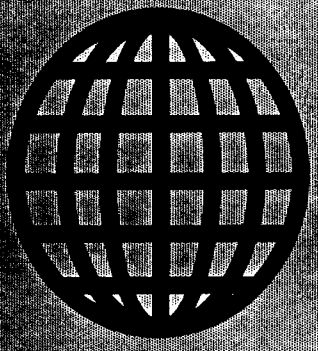
Providing Support Around The World



The Computer Journal

Issue Number 74 July/August 1995 US$4.00

Antique or Junk?

Small System Support

Moving Forth Part 8

The AMSTRAD PCW Now

Real Computing

Palmtech CPUZ180

Dr. S-100

Mr. Kaypro

Disk I/O in Forth

Centerfold - S-100 Power Supplies

The Computer Corner

ISSN # 0748-9331

TCJ - For Having Fun With Old Computers!

The Computer Journal

Founder
Art Carlson

Editor/Publisher
Bill D. Kibler

Technical Consultant
Chris McEwen

Contributing Editors
Herb Johnson
Charles Stafford
Brad Rodriguez
Ronald W. Anderson
Helmut Jungkunz
Ron Mitchell
Dave Baldwin
Frank Sergeant
JW Weaver
Richard Rodman
Jay Sage
Tilmann Reh

The Computer Journal is published six times a year and mailed from *The Computer Journal*, P. O. Box 535, Lincoln, CA 95648, (916) 645-1670.

Subscription rates within the US: $24 one year (6 issues), $44 two years (12 issues). Send subscription, renewals, address changes, or advertising inquires to: *The Computer Journal*, P.O. Box 535, Lincoln,CA 95648.

TCJ The Computer Journal
Issue Number 74 July/August 1995

EDITOR'S COMMENTS

Welcome to a regular feature on junk...oops antique computers. That slip really sets the stage for starting to discuss the idea of what makes an antique computer. That is the special theme this issue.

After a few letters to the editor (we had plenty more letters but no room) you get my main article on "Antique or Junk?" It is a topic starting to catch more interest in other media, but one we have really been dealing with for many years now. I try to explain the reason not much interest exists for PC clones as antiques. But then you read what I have to say and drop me a note on your personal experiences collecting.

Next up is Chuck Stafford and his Kaypro column. In this issue we have a means to hook those much larger composite video monitors on to your Kaypro. Elliam Associates is a good source for old disks and Amstrad support. I guess he read my mind and dropped me a short article on AMSTRAD. Seems they are about to build some new ones, and here everyone thought CP/M was dead.

Ronald Anderson would still like to hear from readers, but that hasn't slowed down his explanation of C or assembler. Between his letter and his article Ron can sure output some good material. I hope many of you have dropped him a note to say how much you enjoy his work. Now all we need to do is get him on internet so it will be easier to pat him on the back for a job well done.

Speaking about jobs well done, Herb Johnson has been getting lots of fan mail and questions about S-100 systems. Seems he has been helping lots of you keep those older classic beasts running, good going Herb, and oh yes GIDE coming soon, all of course in his column.

Herb's column got me thinking about power supplies, and so I thought it was about time to feature S-100 power supplies as a center fold. Since they are simple, I gave you a couple to compare, some wiring information, and some hints on modifications. Hopefully it should answer any lingering questions you might have.

AMSTRAD is not alone in producing a new CP/M machine, we also have Claude Palm of Palmtech making his CPUZ180 machine. Mr. Ramon Gandia of Nome Alaska has under taken the task of being his US representative and prime tester. The results of his action are all summarized in his article. You can tell that Ramon both enjoyed and worked hard testing the CPUZ180, all for our benefit.

Moving on to bigger (in bits) projects is Rick Rodman and changes in Minix are a foot. Seems a new version is about to be released. Linux is still catching fire in more places as well. All this in Rick's column.

Small systems need small languages that can give you the power to do your work. I have always found Forth ideal for small systems, and so has Walter Rottenkolber. Walter returns this issue with a several part report on the Forth file system. His introduction and FIG Forth explanation are in this issue, F83 file support in our next issue. Now Walter has also tried to focus on questions that beginners have in starting to understand Forth, thanks for starting with the basic's Walter.

Brad Rodriguez's 6809 Forth is here in it's entirety and up on Genie as well. Now this is intended for his Scroungemaster II boards, but I am sure a few of you will be able to get it running on a CoCo or Flex machine without much trouble. By the way, please let me or Brad know about the port and for sure send me a short article so others can see how you did it. I just noticed he uses only 8K of RAM and 8K of ROM, nice going Brad.

Well that leaves our groups section, which hasn't changed much lately, although I still get lots of requests for hardware and software vendors on these old systems. One way to get more subscribers to *TCJ* would be if I could increase the list of vendors still selling older items. Looking for advertisers that sell the missing disk drives for your old unit, I think is a number one reason many subscribe to *TCJ*. Got something to sell?

Last as always is my Computer Corner, for far too many times. But have no fear I seldom run out of things to say, and this issue is no exception. A little bit of everything in this issue and more explanation of the format clarification.

INTERNET & WEB

Well the internet connect is working, and the *TCJ* web page has been looked at by many of you. I have noticed some needed changes and would like your comments as well. It has turned out a bit long and the changes I want to make will move most of the other sources on to a web page of it's own. I may also drop much of the HTML formatting so SLIP users are not bothered by lots of funny <xx> stuff.

HTML is really not that hard to master, and mine only took a few days to do. For simple "this is me" and "here is where you should go" pages, just download someone else's, save it off, and use it as a guide for your own. Also it helps to load your web browser, I have to actually use it, then turn off the internet link, and call up my files for review. Switching between your editor and viewing it in the browser, you will have, in no time, a good looking home page. Give our page a try.

Thanks for reading *The Computer Journal*. Bill Kibler. E-mail tcj@psyber.com or http://www.psyber.com/~tcj.

READER to READER

Letters to the Editor
All Readers
MINI Articles

To: Bill Kibler From: Ron Anderson Subject: #73 & Misc.

I agree in general with Frank Sergeant in his "I Hate Change" column. The hardware and software manufacturers have a great thing going. The hardware gets faster via new processor designs and immediately the software people slow the old hardware down to intolerable levels with their new software offerings that make it *necessary* for a user to upgrade to a new and faster computer. As an aside, the programs get bigger so the hard drive suppliers can cash in selling their new 1 Gbyte drive to replace the 420 Mbyte one that replaced the 170 Mbyte that replaced the 40 Mbyte ... ad infinitum. I hated WordStar when it was running on a Z80 system here at the company, but a bit of experience with McroSoft Word convinced me I hated it even more, so I bought WordStar about version 3.0 for DOS. I liked it and then found a package called Emulaser that let me run WordStar and use a nice set of fonts supplied with Emulaser. I had to upgrade to WordStar 7.0 to do that. I found WS for Windows v. 1.5 a couple of years ago for a bargain price and it worked with TruType fonts under Windows and I could throw out Emulaser which worked fine but was very slow. I had upgraded from a 286 to a 386-40 and operation was still a bit slow. I upgraded to WS 2.0 for Windows and a 486SLC66/2 that is 2.5 times faster than the 386-40, but with the latest WS, it was slower yet. I found that I needed to run Smartdrive in order to speed it up. Problem then is that I need 8 Megs of DRAM etc. etc. etc.

The only thing I can't argue with is the result. This is being edited in graphics mode, and the 486 is fast enough to keep up with my fast keystrokes. I have a set of nice fonts, though I generally use only a few, and I can print a letter that looks like typeset on a Canon BJ-200e that I have in my office (this letter printed on it). My Panasonic KXP-1123 24 pin dot matrix does almost as good a job at home.

Reason for my writing however is primarily unrelated to the above. Where on earth did you get your prices for DRAM in your $10 PC article? JDR Microdevices in San Jose CA has 256K SIMM modules for $17.50. My local computer store has 1 Meg versions for $39 each (not $100 as you stated in your article). The 4 Meg modules (30 contacts) sell for about $140 in that same store (70 nanosecond version). I enjoyed your article very much. I think most people get stuck when it becomes necessary to troubleshoot a system. Your advice will be appreciated by many readers.

Second reason for writing is to say that I received a letter from Dr. Alen Gordon in Miami, worried about TCJ and your problem of insufficient income from it. He had a number of suggestions including increasing the subscription rate. You seem to indicate you don't think that would be feasible. My thought is that there are some of us out here who, though not independently wealthy, are better off than a poor College student. Some of us might be willing to subsidize your publication a bit. Charitable organizations, sometimes have various levels of "Membership". A Sustaining Member has to give X dollars to be in that category. A Supporting Member has to give X/2. A Contributing Member has to give X/5 etc. All benefit from getting their names published by category once a year! Somehow I think this would work. Incidentally I've done my bit. I was talking to Jack Crenshaw last night on the phone and mentioned TCJ. He was surprised to find that it is still in existence. He asked me for subscription information and said he would subscribe immediately. I sent him a photocopy of your subscription coupon from issue 73.

Lastly, I realize you are running out of my material again. I have gotten bogged down in thinking about 6809 Assembler code projects that would be more complex than so far but not so complex as to be a month's worth of evenings work that I don't have the time or the inclination to do. I'll check the latest column against my hard drive and send you at least a couple more. I've finished the C course but have gotten bogged down in the Assembler area as I just said.

Realizing you are already overworked, I don't want to burden you with more, but if you have any reader feedback on either the Assembler or the C series or my general chatter, I'd appreciate a copy. Thanks for the pitch for some direct feedback in #73. I had been corresponding with Frank Wilson, but he seems to have lost interest on finding that I don't have a CoCo in my collection.

A fellow named James Ogden has been asking for and getting help writing BASIC programs to analyze the State Lottery in Maryland. The usual stats stuff to find the hot number and the cold one and then bet on the hot one on the theory that it will continue to come up more often than expected or on the cold one on the theory that it is "due" since it hasn't come up in a while. I've expressed my opinion that all this is nonsense. My son-in-law to be (Aug. 12) calls the State

Lottery the Stupid Tax and he says it works quite well because the dumber you are the more you pay!

Alen Gordon has been picking over my supply of software and documentation for FLEX and SK*DOS. Of course I haven't given him any SK*DOS software, more in the area of advice on writing his own utilities using the well spelled out SK*DOS interface in the manual.

John Fiorino in Brooklyn has also acquired copies of all my old FLEX documentation and the hardware documentation for the old SWTPC machines. I enjoy corresponding with all of the above. I've always found it to be a relaxing time to sit down and spend an evening writing a 7 or 8 page letter to someone. I once received an 11 foot fanfold letter from Jack Crenshaw (12 pages each 11 inches long = 11 feet). Last Christmas I received a letter from him that had a 32 page count! That beat anything else I've ever received or written, but then Jack is a contributing editor to Embedded Systems magazine. He must type even faster than I. I clip along about 80 WPM when I don't stop and think about it too much. If I try to take a speed test I generally do far worse than that. I've decided that typists must group motions in order to get real speed. I type small words in one thought, i.e. words like the and and. I'm sure I group common combinations also. Since the group "ion" occurrs so often, I do that as one motion. My evidence is that I can't type the word "ratio". I always type "ration" and have to backspace to delete the "n"!

Well, enough chatter. I really enjoyed the articles about the XT. When we first got one at the company we laughed at it. It was about half the speed of our 2 MHz 6809 system running FLEX. According to Norton Sysinfo, however, our present 486SLC66/2 is 100 times faster than the XT and the math using a fairly slow math coprocessor is about 500 times faster! Our latest Pentium 90,systems at work comes in at about 300 in the Norton sysinfo test. I have several things to say about speed in some of the coming columns so I won't get too repetitive here. Yours truly, Ron

I hate seeing your letters, Ron, since it means I am falling down on the job and not getting back to you soon enough. To start I have always used WordStar, since I got it free when I worked for them. But I have long since learned that everyone is different and you need to try a few processors before finding one that matches your fingering so to speak.

You sure hit it right on the head, but I call it the automobile salesman approach to selling computers. They keep you buying new cars with this or that gimic and it seems to work with computer speed the same way.

Ok prices of RAM/SIMMS, I must admit the article got clipped for space and lost some of it's understanding. I have the same prices here, but all my machines are DX's and as such must use four SIMMs of any size. So I would have to use 4 of the 256K SIMMs or about $100 maximum, and same with the rest. In fact I can do better than you on the 4 meg SIMMs, my local wholesale price is $450 for the four I need to go up to 16 meg. But read my Computer Corner and see that you really need 32 megs if you use Windows much (talk about getting jabbed with a sharp stick! Oh the pain!)

*TCJ is about in the middle for subscription pricing, higher than the no-information PC rags, but not as much as some of the code packed insider magazines. I will consider the substaining member idea, but a better suggestion was having readers donate a subscription to your local library. In California, almost all libraries have cut all services if they are open at all, and would love to get free magazines for their readers. I think the best way to help TCJ out however is just spreading the word. I think too many people think we are rolling in the money and can do it all by ourselves. I have started working the internet usegroups, but lack of time prevents a full time presences. If more CP/M and FLEX/SK*DOS users just added "see old system support at http://www.psyber.com/~tcj" to their signature, I am sure our ranks would double quickly.*

Well it sounds like you have been getting feedback from readers. The most I get is an occassional "how I love Ron" or "I'am really glad you have been able to get Ron to write for you - Thanks!" These usually are scribbled over the renewal form and too short to send you. I know you have helped our readers out much like all of TCJ's readers. But when they say "when TCJ comes, I sit down and read it from cover to cover", it sounds to me they love all the writers including you!

A few last quickies, Frank Wilson is still interested, he was un-employed till recently, so I think his play time dropped off a little. I once was interested in a Lotto program, until I discovered that California uses three sets of balls, all with their own characteristics, and never let anyone know which set is being used when. Basically that blows any tracking program and I think is what most all the states do, they just don't tell anyone that.

As to article slant, I think some of our embedded users would like to know more about your companies use of 6809s, and using 8 biters for other than wordprocessors. Thanks Ron for all your work! Bill.

Dear Bill,
Sorry for the delay in renewing my subscription. Back in April I decided to buy a brand new Micromint SB180LO while they where still available. This cleaned out my U.S. bank account so I had to wait for more U.S. funds to come in.

I am currently using a Morrow MD16 with two Seagate hard disks and running Z-System. As soon as I get the SB180 up and running, I'll retire the Morrow. I use the computer for both business and personal. Business use is primarily with dbase II. The Morrow is a bit slow with the larger dbase files so I decided to upgrade to the SB180.

It would be nice to see more 'user' oriented articles in the magazine. Most of the articles seem to target the 'hobbyist/hacker'. For instance, there has been a lot of talk recently about the Z80/IDE interface but never a clear non-technical

explanation of what it is, who would be able to use it, what is IDE, etc. I thought at first I might be interested in it but have come to the conclusion that with the SB18O the SCSI serves the same purpose.

I suspect over the years that the 8 bit community has lost a lot of people who simply didn't know where to go or what to do when something went wrong with their machines. The only option for most people would be to make Mr. Gates a little bit richer.The reason I've stuck with what I've got is because I'm stubborn. I refuse to buy something new simply because it's new and somebody tells me I should. I like Z-System and I've yet to see anything I like about Messy-DOS. If you could reach more of the people who are simply users and offer, them the kind of help they need to keep their machines running it might help your subscription problems a bit and keep more people using sensible machines.

I think your idea of providing a list of vendors is great. The biggest problem I'm having with the SB180 is finding drives. Your list should include possible sources for generic hardware such as drives, power supplies, etc. It would be nice if you could find someone to do some short pieces on generic type hardware for the none technical types who find themselves needing to replace a drive or power supply or whatever - different types of power supplies, different types of drive interfaces, etc.

The Morrow has been running for ten years and with the new SB180 I should be good for at least another ten years. My biggest worry is that a drive will break down and I won't be able to replace it so I'm going to start stock-piling drives. I hope you'll be able to solve some of your problems and keep the magazine going for as long as there are good computers still running. I'd hate to think that I was all alone in a completely 'blue' computer world.

Yours truly,
Gary Oliver, Canada
(Genie=G.Oliver12)

Thanks for the letter Gary. Please keep us informed about your progress with the SB180 and you are correct that IDE is just like SCSI on your new system. IDE however has gotten cheaper than SCSI for a same size hard drive. Either one will give you more hard disk storage than you will know what to do.

I have gotten my writers to give more intro information, and in fact got Tilmann to pretty fully explain the IDE interface. It took several issues and you might have missed them, see the Back Issues section for the indexes. Dave Baldwin was doing some real basic beginner type of articles, but he is busy on other projects right now (he does promise to do basics computer topics soon.)

So far I think most users have been able to find spare parts for their systems, but that is those in the United States. You Canadians and others outside the US proably aren't so lucky. We have had some comments about JDR and Jameco as good sources for new parts. For old parts, get a copy of Nuts and Volts, it is usually free down here, but sorry to say would cost you $50 a year to get in Canada since it must go First Class. If our readership was higher, I am sure I could get some of the used dealers to advertise, but till then I will have to find some alternatives for those outside the USA. Thanks again Gary. Bill.

Gentlemen;

I have, a problem with my Tandy TRS-80 computer. I will be happy to pay for 6 issues of your magazine if you can find the trouble -(or your free issue unlocks the trouble for me).

I purchased the computer (second hand) at the beginning of the year. I thought it a very good deal. I purchsed the computer, 2 disk drives, the printer and a dozen floppy disks at a very reasonable price. The lady told me that it wouldn't run disks. I thought there couldn't be much wrong and bought it.

After I got it home I went to Radio-Shack and purchased a new operations program for it. I put the program in the machine and turned it on. The Tandy logo came up and asked for the date. I wasn't told that I would have to put in an 83 date and tried several times to put in a date of 95. After doing this a few times the screen went blank and I havn't been able to get the disks to do anything but turn on since (no data in or out).

This machine seems to do everything in Basic, but-,when I try to run a Z-80 program it locks up. I believe this is my trouble. The disks I purchased from Radio-Shack are probably written in Z-80 language and the computer won't except it. I wrote Tandy and told them I thought the Z-80 chip was bad, but they answered and said if it was bad Basic wouldn't run.

I have run programs to reset the I/O buffers, make the machine run at 4 megabites or 2 megabites nothing seems to help. I have put several programs on tape (from the instruction manual that came with the computer) and they load and run in basic, but not in Z-80. There's a program, to display the clock that I use in Basic to display and turn off the clock. When I try it in Z-80 the computer just locks up.

Hope you can help me I am respectfully yours,
Robert Sisson.
84575 Second St.
Hartford, MI 49057

The computer has the following numbers. Model# 26-1067, Serial# 033646, TRS-80, 84k-RAM. The printer also works.

Ok Robert it is hard to know where to start. Your system seems to be working just fine, what you don't understand is that TRS-80's have their own special way of doing things. You have two operating systems possible, from Radio Shack they use TRSDOS, and all programs must be designed for it. You can get CP/M, but without a special modification board, the CP/M programs must be specialy assembled to run at 4000 hex and not the normal 100 hex. You can only run

Continues on page 37

Special Feature

Classic Support

What to Collect?

Antique or Junk?

By Bill Kibler

Welcome to our feature topic of Antique and collectible computers. *The Computer Journal* has a new structure that features or focuses our attention and pages on one of three topics. This issue's topic is Antique systems and in this article more specifically what makes a system an ANTIQUE.

Junk or Antique

North of San Francisco is Santa Rosa, and backing up to Highway 101 that goes through Santa Rosa is a sign advertising a local antique store. In rather large letters it says, "we buy Junk, but sell Antiques." That statement pretty much sums up the concept that one persons junk is another person's prized possession.

Now thinking that someone is just waiting with money in hand to take from you that old computer may or may not be true. Antiques comprise almost any item, and you can find books that list items and their current suggested collectors price. I recently checked one of the books out and found no listings for computers of any kind. So as far as the official collecting community is concerned, computers have yet to be discovered.

I think that may not even happen, since I consider the computer to be more like the automobile when it comes to collecting. Antique cars are a group of their own, requiring specialist in that area to know their worth and importance for others.

Antique cars are also still functioning, they can still full fill their original design, that being moving you from point A to point B. Any car of any age can do that, some just do it with more snap or flare. Computers are like that as well, they still can full fill their original purpose no matter how old they get.

Cars and Computers

I have compared the industries of cars and computers for many years, and the similarities seem to match up very well. A major difference is the automobile industry took over a hundred years to get to this point, while computers have covered the same technological bounds in less than twenty. A similarity is that both groups have new models each year that keep getting better, faster, and come with more features. The old models for the most part get recycled, unless there is some special appeal. It is that special appeal that gets people to collect them. I always wanted to acquire a Stanley Steamer because of the many unusual features that make the vehicle famous. It was extremely fast (in the 80-90MPH range) using steam when most gas cars were lucky to go 30 MPH.

My now collectible and encased in plexiglass Novix system is much like the Stanley Steamer. It is still extremely fast and was unbelievably fast when it came out. The design is very unusual since it is a Forth Engine designed by Chuck Moore who invented if you will Forth. The components are few, simple, and yet it can do incredible amounts of work with very little code. It reads and writes directly to floppy disks, outputs to printer or a Clone compatible monitor, and uses a clone keyboard. All this is on a two sided board with about ten chips and I believe the Novix itself used about 40,000 transistors which is extremely small in comparison to the million transistor CPU's in most systems these days.

The Parameters

The parameters for deciding if a system is an antique and worth having are much the same as might be used for deciding on a car. Almost all drivers have a special love for their first car. That first computer purchased for whatever reason seems very hard to give up. Thus your first system will most likely be your first item you collect. The parameter here is simple fondness and attachment to your coming of age, in this case not learning to drive a car, but drive a computer.

Can we assign a value to that system, generally not. I have watched a few of the collector TV shows, and someone always brings grandfather's what-you-call-it seeking to hear how many dollars it is worth. The host lets them down gently by saying it is priceless only to them and as such he can not put a price on it. Another way of saying it has no real monetary value, just sentimental value. So too for that first computer.

The next area of car collection is uniqueness. Certainly the first cars to ever be produced are very popular for collectors. Model A's by Ford have many groups that meet and tour all summer long. So too are some of the first computer systems of extreme value. I am certain that a Apple I would be a prize in almost any collector stable. In the same sense having a one of a kind or rather unusual car gets many a collector going. What comes to mind here is the Edsel by Ford. This most unusual car was a complete flop and thus having one or more makes you a most ardent of collectors.

So too would be my Novix as unusual, but there were many other early computers that were flashy, worked well, but flopped. Now if your first computer was an Altair, and you saved it, you have both rarity, uniqueness, and personal attachment. The dollars do start adding up.

One system that sold well and has some appeal is the Sol computer. These S-100 units came in a wood sided keyboard case with enough room for a small monitor to sit on top. This is Stan Veit's favorite system and you can see pictures of it and many other collectible systems, in Stan's book "History of the Personal Computer." (Order the book from Worldcomm at 1-800-472-0438.) For any collector this is a must have book since it is one of the few books that attempts to actually document the systems and the times they were built in.

While mentioning books, another is "A Collector's Guide to Personal Computers and Pocket Calculators", by Dr. Thomas F. Haddock. This is a great book as well, but doesn't have the personal stories that Stan's book does. It is much better however at providing a fairly complete list of machines and their features, with a resale value provided. Although printed in 1993, the PC/XT prices are all much too high, and yet the Altair was listed at $800-1200 which I consider accurate. Get your copy from Books Americana, Inc., P.O.Box 2326, Florence, Alabama 35630.

Now it is also at this point that we need to comment on what does not make a collectible system. Age alone is not a prerequisite, antique motorcycles are defined as being 15 years old and reflected the older designs which seldon lasted more than 10 years. Is it still 15 years, I do not know, but doubt it as the new motorcycles last so much longer. But then how long for computers since their life cycle is now considered in months not years?

At our last Forth meeting in Sacramento, the local Corvair club held their regular meeting next door. Now corvairs are certainly collectible for many reasons, but how about a Ford Fairlane club? I think it would seem a bit out of place. The reason being that almost all Fairlanes were stamped one just like another, flat, no zip, no features, mass consumable, designed to provide transportation, to be used, then disposed of. Unless your Fairlane had some special appeal or alterations, you most likely passed it off to some other user on it's way to be recycled.

I think few would disagree that the PC Clone systems of the last 10 years would fall into this group. Now I have and will put together an original IBM everything system, with a real 256K IBM mother board (although a 64K mother board would be better), and IBM labels everywhere. The reason for that being, first model sample, a collectible item. The no-name clones, their boxes, and boards however were massed produced and are much like the Fairlane, without flare.

Now don't get me wrong, these systems if still running, will provide good service and do what they were built to do. That is true of the Fairlane and the clones. That car will get you from point A to point B cheaply and reliably for many years after the dealers would like to have put you in the driver seat of a new model. The same is true of the clones, WordStar will still help you write that letter or report, with just as much speed and flare as it did five or ten years ago.

To Do or Look At

Here is where we separate the collector from the prudent buyer. A collector is looking for a system that meets one or more of the collectible guide lines. Those guidelines being age, uniqueness, history, or flare. The prudent buyer on the other hand is looking for a good buy that can be made to do more than originally designed for. A case in point is using old systems as backbones for alarm systems, or pump controllers. The old clones have replaced most collector's terminals and provide the extra feature of saving to disk the screen displays of the sign on to your prized antique system.

Last issue I touched lightly on what to do with these $10 systems and their boards. The I/O can be used in newer systems, or might turnout to be the original card you were looking for to complete one of those early collectible models. But by and large you will find little if any collectible value in these older units that have no-names of any kind on them. I rather doubt that clones will ever be worth much of anything, except at doing plain simple computing.

Documents

One area that is currently a problem with collectible systems, is the lack of documentation. Unlike cars which have been well written about, many of the older systems have no such written history. The systems were built with a different concept in mind, get rich quick and get out of the business. Many got out and few got rich and this all happened very fast, in less than ten years. Thus the documentation of these systems and their history has been a real after thought. I have also talked to many, who unfortunately want to forget it ever happened, and definitely not write about their blunders. I think the industry if you will is starting to move toward recognizing this problem and you will start to see some books and more articles by those who actually lived that time.

An important document problem is the users. When people bought these systems, little importance was placed on the owners manuals and software books. Much as many car buyers take the owners manual and file it away to be lost or trashed latter, so too are computer users trashing the manuals. As collectors, we need to try and get the word out that books, manuals, software listings, and old disks, are of value to collectors. I all to often find some one selling a system with no books or manuals. When asked about them, they usually say, "oh I threw those away before I came here, you mean you wanted them too? But why?"

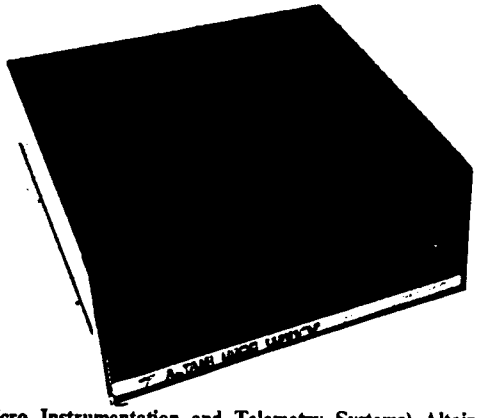
Collecting and acquiring antiques is as much an educating process as it is a hobby or business. We have to educate the consumer as to what helps make a system a collectible (being in good shape with all boards, software, manuals) or an antique (all of the above plus rare and unique also). We do that by getting the word out, talking whenever possible on the topic, writing articles for local papers, and just shopping for that one special system you can't live without.

The *TCJ* Slant

Here at *TCJ* we want to help people collect systems and enjoy the rewards that such a hobby can give them. Currently only a few systems are in demand by collectors. I have had two requests for Altair's, one for an Apple I, all pretty much indicated price is not an issue. As I said earlier, paying $1000

Continues on page 19

Below is a page from the book "A Collector's Guide To Personal Computers and Pocket Calculators" by Dr. Thomas F. Haddock. ISBN 0-89689-098-8, $14.95, from Books Americana, Inc. (800) 726-996 or (205) 757-9966. This page represents the typical information presented on the more than 700 computers and calculators referenced.



MITS (Micro Instrumentation and Telemetry Systems) Altair 8800
Processor: Intel 8080, running at 2 MHz.
RAM: 256 bytes intially standard. 1,024 and 4,096 boards were soon available.
Data storage: Paper tape or cassette initially, later floppy disks.
Physical: S-100 backplane bus in a metal box.
Production: The 8800 was announced in January 1975 but was not shipping by April. Production did not really get going until summer, but by the end of the year over 5000 had been shipped. MITS was sold to Pertec on May 22, 1977, and in about two years it failed.
Cost: Prices were initially $395 as a kit and $650 assembled, but later in production were $439 as a kit and $621 assembled.

This was the first mass-marketed personal computer and originator of the S-100 bus standard. Due to its relatively high production, it is likely the earliest really collectible personal computer. This machine is considered by many to be the first true microcomputer.

Value: $800-$1200

Processor Technology Corp SOL System 1
Processor: Intel 8080A.
RAM: 10 K.
Operator input: Integral keyboard.
Data storage: Cassette and floppy drive.
Production: The company started in April of 1975.
Cost: $1649 as a kit, $2129 assembled.

This was an S-100 machine, following closely after the Altair.

Value: $300-$500

Dr. S-100 Comments

Bill Kibler talked a bit on his comments in this issue of *TCJ* about collecting computers as antiques. While we agree that one or two obvious and well-known computers (the Apple I and the Altair) have reached "collectible" status, we disagree on how far this might go. I remember a recent discussion with a person who makes quilts, a woman of 60 or so years. "Antique COMPUTERS? I can't imaging such a thing! Antiques are OLD - and computers aren't OLD?"

Well, that's one point. Cars go back 50, 100 years; other antiques much longer. But personal computers are maybe 20 years old, even less. Hardly "antiquated" except in a technological sense. You can still drive a car of the 1920's or 30's on the expressway; you can't run Windows on an IMSAI.

Computers as collectibles have other disadvantages. Documentation is hard to get, software harder. Parts are still around but getting scarce. And, the KNOWLEDGE to repair at the part level is getting thin. Many S-100 owners I know can't repair their machines - of course, some can and do, and design more stuff. That's the attraction of S-100 stuff to its users, even today. I might note that COLLECTORS don't use these machines - they want them for show. They like the Altair and IMSAI for the "blinking lights".

But for the real antique collector, a computer is TECHNOLOGY. Not much technology gets collected, and only that which is accessible to the common person: hand tools, radios, cars, and so on. Cars have quite a bit of technology, but so many people of the time were familiar with it, and it was part of the culture of the past. Today, a car is either a consumer item or a means to an end. Few people repair - shudder! - their own cars, yet car repairs occur so often that owners can also get parts and still fix their own cars. But computers are something you buy at K-Mart, so why keep an old one around? Why fix it when you can upgrade? And so on.

So the collectible S-100's will likely be the most popularly known stuff, the stuff that gets hyped. And, it must be rare. As for the machines and cards in demand by USERS, it will be those machines that were well-loved at the time and/or have some aesthetic value (Processor Tech SOL's are a good example), or machines that had considerable respect (Compupro, Cromemco, Morrow). My years of S-100 support are pretty much centered on these companies, because that is what is demanded. Other machines may stand out for different reasons in different parts of the country. Heath computers are well-known, but not so rare so far, and they don't break often anyway. Ithica Intersystems and TDL (Technical Design Labs from NJ) are popular here in the Northeast.

But as for COLLECTIBLE VALUE, the public perception of the Altair 8800 will put a value on it that exceeds any other S-100 machine, even though the IMSAI is a superior machine and probably had more impact in the industry and in personal computing. More recent machines were produced in greater quantity, and since value is inversely related to quantity, their value will remain low for some time.

Ask me again in 30 years, when the first Altair is 50.

Herb Johnson.

Mr. Kaypro

By Charles B. Stafford

Regular Feature

Kaypro Support

Composite Video

84 Kaypro external video

Wherein we devise a magic spell to allow external video for K-10s and '84 K-2s & 4s. This will allow you to use a full size monitor instead of the rather small but usable built in screen.

Gather 'round all you "apprentices" and "neophytes" and ye shall hear of a project which can be as simple or as sophisticated as you wish to make it. It requires only 13 parts, all easily obtainable (even at Radio Shack) for somewhere around $10.00 total. The real "magic" is in the spell, for which we'll use Mbasic. You can use either a "prototype" printed circuit board, or build your own custom PCB, using the pattern included herein.

THEORY

Until recently the most easily obtainable external monitors have "composite" signal inputs, usually an RCA jack with one pin. Several signals are transmitted over this single conductor and the challenge is to create a device that will combine all these signals without affecting the original display. The signals involved are horizontal sync, vertical sync and video.

Inside the '84 K-2s,4s & 10s and the '83 K-10, the generation of the video signal is handled by a 6845 video controller chip. The sync signals come from an oscillator through a divider network. All three signals are routed through a 7406 buffer (U1) on the way to the video board. The video signal leaves U1 on pin 6, while the buffered vertical sync appears on pin 2 and the inverted horizontal sync is on pin 11. Since all the signals go through the same chip, we can remove that chip and plug in our circuit there. Fortunately power is available there as well (pin 14) so this will be the only connection we'll have to make to the Kaypro. By using a wire wrap socket on our PCB, we'll be able to plug the extended legs right into the U1 socket vacated by removing the 7406. We'll use a 74LS00 (see Fig 1) to buffer the signals to avoid affecting the original display, sections C & D of which will buffer the video signal. R1 & R2 form a voltage divider: when the video signal is high (producing white on the screen), the voltage at the base of Q1 goes close to 5 volts to turn on the transistor. When the video signal goes low, R1 & R2 divide the voltage, reducing it to about 2.5 volts, providing a black level on the screen. Sections A & B allow either the Vertical or Horizontal sync to reduce the base voltage to a diode drop below 0 via CR1, turning off Q1. At the same time CR2 speeds up the turn off time of Q1, and CR3 guarantees the base will be a diode drop below the voltage at the emitter, ensuring that Q1 is off. Using a 2N3904 transistor and the resistor values shown should produce a white level of about 2 volts, a black level of about 1 volt and sync levels in the neighborhood of 1/4 volt. These signal levels seem to satisfy most monitors.

CONSTRUCTION

I used a piece of prototype board for the first one I made. It was one of those that had two columns of 4 hole pads spaced just far enough apart that a socket would straddle the space between with one leg each in a 4 hole pad. I put the wire wrap socket in first, a regular socket following that and the transistor socket on the left side below the second socket. That made it easy to wire using jumpers and putting the diodes in from pad to pad. A friend of mine thought he could do better, and laid-out a single sided PCB, the pattern for which has been included, should you wish to etch your own. Where the schematic says "out", connect a two conductor piece of wire long enough to reach the video connector which will be mounted in one of the cooling slots just beneath the mother-board. I used a piece of small shielded coax out of the junk box, but the shielding isn't really necessary, since the distance is short.

THE MAGIC

After building the external adapter, and double checking the circuit against the schematic, plug the adapter into the host machine (U1), hook up your external monitor and turn the whole works on. The odds are that the monitor has been adjusted for standard video timing and the display will be rolling and torn.

Commercial television signals use a 60 Hertz vertical sync as do the 83 K-2s & 4s. The 84 K-2s & 4s and all the K-10s, however, use a vertical sync of 51 Hertz so we'll have to adjust the monitor's vertical hold.

Commercial television signals also provide a horizontal sync signal every 63.5 micro-seconds (corresponding to a sweep frequency of 15,750 Hertz) while the 84 Kaypro provides one every 47.56 micro-seconds (corresponding to a sweep frequency of 21,028 Hertz). Adjusting the monitor's horizontal hold will help solve the tearing and may, in fact, cure it

completely. If so, you're finished with the project.

Many monitors, however, cannot handle the higher frequency, even with the horizontal hold turned as far as it will go, BUT since the '84 Kaypros use a 6845 video controller, the horizontal frequency can be adjusted using software.

HOW WE DO IT

The 6845 has several registers (storage locations) which can be loaded with data to control its operation. Register 0 controls the horizontal sweep rate and contains a value equal to the total number of characters on a horizontal line minus 1. The dot clock (which determines the rate at which dots are displayed on the screen) is 18 MHz. Kaypro uses 8 horizontal dots in each character cell. When the Kaypro is turned on, register 0 is set to 106, so (here comes the math !!) (106+1)*(8/18MHz)=47.56 microseconds or 21,028 Hertz. To increase the time for each line, (that is, to reduce the horizontal sweep frequency) to something your monitor can handle, we must increase this number stored in register 0. This is achieved by outputting to port 1C hex (28 decimal) the register we wish to address, followed by outputting to port 1D, the value we wish to put into the register. In Mbasic the statement is: OUT (28),0 : OUT (29),107

As the value (107) is increased, you will reach a point where the horizontal hold on your monitor will lock onto the signal and produce a stable display. If the number is increased too much, The internal screen will go out of sync and you'll lose the internal display. Increase the number gradually, so you don't go any farther than necessary. Usually the number will wind up some where between 108 & 113.

This process is a balancing act. As the time per horizontal line is increased, the vertical time per frame is also increased, reducing the frame rate, and the external monitor screen will begin to blink slowly enough to be seen. This will be most noticeable when the screen is full. The Kaypro's screen has a long enough persistence to minimize this effect, but the external monitor likely won't. There are two registers in the 6845 to control the vertical frame rate: register 5, a fine adjust (currently set to 10), and register 9, the number of scan lines per character. Setting register 5 to 0 will help considerably. In Mbasic: OUT (28),5 : OUT (29),0 To gain any more we'll have to reduce the number in register 9, but doing this will cause a change in the characters on display.

Each character cell is 16 lines high. If we reduce the number to 14 (the number stored is scan lines per character minus 1), we lose the underline capability. (In Mbasic: OUT (28),9 : OUT (29),14) This won't hurt the text much, but will upset the graphics which need all 16 lines.If we reduce the number to 13, we'll lose the lower line of the descenders, "yqgj", making the text pretty tough to read. If an even higher number is necessary, then the character ROM will have to be changed.

Once you have a readable display on your external monitor, the horizontal position of the displayed line can be adjusted by changing the value in register 2 (default 86) to move the line right or left. There are other registers in the 6845 that can be tinkered with, but not without consequences. Since the video driver software in the Monitor ROM expects the screen to be 25 lines of 80 characters, it too would have to be changed.

Now that you've twiddled the bits and figured out what parameters are right for your monitor, you can enter them interactively in Mbasic. You must do this each time you boot up, however. One alternative would be to write an Mbasic program to do all the dirty work for you and call it with an embedded initial command line, or with very little more work you could use the Sbasic compiler to produce a .com file to do the same job. There is no end to the possibilities. Whatever you come up with, drop me a line and let me know, and sometime in the future we'll publish a compendium of solutions.

PARTS LIST

1 14 pin wire wrap socket
1 14 pin socket
1 74LS00 integrated circuit
2 220 ohm resistors
1 100 ohm resistor
1 68 ohm resistor
3 1N914 diodes
1 2N3904 transistor
1 video connector (RCA type or F type for rear panel mounting)
1 prototype board

Note:

The PCB drawings are 2X actual size and printing may have distorted them as well. Reduce with copying machine and use transfer paper to iron on the PCB traces. Also note the backside is flipped end for end, check hole alignment for proper orientation.

The 6845 is the same chip used on IBM PC's for monichrome adapters. More information on setting the device up can be found in many of the PC hardware support books, and the chip is manufactured under license from Motorola to numberous vendors. BDK

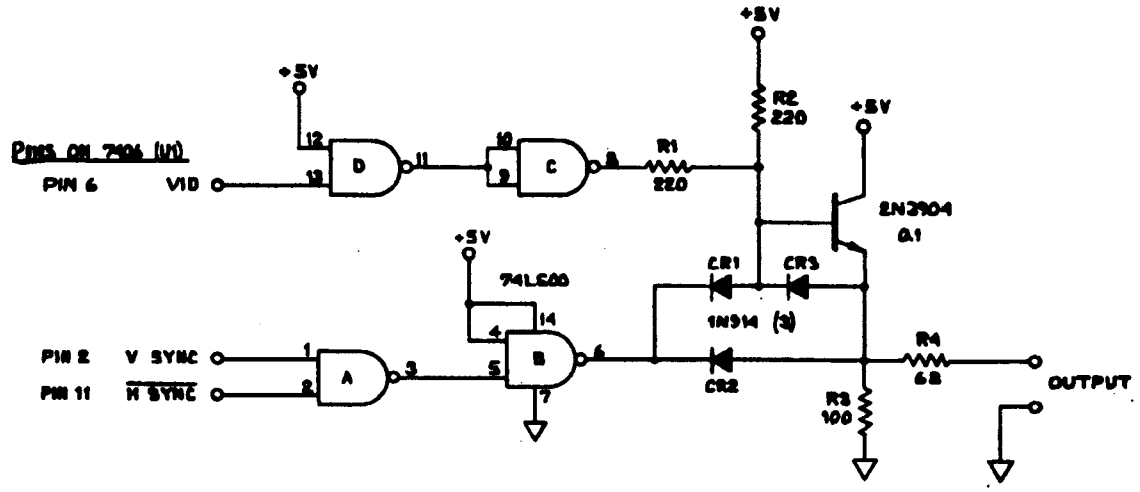


Figure 1. Schematic.

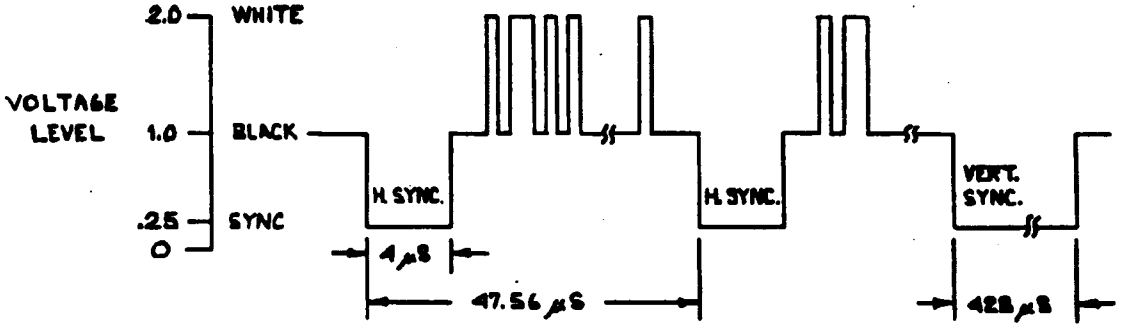


Figure 2. Video Signal.

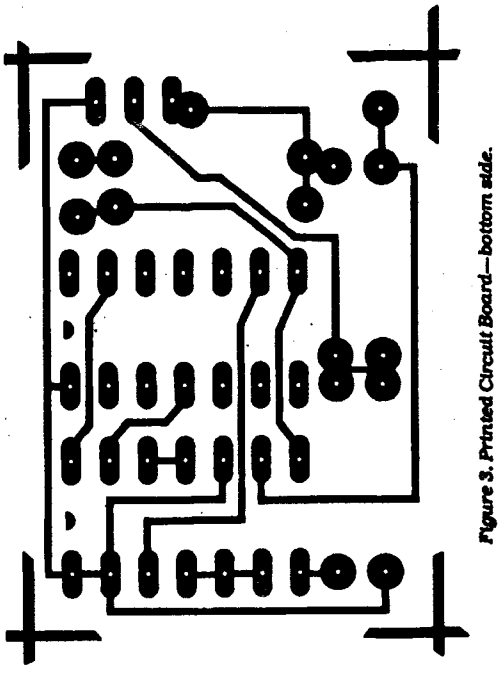


Figure 3. Printed Circuit Board—bottom side.

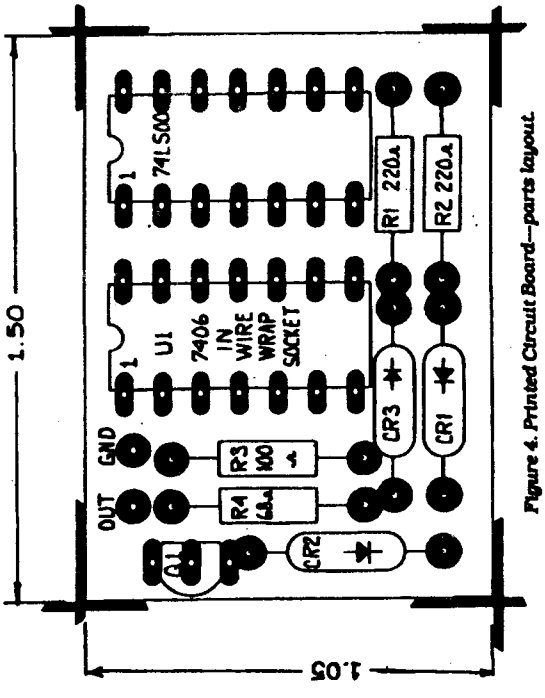


Figure 4. Printed Circuit Board—parts layout.

Special Feature

Classic Support

Best CP/M Machine?

The AMSTRAD PCW Now

By Bill Roch

One of the best kept computer secrets of a few years past was the Amstrad PCW 8256, 8512 and 9512. It was the most advanced CP/M mass marketed computer sold in this country. The regular CP/M community completely over looked them because they were sold as a Word Processor. As I understand it Sears bought some 70,000 8256's and 8512's from the English company Amstrad. In 1986 they spread them all over the USA and supported them for years. Unfortunately there was never a follow up by Sears.

Amstrad then came to the USA and set up an office and sold the 9512 as an upgrade to the ones sold by Sears. Amstrad then came out with some "almost" IBM PC's and later some other DOS models. They got clobbered by the clones and abandoned the U.S.

While there are no more Amstrad's coming to the USA they are still going strong in Great Britain. There were two British glossy magazines devoted to the Amstrad being printed. One magazine stopped publication in the first part of 1995.

I received a Fax from one of my suppliers of Amstrad products indicating that another batch of PCW's will be released this September. They will probably produce another 10,000 for local consumption - and it is basically the same machine as the 1986 model except with a 3-1/2" floppy disk drive instead of the original 3" drive. Also they will put a printer port that will handle popular printers.

Seems that I read somewhere that there are some two million PCW's in the British Isles. There are user groups or clubs all over the British Isles as well as in most European countries and Australia and New Zealand.

Al Warsh, produces an Amstrad News Letter "AMSTRAD PCW USER'S SIG" here in the U.S. and called me a few years ago and wanted me to copy an Amstrad disk to a DOS disk. I had never heard of Amstrad but I figured with Uniform, Media Master, Smart Disk and some home grown software written by Mike Karas I could find a format that would work. Then he told me that it was a 3 inch disk. I new better. It must be 3-1/2 inch. Anyhow it was a 3 inch disk and I couldn't handle that.

To make a long story shorter Al convinced me that I should get a 3" drive and hook it into my Tarbell S-100 system. I bought a 3" drive from Sinotech and Mike Karas did what was necessary and soon I was copying Amstrad files to DOS and CP/M disks.

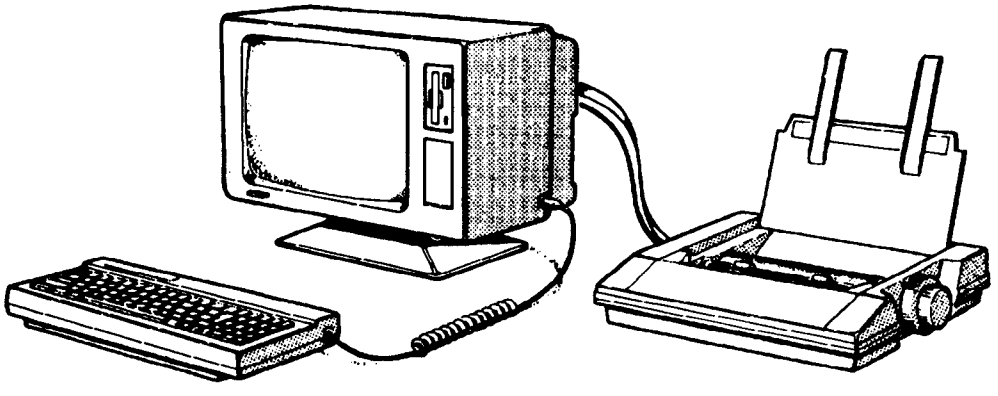
About this time I met a man who was using an Amstrad and wanted a Word Processing program to use on his XT. I traded my copy of WordStar 5 for his Amstrad. The machine I got was just like the ones sold by Sears and consisted of:

Monitor - 30 x 90 green graphic screen. Acts like a VT52.
Z-80 4mh CPU card with 256K of memory, expandable to 512K

One 3" single sided, double density disk drive using flippy disks with 173K capacity per side.

Cables for a second floppy drive.

82 key keyboard with math pad, arrow keys and eight function keys as well as some 10 special keys.



Epson MX type 9 pin Amstrad printer.

There is no serial or parallel ports on the computer. The printer is tied directly to the computer.

CP/M Plus with Utilities, Mallard BASIC and DR Logo.
Locascript - Self Booting Word Processing program that takes over the machine.

Manual for LocaScript word processor including CP/M Plus and DR Logo.

The first thing I did was to install FileBuster,(CP/M Xtree), then WordStar 3.3, dBASE II, ZBASIC and some other programs. The Mallard BASIC that comes on the Amstrad CP/M disk is like Microsoft BASIC 80 and has a B-Tree system for random files built in. No compiler though. My CP/M programs worked fine after being installed properly.

The more I found out about the Amstrad the better I liked it and since I was still supporting CP/M software I might as well support the Amstrad also. Sinotech in Chicago was my main supplier and soon Elliam Associates was in the business of supporting Amstrad users. After Sinotech dropped out of the Amstrad software business a few years ago Elliam Associates became the main source of supplies, software and hardware for the PCW in the U.S.

The 8000 PLUS Amstrad magazine from England listed all kinds of wonderful software and hardware that could enhance the machine. We have since been selling software:

MicroDesign - a desk top publishing program that uses graphics and scalable fonts and with the ability to use a mouse and a scanner.
LocoScript 3 - a word processor with built in "on the fly" spelling checker, four fonts per document with scalable fonts from 6 point to 72 points.
LocoMail - a mail merge type program.
LocoFile - a card file type data base.
2IN1 and MFU disk copy/conversion software.
2IN1 - Amstrad software to read and write DOS disks.

And selling hardware additions:

Serial/Parallel interface so that any printer can be used as well as a modem. (I use a H/P Laser).
Additional memory - up to 2 Meg.
B drives - 3", 3-1/2", 5-1/4" - 720K ds/dd disks.
Sprinter - A add-on board that almost doubles the speed of the Z80 and also holds additional memory.
Flash Drive - CMOS memory that can be loaded with programs and boots in CP/M, LocoScript or from the A drive disk.

Other items needed to keep it running:

Games and Educational Programs. Disks and printer ribbons. All kinds of public domain and standard commercial CP/M software. Amstrad "how-to" books. Amstrad specific software such as FLIPPER and NETWORK. FLIPPER lets you multi-task up to 8 programs and NETWORK acts kinda like Windows for launching programs.

Locomotive Software, authors of the Amstrad word processing programs, have produced a DOS version so folks moving to a PC can still use a program that has the look and feel of LocoScript. They just recently came out with "LocoLink", a Windows program for copying files from a PCW via a special cable to a PC. Files may be converted during the copying to many of the most popular DOS and Windows word processing file formats. Data files may be converted to a dBASE or ASCII format.

While the original Amstrad's had only one disk drive it was possible to add a second B drive. The Amstrad B drive used a 3" double sided disk drive that had a capacity of 720K - quite an improvement over the 173K on the A drive. Sinotech also had a external 5-1/4" B drive that plugged into the B drive cables.

MoonStone Computing came up with software that faked out the 5-1/4" B drive into acting like a 360K DOS drive. So now files could be moved between the Amstrad and a PC. Since 5-1/4" and 3-1/2" drives act the same hardware wise it wasn't long before I had an internal 3-1/2" 720K drive for the Amstrad. The mounting bracket allowed the 3-1/2" drive to be installed in the spot reserved for a 3" drive right under the 3" A drive. Harley Ristad figured out what leads to cut and what jumpers to solder so that a standard Teac drive would work.

There have been hard drives for the PCW's for a long time. I have one for a 9512 but the cost has been to high to be practical for the 8256 and 8512. Even when the DOS world was spending $600.00 for a 125 Meg they were too expensive. Spending $600.00 for a 40 Meg hard drive for a CP/M machine just does not make sense.

My answer to the hard drive is to install a switchable A drive with both 3" and 3-1/2" drives, Sprinter with 1.5 Meg of memory and a 1 Meg Flash Drive. With the Flash Drive CP/M is up and ready to use by the time the screen is warm enough to display the characters.

As you can tell I have high regards for the Amstrad PCW. It is a terrific machine for folks who like to fool around with CP/M and also it is ideal for those people who write a few letters a month. With appropriate software it rivals some of the fancier DOS and Windows software. It is probably the best designed niche market computer ever made and that is why it is still going while other 1986 vintage machines have long since bit the dust.

Bill Roch, Elliam Associates

Regular Feature

68xx/68xxx Support

C & Assembly

Small System Support

By Ronald W. Anderson

It is labor day weekend (1994) as I start this column. I have my 6809 system back from being on loan to the company while we formatted about 1000 disks. We decided to make a second backup set of our customer's program files and store them somewhere remote from the company in case of a fire or flood. Now I can easily work on Assembler programming projects again. I have a set of nine lessons from the C programming project, and have only to convert them from WordStar for Windows format to simple ASCII text. I have been filling in some missing information as I go along with these. Well, let's go.

C Tutorial #3

Last session we talked about assignment statements and assignment operators. I think we covered them all. This time we will talk about "bitwise operators", "logical operators", and "relational operators". Relational operators are used in logical comparisons as in:

if(a > b) ... This is a "logical expression" as distinguished from an "arithmetic expression".

Relational Operators

a > b	if a is greater than b evaluates to TRUE if a is less than or equal to b evaluates FALSE
a < b	if a is less than b evaluates to TRUE if a is greater than or equal to b evaluates FALSE
a <= b	if a is less than or equal to b evaulates TRUE if a is greater than b evaluates FALSE
a >= b	if a is greater than or equal to b evaluates TRUE if a is less than b evaluates FALSE
a != b	if a is not equal to b evaluates TRUE otherwise evaluates FALSE
a == b	if a is equal to b evaluates TRUE otherwise evaluates FALSE

Be careful with the last two. These are different (known as Murphy's law of the perverse program designer). In particular the "==" comparison operator causes problems. In C the value FALSE is defined as 0 and the constant value TRUE is defined as 1. However any value that is not zero evaluates to true, so a test like "if(a != 0)" can just as well be written as "if (a)". You can think of it as "!=0" being "understood". You can always leave it out when you are testing for zero or non- zero).

Logical Operators

((a == 0) && (b == 17))	TRUE if a is zero AND b is 17
((a == 0) || (b == 17))	TRUE if a is zero OR b is 17 (or both)
!	is the NOT operator
if(!(feof(infile))	TRUE if NOT end of file infile (feof() is a standard library function)

When using logical operators it is important to use parentheses wherever the evaluation might be ambiguous. C has a number of precedence rules which you will eventually learn. Meanwhile, when in doubt use parentheses.

Bitwise operators

00001111 & 10101010	= 00001010 (the AND function)
01010000 | 00000101	= 01010101 (the OR function)
~00100111	= 11011000 (the inversion or NOT)
01011100 ^ 10101100	= 11110000 (exclusive OR)
~(01011100 ^ 10101100)	= 00001111 ("neither or both function")

As you can see by the last example, these operators can be combined into expressions. Again it is a good idea to use parentheses liberally. We've now covered the arithmetic operators, the logical operators, the relational operators and the bitwise operators.

Control Statements

You have all been exposed to enough BASIC to have run across the IF THEN ELSE statement. In BASIC it looks like this:

IF logical expression THEN statement ELSE statement

The logical expression is enclosed between IF and THEN for the BASIC interpreter. In C the keyword "then" does not exist. Logical expressions are always enclosed in parentheses, which define the end of the expression so THEN is not required:

```
if(a<b) puts("a is less than b");
      else puts("a is greater than or equal to b");
```

Notice in particular that the first part of the statement is terminated by a semicolon. The "else" part is optional as in BASIC or Pascal. Furthermore in C either statement may be compound:

```
if(logical expression)
{
      statement;
      statement;
      statement;
}
else
{
      statement;
      statement;
}
```

Again the else is optional and either of the statements may be compound or simple.

Loop Control

C has three loop control structures. The first is equivalent to the FOR NEXT loop in Basic:

```
int n;

for(n=0; n<10; n++) printf("%d ",n);
```

The parentheses enclose three items. The first is the initial value for the index variable n. The second is the condition for which the loop will MAINTAIN (not the condition for which it will terminate as in BASIC). The third is the "increment". In BASIC this information is scattered. The NEXT statement may be a page away from the FOR statement. C puts them all together in one (of course).

```
for(n=1; n < 32768; n*=2) printf("%d ",n);
```

This will give us a printout of the powers of 2 since n will take on the values 1, 2, 4, 8, etc.

Of course the statement here may be replaced by a compound statement that could be two lines or two pages of code.

For the moment, please don't worry about the printf() function. We will get into that in detail in a lesson or two. Let me just say that the part in quotes is a "format string" that tells the function to print n as a decimal number and put a space after it.

The second loop control statement is the "while":

```
n=0;
while(n < 10)
{
      printf("%d ",n);
      n++;
}
```

The most common mistake in using this loop control is to forget to increment the variable, in this case n. If you don't do that, the loop will print "0" forever! There is a simpler way to increment n:

```
n=0;
while(n < 10) printf("%d ",n++);
```

This does exactly the same thing as the previous example (and the first for - next example). You can see that C can do it in one line.

There is another loop that puts the test at the end of the loop:

```
n=0;
do
{
      printf("%d ",n);
      n++;
} while(n < 10);
```

I like to put the condition on the same line as the ending curly brace. It can go just as well on the next line, but I think this ties it to the loop better visually when you read the listing. Again if we run this it will do the same thing as the other examples. If we initialize n to 100 this time, however, the loop will print "100" because the code is executed BEFORE the condition is checked. Notice in each of these three cases the condition is that which will MAINTAIN the loop, not the condition that TERMINATES it. If you remember that you will be able to keep things straight. Incidentally, what will these examples print out?

0 1 2 3 4 5 6 7 8 9

When n reaches the value of 10 the loop will end.

Next time we will talk about printf() (the formatted print function) and the associated input function scanf(). If we have room we will talk about functions in general and function parameters in particular. At that point we will be able to write a program a bit longer than two or three lines!

Assembler

This time let's extend the program of last time, which listed a text file to the terminal so that it will copy a file (text or otherwise) to a file of a different name or the same name on another disk. The changes are not very extensive. We get two filenames from the command line and open the first one for read and the second for write. We then read the first file and instead of writing it out to the terminal we write it out to another file. Then we close both files and exit to FLEX.

Before we get into that, however, let me tell you about a major simplification of the program we called LIST last time. I showed you how to read a filename from the command line and insert it in the file control block one character at a time. We had to look for special "separator" characters, namely the period that separates the filename from the extension and a CR to tell

us when we got to the end of the line. I did it the hard way purposely to show you a little concentrated assembler code. We didn't use much in the way of new instructions, but we ended up with some code with multiple branches that require a bit of attention to follow. Flex has a system call named GETFIL which gets a filename from the command line and stuffs it into the FCB pointed at by the X register on entry. GETFIL sets the drive number in the FCB to the working drive. It correctly handles the case of no extension, and we can again use SETEXT to take care of that case with the default extension .TXT. Here is the LIST program I have called LIST1. It uses GETFIL and so is quite a bit shorter than LIST:

```
* PROGRAM TO LIST A FILE TO THE SCREEN
* GETS FILENAME FROM COMMAND LINE PARAMETER

NAM LIST

PUTCHR EQU $CD18
WARMS  EQU $CD03
RPTERR EQU $CD3F
FMS    EQU $D406
FMSCLS EQU $D403
NXTCH  EQU $CD27
SETEXT EQU $CD33
WRKDRVEQU $CC0C
PCRLF  EQU $CD24
GETFIL EQU $CD2D
CR     EQU $0D
FOPENR EQU $01
FCLOSE EQU $04
FEOF   EQU $08

* THIS VERSION USES FLEX GETFIL

START  LDX  #FCB
       JSR  GETFIL
       LDA  #1
       JSR  SETEXT   SETS EXTN TO .TXT IF NONE SPECIFIED
       LDA  #FOPENR  OPEN FOR READ CODE
       STA  0,X
       LDA  WRKDRV
       STA  3,X
       JSR  FMS
       BNE  ERROR    FMS SETS NON ZERO ON ERROR
LOOP   JSR  FMS       READ A CHARACTER
       BNE  ERROR
       JSR  PUTCHR    WRITE IT TO SCREEN
       CMPA #$0D     IS IT A CR?
       BNE  LOOP      IF NOT, OK
       JSR  PCRLF
       BRA  LOOP      GO AROUND AGAIN

ERROR  LDB  1,X
       CMPB #FEOF    TEST FOR END OF FILE
       BEQ  DONE
       JSR  RPTERR      TELL USER WHICH ERROR - X
            *         POINTING AT FCB
       JSR  FMSCLS        CLEAN UP BY CLOSING ALL
            *         OPEN FILES ON ERROR
       JMP  WARMS

DONE   LDA  #FCLOSE     BRANCH HERE ON EOF
       STA  0,X         CLOSE THE FILE
       JSR  FMS
       JMP  WARMS

       ORG  $200
FCB    FCB  0,0,0,1
       FCB  0,0,0,0,0,0,0,0,0,0,0 ELEVEN ZEROS FOR
                    * FILENAME AREA
       RMB  305

       END  START
```

As you can see, GETFIL took the place of a lot of code that we wrote earlier. It pays to read the FLEX manual carefully. There are quite a few very useful operating system calls there.

Well, now for the new program that will copy a file:

```
* COPY A FILE

       NAM MYCOPY

PUTCHR EQU $CD18
WARMS  EQU $CD03
RPTERR EQU $CD3F
FMS    EQU $D406
FMSCLS EQU $D403
NXTCH  EQU $CD27
SETEXT EQU $CD33
WRKDRVEQU $CC0C
PCRLF  EQU $CD24
GETFIL EQU $CD2D

CR     EQU $0D
FOPENR EQU $01
FOPENW EQU $02
FCLOSE EQU $04
FEOF   EQU $08

* THIS VERSION USES FLEX GETFIL

START  LDX  #IFCB
       JSR  GETFIL
       LDA  #1
       JSR  SETEXT   SETS EXTN TO .TXT IF NONE SPECIFIED
       LDA  #FOPENR  OPEN FOR READ CODE
       STA  0,X
       JSR  FMS
       BNE  ERROR    FMS SETS NON ZERO ON ERROR
       LDB  #$FF
       STB  59,X     MAKE IT A BINARY READ

* NOW DO IT AGAIN FOR THE OUTPUT FILE
       LDX  #OFCB
       JSR  GETFIL
       LDA  #1
       JSR  SETEXT
       LDA  #FOPENW  OPEN FOR WRITE CODE
       STA  0,X
       JSR  FMS
       BNE  ERROR
       LDB  #$FF
       STB  59,X     MAKE IT A BINARY WRITE

* THIS LOOP DOES THE ENTIRE COPY PROCESS
LOOP   LDX  #IFCB
       JSR  FMS       READ A CHARACTER
       BNE  ERROR
       LDX  #OFCB
       JSR  FMS       WRITE A CHARACTER
       BNE  ERROR
       BRA  LOOP      GO AROUND AGAIN

ERROR  LDB  1,X
       CMPB #$08     TEST FOR END OF FILE
```

```
        BEQ  DONE
        JSR  RPTERR      TELL USER WHICH ERROR - X
             *         POINTING AT FCB
        JSR  FMSCLS      CLEAN UP BY CLOSING ALL
             *         OPEN FILES ON ERROR
        JMP  WARMS

DONE    LDX  #IFCB
        LDA  #FCLOSE
        STA  0,X           CLOSE THE FILE
        JSR  FMS

        LDX  #OFCB
        LDA  #FCLOSE
        STA  0,X           CLOSE THE FILE
        JSR  FMS

        JMP  WARMS

        ORG  $200
IFCB    FCB  0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        RMB  305
OFCB    FCB  0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        RMB  305

        END  START
```

The first thing to mention is that FLEX allows the use of more than one file control block. One is required for each file that is to be open simultaneously, in this case two. I've called the input file control block IFCB and the output file control block OFCB.

Note that if and when you run this program you will be opening a file and writing to it. As I mentioned in an earlier installment it might be prudent to put this program on a freshly formatted disk and perhaps even to make a backup copy before you try to run it. You can run it on your working drive only, if you copy one of these files to one of a different name. In that case you can open the door of the system drive and even remove the disk if that makes you feel better. That way, there is no chance of clobbering anything but the working disk with only this program on it. If you made an error and the program bombs, the worst thing that can happen is that you will have to re-boot the system and look for the error on the backup disk copy.

Opening a file for write is just like opening one for read except for the function code you put in the first byte of the FCB. GETFIL can get as many filenames as you need to put on the command line. If you have three filenames, you call GETFIL three times, each time pointing X at a different file control block.

You will notice that we have to keep reloading the X register (twice per pass through the loop to read and write one character). While that looks like a lot of busy work, the instructions are executed quickly.

I have done one other thing here. I mentioned in an earlier installment of this that FLEX does some file compression. If there are 12 spaces in a text file, FLEX writes them to disk as a TAB character $09 followed by the binary code for 12 $0C. When FLEX reads a file from a disk it expands these tab codes into spaces again. You never see the file with tab codes unless you look directly at the disk file contents. When you copy a file it makes no sense to expand the spaces only to compress them again. An executable file (.bin or .cmd) is not compressed. There is a "flag byte" in the FCB that tells FLEX not to expand or compress a file, that is, treat it as a binary file. This flag is set in the above program with the instructions LDB #$FF, STB 59,X.

Having done this, the program can copy text files or binary executable files. The command line is:

1.MYCOPY.BIN INFILE OUTFILE

That is, you put the source and destination filenames on the command line. You can include a drive number and an extension, but if you don't, the drive defaults to the working drive and the extension defaults to .TXT. If I wanted to copy this program to the system drive, I would use the command:

1.MYCOPY.BIN MYCOPY.BIN 0.MYCOPY.CMD

This assumes that the working drive is 1 and the system drive is 0. Having copied the file and renamed it to a .CMD extension, the command simply becomes MYCOPY. The following command will copy the MYCOPY.TXT file on the working drive to a file named YOURCOPY.TXT on the working drive.

MYCOPY MYCOPY YOURCOPY

Other than that, this program is not much different than our LIST program. It checks for the end of file code on an error, and if found it closes the files and returns to FLEX. If the error is not End of File it reports the error and returns to FLEX.

Oh, yes, you might wonder about the ORG $200 statement before the IFCB and OFCB. A program and its data or variables don't have to be "contiguous", that is, they can be at different locations. I had added the ORG statement so I would know exactly where the file control blocks were in memory. Then I could check with a memory dump utility or the one in the SBUG monitor to see what got stored in the file control blocks. Knowing the exact address of a variable is helpful in debugging a program.

Position Independent Code

Since this column is still a bit short, and I don't have a lot of chatter, let's move on to position independent code. The copy program above can't be loaded anywhere in memory except at the default address zero. At least if we remove the ORG $200 for the FCB definitions it can't be. Of course it wouldn't be position independent if we left the ORG statement anyway, since the FCB would always be at $200.

Calls to FLEX will remain "absolute" since FLEX doesn't move around if we load our program at a different address. Just what, then ties the program to a particular load address? In the

case of this program it is only the variable references provided all the branch instructions in the program code are "relative" that is, provided we use BRA or LBRA instructions and not JMP instructions. Branch instructions on the 6809 are all relative to the current instruction address. That is, the assembler codes an "offset". The offset can be positive or negative. The BRA LOOP instruction in the program has a negative offset. The BEQ DONE instruction has a positive one. A BRA (any of the conditional branches such as BEQ, BNE etc.) instruction has a signed 8 bit offset generated by the assembler as part of the instruction code. That is, you can branch ahead up to 127 bytes or back as far as -128 bytes. If your code gets long and the branch needs to go farther than that, the assembler will indicate a "BRANCH OUT OF RANGE" error. You can simply change the BRA instruction to LBRA. LBRA stands for Long BRAnch. A long branch instruction has a 16 bit offset so for all practical purposes you can branch just about anywhere in memory space.

If you code a LBRA instruction where only a BRA is needed, the assembler flags the instruction for you with a < sign meaning the branch is less than the limit for a BRA instruction and you can so change it.

That takes care of the code, but what about the FCB. The thing that kills position independence in our copy program is the LDX #IFCB and LDX #OFCB instructions. The file blocks are calculated to be at some address (even without the ORG $200 statement). They in fact simply come after the last byte of code. To make this program position independent, all we have to do is to make the following change and others like it:

Change LDX #IFCB to LEAX IFCB,PCR

LEAX, as I mentioned before, means Load Effective Address X. The PCR means Program Counter Relative. The assembler determines the program counter at the LEAX instruction and calculates how many bytes to the IFCB label. It codes the LEAX instruction and the offset, just as the BRAnch instruction codes the offset or relative position of the branch destination. The entire program is then set up to key jumps and variable locations to the present program counter. To run a position independent program at different locations in memory we simply copy the program into memory at the desired location and jump to the first byte of the program.

It is helpful in writing a position independent program to make the first instruction start at the first byte (i.e. the load address). It can simply be a branch or LBRA to the actual starting point for the program. Next time, obviously, we ought to write a simple offset loader that takes a command line address parameter, an executable file name, and loads the file at the address and then jumps to that address to execute the program. Armed with such a loader, we can test our program to see if it is truly position independent. Here is the position independent version of MYCOPY.

```
* COPY A FILE

        NAM COPY

PUTCHR EQU  $CD18
WARMS  EQU  $CD03
RPTERR EQU  $CD3F
FMS    EQU  $D406
FMSCLS EQU  $D403
NXTCH  EQU  $CD27
SETEXT EQU  $CD33
WRKDRV EQU  $CC0C
PCRLF  EQU  $CD24
GETFIL EQU  $CD2D

CR     EQU  $0D
FOPENR EQU  $01
FOPENW EQU  $02
FCLOSE EQU  $04
FEOF   EQU  $08

* THIS VERSION USES FLEX GETFIL and is position independent

START   LEAX IFCB,PCR
        JSR  GETFIL
        LDA  #1
        JSR  SETEXT    SETS EXTN TO .TXT IF NONE SPECIFIED
        LDA  #FOPENR   OPEN FOR READ CODE
        STA  0,X
        JSR  FMS
        BNE  ERROR     FMS SETS NON ZERO ON ERROR
        LDB  #$FF
        STB  59,X      MAKE IT A BINARY READ

* NOW DO IT AGAIN FOR THE OUTPUT FILE
        LEAX OFCB,PCR
        JSR  GETFIL
        LDA  #1
        JSR  SETEXT
        LDA  #FOPENW     OPEN FOR WRITE CODE
        STA  0,X
        JSR  FMS
        BNE  ERROR
        LDB  #$FF
        STB  59,X        MAKE IT A BINARY WRITE

LOOP    LEAX IFCB,PCR
        JSR  FMS         READ A CHARACTER
        BNE  ERROR
        LEAX OFCB,PCR
        JSR  FMS         WRITE A CHARACTER
        BNE  ERROR
        BRA  LOOP        GO AROUND AGAIN

ERROR   LDB  1,X
        CMPB #FEOF       TEST FOR END OF FILE
        BEQ  DONE
        JSR  RPTERR        TELL USER WHICH ERROR - X
                         * POINTING AT FCB
        JSR  FMSCLS        CLEAN UP BY CLOSING ALL
                         * OPEN FILES ON ERROR
        JMP  WARMS

DONE    LEAX IFCB,PCR
        LDA  #FCLOSE
        STA  0,X         CLOSE THE FILE
        JSR  FMS

        LEAX OFCB,PCR
        LDA  #FCLOSE
        STA  0,X         CLOSE THE FILE
        JSR  FMS
```

```
        JMP   WARMS

IFCB    FCB   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        RMB   305
OFCB    FCB   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        RMB   305

        END   START
```

One thing that might tend to be confusing is the difference between LDX #OFCB and LEAX OFCB. The first says to load the X register with the value of the symbol OFCB, which is the address of OFCB. The second says to load the effective address of OFCB into the X register. The immediate sign is not used with LEAX.

Next time we will do an offset load and run program. I wrote one years ago called LOGO for LOad and GO. I'll dig it up somewhere and present it, and then we can try it on the above program. Where PIC really makes sense in the FLEX environment is for something like a memory dump program. You might want to dump memory at just about any address. You can load the dump program at another address and not destroy what was there before you can dump it.

Spelling

I see I have some space left, so I am going to copy an item found on Internet recently. It is a poem written by Larry Hatfield and revised by Wade Price. The interesting thing about this poem is that it was checked with a spelling checker, and the checker reported no errors!

Eye have a gnu spill checker
Hit come width my Pea See
Know longer dew eye knead two spill
Hit does the work fur me

Hit makes things sew much bidder
Par faction is hear Tuesday (perfection here to stay)
Eye tail ewe, righting is sew much fun
Eye can't pool myself a weigh

Sew fallow my eggs ample
Every time ewe rite
Four lung Yule harem seine ('fore long you'll hear 'em saying)
That ewer reel outer site

Wail, hive eight my bole of serial
Booted up and hit f-3
And fur the next too ours
This is wear aisle bee

In front of my Pea See
My marvel us Pea See

The obvious point is that you can freely use the wrong word as long as you spell it correctly. Why, a native Chicagoan like me can even write about his neck store neighbor* (exactly the way it sounds) and not get a spelling error.

Chicagoans also say something like "tempechur" for temperature. I know. I did it myself for years and probably still do (though I moved from the Chicago area nearly 30 years ago). I once was on a business trip to Dallas and the local weather man said "tem-per-uh-tyoor". I thought to myself "of course" that was the proper way to say it, but I had never heard it quite that way before. The other day I heard someone on the radio say "groshries" for "groceries". I was reminded that I had heard it (and said it) that way in Chicago. My mother always had me drink "ornch" juice (one syllable) rather than "orange" juice too.

* If that one is a bit too distorted, I am referring to my "next door" neighbor, of course.

By the way, all those r's that are lost on the East coast when the parking attendant pahks your cah, don't really get lost. They turn up in Indiana and environs where people warsh their clothes.

There are localisms here in Michigan that sound strange to me. For example in Grand Rapids (and nowhere else that I have discovered) people eat hamburgs (not hamburgers). My favorite sign in Grandville Michigan is one in front of a funeral home. The place is owned by a family whose name is "Posthumus". I've always thought that their slogan ought to be something to the effect that people ought to be burried posthumusly. Maybe a little too light for a place with such a serious business...

There is (or was) a hot dog and root beer place in G.R. called Frank 'n Stein. In Chicago when one drinks root beer, root rhymes with foot, not with loot.

Continued from page 8, Antique or Junk ?

for an Altair now probably would be a good purchase. Two to three years from now, 10 or 20 thousand just might be the going price. Only time will tell.

What sort of support is needed. A forum for discussion, a place to seek documents or help, training and teaching material for doing repairs yourself, any other hobby projects that keep your skills sharp and sense of humor robust. That is pretty much our format as it sits now. The classic issue to focus on collecting and repairing older systems, some robotics and embedded work to sharpen your skills, and for humor we have supporting the PC/XT or "how to make a flop do real work."

As always, we want to hear from you and have you participate in keeping the topic alive. Internet is nice, but being in real print in *The Computer Journal* is even better. Try it! Send those letters, email, and stories about way back when for others to enjoy.
Bill Kibler.

Regular Feature

Intermediate

The Mailbag

Dr. S-100

By Herb R. Johnson

As I write, Spring has just turned to Summer, and the lawn has finally stopped growing an inch a day, so I can stop mowing every few days and take a break to "mow" through all my new (?) S-100 systems. I'm still stepping over the systems I got from LAST YEAR's Trenton computer fest! You'll read more about the latest acquisitions - I mean donations! - in this column.

GIDE status

The other time-consuming activity in my life has been my correspondence with Tilmann Reh on the GIDE Z80 IDE plug in. You'll remember this is the little card that plugs into a Z80 socket and provides an IDE and time-of-day clock. Well, the latest word from Tilmann is that he plans a run of boards for early August, for availability in mid-late August. As I write this at the end of June it seems like a long wait. But, we must remember that this is a sideline activity for all involved; Tilmann and I are "donating" time from other, more profitable, work to tinker with this product.

As I've said to those wanting to order, this is a PROTOTYPE with no "ready to run" or "plug and play" software, so I will give ordering priority to those who can write assembly-language BIOS drivers for this card. You can get the specifications from the series Tilmann wrote in this magazine: I have arranged with both Tilmann and Bill Kibler (TCJ Editor) to offer reprints of these articles for a modest fee, probably $5 or so. So write and E-mail me, and get on the list; but tell me how you can contribute to the project too! Eventually, there will hopefully be enough drivers to support the major Z-80 based systems. AND, if there continues to be interest, an 8085 version will be developed. That will cover the Zenith Z-100, Compupro 8/16, and some of the IMSAI computers.

Let me remind all that Tilmann's prices are FOB Germany: I have to add import duty, shipping, repackaging and reshipping. Until I get a US license from Tilmann, I have to import the PC cards and PAL's from him.

So, between the mower and the modem, I've been off-line from my S-100 duties. Sounds like time to read the mail! But first, some bits of S-100 business:

IMSAI donation

Fred Hatfield of New Orleans LA has graciously accepted the donation of an IMSAI from Claude Kagan of NJ via the Good Doctor (me). Claude and Fred were colleagues at Western Electric many years ago, and both are collectors and users of old technology: Claude, also known for the SAM 76 programming language, is a collector of very early teletype and telegraph equipment; Fred is an old friend and occasional employer of mine, is known on Fidonet and Internet for his classic computer correspondence. Fred will refurbish the IMSAI and provide it to a local children's museum for display in Claude's name, soon as I check it out and ship it. Thanks to all and keep the faith!

Z-100 donation

Someone in the Internet news group comp.os.cpm was looking for a CP/M-86 for the Heath/Zenith Z-100. This is an S-100 bus system with a dual processor motherboard - 8088 and 8085 - which supports CP/M, MS-DOS and the Heath H-DOS and Z-DOS operating systems. It is *very much* like the Compupro 8/16 system I wrote about a few months ago, and I have a few of them with some docs. Mostly I turn to Lee Hart at TMSI and his "lending library" for documentation on any Heath system. Meanwhile, I put out a call for Z-100 software on the 'Net.

LtC Steven C. Marcussen, in the Washington DC area, responded to my email request as follows:

*> [Herb], you wrote:
I'm looking for a CP/M 86 for the Z-100. Much prefer ORIGINAL disks from Zenith, but will accept copies. A Zenith CP/M 86 manual would be nice also.
*>
*> I may have what you are looking for at home. I have a plethora of old Z-100 stuff, that I am looking for a good home. Just gotta make room in the garage. I have hard drive controller sets, spare video cards, ram cards, you name it, I probably have it. I have the multi-I/O cards if you are playing with serial port networking. Leave me Email at smarcuss@pafosu3.hq.af.mil if you are interested.

*>I am getting into Unix with AT&T 3B1s and 3B2s. I know I can't get much for this stuff, but am willing to barter equipment.

Steve, I'd be interested in CP/M 86 and CP/M 85 for the Z100. I have the 85 disks, but I'd take both disks and manuals. I don't have the 86 disks but have a copy of the manual (but an original would be nice). Mostly I need the stuff to support other Z100 users (as you might guess from my Dr S-100 title, I support S-100 stuff of all kinds, not just "Z's".) Some of the odd video stuff for the Z-

100, and maybe the serial port networking- someone just asked me about networking Z-100's.

*I've had some other offers, but I'd be interested in what you have. I can always run it in my S-100 column in The Computer Journal. As for trade, I do have some 3B1 and 3B2 software, I think, and some software manuals. I'd advise you to go the Linux route, but I'd certainly be interested in getting rid of *these* manuals! Why don't you give me a list of your stuff. And, more importantly, WHERE ARE YOU? Shipping can get expensive!*

Steve replied:
*>Like you are with the Z-100, I once was. I ran a BBS called the Jolly Green Giant on my Z-100 for about 8 years. I had a modified hard drive controller board that allowed me to run four 40 meg hard disks simultaneously, and a pair of 8" floppy disk drives for backup. I was a system administrator on a AT&T 3B2/600G for 2 1/2 years. I own a 3B2/300, a 3B2/400, and am presently in search of a 3B2/600G. I also own a pair of 3B1s and am bargaining for 3 more. I will be setting up a network in either 10BaseT or Thicknet at home, and teaching my kids about networking.

*>I am already planning a trip up to New Brunswick [NJ] to get a pair of 3B1s. I will scour out my stuff, but needless to say, it is not being used, my wife wants it out of the garage, and I would love to see it with someone intent on keeping the Z-100 going.

Well, to make a long story short, Stephen arrived about a week later with a truckload of Z100's! And six shelf-feet of Z100 docs and software! These parts are the remnants of his original purchase of 30-some systems, so I don't expect them to all work. But they include Gemini cards (improved IBM-PC compatibility), Heath hard-disk controllers, memory expansion, and Compupro I/O cards. And, you can't find a more completely-documented system *anywhere* in the S-100 (or PC, or Mac!) world. ROM listings, BIOS source code, hardware documentation, MAINTENANCE documentation, and more: all in the Heath tradition. It was a great loss to the technical hobby world when Heath died a few years ago (the Heath name was purchased by another company). Just for yucks I'll call Zenith sometime and see what they will still support for the Z-100.

Y'know, the military bought most of these machines, and they are turning up at military surplus auctions. So, they are ALSO appearing at hamfests and computerfests as "that old computer I got in the lot with the UHF transmogrifiers from the airbase" and are cheap to free! And, I took some time recently to compare the Compupro 8/16 CPU and the Z-100: they look very compatible, with a few provisions! I'll report more if there is any interest. *(See support groups for address of Z-100 Lifeline, a newsletter supporting Z-100's. BDK.)*

Mailbag: 19th century paper stuff

Leon Howell from Ashland OR writes:

"I am interested in upgrading my Northstar Horizon to a Horizon 8/16. What is the 16-bit CPU on the NorthStar 8/16 board? Can I use a 286 to 486 upgrade on an S-100 286 chip? Can I get an expansion box for an S-100? Since I don't have enough slots for the entire Microangelo graphics subsystem, I've been thinking of using my other Horizon for that purpose. What do you think?"

I'm not familiar with the 8/16 version, but I would assume it is an 8088 processor. And I have heard of people who have used 8088 to 80286 and even 80386 adaptors: I'm not sure they are using the '286 or '386 INSTRUCTIONS, but the 8088 code runs faster!

The S-100 bus is passive (that is, there are no electronics on the bus card) but it is not a good idea to extend it more than a few inches to another bus card. You COULD, I suppose, plug in an extender card and run another bus "above" the first or "behind" the first, depending on your bus speed. Watch that you have adequate power for all those cards!

Tim Deaton, Vice President of Central Computer Company in Chelbyville IN..

Tim told me in late April that he'd send a list of all the NorthStar equipment and related CP/M stuff they have. They've apparently been supporting NorthStar equipment and related old systems for years, and may be now selling off some of it. But I haven't heard from him since. Tim, what's up?

Scott Ellinport of Miami FL says:

"About 10 years ago I rescued a MITS Altair 8800 from my neighbor's trash pile. Recognizing it to be a historical piece of the computer era I have retained it to restore one day. While reading my magazine, I came across your ad. I also recall reading your Dr S-100 articles in my two sample copies of The Computer Journal. I'm interested in receiving your list of what you have available."

Well...I've been pretty busy Scott, but briefly I have a lot of docs for the Altair, and I can be helpful on boards and questions and such. I don't have Altairs for sale, but I can copy the docs for a per-page price. It would help me if you could tell me what boards you have: there are HUNDREDS of pages of docs for the Altair, so particulars would help.

Arthur H Smith of Alexandria VA writes:

"Thanks for responding to my email request [and sending the cards]. Do you have any Macrotech 286/Z80 cards? What chip speeds? Tested? Do you have any small S-100 chassis that will accommodate the Compupro 12-slot motherboard? If not, I'm going to go out and buy a "minitower" and install the S-100 motherboard. I really am tired of lifting computers with large transformers.

There seems to be a lot of [Internet] traffic regarding using 5-inch high density drives in place of 8-inch drives. I've tried this and the results were lousy. What am I doing wrong?

BTW, if you are ever in the DC area, [Jerry] Pournelle's old Compupro "Zeke II" is on exhibit in the Smithsonian (near a VIC-20, no less). They also have an

Altair, but no IMSAI. Thanks again for your help."

Anybody have some '286 cards for Arthur? And I'd like to hear about that "minitower" S-100 project! Of course, you can run S-100 cards on +5, +12, and -12 directly. Just jumper over the on-board regulators (or remove them and jumper). Compupro did this on many of their systems, and so did a few others.

20th century Internet mail

Joshua A. Barrett of Shawnee, OK - 30 miles east of Oklahoma City says:

***>I have 20 Z-100's - I would like some info on where to find hardware and software for them. I want to use them for network workstations.**

I think it not likely you'd find cheap network cards per se for Z-100's. You MIGHT be able to use some serial-to-network adapters, but you would need serial ports that are PC compatible, and the network software might choke on the Z-100's anyway. There are some generic serial networks out there, like "the Cheap Network" for $75, which only requires ONE purchase per site, that might network your 20 systems together. Let me know if you are interested.

I have some Z-100 info, but I don't have time now to copy all the manuals. Lee Hart of TMSI has some manuals too that you can borrow through the mail. If he is not in the CP/M FAQ, contact me for an address.

You know, I should also refer him to Rick Rodman's Tiny TCP/IP, as he wrote up a few months ago in TCJ! Hey Rick, what's the latest on this?

Josh's reply: "I have some books for them - I'll think about the network - Thanks"

From: John H Maxwell near Buffalo NY:

"Hello Herb,

You probably don't remember me, but I purchased a couple hundred 8-inch diskettes from you several years back for my Imsai 8080 system. I was living in Lockport, NY at the time and I believe that you were living in Colorado somewhere. I had spoken to you on the phone and we talked and you sent me a copy of the Morrow "Thinker Toys" 8-inch controller manual. Anyway, I was pleased to see your name in "The Computer Journal" when I had first subscribed and am still enjoying your column. I look forward to it every month.

This particular issue, TCJ #72, you mention a diagnostic tool that you use for troubleshooting S-100 systems. A Jade Bus Probe, is what I need to diagnose my Dynabyte and my other S-100 systems. You mention the Imsai's front panel to get an idea of what's happening in it, but is a pain to fire up two systems just to use one as a bus probe. If it wouldn't be too much trouble, could you send a copy of the schematic for the Jade Bus Probe? I don't know the current copyright status of such a device but I'd like to build one for use in my troubleshooting techniques. I have a few used proto-boards that I can add buffers and LEDs to. I was planning to use an extra board mounted to the top of the proto-board to mount the LEDs for ease of reading and labeling. I suppose I could homebrew it from scratch but if you have plans that you could share, it would save me valuable time between school and work and family demands.

Of course, I'm open to any other suggestions that you may have that I could try as well. I'm glad to see that you're still active in the S-100 field and perhaps I could 'bug' you again in the future to answer a few questions that I may have.

Thanks for your time and hope to hear from you soon."

John got the Jade docs some time later (much later, I'm afraid - sorry!). It's actually not that bad a circuit, just an AWFUL lot of chips to wire. People sure do like them blinkin' lights: should I go into production with these?

Gary Van Cott of Las Vegas NV has some stuff:

"I saw your name on the CP/M FAQ. I hope you don't mind me directing this question to you as I don't have full access to newsgroup. I have some S-100 equipment I would like to dispose of. It consists of an (upgraded) Vector Graphics MZ plus other equipment. It is old enough to be classified as a collector's item and I have never been able to bring myself to through it away. I would be willing to send this equipment to anyone willing to pay the shipping charges."

"Any assistance you can provide would be very welcome. Thank you." (See Help Wanted Ad from Mr. Dolan. BDK.)

John Henderson of Australia e-mails:

"I recently was given a Systems Group System 2900, sans manuals or any printed documentation. The system contains 4 S100 boards, 2 8" floppies, enormous power supply, 1/4" thick mother board, 4 RS232 I/F boards, and an Australian made "Computer Patch Board". "The S100 boards that make up the unit are:

CPU board - Model CPC2810 Rev D. The board has a Z80A CPU, CTC, PIO, DART and SIO. U16 (24 pin) is missing (boot EPROM?). It has a lot of jumpers and an 8 way DIP switch.

Memory board (?) - Model HDM2800 Rev B. This contains 18 4164's and a lot of logic. Again, a lot of jumpers, and several banks of links.

Disk Controller board - Model FDC2800 Rev D. This board is damaged. The small voltage regulators for +/- 12V have been fried. Also, 3 sockets are empty - U1 and U38 (16 pin) appear to be spares, and U10 (14 pin).

8 port Serial I/O board - Model INO2808 Rev C. Room for 8 8251 USART's. I have traced out this board and managed to get it working in another S100 system.

The "Computer Patch Board" seems to provide some sort of watch dog facility.

ANY information about the missing IC's (a source listing of the EPROM would

be wonderful), jumper/DIP/link settings of the above boards, or anything really, that helps me get this system up and running again will be very much appreciated.

Harold J. Hoover of Anchorage AK emails:

"If anyone is still interested in 8-inch floppy drives but doesn't want to pay California Digital's prices, I have the following USED but functional drives available:

TANDON TM848-1 SSDD 1/2HT 1Mb; and NEC FD1165F DSDD 1/2HT 2Mb

Looks like I'll still have over a half dozen drives left as well as some 24v power supplies and ribbon connectors.

Waiting for the latest issue of TCJ is half the fun of being a subscriber. :-)

"ALASKA! I can barely imagine the shipping costs!" However, I was assured by Ramon Gandia that US Mail costs to and from Alaska are actually reasonable.

"Mr. PDP" returns

[And here's some recent correspondence from my friend, John Wilson in Rochester NY:]

*>I just picked up the finest IMSAI 8080 that $25 can buy... I was wondering, since you deal regularly in such things, would you be able to sell me a photocopy of the documentation for this thing? I can't even tell how broken it is since I don't know how the front panel is even supposed to work, although it's certainly clear that the address counters have a few flaky bits in them and the data bus seems to be floating (or my RAM boards are cooked). Also do you know of a good source for replacement bat handles and/or entire switches? I'm missing the cover too but realistically, if I had it, I'd probably never put it on!

This sound like every other IMSAI owner in the world, so congratulations! There is a basic set of manuals for the front panel, cpu, power supply, box etc. When I make another copy of them, I'll quote you: I'm very busy currently. Figure $30 or so.

*>The thing has no disk controller of any kind... Naturally I plan to change that, but meanwhile, is there such thing as a "standard" config for this machine?

No.

*>It's got the usual array of totally unrelated cards... The CPU is IMSAI, there's an IMSAI "VIO" board which I'm guessing is some kind of video board? Then a couple of RAM boards by Processor Technology, as well as a pair of 3P/1S boards. Then an 8x1702 ROM board that's been modified to be 1x2708, and an unspecified "8800 INTERFACE" board with a Berg connector on it.

You'll have to be specific on the card names, but I know them anyway... If you give me names I can sell you docs.

The CPU is probably MPU-A (8080) or MPU-B (8085). The VIO card is an 80 X 25 character display, or some dot graphics. The RAM cards are as you've noted. The 3P+S are Processor Tech also, very popular.

*>I haven't had time to dig up an NTSC monitor and plug it into the VIO board to see if anything comes up (I've been in Boston fixing my minivan, I'm only back in Troy for a day to pick up a piston and some other parts), with all the ROM on this thing it wouldn't surprise me if there were something non-trivial already loaded...

Probably BASIC or a register-level monitor..

*>The guy also gave me a totally bizarre word processor/data base machine that he thought was the terminal, but I don't think so. I think the brand was Northern Telecom, anyway they were in MN somewhere. It has a keyboard very similar to an IBM 3277 terminal, not surprising since the manual says there was an emulation package for that. It has an 8085A with 64KB, and an external box with four 5.25" floppy drives in it. Funny looking machine, in profile everything is horribly curved, like an ADM3A, but black.

Give me more info, maybe I or my readers can identify it for you. John later wrote:

Re GIDE board — so what is the final price going to be? My application would be for a TRS-80 Model II, I'm not sure that the world is dying for an IDE BIOS for that machine, so maybe I should wait. Also I don't have BIOS sources for that machine, although it might be fun to start from scratch anyway, I've already written some FDC code and the video hardware is straightforward.

Meanwhile I'll continue with my own nefarious plans, as soon as I get this S-100 machine going. $30 sounds fine, let me know when you have time to run off a copy. My CPU is the MPU-A board. WRT the other boards, I think the description I gave you was as complete as it can be, i.e. not complete enough. "3P/1S" appears to be the actual name of the Processor Tech boards, the ROM board has no brand or model at all, the "8800 INTERFACE BD" has no other markings besides a digit string (P/N?) and address ranges printed next to some ROM sockets, and the IMSAI VIO board just has the rev and the "VID" output (funny, "VIO" implies input too). FWIW, the power switch has a sticker on it saying that it selects between FDOS and CP/M, and anyway it doesn't work as a power switch."

MITS 20th reunion celebration?

*>The following message is forwarded from CPMFORUM by donm@crash.cts.com (Don Maslin)

*> 08-Jun-95 04:46:08
*> Sb: Altair/NM Comp Faire
*>
*>I read somewhere recently that there will be an Altair convention at the New Mexico computer Faire & Expo starting June 10. I'm interested in making contact with this group, but the phone number that I got for the Expo is wrong. Can anyone get me an accurate number or tell me how to contact someone involved?
*>

*>I have an Altair 8800B and am desperate for manuals and OS and hope to make contact with some via this convention.
*>
*>Thanks, Mark Payton 102056.3342@compuserve.com

I have some Altair manuals I can copy for you, but my time is very scarce so it may take awhile. Typical cost to you is 15 cents/page, plus shipping; payable by your check sent to me. E-mail me if interested and include your shipping address. I have a list I can send you.

I recommend my article series in The Computer Journal, by the way.

Further reports on this convention trickled in:

From: shoppa@almach.caltech.edu (Timothy D. Shoppa) Newsgroups: comp.os.cpm,alt.folklore.computers
Subject: MITS (Altair) 20-year reunion
Date: 15 Jun 1995 07:36 PST
Organization: California Institute of Technology

The "Business" section of Wednesday LA Times had a good-sized spread on the 20th reunion of the employees of MITS. There were a few really glaring errors, such as referring to the Altair's CPU chip as being a "Intel 8800" and even a quote which said, in effect, "The MITS Altair was just like the IBM PC, but much earlier". Someone who didn't know what the Altair was would get the impression that you just ordered one from MITS, plugged it in, and you were up and running WordPerfect and Lotus 1-2-3, I'm afraid!

Even worse, I don't think the article did any justice to the community of microcomputer hobbyists which, IMHO, the Altair represents. There are a few paragraph about business users of the Altair, but almost nothing about the hobbyists! And there is *nothing* about the whole industry of Altair-bus (S100-bus) compatible board manufacturers which sprung up in garages and which drove the early microcomputer business.

That ranting and raving over, I'll ask if anyone can give a first-hand report of their attendance at the reunion.

Anyone? Anyone?

Doug Jones <jones@cs.uiowa.edu>
Date: 15 Jun 1995 15:36:11 GMT
Organization: University of Iowa, Iowa City, IA, USA

[quoted by permission]

Dunno. I helped build Altair 8800 serial number 39, (if my memory serves me correctly), and we were no hobbyist group.

Number 39 was purchased by the U of Illinois Medical Computing Lab. We needed more memory almost immediately, so we wire-wrapped our own memory board. We also made a quad UART board that way, and two custom parallel interface cards.

The machine served us for years, serving as a code converter behind a pair of PLATO IV student terminals (512x512 dot addressable plasma display panel technology). The Altair served as a protocol converter, speaking a sensible ASCII-based protocol to our minicomputer and looking like a pair of non-standard PLATO uarts to the PLATO terminals (those wonderful monsters wanted 21 bit packets, counting start, stop and parity bits, leaving 18 data bits per packet — either 3 characters 6 bits each, or 2 screen coordinates 9 bits each.

By design, the Plato terminals could handle 60 packets a second, which translates to 180 characters a second, but with our Altair based interface, we found that we could drive them significantly faster — we used a pair of 2400 baud lines between the Altair and the host minicomputer. The Altair had no problem at all keeping up with 2 bidirectional asynchronous data streams plus two parallel terminal interfaces.

As a result of this experience, the Mecical Computing Lab got into S-100 systems in a big way, helping various research groups use them as lab minicomputers. One IMSAI machine, for example, went into a population epidemilolgy study where there were two colonies of 1000 mice where each mouse had to be weighed daily. The machine was interfaced to a scale and a microphone, and we used some crude voice recognition software to speed data gathering. The operator would read the mouse's serial number out loud while bringing the mouse to the scale, then weigh it, then return it to its cubby before getting the next one. Our worst problems there were with escaped mice getting into the cableways.

The whole point is, lots of those machines that were nominally made for hobbyists went into non-hobby applications, and this happened from the very start. There were plenty of minicomputer users who'd cut their teeth on PDP-8, PDP-11 or Nova systems who immediately recognized that machines like the Altair 8800 were the obvious next step in low-cost laboratory automation. The IMSAI machine was even better, being built to hardware standards that met industrial expectations.

References:

Herb Johnson, CN 5256 #105, Princeton NJ 08543 609-771-1503 <hjohnson@pluto.njcc.com>

Lee Hart / TMSI, 4209 France Ave N., Robbinsdale MN 55422 612-533-3226

Leon Howell, 493 Clay St Apt 6A, Ashland OR 97520

Tim Deaton, Vice President of Central Computer Company, 1641 S Riley Highway, Chelbyville IN 46176

Scott Ellinport, 17815 N E 7th Court, North Miami Beach FL 33162-1103

Arthur H Smith, 1808 Jamestown Rd, Alexandria VA 22308; or ahsmith@delphi.com

Joshua A. Barrett of Shawnee, OK - 30 miles east of Oklahoma City, E-mail: barrettj@mailhost.qns.com (Joshua A. Barrett)

LtC Steven C. Marcussen <smarcuss@pafosu3.hq.af.mil>

John H Maxwell <maxwell@cadman.cit.buffalo.edu>

Gary Van Cott, P.O. Box 9569, Las Vegas, NV 89191

John W. Henderson, hendersonj@msmailnt.bhpese.oz.au

Harold J. Hoover, ASHJH@Orion.Alaska.edu
2440 East Tudor Road, #870; Anchorage, AK 99507-1185

John Wilson <wilsonj@rpi.edu>

TCJ Center Fold

Special Feature

All Users

S-100 Power Supplies

In Herb Johnson's Dr S-100 column, a letter about removing regulators from S-100 cards prompted me to feature the S-100 power supply. I have talked before about these wonderful old monsters with affection, since they work much better than switching supplies. The reasons are many and as you look at the schematic their simplicity certainly stands out.

This design was cast back when switching supplies were just being thought of and cost millions of dollars. Also the early S-100 systems needed lots of power to handle all the cards. A system in the mid seventies would have eight or ten cards for a minimal system. The 8V supply would need to provide at least 20 Amps for such a simple system. The IMSAI manual says typically 50 Watts is used, but their power supply provides 500 Watts, just in case. I saw many systems working as severs running MP/M that had every slot filled. Using 30 to 40 Amp peak was not unusual.

The cheapest way to obtain this much power was from an unregulated supply and let each board do their own regulation. We need to consider as well that transistors that could be used for regulating 20 or 30 amps might have cost as much as the whole supply, were they available at all. Having separate regulators also allowed each card to only filter and regulate those voltages used. In later years some cards only used the 8V supply for 5V was the only voltage needed.

The reason I love these supplies over switchers is their reliability and reserve power. I once saw a system that could save all information in memory to disk, before the capacitors allowed the voltage to drop off should the power go out. If you look at the schematic you will see that we are talking big, and I do mean big capacitors here. We are talking 1/2 a Farad on the 8V supply in some cases. Typical and those shown here are in the 10 to 40 thousand micro farad size, with 100 thousand micro farads (.1 farad) on the 8 Volt supply.

This monstrous amount of capacitance helped smooth out and stabilize the supply voltage and as I have seen, pretty much kept the systems running through most noisy power line problems. The early switcher supplies often passed noise spikes through the supply and right into the computer, often shutting them down or at minimum resetting the system for you, whether wanted or not.

The Parts

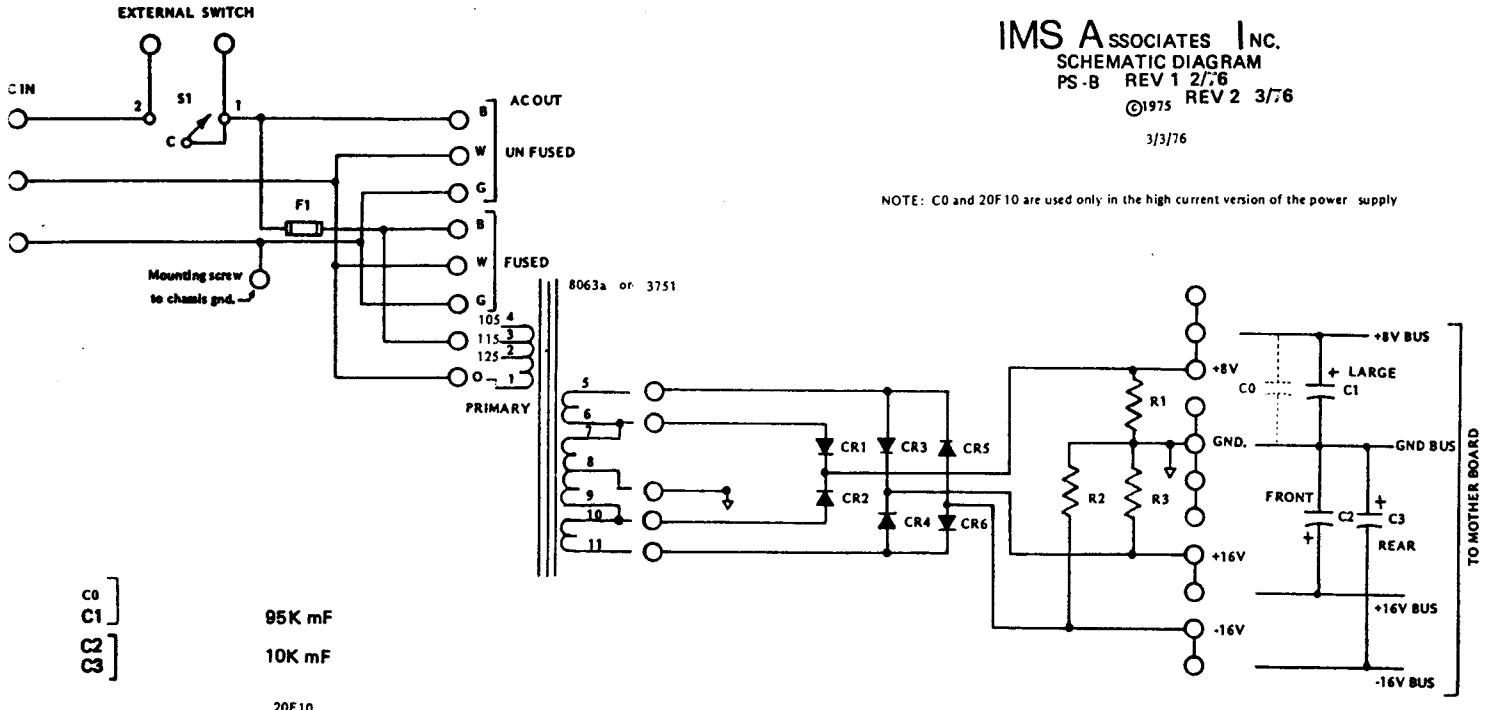
The parts used were the simplest but not necessarily the cheapest. The transformers contain considerable amounts of iron and copper and now are rather expensive. I believe that Godbout used ferro-resonance transformers in many of their later units (that is the winding 'R' in the drawing hooked to a capacitor). What that means is the winding and laminations of the transformer are done in such a way as to provide a constant output voltage even with a large variance in load. Without the better transformer, the 8 Volt supply typically will bounce from about 8 to as high as 10 Volts. Don't let it get too low, as the 7805 regulator stops regulating at about 7.5 volts.

Your on board regulators will just turn the extra voltage into more heat. If your cooling is rather poor, this extra heat however could cause minor problems, so it is best to not let it get too high. The drawing shows that several taps were typically provided to give you minor control over the supply output voltage. Should it be high, just move the AC tap to a higher voltage setting, which is actually less windings (125). If low, move to a lower input voltage option (105).

The early supplies used discrete diodes, while the later units went to diode bridge "cubes'. These cubes contained four diodes arranged in a bridge rectifier pattern with a center hole for mounting and large spade lug terminals for hooking wires on. I had one system that used diodes in a fuse like holder since they are carrying 20 to 30 amps of current. Remember, that high of a current requires larger wire sizes, typically a minimum of #10 is needed. Now that goes for hooking to the bus, you need a good clean and large surface area for passing that much current. Most good systems had more than one connection and soldered or used soldered on spade lug connectors.

What can I say about the capacitors, they are big! The size was typically three to four inches round and often that high as well. The voltage ratings are typically rather close to the supply rating, since more of either voltage or farads runs the size and price up. I don't think I ever found a bad capacitor, although I did have a few problems with connections. Most have 10/32 screw terminals and some over tightening might break them off, but short of that they will last almost forever.

Continued on page 28



C0 C1	95K mF
C2 C3	10K mF
CR1 CR2	20F10 OR MR 1121
CR3 thru CR6	MR501
F1	5-6 Amp Fuse
R1	470 ½W
R2 R3	1K ½W
S1	SPDT (Not Supplied)

All D.C. Wiring:

Black	D. C. Ground
White	+8 Unregulated D. C.
Orange	+16 Unregulated D. C.
Yellow	-16 Unregulated D. C.

All A.C. Wiring

Black	A. C. High Side
White	A. C. Neutral Side
Green	A. C. Ground
Brown Pair	AC for Fan (should be twisted)
Brown Pair	AC for Switch (may be twisted)
Brown Pair	AC for Primary to Transformer

DO NOT connect AC Ground to DC Ground in this machine.

IMSAI Mainframe Modification

Bus pins 20 & 70 – The IEEE 696 standard states these pins should be at ground, but if you try to put a **CompuPro MotherBoard** with these pins grounded into an **IMSAI type mainframe,** the front panel stops working. This occurs because there are some front panel signals pulled up with a pull-up resistor, which also connects to bus pins 20 & 70. Grounding these pins pulls down the front panel signals and disables the front panel. The solution is to disconnect the front panel signals that connect to these bus pins. This fix *must be implemented on the front panel itself!* There is no reason to have them connected, and the panel will still function properly.

AC Power Distribution

The System Enclosure is color coded to provide easy indentification of voltages. The following drawing and schematic can be used to identify any system power problems.

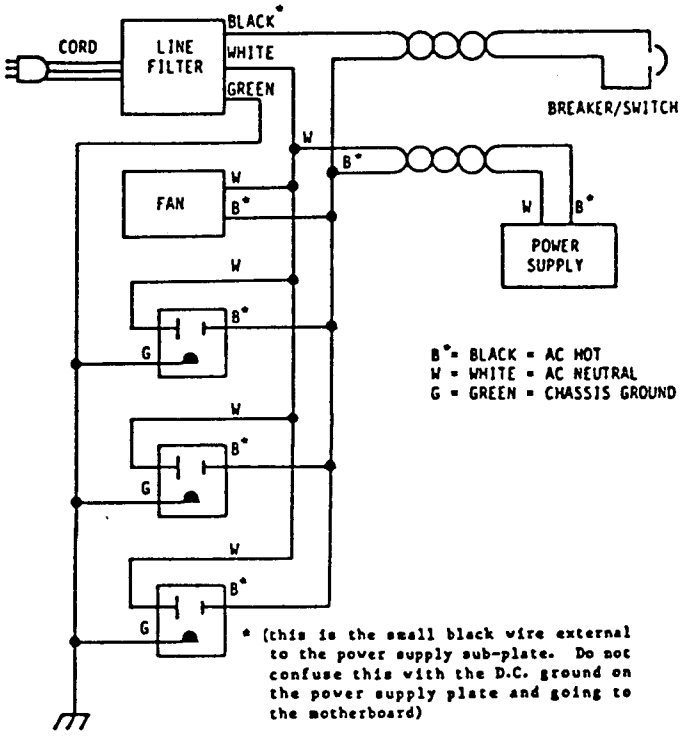


Figure 1. AC Power Distribution

CAUTION: As with any electrical device which plugs into the wall there are voltages present that can cause a bad shock or possible death. The power supply should not be touched except by experienced technicians. The oval capacitor near the transformer has very high voltage on it and should be avoided at all times.

If there is any reason to put your hand down into the power supply area, make sure the system is off, unplugged from the wall, and let set for at least 5 minutes to let the stored charge bleed off. Remove any rings, watches or other metal objects before reaching carefully into the power supply area.

NOTE: For best system cooling, always have a board in the rearmost slot, and keep the cover on. **To ensure proper cooling and electrical safety, never operate your CompuPro system with the cover removed.**

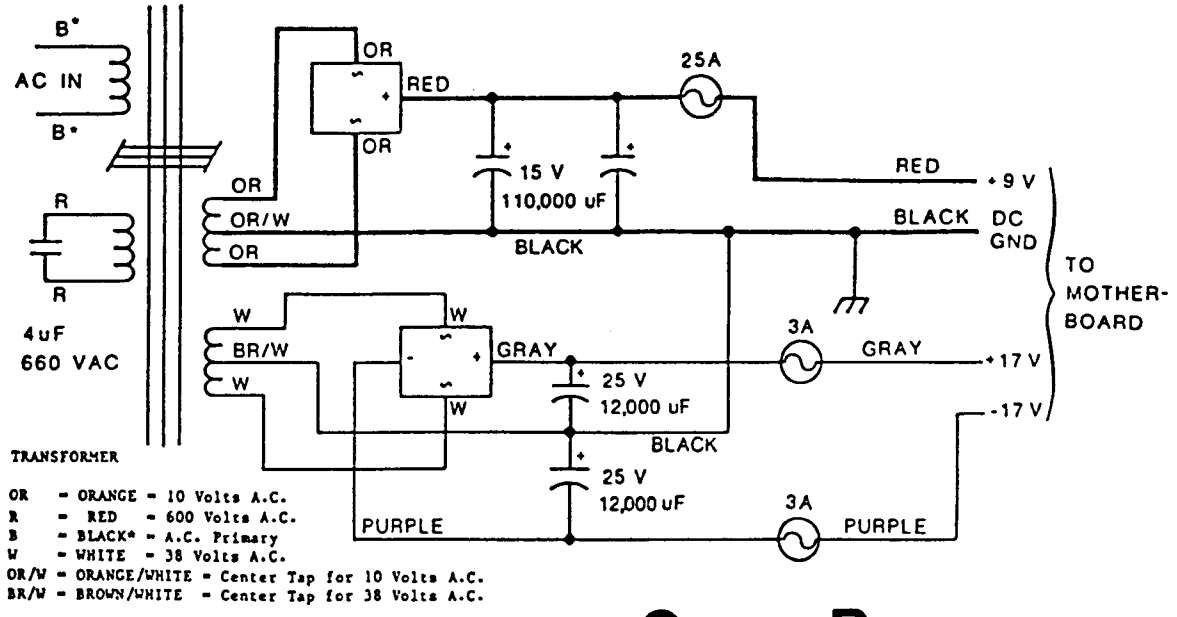


Figure 2. Power Supply Schematic

CompuPro DIVISION

GODBOUT ELECTRONICS

To Switch or Not

As time went on, the loads needed decreased and systems of two and three cards became more normal. Switcher supplies have certainly improved both in cost and filtering. Size of switchers are certainly in the micro range and the need for the 10 to 20 pound transformer used in S-100 systems is questionable. A major problem in the early systems was heat from all the regulators. If your system was full, winter heating was not needed, simply run the S-100 full time, and your room remained comfortable through even the worst of blizzards.

So how do you change to a switcher supply if you want to go non-standard S-100. Well first would be removing the power transformer and capacitors. This typically took up one whole side of the case, from front to back. Place your switcher supplies there and hook them to the BUS. Guess that each card will need one or two amps each (two to three to be safe) and adjust the switcher supply and wiring accordingly.

The S-100 cards will need the regulators removed, but not the capacitors. Sending regulated voltage down the bus, especially a 18 slot bus, opens them to all kinds noise. Those remaining filter caps will hopefully clean the power sufficiently to prevent false operations.

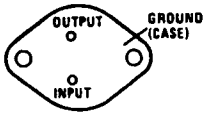
Now you need to watch out on the pin arrangements of the regulators. The pin numbers and functions vary and you will find the negative regulators different. The idea will be to run a jumper wire of reasonable size from the input to the output pins. The third pin is ground and can be ignored.

Now how much power is this in Watts? Remember that Ohms law for power is I times E. So we take the volts, say 5 Volt and 3 amp for each card to be safe, and lets say a simple system of five cards. That means 3 amp times five (cards) times five (volt), or 75 watt, or a 5V/15 amp supply. You will need to do the same for the plus and minus 12V supplies. Those switcher supplies can be had new for about $125 to $200, although used switchers often go for less than a S-100 transformer if you could find it.

When figuring out the cost and size needed to use a switcher, I would not do it for a large system. Should you want to use one or two cards for a dedicated purpose, a small switcher might make sense. For large systems, the big transformer is still the most economical and reliable choice.

Connection Diagrams

TO-3 Metal Can Package (K and KC)



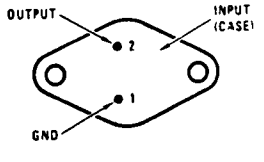
TL/H/7781-11

Bottom View

Steel Package Order Numbers:
LM140AK-5.0 LM140K-5.0 LM340AK-5.0 LM340K-5.0
LM140AK-12 LM140K-12 LM340AK-12 LM340K-12
LM140AK-15 LM140K-15 LM340AK-15 LM340K-15
See Package Number K02A

Aluminum Package Order Numbers:
LM340KC-5.0
LM340KC-12
LM340KC-15
See Package Number KC02A

TO-3 Package



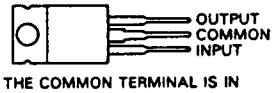
TL/H/7340-10

Bottom View
Order Number LM7905CK, LM7912CK or LM7915CK
See NS Package Number KC02A
TO-220 Package

NOMINAL OUTPUT VOLTAGE	REGULATOR
5 V	uA7805C
6 V	uA7806C
8 V	uA7808C
8.5 V	uA7885C
10 V	uA7810C
12 V	uA7812C
15 V	uA7815C
18 V	uA7818C
24 V	uA7824C

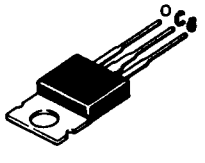
KC PACKAGE

(TOP VIEW)



THE COMMON TERMINAL IS IN ELECTRICAL CONTACT WITH THE MOUNTING BASE

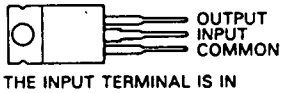
TO-220AB



NOMINAL OUTPUT VOLTAGE	REGULATOR
-5 V	uA7905C
-5.2 V	uA7952C
-6 V	uA7906C
-8 V	uA7908C
-12 V	uA7912C
-15 V	uA7915C
-18 V	uA7918C
-24 V	uA7924C

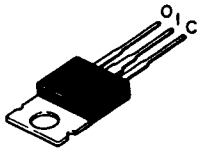
KC PACKAGE

(TOP VIEW)



THE INPUT TERMINAL IS IN ELECTRICAL CONTACT WITH THE MOUNTING BASE

TO-220AB



Palmtech CPUZ180

By Ramon F. Gandia

Special Feature

Classic Support

New CP/M Machine

CHECKING OUT THE PALMTECH CPUZ180

Two simultaneous programs and hundreds of hard disk megabytes is no longer exclusive to the PC and Windows. It is here today for your favorite CP/M programs. Terse descriptions of the CPUZ180 board in earlier issues of TCJ and the Z-Letter did not do justice to the capabilities of this powerful board.

Contained in a 4" x 6" pcb, the footprint of a 3.5" disk drive, there is

18 mHz Z180 cpu, which runs 8080/Z80 code
Task switching ZCPR3 clone DOS
2 external serial ports
1 Centronics printer port (usable for i/o)
3 8-bit i/o ports (8255 variety)
one bit "speaker" or LED port
Floppy disk controller
IDE hard disk interface
Flash EEPROM for DOS and ROM-disk
Two channel timer-counter and watchdog
Battery backed real time clock
128K static RAM
Interface for PC XT keyboard
Interface for an MDA, CGA or VGA PC monitor

Software and documentation is supplied on disk and Flash memory. Among the supplied items were the IDE/Floppy drive setup utilities, CP/M compatible ZCPR, Z180 assembler, debugger, etc. While at first blush the board appears expensive, it is remarkably complete and uses standard and inexpensive PC peripherals.

Although I mounted it in a diminutive 3" slimline PC case, most of the cabinet remains empty. Wiring is mostly ribbon cables and it only took me an hour or so to make the cables and have everything ready to go. Power consumed is 5 volts only at 400 ma. No 12 volts or negative voltages are used by the board, although the external disk drives do use 12 volts.

The overall "feel" of this machine is impressive. When something is entered on the keyboard, there is just a faint "tick" from the IDE drive, and Presto! program loaded! Sitting at the console is akin to driving a Jaguar. The CPUZ is incredibly fast, smooth, comfortable and most definitely a muscle car. Lets take it for a test ride before opening the hood.

Test Driving the CPUZ180

Two TPAs are a tremendous productivity tool. You could run an editor in one, and a spreadsheet in the other. Write a letter to your accountant, flip TPAs with the hot key, look at or manipulate the spreadsheet, and flip back to your letter. It is not true multitasking since only the "foreground" TPA is executing. With the other TPA in hot RAM (not in a disk image as per older computers), task switching is instantaneous and eminently practical. Each TPA's video display is preserved during the switch and reappears exactly as you left it.

As shipped, the CPUZ came configured for 1.44mb 3.5" drives and a different IDE drive than the one I had. The PDRIVE utility allows setting up any of six active logical drives and I assigned the 3.5" floppies as drive E and F, and portions of my 106 MB Seagate drive as A,B,C,D. (A seventh drive, Flash Disk O: is always available). First, I ran HDID to get the hard disk parameters:

1024 cylinders
17 sectors per track
512 byte sector size
12 heads

With the PDRIVE utility I partitioned my hard drive into ten 8-mb partitions, the largest allowed under CP/M. I could have squeezed twelve partitions, but I decided to have the partitions on 100-cylinder boundaries for simplicity's sake. The PDRIVE utility is very easy to use and performs automatically all the tedious mathematics needed in older computers. Further, the definitions can be changed on the fly, and there is nothing to reassemble. This is a far cry from the traumatic experience of adding a hard disk to a normal CP/M computer!

In my case, all I had to do was decide on directory and group size, and which partition gets assigned to which logical drive. Of the six allowed drives, two were floppies, so I assigned partitions starting at cylinders 0, 100, 200 and 300 to logical drives A, B, C, D. However, at any given time any partition can be reassigned to any logical drive. Thus, little used backup data can reside in a partition at 800, and when needed, that partition can be renamed drive C: In effect, this means that my on-line disk capacity is 32-Mb, but the remaining 48-Mb is only a few keystrokes away.

Using an IDE drive under CP/M is simply breathtaking. All

disk operations are a magnitude of speed (10X) faster than with floppies, particularly first access to a file. Needless to say, IDE drives are much more reliable than floppies, and makes everyday use of the CPUZ a pleasurable reality.

Although not specifically marketed as such, the CPUZ is the fastest Z180 machine out now. At 18 mHz, it is even with a 386SX-33 computer, each running comparable assembly language code. I wrote up a simple assembler program containing four nested loops; the innermost having a few typical instructions such as rotates, pushes, pops and logical operations. I wrote the program in 8080 code for the CPUZ and in 8088 code for the PC. Neither program was optimized for its native processor, but rather to have basically identical instructions. Then I ran it in four different computers: The CPUZ180, a 486SLC-66, a 386SX-40 and my trusty Altair with a Cromemco 4mHz Z80 board. In addition, I also ran the 8080 coded program in the 486SLC-66 with the 22NICE CP/M emulator. Results were:

486-66 PC	running 8088 machine code	1:47
386-40 PC	running 8088 machine code	4:15
CPUZ180	running 8080 machine code	5:28
Altair	running 8080 machine code	16:00
486-66	running the 22NICE emulator	16:52

The above times give a good approximation of each machine's raw computational power. Unfortunately for the PCs, most of today's software is written in C or other high level language and under a typical application like a word processor or spreadsheet, the CPUZ is much faster.

CP/M VEDIT can be loaded, simple editing done, and the program exited before the PC can even load Microsoft Works for DOS and it's text file. Last place goes to Works for Windows where both CP/M and DOS programs can load, edit and finish while Windows still grinds away. It is sad to see the PC programming community so waste the capabilities of their hardware by using inappropriate programming languages and a slow, cumbersome operating system. Lack of assembly language programming skills has resulted in a generation of functionally illiterate programmers.

Booting is equally quick. It is not uncommon to be already doing useful work with the CPUZ while the PC's are still churning through the boot process. Us older folks do not want to waste our golden years waiting for a PC to boot, so this is an endearing quality!

The supplied operating system is Z-system compatible, and of course runs CP/M programs. Enhancements are many. As supplied, the operating system has the aforementioned twin TPA's and task switching at the stroke of a hot key. Repetitive operations can be automated with ALIAS, and HELP is built-in. All common CP/M and Z-system commands work as expected, with a few exceptions. In those few cases, the enhancements were deemed more important than strict compatibility. A CP/M TYPE would merely result in a blur of text, so PAGE is supplied, which allows a file to be viewed bidirectionally a screenfull at a time. The Console, List, Punch or Reader can be operated as 7- or 8-bit devices, allowing for graphics.

The CPUZ180 as an industrial controller

The CPUZ is unique in that it is a complete controller, operating and development system. Thus, it is possible to write, edit, and debug not only the industrial control process but the program as well, all within the CPUZ. When everything is running properly, any unneeded disk drives, keyboard or video monitor can be disconnected and the machine will power up on your FLASHed program.

For trivial or simplistic applications, it is hard to beat the new Basic Stamps or PIC controllers. However, those small controllers, as well as the crop of 8051's etc. out there all require a PC to develop and download the software to the controller. Then, if the controller does not do its work, you somehow have to determine if the bug is in your program, the algorithms for the control process, or just a loose wire. Without an operating system in the controller, the PC is nothing more than a one-way loader. Debugging often is tedious trial and error, often involving dozens or hundreds of EPROM burns.

With the CPUZ, you use familiar CP/M tools such as your favorite assembler, DDT, etc not only to debug your program, but also to patch, write and develop it. The supplied debugger, ZBUG, allows you to monitor memory and I/O ports in real time and in full screen mode, thus you can see the industrial process at work. Time consuming and expensive development time is cut to a fraction.

The CPUZ180 does not have A-D converters built in, all of its I/O is digital and would require the use of a one-chip A-D converter to read analog signals. However, the digital I/O is very generous; not only is the 82C55 with its 24 I/O lines available, but so are the one bit "speaker" port, the centronics printer, and possibly even the IDE port.

Under the Hood

The cpu is the Zilog Z8S180 (nickname Z180), a licensed variant of the popular Hitachi 64180 but with 20 address lines for a possible 1 megabyte of memory. This is a Z80 with built-in memory management unit, or MMU, that maps portions of the address space into the logical 64K that its Z80 core executes. To round out the device, the engineers threw in two async serial ports, a synchronous serial port, and a DMA controller. All of these devices are implemented in the CPUZ180.

The two async serial ports are free and available for use. Software provided allows them to be setup as Reader, Punch, Printer, Modem link, in addition to default parameters of baud rate, parity, number of bits and handshaking status. The synchronous serial port is devoted to the PC XT-style keyboard.

Two Dallas Semiconductor chips handle reset/power monitoring and Real Time Clock. The purpose of the reset chip is to

provide a clean reset signal after power stabilizes. This prevents glitches or multiple resets, which might be important in an industrial controller environment. The RTC chip allows for a time/date/calendar function, countdown timers and a watchdog timer to oversee such functions as the floppy controller. Thus a bad floppy does not hang the machine, but allows you to recover gracefully.

Video is from the CGMA chip. Akin to the 6845 video controller found on PC's, it generates characters, attributes, etc. 64K of special video RAM is dedicated to it. Although a very capable chip, programming it is not for the faint of heart. A custom Palmtech PLD provides CGMA clocks, and external interface to MDA, color or mono CGA, composite, or VGA monitors. Sync polarity, video rates, refresh and colors are all programmable with the SV utility provided, and virtually any monitor will work. Claude cautioned me that the VGA is untested because, in his words, "my Multisync displays anything thrown at it, so its no real test!" In my case, I connected a mono MDA monitor, and it worked perfectly. For MDA or CGA monitors, the ribbon cable connectors clamp on directly, whereas with VGA there may be some wires to change.

Video is crisp and snappy. Mine is set up for conventional 80 x 24 display, with a 25th line at the bottom showing the time, date, TPA in use, program "playing" at the time, etc. This line can be turned off via software. Unlike PCs, the clock is always dead on time. It is adjustable in software and also by a trimmer capacitor. Quite nice!

The IDE interface is a PLD device, which also provides a Centronics printer port. This port has full printer handshaking, and all functions and lines are implemented and supported. A jumper allows it to be configured as a bi-directional port, but Claude didn't mention that feature, and I didn't ask!

The IDE section is interesting. IDE drives are 16-bit devices, whereas the Z180 is an 8-bit cpu. The IDE chip takes the 16-bit word and shuffles it into two consecutive 8-bit bytes for the dma controller. Thus the IDE 16 bit stumbling block is overcome and is transparent to the user. Any IDE drive will work. My choice was a Seagate 3120A which was on sale for $100. With falling prices, it might not be inappropriate to connect a 540-Mb IDE drive! Again, I didn't ask and Claude didn't say, but I see no reason why two drives cannot be connected in a Master-Slave configuration for 1080 Mb on line.....

Besides their great capacity, speed and low cost, IDE drives are quite uniform in characteristics. Due to the overwhelming PC market, they are all preformatted to 512 byte sectors. To my knowledge, most are all hard sectored with servo tracks and never need low level formatting. An IDE format operation merely fills the 512 data bytes with E5h. This in effect empties the disk but does not affect the inter sector gaps etc. Older MFM and RLL drives with step motor heads generally "aged" and as their mechanical parts wore, the adjustment would drift off. Therefore it was essential to low-level format these drives from time to time. This limitation does not apply to IDE drives. IDE heads are voice coil operated, and they seek the track until they find it without much regard for physical wear. Time and experience has proven that this is the correct approach, and barring a case of "infant mortality", your drive will live practically forever.

The Floppy controller is an implementation of the NEC765 variety. These can read and write 3.5", or 5" drives in single, double or quad density, single or double sided. Due to a limitation in the disk controller chip, single density disks cannot be formatted, although they can be written or read. The CPUZ180 software accesses all sectors in sequential numerical order and cannot understand logically skewed disks. This means that CP/M disks with logical interleave, eg, the first sector written is 1, the next 6 etc. are NOT supported. What IS supported is physical interleave, where the sectors in the disk are physically in any order desired. This is not much of a limitation because the vast majority of disk formats with sector sizes of 256 and up have consecutive sector access. Optimum physical interleave in the CPUZ180 is 4.

No mention is made of 8" drives; my impression is that they can be connected if a little ingenuity is exercised in making up the ribbon cable. Data rates are software configurable, so it should be possible to utilize single and double density 8" drives. Be forewarned that 8" drives predate the IBM PC and come in a variety of configurations, with sometimes a score or more option jumpers, etc. You are on your own when it comes to eight inchers.

Parallel ports are from a garden variety 82C55 PPI. The three 8-bit ports can be configured in a variety of modes such as input, output, bidirectional, bit programmable, or use port C for handshaking. The PPI is fully available to the user as the CPUZ180 makes no use of it. There is plenty of literature on the 8255 PPI chip, and it remains the most popular of its type. If the CPUZ180 is to be an industrial controller, obtain a book explaining the 8255 first.

There is also a "speaker port", actually a single transistor switch, suitable for driving an LED through an external resistor. Claude says that beeping speakers annoy him, and he never implemented the software to generate the tones. He uses the LED.... Being a right-winger, I agree with him. If you absolutely have to have noise, use the transistor to drive a relay and have it ring a doorbell. Or treat it as a one-bit output port and use it to turn on the coffee percolator.

Z180 Memory management is complex, and the CPUZ180's drove me to distraction. I can guarantee that unless you have a high IQ, full understanding will challenge you. The following description is simplistic, but will give you a good idea of the inner workings of the Z180 in the CPUZ180.

CPUZ memory resides in three devices: the 64K video memory, the 128K Flash EEPROM and the 128K RAM chip. There is the option of using larger chips and ending up with 512K of

RAM or EEPROM, but the software does not support it, so you are on your own if you want to use the larger chips. Have no fear, the standard memory is ample, specially considering the IDE drive on-tap. There would be no need to use a RAM disk as the IDE drive is for all practical purposes instantaneous.

Like any Z80, reset lets execution begin at 0:0000 in memory and the Flash chip is there. Thus the Z180 comes up on Flash and does its cold boot. The cold boot rearranges the memory map with help from the Z180's MMU. When the dust settles, the A0> prompt appears, and the Z180 is looking at 64K of RAM. On a standard 64K CP/M computer with a minimal BIOS, the CCP is at DC00, whereas it is at F000 on the CPUZ180. TPA is thus 4.2K larger. A DOS or BIOS call to high memory merely sets up the MMU, and the Z180 jumps outside the basic 64K map. When BIOS or DOS gets done, the MMU rearranges things back to normal and the application program is reentered.

This is done transparently to the user, with perhaps one noticeable exception. A DOS call 1Bh, Get Allocation Vector, does not return meaningful data. Fancy directory programs will thus not show actual disk size and space. This is no hardship as the CPUZ180 comes with a very good version of XDIR, specially tailored to this machine. One of the things that Claude and I straightened out was this particular program. I thought it was too fancy, but the latest version has had all the fat trimmed from it. It displays the file name, size and attributes, plus a title showing the drive and user area/name and lastly the number of files in that directory, in the whole drive and the free disk space.

Upon switching to another TPA, the MMU takes the first 60K of RAM (0000-EFFF) and switches it out to standby, and the other 60K is switched in. This happens at electronic speed; there is no wait like on older systems that saved the TPA image to disk and then brought another one in.

The upper 4K of RAM memory area (F000-FFFF) is public and does not get switched around. Besides containing jump tables and MMU setup data, there is user available RAM areas, environment tables, TCAP support etc. Don't forget that the CPUZDOS is very like ZCPR3 and fully supports Z-system programs.

Where memory management gets arcane is in the programming of the Flash memory chip. Some of the common operations, like saving PDRIVE and SETUP data to Flash, as well as the use of the "Flash drive" (O:) have been automated with utility programs and Aliases. However, any further delving will require you to use the dreaded FLASH program. Not that the program is hard to use.... it is conceptually quite simple. All it does is read in a portion of Flash memory into a buffer. There you modify it in a full screen editor, and write it back to Flash. You can also bring in disk files, modify and save it back to disk or flash, or conversely bring in a flash image and save it to disk. The program allows you to specify the flash memory area, how much of it to bring in, and filenames.

What it doesn't do is tell you where in Flash is the data you want to alter. Since the MMU can map any 4K portion of flash memory to any other 4K location, there is no simple means to determine where it is, except for the documentation. There is a LOT of material here, and LOTS of memory locations. The entire operation is full of "gotchas" and is not for the faint of heart.

The lower 64K of Flash has boot, BIOS and DOS code. Because of the MMU, the DOS and BIOS does not have to be sandwiched in upper memory, but actually has the 64K to itself. Different functions are in different portions of the flash chip and the MMU switches them around as needed. Simple in concept, not simple in actual implementation.

The upper 64K of Flash is memory disk. You can put anything you want here; in my case I have some essential utilities and some room to spare. If you are using the CPUZ180 as a controller, you will want to save your final program to this Flash area, and have the CPUZ180 execute it on power-up or reset. The directory for the Flash Drive is sort of CP/M compatible, but it does not allocate disk space in blocks. Instead it is allocated in 128 byte records and wasted space is at a minimum. This non-standard directory allows CPUZDOS to access the files, but programs like XDIR will not report the correct file size nor free space on the Flash Disk. However, the utility program FLASHDRV which allows you to save files from regular disk to the Flash Drive does tell you how large each file is, how much room is available, etc. Because of the efficient utilization of space, 64K of Flash Disk is probably equivalent to about 120K of disk. You are, however, limited in size and must be careful.

It is possible to use 512K RAM and Flash devices but the onboard Flash programmer will not work with the larger chip. This is a pity as it would allow a 448K Flash Drive.... The larger size Flash chip can be used, but only as a read-only device, and must be externally programmed. As this completely negates the advantages of Flash, it is not worth considering. Claude Palm informs me that he is considering the use of a battery-backed static RAM but demand is very low and he cannot presently justify it. Stay tuned.

Rough Edges

"That's not a bug, it's a feature!" In the 1970's and 1980's this used to be the favorite saying of software salesmen. I would be remiss if I didn't admit that the CPUZ180 has its share of "features". Right away I want to make it clear that the hardware is flawless, and my difficulties were with the software alone. When I bought the CPUZ180 I discovered that Claude had become comfortable with the way it worked for him and didn't mind. Quickly I pointed out problem areas, in particular documentation written (for all practical purposes) in greek.

This resulted in over 700 pages of FAX exchanges and eventually a new disk to reload the Flash chip with a revised operating system. I am happy to report that the end result is

definitely worth it. Outright bugs have all been eliminated, and any idiosyncrasies explained and documented. If you obtain a CPUZ180 now, you can be assured of a system that is not only usable, but fun, powerful and productive.

One of my main problems was with the keyboard producing garbage characters. Keyboard timing was altered and should now work with a wide range of keyboards. I also discovered that XT keyboards are not all created equal. The original PC keyboard had 83 keys and no separate arrow keys. The numeric keypad produced arrow key codes, or, if NumLock is engaged, numbers. Later on, clone makers came out with a keyboard with separate arrow and edit keys that produced key codes identical to the old PC keyboard, but from dedicated keys. My 1986 clone XT keyboard was of that type, and has a nice feel to it so I used it. But the arrow and edit keys produced garbage characters!

Finally, after 100 pages of FAX messages we discovered that modern XT keyboards produce different codes! A quick look in PC books showed that Claude's keyboard produces "PS/2" codes. Thus we got an education in keyboard codes and the various modes. Apparently the PS/2 keyboard uses the same hardware interface as the XT, with make and break codes the same save for the high bit set on break. However, the extra arrow and edit keys were new to IBM, so they added a prefix byte to these new keys. For example, the old XT "ALT" key and the PS/2 left hand ALT key produce the same codes. But the PS/2 keyboard also has a right hand ALT key, and THAT key has the ALT codes plus a prefix - 3 bytes! Arrow and edit keys operate in similar fashion. Thus, Claude and I had to rewrite parts of the keyboard translation tables to accommodate my keyboard. Keyboards advertised as being switchable between XT and AT modes are of this type and are supported in the default translation table. However, a patch is available for the older variety. AT-only keyboards will NOT work (different interface).

Other bugs that were fixed was PDRIVE that couldn't handle IDE drives with over 8 heads; slow floppy interleave factor and control-C that would not cancel a command. Each of these was a saga in itself and could fill pages here in TCJ! Happily, they are now in the past.

The SAVE command has a "feature" in that it saves 128 byte records instead of 256 byte pages. This was anathema to me and generated countless complaints. Finally, Claude gave in and Created a patch. However, by this time I had got used to the 128 byte saves and decided not to implement it! In truth, 128 bytes make more sense; 256 byte operation would be strictly for compatibility. But the first time you SAVE only half of your work you will understand why I grumbled so much at first.

There are other "features" out there, but they are either innocent, like SAVE, or a definite improvement on the CP/M way.

CPUZ180 video is memory mapped and applications software normally accesses it via a serial terminal emulation complete with escape codes for cursor position, clear screen, etc. Claude's old computers had an obscure serial terminal with unique codes. To maintain compatibility with his software, he incorporated the strange codes into the CPUZ180. We both agree that it would make life much easier for everyone if a more popular terminal was emulated. Claude favors the ANSI standard so that industrial controller software would be compatible with that for the PC. I favor supporting the ADM-3 terminal codes as the ADM-3 is probably universally understood by all CP/M software. It appears that both sets of codes could coexist at the same time; by the time you read this the matter may be resolved.

Of course, it is possible for you to write software that accesses the video memory directly in true memory-mapped fashion. However, the terminal emulation effectively operates at 2 gazillion kilobaud and is much simpler.

Support and Upgrades

Claude or myself can field any questions on the CPUZ180. Periodic software updates and fixes are always in the works, and Claude has a warehouse of parts as well as manufacturing facilities. This is no orphan... 8-bit computers are alive and well in Australia and Europe where they are popular for industrial control as well as general computing.

Configuring it

I strongly recommend using fast, modern and up to date disk drives. New IDE drives are inexpensive, and many vendors have unadvertised stocks of 120 Mb or smaller units at bargain prices. Likewise, 3.5" floppy drives and disks are much smoother and more reliable than the obsolescent 5-inchers. I do any required 5" to 3.5" transfers in my PC using 22NICE and 22DISK. However, the CPUZ180 fully supports 5" drives.

If you have a comfortable XT style keyboard available, you can certainly use it. Otherwise, new keyboards are inexpensive. Just make sure that it is of a type that will work with a PC or XT. AT-only types do NOT work. Likewise, the older monochrome MDA monitors work OK and is the best choice for their crisp display. Since CP/M software does not support color, I cannot conceive of any use for a color monitor unless you are into such experiments.

Because of the modest 5 volt, 400 ma current draw, it is also possible to mount it inside your PC, either on an empty drive bay or on a blank proto board.

Utilities supplied

ZBUG - full screen debugger
CPUZASM - Z180 assembler

Continues on page 36

Real Computing

By Rick Rodman

Linux Slackware, March 1995

Jack at Walnut Creek responded to our less-than-stellar review of Toolkit for Linux by sending me, gratis, a copy of the Linux Slackware March 1995 CD set. This set is the Slackware 2.2.0 release, which is based on Linux 1.2.1.

My main complaints about T.F.L. concerned its installation program. Poor installation programs can have a number of basic problems: questions which the user can't answer, or Who Knows (WK); error messages that are meaningless to the user, or What In The - (WIT); and places that the user gets dumped off to a prompt without knowing what to do, or You're On Your Own (YOYO). The T.F.L. installation program(s) had all of these problems.

The new installation is much better, but still has some WITs and YOYOs. Under DOS, you should format 3 floppy disks before starting. Then log over to your CD-ROM and enter the command "go". There is a very nice file viewer that allows you to move around the CD reading text, and select, and produce, the bootdisk and rootdisk. When you're finished there, put in the bootdisk and boot it.

The first WIT occurs right at the beginning - the boot disk wants a disk for a RAM disk, and you don't know what to put in. The thing to do is, at the boot prompt, enter "mount root=/dev/fd0". Later, when prompted, put in the rootdisk. *It must not be write-protected.*

Finally, you come to a login prompt. Enter "root" and press Enter. Here's your first YOYO, right? Not exactly; there was some text you might still be able to read about setting up a hard disk in Linux format and then running "setup".

If you were watching the rapidly-scrolling text during the booting, you might know what hard disk device name(s) Linux identified. If not, reboot and watch carefully. I rebooted twice at this point. In my case, I wanted to use the Maxoptix external optical, which Linux identified as /dev/sda. Then you can run "fdisk" and create a Linux partition on the device (or media) with the "n" command. Linux's fdisk program isn't too bad, although it does generate some obscure messages about end sectors or cylinders for which it offers no apparent methods to correct.

Once you finish with that, enter "setup" and press enter. The setup program is quite user-friendly. For me, it crashed at the point of making a boot floppy. You can reboot, entering the command (for example) "mount root=/dev/sda1", re-enter setup and continue where you left off. Eventually you come to the end.

On my system, the boot floppy doesn't work; it dies with a kernel panic. I was able to configure the networking stuff, but the network doesn't work either. Apparently Linux doesn't support the SMC8000 board I have installed.

When it comes to X Window, YOYO again. Looking around through the maze of symbolic links, I found a program called xf86config in the subdirectory / usr/X11/bin. This program is very user-friendly; it creates a file /etc/XF86Config, which you can edit later if things don't work right. One thing to remember here is that, although there are options for PS/2 mouse ports, they don't work. Stick with the Microsoft serial mouse on a regular COM port.

"startx" is the command to start up X window with FVWM, and it works very nicely. If things go haywire, you can exit the server with control-alt-backspace.

This installation procedure is a big improvement, at least in some areas, over the one for the Toolkit for Linux. Considering the tremendous variety of PC hardware that it supports, and the wide array of add-on packages that come with it (four - no, five - shells - at least a dozen editors - several compilers - word processors - spreadsheets), making an automated procedure that works at all is a pretty tall order. Still, it needs a little more work before I'd consider it "easy to install".

Once you're over the installation hurdles, this is a truly incredible package. The mountain of software that's included with it is nothing short of astonishing. It's almost like getting every software package ever offered for DOS, plus Windows, plus all the source code for everything.

Many more devices are now supported. The SCSI adapters supported now include Adaptec, Always, Buslogic, DPT, Future Domain, NCR 5380, Pro Audio Spectrum 16 (sound board with SCSI port), the venerable Seagate ST-02, Trantor T128 and T228, Ultrastor, and WD-7000 FASST. IDE and EIDE hard drives are supported, and many IDE CD-ROMs. The XFree86 X Window server supports many different kinds of video boards, too, including 8514-compatible boards and Number 9 boards.

There are many other Linux packages available. Just Computers is offering a Slackware 2.2 release with 4 CDs and a printed manual. I haven't seen that package, but would hope that the manual would help with installation problems. Yggdrasil, mentioned earlier, comes with one CD and a printed manual. WorkGroup Solutions, Inc. offers three Linux packages as well as a Clipper database clone which works under Linux. And in a possibly ominous vein, a company in Utah called Caldera Inc. is looking to "commercialize" Linux by incorporating some networking software and Internet tools; they appear to be backed by Novell.

The Amiga port of Linux is well along; ports are underway to the Power PC and to the MIPS RISC processor. Linux is taking the Unix world by storm, but unlike Unix itself, there is no AT&T Bell Labs to threaten its freedom.

But what about our readers' 16-bit machines, 8088s and 286s? Are they to be consigned to the storage bin or the yard sale, unsupported and unloved? Don't be too hasty - there are other stirrings on the horizon.

Minix 1.7.0 Beta

Take a look at the following message posted to comp.os.minix by Andy Tanenbaum (I've shortened it a little to save space):

"Prentice-Hall and I have decided to change our policy concerning MINIX. Starting shortly, MINIX 1.7.0 will be distributed free by FTP for educational and research use.

"The differences between Minix 1.6.25 and 1.7.0 for the PC are: a 386 version; ANSI C compiler included; TCP/IP drivers and utilities (not as complete as one may hope); Adaptec 1542A disk and tape driver (SCSI).

"A beta test version is now available for FTPing on ftp.cs.vu.nl in directory pub/minix/beta. See the README file there for further instructions. We would like people who want to be beta testers to fetch MINIX and test it. Please report bugs to Kees Bot (kjb@cs.vu.nl). Kees has done a huge amount of work preparing 1.7.0. He and Philip Homburg have also produced a 32-bit virtual memory MINIX, which will be released to the net by FTP as soon as "classic" MINIX has been released and is stable.

"Manual page references you should read first: usage(8), boot(8), hier(7). Manual pages that document some of the differences with 1.6.25: boot(8), loadkeys(8), ip(4), hd(4), sd(4), fd(4), shutdown(8), usage(8).

"In addition, the MINIX book is being rewritten by myself and my co-author, Al Woodhull. It will probably include all of MINIX on a CD-ROM in the back of the book. We are hoping to finish it by Spring of 1996."

I don't know whether TCJ readers' use would qualify as educational, but it would certainly qualify as research. Those using FTPMAIL will need the exact filenames. In directory /pub/minix/beta/1.7.0, files cat.tar.Z (395,754 bytes), crc.c (4026 bytes), man.tar.Z (408,363 bytes), README (2450 bytes). The README suggests that the file cat.tar.Z is not needed.

In directory /pub/minix/beta/1.7.0/i386, files ROOT (614,400 bytes), USR (614,400 bytes), and USR.01 to USR.05 (mostly 737,280 bytes). These are the 386 version.

In directory /pub/minix/beta/1.7.0/i86, files ROOT (614,400 bytes), TINYROOT (329,728 bytes), USR (614,400 bytes), and USR.01 to USR.05 (mostly 737,280 bytes).

In directory /pub/minix/beta/1.7.0/src, files SRC.01 to SRC.08 (mostly 737,280 bytes).

I've transferred the i86 version using Compuserve's FTP. It's about 10 megabytes, and took about 3-1/2 hours to transfer. Still, note that the ftp address ends in "nl" - that means Netherlands. Imagine a 3-1/2 hour long distance call to Holland!

To install, you copy each of the above files to a 720K floppy. You can use "rawrite" or "putdisk". Boot the ROOT floppy. After pressing "=", complete the mount string to /dev/fd0 and put in the USR floppy.

The new installation procedure is called "instdist". It runs a new, "visual" partition tool called "part". After that, it prompts you for each floppy in turn. The distribution floppies are compressed. And that's about all there is to it... amazingly painless.

I installed this on a magnetic hard drive connected to an Adaptec 1542B SCSI board. Minix expects the Adaptec, if you use one, to be at I/O address 330 (the standard address). Numbering of SCSI devices is a little strange - sd0 is PUN 0, with sd1 to sd4 being the 4 partitions on it; then sd5 is PUN 1, and so on.

I hoped to have a more complete report, but I've only gotten this package a couple of days ago and haven't had enough time to play - er, work with it. We'll have more information next time. Maybe some intrepid TCJ readers will have tried it out, too. But, if you do, remember, this is beta code. In the non-Microsoft world, this means that you're expected to test things and *send in reports.* They want, and *need,* those reports.

That the Minix book is being revised is actually the best news of all. Operating Systems Design and Implementation is one of the most popular, and certainly the most understandable, book anyone has ever written on operating systems. If it has any deficiency at all, it's the Minix 1.1 listings in the back. I'd hope that the new version will still include printed excerpts, at least, of the Minix 1.7 code. I'll be keeping a close watch for this book.

For more information

Real Computing BBS or Fax: +1 703 330 9049
E-mail: 102046.1656@compuserve.com (102046,1656 on Compuserve)
Mail: 8329 Ivy Glen Court, Manassas VA 22110-4631

Walnut Creek CD-ROM (Linux Slackware, $39.95, 2 CDs)
1547 Palos Verdes Mall Suite 260, Walnut Creek CA 94596
+1 510 674 0783

Just Computers (Linux Slackware Pro, $61.95, 4 CDs plus manual)
P.O. Box 751414, Petaluma CA 94975-1414
1 800 800 1648 or +1 707 586 5600

Workgroup Solutions, Inc.
P.O. Box 460190, Aurora CO 80046-0190
+1 303 699 7470

Palmtech CPUZ180 *Continued:*

PDRIVE - Hard and floppy drive configuration
FLASHDRV - Loads disk files to Flash memory "disk"
BOOTDSK - Saves operating system image to disk
FLASH - A simply fantastic RAM, Flash and Disk editor
COPYIBM - Allows file transfers to/from PC disks
FORMAT - Formats floppies
HDID - Obtain hard disk information
SU - Setup utility for defaults, etc
SV - Setup utility for Video
VEDIT - Commercial editor licensed to Palmtech

In addition, a basketful of Public Domain Z-system and CP/M utilities are included, some of them optimized for the CPUZ180. Claude and I both use VEDIT from Compuview (now Greenview Data) and the company has licensed this product to Palmtech and is bundled with the CPUZ180. Documentation is on disk. The CPUZ180 is shipped with VEDIT installed.

The supplied debugger and assembler specifically support the Z180, however, other assemblers work just fine. I have used ZASMB, Digital Research's ASM etc. They all work, but do not support the extra capabilities of the Z180. For other than Z180 specific instructions I have found the public domain WADE debugger or the commercial EZDT to be very good. Both of these are Z-80 versions of SID or DDT respectively and both implement DDT commands as a subset, so if you are familiar with DDT, the learning curve is very short indeed.

Software

Speaking of learning curves, my experience has been that the older the software, the easier it is to learn. CP/M versions of dBASE-II, SuperCalc, VEDIT, Checks & Balances etc. can be mastered in a short time. Modern computer users term them "hostile" because they work from a command prompt instead of the comical icons favored by illiterates. The truth is that with CP/M software you can be doing useful work in much less time and with a minimum of fuss and useless complications.

Ordering

You may order from Claude Palm in Australia, or from myself here in the U.S.

Claude Palm, Palmtech
cnr Moonah & Wills Streets
Boulia, QLD 4829
Australia
tel 011-6177-463-109
fax 011-6177-463-198

Ramon F. Gandia
P.O. Box 970
Nome, AK 99762
tel 907-443-2437
fax 907-443-2487

VISA / MC accepted.

Reader to Reader Continued

the what you call Z-80 programs from one of these operating systems. You get the disks to load and give you a normal prompt, only then can you run other than Basic programs.

Now if I have lost you at this point, I think you may need some hand holding to get you started. We really have fallen down on covering the TRS series and will try and do a special issue on them, say Antiques #77. That will be the next classic support issue in six months. I fear that will be too long to wait so try Lee Hart's mail order book renting service, I would think he has a book or two on the machines. (phone number and address in Groups section.)

If we didn't lose you, then running the Basic programs and saving to tape means that all things work, except the disk drive. It is possible the drive is out of align after many years of heavy use and can read the first few tracks of the disk. After starting TRSDOS it goes looking for more files and can't find them. I do remember something about their startup procedure, but it has been far too many years and I don't have any TRS books to jog my memory.

So the hardware is most likely good, you just got to find the right combination to get it going, much like you did with the date response. Thanks and let us know how you got it going. Bill.

Dear Mr Kibler,

Enclosed is a money order for $34.25 for the following back issues: 40, 41, 60, 62, 64, 67, and 69. In reading the issues I already have, I've noticed a thread going back to issue 56 (a letter from Herb Johnson) regarding a single board Z180 system. The latest I've read on this was your reply to a letter in issue 70. Is this still in the works? I think ISA is a good idea, however I think the Z180 would be better for it than the Z380... better meaning easier to find and less expensive. One thing I would like to see in the magazine is a list of vendors selling Z80/Zx8O and Motorola 68OxO based cards.

Sincerely, Ken Deboy

Ok Ken thanks for the order. Our project had some interest, but no time to bring it to life and just as well, since Z-world in Davis California has done it. They make Z180 based embedded systems and built two units that are PC compatible. One is for the PC-104 standard (the ISA bus in a header design) and one for regular PC bus use. I have been asking and waiting to hear about their use as CP/M machines but haven't got any response. Maybe you might give them a try (916-757-3737 say you heard about it in TCJ.)

Another alternative are the other Z180 boards, checkout Palmtech in this issue, or Micromint SB180, and Tilmann might have some Yasbec boards for a do it yourself approach. Thanks. Bill.

Gentlemen:

The strange words, used in your advertisement in the May 95 edition of Popular Electronics, suggest that you may have made the acquaintance of some wizened old-timer, recounting his grampals tales about eight-bit computers, clocked at two Megahertz and running on memories defined in kilobytes. Did he tell you that CP/M was a lubricant for mechanical adding machines?

Well, I am glad I found you, because I have a handful of Kaypros and am not about to throw them out any time soon. We had learned to mentally associate numerical data with our imagination, and this handicap deprives me of the ability to fully appreciate the value of 60 MHz - Pentim - 10 Mbyte memory - Gigabyte harddisk - 256 colour monitor - etc.etc., just to look at a bunch of pictures.

So, here is a money order for US$ 24.-, to buy a subscription to your publication, in the hope that it represents a substantial effort on your part. Please let me know what back issues you are still able to supply and cost of same.

Are you in a position to gain the cooperation of some former employees of Nonlinear Systems or Kaycomp, to help educate your readers in the design details of the 8-bit machines? Some schematics and circuit descriptions have been published by others, but they have raised far more questions in my mind, than they have answered. This is due to my inferior intellect, I am sure; but I would still like to overcome that barrier.

In the distant past, like six or seven years ago, there was a number of enterprises offering an abundance of CP/M programs. Lack of interest on my part, at that time, prevented me from taking advantage of that fact. Do you plan to, or have you established contact with any of these suppliers or individuals, who were associated with expired firms of this nature? In other words are there copies of all that software still available? Here are some names I have lying around. I have not checked, whether they are still alive (doubt it): Sheepshead Software, SLR Systems, PDUG of Florida, Cranberry Software Tools, Elliam Associates, Sound Potentials.

How about contact with people who were involved with Kaypro Users' Groups? I would like to see the annotated source code of CP/M, that tells me what each instruction is supposed to accomplish, so I can play around with it. I have run the machine code through a disassembler and tried to guess my way through parts of it, with negligible success.

Hope to hear from you. Sincerely, Armin Auerswald, P. 0. Box 122, Stoney Point, Ont., Canada N0R 1N0

Well actually Armin, you can find a few of the contacts right in TCJ. Elliam's add is in the back with the Z-letter who sell Sound Potential disks, not to mention MicroCornucopia disks and back issues (old Kaypro support magazine). I forget who sells SLR software and for Sheepshead and Cranberry I have no Idea, maybe we will get some letters. Lastly I hope you have discoverd our own Kaypro support articles and person, mainly Chuck Stafford. Now for the insides of CP/M you might try MicroMethods at PO Box 909, Warrenton OR 97146. He wrote a book about the insides of CP/M and the program RP/M which came with the source code on disk. RP/M is orginal work, but close enough that your disassembly would make sense. Thanks. Bill

Special Feature

Beginning Forth

Part 1: FIG Forth

Disk I/O in Forth

By Walter J. Rottenkolber

This special on keeping old systems running would not be complete without some fundamental Forth. Why Forth? The Forth programs available for all the old machines work quite well and can provide the user with advanced tools. The source code is almost always included, so bug fixes and changes are possible and easy once Forth is mastered. Mastering Forth can be a problem due to a lack of books for the beginner. To help those starting down the mastering path of Forth, Walter has undertaken the project of explaining how file I/O is done in FigForth and F83. This is beginning material that we hope will uncover a few of the mysteries of Forth. Due to the details covered it will appear as a multi part article. BDK.

Introduction

Charles Moore created Forth, in 1968, to be an integrated programming environment. The Forth kernel, editor, and utilities were memory resident and interactive. Compiling was done incrementally, and the code tested as you went along. His problem was to provide adequate RAM space for programming, and still allow for fast access to source code and other data.

To do this, he divided mass storage into blocks of 1024 bytes, and numbered them in order from zero. The blocks could be accessed randomly by this number with Forth handling the details. To save RAM, the blocks are swapped between the mass storage and a fixed group of buffers in memory. This creates a form of virtual memory with a write back cache.

Forth stores data like index cards in a file box. Disk, tape, and RAM drive can be used for storage, provided the block can be addressed directly. Blocks holding Forth source code are called Screens, and the two terms are often used interchangeably.

Though virtual memory is useful just for random data storage, it also allows a program to extend core RAM onto mass storage. However, the Forth kernel contains only the basic mechanism. You must connect it to the program by Forth's ability to self extend. Self Extension means that Forth can define new data structures, operators and notations, and make them an integral part of itself.

The Forth system begins as a micro kernel that is barely viable on the target computer. This kernel creates a "virtual stack machine" from the CPU's architecture. This virtual machines has two stacks, one for data operations, and one for return addresses. All Forth's have this same virtual design. You perform operations by putting items on the data stack and calling a dictionary word. The word performs the desired operation and leaves the results on the stack to be retrieved or worked on by the next word.

Programming in Forth occurs in two phases. First is extending the Forth kernel with the data structures, operators, and notations required to use all the system resources and provide the needed tools. Second is using this extended Forth and tool set to create the program that solves your problem. Eventually you will gather or develop a library of extensions so you don't start at point zero every time you program. The advantage of buying a commercial Forth is their tested libraries or extensions.

The public domain Forths for 8-bit legacy systems are primarily based on figForth and Laxen & Perry's 1983 Forth Standard F83. They come as a kernel, both compiled and as source, and a set of extensions that supply an editor, native code assembler, and an assortment of utilities. Together, these form a basic working Forth. Forth programming consists of extending this kernel both to improve the working environment, and to fulfill your programming needs.

Most commercial Forths of the past followed the 1979 Forth Standard, whose definitions fell somewhere between that of figForth and the 1983 Standard. Their disk routines follow figForth usage. FigForth and Laxen & Perry's F83 have some notable differences in Word definitions and internal structure, and they are not 100% source code compatible. I'll describe their disk operation separately.

FigForth

FigForths behave as a classic Forth disk system. Some, such as the Atari-800 figForth I have, take over as the operating system. It can boot from a Forth disk because in many older systems, the ROM bootstrap only loads the first disk sector or two. These contain the full load routine for the operating system, in this case, Forth itself. Utilities in Forth enable you to format a disk, do a single drive disk copy, copy screens, and generate a new core binary.

The Kaypro figForth shows a different approach. By this time, CP/M v2.x had calls for random access. Moreover, the Kaypro

loader is entirely in ROM and is tailored for its version of CP/M. So this Forth uses standard CP/M files to hold the Forth binary and the data blocks. The block file is always named FORTH.SCR and must hold 150 blocks. Though Forth manages the block file with CP/M BDOS calls, the user sees an apparently pure Forth system. Forth utilities will backup a disk, copy blocks, and generate a new FORTH.SCR file.

There must be at least two block buffers. No maximum number is specified, though available memory establishes a practical limit. The Atari-800 figForth has two buffers, and the Kaypro's has eight. The buffers are located as a block in RAM with the constant FIRST returning the base address of the first buffer, and LIMIT the top address of highest buffer.

FigForth uses age to determine paging. The oldest block is swapped when the buffers are all in use. The block buffers have a three part structure: Header, data buffer, and terminator. The two byte header uses the msbit (most significant bit or 8th bit) for the update flag, and the remaining bits for the block number. In theory, the system can address blocks 0..32767, or over 32.7 Megabytes. The data buffer is 1024 bytes. The terminator is two null bytes, and is required because the figForth interpreter treats source code in the data buffer as a null terminated string.

The buffers are arranged in a circular queue. The header address of the most recently accessed block is kept in the pointer, PREV. The header address of the next buffer to use is pointed to by USE. This will be either an unassigned buffer or the oldest block in the queue.

When another block is requested, PREV is first checked to see if that block was the most recently used. Otherwise, the queue is searched. If the block is already in a buffer, its header address is placed in PREV to mark it as the new most recent block, and its data address returned on the stack. Since the recently used blocks are the ones most likely to be requested again, this reduces the need for repeated disk access.

If the block is not found, it is assigned the buffer pointed to by USE. But before this buffer is discarded, its Update Flag is checked, and if set, the block is written out. The new block data is read into the buffer from the disk. PREV is then set to this buffer so it becomes the most recent, and the buffer base address returned. USE is advanced to the next (oldest) buffer to use.

FigForth preassigns blocks #0 to #5.

- Block #0 can serve as a convenient title page, but you must take care when using block0 of drive0 (A:) in figForth. First, the outer interpreter treats the source of block0 as coming from the keyboard, and it will crash if you try Loading block0 from mass storage. Second, emptying the buffers resets the buffer headers to zero, which effectively sets all buffers to block0. This confuses the block search routine, causing buffers to be assigned randomly to PREV. To access Block #0 reliably, you must load all the buffers with non-zero blocks first. This Word will do it:

```
: SETBLK0   #BLK 1+ 1 DO I BLOCK DROP LOOP ;
```

Where #BLK is a constant returning the number of buffers. Use as:

```
SETBLK0 then 0 LIST <cr> etc.
```

- Block #1 is the load screen which controls the order of loading (compiling) the source code screens. It is the Forth equivalent of a Make File.
- Blocks #2 and #3 usually hold additional load screens or the source of small utility Words.
- Blocks #4 and #5 hold System and Error Messages.

One aspect of figForth that had me completely baffled for a time was how to copy a block from one disk to another in a multi-drive system. The copy routine, MOVE-SCR (Kaypro), required only From and To block numbers. No mention of drives at all!

This is how it is done. If each disk holds 150 blocks, the blocks in drive A: would number from 0 to 149. Forth then sees the blocks numbered from 150 to 299 as being in drive B:. If there were more drives, the block numbers would continue upward to the other drives. In the two drive Kaypro, block numbers above 299 simply fold back onto drive B:. This forms a poorman's drive array. The blocks have no numbers attached to them. Block #150 in drive B: would become block #0 in drive A:. It's up to the programmer and the program to keep track of which block contains what data.

To copy block #5 on drive B: to block #5 on drive A:, you would write the command: 155 5 MOVE-SCR <cr>. Simple, once you know.

Some early figForths split blocks between drives. This made efficient use of all disk sectors, but it was inconvenient, because disks had to be assigned to specific drives. An early revision eliminated this practice from the standard.

Let's go over the basic disk Words for a figForth system:

BLOCK	BUFFER
R/W	UPDATE
* DISCARD	* SAVE-BUFFERS
EMPTY-BUFFERS	FLUSH
DR0	DR1

Note: * marks a figForth extension which may not be in your implementation.

The parenthesis enclose a comment that shows the data stack picture. The arguments to the left of the dash (—) show the data stack before the Word is run, and those to the right, the data stack afterwards. The top of stack (TOS) is the rightmost of multiple arguments.

BLOCK (blk# — data-buff-adr) Takes a block number, adds the value in the variable OFFSET, and returns the base address

of the data buffer. If the block is not in a buffer, it is assigned the oldest buffer, which then becomes the most recent buffer, and the block contents are read in from the disk.

BUFFER (blk# — data-buff-adr) Assigns the block number to the next buffer pointed to by USE after first writing out the existing block data if the Update flag is set. This then becomes the most recent buffer. BUFFER is useful when writing data to a block as it eliminates an unnecessary disk read. But you must be aware of how it differs from BLOCK, or you can accidentally destroy data.

Note: In figForth, BUFFER behaves differently than BLOCK:
1. BUFFER does NOT add the value in OFFSET, so block numbering starts at Block #0 of drive0 (A:).
2. It does not check for preexisting block assignment, but simply assigns the block pointed to by USE, so you could end up with more than one buffer with the same block number. Before using BUFFER, it is best to clear the buffers with FLUSH or EMPTY-BUFFERS.
3. It does not read in the block contents.

R/W (data-buff-addr blk# r/w-flg) Word that moves one block of data between the data buffer and the block# on mass storage (disk). A True flag will cause a read from disk, and False, a write to disk. This Word is implementation dependent. The location of the block is an absolute offset from the start of disk #0. R/W must also do any required error checking.

UPDATE — sets the update flag on the current block pointed to by PREV. This will cause the block to be written to disk before it is discarded.

DISCARD — resets the update flag on the current block pointed to by PREV. This negates an Update if you want the buffer data to be dumped when the block is discarded.

SAVE-BUFFERS — forces a write of updated buffers to disk, and resets the update flag without changing the block numbers or the buffer contents. Updated buffers normally wait to be written until just before discard.

EMPTY-BUFFERS — resets update flags and sets buffers to zero. In some systems, the buffers are filled with nulls. This effectively dumps all data and blocks. If you have corrupt data in a buffer and want a fresh read of block data from disk, DISCARD is not enough as the buffer is still assigned to the block. A repeat BLOCK would simply return you to the corrupt data. Empty-buffers will force a new block read from disk.

FLUSH — combines the effects of Save-buffers and Empty-buffers. It ensures saving Updated buffers before they are dumped. This is the preferred Word at the end of a work session or before leaving Forth.

Some Forths, such as the Kaypro's under CP/M, function on top of a disk operating system (DOS) which has its own sector buffer. The SAVE-BUFFERS command only forces a write of the Forth buffers, not that of the DOS. Additional code is needed to force a write of the sector buffer or its data could be lost. A CLOSE file or a read/write of a new sector will do it. However, Forths using blocks may not Close files because the directory entries are just read, not changed. So a forced block read is more common. It pays to check how well this code works, as it may make assumptions that don't always hold true. For example, the Kaypro figForth's FLUSH writes out updated blocks and zeros out the header, but doesn't force a write of the DOS buffer. One fix redefines FLUSH:

```
: FLUSH   FLUSH  1 BLOCK DROP  EMPTY-BUFFERS ;
7 LOAD
BYE
SAVE nn FORTH.COM
```

Screen #7 in the original FORTH.SCR (from the MicroCornucopia collection) has a routine to make additions to the Forth dictionary permanent. When you leave figForth, it prints a message giving its size in nn Pages. You use this value in a CP/M Save routine.

The Words DR0 and DR1 select either the A: or B: drives as the start of block0. This is not the same as a drive change from the CP/M command line. Instead, DR1 changes OFFSET to the maximum number of blocks in drive0. This value shifts access to Block #0 up one drive to B:. This shift also occurs up the drive chain if there are more drives. For the Kaypro figForth, this value should be 150, but a bug causes 800 to be set. A simple fix is to redefine DR1, and save the changes as in FLUSH.

```
: DR1   150 OFFSET ! ;
```

The error is in the Kaypro figForth v1.16 8080-assembler code. Leftover from a previous incarnation using 8" disk drives is code for single density disks. The variable DENSITY contains a flag that is compiled as True for double density. DR1 uses this flag to choose between setting OFFSET for double density (150) or single (800). The code mistakenly bypasses the Word DENSITY, and this causes the bug. The fix is simple, but requires a recompile of the Fig kernel.

```
;
        DB      85H             ; #BUF
        DB      '#BUF'
        DB      'F'+80H
        DW      SPBLK-10
NOBUF   DW      DOCON,NBUF
;
        DB      87H             ; DENSITY (0 =SINGLE 1 =DOUBLE)
        DB      'DENSIT'
        DB      'Y'+80H
        DW      NOBUF-8
DENSTY  DW      DOVAR
        DW      1
;
        DB      8AH             ; DISK-ERROR
        DB      'DISK-ERRO'
        DB      'R'+80H
;       DW      NOBUF-8         ; Bad Code — Bypasses DENSITY.
        DW      DENSTY-10       ; Correct code.
DSKERR  DW      DOVAR,0
;
```

Next issue continues with: Laxen and Perry F83

Moving Forth

by Brad Rodriguez

Special Feature

Intermediate Users

Part 8 6809 CamelForth

Finally, the last installment of "Moving Forth!" Here is the long-promised ANSI CamelForth for the Motorola 6809, and specifically for the Scroungmaster II processor board.

Unlike the Z80 and 8051 CamelForth, the 6809 Forth was produced with my "Chromium 2" Forth metacompiler [ROD92]. Right away you'll notice two things: first, the metacompiler runs on an older Forth (F83), and so the source code is contained in 16x64 Forth "screens". I've converted these to an ASCII file for TCJ, but the original formatting is still evident.

Second, source code for a Forth metacompiler looks like ordinary Forth code (with a few changes, which I'll discuss shortly). Thus the definition of 1+ is given as

```
CODE 1+   1 # ADDD,   NEXT   ;C
```

The assembler used is the 6809 assembler I've described previously in TCJ [ROD91].

I typed the high-level source code directly from the already-published listings (converting to the Forth syntax in the process). Unfortunately, this was done over the space of a few days, and sometimes I worked from the Z80 listing, and sometimes the 8051...with the result that the Harvard-architecture constructs (such as I@ and IALLOT) are not consistently used in the 6809 code. This is of no consequence for the non-Harvard 6809, but I'll have to correct this before porting the Forth code to a Harvard CPU.

Also, since I was working from published listings, I often neglected typing the detailed comments for the high-level words. For this I apologize. You can find how any word works by consulting the previous listings, but I shouldn't force you to do this.

6809 CAMELFORTH SOURCE CODE

The 6809 CamelForth model holds top-of-stack in D, and uses the S stack pointer for the Parameter Stack. The U stack pointer is the Return Stack Pointer, and Y is the Interpreter Pointer. X is the temporary register "W". The 6809 direct page pointer DPR holds the high byte of the User Pointer (the low byte is assumed to be zero).

The memory map for a Scroungemaster II with 8K of RAM and 8K of EPROM is as follows:

6000-797Fh	RAM dictionary (for new definitions)
7980-79FFh	Terminal Input Buffer
7A00-7A7Fh	User Area (USER variables)
7A80-7AFFh	Parameter Stack (grows downward)
7B00-7B27h	HOLD area (grows downward)
7B28-7B7Fh	PAD area (general purpose buffer)
7B80-7BFFh	Return Stack (grows downward)
E000-FFFFh	Forth kernel in EPROM

All of the RAM data areas are referenced to the User Pointer, whose starting value is given by UP-INIT: in this case, 7A00h. (Note the use of UP-INIT-HI for the high byte of this value.) When CamelForth starts, it will set its Dictionary Pointer to DP-INIT, which must be in RAM so you can add new definitions to the Forth dictionary. These are all specified with the metacompiler's EQU directive. An EQU is like a CONSTANT, except that it is *only* known to the metacompiler. These EQUates take up no space in the 6809 kernel, and will not appear in the 6809 Forth's dictionary.

DICTIONARY tells the metacompiler where to compile the code, in this case for an 8K EPROM from E000-FFFFh. The new dictionary is named "ROM", and then ROM is specified to select that dictionary. (If you're familiar with Forth vocabularies, you'll see a strong resemblance.)

AKA ("also known as") defines a synonym for a Forth word. Since the 6809 is a non-Harvard machine, we should compile @ wherever I@ appears in the source code, and likewise for the other "I-prefix" (instruction-space) words. AKA will do this. These synonyms are like EQUates — they don't appear in the 6809 dictionary.

The metacompiler allows you to use forward references, i.e., Forth words which haven't been defined yet. (You must of course define them before you finish!) Often this is automatic,

but AKA requires you to explicitly declare a forward reference with PRESUME. Thus

```
PRESUME WORD  AKA WORD IWORD
```

is needed to create the IWORD synonym. @ ! HERE ALLOT and the others are PRESUMEd by the metacompiler, so we don't have to do so here.

The CODE definitions are conventional. Note that you can use

```
HERE EQU labelname
```

to generate a label when metacompiling. (This is a function of the metacompiler, not the assembler.) Also, ASM: begins a "fragment" of assembler code (i.e., not part of a CODE word).

The phrase

```
HERE RESOLVES name
```

is used to resolve certain forward references which are made by the metacompiler (for example, the metacompiler has to know where the code for the DOCOLON action is). You should leave these alone. Otherwise, feel free to add any CODE definitions to the source code.

The code for defining words and control structures (IMMEDIATE words) is rather opaque. This is because these words *must also perform some action while metacompiling.* For example: the 6809 Forth includes the standard word CONSTANT, to define new constants. But CONSTANTs may also appear in the 6809 kernel; we may have to define a CONSTANT *while metacompiling.* The EMULATE: phrase instructs the metacompiler how to handle the word CONSTANT if it is encountered. This phrase is written entirely using metacompiler words, and so may appear to be total gibberish.

Likewise, IF THEN and their ilk include the metacompiler phrases to build and resolve branches in the 6809 image. Some Forth metacompilers bury this code inside the compiler. This makes for prettier target code, but if you change the way branches work (for example), you have to perform surgery on the metacompiler. I preferred to make these actions easily changeable, and so I designed Chromium to put them in the target source code. (The most horrific examples are the definitions of TENDLOOP and TS", which actually extend the metacompiler vocabulary in the middle of the target source code.)

If you're new to Forth and the metacompiler, it's best to just accept these as given. "Ordinary" colon definitions are easy to add. Just follow the example of the rest of the 6809 source code. You can even make CREATE..DOES> definitions, as long as you don't need to use them within the metacompiler.

FUTURE WORK

On a 1 MHz 6809, a line of text input takes a noticeable time to process (up to 1 second at a rough estimate). This is partly becuase so much of the interpreter is written in high-level Forth, and partly because CamelForth uses a single-linked-list dictionary. These handicaps only affect *compilation* speed, not execution speed, but the delays can be annoying. Maybe someday I'll do an article on "Accelerating Forth".

Currently, the User Pointer never changes. The reason we have a User Pointer is to support multitasking — each task having separate user area, stacks, etc. I'll be working on this soon. I may also explore using the SM II's memory management to give each task a full 32K private dictionary. And of course, I intend to write a true *multiprocessor* Forth kernel using the shared bus. If I live long enough, a *distributed* Forth kernel using the serial ports (a la Transputer) is the logical next step.

The source code for 6809 CamelForth, version 1.0, is available on GEnie's Forth Roundtable in the file CAM09-10.ZIP. This file includes the Chromium 2 metacompiler, complete and ready to run. You'll need a copy of F83. Then you merely type

```
F83 CHROMIUM.SCR
1 LOAD
BYE
```

This will load the metacompiler, compile the 6809 CamelForth, and write the result to an Intel hex file 6809.HEX. Note: if you're using the CP/M or Atari ST versions of F83, you'll have to edit the load screen to delete the hex file utility, since this only works under MS-DOS. I haven't yet tested Chromium 2 with CP/M or Atari ST, so if you need assistance, please contact me.

Which reminds me: I have a *new email address!* You can now reach me as bj@genie.com, or just BJ if you're a GEnie user. It's a lot easier to type.

ERRATA

There were some errors in the Harvard memory access in CamelForth/8051. The corrected file is on GEnie as CAM51-11.ZIP. I've also uploaded the current Z80 CamelForth, CAM80-12.ZIP, which incorporates all the fixes which have been published in TCJ.

REFERENCES

[ROD91] Rodriguez, B. J., "B.Y.O. Assembler," The Computer Journal #52 (Sep/Oct 1991) and #54 (Jan/Feb 1992).

[ROD92] Rodriguez, B. J., "Principles of Metacompilation," Forth Dimensions XIV:3 (Sep/Oct 1992), XIV:4 (Nov/Dec 1992), and XIV:5 (Jan/Feb 1993). Describes the "Chromium 1" metacompiler.

```
CamelForth for the Motorola 6809 (c) 1995 Bradford J. Rodriguez
*  Permission is granted to freely copy, modify, and          *
*  distribute this program for personal or educational use.   *
*  Commercial inquiries should be directed to the author at   *
*  221 King St. E., #32, Hamilton, Ontario L8N 1B5 Canada     *

Direct-Threaded Forth model for Motorola 6809
16 bit cell, 8 bit char, 8 bit (byte) adrs unit
  X = Forth  W     temporary address register
  Y =        IP    Interpreter Pointer
  U =        RSP   Return Stack Pointer
  S =        PSP   Parameter Stack Pointer
  D =        TOS   top parameter stack item
 DP =        UP    User Pointer (high byte)

v1.0  alpha test version, 28 Apr 95

\ 6809 Source Code: boot parameters          (c) 28apr95 bjr
HEX 0E000 FFFF DICTIONARY ROM  ROM
  7A00 EQU UP-INIT    \ UP must be page aligned.  Stacks,
  7A  EQU UP-INIT-HI  \  TIB, etc. init'd relative to UP.

  6000 EQU DP-INIT    \ starting RAM adrs for dictionary
  \ SM2 memory map with 8K RAM: 6000-7BFF RAM,
  \  7C00-7FFF I/O

\ Harvard synonyms - these must all be PRESUMEd
AKA , I,   AKA @ I@   AKA ! I!
AKA C, IC,  AKA C@ IC@  AKA C! IC!
AKA HERE IHERE   AKA ALLOT IALLOT
PRESUME WORD     AKA WORD IWORD

\ 6809 DTC: SCC initialization          (c) 17apr95 bjr
HERE EQU SCCATBL HEX
  7C02 , 2500 ,  \ port address, #bytes, reset reg ptr
  09C0 , 0444 , 0100 , 0200 , 03C0 , 0560 ,
  0901 , 0A00 , 0B50 , 0C18 , 0D00 , 0E02 ,
  0E03 , 03C1 , 0568 , 0F00 , 1010 , 0100 ,

HERE EQU SCCBTBL
  7C00 , 1F00 ,  \ port address, #bytes, reset reg ptr
  0444 , 0100 , 03C0 , 0560 , 0A00 , 0B50 ,
  0C18 , 0D00 , 0E02 , 0E03 , 03C1 , 0568 ,
  0F00 , 1010 , 0100 , \ 0909 ,

ASM: HERE EQU SCCINIT  \ set up on-board i/o
  X ,++ LDY,  X ,+ LDB,
  BEGIN,  X ,+ LDA,  Y 0, STA,  DECB,  EQ UNTIL,  RTS, ;C

\ 6809 DTC: serial I/O                  (c) 31mar95 bjr
HEX 7C02 EQU SCCACMD  7C03 EQU SCCADTA

CODE KEY  \ — c  get char from serial port
  6 # ( D) PSHS,  BEGIN,  SCCACMD LDB,
  1 # ANDB,  NE UNTIL,
  SCCADTA LDB,  CLRA,  NEXT ;C

CODE KEY?  \ — f  return true if char waiting
  6 # ( D) PSHS,  CLRA,  SCCACMD LDB,  1 # ANDB,
  NE IF,  -1 # LDB,  THEN,  NEXT ;C

CODE EMIT  \ c —  output character to serial port
  BEGIN,  SCCACMD LDA,  4 # ANDA,  NE UNTIL,
  SCCADTA STB,  6 # ( D) PULS,  NEXT ;C

\ 6809 DTC: interpreter logic           (c) 17apr95 bjr
ASM:  HERE RESOLVES DOCOLON
   HERE EQU <DOCOLON>
   HEX 20 # ( Y) PSHU,  20 # PULS,  NEXT, ;C

ASM:  HERE RESOLVES DOCREATE
   HERE EQU <DOCREATE>
   10 # ( X) PULS,  6 # ( D) PSHS,  X D TFR,  NEXT, ;C

CODE EXIT  \ —  exit a colon definition
   HEX 20 # ( Y) PULU,  NEXT ;C

CODE LIT  \ — x  fetch inline literal to stack
   6 # ( D) PSHS,  Y ,++ LDD,  NEXT ;C

CODE EXECUTE  \ i*x xt — j*x  execute Forth word at 'xt'
   D X TFR,  6 # ( D) PULS,  X 0, JMP, ;C

\ 6809 DTC: stack operations            (c) 31mar95 bjr
CODE DUP  \ x — x x   duplicate top of stack
   6 # ( D) PSHS,  NEXT ;C

CODE ?DUP  \ x — 0 | x x  DUP if nonzero
   0 # CMPD,  NE IF,  6 # ( D) PSHS,  THEN,  NEXT ;C

CODE DROP  \ x —  drop top of stack
   6 # ( D) PULS,  NEXT ;C

CODE SWAP  \ x1 x2 — x2 x1  swap top two items
   S 0, LDX,  S 0, STD,  X D TFR,  NEXT ;C

CODE OVER  \ x1 x2 — x1 x2 x1  per stack diagram
   6 # ( D) PSHS,  S 2 , LDD,  NEXT ;C

\ 6809 DTC: stack operations            (c) 31mar95 bjr
CODE ROT  \ x1 x2 x3 — x2 x3 x1  per stack diagram
   S 0, LDX,  S 0, STD,  S 2 , LDD,  S 2 , STX,  NEXT ;C

CODE NIP  \ x1 x2 — x2          per stack diagram
   S 2 , LEAS,  NEXT ;C

CODE TUCK  \ x1 x2 — x2 x1 x2   per stack diagram
   S 0, LDX,  S 0, STD,  HEX 10 # ( X) PSHS,  NEXT ;C

CODE >R  \ x —  R: — x     push to return stack
   6 # ( D) PSHU,  6 # ( D) PULS,  NEXT ;C

CODE R>  \ — x  R: x —     pop from return stack
   6 # ( D) PSHS,  6 # ( D) PULU,  NEXT ;C

\ 6809 DTC: stack operations            (c) 31mar95 bjr
CODE R@  \ — x  R: x — x   fetch from return stack
   6 # ( D) PSHS,  U 0, LDD,  NEXT ;C

CODE SP@  \ — a-addr       get data stack pointer
   6 # ( D) PSHS,  S D TFR,  NEXT ;C

CODE SP!  \ a-addr —       set data stack pointer
   D S TFR,  6 # ( D) PULS,  NEXT ;C

CODE RP@  \ — a-addr       get return stack pointer
   6 # ( D) PSHS,  U D TFR,  NEXT ;C

CODE RP!  \ a-addr —       set return stack pointer
   D U TFR,  6 # ( D) PULS,  NEXT ;C

\ 6809 DTC: memory operations           (c) 31mar95 bjr
CODE !  \ x a-addr —       store cell in memory
   D X TFR,  6 # ( D) PULS,  X 0, STD,  6 # ( D) PULS,
   NEXT ;C

CODE C!  \ char c-addr —   store char in memory
   D X TFR,  6 # ( D) PULS,  X 0, STB,  6 # ( D) PULS,
   NEXT ;C

CODE @  \ a-addr — x       fetch cell from memory
   D X TFR,  X 0, LDD,  NEXT ;C

CODE C@  \ c-addr — char   fetch char from memory
   D X TFR,  X 0, LDB,  CLRA,  NEXT ;C

\ 6809 DTC: arithmetic operations       (c) 26apr95 bjr
CODE +  \ n1/u1 n2/u2 — n3/u3  add n1+n2
   S ,++ ADDD,  NEXT ;C

CODE M+  \ d n — d         add single to double
   S 2 , ADDD,  S 2 , STD,
   6 # ( D) PULS,  0 # ADCB,  0 # ADCA,  NEXT ;C

CODE -  \ n1/u1 n2/u2 — n3/u3  subtract n1-n2
   S ,++ SUBD,  COMA,  COMB,  1 # ADDD,  NEXT ;C

CODE NEGATE  \ x1 — x2     two's complement
   COMA,  COMB,  1 # ADDD,  NEXT ;C

\ 6809 DTC: logical operations          (c) 31mar95 bjr
CODE AND  \ x1 x2 — x3     logical AND
   S ,+ ANDA,  S ,+ ANDB,  NEXT ;C

CODE OR  \ x1 x2 — x3      logical OR
   S ,+ ORA,  S ,+ ORB,  NEXT ;C

CODE XOR  \ x1 x2 — e3     logical XOR
   S ,+ EORA,  S ,+ EORB,  NEXT ;C

CODE INVERT  \ x1 — x2     bitwise inversion
   COMA,  COMB,  NEXT ;C

CODE ><  \ x1 — x2         swap bytes
   A B EXG,  NEXT ;C

\ 6809 DTC: arithmetic operations       (c) 31mar95 bjr
CODE 1+  \ n1/u1 — n2/u2   add 1 to TOS
   1 # ADDD,  NEXT ;C

CODE 1-  \ n1/u1 — n2/u2   subtract 1 from TOS
   1 # SUBD,  NEXT ;C

CODE 2*  \ x1 — x2         arithmetic left shift
   ASLB,  ROLA,  NEXT ;C

CODE 2/  \ x1 — x2         arithmetic right shift
   ASRA,  RORB,  NEXT ;C

CODE +!  \ n/u a-addr —    add cell to memory
   D X TFR,  6 # ( D) PULS,  X 0, ADDD,  X 0, STD,
   6 # ( D) PULS,  NEXT ;C

\ 6809 DTC: arithmetic operations       (c) 31mar95 bjr
CODE LSHIFT  \ x1 u — x2   logical shift left u places
   D X TFR,  6 # ( D) PULS,  X 0, LEAX,  NE IF,
     BEGIN,  LSLB,  ROLA,  X -1 , LEAX,  EQ UNTIL,
   THEN,  NEXT ;C

CODE RSHIFT  \ x1 u — x2   logical shift right u places
   D X TFR,  6 # ( D) PULS,  X 0, LEAX,  NE IF,
     BEGIN,  LSRA,  RORB,  X -1 , LEAX,  EQ UNTIL,
   THEN,  NEXT ;C

\ 6809 DTC: comparison operations       (c) 31mar95 bjr
CODE 0=  \ n/u — flag   return true if TOS=0
   0 # CMPD,  EQ IF,
     HERE EQU TOSTRUE  -1 # LDD,  NEXT
   THEN,  CLRA,  CLRB,  NEXT ;C

CODE 0<  \ n/u — flag   true if TOS negative
   TSTA,  TOSTRUE BMI,  CLRA,  CLRB,  NEXT ;C

CODE =  \ x1 x2 — flag  test x1=x2
   S ,++ SUBD,  TOSTRUE BEQ,  CLRA,  CLRB,  NEXT ;C

CODE <>  \ x1 x2 — flag  test not equal
   S ,++ SUBD,  TOSTRUE BNE,  CLRA,  CLRB,  NEXT ;C

\ 6809 DTC: comparison operations       (c) 31mar95 bjr
CODE <  \ n1 n2 — flag  test n1<n2, signed
   S ,++ SUBD,  TOSTRUE BGT,  CLRA,  CLRB,  NEXT ;C

CODE >  \ n1 n2 — flag  test n1>n2, signed
   S ,++ SUBD,  TOSTRUE BLT,  CLRA,  CLRB,  NEXT ;C

CODE U<  \ n1 n2 — flag  test n1<n2, unsigned
   S ,++ SUBD,  TOSTRUE BHI,  CLRA,  CLRB,  NEXT ;C

CODE U>  \ n1 n2 — flag  test n1>n2, unsigned
   S ,++ SUBD,  TOSTRUE BLO,  CLRA,  CLRB,  NEXT ;C

\ 6809 DTC: branch and loop operations  (c) 31mar95 bjr
CODE BRANCH  \ —  branch always
   Y 0, LDY,  NEXT ;C

CODE ?BRANCH  \ x —  branch if TOS zero
   0 # CMPD,  EQ IF,  6 # ( D) PULS,  Y 0, LDY,  NEXT
        THEN,  6 # ( D) PULS,  Y 2 , LEAY,  NEXT ;C

CODE (DO)  \ n1|u1 n2|u2 —  R: — sys1 sys2
   D X TFR,  HEX 8000 # LDD,  S ,++ SUBD,  \ fudge=8000-limit
   6 # ( D) PSHU,  X D, LEAX,  10 # ( X) PSHU,  \ start+fudg
   6 # ( D) PULS,  NEXT ;C

CODE UNLOOP  \ —  R: sys1 sys2 —  drop loop parameters
   U 4 , LEAU,  NEXT ;C

\ 6809 DTC: branch and loop operations  (c) 31mar95 bjr
CODE (LOOP)  \ R: sys1 sys2 — | sys1 sys2  run-time for LOOP
   6 # ( D) PSHS,  U 0, LDD,  1 # ADDD,  VC IF,
     HERE EQU TAKELOOP  U 0, STD,  Y 0, LDY,
     6 # PULS,  NEXT
   THEN,  Y 2 , LEAY,  U 4 , LEAU,  6 # PULS,  NEXT ;C

CODE (+LOOP)  \ n —  R: sys1 sys2 — | sys1 sys2  for +LOOP
   U 0, ADDD,  TAKELOOP BVC,
   Y 2 , LEAY,  U 4 , LEAU,  6 # PULS,  NEXT ;C

CODE I  \ — n  R: sys1 sys2 — sys1 sys2  loop index
   6 # ( D) PSHS,  U 0, LDD,  U 2 , SUBD,  NEXT ;C

CODE J  \ — n  R: 4*sys — 4*sys  2nd loop index
   6 # ( D) PSHS,  U 4 , LDD,  U 6 , SUBD,  NEXT ;C

\ 6809 DTC: multiply                    (c) 25apr95 bjr
CODE UM*  \ u1 u2 — ud  16*16->32 unsigned multiply
   16 # ( X,D) PSHS,          \ push temporary, u2
   S 5 , LDA,  S 1 , LDB,  MUL,  S 2 , STD,  \ 1lo*2lo
   S 4 , LDA,  S 1 , LDB,  MUL,          \ 1hi*2lo
   S 2 , ADDB,  0 # ADCA,  S 1 , STD,
   S 5 , LDA,  S 0, LDB,  MUL,           \ 1lo*2hi
   S 1 , ADDD,  S 1 , STD,  CLRA,  ROLA,   \ cy in A
   S 0, LDB,  S 0, STA,  S 4 , LDA,  MUL,  \ 2hi*1hi
   S 0, ADDD,                  \ hi result in D
   S 2 , LDX,  S 4 , LEAS,  S 0, STX,  NEXT ;C  \ lo result

\ 6809 DTC: divide                      (c) 25apr95 bjr
CODE UM/MOD  \ ud u1 — rem quot  32/16->16 divide
   HEX 6 # PSHS,  10 # LDX,  \ save u1 in mem
   S 5 , ASL,  S 4 , ROL,    \ initial shift (lo 16)
   BEGIN,
     S 3 , ROL,  S 2 , ROL,  S 2 , LDD,  \ shift left hi 16
     CS IF,            \ 1xxxx: 17 bits, subtract is ok
       S 0, SUBD,  S 2 , STD,  0FE # ANDCC,  \ clear cy
     ELSE,             \ 0xxxx: 16 bits, test subtract
       S 0, SUBD,  CC IF,  S 2 , STD,  THEN,  \ cs=can't subtr
     THEN,             \ cy=0 if sub ok, 1 if no subtract
     S 5 , ROL,  S 4 , ROL,  \ rotate cy into result
```

```
  X -1 , LEAX,  EQ UNTIL,  \ loop 16 times
  S 4 , LDD,  COMA,  COMB,  \ invert to get true quot in D
  S 2 , LDX,  S 4 , STX,  S 4 , LEAS,  \ save rem, clean stack
  NEXT ;C

\  6809 DTC: block and string operations      (c) 31mar95 bjr
CODE FILL  \ c-addr u char —   fill mem with char
  HEX 20 # ( Y) PSHU,  30 # ( X,Y) PULS,  \ D=char X=u Y=adr
  0 # CMPX,  NE IF,
    BEGIN,  Y ,+ STB,  X -1 , LEAX,  EQ UNTIL,
  THEN,  6 # ( D) PULS,  20 # ( Y) PULU,  NEXT, ;C

CODE B=  \ c-addr1 c-addr2 u — n   string compare 1:2
  S 2 , ADDD,  S 2 , LDX,  S 2 , STY,  \ X=src D=end
  S 0, LDY,  S 0, STD,  CLRB,  \ Y=dst B=0
  BEGIN,  S 0, CMPX,  NE WHILE,  X ,+ LDA,  Y ,+ SUBA,
    NE IF,  0 # SBCB,  B A TFR,  1 # ORB,
        HEX 30 # ( X,Y) PULS,  NEXT,  THEN,
  REPEAT,  B A TFR,  HEX 30 # ( X,Y) PULS,  NEXT, ;C

\  6809 DTC: block and string operations      (c) 31mar95 bjr
CODE CMOVE  \ c-addr1 c-addr2 u —   move from bottom 1->2
  S 2 , ADDD,  S 2 , LDX,  S 2 , STY,  \ X=src D=end
  S 0, LDY,  S 0, STD,  \ Y=dst
  BEGIN,  S 0, CMPX,  NE WHILE,  X ,+ LDB,  Y ,+ STB,
  REPEAT,  HEX 30 # ( X,Y) PULS,  6 # ( D) PULS,  NEXT ;C

CODE CMOVE>  \ c-addr1 c-addr2 u —   move from top 1->2
  S 2 , LDX,  X D, LEAX,  S 2 , STY,  \ X=src D=u
  S 0, LDY,  Y D, LEAY,  \ Y=dst
  BEGIN,  S 0, CMPY,  NE WHILE,  X -, LDB,  Y -, STB,
  REPEAT,  HEX 30 # ( X,Y) PULS,  6 # ( D) PULS,  NEXT ;C

\  6809 DTC: block and string operations      (c) 31mar95 bjr
ASM: HERE EQU SKIPEXIT  Y -1 , LEAY,
HERE EQU SKIPDONE  HEX 20 # PSHS,  X D TFR,
          20 # PULU, NEXT ;C

CODE SKIP  \ c-addr u c — c-addr' u'   skip matching chars
  HEX 20 # ( Y) PSHU,  30 # ( X,Y) PULS,  \ D=char X=u Y=adr
  0 # CMPX,  NE IF,
    BEGIN,  Y ,+ CMPB,  SKIPEXIT BNE,  X -1 , LEAX,
  EQ UNTIL,  THEN,  SKIPDONE BRA, ;C

CODE SCAN  \ c-addr u c — c-addr' u'   find matching char
  HEX 20 # ( Y) PSHU,  30 # ( X,Y) PULS,  \ D=char X=u Y=adr
  0 # CMPX,  NE IF,
    BEGIN,  Y ,+ CMPB,  SKIPEXIT BEQ,  X -1 , LEAX,
  EQ UNTIL,  THEN,  SKIPDONE BRA, ;C

\  6809 DTC: system dependencies      (c) 21apr95 bjr

\ These words are shorter in CODE than as colon definitions!
CODE ALIGNED  NEXT ;C      \ a1 — a2   align address
CODE ALIGN  NEXT ;C      \ —   align HERE
CODE CELL+  2 # ADDD,  NEXT ;C   \ a1 — a2   add cell size
CODE CELLS  ASLB,  ROLA,  NEXT ;C  \ n1 - n2   cells->adr units
CODE CHAR+  1 # ADDD,  NEXT ;C   \ a1 — a2   add char size
CODE CHARS  NEXT ;C      \ n1 — n2   chars->adr units
CODE >BODY  3 # ADDD,  NEXT ;C   \ xt — a-addr   cfa->pfa

AKA 1- CHAR-
\ Note: CELL, a constant, must be defined after CONSTANT.

\  6809 DTC: system dependencies      (c) 21apr95 bjr
HEX
: COMPILE,  , ;      \ xt —   append execution token
: !CF    0BD OVER C! 1+ ! ;  \ adrs cfa —   set code field
: ,CF    HERE !CF 3 ALLOT ;  \ adrs —   append code field
: !COLON  -3 ALLOT <DOCOLON> ,CF ;  \ —   changes last c.f.
: ,EXIT  ['] EXIT COMPILE, ;  \ —   append EXIT action
: ,BRANCH  , ;      \ xt —   append branch instr.
: ,DEST  , ;      \ dest —   append dest'n adrs
: !DEST  ! ;      \ dest adr —   change dest'n

\  6809 DTC: dodoes (does>) does>      (c) 18apr95 bjr

ASM: HERE RESOLVES DODOES  HERE EQU <DODOES>
  HEX 20 # ( Y) PSHU,  20 # ( Y) PULS,  \ adrs of DODOES code
  10 # ( X) PULS,  6 # ( D) PSHS,  X D TFR,  \ adrs of data
  NEXT, ;C
DECIMAL  \ to keep ,CF from compiling as a hex number

: (DOES>)  R>  LATEST @ NFA>CFA  !CF ;

: DOES>  ['] (DOES>) COMPILE,
        <DODOES> ,CF ;  IMMEDIATE

\  6809 DTC: defining words      (c) 21apr95 bjr
: :  CREATE HIDE ] !COLON ;

: ;  REVEAL ,EXIT [COMPILE] [ ;  IMMEDIATE

: CONSTANT  CREATE , ;CODE
  HEX 10 # ( X) PULS,  6 # ( D) PSHS,  X 0, LDD,  NEXT, ;C
EMULATE: TCREATE T, MDOES> T@  ;EMULATE

: VARIABLE  CREATE CELL ALLOT ;
EMULATE: TCREATE 0 T, MDOES>  ;EMULATE

: USER  CREATE , ;CODE
  HEX 10 # ( X) PULS,  6 # ( D) PSHS,  \ get pfa in X
  DPR A TFR,  CLRB,  X 0, ADDD,  NEXT, ;C  \ UP+offset -> D
EMULATE: TCREATE T, MDOES> .UNDEF  ;EMULATE

\  High level: control structures      (c) 21apr95 bjr
: IF  \ — adrs   conditional forward branch
  ['] ?BRANCH ,BRANCH  HERE DUP ,DEST ;
EMULATE: M['] ?BRANCH T,  THERE DUP T, ;EMULATE
IMMEDIATE

: THEN \ adrs —   resolve forward branch
  HERE SWAP !DEST ;
EMULATE: THERE SWAP T! ;EMULATE      IMMEDIATE

: ELSE \ adrs1 — adrs2   branch for IF..ELSE
  ['] BRANCH ,BRANCH  HERE DUP ,DEST
  SWAP [COMPILE] THEN ;
EMULATE: M['] BRANCH T,  THERE DUP T,
  SWAP  THERE SWAP T! ;EMULATE      IMMEDIATE

\  High level: control structures      (c) 21apr95 bjr
: BEGIN  HERE ; \ — adrs   target for backward branch
EMULATE: THERE  ;EMULATE      IMMEDIATE

: UNTIL      \ adrs —   conditional backward branch
  ['] ?BRANCH ,BRANCH  ,DEST ;
EMULATE: M['] ?BRANCH T, T, ;EMULATE  IMMEDIATE

: AGAIN      \ adrs —   unconditional backward branch
  ['] BRANCH ,BRANCH  ,DEST ;
EMULATE: M['] BRANCH T,  T, ;EMULATE  IMMEDIATE

: WHILE      \ — adrs   branch for WHILE loop
  [COMPILE] IF ;
EMULATE: M['] ?BRANCH T,  THERE DUP T, ;EMULATE
IMMEDIATE

\  High level: control structures      (c) 21apr95 bjr
: REPEAT      \ adrs1 adrs2 —   resolve WHILE loop
  SWAP [COMPILE] AGAIN [COMPILE] THEN ;
EMULATE: SWAP  M['] BRANCH T,  T,
  THERE SWAP T! ;EMULATE      IMMEDIATE

: >L  CELL LP +!  LP @ ! ;
: L>  LP @ @  CELL NEGATE LP +! ;

: DO      \ — adrs  L: — 0
  ['] (DO) ,BRANCH  HERE  0 >L ;
EMULATE: M['] (DO) T,  THERE  0 T>L  ;EMULATE
IMMEDIATE

: LEAVE  ['] UNLOOP COMPILE,
  ['] BRANCH ,BRANCH  HERE DUP ,DEST  >L ;
EMULATE: M['] UNLOOP T,
  M['] BRANCH T,  THERE DUP T, T>L  ;EMULATE
IMMEDIATE

\  High level: control structures      (c) 21apr95 bjr
: ENDLOOP  ,BRANCH ,DEST  \ adrs xt —  L: 0 a1 a2 .. aN —
  BEGIN L> ?DUP WHILE [COMPILE] THEN REPEAT ;

ALSO FORTH ALSO META DEFINITIONS
: TENDLOOP  T, T, BEGIN TL> ?DUP WHILE THERE SWAP T!
REPEAT ;
PREVIOUS PREVIOUS DEFINITIONS

: LOOP  ['] (LOOP) ENDLOOP ;
EMULATE: M['] (LOOP) TENDLOOP  ;EMULATE  IMMEDIATE

: +LOOP  ['] (+LOOP) ENDLOOP ;
EMULATE: M['] (+LOOP) TENDLOOP  ;EMULATE
IMMEDIATE

\  High level: system variables and constants  (c) 21apr95 bjr
HEX  2 CONSTANT CELL   \ system dependent constant
  20 CONSTANT BL
  7E CONSTANT TIBSIZE

\  High level: system variables and constants  (c) 31mar95 bjr
HEX -80 USER TIB    \ — a-addr   Terminal Input Buffer
   0 USER U0     \ — a-addr   current user area adrs
   2 USER >IN    \ — a-addr   holds offset into TIB
   4 USER BASE    \ — a-addr   holds conversion radix
   6 USER STATE    \ — a-addr   holds compiler state
   8 USER DP     \ — a-addr   holds dictionary pointer
  0A USER 'SOURCE  \ — a-addr   two cells: length, address
  0E USER LATEST  \ — a-addr   last word in dictionary
  10 USER HP     \ — a-addr   HOLD pointer
  12 USER LP     \ — a-addr   leave-stack pointer
 100 USER S0     \ — a-addr   end of parameter stack
 128 USER PAD    \ — a-addr   user PAD buffer/end of hold
 180 USER L0     \ — a-addr   bottom of leave stack
 200 USER R0     \ — a-addr   end of return stack

\  High level: arithmetic operators      (c) 31mar95 bjr
: S>D      \ n — d   single -> double precision
  DUP 0< ;
: ?NEGATE   \ n1 n2 — n3   negate n1 if n2 negative
  0< IF NEGATE THEN ;
: ABS      \ n1 — n2    absolute value
  DUP ?NEGATE ;
: DNEGATE   \ d1 — d2    negate, double precision
  SWAP INVERT SWAP INVERT 1 M+ ;
: ?DNEGATE   \ d1 n — d2   negate d1 if n negative
  0< IF DNEGATE THEN ;
: DABS      \ d1 — d2    absolute value, double precision
  DUP ?DNEGATE ;

\  High level: arithmetic operators      (c) 31mar95 bjr
: M*      \ n1 n2 — d    signed 16*16->32 multiply
  2DUP XOR >R
  SWAP ABS SWAP ABS UM*
  R> ?DNEGATE ;

: SM/REM    \ d1 n1 — n2 n3   symmetric signed division
  2DUP XOR >R
  OVER >R
  ABS >R DABS R> UM/MOD
  SWAP R> ?NEGATE
  SWAP R> ?NEGATE ;

\  High level: arithmetic operators      (c) 31mar95 bjr
: FM/MOD    \ d1 n1 — n2 n3   floored signed division
  DUP >R
  SM/REM
  DUP 0< IF
    SWAP R> +
    SWAP 1-
  ELSE  R> DROP  THEN ;

: *       \ n1 n2 — n3      signed multiply
  M* DROP ;
: /MOD     \ n1 n2 — n3 n4    signed divide/remainder
  >R S>D R> FM/MOD ;
: /       \ n1 n2 — n3      signed divide
  /MOD NIP ;

\  High level: arithmetic operators      (c) 31mar95 bjr
: MOD      \ n1 n2 — n3     signed remainder
  /MOD DROP ;
: */MOD     \ n1 n2 n3 — n4 n5   n1*n2/n3, remainder&quotient
  >R M* R> FM/MOD ;
: */      \ n1 n2 n3 — n4     n1*n2/n3
  */MOD NIP ;

: MAX      \ n1 n2 — n3      signed maximum
  2DUP < IF SWAP THEN DROP ;
: MIN      \ n1 n2 — n3      signed minimum
  2DUP > IF SWAP THEN DROP ;

\  High level: double operators      (c) 31mar95 bjr
: 2@       \ a-addr — x1 x2    fetch 2 cells
  DUP CELL+ @ SWAP @ ;
: 2!       \ x1 x2 a-addr —    store 2 cells
  SWAP OVER ! CELL+ ! ;
: 2DROP     \ x1 x2 —        drop 2 cells
  DROP DROP ;
: 2DUP      \ x1 x2 — x1 x2 x1 x2   dup top 2 cells
  OVER OVER ;
: 2SWAP     \ x1 x2 x3 x4 — x3 x4 x1 x2   per diagram
  ROT >R ROT R> ;
: 2OVER     \ x1 x2 x3 x4 — x1 x2 x3 x4 x1 x2   per diagram
  >R >R 2DUP R> R> 2SWAP ;

\  High level: input/output      (c) 31mar95 bjr
HEX
: COUNT     \ c-addr1 — c-addr2 u   counted->addr/length
  DUP CHAR+ SWAP C@ ;
: CR       \ —          output newline
  0D EMIT 0A EMIT ;
: SPACE     \ —          output a space
  BL EMIT ;
: SPACES     \ u —          output u spaces
  BEGIN DUP WHILE SPACE 1- REPEAT DROP ;
: UMIN      \ u1 u2 — u        unsigned minimum
  2DUP U> IF SWAP THEN DROP ;
: UMAX      \ u1 u2 — u        unsigned maximum
  2DUP U< IF SWAP THEN DROP ;

\  High level: input/output      (c) 31mar95 bjr
: ACCEPT    \ c-addr +n — +n'   get line from terminal
  OVER + 1- OVER
  BEGIN KEY
  DUP 0D <> WHILE
    DUP EMIT
    DUP 8 = IF  DROP 1-  >R OVER R> UMAX
      ELSE  OVER C!  1+ OVER UMIN
    THEN
  REPEAT
  DROP NIP SWAP - ;
```

```
: TYPE    \ c-addr +n —     type line to terminal
  ?DUP IF
    OVER + SWAP DO I C@ EMIT LOOP
  ELSE DROP THEN ;

\ High level: input/output              (c) 31mar95 bjr
: (S")    \ — c-addr u     run-time code for S"
  R> COUNT 2DUP + ALIGN >R ;

ALSO FORTH ALSO META DEFINITIONS
: TS"  22 WORD DUP C@ 1+ THERE OVER
          TALLOT SWAP >TCMOVE ;
PREVIOUS PREVIOUS DEFINITIONS

: S"      \ —          compile in-line string
  ['] (S") COMPILE,
  22 WORD  C@ 1+ ALIGNED ALLOT ;
EMULATE: M['] (S") T, TS" ;EMULATE  IMMEDIATE

: ."      \ —          compile string to print
  [COMPILE] S" ['] TYPE COMPILE, ;
EMULATE: M['] (S") T, TS" M['] TYPE T,
                ;EMULATE  IMMEDIATE

\ High level: numeric output           (c) 31mar95 bjr
: UD/MOD    \ ud1 u2 — u3 ud4    32/16->32 divide
  >R 0 R@ UM/MOD  ROT ROT R> UM/MOD ROT ;
: UD*       \ ud1 u2 — ud3      32*16->32 multiply
  DUP >R UM* DROP  SWAP R> UM* ROT + ;
: HOLD      \ char —            add char to output string
  -1 HP +!  HP @ C! ;
: <#        \ —                 begin numeric conversion
  PAD HP ! ;
: >DIGIT    \ n — c             convert to 0..9A..Z
  DUP 9 > 7 AND + 30 + ;
: #         \ ud1 — ud2         convert 1 digit of output
  BASE @ UD/MOD ROT >DIGIT HOLD ;
: #S        \ ud1 — ud2         convert remaining digits
  BEGIN # 2DUP OR 0= UNTIL ;

\ High level: numeric output           (c) 31mar95 bjr
: #>        \ ud1 — c-addr u    end conversion, get string
  2DROP HP @ PAD OVER - ;
: SIGN      \ n —               add minus sign if n<0
  0< IF 2D HOLD THEN ;
: U.        \ u —               display u unsigned
  <# 0 #S #> TYPE SPACE ;
: .         \ n —               display n signed
  <# DUP ABS 0 #S ROT SIGN #> TYPE SPACE ;
: DECIMAL   \ —                 set number base to decimal
  0A BASE ! ;
: HEX       \ —                 set number base to hex
  10 BASE ! ;

\ High level: dictionary management    (c) 31mar95 bjr
: HERE      \ — addr            returns dictionary ptr
  DP @ ;
: ALLOT     \ n —               allocate n adr units in dict
  DP +! ;
: ,         \ x —               append cell to dict
  HERE ! 1 CELLS ALLOT ;
: C,        \ char —            append char to dict
  HERE C! 1 CHARS ALLOT ;

\ High level: interpreter              (c) 31mar95 bjr
: SOURCE    \ — adr n           current input buffer
  'SOURCE 2@ ;
: /STRING   \ a u n — a+n u-n   trim string
  ROT OVER + ROT ROT - ;
: >COUNTED  \ src n dst —       copy to counted string
  2DUP C! CHAR+ SWAP CMOVE ;
: WORD      \ char — c-addr     word delim'd by char
  DUP  SOURCE >IN @ /STRING
  DUP >R  ROT SKIP
  OVER >R  ROT SCAN
  DUP IF CHAR- THEN
  R> R> ROT -  >IN +!
  TUCK -
  HERE >COUNTED  HERE
  BL OVER COUNT + C! ;

\ High level: interpreter              (c) 31mar95 bjr
: NFA>LFA   \ nfa — lfa     name adr -> link field
  3 - ;
: NFA>CFA   \ nfa — cfa     name adr -> code field
  COUNT 7F AND + ;
: IMMED?    \ nfa — f       fetch immediate flag
  1- C@ ;
: FIND      \ c-addr — c-addr 0/1/-1  not found/immed/normal
  LATEST @ BEGIN            \ — a nfa
    2DUP OVER C@ CHAR+      \ — a nfa a nfa n+1
    S= DUP IF  DROP NFA>LFA @ DUP THEN  \ — a link link
  0= UNTIL                  \ — a nfa  OR  a 0
  DUP IF                    \ if found, check immed status
    NIP DUP NFA>CFA         \ — nfa xt
    SWAP IMMED? 0= 1 OR     \ — xt 1/-1
  THEN ;

\ High level: interpreter              (c) 31mar95 bjr
: LITERAL   \ x —       append numeric literal
  STATE @ IF ['] LIT COMPILE, I, THEN ;
EMULATES TLITERAL               IMMEDIATE

HEX
: DIGIT?    \ c — n -1 | x 0   true if c is a valid digit
  DUP 39 > 100 AND +        \ silly looking,
  DUP 140 > 107 AND - 30 -  \ but it works!
  DUP BASE @ U< ;
: ?SIGN     \ adr n — adr' n' f  get optional sign
  OVER C@                   \ — adr n c
  2C - DUP ABS 1 = AND      \ — +-1, -=+1, else 0
  DUP IF 1+                 \ +=0, -=+2      NZ=negative
    >R 1 /STRING R>         \ adr' n' f
  THEN ;

\ High level: interpreter              (c) 31mar95 bjr
: >NUMBER   \ ud adr u — ud' adr' u'  conv. string to number
  BEGIN DUP WHILE
    OVER C@ DIGIT?
    0= IF DROP EXIT THEN
    >R 2SWAP BASE @ UD*
    R> M+ 2SWAP  1 /STRING
  REPEAT ;

: ?NUMBER   \ c-addr — n -1 | c-addr 0   string->number
  DUP 0 0 ROT COUNT     \ — ca ud adr n
  ?SIGN >R  >NUMBER     \ — ca ud adr' n'
  IF R> 2DROP 2DROP 0   \ — ca 0  (error)
  ELSE 2DROP NIP R>
    IF NEGATE THEN -1   \ — n -1  (ok)
  THEN ;

\ High level: interpreter              (c) 31mar95 bjr
: INTERPRET  \ i*x c-addr u — j*x  interpret given buffer
  'SOURCE 2! 0 >IN !
  BEGIN BL WORD  DUP C@ WHILE     \ — textadr
    FIND ?DUP IF                  \ — xt 1/-1
      1+ STATE @ 0= OR            \ immed or interp?
      IF EXECUTE ELSE COMPILE, THEN
    ELSE                          \ — textadr
      ?NUMBER IF [COMPILE] LITERAL  \ converted ok
      ELSE  COUNT TYPE 3F EMIT CR ABORT  THEN  \ error
    THEN
  REPEAT DROP ;

: EVALUATE  \ i*x c-addr u — j*x  interpret string
  'SOURCE 2@ >R >R  >IN @ >R  INTERPRET
  R> >IN !  R> R> 'SOURCE 2! ;

\ High level: interpreter              (c) 28apr95 bjr
: QUIT      \ — R: i*x —    interpret from keyboard
  L0 LP !  R0 RP!  0 STATE !  \ reset stacks, state
  BEGIN
    TIB DUP TIBSIZE ACCEPT SPACE
    INTERPRET
    STATE @ 0= IF CR ." OK " THEN
  AGAIN ;

: ABORT     \ i*x — R: j*x —    clear stack and QUIT
  S0 SP!  QUIT ;
: ?ABORT    \ f c-addr u —      abort and print message
  ROT IF TYPE ABORT THEN 2DROP ;
: ABORT"    \ i*x 0 — i*x       abort, print inline msg
  [COMPILE] S" ['] ?ABORT COMPILE, ;
EMULATE: M['] (S") T, TS" M['] ?ABORT T,
                ;EMULATE IMMEDIATE

\ High level: interpreter              (c) 31mar95 bjr
: '         \ — xt      find word in dictionary
  BL WORD FIND  0= ABORT" ?" ;
: CHAR      \ — char    parse ASCII character
  BL WORD 1+ C@ ;
: [CHAR]    \ —         compile character literal
  CHAR ['] LIT COMPILE, I, ;  IMMEDIATE

: (         \ —         skip input until )
  29 WORD DROP ;        IMMEDIATE

\ High level: compiler                 (c) 31mar95 bjr
: CREATE    \ —         create an empty definition
  LATEST @ I, 0 IC,     \ link & immediate field
  IHERE LATEST !        \ new "latest" link
  BL IWORD IC@ 1+ IALLOT \ name field
  <DOCREATE> ,CF ;      \ code field

: RECURSE   \ —         recurse current definition
  LATEST @ NFA>CFA COMPILE, ;  IMMEDIATE

: [         \ —         enter interpretive state
  0 STATE ! ;  IMMEDIATE
: ]         \ —         enter compiling state
  -1 STATE ! ;

\ High level: compiler                 (c) 31mar95 bjr
HEX
: HIDE      \ —         "hide" latest definition
  LATEST @ DUP  IC@ 80 OR SWAP IC! ;
: REVEAL    \ —         "reveal" latest definition
  LATEST @ DUP  IC@ 7F AND SWAP IC! ;
: IMMEDIATE \ —         make last definition immediate
  1 LATEST @ 1- IC! ;
: [']       \ —         find word and compile as literal
  ' ['] LIT COMPILE, I, ;  IMMEDIATE

\ High level: compiler                 (c) 31mar95 bjr
: POSTPONE  \ —     postpone compile action of word
  BL WORD FIND  DUP 0= ABORT" ?"  \ find word
  0< IF ['] LIT COMPILE, I,  \ non-immed: compiles later
    ['] COMPILE, COMPILE,  \ add "LIT xt COMPILE," to df
  ELSE COMPILE,
    THEN ;  IMMEDIATE  \ immed: compile into df

\ High level: other operations         (c) 25apr95 bjr
: WITHIN  \ n1|u1 n2|u2 n3|u3 — f  n2<=n1<n3?
  OVER - >R - R> U< ;

: MOVE    \ addr1 addr2 u —   smart move
  >R 2DUP SWAP DUP R@ +
  WITHIN IF R> CMOVE> ELSE R> CMOVE THEN ;

: DEPTH   \ — n
  SP@ S0 SWAP - 2/ ;    \ 16 BIT VERSION!

: ENVIRONMENT?  \ c-addr u — i*x true   system query
  2DROP 0 ;  \           — false

\ High level: utility words            (c) 25apr95 bjr
: WORDS     \ —         list all words in dictionary
  LATEST @ BEGIN
    DUP COUNT TYPE SPACE
    NFA>LFA @
  DUP 0= UNTIL
  DROP ;
EMULATES WORDS

: .S        \ —         print contents of stack
  SP@ S0 - IF
    SP@ S0 2 - DO  I @ h.  -2 +LOOP
  THEN ;
EMULATES .S

\ High level: startup                  (c) 25apr95 bjr
: COLD      \ —         cold start Forth system
  UINIT U0 #INIT CMOVE
  ." 6809 CamelForth v1.0  25 Apr 95"  CR
  ABORT ;

\ Testing words
HEX
: .H ( n - )  0F AND 30 + DUP 39 > IF 7 + THEN EMIT ;
: .HH ( n - )  DUP 2/ 2/ 2/ 2/ .H .H ;
: .HHHH ( n - )  DUP 2/ 2/ 2/ 2/ 2/ 2/ 2/ 2/ .HH .HH ;
: H. ( n - )  .HHHH SPACE ;
: .B ( a - a+1 )  DUP C@ .HH SPACE 1+ ;
: DUMP ( a n - )  0 DO  DUP CR H. SPACE
  .B .B .B .B .B .B .B .B SPACE .B .B .B .B .B .B .B .B
  10 +LOOP DROP ;

\ 6809 DTC: reset initialization       (c) 25apr95 bjr
ASM: HERE EQU ENTRY  HEX
  CLRA, F000 STA, INCA, E000 STA, INCA, D000 STA,
  INCA, C000 STA, INCA, B000 STA, INCA, A000 STA,
  INCA, 9000 STA, INCA, 8000 STA, \ init mem mapping
  UP-INIT-HI # LDA,  A DPR TFR,  \ initial UP
  UP-INIT 100 + # LDS,          \ initial SP
  UP-INIT 200 + # LDU,          \ initial RP
  SCCATBL # LDX,  SCCINIT JSR,  \ init serial ports
  SCCBTBL # LDX,  SCCINIT JSR,
  ' COLD JMP,  ;C       \ enter top-level Forth word

ASM: HERE EQU IRET  RTI,  ;C
HERE 0FFF0 ORG  \ 6809 hardware vectors
  IRET, IRET, IRET, IRET,  \ tbd, SWI3, SWI2, FIRQ
  IRET, IRET, IRET, ENTRY,  \ IRQ, SWI, NMI, RESET
ORG

\ 6809 DTC: user area initialization   (c) 25apr95 bjr
DECIMAL 18 CONSTANT #INIT  \ # bytes of user area init data

CREATE UINIT  HEX
  0,0,0A,0,     \ reserved,>IN,BASE,STATE
  DP-INIT,      \ DP
  0,0,          \ SOURCE init'd elsewhere
META ALSO FORTH TLATEST @
        T, PREVIOUS TARGET  \ LATEST
  0,            \ HP init'd elsewhere
\ Note that UINIT must be the "last" word in the kernal, in
\ order to set the initial LATEST as shown above.  If this is
\ not the last word, be sure to patch the LATEST value above.
```

Regular Feature

Contact Listing

SUPPORT GROUPS FOR THE CLASSICS

TCJ Staff Contacts

TCJ Editor: Bill D. Kibler, PO Box 535, Lincoln, CA 95648, (916)645-1670, GEnie: B.Kibler, CompuServe: 71563,2243, E-mail: B.Kibler@Genie.com, tcj@psyber.com.

Z-System Support: Jay Sage,1435 Centre St. Newton Centre, MA 02159-2469, (617)965-3552, BBS: (617)965-7259; E-mail: Sage@ll.mit.edu. Also sells Z-System software.

32Bit Support: Rick Rodman, BBS:(703)330-9049 (eves),.

Kaypro Support: Charles Stafford, 4000 Norris Ave., Sacramento, CA 95821, (916)483-0312 (eves). Also sells Kaypro upgrades, see ad inside back cover. CompuServe 73664,2470 (73664.2470@cis).

S-100 Support: Herb Johnson, CN 5256 #105, Princeton, NJ 08543, (609)771-1503. Also sells used S-100 boards and systems, see inside back cover. E-mail: hjohnson@pluto.njcc.com.

6800/6809 Support: Ronald Anderson, 3540 Sturbridge Ct., Ann Arbor, MI 48105.

Regular Contributors:
Dave Baldwin, Voice/FAX (916)722-3877, or DIBs BBS (916) 722-5799 (use "computer", "journal", pswd "subscriber" as log on), Internet dibald@netcom.com, CompuServe 70403,2444.

Brad Rodriguez,Box 77, McMaster Univ., 1280 Main St. West, Hamilton, ONT, L8S 1C0, Canada, Genie: BJ, E-mail: bj@genie.com.

Frank Sergeant, 809 W. San Antonio St., San Marcos, TX 78666, E-mail: fs07675@academia.swt.edu.

Tilmann Reh, Germany, E-mail: tilmann.reh@hrz.uni-siegen.d400.de. Has many programs for CP/M+ and is active with Z180/280 ECB bus/Modular/Embedded computers. Microcontrollers (8051).

Helmut Jungkunz, Munich, Germany, ZNODE #51, 8N1, 300-14.4, +49.89.961 45 75, or CompuServe 100024,1545.

Ron Mitchell, Apt 1107, 210 Gloucester St., Ottawa Ontario, Canada, K2P 2K4. GEnie as R.Mitchell31, or CompuServe 70323,2267.

USER GROUPS

Connecticut CP/M Users Group, contact Stephen Griswold, PO Box 74, Canton CT 06019-0074, BBS: (203)665-1100. Sponsors Z-fests.

Sacramento Microcomputer Users Group, PO Box 161513, Sacramento, CA 95816-1513, BBS: (916)372-3646. Publishes newsletter, $15.00 membership, meetings at SMUD 6201 S st., Sacramento CA.

CAPDUG: The Capital Area Public Domain Users Group, Newsletter $20, Al Siegel Associates, Inc., PO Box 34667, Betherda MD 20827. BBS (301) 292-7955.

NOVAOUG: The Northern Virginia Osborne Users Group, Newsletter $12, Robert L. Crities, 7512 Fairwood Lane, Falls Church, VA 22046. Info (703) 534-1186, BBS use CAPDUG's.

The Windsor Bulletin Board Users' Group: England, Contact Rodney Hannis, 34 Falmouth Road, Reading, RG2 8QR, or Mark Minting, 94 Undley Common, Lakenheath, Brandon, Suffolk, IP27 9BZ, Phone 0842-860469 (also sells NZCOM/Z3PLUS).

L.I.S.T.: Long Island Sinclair and Timex support group, contact Harvey Rait, 5 Peri Lane, Valley Stream, NY 11581.

ADAM-Link User's Group, Salt Lake City, Utah, BBS: (801)484-5114. Supporting Coleco ADAM machines, with Newsletter / BBS.

Adam International Media, Adam's House, Route 2, Box 2756, 1829-1 County Rd. 130, Pearland TX 77581-9503, (713)482-5040. Contact Terry R. Fowler for information.

AUGER, Emerald Coast ADAM Users Group, PO Box 4934, Fort Walton Beach FL 32549-4934, (904)244-1516. Contact Norman J. Deere, treasurer and editor for pricing and newsletter information.

MOAUG, Metro Orlando Adam Users Group, Contact James Poulin, 1146 Manatee Dr. Rockledge FL 32955, (407)631-0958.

Metro Toronto Adam Group, Box 165, 260 Adelaide St. E., Toronto, ONT M5A 1N0, Canada, (416)424-1352.

Omaha ADAM Users Club, Contact Norman R. Castro, 809 W. 33rd Ave. Bellevue NE 68005, (402)291-4405. Suppose to be oldest ADAM group.

Vancouver Island Senior ADAMphiles, ADVISA newsletter by David Cobley, 17885 Berwick Rd. Qualicum Beach, B.C., Canada V9K 1N7, (604)752-1984.

Northern Illiana ADAMS User's Group, 9389 Bay Colony Dr. #3E, Des Plaines IL 60016, (708)296-0675.

San Diego OS-9 Users Group, Contact Warren Hrach (619)221-8246, BBS: (619)224-4878.

ACCESS, PO Box 1354, Sacramento, CA 95812, Contact Bob Drews (916)423-1573. Meets first Thurdays at SMUD 59Th St. (ed. bldg.).

Forth Interest Group, PO Box 2154, Oakland CA 94621 510-89-FORTH. International support of the Forth language, local chapters.

The Pacific Northwest Heath Users Group, contact Jim Moore, PO Box 9223, Seattle, WA 98109-0223.

The SNO-KING Kaypro User Group, contact Donald Anderson, 13227 2nd Ave South, Burien, WA 98168-2637.

SeaFOG (Seattle FOG User's Group, Formerly Osborne Users Group) PO Box 12214, Seattle, WA 98102-0214.

OTHER PUBLICATIONS

The Z-Letter, supporting Z-System and CP/M users. David A.J. McGlone, Lambda Software Publishing, 149 West Hillard Lane, Eugene, OR 97404-3057, (503)688-3563. Bi-Monthly user oriented newsletter (20 pages+). Also sells CP/M Boot disks, software.

The Analytical Engine, by the Computer History Association of California, 1001 Elm Ct. El Cerrito, CA 94530-2602. A ASCII text file distributed by Internet, issue #1 was July 1993. E-mail: kcrosby@crayola.win.net.

Z-100 LifeLine, Steven W. Vagts, 2409 Riddick Rd. Elizabeth City, NC 27909, (919)338-8302. Publication for Z-100 (a S-100 machine).

The Staunch 8/89'er, Kirk L. Thompson editor, PO Box 548, West Branch IA 52358, (319)643-7136. $15/yr(US) publication for H-8/89s.

The SEBHC Journal, Leonard Geisler, 895 Starwick Dr., Ann Arbor MI 48105, (313)662-0750. Magazine of the Society of Eight-Bit Heath computerists, H-8 and H-89 support.

Sanyo PC Hackers Newsletter, Victor R. Frank editor, 12450 Skyline Blvd. Woodside, CA 94062-4541, (415)851-7031. Support for orphaned Sanyo computers and software.

the world of 68' micros, by FARNA Systems, PO Box 321, Warner Robins, GA 31099-0321. E-mail: dsrtfox@delphi.com. New magazine for support of old CoCo's and other 68xx(x) systems.

Amstrad PCW SIG, newsletter by Al Warsh, 2751 Reche Cyn Rd. #93, Colton, CA 92324. $9 for 6 bi-monthly newsletters on Amstrad CP/M machines.

Historically Brewed, A publication of the Historical Computer Society. Bimonthly at $18 a year. HCS, 2962 Park Street #1, Jacksonville, FL 32205. Editor David Greelish. Computer History and more.

IQLR (International QL Report), contact Bob Dyl, 15 Kilburn Ct. Newport, RI 02840. Subscription is $20 per year.

QL Hacker's Journal (QHJ), Timothy Swenson, 5615 Botkins Rd., Huber Heights, OH 45424, (513) 233-2178, sent mail & E-mail, swensotc@ss2.sews.wpafb.af.mil. Free to programmers of QL's.

Update Magazine, PO Box 1095, Peru, IN 46970, Subs $18 per year, supports Sinclair, Timex, and Cambridge computers.

Other Support Businesses

Hal Bower writes, sells, and supports B/PBios for Ampro, SB180, and YASBEC. $69.95. Hal Bower, 7914 Redglobe Ct., Severn MD 21144-1048, (410)551-5922.

Sydex, PO Box 5700, Eugene OR 97405, (503)683-6033. Sells several CP/M programs for use with PC Clones ('22Disk' format/copies CP/M disks using PC files system).

Elliam Associates, PO Box 2664, Atascadero CA 93423, (805)466-8440. Sells CP/M user group disks and Amstrad PCW products. See ad inside back cover.

Discus Distribution Services, Inc. sells CP/M for $150, CBASIC $600, Fortran-77 $350, Pascal/MT+ $600. 8020 San Miguel Canyon Rd., Salinas CA 93907, (408)663-6966.

Microcomputer Mail-Order Library of books, manuals, and periodicals in general and H/Zenith in particular. Borrow items for small fees. Contact Lee Hart, 4209 France Ave. North, Robbinsdale MN 55422, (612)533-3226.

Star-K Software Systems Corp. PO Box 209, Mt. Kisco, NY 10549, (914)241-0287, BBS: (914)241-3307. SK*DOS 6809/68000 operating system and software. Some educational products, call for catalog.

Peripheral Technology, 1250 E. Piedmont Rd., Marietta, GA 30067, (404)973-2156. 6809/68000 single board system. 68K ISA bus compatible system. See inside front cover.

Hazelwood Computers, RR#1, Box 36, Hwy 94@Bluffton, Rhineland, MO 65069, (314)236-4372. Some SS-50 6809 boards and new 68000 systems.

AAA Chicago Computers, Jerry Koppel, (708)681-3782. SS-50 6809 boards and systems. Very limited quanity, call for information.

MicroSolutions Computer Products, 132 W. Lincoln Hwy, DeKalb, IL 60115, (815)756-3411. Make disk copying program for CP/M systems, that runs on CP/M sytems, UNIFROM Format-translation. Also PC/Z80 CompatiCard and UniDos products.

GIMIX/OS-9, GMX, 3223 Arnold Lane, Northbrook, IL 60062, (800)559-0909, (708)559-0909, FAX (708)559-0942. Repair and support of new and old 6800/6809/68K/SS-50 systems.

n/SYSTEMS, Terry Hazen, 21460 Bear Creek Rd, Los Gatos CA 95030-9429, (408)354-7188, sells and supports the MDISK add-on RAM disk for the Ampro LB. PCB $29, assembled PCB $129, includes driver software, manual.

Corvatek, 561 N.W. Van Buren St. Corvallis OR 97330, (503)752-4833. PC style to serial keyboard adapter for Xerox, Kaypros, Franklin, Apples, $129. Other models supported.

Morgan, Thielmann & Associates services NON-PC compatible computers including CP/M as well as clones. Call Jerry Davis for more information (408) 972-1965.

Jim S. Thale Jr., 1150 Somerset Ave., Deerfield IL 60015-2944, (708)948-5731. Sells I/O board for YASBEC. Adds HD drives, 2 serial, 2 parallel ports. Partial kit $150, complete kit $210.

Trio Company of Cheektowaga, Ltd., PO Box 594, Cheektowaga NY 14225, (716)892-9630. Sells CP/M (& PC) packages: InfoStar 1.5 ($160); SuperSort 1.6 ($130), and WordStar 4.0 ($130).

Parts is Parts, Mike Zinkow, 137 Barkley Ave., Clifton NJ 07011-3244, (201)340-7333. Supports Zenith Z-100 with parts and service.

DYNACOMP, 178 Phillips Rd. Webster, NY 14580, (800)828-6772. Supplying versions of CP/M, TRS80, Apple, CoCo, Atari, PC/XT, software for older 8/16 bit systems. Call for older catalog.

The Computer Journal

Back Issues

Sales limited to supplies in stock.

Volume Number 1:
- Issues 1 to 9
- Serial interfacing and Modem transfers
- Floppy disk formats, Print spooler.
- Adding 8087 Math Chip, Fiber optics
- S-100 Hi-RES graphics.
- Controlling DC meters, Multi-user column.
- VIC-20 EPROM Programmer, CP/M 3.0.
- CP/M user functions and integration.

Volume Number 2:
- Issues 10 to 19
- Forth tutorial and Write Your Own.
- 68000 CPU for S-100.
- RPM vs CP/M, BIOS Enhancements.
- Poor Man's Distributed Processing.
- Controlling Apple Stepper Motors.
- Facsimile Pictures on a Micro.
- Memory Mapped I/O on a ZX81.

Volume Number 3:
- Issues 20 to 26
- Designing an 8035 SBC
- Using Apple Graphics from CP/M
- Soldering & Other Strange Tales
- Build an S-100 Floppy Disk Controller: WD2797 Controller for CP/M 68K
- Extending Turbo Pascal: series
- Unsoldering: The Arcane Art
- Analog Data Acquisition & Control: Connecting Your Computer to the Real World
- Programming the 8035 SBC
- NEW-DOS: series
- Variability in the BDS C Standard Library
- The SCSI Interface: series
- Using Turbo Pascal ISAM Files
- The Ampro Little Board Column: series
- C Column: series
- The Z Column: series
- The SCSI Interface: Introduction to SCSI
- Editing the CP/M Operating System
- INDEXER: Turbo Pascal Program to Create an Index
- Selecting & Building a System
- Introduction to Assemble Code for CP/M
- Ampro 186 Column
- ZTime-1: A Real Time Clock for the Ampro Z-80 Little Board

Volume Number 4:
- Issues 26 to 31
- Bus Systems: Selecting a System Bus
- Using the SB180 Real Time Clock
- The SCSI Interface: Software for the SCSI Adapter
- Inside Ampro Computers
- NEW-DOS: The CCP Commands (continued)
- ZSIG Corner
- Affordable C Compilers
- Concurrent Multitasking: A Review of DoubleDOS
- 68000 TinyGiant: Hawthorne's Low Cost 16-bit SBC and Operating System
- The Art of Source Code Generation: Disassembling Z-80 Software
- Feedback Control System Analysis: Using Root Locus Analysis & Feedback Loop Compensation
- The C Column: A Graphics Primitive Package
- The Hitachi HD64180: New Life for 8-bit Systems
- ZSIG Corner: Command Line Generators and Aliases
- A Tutor Program in Forth: Writing a Forth Tutor in Forth
- Disk Parameters: Modifying the CP/M Disk Parameter Block for Foreign Disk Formats
- Starting Your Own BBS
- Build an A/D Converter for the Ampro Little Board
- HD64180: Setting the Wait States & RAM Refresh using PRT & DMA
- Using SCSI for Real Time Control
- Open Letter to STD Bus Manufacturers
- Patching Turbo Pascal
- Choosing a Language for Machine Control
- Better Software Filter Design
- MDISK: Adding a 1 Meg RAM Disk to Ampro Little Board, Part 1
- Using the Hitachi hd64180: Embedded Processor Design
- 68000: Why use a new OS and the 68000?
- Detecting the 8087 Math Chip
- Floppy Disk Track Structure
- Double Density Floppy Controller
- ZCPR3 IOP for the Ampro Little Board
- 3200 Hackers' Language
- MDISK: Adding a 1 Meg RAM Disk to Ampro Little Board, Part 2
- Non-Preemptive Multitasking
- Software Timers for the 68000
- Lilliput Z-Node
- Using SCSI for Generalized I/O
- Communicating with Floppy Disks: Disk Parameters & their variations
- XBIOS: A Replacement BIOS for the SB180
- K-OS ONE and the SAGE: Demystifying Operating Systems
- Remote: Designing a Remote System Program
- The ZCPR3 Corner: ARUNZ Documentation

Issue Number 32:
- 15 copies now available -

Issue Number 33:
- Data File Conversion: Writing a Filter to Convert Foreign File Formats
- Advanced CP/M: ZCPR3PLUS & How to Write Self Relocating Code
- DataBase: The First in a Series on Data Bases and Information Processing
- SCSI for the S-100 Bus: Another Example of SCSI's Versatility
- A Mouse on any Hardware: Implementing the Mouse on a Z80 System
- Systematic Elimination of MS-DOS Files: Part 2, Subdirectories & Extended DOS Services
- ZCPR3 Corner: ARUNZ Shells & Patching WordStar 4.0

Issue Number 34:
- Developing a File Encryption System.
- Database: A continuation of the data base primer series.
- A Simple Multitasking Executive: Designing an embedded controller multitasking executive.
- ZCPR3: Relocatable code, PRL files, ZCPR34, and Type 4 programs.
- New Microcontrollers Have Smarts: Chips with BASIC or Forth in ROM are easy to program.
- Advanced CP/M: Operating system extensions to BDOS and BIOS, RSXs for CP/M 2.2.
- Macintosh Data File Conversion in Turbo Pascal.

Issue Number 35:
- All This & Modula-2: A Pascal-like alternative with scope and parameter passing.
- A Short Course in Source Code Generation: Disassembling 8088 software to produce modifiable assem. source code.
- Real Computing: The NS32032.
- S-100: EPROM Burner project for S-100 hardware hackers.
- Advanced CP/M: An up-to-date DOS, plus details on file structure and formats.
- REL-Style Assembly Language for CP/M and Z-System. Part 1: Selecting your assembler, linker and debugger.

Issue Number 36:
- Information Engineering: Introduction.
- Modula-2: A list of reference books.
- Temperature Measurement & Control: Agricultural computer application.
- ZCPR3 Corner: Z-Nodes, Z-Plan, Amstrand computer, and ZFILE.
- Real Computing: NS32032 hardware for experimenter, CPUs in series, software options.
- SPRINT: A review.
- REL-Style Assembly Language for CP/M & ZSystems, part 2.
- Advanced CP/M: Environmental programming.

Issue Number 37:
- C Pointers, Arrays & Structures Made Easier: Part 1, Pointers.
- ZCPR3 Corner: Z-Nodes, patching for NZCOM, ZFILER.
- Information Engineering: Basic Concepts: fields, field definition, client worksheets.
- Shells: Using ZCPR3 named shell variables to store data variables.
- Resident Programs: A detailed look at TSRs & how they can lead to chaos.
- Advanced CP/M: Raw and cooked console I/O.
- ZSDOS: Anatomy of an Operating System: Part 1.

Issue Number 38:
- C Math: Handling Dollars and Cents With C.
- Advanced CP/M: Batch Processing and a New ZEX.
- C Pointers, Arrays & Structures Made Easier: Part 2, Arrays.
- The Z-System Corner: Shells and ZEX, new Z-Node Central, system security under Z-Systems.
- Information Engineering: The portable Information Age.
- Computer Aided Publishing: Introduction to publishing and Desk Top Publishing.
- Shells: ZEX and hard disk backups.
- Real Computing: The National Semiconductor NS320XX.
- ZSDOS: Anatomy of an Operating System, Part 2.

Issue Number 39:
- Programming for Performance: Assembly Language techniques.
- Computer Aided Publishing: The Hewlett Packard LaserJet.
- The Z-System Corner: System enhancements with NZCOM.
- Generating LaserJet Fonts: A review of Digi-Fonts.
- Advanced CP/M: Making old programs Z-System aware.
- C Pointers, Arrays & Structures Made Easier: Part 3: Structures.
- Shells: Using ARUNZ alias with ZCAL.
- Real Computing: The National Semiconductor NS320XX.

Issue Number 40:
- Programming the LaserJet: Using the escape codes.
- Beginning Forth Column: Introduction.
- Advanced Forth Column: Variant Records and Modules.
- LINKPRL: Generating the bit maps for PRL files from a REL file.
- WordTech's dBXL: Writing your own custom designed business program.
- Advanced CP/M: ZEX 5.0=The machine and the language.
- Programming for Performance: Assembly language techniques.
- Programming Input/Output With C: Keyboard and screen functions.
- The Z-System Corner: Remote access systems and BDS C.
- Real Computing: The NS320XX

Issue Number 41:
- Forth Column: ADTs, Object Oriented Concepts.
- Improving the Ampro LB: Overcoming the 8Mb hard drive limit.
- How to add Data Structures in Forth
- Advanced CP/M: CP/M is hacker's haven, and Z-System Command Scheduler.
- The Z-System Corner: Extended Multiple Command Line, and aliases.
- Programming disk and printer functions with C.
- LINKPRL: Making RSXes easy.
- SCOPY: Copying a series of unrelated files.

Issue Number 42:
- Dynamic Memory Allocation: Allocating memory at runtime with examples in Forth.
- Using BYE with NZCOM.
- C and the MS-DOS Screen Character Attributes.
- Forth Column: Lists and object oriented Forth.
- The Z-System Corner: Genie, BDS Z and Z-System Fundamentals.
- 68705 Embedded Controller Application: An example of a single-chip microcontroller application.
- Advanced CP/M: PluPerfect Writer and using BDS C with REL files.

Issue Number 43:
- Standardize Your Floppy Disk Drives.
- A New History Shell for ZSystem.
- Heath's HDOS, Then and Now.
- The ZSystem Corner: Software update service, and customizing NZCOM.
- Graphics Programming With C: Graphics routines for the IBM PC, and the Turbo C graphics library.
- Lazy Evaluation: End the evaluation as soon as the result is known.
- S-100: There's still life in the old bus.
- Advanced CP/M: Passing parameters, and complex error recovery.

Issue Number 44:
- Animation with Turbo C Part 1: The Basic Tools.
- Multitasking in Forth: New Micros F68FC11 and Max Forth.
- Mysteries of PC Floppy Disks Revealed: FM, MFM, and the twisted cable.
- DosDisk: MS-DOS disk format emulator for CP/M.
- Advanced CP/M: ZMATE and using lookup and dispatch for passing parameters.
- Forth Column: Handling Strings.
- Z-System Corner: MEX and telecommunications.

Issue Number 45:
- Embedded Systems for the Tenderfoot: Getting started with the 8031.
- The Z-System Corner: Using scripts with MEX.
- The Z-System and Turbo Pascal: Patching TURBO.COM to access the Z-System.
- Embedded Applications: Designing a Z80 RS-232 communications gateway, part 1.
- Advanced CP/M: String searches and tuning Jetfind.
- Animation with Turbo C: Part 2, screen interactions.
- Real Computing: The NS32000.

Issue Number 46:
- Build a Long Distance Printer Driver.
- Using the 8031's built-in UART for serial communications.
- Foundational Modules in Modula 2.
- The Z-System Corner: Patching The Word Plus spell checker, and the ZMATE macro text editor.
- Animation with Turbo C: Text in the graphics mode.
- Z80 Communications Gateway: Prototyping, Counter/Timers, and using the Z80 CTC.

Issue Number 47:
- Controlling Stepper Motors with the 68HC11F
- Z-System Corner: ZMATE Macro Language
- Using 8031 Interrupts
- T-1: What it is & Why You Need to Know
- ZCPR3 & Modula, Too
- Tips on Using LCDs: Interfacing to the 68HC705
- Real Computing: Debugging, NS32 Multitasking & Distributed Systems
- Long Distance Printer Driver: correction
- ROBO-SOG 90

Issue Number 48:

The Computer Journal Back Issues

- Fast Math Using Logarithms
- Forth and Forth Assembler
- Module-2 and the TCAP
- Adding a Bernoulli Drive to a CP/M Computer (Building a SCSI Interface)
- Review of BDS "Z"
- PMATE/ZMATE Macros, Pt. 1
- Z-System Corner: Patching MEX-Plus and TheWord, Using ZEX

Issue Number 49:

- Computer Network Power Protection
- Floppy Disk Alignment w/RTXEB, Pt. 1
- Motor Control with the F68HC11
- Controlling Home Heating & Lighting, Pt. 1
- Getting Started in Assembly Language
- LAN Basics
- PMATE/ZMATE Macros, Pt. 2
- Z-System Corner/ Z-Best Software

Issue Number 50:

- Offload a System CPU with the Z181
- Floppy Disk Alignment w/RTXEB, Pt. 2
- Motor Control with the F68HC11
- Module-2 and the Command Line
- Controlling Home Heating & Lighting, Pt. 2
- Getting Started in Assembly Language Pt 2
- Local Area Networks
- Using the ZCPR3 IOP
- PMATE/ZMATE Macros, Pt. 3
- Z-System Corner, PCED/ Z-Best Software
- Real Computing, 32FX16, Caches

Issue Number 51:

- Introducing the YASBEC
- Floppy Disk Alignment w/RTXEB, Pt 3
- High Speed Modems on Eight Bit Systems
- A Z8 Talker and Host
- Local Area Networks—Ethernet
- UNIX Connectivity on the Cheap
- PC Hard Disk Partition Table
- A Short Introduction to Forth
- Stepped Inference in Embedded Control
- Real Computing, the 32CG160, Swordfish,
- PMATE/ZMATE Macros
- Z-System Corner, The Trenton Festival
- Z-Best Software, the Z3HELP System

Issue Number 52:

- YASBEC, The Hardware
- An Arbitrary Waveform Generator, Pt. 1
- B.Y.O. Assembler...in Forth
- Getting Started in Assembly Language, Pt. 3
- The NZCOM IOP
- Servos and the F68HC11
- Z-System Corner, Programming for Compatibility
- Z-Best Software
- Real Computing, X10 Revisited
- PMATE/ZMATE Macros
- Controlling Home Heating & Lighting, Pt. 3
- The CPU280, A High Performance Single-Board Computer

Issue Number 53:

- The CPU280
- Local Area Networks
- Am Arbitrary Waveform Generator
- Zed Fest '91
- Getting Started in Assembly Language
- The NZCOM IOP
- Z-BEST Software

Issue Number 54:

- B.Y.O. Assembler
- Local Area Networks
- Advanced CP/M
- ZCPR on a 16-Bit Intel Platform
- Real Computing
- Interrupts and the Z80
- 8 MHZ on a Ampro
- Hardware Heavenn
- What Zilog never told you about the Super8
- An Arbitary Waveform Generator
- The Development of TDOS

Issue Number 55:

- Fuzzilogy 101
- The Cyclic Redundancy Check in Forth
- The Internetwork Protocol (IP)
- Hardware Heaven
- Real Computing
- Remapping Disk Drives through the Virtual BIOS
- The Bumbling Mathmatician
- YASMEM
- Z-BEST Software

Issue Number 56:

- TCJ - The Next Ten Years
- Input Expansion for 8031
- Connecting IDE Drives to 8-Bit Systems
- 8 Queens in Forth
- Kaypro-84 Direct File Transfers
- Analog Signal Generation

Issue Number 57:

- Home Automation with X10
- File Transfer Protocols
- MDISK at 8 MHZ.
- Shell Sort in Forth
- Introduction to Forth
- DR. S-100
- Z AT Last!

Issue Number 58:

- Multitasking Forth
- Computing Timer Values
- Affordable Development Tools
- Mr. Kaypro
- DR. S-100

Issue Number 59:

- Moving Forth
- Center Fold IMSAI MPU-A
- Developing Forth Applications
- Mr. Kaypro Review
- DR. S-100

Issue Number 60:

- Moving Forth Part II
- Center Fold IMSAI CPA
- Four for Forth
- Debugging Forth
- Support Groups for Classics
- Mr. Kaypro Review
- DR. S-100

Issue Number 61:

- Multiprocessing 6809 part I
- Center Fold XEROX 820
- Quality Control
- Real Computing
- Support Groups for Classics
- Operating Systems - CP/M
- Mr. Kaypro 5MHZ

Issue Number 62:

- SCSI EPROM Programmer
- Center Fold XEROX 820
- DR S-100
- Moving Forth part III
- Programming the 6526 CIA
- Reminiscing and Musings
- Modem Scripts

Issue Number 63:

- SCSI EPROM Programmer part II
- Center Fold XEROX 820
- DR S-100
- Multiprocessing Part II
- 6809 Operating Systems
- Reminiscing and Musings
- IDE Drives Part II

Issue Number 64:

- Small-C?
- Center Fold last XEROX 820
- DR S-100
- Moving Forth Part IV
- Small Systems
- Mr. Kaypro
- IDE Drives Part III

Issue Number 65:

- Small System Support
- Center Fold ZX80/81
- DR S-100
- Real Computing
- European Beat
- PC/XT Corner
- Little Circuits
- Levels of Forth
- Sinclair ZX81

Issue Number 66:

- Small System Support
- Center Fold: Advent Decoder
- DR S-100
- Connecting IDE Drives
- PC/XT Corner
- Little Circuits
- Multiprocessing Part III
- Z-System Corner

Issue Number 67:

- Small System Support
- Center Fold: SS-50/SS-30
- DR S-100
- Serial Kaypro Interrupts
- Little Circuits
- Moving Forth Part 5
- European Beat

Issue Number 68:

- Small System Support
- Center Fold: Pertec/Mits 4PIO
- Z-System Corner II
- PC/XT Corner
- Little Circuits
- Multiprocessing Forth Part 4
- Mr. Kaypro

Issue Number 69:

- Small System Support
- Center Fold: S-100 IDE
- Z-System Corner II
- Real Computing
- PC/XT Corner
- DR. S-100
- Moving Forth Part 6
- Mr. Kaypro

Issue Number 70:

- Small System Support
- Center Fold: Jupiter ACE
- Z-System Corner II
- PC/XT Corner: Stepper Motors
- DR. S-100
- Multiprocessing Part 5
- European Beat

Issue Number 71:

- Computing Hero of 1994
- Small System Support
- Center Fold: Hayes 80-103A
- Power Supply Basics
- PC/XT Corner: Stepper Motors
- DR. S-100
- Moving Forth Part 7
- Mr. Kaypro
- 8048 Emulator Part 1

Issue Number 72

- Beginning PLD
- Small System Support
- Center Fold: Rockwell R65F11
- Playing With Micros
- Real Computing
- Small Tools Part 1
- DR. S-100
- Moving Forth Part 7.5
- 8048 Emulator Part 2

Issue Number 73

- $10 XT
- Small System Support
- Center Fold: 640K XT
- IDE Part 6
- Real Computing
- Small Tools Part II
- DR. S-100
- Mr. Kaypro
- PC/XT Corenr
- 8048 Emulator Part 3

Subscriptions (CA not taxable)	U.S.	Canada/Mexico (Surface)	Canada/Mexico (Air)	Europe/Other (Surface)	Europe/Other (Air)
1year (6 issues)	$24.00	$32.00	$34.00	$34.00	$44.00
2 years (12 issues)	$44.00	$60.00	$64.00	$64.00	$84.00
Back Issues (CA tax)	add these shipping costs for each issue ordered				
Bound Volumes $20.00 ea	+$3.00	+$3.50	+$6.50	+$4.00	+$17.00
#20 thru #43 are $3.00 ea.	+$1.00	+$1.00	+$1.25	+$1.50	+$2.50
#44 and up are $4.00ea.	+$1.25	+$1.25	+$1.75	+$2.00	+$3.50

Items: _____

_____

_____ Back Issues Total _____

_____ Shipping Total _____

California state Residents add 7.25% Sales TAX _____

Subscription Total _____

Total Enclosed _____

Name: _____

Address: _____

_____

_____

_____

Credit Card # _____-_____-_____-_____ exp ___/___

Payment is accepted by check, money order, or Credit Card (M/C, VISA, CarteBlanche, Diners Club). Checks must be in US funds, drawn on a US bank. Credit Card orders can call 1(800) 424-8825.

TCJ The Computer Journal

P.O. Box 535, Lincoln, CA 95648-0535
Phone (916) 645-1670

Regular Feature

Editorial Comment

Projects?

The Computer Corner

By Bill Kibler

Welcome to number 74 or about my 65th Computer Corner. One of these days I must research exactly how many of these columns I have done. It sure seems like for ever, but one loses track of time while having fun.

Pending Projects

Their are a number of projects looking for feedback or assistance before completion. Building a PLC (Programmable Logic Controller) to use any parallel port to control industrial items is high on the list. We need some feedback about using the program INTERLNK to transfer files on non-standard PCDOS machines (see letter from Lee Hart in #73.) Need the latest status of CheapNet for ZDOS machines. The GIDE should be in use by next issue we hope. And some work on *TCJ*'s CDROM is over due. The biggest help however will be getting some assistant editors to take over a part of the work load.

If you have been considering the idea, let me fill you in on some of the missing ideas not covered in the flyer that everyone should have gotten. For many issues we have been focusing on supporting three fronts or interrelated topics. We have always supported eight bit systems, strongly those running CP/M, but now almost any CPU type in use. Those eight bitters are now used extensively in small embedded and robotic systems, and so we spread our basic teaching skills into that area as well. After all I find the tools and needed skills are identical whether it is a CP/M program or opening and closing doors.

Since part of supporting older systems is helping out when the manufacturer forgets they ever produced the product, PC/XTs naturally dropped into our laps. You readers have always been using these systems, usually without any fun at work, and thus we hoped to bring the hobby back into dealing with those unfriendly machines. Since many of us like to tinker with operating systems just for fun (yes I know, we must be a bit brain dead to think of that as fun - but each person has their quirks), we have started playing with Linux and MINIX.

Having provided the above coverage for many issues now, but mostly in a anything-goes approach that prevented others from helping, minor changes were needed. These changes are really cosmetic and designed to make it possible for others to take on parts of the work load. As editor I will still grind out the issue and fill many pages. The difference will be having someone else provide the main focus material twice a year. That means the assistant editor has just under six months to seek out those articles, or complete that set of tests waiting to be done, and edit the results for me to include in the next issue.

Now don't get me wrong, I will have the editors ship me some of your letters to be put in reader to reader, as well as the item that just can't wait till the next focus issue. Most of our regulars have been rather busy and as you know occasionally miss an issue. This change just formalizes that happening and gives them more time to be really late than just sort of late. Overall it should still keep you well informed, educated, entertained, and wondering what will happen next.

My Antiques

Well I was looking for the centerfold material this issue and had trouble finding it for all the older systems and disks. I have been collecting and sorting disks for the pending *TCJ* CDROM and thus have piles of them all over the place. What got me thinking was how some of the items were sitting out there waiting for renewed interest, and all the history and stories that need telling, but by whom?

As I stated in the Antique article, we really have not seen much history recounted about the early days. When I gave David Jaffe his actual award at the last Forth meeting, he started talking about way back when. I could see by the glint in his eye that those times were indeed wonderful and exciting and needing to be retold. What about it David, a few stories? What about you other readers, where are those stories? I enjoyed the short comments made in Dr. S-100 by Doug Jones about his experience with those first Altair's stuck in the University. How about your stories?

Repairing Antiques

I need articles on fixing and repairing these older systems as well. I get plenty of letters saying how they just started collecting and need help finding and fixing this or that system. The best call lately was from a person who had just opened the box they purchased their Kaypro in, for the first time. A real untouched Kaypro being brought to life after 12 years! What fun, but where did they put that boot disk?

At a recent SMUG meeting the topic surfaced about fixing connectors. The person said no erasers. Well now wait a minute, I have been doing that for too many years to think about, and now that's all wrong. Well...

Actually it has been an active topic for some time. The problem we are speaking about here are the gold plated edge connectors on S-100, STD bus cards, and any number of drive and system sockets. You can have a bit of flaky operation, where it works find for awhile, then stops, you tap it, it works again for awhile and so on. The usual problem is oxidation on the gold surface. This adds up to some resistance or occasionally complete loss of contact and conduction of electrons.

For many years everyone used a soft pencil eraser to burnish up the surface and renew operation. I did it many times at two in the morning so I could go back home to bed. The problem is, you are actually removing some of the gold, not just changing the chemical makeup of the surface. Gold wipes are probably the best for the newer boards. I say newer since I feel the gold surface is probably thinner than the older boards. Being thinner and maybe even different formulation will allow for the wipes to work fine.

Simply remove the boards one at a time and wipe the pins down well with the chemical saturated cloth. You should see a gray coloring in the cloth showing that it has worked. Now I have tried them on some of the very old S-100 boards with no effect. The oxidation is far too thick for these proper tools to work. So what do I do, get my eraser out and polish up, hopefully while no one is watching. Watch out for the eraser crumbs, if they fall into the fingers of the socket, they can be worse than the oxidation.

I must stress that you need also to look at the socket and attempt to clean them as well. Although I find the board edge the most likely problem, and cleaning it usually solves things, there is no reason that the socket should not be just as oxidized as the board. Also check for bent or damaged fingers, even a small piece of wire or solder that might short out the contacts. I don't know how many times I have seen other techs pull a card and never once look at the socket. I found a pin smashed against the opposite side and thus kept a PC from running.

PS2's

The IBM PS2 machine is starting to appear on the used market. My advice for collectors is to stay far away from them, unless you know the technical details of them. The first problem is the need for set up disks. If any cards are removed, you must have the proper setup disk to get the system to boot. I was an un-official support person for them a few jobs back and had a full box of setup disks for each of the different systems possible.

Not only is the setup disk needed for the model number (such as 55SX) but there may be various board changes that will require a special disk ('planar' 55SX). I checked out the IBM support section on CompuServe and found many of the disks there. The sizes run from 500K to 750K in a image file. They require a special loader program available in the same directory that takes the file and formats a disk with the programs needed.

These disks have their own boot sectors and DOS, as well as files to check out the system and alter the CMOS control information. There is also an assortment of information files that go with each card you want to use. The setup program reads a serial number from the card's onboard ROM and then goes looking on the root directory for a file with that name. The file contains the addresses used by the card and how it can be changed.

The idea was to have a very simple setup system for non-literate users. The problem is that there are far too many options for the system to properly handle and thus a super user must often get in and hand edit the setup information to keep the conflicts from shutting down the machine. I also discovered that some cards had unlisted RAM or ROM portions that the setup would assign to other cards. All this equated to having a system that ended up being far from the simple concept IBM though they had designed.

I am sure there are plenty of books and manuals that will help you out, assuming you can get them when you buy one of the old machines. You will be surprised however to find that many of the units are slower than similar clone units. Besides the special boards, hard drives often had built in controllers making upgrading another problem. Oh, and don't try loading Linux or Unix as the PS2's are so non-standard you can't get them to work with anything other than a plain DOS.

What Speed

Another topic surfaced at the SMUG meeting after my talk, it seems one of the engineers from Intel spoke about the 486/586 and how they worked on the various buses in use. Now since I wasn't actually there, this is pretty much second or third hand information, which I would like clarified in letters or articles if possible. The slant went something like this, the 486 is basically faster than 586 when dealing with I/O. The PCI bus is a better design, but slower than some other options. Missed cache hits are probably killing the 586, making the 486 with it's smaller cache faster. Intel knows this and is designing the 686 to solve some of the problems. The person telling us this, says the speaker had charts and graphs that clearly proved the points being made. The speaker's final words were that adding more memory, about 32 megs, basically solved all the problems.

I think the information is some what correct as I stated, however I know I missed some of what was said, and of course not being there myself, didn't see the real proof in the overheads. I did mention the topic to a fellow worker who does heavy graphics and he indicated it matched up with his experiences. His 486 graphic machine could do better in some cases than his newest 586 which he can't get to run if the cache is turned on. And yes 32 megs did make a BIG difference.

I have been reluctant, mostly due to cost, to get a 586 machine until now. Now, I add more questions about real speed and compatibility to the list of concerns, and now add buying 32 megs of RAM as well. Well that's it...Keep hacking.
Bill.

TCJ CLASSIFIED

CLASSIFIED RATES!
$5.00 PER LISTING!

TCJ Classified ads are on a prepaid basis only. The cost is $5.00 per ad entry. Support wanted is a free service to subscribers who need to find old or missing documentation or software. Please limit your requests to one type of system.

Commercial Advertising Rates:

Size	Once	4+
Full	$120	$90
1/2 Page	$75	$60
1/3 Page	$60	$45
1/4 Page	$50	$40
Market Place	$25	$100/yr

Send your items to:
The Computer Journal
P.O. Box 535
Lincoln, CA 95648-0535

Historically Brewed. The magazine of the Historical Computer Society. Read about the people and machines which changed our world. Buy, sell and trade "antique" computers. Subscriptions $18, or try an issue for $3. HCS, 2962 Park Street #1, Jacksonville, FL 32205

Start your own technical venture! Don Lancaster's newly updated **INCREDIBLE SECRET MONEY MACHINE II** tells how. We now have autographed copies of the Guru's underground classic for $21.50. Synergetics Press, Box 809-J, Thatcher AZ, 85552.

THE CASE AGAINST PATENTS Throughly tested and proven alternatives that work in the real world. $33.50. Synergetics Press, Box 809-J, Thatcher AZ, 85552.

FOR SALE: Two Kayrpo II Computers. Each is portable with keyboard, monochrome monitor, dual 5 1/4 floppy drives and 64K memory. Also included, one padded carrying case, RS232 cable, many orginal applications (CP/M, word processor, accounting, communications, etc.) and manuals. Whole package - postpaid $160. Call (914) 331-5440, ask for Michael. (in NY)

Help Wanted: Need following items for Vector Graphics SX-3000: 1) Maintenance manual; 2) Programmers Diskette (CP/M86); 3) Vector SX MS-DOS 2.11 Users Diskette. I have all other documentation, including CP/M, MS-DOS, Programmers guide, Concurrent CP/M users guide and a CP/M Boot disk. Also trying to locate bootdisk and manuals for Vector 4. Clarence Dolan, 4553 White Horse Trail, Rockford, IL 61101-6147.

TCJ ADS WORK!

Classified ads in *TCJ*
get results, FAST!
Need to sell that special older system - TRY *TCJ*.
World Wide Coverage with Readers interested in what **YOU** have to sell.
Provide a support service, our readers are looking for assistance with their older systems - all the time.
The best deal in magzines,
TCJ Classified
it works!

Wanted: Have Apple IIgs, need to get hard drive and modem, but where??? David Kruszyna, 19055 Lister Ave. Eastpointe MI, 48021-2739.