare IC pins, I suggest that you strip each wire much less than you would for a normal wire-wrap. No more than 1/2 inch of bare wire should be exposed. Use a hand-wrapping tool, rather than a battery-powered gun. The wire should be slightly "backed out" of the tool. Hold the wire firmly while rotating the tool gently. The insulated portion should not become wrapped. You might want to practice on a "junk" IC before committing yourself.

Wire Z2-4 up to but NOT including the MMU connections and the cutting of resistors R29 and R30. Refer to the Signal Location table accompanying figure 1. Be sure to use the correct list for your machine("Flat" or 'D). C-128D owners should remember to use the U) pin numbers in parentheses in the schematic.

Piggy-Backing

The additional dynamic RAMs and second MMU will have to be "piggy-backed" to the old. Brand new ICs come with the pins splayed out, not at right angles to the body of the chip. This makes for a tight fit when installed by auto-insertion manufacturing equipment, but makes installation by hand difficult. For piggy-backing, to ensure good electrical connections between the pins of the upper and lower ICs, the pins of the upper chip must be straightened.

If you have a tool known as (what else?) a pin-straightener, the process is simple: stick the chip in the tool and squeeze. If not, straightening the pins one by one with pliers is NOT practical, especially for the MMU, which has 48 pins.

Begin with the DRAMs. Electrically, MOS ICs are more rugged now than they were when the technology was first invented. Still, it can't hurt to ground yourself when handling these ICs, but -- DO NOT GROUND YOURSELF AROUND "LIVE" EQUIPMENT!!! Look at each chip on-end to see how badly splayed out the pins are. Then lay the chip on its side, on a smooth, hard surface, and holding the chip by its ends, with the pins firmly against that hard surface, GENTLY "lean on it". I don't know how else to describe the process.

The goal is to bend all the pins simultaneously, by the same degree. Again, you might want to practice on a junk IC.

When you're satisfied that both rows of pins are reasonably perpendicular, slip the chip over a similarly sized IC on the main board to test the fit. Once this is acceptable, you can use pliers to bend outward the pins that will not be in contact with the lower IC. Don't try to make them horizontal. Bend them out just enough to clear the lower pins. C-128D owners need to bend out ONLY pin 16 of the DRAMs. low-profile owners, see the text below. With your soldering iron, "tin" the skinny parts of the pins that will be in contact with the lower ICs. Repeat this straightening, bending and tinning process for all the DRAMs. Don't solder them in just now. Set the prepared DRAMs aside and straighten the pins of your "extra" MMU. Before you check its fit over the socketed MMU (U7), remove any heat-sink compound from the original. Then clean your hands. If you get any of that stuff on the pins, it will be impossible to get a good, solid solder joint.

Remove the socketed MMU and decide which MMU will be the bottom one (MMU-1), and which will be the top (MMU-2). It really doesn't matter which is which.

Check the fit of the two MMUs again. Now bend outward (GENTLY!) the specified pins of what will be the upper MMU; 3 through 15, 23, and 41 through 48. Note the notch that indicates which end is pin 1. (For 40-pin and larger ICs, I put a drop of White Out or Liquid Paper beside each "5" pin -- 5, 10, 15, etc. That way, I'm not always counting pins from one end or the other. Remember, the MMU has 48 pins. Solder a wire to the wide portion of pin 23 of what will be the lower MMU. This will be the A6/7 signal to ICs Z3 and Z4. C-128D owners can skip this step and instead obtain A6/7 at pin 8 of U54, a 74LS32. In the flat 128, this signal is at pin 6 of U54, but that IC is at the left side of the video box, quite some distance away. Tin the slender portions of the pins of the upper MMU that will be in contact with the lower MMU. Place the second MMU atop the original, making sure the notches are at the same end, and slip thin cardboard, manila or pieces of a match-book cover, between the two ICs, at the ends, so that there is some space between them. Solder the skinny part of the upper pins to the wide part of the lower. This is another point where you may want to first experiment on junk ICs. Avoid excessive heating; do not linger on any one pin for an extended length of time. Once cool, all solder joints should be shiny. Dullness indicates a "cold" and possibly intermittent solder joint. Confirm that the connections are intact before inserting the MMUs in their socket. Once the MMUs are socketed, soldering to the pins risks having solder flow down into the socket, making removal of the chips difficult. Except for R/W, all needed MMU connections will be made to the "shoulders" of the bent-out pins of the upper MMU.

Now you are ready for the DRAMs.

Memory - C-128D

For the 128D, DRAM installation is straightforward. Slip the prepared DRAMs over U38-U41 on the main board, making sure the notches are in the same direction, and solder all pins except the bent-out pin 16. Solder a wire from pin 16 of the chip on top of U38 to pin 16 of the chip on top of U39. These chips are NOT next to each other, so the wire will "leap-frog" U40. Do the same thing for the DRAMs atop U40 and U41. Slip "spaghetti" on the leads of the 68 Ohm resistors and solder one end of each to the two pairs of pin 16s. (Or, snip off the excess and use heat-shrink tubing or electrical tape to insulate the wires you splice to the resistors). The free ends of the resistors go to pins 9 and 10 of Z2. Snip the U9 ends of resistors R29 and R30 on the main board (the ends toward the rear,) and connect these to pins 13 and 14 of Z2. Note that the order shown in figure 1 is correct. Z2 output Y1 selects RAM 1 and Y2 selects RAM 0; outputs Y5 and Y6 select RAM 3 and RAM 2 respectively. This is significant only if you run into trouble, and need to determine which block is malfunctioning.

Memory - Low profile

For the low profile 128, I found the prospect of piggy-backing 16 64K-by-1 DRAMs unappealing, so I used 4 64K-by-4s. In figure 3, you will see that, even though the 64K-by-4 is longer, when aligned as shown most of the signals line up properly. It doesn't matter that the Address bits are numbered differently. So, if we bend out pins 1-3 and 15-18, and let pins 1 and 18 over-hang the existing DRAMs, the rest of the signals will be taken care of by piggy-backing. (Note that +5v and ground are at positions the reverse of ordinary logic chips. Note further that the 64K-by-1 has separate Data in and out pins, which are wired together in the flat 128, and the 64K-by-4 has bi-directional Data pins -- which are numbered beginning with "D1" instead of "D0.")

Figure 4 shows my DRAM installation and part of its wiring. The easy part. Note that, because the chips are so close together, pins 2, 3 and 15-17 of the added DRAMs are clipped short. Pins 1 and 18 of the upper ICs are daisy-chained and connect to pin 16 of U49. This takes care of Gnd and /OE. The /CAS lines and 68 Ohm resistors are dealt with in the same manner as described above for the 128D. All that remain are the Data lines.

Solder wires from pins 2, 3, 15 and 17 of the chip atop U53 to the same pins of the chip atop U52. Wire each of these to each pin 2 of U42 through U45. This takes care of Data bits 4-7. Wire the same pins of the DRAMs atop U50 and U51 in the same way, with the final connections to each pin 2 of

U38-U41. And that takes care of bits D0-D3.

This "daisy-chaining" of the data lines will be much easier if the wires are prepared in advance. Cut short lengths of wire-wrap and strip the ends to leave about 5/8 inch of insulation. Tightly "pigtail" splice a longer wire to each short wire to provide the connection to pin 2 of each of the front row of mother board DRAMs. Solder the splices, then clip off the excess and you have a pre-tinned assembly to solder to the appropriate DRAM pins. The mother board DRAMs in the low-profile 128 are rather close together and numerous p.c. traces run in between the pins, providing many opportunities for the creation of short circuits. Take your time, and inspect your work often.

Not shown in figure 4 are the bypass capacitors on the main board, between the two rows of DRAMs. These make for slightly cramped quarters for the piggy-backed chips. Why not use the front row of DRAMs, you ask, where there is more room? Because the R.F. shield slopes in the front, to accommodate the keyboard, and will not clear double-high chips. Even if you dispense with the shield, the keyboard might not clear the piggy-backed chips if they were mounted along the front row. So sixteen 64K-by-1s would not have fit anyway!

Completion

Insert the double MMU into its/their socket and make the remaining connections shown in figure 1 between it/them and Z2, Z3 and Z4. Signals D6 and D7 to Z3 could be obtained at pins 41-42 of the lower MMU, but these will be difficult to get at once the two MMUs are piggy-backed. See the Signal Location table accompanying figure 1.

Testing

Check and re-check your wiring several times. Having come this far, it would be tragic to let the simplicity of the schematic mislead you into thinking that mistakes cannot be made. Make sure no little bits of wire or blobs of solder are left on the circuit board. Reassemble your machine as much as necessary to make it operable. This is where things can go wrong, so use caution. Turn your monitor on first so you won't have to wait for it to warm up. A blank screen can be very disconcerting, so make sure the monitor is in the correct mode with respect to the 40/80 key. If, when you turn on the computer, you fail to get the sign-on screen, immediately power down and find your mistake. Are the right pins of MMU-2 in contact with MMU-1? Are the right pins bent out and NOT in contact? Check the DRAM wiring. Did you snip the correct ends of R29 and R30? Did you use the correct Signal Locations for your machine? (See "Troubleshooting" at the end of this article.)

When you get a normal start-up -- complete with an attempt to boot a disk -- you'll see that the "bytes free" message still says "122365." That's built into ROM and remains unchanged by the modification. Run the short program from the Introduction. If you don't get

```
 0   1   2   3
```

the modification still isn't right.

For a more convincing test, use the Machine Language Monitor to fill the same locations, say, $n8000-n80FF (where n = 0 to 3), in the four banks with different values, then examine them with "M".

Disassemble $FF05-FF44. All four banks should be identical. The last six bytes of Ram, from $FFFA to $FFFF should contain 05 FF 3D FF 17 FF in all four banks. If you use the MLM "C" command for this, ignore the "?" it prints when done. If locations $3FFF5-3FFF7 (RAM 3) contain 43 42 4D, the codes for the characters "CBM", then it isn't really RAM 3; only RAM 1 should contain that string.

Use your 128 disassembled for a few days, to confirm that all is well. If your 128 is low-profile, before you replace the R.F. shield, put masking or electrical tape on the underside, above the upside-down ICs. They should not hit, but 'tis better safe than sorry. Flatten the heat-sink tab for the MMU and put tape around the edges of the opening. If you're handy with tin-snips, you can cut off the tab and widen the opening.

You can dispense with the R.F. shield, but if any neighbors complain of interference with TV, radio, or telephone, you may find yourself in conflict with government regulatory agencies, which is never pleasant. In general, if your computer does not disrupt YOUR over-the-air (NOT cable,) TV reception, it should not bother your neighbors. If in doubt, ask them!

Software

Program 1 tests a few features of the modification. It BLOADs four Doodle format pictures to each of the four banks, demonstrating that BLOAD can access the new blocks. (Except for the load address, Doodle pictures are completely compatible with the 128's hi-res format. If you do not have Doodle, substitute the names of any 40-column hi-res 128 format pictures you may have. Multi-color pictures such as Koala CANNOT be used for this demo.) The program rapidly displays the four pictures via a simple POKE, altering the VIC/DMA block bits of $D506/54534. (If you see only two pictures, you have only RAM 0 and 1.) Halt the program by pressing any key. Adjust X in line 240 to change the speed.

Restart the program with RUN 220.

The picture in RAM 1 occupies 9K of variable space. Since this program has few variables, there is little chance of them bumping into the picture.

While it is true that the same thing could be done on a C64, by selecting different 16K video blocks, you'd be hardpressed to find the room for four pictures and a program that could do anything more than display them. (In 128 mode, the presence of the MMU registers at $FF00-04 complicates the use of the "last" 16K video block from $C000-$FFFF.) Note that while it's easy to display a bit-map in any RAM block, the Basic graphic commands only work in RAM 0.

Program 2 is the Basic loader for a relocatable Bank-to-Bank memory mover. If MOVE is where the routine is located, it can be called from Basic with

SYS MOVE, source bank, destination bank,,, source start, source end+1, destination start

The 128 must be in BANK 15. The three commas must be present. Source and Destination Banks are 0-15. Source end+1 must be greater than source start or you'll receive ?ILLEGAL QUANTITY. Upon return, Carry will be 0 if the move was complete. This can be determined by reading the status register into a variable with something like RREG,,,SR: IF SR AND 1=1 THEN the move was incomplete. The routine will not let the destination reach page $FF, and thereby clobber the MMU registers or the IRQ, NMI and other routines and pointers. However, the destination can begin in page $FF, if you're the adventurous type. In that case, if the number of bytes being moved causes the destination to "roll over" to zero page, the move will be halted and Carry set to 1. There are no restrictions on the source address, except that the last byte of memory cannot be moved since, with regard to addresses, 65535 +1 = ILLEGAL QUANTITY. (The same restriction, by the way, applies to BSAVE.)

If you need a non-standard configuration (like RAM 2 with I/O,) POKE the source and destination CONFIGURATIONs (NOT Bank #s,) in 206 and 207 respectively, and call the Mover with SYS MOVE+25,,,,, srce start, srce end+1, dest start. These zero page locations are used as pointers for PRIMM which, as far as I can tell, is the only use the system makes of them. To utilize the Mover in machine language, store the srce and dest CONFIGs in $CE and $CF, store srce start in low-byte high-byte form in $AC-AD, srce end+1 in $AE-AF. Then LDY and LDA with the low and high bytes of dest start and JSR MOVE+47.

The Mover doesn't disable interrupts, and will not interfere with IRQ-driven activities, such as split-screens and sprite motion. It owes its small size to the use of several system routines. It owes its speed -- which isn't exactly blazing -- to the fact that two system routines are NOT entered via the Kernal jump-table. When called the prescribed way, INDFET and INDSTA each call GETCFG to convert the Bank number in .X to a configuration value. Two conversions for each byte moved proved to be too time consuming. The Mover performs the bank-to-config conversions just once, at the start of the routine, and saves them in $CE-CF. The routine later picks these up and calls INDFET and INDSTA at their common memory resident locations.

Locations $AC-AD and $AE-AF are used as start and end pointers for the source. A Kernal routine at $EEB7 is used to check that end is greater than start. Later on, to save time that would otherwise be spent in JSR and RTS, the Mover does its own comparison to determine when the end has been reached. The Basic ROM routine at $880F examines Basic text for a comma and evaluates the expression which follows to a two-byte value, 0-65535, and ReTurnS with the low-byte in .Y and the high-byte in .A.

Program 3 is an adaptation of the "Globe" animation demo on the 1764 REU disk. You need a copy of that disk for this program to work. Like the original, this program BLOADs more than thirty compressed hi-res images. The original program de-compressed the pictures and stashed them in the REU. Since there isn't enough room for that many bit-maps in the slightly less than 128K of RAM 2/3, this adaptation keeps them compressed until just before they're displayed. The previously described Mover transfers the compressed data from RAM 2/3 to a graphic area in either RAM 0 or RAM 1. Then they're de-compressed and bits 6/7 of the Ram Configuration Register at $D506/54534 are changed to display the picture. I call this sort of animation, moving a full bit-map, "brute force." For "Globe" (and especially "Pound", which is multi-color,) it works only because the colors are static and unmoving.

Although the machine language program "compress.bin" was intended for 64 mode, when located in the same RAM block as a bit-map, it works fine in 128 mode. (It is BLOADed twice to put it in both RAM 0 and 1.) It assumes the bit-map is at $2000/8192. SYS 49155 uncompresses and SYS 49152 compresses a bit-map. PRINT PEEK(253) + 256 * PEEK(254) then gives the end address of the compressed bit-map data.
Colors are not affected.

The demo sets aside 36 2304-byte buffers – 18 each in RAM 2 and 3 -- to hold the compressed data. There is room in each bank for 27 such buffers. That size represents the largest .cmp file in disk blocks. For some reason, when Globe files are de-compressed, some debris is left in the first 768 bytes of the bit-map. Line 400 clears that by using a short-coming of the Mover as a feature: When source and destination over-lap in the same Bank, the move becomes a fill. Delete line 400 for "Pound." (Which explains the need for the seemingly redundant BANK 15 in line 420.)

Suggestions (software)

Program 1 and the Mover can be combined for a "brute force" form of hi-res animation. Banks 2 and 3 can each hold up to 6 hi-res pictures. You could move a picture into the graphic area of RAM 0 and while it is being displayed, move the next picture into the same area of RAM 1, then change the VIC/DMA bits to display that RAM. Move the next picture into RAM 0, etc. Multi-color pictures have the additional complication of Color Memory.

The capacity of Banks 2 and 3 equals 2/3 of a single-sided floppy. While this is not as spacious and cannot be as fast as a 1750 REU, it's still nothing to turn up one's nose at. Someone out there should be able to coax Geos 128 into utilizing the new Banks, alleviating some of its tedious disk activity.

On the Basic side, the Mover could form the basis of a RAM-drive. Or it could completely stash Banks 0 and 1 in 2 and 3, saving your current program and all its variables while you load and run a second program, then return to where you left off. The Mover would need some modification to do this though.

To determine compatibility, the Intro program, or its MACHINE LANGUAGE equivalent, could determine if a 128 really has Banks 2 and 3 or is non-modified.

And what could CP/M do with 256K of memory?

Suggestions (hardware)

I do not anticipate any software incompatibility with this modification. But should any arise, a two position switch can be installed to cause the machine to appear to be non-modified. When the switch connects input "C" of Z2 to MMU-2, Banks 2 and 3 are available. When the switch is in the Gnd position, the new Banks disappear, and accesses of them default to RAM 0 and 1, as in a non-modified C128. Use a double-pole switch. The other pole might come in handy.

JiffyDOS and others...

For compatibility with JiffyDOS -- and perhaps other disk speed-ups -- combined with (some) serial printers, it may be necessary to make a small addition to the system mother board. Connect a 1000 Ohm 1/4 Watt resistor between +5 Volts and "FSDIR" -- the line from MMU-1 that tells the Fast Serial circuitry which way the data is going. This will prevent spurious FILE NOT FOUND errors when JiffyDOS 128 and the memory expansion modification are both enabled. This problem occurs ONLY in 128 mode when the printer is on AND the 128 is in SLOW mode ??? FSDIR is available at pin 4 of U58 (74LS03) in both the low-profile and the 'D. Note that the IC designators on some 128s are not very readable. The slashed zero can easily be mistaken for an "8."

How does the pull-up resistor eliminate FILE NOT FOUND? I wish I knew. JiffyDOS does not use the Fast Serial capability, but I can see (vaguely,) how a serial printer might be involved in the problem. C64 mode automatically disables the Fast Serial circuitry, so any glitches therein would have nowhere to go. But why does FAST mode, specifically, the blanking of the 40 column screen, eliminate the problem?

One problem may remain: In 128 mode, the JiffyDos wedge command (Load&Run) sometimes doesn't. This might be caused by a bug in the wedge. With JiffyDOS enabled but its wedge disabled, and a 128 Dos wedge from Compute!'s Gazette enabled, the (Load&Run) command works fine. Run"prog.name" ALWAYS works. So does JiffyDos's (Load&Run) in C64 mode ?

Troubleshooting

As is usually the case in a project such as this, the most common causes of it failing to work are wiring errors and bad connections; intermittent or "cold" solder joints and short circuits caused by solder bridges. The chances of shorts causing any permanent damage to the 128 are minimal. The difficulty is in locating them.

If the 128 powers up "dead" -- blank screen, no drive activity -- the problem could be anywhere, but you may not have to look everywhere. Look for wiring errors in the enabling of the DRAMs serving as RAM 0 and 1 (Z2) or shorts on the address or data lines of those DRAMs as a result of the piggy-backing. If the 128 powers up with a garbled screen, characters that don't look right, or no disk activity, the two MMUs are probably interfering with each other. Check the MMU piggy-back connections and the wiring of Z3/4. If you run the multiple DOODLE program and the pictures in RAM 0 and 1 are fine, but those in the added RAM(s) are complete garbage, check the wiring of Z2 and the two MMUs. If the pictures in additional RAM are partially garbled, one or more address and/or data connections to those RAMs might be missing. This could be caused by poor piggy-back connections. (Remember, in the low-profile it doesn't matter if the address/data bits to added RAM are "scrambled" -- not in order -- as long as you have all of them and the scrambling is consistent.) Examine those banks with the MLM .m command. If bytes change every time you display them, that's a sure sign that one or more address or data bits are missing. Again, shorts on the address or data lines would cause the 128 to not work at all.

Supplement

Low-profile Mechanical Alternatives or
If I knew then what I know now...

When I expanded my low-profile 128 to 256K, the second 128K, consisting of four 64K-by-4 dynamic RAMs, was added using piggy-backing. To increase the modification to 512K, eight more DRAMs were installed using more piggy-backing for the first four and a small board for the remaining four. Now that I've modified more than one "flat" 128, I've devised other strategies.

1. There is room just in front of the video box for a small perforated "daughter" board. This could hold 4 DRAMs for the 256K modification, and could be large enough to include the support chips -- except for the second MMU which should still be piggy-backed because of the many connections. The daughter board could even be large enough to include the chips needed for the 512K expansion, which could be added at a later time. The four DRAMs mounted on the daughter board would be wired to the appropriate points on the mother board. When the time came to expand to 512K, the eight additional DRAMs could be piggy-backed to the daughter board DRAMs or to the second row of mother board DRAMs. (Or any combination thereof.)

2. The method I've actually been using has been to use a small board only for the DRAMs; 4 for the 256K modification or, using piggy-backing, a total of 12 for the 512K. (The piggy-backing is as described for the 128D in the 256K plans.) The support chips are still mounted upside-down using double-stick foam tape and the necessary connections made by wire-wrapping to the pins.

Method 1 has the advantage of being more easily undone, should the need ever arise. I've found method 2 to be the fastest and least troublesome, since the only "hidden" wiring is on the underside of the DRAM board. For EITHER method, if dip sockets are used, which I STRONGLY recommend, the R.F. shield won't fit -- with or without piggy-backing! (There is enough headroom -- though just barely -- for three levels of DRAMs.) You could dispense with the shield, or make a cut-out, using tin-snips, sufficient to clear the add-on board components. The area of the mother board beneath the add-on board MUST be insulated with a layer of vinyl tape. The add-on or daughter board is held in place with small pads of double-stick foam tape.

When positioned in the logo (C=) area of the low-profile mother board, the daughter board is centrally located to all the signals it needs. As illustrated, the daughter board is tight, but workable. the dimensions given are the largest possible for the area involved. The positions of the ICs (particularly Z2,) were chosen to keep the signal leads as short as possible. Z1, the second MMU, should still be piggy-backed due to the many connections involved.

Flat ribbon-cable is handy for groups of conductors, but its stiffness sometimes makes it difficult to position. I have found "tone-arm" cable, the thin wires used inside a phonograph tone arm, very useful. (The smallest gauge is a bit too fine to work with. Choose a size similar to or slightly larger than wire-wrap, which is usually #28 or #30.) It comes jacketed and unjacketed, shielded and unshielded. The shield and jacket are easily removed and the four conductors are flexible enough to make turns that ribbon cable won't. (For a little more work and a little less flexibility, you could also assemble your own bundles of twisted wire-wrap, in different colors so you can tell them apart.) Bundles of four conductors can be assigned to lower and upper address bits and lower and upper data bits. Once everything is tested and functioning properly, a dab or two of hot-melt glue will hold the cabling in place.

Materials List

8722 MMU (*CO389 $16.95)
Available from Grapevine Group

ICs Z3 and Z4 ($ 8.00 per pair)
Available from Richard Curcio
or Parsec, Inc.

The below parts are available from
B.G. Micro, the Grapevine Group
and supplies are available from
Radio Shack.

1) 74LS138 (B.G. Micro)
4) 64Kx4 dynamic RAMs
(4464, 41464, etc.)
150ns or faster (* call)
2) 68 Ohm, 1/4W resistors
1) 2K Ohm, 1/4W resistor
1) 1K Ohm, 1/4W resistor (for JiffyDos)
1) DPDT switch (disabling)
solder
30 gauge wire-wrap wire
electrical tape,
"spaghetti", etc.

* Grapevine Group part numbers

Richard Curcio can upgrade your computer for you. Contact Richard at the address listed for the latest pricing and shipping information. Be sure to use a self addressed stamped envelope!

Addresses of companies and persons mentioned in this article:

Grapevine Group, Inc.
3 Chestnut St.
Suffern, NY 10901
(914) 357-2424

Richard Curcio
22 Seventh Ave.
Brooklyn, NY 11217

ICs Z3 and Z4 may be obtained from Richard for $8.00 the pair for the US and Canada or $10 (US) outside North America.

B.G. Micro
P.O.Box 280298
Dallas, TX 75228
1-214-271-5546

Parsec, Inc.
PO Box 111
Salem, MA 01970-0111

Foreign orders (excluding Canada) of Z3 and Z4 MUST include, on a separate sheet, the following:

Importer's Statement:

We will import these COCOM controlled goods and will not ship them outside COCOM participating countries without prior authorization from the appropriate national authorities.

____
(signed)

Program 1

```
100 GRAPHIC1:TRAP310:GOSUB340:BANK15
110 REM BLOAD 4 DOODLE FORMAT PICTURES INTO THE 4 BANKS
120 POKE54534,PEEK(54534)AND63:REM VIC DISPLAYS RAM 0
130 BLOAD"DDMIDDLE EARTH",B0,P7168
140 POKE54534,PEEK(54534)AND63ORCR(1):REM DISP. RAM 1
150 BLOAD"DDGRAPHIC BLOCKS",B1,P7168
160 POKE54534,PEEK(54534)AND63ORCR(2):REM DISP. RAM 2
170 BLOAD"DDLETTERS",B2,P7168
180 POKE54534,PEEK(54534)AND63ORCR(3):REM DISP. RAM 3
190 BLOAD"DDPLAN",B3,P7168
200 REM RE-READ ARRAY SO PROG MAY BE RESTARTED
210 REM WITHOUT RELOADING PICTURES
220 TRAP310:GOSUB340
230 BANK15:GRAPHIC1:C=0
240 X=199:REM DISPLAY SPEED
250 C=C+1:IFC>3THENC=0
260 GETA$:IFA$<>""THEN310:REM EXIT IF KEY PRESSED
270 POKE54534,PEEK(54534)AND63ORCR(C)
280 FORT=0TOX:NEXT
290 GOTO250
300 REM SET VIC DISPLAY FOR NORMAL
310 POKE54534,PEEK(54534)AND63:GRAPHIC0:END
320 :
330 REM READ VALUES FOR RAM CONFIG REGISTER
340 RESTORE:FORI=0TO3:READ D:CR(I)=D:NEXT:RETURN
350 DATA 0,64,128,192
```

Program 2

```
100 REM RELOCATABLE BANK-TO-BANK MEMORY MOVER
110 REM CHANGE "SA"
120 :
130 BANK15:SA=4864
140 FORI=0TO107:READD:POKESA+I,D:NEXT
150 DATA 201, 16,144,  3, 76, 40,125,224
160 DATA  16,176,249,133,206, 32,107,255
170 DATA 133,207,166,206, 32,107,255,133
180 DATA 206, 32, 15,136,132,172,133,173
190 DATA  32, 15,136,132,174,133,175, 32
200 DATA 183,238,176,216, 32, 15,136,132
210 DATA 195,133,196,169,172,141,170,  2
220 DATA 169,195,141,185,  2,160,  0,166
230 DATA 206, 32,162,  2,166,207, 32,175
240 DATA   2,230,172,208,  2,230,173, 56
250 DATA 165,172,229,174,165,173,229,175
260 DATA 240, 16,230,195,208,225,230,196
270 DATA 240,  6,165,196,201,255,144,215
280 DATA  56, 96, 24, 96
```

Program 3

```
100 IFRGR(0)=5THEN
  PRINTCHR$(15)"*** 40 COLUMNS ONLY ***"
110 TRAP440
120 COLOR0,1:COLOR4,1:GRAPHIC0:SCNCLR:COLOR5,15
130 POKE57,0:POKE58,28:CLR:
  REM LIMIT VARIABLES TO BELOW BIT-MAP IN RAM 1
140 SLOW:INPUT"LOAD PICTURES (Y/N)";A$
150 IFA$="Y"THEN190
160 IFA$<>"N"THEN140
170 BANK0:IFPEEK(4864)<>201OR
  PEEK(49155)<>76THENPRINT"NO ML!":END
180 FAST:GOSUB480:GOTO300
190 BLOAD"COMPRESS.BIN",B0
200 BLOAD"COMPRESS.BIN",B1
210 FAST:GOSUB480:GOSUB680:SLOW
220 PRINT""
230 P=0:B=2
240 P=P+1:IFP>36THEN300
250 IFP>18THENB=3
260 PRINT"LOADING IMAGE"P"OF 36"
270 :
280 BLOAD"GLOBE"+STR$((P-1)*10)+".CMP",B(B),P(SS(P-1))
290 GOTO240
300 COLOR1,10:GRAPHIC1,1
310 REM COPY CLR'D & COLORED GRAPHIC AREA TO RAM 1
320 BANK15:SYSMOVE,0,1,,,7168,16384,7168:SLOW
330 :
340 P=36
350 P=P-1:IFP<0THENP=35
360 REM MOVE COMPRESSED PIX TO BIT-MAP AREA
370 BANK15:SYSMOVE,SB(P),DB(P),,,SS(P),SS(P)+SZ,BM
380 BANK(DB(P)):SYS49155:REM DE-COMPRESS IT
390 REM CLEAN IT UP (USE MOVER TO FILL 1ST 3 PGS
  W/ZEROES) * GLOBE ONLY *
400 POKEBM,0:BANK15:SYSMOVE,DB(P),DB(P),,,BM,BM+767,BM+1
410 REM CHANGE MMU RCR BITS 6-7
420 BANK15:POKE54534,PEEK(54534)AND63ORVB(P)
430 GETA$:IFA$=""THEN350
440 POKE54534,PEEK(54534)AND63:GRAPHIC0
450 END
460 :
470 REM INITIALIZE VARIABLES & ARRAYS
480 SZ=2304:A=1024:REM SIZE, START
490 MOVE=4864:BM=8192:REM MOVER, BIT-MAP
500 DIM VB(35):REM VIC  BANK VALUES
510 DIM SB(35):REM SRCE BANK FOR MOVER
520 DIM DB(35):REM DEST BANK FOR MOVER
530 DIM SS(35):REM SRCE START ADDRESSES
540 :
550 REM FILL VB() & DB()
560 FORI=0TO34STEP2:VB(I)=0:DB(I)=0:NEXT
```

(continued on next page)

Program 3(continued)

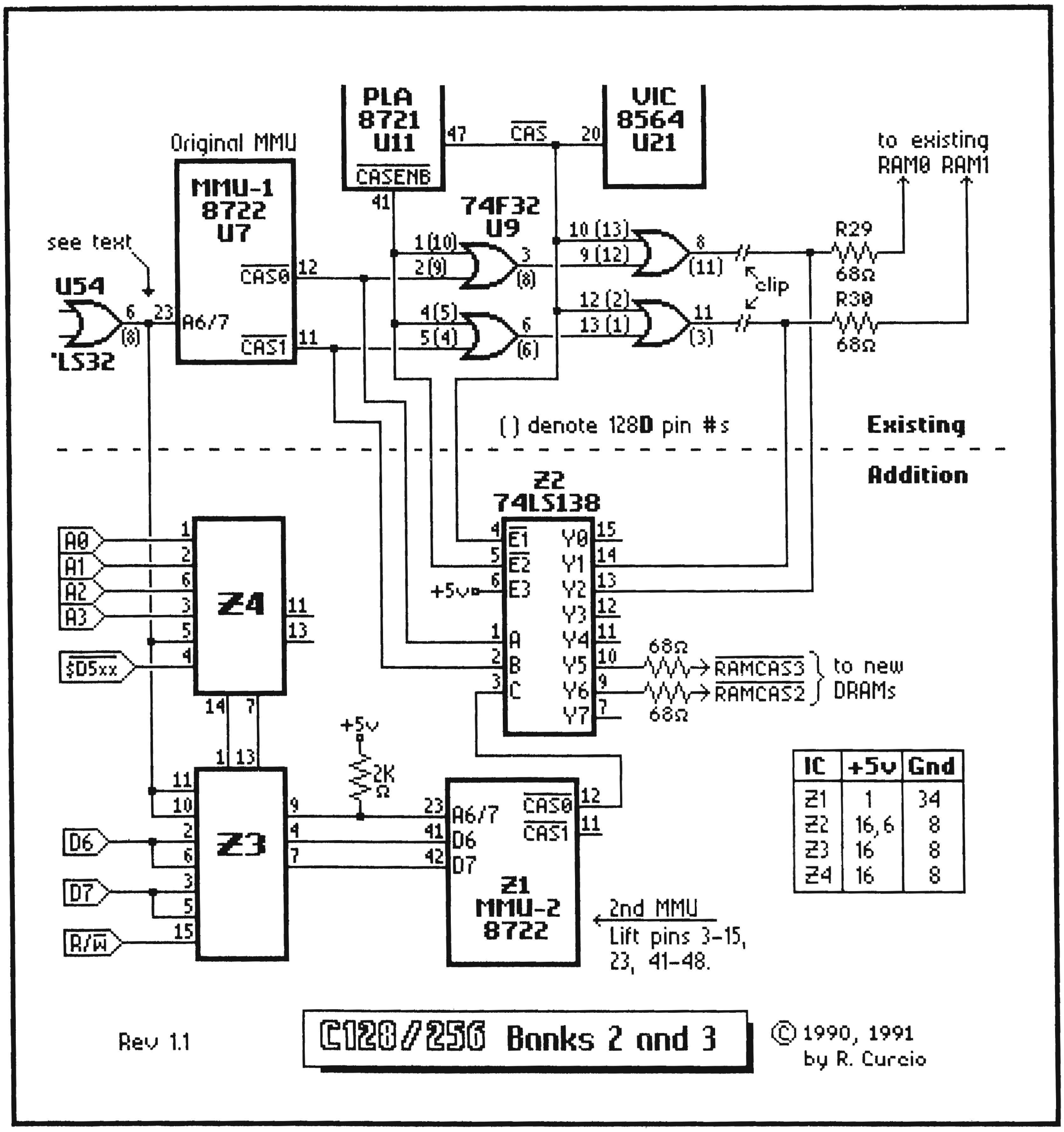
```
570 FORI=1TO35STEP2:VB(I)=64:DB(I)=1:NEXT
580 :
590 REM FILL SB()
600 FORI=0TO17:SB(I)=2:NEXT
610 FORI=18TO35:SB(I)=3:NEXT
620 :
630 REM FILL SS()
640 FORI=0TO17:SS(I)=A:SS(I+18)=A:A=A+SZ:NEXT
650 RETURN
660 :
670 REM *** INSTALL MOVER ***
680 RESTORE:BANK15
690 FORI=0TO107:READD:POKEMOVE+I,D:NEXT
700 RETURN
710 DATA 201, 16,144,  3, 76, 40,125,224
720 DATA  16,176,249,133,206, 32,107,255
730 DATA 133,207,166,206, 32,107,255,133
740 DATA 206, 32, 15,136,132,172,133,173
750 DATA  32, 15,136,132,174,133,175, 32
760 DATA 183,238,176,216, 32, 15,136,132
770 DATA 195,133,196,169,172,141,170,  2
780 DATA 169,195,141,185,  2,160,  0,166
790 DATA 206, 32,162,  2,166,207, 32,175
800 DATA   2,230,172,208,  2,230,173, 56
810 DATA 165,172,229,174,165,173,229,175
820 DATA 240, 16,230,195,208,225,230,196
830 DATA 240,  6,165,196,201,255,144,215
840 DATA  56, 96, 24, 96
```

MOVE.SRCE

```
1000 SYS4000
1010 ;
1020 ;POWER ASSEMBLER
1030 ;
1040 *= $1300
1050 ;
1060 .MEM
1070 ;
1080 ;***  BANK-TO-BANK MEMORY MOVER  ***
1090 ;
1100 SETUP  CMP #$10
1110 :      BCC XCP   ; SRCE BANK<16
1120 ERROR  JMP $7D28 ; ILLEGAL QUANTITY
1130 XCP    CPX #$10
1140 :      BCS ERROR ; DEST BANK>15
1150 :      STA $CE
1160 :      JSR $FF6B ; GET DEST CNFG
1170 :      STA $CF
1180 :      LDX $CE
1190 :      JSR $FF6B ; SRCE CNFG
1200 :      STA $CE
1210 :      JSR $880F ; SRCE START
1220 :      STY $AC
1230 :      STA $AD
1240 :      JSR $880F ; SRCE END+1
1250 :      STY $AE
1260 :      STA $AF
1270 :      JSR $EEB7 ; COMPARE START & END
1280 :      BCS ERROR ; START>END
1290 :      JSR $880F ; DEST START
1300 :      STY $C3
1310 :      STA $C4
1320 :      LDA #$AC
1330 :      STA $02AA ; INDFET POINTER
1340 :      LDA #$C3
1350 :      STA $02B9 ; INDSTA POINTER
1360 ;
1370 :      LDY #$00
1380 MOVIT  LDX $CE   ; GET SRCE CNFG
1390 :      JSR $02A2 ; DO INDFET
1400 :      LDX $CF   ; GET DEST CNFG
1410 :      JSR $02AF ; DO INDSTA
1420 :      INC $AC   ; INCRMNT PNTR
1430 :      BNE CSE
1440 :      INC $AD
1450 CSE    SEC
1460 :      LDA $AC
1470 :      SBC $AE   ; COMPARE TO END
1480 :      LDA $AD
1490 :      SBC $AF
1500 :      BEQ EXIT  ; DONE
1510 :      INC $C3
1520 :      BNE MOVIT
1530 :      INC $C4
1540 :      BEQ HALT
1550 :      LDA $C4
1560 :      CMP #$FF
1570 :      BCC MOVIT
1580 HALT   SEC       , ROLLED OVER
1590 :      RTS
1600 EXIT   CLC       ; COMPLETE MOVE
1610 :      RTS
1620 ;
```

CHANGES FOR "POUND.DEMO"

```
120 COLOR0,16:COLOR4,16:GRAPHIC0:SCNCLR:COLOR5,7
240 P=P+1:IFP>32THEN300
260 PRINT"     LOADING IMAGE"P"OF32"
280 BLOAD"POUND"+STR$(P-1)+".CMP",B(B),P(SS(P-1))
300 COLOR1,11:COLOR2,3:COLOR3,7:GRAPHIC3,1
340 P=0
350 P=P+1:IFP>31THENP=0
```



IC	+5v	Gnd
Z1	1	34
Z2	16, 6	8
Z3	16	8
Z4	16	8

Signal	**Location** "Flat"	"D"
$\overline{\text{\$D5xx}}$	U3, pin 14	-same-
A0	U10, 30	"
A1	U10, 31	"
A2	U10, 32	"
A3	U10, 33	"
R/$\overline{\text{W}}$	MMU, 32	"
D6	U13, 16	U13, 5
D7	U13, 18	U13, 9
$\overline{\text{CAS0}}$	U9, 2	U9, 9
$\overline{\text{CAS1}}$	U9, 5	U9, 4
$\overline{\text{CASENB}}$	U9, 1	U9, 5
$\overline{\text{CAS}}$	U9, 10	U9, 2
A6/7	see text	U54, 8

Notes:

Resistors are 1/4 Watt. Values non-critical. Use 39-82Ω for 68Ω, 1K-4.7K for 2KΩ. Leave unused IC pins un-connected.

Figure 1

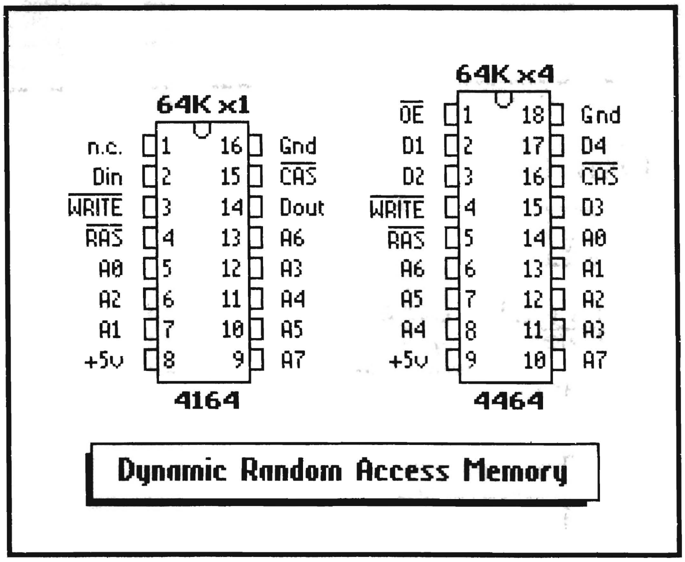

64K x1
n.c. 1
Din 2
$\overline{\text{WRITE}}$ 3
$\overline{\text{RAS}}$ 4
A0 5
A2 6
A1 7
+5v 8
16 Gnd
15 $\overline{\text{CAS}}$
14 Dout
13 A6
12 A3
11 A4
10 A5
9 A7
4164
64K x4
$\overline{\text{OE}}$ 1
D1 2
D2 3
$\overline{\text{WRITE}}$ 4
$\overline{\text{RAS}}$ 5
A6 6
A5 7
A4 8
+5v 9
18 Gnd
17 D4
16 $\overline{\text{CAS}}$
15 D3
14 A0
13 A1
12 A2
11 A3
10 A7
4464
Dynamic Random Access Memory

Figure 3

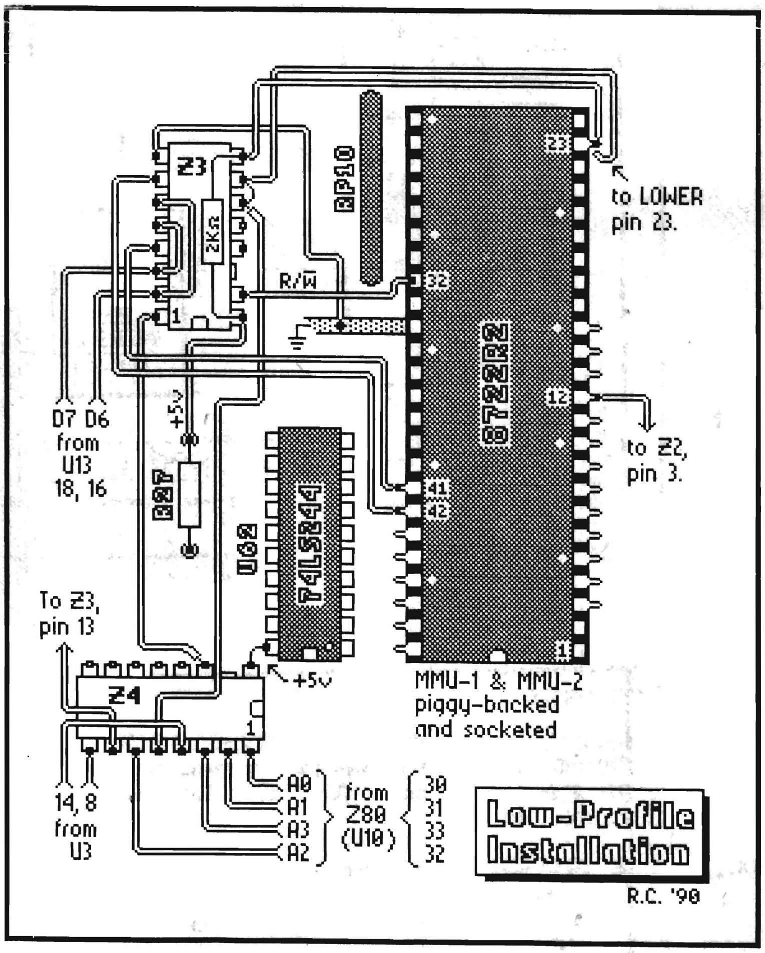
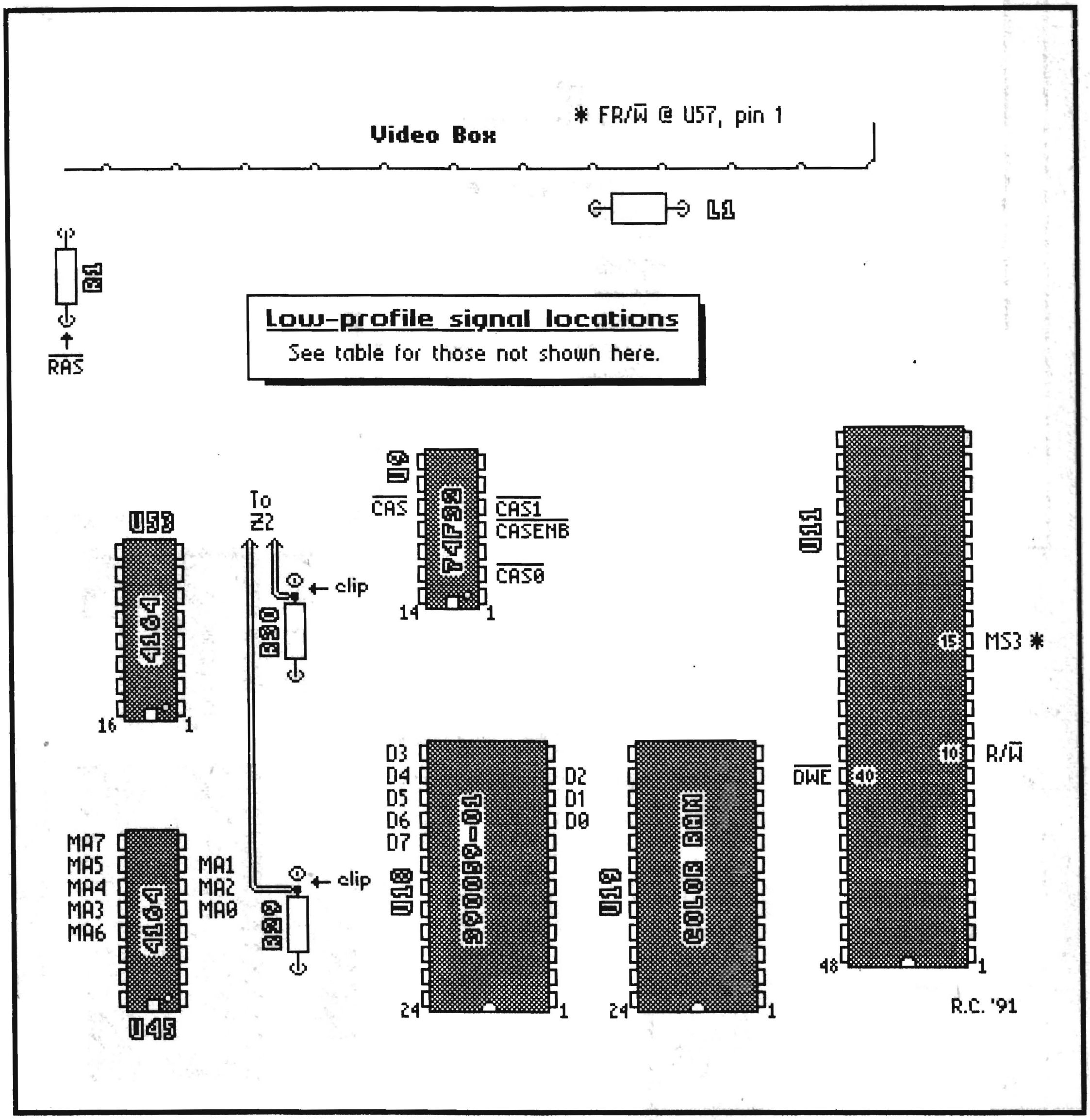


Figure 2

Notes:

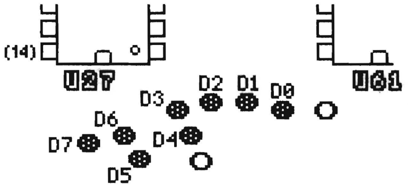
Z3 and Z4 upside-down. Z2 not shown.
U13 pin #s for flat 128 ONLY.

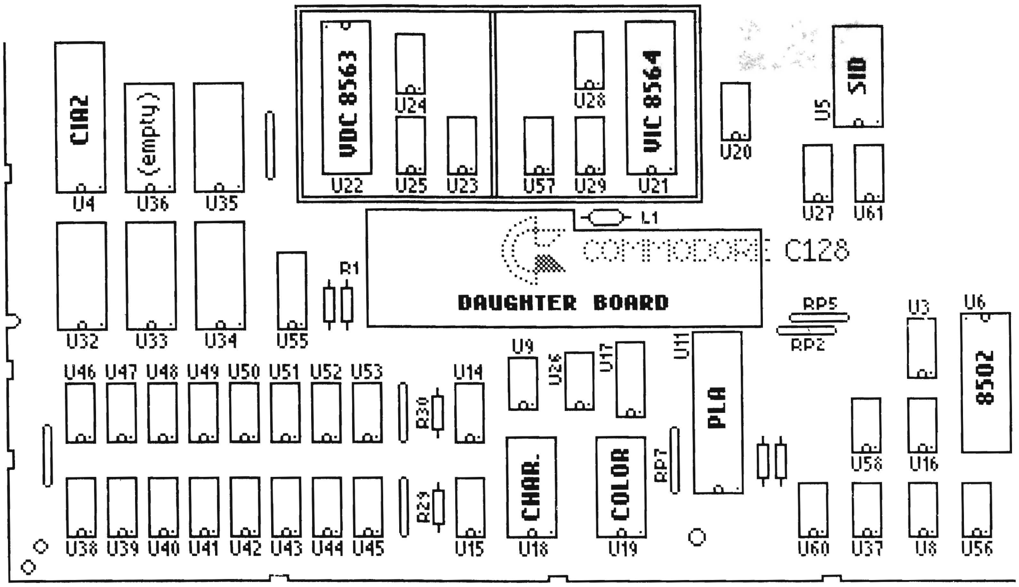


MA0-7, D0-7, /DWE and /RAS needed for DRAMs on daughter board. Be sure to take D0-7 from Character ROM U18, not from Color RAM U19.

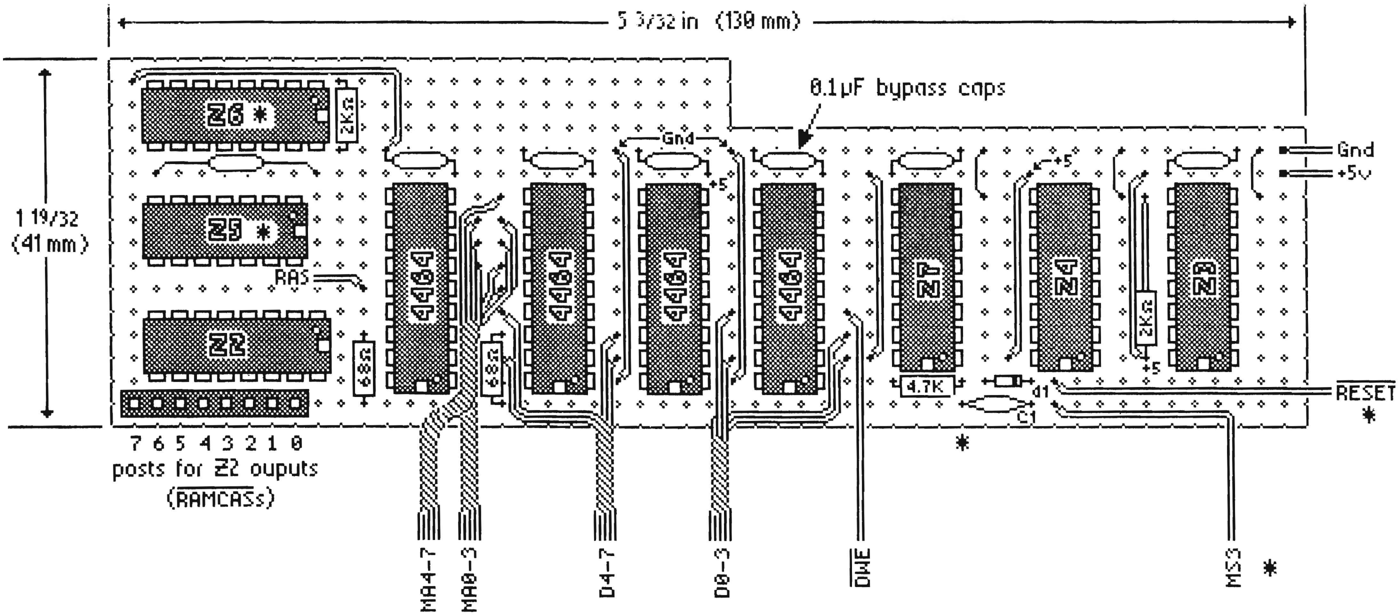
✱ MS3 and FR/W needed for 512K only.

Another place to get D0-7 on the low-profile.





Low-profile mother board with daughter board placement. (Partial view; not to scale.)



✱ ICs Z5, Z6, Z7, associated components and signals required for 512K only. Note reversed locations of +5v and ground for DRAMs.

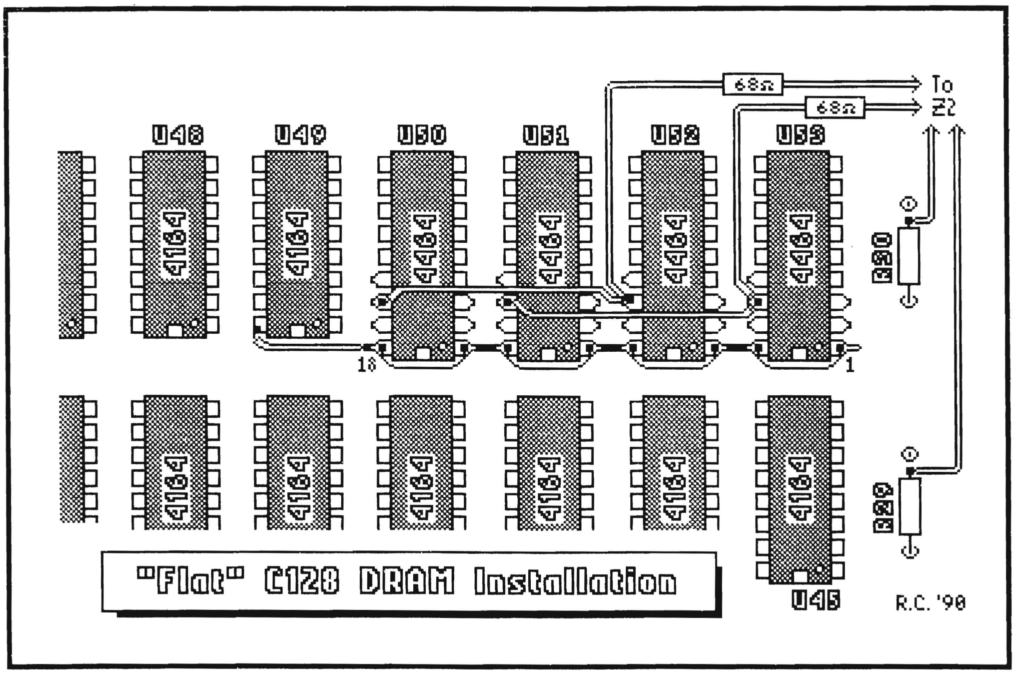


Figure 4

Piggy-backing 18-pin 64K x4s to 16-pin 64K x1s.
(Data lines not shown. See text.)

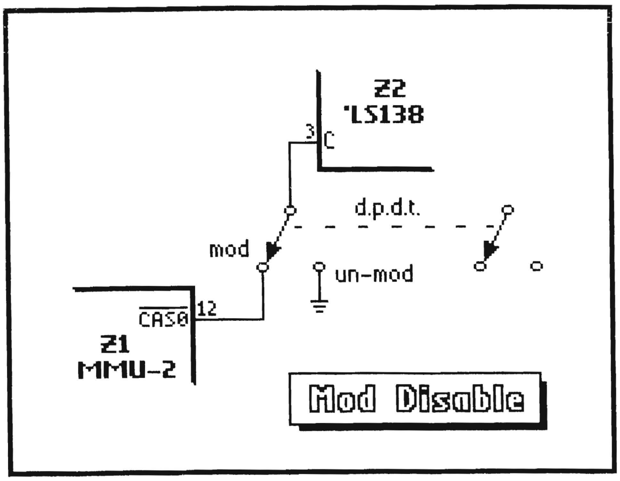
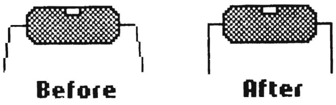
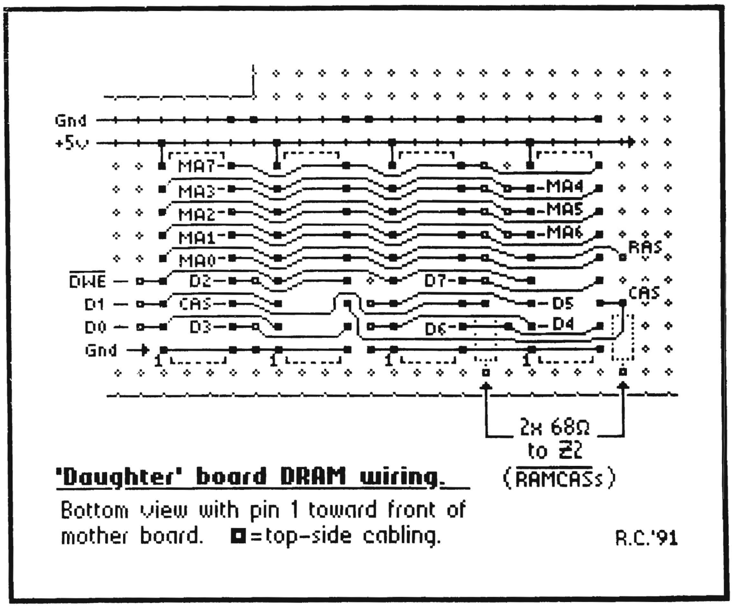


Figure 5





'Daughter' board DRAM wiring.

Bottom view with pin 1 toward front of mother board. ▣=top-side cabling.

Here To There by Frank Hudson

One of the most common questions asked on networks is "How do I get it from here to there?" The "it" is some important word processing file, picture, or other data; the "here" and "there" are often two computers with different disk and data formats. If you are a new owner or potential owner of an new PC other than a C-128 or if you have a friend with a PC other than a C-128 you may be asking some form of this question yourself.

Data is another matter.

If you have been using your C-128 for a long enough time you could have a good deal of stored data (word processor files, art work, BASIC program code, database and spreadsheet files) stored on C-128 disks. If any of this has continuing value to you, it can most likely be moved over to another PC by one of several methods. You have a couple of bridges to cross in transferring your data.

1) Disk format The C-128 does not share a common disk format with other computers. The C128 uses a disk storage method called GCR, most other computers such as IBM clones use a storage method called MFM. The Macintosh uses a very different disk storage method that is in a class by itself so we will not bother referring to that one here.

The most popular disk format now for PC clones is the 3.5 High Density diskette (1.44 megs). The fact that the most popular PC format is 3.5 high density disks while the most popular Commodore format is the 5.25 double density does pose some obstacles. But they can be overcomed. Luckily the C128 has excellent products that support reading and writing to the MS-DOS disk format. The two well known ones as Format Executive (CP/M mode) and Big Blue Reader 128 (native mode). By formatting a few MS-DOS disks and then using the MS-DOS transfer program, data files can be moved from one machine to the other fairly quickly. Furthermore, purchase of these packages allows you to exchange data with the millions of MS-DOS compatible computers in use today.

Most PCs that come with 3.5 or 5.25 High Density disk drives also have the ability to format and read low density disks. The other obstacle when it comes to disk formats is the CP/M format. Though there is a large collection of CP/M programs out there to be used by C-128 owners, sometimes dealing with the CP/M format can be frustrating. By using software such as Big Blue Reader 128 you can ease any pain of the different disk formats by formatting, reading, and writing to CP/M and PC disks directly on your C-128 (JBEE's note: I was really hesitant about buying BBR128 because of the high price tag. I would like to report that it has been worth every penny when working with PC clone disks and Ensemble files).

2) Null modem transfer. First off, a "null modem" is not a modem at all, it is just an adaptor that crosses a couple of RS-232 (modem/serial communications port) lines so that the computers on each end think that they are connected over the phone system. "Null modem" is a word combination like "non-starter", in effect "null modem" means you do not use a modem at all. With that out of the way, working with a null modem is one of the easiest ways to connect two computers, it is EASIER than connecting over the phone lines or modem to modem. But first you need the right equipment. On your C-128 you will need something called a "RS-232 interface". Like a printer interface, a RS-232 interface is designed to make your "non-standard" C128 port work with standard (meaning MS-DOS computers) peripherals. Various models are out there. I use one made by Supra, CMD will sell you a real deluxe model called SwiftLink, and there are others too. Once the interface is plugged into your C-128 you have what is in effect a standard MS-DOS serial port on the other end, ready to plug into a modem, or in this case into the non-modem thing we call a "null modem". Null modems are cheap, often $5 or so, and can be bought at places such as Radio Shack. You only need to be sure to get the right gender and size for your setup on each end. You can hook up directly from serial port to serial port with the C-128 RS-232 adaptor or use cables to allow the machines to be farther apart. Just be sure that the null modem you get has the right "innie" or "outie" pins for whatever setup you want to use.

After the shopping for a RS-232 interface and the correct null modem comes the easy part, actually transferring the files. With both machines off, connect them, then boot them up and load your terminal software on each machine. When your terminals are loaded they will tell you that they have sensed "carrier" i.e., they think they are "online". What you type on one computer should show up on the other. You can then use any transfer protocol (I like Y Modem Batch for this) to move files from one machine to the other, just as if you were online. Null modems can transfer as fast as your computers can handle the data. With a C64 I doubt you will get much better than 2400 baud and on the 128 I top out at 4800. The Swiftlink I mentioned earlier removes this limitation and will allow the C-128 to take data as fast as the disk drive can load or save it!

3) Modem to modem. Of course if the rigors of shopping for a null modem or RS-232 interface have got you down, then the simplicity of modem to modem transfers may appeal to you. All you need is two Hayes compatible modems (and the Commodore 1670 is Hayes enough for our purposes here) and a standard modular phone line cord (such as you would use to connect a desk phone or modem to the wall jack). Boot up your terminal software on both computers, set your terms to the speed of the slowest modem in the set of two, and connect the two modem's line jacks to each other. Then type: "+ + +ATZ" and "+ + +ATD" on the 1670/128 side and then quickly answer on the other computer side with: "+ + +ATZ" and "+ + +ATA". Make sure to use the letters in caps, they are important for some modems to interpret them as Hayes commands. The end result should be like the null modem situation described above, with the keyboard input of one computer displaying on the other. From here on in, repeat the file transfer steps outlines in the null modem section.

Some users have reported little luck with this procedure and others use it daily. Apparently some modems are a little finicky about connecting this way. One solution is to go an extra step to convince the modems they are online by simulating normal phone line voltage. The following is a simple project, but like all hardware projects has to be undertaken at your own risk: To simulate a line voltage:

A. Obtain an ordinary 4-wire phone cord.

B. Use a knife to cut open the cord casing somewhere along the length of the cord.

C. Find the RED wire and connect a 9-volt battery to it. For your convenience, you might want to get a 9-volt connector from Radio Shack and solder the wire leads from this to the RED wire and then plug the battery into it.

D. Connect this special cord between your modems. The 9-volt battery will simulate line voltage enough to get the originating modem to dial. If it does not seem to work at first, you might have to swap the ends of the cord so the voltage is on the proper lead for the originating modem.

4) The BBS Bounce. It's slow, it is dumb, and it will not fit all situations, but you sometimes may choose to do the "BBS Bounce" to get data between machines. Upload the file to a local BBS with one computer and then download it back with the other. If the files you are uploading are software of general interest, then you may perform a good deed for your fellow BBS users as a side effect. If they are just your own word processor files, look for a BBS that will let you include an enclosed text file with private electronic mail, send electronic mail to yourself, and then clean up after yourself by deleting your received mail. If you can not do your transfer business in a reasonable amount of time, or if you can not clean up (delete after your transfer is complete) files that are not of any general interest to other users of the BBS, then courtesy demands that you choose another method than this one.

5) Another Bridge To Cross In almost all cases you have further work to do once the data has been transferred over to the PC disk. Here you have to confront the question of file formats. First off you should know about the ground floor of compatibility, the ASCII file (pronounced "Ass Key" or for the more discrete we'll allow "Ash Key"). ASCII is a nearly universal standard of computer data exchange that says the same binary number will represent a "A" or the number "9" no matter which computer created the file. For word processors and databases, and sometimes spreadsheets, ASCII is the file format that can carry your data from one place to another, though you may loose some of the accompanying decoration in this process.

The Commodore 128 uses a now peculiar variation of ASCII called PETSCII (pronounced "Pet Ski"). The only real difference between ASCII and PETSCII is that the code in ASCII for "A" is the code in PETSCII for "a" and so on through the alphabet. So the easy way to see that you have a conflict between different standards is that your file will have all its capitols and small letters mixed up. Converting between the two "SCII's" is a trivial programming challenge and it is a common option to allow this in a disk format or modem transfer program. If you can get your files on the C-128 into a file that shows on your directory as a SEQ file, then chances are you have created a PETSCII file and it can be read and used on the PC after conversion to ASCII. SEQ and ASCII files can contain other information however. Formatting information, such as the codes to create bold face or set margins might be in there or "high bit information", as yet non-standard ASCII information for characters beyond the alphabet, numerals, and punctuation marks such as the character graphics found on the front of the C64's key caps.

A "Pure or Straight ASCII file" is one that has had this information stripped out, a process that often must be done by hand with word processing files, but is often unnecessary in database files which for the most part sensibly avoid confusing data interchange with these non-standard codes. If there is no better choice try creating as pure an ASCII SEQ file as you can on the C-128 before transferring word processor files. If you use a C-128 word processor that normally uses the

PRG format for it files, be sure that you utilize the options they offer to save as a SEQ file (example: using Control Z with ASCII turned on (f1) with Paperclip iii).

6)One Last Hitch Conflicts with carriage returns and line feeds (CR and LF) at the end of lines. A carriage return "chr$(13)" is a "hard return". Wherever it occurs on the right hand margin of the file is the end of the line regardless of the margins or screen width of the program that views it. A line feed "chr$(10)" is a "soft return", and if the reading program is expecting to have the lines broken up by a carriage returns, each paragraph will extend in an infinitely long line that runs off the right hand side of the screen. Some of the disk format conversion programs can adjust the files they produce to have lines end with either.

7) Database Conversion It is usually a matter of creating a SEQ ASCII file with a known character as a "delimiter". The delimiter is a marker between each field in a database record. If for example the delimiter in a database ASCII export is a "," and the database stores only the first and last names of people followed by their phone numbers, three records might look like this:

```
Frank,Hudson,555-1234,
Loren,Lovhaug,555-4321,
Mary Jo,Smith,555-3214,
or
Frank,Hudson,555-1234,Loren,Lovhaug,555-4321,Mary Jo,Smith,555-3214,
```

As long as the record structure in your new database reflects the order of the delimitated ASCII file, the old data will "pour" into the new database "container" in the right place.

8) Getting the Picture With Graphics Files Although programs to convert C-128 word processor and database files to other file formats are unaccountably rare (thus forcing us into using the ASCII "common denominator"), the need for graphics file format conversions is better served. Freely distributable converters exist for Koala, Doodle, GEOS, Print Shop, GIF, and Newsroom graphics files. Once you get the graphics onto a PC disk, the rest is pretty much automatic, since the graphic file can easily emulate the colors and resolutions of most C64 and 128 pictures. There are commercial and public domain converters for the IBM that will handle some of the C-128 graphic formats. The easiest way to transfer graphics 1) use "Mac Attack++" by Joe Buckley to transfer your Geos files into Macpaint format and reading them with a Mac reader on a PC clone or other computer. 2) Using the Geopaint to PCX program from Jim Collette to transfer your Geopaints into a format called PCX which is used by Ensemble on a PC clone.

9) BASIC programs If you wish to use portions of BASIC program code that you have written on the C128 in another version of BASIC, you first need to store your C128 program as a SEQ file on disk. To do this you redirect the LIST command with this simple immediate mode program:

```
DOPEN#1,"filename,s",w:CMD1:LIST:DCLOSE1
```

This example will produce a PETSCII file on disk which can be transferred and converted to ASCII format in much the same way as a word processor file. Some "plain vanilla" BASIC C128 programs will run unaltered on other computers, but of course most useful BASIC programs include machine specific calls that will require alteration on the PC side.

This column kicks off our new telecommunications column. Over the next few issues we will be covering Swiftlink, 2400 modems, 2400/9600 fax modems, 9600 modems, commercial/shareware/public domain BBS and terminal programs for your C-128. Authors and vendors of the above mentioned products (and ones we have not) that deal with telecommunication on the C-128 should send copies of the same for both our files and reviews.

(Changing of the Guard - continued from page 04)

think you will be every pleased to have the territory covered.

We are adding type-in programs and a monthly disk. The monthly disk will contain everything from the programs listed in the magazine (so you do not have to type them in), graphics, games, and more. This disk starts with issue #31 and we are now accepting programs. More on that later in the magazine.

A long editorial, put hopefully an informative one. There are more questions to be answered but I will leave that to the Q&A column where I will provide the answers to various letters we have received over the past few months.

Enjoy your magazine and
the ones to follow in the future.
Yours,

John Brown

The MPS-1270 Printer *by JBEE*

The Commodore MPS-1270 is an ink jet printer manufactured for Commodore by Kodak. The model is basically the same as the Kodak 150 inkjet printer and comes with a print head made by Hewlett Packard. This printer emulates the industry standard 9 pin Epson printer. Unlike other ink jet printers that offer an improvement and refinement over comparable impact dot matrix printers the printouts from this printer are exactly the same as from Epson Fx-80. I made this comparison by comparing how graphics looked on my Epson RX-80 vs. how they look printed out on the MPS-1270. This printer should work with any Epson supported software and hardware. Though I did not try it with the hardware and software fonts for a Xetec interface it worked good with the various Geos drivers I tried, including the multi-pass printer drivers.

The printer made by Kodak that the MPS-1270 is based on is meant to be used as a light weight portable printer for laptops. This is basically the same model sans the battery power. This printer is good looking, small, sleek, very light, does not use much power, and is very quiet. If you had to sleep in a room with a printer working this is the one to pick. Often inkjet printers are much quieter than impact dot matrix printers but this one is exceptionally quiet.

There is a downside to all these advantages. This is not a heavy duty printer. The mean time between failures is about 1/4 of comparable impact dot matrix printers, though it is doubtful under average use that you will wear out the printer in two to three years. If you use a printer heavily, such as printing dozens of copies of your User Group newsletters and labels then this printer is not for you. This is a "personal" printer meant for light personal use.

The cost of consumables is high compared to a impact dot matrix printer. If your ribbon starts to get light and wear out on an impact dot matrix printer you can still keep on printing for some time. If you run out of ink with the MPS-1270, you run out of ink, and can't continue printing. Also, you will find the cartridge harder to find that a cheap ribbon being sold mail order or in a superstore. Like the H.P. Deskjets, you can refill the MPS-1270, though I have not tried it personally.

Paper handling with the MPS-1270 is pretty straight forward. You load the sheet feed paper from the front. With each sheet feed manually you have to pull this and that to make sure the paper is guided into the printer the right way. You would not want to do a long report this way. It does have a built in tractor feed unit so I would recommend only using the printer with pin feed paper.

The MPS-1270 does not come with many features, especially compared to many of the under $200 impact dot matrix printers that can be bought mail order. If your main use for the printer will be to use it under GEOS than most of the features offered by other printers in its price class are not going to be used anyway. The quality, speed, and *time* of the MPS-1270 compared to Epson Fx-80 clones is about the same when using Geos because Geos printing is of a graphic nature, where there is almost no difference between the Epson Fx-80 clones.

If the main requirements in the printer you need are quietness and portability, this small sized printer is for you. If quietness and portability are not your main concerns you are far better off with an inexpensive Star, Panasonic, or Epson 9 pin dot matrix printer.

A Centronics parallel interface such as the Xetec series of interfaces will be needed to connect the printer to your C-128.

If you print more than ten pages a day or use many labels on a regular basis, or if all you need is an inexpensive Epson compatible printer
Rating: C-

If piece and quiet, portable, and a very small footprint are the most important features
Rating: B+

Source for the MPS-1270 printer
Your Commodore dealer or mail order houses

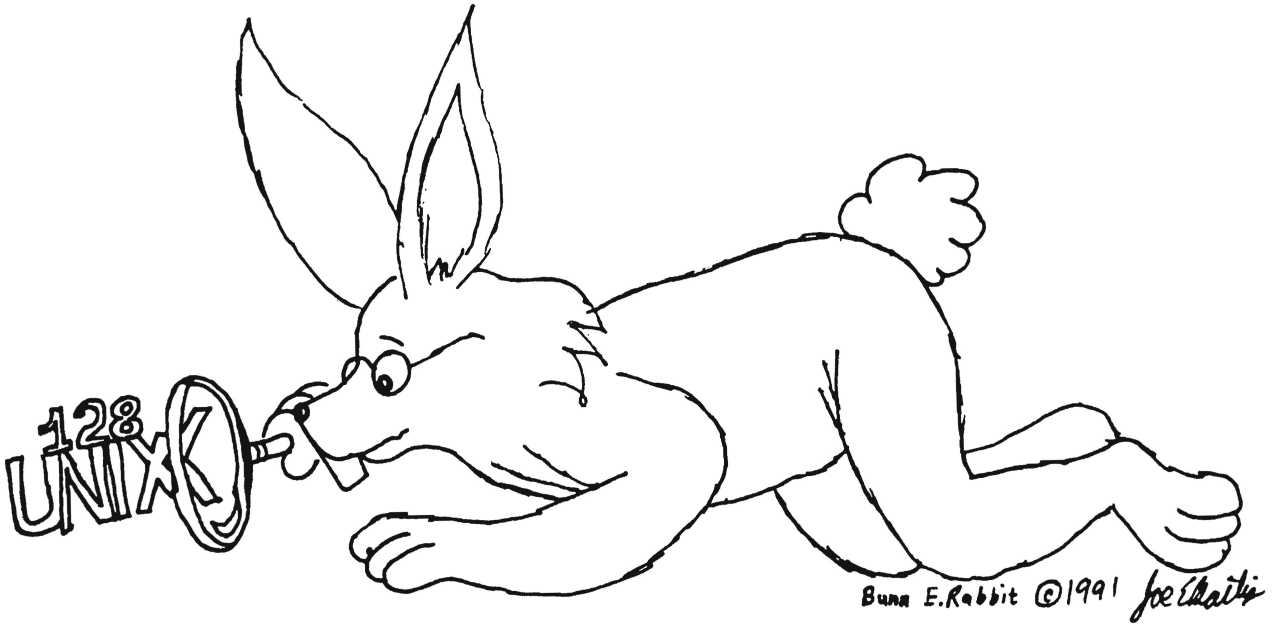
Source for OEM ink cartridges:
Anyplace that sells cartridges for the Kodak Diconix 150 printer, cartridges for the Hewlett Packard Thinkjet and Quietjet printers should work.

Sources for reinking supplies:

Computer Friends
14250 NW Science Park Dr. Portland, OR 97229
1-800-547-3303 (orders) 1-503-626-2291 (support)
Kit # JM3 Cost $29.95 (ink may cost extra)

V-Tech Inc.
2223 Rebecca Hatfield, PA 19440
1-215-822-2989 (orders) 1-215-822-6394 (fax)

They may not have a reinking kit available yet for the Kodak Diconix 150 as this issue goes to the printer so call and ask them. For my Hewlett Packard 500P printer I use V-Tech for the reinking supplies and recommend them for both their price and service.



Unix 128 V3.00 *review by Andy Figueroa*

Unix 128 v3.00 is a shareware program created by Gregg Riedel of Smithtown, NY. Gregg is a pre-med /computer science student at SUNY-Binghamton and expects to graduate in 1993. I am a Unix user, and a user of Commodore 8 bit computers for over seven years. But, enough about Gregg and I.

Available for downloading on GEnie and Quantum Link, Unix 128 v3.00 is intended to give you the look and feel of running a real Unix operating system on your Commodore 128. This article will cover how to get it, the process of installing it so it works for you, and will also describe major features and glitches.

But, first, what is it? If you don't already know what Unix is, it may be a little hard to understand. Real Unix (as opposed to Unix 128), is a multi-tasking, computer operating system developed by AT&T. It is very common in academic, business, and government computing circles. An operating system is the computer program (or collection of programs) that controls the underlying functions of a computer's hardware and its input/output processes. From that perspective, many Commodore 128 users do not really deal directly with their operating system. When you turn on your computer and you see "READY", you are not dealing with the operating system but with BASIC 7.0. BASIC is not an operating system, but a programming language that provides the normal, initial user interface upon power-up. Through BASIC, however, you can get to and control the operating system, i.e. PEEK and POKE memory or issue commands to the Disk Operating System (DOS) residing in the ROMs in your disk drive. However, if you boot up CP/M, and you see

```
A>
```

you are now dealing with the CP/M operating system which really has taken control of the operation of the Commodore 128's hardware, replacing the native Commodore operating system and disk operating system with Digital Research's CP/M 3.0.

So, is Unix 128 v3.00 an operating system? No, not really. Unix 128 is a collection of compiled programs (Abacus's BASIC 128 compiler) that provides a Unix-like user interface. Unix 128, rather than taking over the underlying operations of the Commodore 128, sits on top of the native operating system of your computer and issues appropriate instructions to it, as needed, and as you enter the necessary Unix 128 commands. However, with some exceptions (both good and bad), Unix 128 provides the look and feel of real Unix and may be a good platform upon which to learn about Unix for those who do not have access to university, government or business Unix computers.

Unix 128, like most implementations of Unix, provides the user a command line interface, where the user types in commands, filenames, and appropriate modifiers, in order to make the computer do something useful. From that perspective it is similar to CP/M, MS-DOS, and other command line interfaces or shells.

Getting down to business; how does one get Unix 128 V3.00? For most Commodore users, the practical way to get it is to download the files from GEnie or Qlink.

On GEnie file numbers and names are
#8973 1UNIX128V3.LZH uploader K.Phelan1
#8974 2UNIX128V3.LZH uploader K.Phelan1

These files equal 576 CBM disk blocks and the download will fit on a single 1541 disk. To dissolve these files CS-DOS and the LHA module will be needed. Both of these utilities can be found in the GEnie Flagship$ library (M625;3)

On Qlink the files are named
"unix128v3.0a.sfx"
"unix128v3.0b.sfx"
"unix128v3.0c.sfx"
"unix128v3.0d.sfx"
"unix128v3.0e.sfx"

They were uploaded by JBEE and equal 664 CBM disk blocks. Two 1541 disks will be needed to download all the files. You can find these files through this path Programmers Workshop/128 Programs/Demos & Utilities Library. Or, simply search on the keyword "unix" and these five files will come up. Searching by specific filename and uploader will work too. The SFX files self-dissolve on the Commodore 128 in 40 or 80 column mode.

When dissolved either set of files will fit easily on a double sided 1571 disk or a 1581 disk and will leave plenty of extra room for workspace. You will need an eighty column monitor and **at least a 1571 or 1581 disk drive** to run Unix 128 because of its large size!

Those who have access to a real networked Unix machine can also get them in two very large Lynx'ed files by anonymous FTP to the Milton FTP site. If that last sentence is Greek to you, don't worry, you are partially right. What it is, is Unix network talk. Hang in there, and you'll start to catch on.

Back to the files you have dissolved, I strongly recommend that you label the disk you have dissolved all the files to as "Unix 128 v3.0 - SOURCE -" and write protect it. That will protect your investment in time and effort to this point and you still have the downloaded files to back that up. At this point you really have a choice regarding creation of one or more installed work disks. One option is to copy all the source files to the work disk. The option I recommend to keep disk overhead as low as possible, is to copy all the files except: SHAR.BAS, SHAR.64 and MAKE. The first two are not part of the Unix 128 v3.00 system, and MAKE is an archive we'll dissolve as soon as the other files are copied to the work disk. If you follow the recommended option, once you have copied the files except those mentioned to your work disk, remove the work disk from drive 8 and insert your SOURCE disk. LOAD, but do not RUN the program file MAKE. Remove your SOURCE disk and insert your work disk in drive 8 and RUN the program you just loaded. This writes the remaining Unix system files on your work disk. A good idea at this point would be to back up your new work disk. Otherwise, you may eventually have to do all of this again if things go poorly.

This is a good time to mention a couple of Unix 128 V3.00 flaws. First, it must be run from drive 8, either a 1571 or 1581. Second, it recognizes, but does not use, the RAM Expansion Unit, so you can save a lot of time by not worrying about it. Do not try to use RAM DOS with Unix 128 v3.00. They are not compatible. Also, although multiple drives are recognized, most commands do not successfully work across multiple drives.

To get up and running, type RUN"STARTUP". STARTUP redefines a few characters then loads and runs the main program UNIX 128 V3.00. You will be informed of your system's configuration and prompted to enter the date and the time. A "military" 24 hour time is required, i.e. 1:30.30 PM is 133000. You will then be prompted to log in. At the "login:" prompt you enter a user id and at the "password:" prompt you enter a password. You must use the user id "root" and the password "root". Nothing else will let you in. (When you are better acquainted with the program, you can use the ADDUSER command to add additional user ids and passwords.) After you are logged in you will be presented the current Unix prompt which is set to %. This is the traditional Unix shell prompt. The shell is the Unix command interpreter that reads what you type and passes the instructions to the operating system or to other programs.

At this point, you take the final installation step by entering the command FSCK. This writes the system file to the diskette. You may find as you add or delete files, you will have to run FSCK or the equivalent UPDATE to bring the system file up-to-date so files will show up when you look at the directory with the command LS (list).

You can set the prompt to be anything you want with the PROMPT command, and you can customize the colors, prompt, aliases, and so on by editing the file ".login" and putting in the appropriate Unix instructions. Now, if this "UnixSpeak" is starting to make you leary, perhaps it is time to print out the available documentation. Here is how you do it:

PRINTING OUT UNIX 128 V3.00 DOCUMENTATION

The available documentation is contained in five files: INTRO, MAN1, MAN2, 00UNIX12.NOT and .AUTHOR. Assuming you have a printer connected and online, you can print any text file with the command LPR. Just type LPR INTRO and that file will be output to a printer device #4. A nice feature of LPR is that you can give it a list of files to print out by entering LPR INTRO MAN1 MAN2 00UNIX12.NOT .AUTHOR. This will print the files one at a time till done. It is not fast, and will run about 25 pages. One of the nicest features, and typical of Unix, is the ability to do tasks in the background, and that includes printing. The "&" is used after a command to tell it to run in the background. If you had entered

LPR INTRO .AUTHOR 00UNIX12.NOT MAN1 MAN2 &

you would get your prompt back and the printer would work while the computer was not busy doing something else for you. You can use the command LPQ to see what is lined up to print next. LPQ stands for "line printer queu". LPR, by the way, stands for "line printer".

In any case, when the printing is done, you will have a hard copy of all of the available documentation. The file INTRO contains an introduction to the program and explains the major application programs included, which include the EMACS text editor; TIP, a terminal program; FTP, a file transfer program; NROFF, a text formatter; and AS, an 8510 assembler. An additional application program SPREAD, a spreadsheet, is included but not documented in the file INTRO.

The core of the "operating system" documentation is contained in the two files MAN1 and MAN2. These provide a brief description of how to use each command. Sit down with list and pound away at the keyboard and you will learn how to use Unix.

THE GOOD THE BAD AND THE UGLY

Many of the shortcomings of Unix 128 v3.00 are acknowledged and documented by the author, Gregg Riedel. In a long telephone conversation, Gregg explained that earlier versions, 1.0, 2.0, and 2.5 really did not work at all. They spent most of their time crashing. Writing a Unix "operating system" for the C-128 is an ambitious undertaking, so do not be deceived by the 3.00 version number into believing that this is a well tested, mature program. It is not, but it is slick enough to show that the program has real potential, and Gregg intends to continue to work on the current version to kill known bugs. Some of the shortcomings are due to Gregg's humble C-128 hardware. He does not have a working REU, and his 1571 will not read or write with head #1.

Unix 128 v3.00 has an impressive number of documented commands. These number 108, and will be listed to the screen by typing the command COMMANDS. There are some undocumented commands as well, like WHOAMI. I will let you try it for yourself, it does not do much, but I've found a "whoami" command on all of the several Unix systems I've worked with. Most of the commands are built into the shell, which means they do not have to be loaded from disk. And, most of the commands work just like described. Just a few don't work right or don't seem to work at all, and at least one, the command BASIC crashes the computer.

Sometimes you can recover from a crash by doing a RunStop/Restore and then typing RUN"REBOOT". If this does not work, you can get back by typing Run"UNIX*" as long as you have not done a hardware reset. If you have done a hardware reset you will need to RUN"STARTUP".

The fact that the shell has to reload from disk after many commands have run gets a little annoying, after the novelty of Unix wears off. Gregg knows what the real fix for this is, and that is to stuff all of the operating system code in the REU. He's working on that. Another annoyance is the three commands that are intended to show how much free disk space you have left (DU, DF, and QUOTA) which force the 1571 drive into the 1541, single sided mode. As is, they are not much use with the 1571. But, you can recover from it! One of the slickest pieces of work I've seen in a long time is the command DOS. This opens up a window with what appears to be an MS-DOS command line interpreter. So, if your 1571 gets stuck in the 1541 mode, type DOS, and then at the A> prompt enter CHKDSK and you will get a nice report on your drive and disk, and the drive will be reset back to 1571 mode. Spend some time in this DOS window if you like messing with "messy DOS".

Three Unix games are included. These are PUZZLE, MAZE and WUMP. They work fine. Try them. They will make you smile.

The serious applications, EMACS, SPREAD, and TIP are very artfully done. They can be made to crash, but they are well implemented from a user interface perspective, and are very unix-like. A word of warning about EMACS: if and when you use EMACS to edit your .login file, be sure to go to the end of the file and remove the carriage return at the end of the last line. EMACS will add a carriage return to your file every time you load it, and if you do not delete the trailing carriage return from your .login file before you resave it, the system will hang up when trying to boot or return to the shell from a program. Well, that's what backup disks are for anyway. It is a fatal error, and one that is so small.

For both EMACS and SPREAD, if you type CONTROL H you get a nice help screen. For SPREAD, that is all the documentation there is. Gregg said he built SPREAD from the ground up in an afternoon. It works! I only found one glitch with the spreadsheet SPREAD while working with ranges. I guess he needs to spend a couple of more minutes on it.

Well, I don't want to ruin the thrill of discovery for you by telling you all the little things; you will have great fun finding them out yourself. But if you establish user ids and passwords for your friends, you can have fun sending "Electronic Mail" to each other using the MAIL command. Yes, MAIL works well.

Finally, when you are done with Unix 128 v3.00 for a session, be sure to type LOGOUT. That's what all polite Unix users do.

Note: If you'd like to encourage Gregg Riedel to continue to squash bugs, develop enhancements, and further versions, give serious consideration to sending him the $10 shareware fee. Collectively, we will all be glad you did.

Original basis for this article and the credit for the upload to GEnie goes to "K.Phelan1" ! If he had not tracked down the rumors concerning that this file existed and had not gone out of his way to get the file and then upload it to GEnie, which made it widely available, you probably would not have read this review!

Unix 128 V3
Rating : a solid B
Cost $10.00

Gregg Riedel
20 Ellbert St
Smithtown, NY 11787
At Suny-Binghamton
CONSP24 @ BINGVAXU.CC.BINGHAMTON.EDU

GEnie 1-800-638-9636

Q-Link 1-800-827-8444

Parsec Inc.
Disk number 8DU03 contains the .SFX files that were uploaded to Q-Link and the files need to be dissolved onto a 1571 or 1581 disk before use.

Price and Progress Report

(by Loren Lovhaug/Frank Hudson, edited by JBEE)

Things sure do change in the furious computer industry don't they? When Loren founded Twin Cities 128 back in 1986 the computer industry was in turmoil. Commodore's future was in doubt. Apple was trying to sell a lot of little black and white MacIntoshes, while IBM had just introduced a model based on the Intel 80286 chip.

In those heady days, we all had Commodore 128 computers, a computer that the industry was proud to pay no mind to. Unenlightened individuals that we were, we only noticed that the 128 was affordable, was an easy move from our previous C64 computers, and did the things we wanted to do with it. No way to buy a computer I guess.

Into this vacuum of information, Twin Cities 128 was launched, a frail if intelligent probe to see who else cared about using this computer. The response from you, its dedicated readers, was enough to keep it going, and even growing over the years, and this column "The C128 Price and Progress Report" was a constantly updated marker of the Commodore times. With this issue, the editorship of Twin Cities 128 passes to John Brown of Parsec Inc., another dedicated Commodore enthusiast. It seems appropriate, then to at least briefly mark the C128's current station one more time.

Prices for basic C-128 components have never been better, though of course at this time they are most often "used" prices. Availability is still holding up fairly well, assuming that you are in a metropolitan area with a good concentration of other Commodore users or can tap into the online used equipment market.

The peripheral situation is also reasonably healthy, and in the area of hard drives and alternative mass storage, Creative Micro Designs offers a 1991 lineup that contains the best C-128 options ever and in some cases the best options on any computer.

Telecommunications has always been close to Twin Cities 128's heart, and excellent software like Dialogue and Desterm, the CMD Swiftlink, and the ever falling prices of modems make getting o nline with the C-128 more pleasurable than ever before and networks like GEnie have gone to 9600 baud.

From you we hear that many of you <u>are</u> staying with your machines, adding new software and hardware, and the most important peripheral of all, new knowledge. So we're turning the over the controls to John, who will continue to report the C-128's progress.

We wish him the best.

Masterpaint Review *by Frank Hudson*

The world of C-128 graphics can be easily divided into two hemispheres, the GEOS half and the BASIC 8 half. Each side of the world has its advantages and limitations, but on balance my judgement has always been to favor geoPaint as my C-128 paint program of choice. Though I was willing to concede the superiority of the BASIC 8 graphics file format, especially in the area of color map freedom, the BASIC 8 paint programs were all too often clumsy to use. 1990 saw the introduction of two new BASIC 8 compatible paint programs which substantially advanced the cause of BASIC 8 graphics. I Paint and Master Paint.

The author of Master Paint, Roger Silva, has distinguished himself as one of the better artists working with the C-128. It stands to reason then that the paint program produced by an artist would show a good understanding of "user interfaces" - the tools and methods used by a program to translate what you want to do into what appears on your screen. In this area MasterPaint does not disappoint. On first bootup MasterPaint greets the C-128 artist with a familiar looking icon toolbox on the left side of the screen. To anyone who has used a similar product, the icons clearly lead the artist/user to the action they want to perform. Like other good C-128 paint programs, MasterPaint works best with the C-1351 mouse or it's equivalent, though a joystick is also supported. Both buttons of the 1351 mouse are used intelligently by MasterPaint. The left button is the "action button" while the to-often ignored right mouse button serves as a menu operator, bringing back the toolbox or stepping back through a series of choices, more or less duplicating the function of the keyboard's ESC key. Pressing the right mouse button while over the toolbox swaps in a second set of toolbox icons. The farthest left margin of the toolbox is devoted to a well thought-out palette requestor. The tools available from the MasterPaint toolbox generally duplicate all those found in BASIC 8 compatible paint programs, including the Rylander solids, BASIC 8 format scalable fonts, BRUS. clips and patterns.

However, MasterPaint adds additional tools such as placeable Rulers, an Eraser, and an Undo. The last of these, Undo, was previously unavailable in any other BASIC 8 compatible paint program and its presence is very welcome. MasterPaint's Zoom or "Fat Bits" mode offers another unique feature: it's moveable. The area being magnified can be scrolled around the screen without exiting the Zoom mode. The 24 page manual is cleanly printed, illustrates the icons and does a good job of explaining these MasterPaint features. As a result of the good design reflected in these icon-based tools, MasterPaint is the easiest to use BASIC 8 paint package I've ever seen.

There are certain limitations which may keep it from becoming the only C-128 paint program you'd ever want to own. MasterPaint uses only one of the BASIC 8 color standards, the 8 X 2 color cell format. As a result, 64K VDC RAM is required and the more memory efficient and printer-oriented monochrome modes cannot be created in MasterPaint. Fill operations, as in every other BASIC 8 compatible program, work very slowly. The MasterPaint mouse pointer, like many other 80 column pointers, is a bit jumpy and sluggish compared to the pointer sprites available in the 128's 40 column mode. Some elements of the BASIC 8 environment are beginning to look outdated as well, particularly the low resolution of the bitmapped fonts which results in objectionable "stair-step jaggies" whenever they are enlarged. No Commodore 128 program offers pixel by pixel color freedom, and even though MasterPaint uses the best color resolution possible in a non-interlaced 128 display, C-128 artists still need to understand the nature of the color cell system to produce vibrant color images. Masterpaint's placeable rulers and excellent moveable zoom window are a good aid to working with these color limitations. Many of these limitations would be difficult for any programmer to surmount, and given it's ease of use and low cost, MasterPaint is well worth considering. For working with color in the C-128's 640 x 200 hi-res non-interlace mode, MasterPaint is now the best program available. A better Zoom, an unduplicated Undo command, good color management tools, and a fine user interface make it a noticable improvement over other programs that use this mode.

Editor's note: In most respects Masterpaint is a superior paint program, but it is not "cutting edge". With the same flaws as other Basic 8 paint programs, ONLY ONE low resoultion SCREEN SIZE (640x200), one color resolution (8x2), and it is an *uncomplied Basic program*, so... If you have the Basic 8 programming package and you can get by with it I would not upgrade, except to Ipaint. If you have Geopaint but hate the 40 columns and color cell limitations then it is a good upgrade. If you are using any of the Free Spirit paint programs, format the disks if you haven't already, and rush an overnight order to Silvasoft.

Rating: B-

Masterpaint $28.50 (includes shipping)

Silvasoft
P.O. Box 1006
Charlotte, VT 05445

RAMDrive is a fresh new approach to RAM expansion for the Commodore 64 and 128 RAMDrive is a self contained battery backed cartridge available in three standard capacities. RAMDrive incorporates RL-DOS, an ultra-fast, versatile operating system similar to the HD-DOS found in CMD hard drives and RAMLink. This DOS allows various types of partitioning along with compatibility and speed unsurpassed by previous RAM Expansion systems. RAMDrive is particularly useful to users in search of a portable battery backed RAM unit.

RAMDrive w/512K $199.95 **RAMDrive w/1 Mb $249.95** RAMDrive w/2 Mb $299.95

Shipping: US **$7.50** (UPS), **$14.00** (2nd day), COD add $4.00 Canada $15.00

RAMDrive designed and manufactured by Performance Peripherals Inc

RAMDrive™

High Performance Battery-Backed RAM Cartridge

HD Series™ HardDrives

The Standard in 64/128 hard drives

The CMD HD series of hard drives for the Commodore 64 and 128 have been designed to provide the highest level of compatibility and performance These drives incorporate the latest in SCSI technology with advanced circuit design to bring you the most features at an affordable price CMD drives allow you to achieve the higher level of productivity you require to get the most from your existing software - today! HD Series hard drives are ideal for BBS, GEOS, word processing, database, accounting, graphics and many other applications commonly used with Commodore 64 and 128 computers

HD Series: HD-20 $499.95 • HD-40 $659.95 • HD-100 $999.95 • HD-200 $1299.95

Add-on drives to expand HD Systems: SD-40 $549.95 • SD-100 $899.95 • SD-200 $1199.95

More than a replacement for the deskTop, the gateWay offers a unique streamlined approach to file management under GEOS With a resizable FilePad and proportional slider the gateWay makes it easier to get where you're going. While preserving the functionality of the original deskTop gateWay adds powerful new features like task switching, three drive support full CMD device support extended RAM disk support and allows you to boot GEOS from any supported device type At CMD [illegible] powerful new hardware development - we're creating new ways to use it

gateWay 64 $29.95 • gateWay 128 $29.95 • **Both $44.95** • Shipping: $5.00

Note. The gateWay requires a Commodore 64 or 128, a mouse or joystick and GEOS 2.0 for operation

gateWay™

A Whole New Way of Working with GEOS

RAMLink™

Power Backed REU Interface and Expandable RAM Disk

RAMLink is a multi-purpose hardware interface with a power battery back-up system for Commodore 64 and 128 computers. RAMLink is designed to overcome the limitations of existing REU's RAMLink delivers the performance and compatibility lacking in past REU's by incorporating CMD's RL-DOS and advanced features into a unit which allows most software to utilize Commodore 17XX REU's, GEORAM, and CMD's RAMCard as if they were a disk drive RAMCard installs internally and can be used alone or in conjunction with an external REU for a total capacity of 16 Mb. RAMLink also provides a pass-thru port for use with other cartridges and a parallel interface to the CMD hard drive. RAMLink is ideal for users who require larger RAM disk storage

RAMLink (without RAMCard)	**$179.95**	RAMLink Battery back-up unit	$24.35
RAMLink w/ RAMCard (0 Mb)	**$219.95**	Parallel Cable for CMD HD	$14.95
RAMCard w/ 0 Mb (if purchased separately)	**$59.95**	1 Mb & 4 Mb SIMMs	CALL

Shipping

US **$10.50** (UPS) **$18.00** (2nd day) COD add **$4.00**

Canada **$20.00** Foreign Contact CMD

CMD / Dr. Evil Labs *SWIFT*Link-232 Serial Cartridge

A Real RS-232 Serial Port for the Commodore 64 or 128

SWIFTLink-232 Cartridge $39 95 SWIFTLink Cable $9 95

Shipping: US **$5.00** (Cartridge), **$3.00** (Cable), **$6.00** (Both) Canada add $4 50

CMD / Dr. Evil Labs SID SYMPHONY Stereo Cartridge

The Realism of Stereo Sound on your Commodore 64 or 128

SID SYMPHONY Cartridge $39 95

Shipping: US $5 00 Canada $9.50

JiffyDOS™ V 6.0

The Ultimate Disk Drive Enhancement System

JiffyDOS 64 or SX-64 - $59.95 JiffyDOS 128 or 128D - $69.95

Each system includes computer Kernal ROM(s) and one drive ROM

Please specify computer and drive model numbers and serial numbers when ordering.

Additional Drive ROMs - $29.95 each

OTHER CMD HARDWARE & ACCESSORIES

Lt. Konnector	- Allows use of Lt. Kernal® drive mechanism with CMD HD System	**$19.95**
Disk Drive Power Supply	- Heavy Duty Replacement for 1541-II, 1581, & CMD HD	**$49.95**
JiffyMON ML Monitor	- 64 Mode ML and disk drive Monitor (requires JiffyDOS)	**$19.95**
Serial Cables	- Quality 36" Commodore Serial Cables (6 pin DIN male to male)	**$3.95**

Shipping (U.S.): Lt Konnector $5 00 Power Supply $6 00, JiffyMON $2 00, Serial Cables $2 00

Ordering Information and Shipping Charges

HD and SD Hard Drives: Continental US $25.00 per drive (UPS ground), $35 00 (2nd-Day) $45 00 (Next Day) Canada $50 00 (Airmail) COD to U S only $4 00 add'l charge Foreign prices Add $100 00 to U S. Retail Price

JiffyDOS: Add $5.00 per order (UPS ground), $9 00 (2nd-Day Air) plus $4 50 for APO FPO AK HI and Canada, or $15 00 for overseas orders No add'l shipping if ordered with any hard drive COD's add'l $4 00

Tax: MA residents add 5% sales tax.

Terms: We accept VISA, MasterCard, Money Orders C O D and personal checks (allow 3 weeks for personal checks to clear) Credit card orders provide the following Card holders name billing address, home/work phone, card number, expiration date and issuing bank

• WE VERIFY ALL CREDIT CARD INFORMATION AND PROSECUTE INDIVIDUALS ATTEMPTING TO PERPETRATE FRAUD •

Note: Prices and specifications subject to change without notice. Normal delivery is 2 to 4 weeks, backordered times may take longer GEOS, GEOS 2.0 GEOS deskTop, GEOWRITE, GEOPAINT GEOS CONFIGURE and GEORAM are registered trademarks of Berkeley Softworks, Inc., CP/M is a trademark of Digital Research, Commodore 64, Commodore 128, Commodore SX-64, Commodore 128D Commodore 1700 1764 1750 and Commodore RAMDOS are registered trademarks of Commodore International Amiga is a trademark of Commodore-Amiga. Macintosh is a trademark of Apple Computer, IBM is a trademark of International Business Machines. Lt. Kernal® is a trademark of Fiscal Information Inc. RAMLink, RAMDrive, CMD HD Series and gateWay are registered trademarks of Creative Micro Designs, Inc.

CMD

Creative Micro Designs, Inc.

15 Benton Drive, P O. Box 646
East Longmeadow, MA 01028

ORDERS ONLY: 1-800-638-3263

BBS: 1-413 525-0148 • **Questions/Support:** 1-413-525-0023 • **FAX:** 1-413-525-0147

Office Hours 10 am - 5 pm Monday through Friday • Closed weedends and holidays