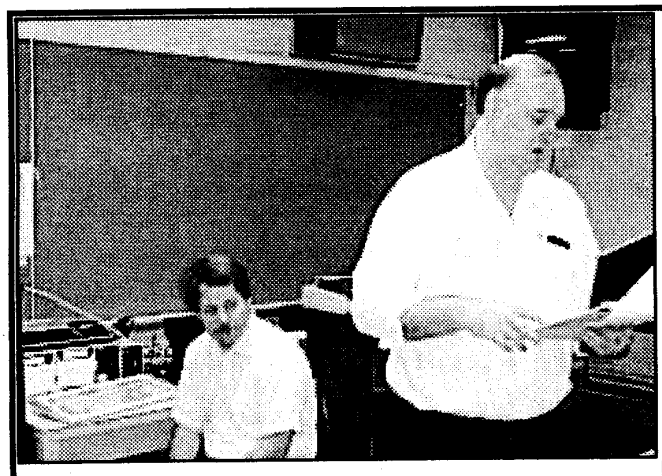
Covering the TI99/4A and the Myarc 9640

MICROpendium

Volume 14 Number 4 July/August 1997 $6



LIMA MULTI USER GROUP CONFERENCE

see page 6

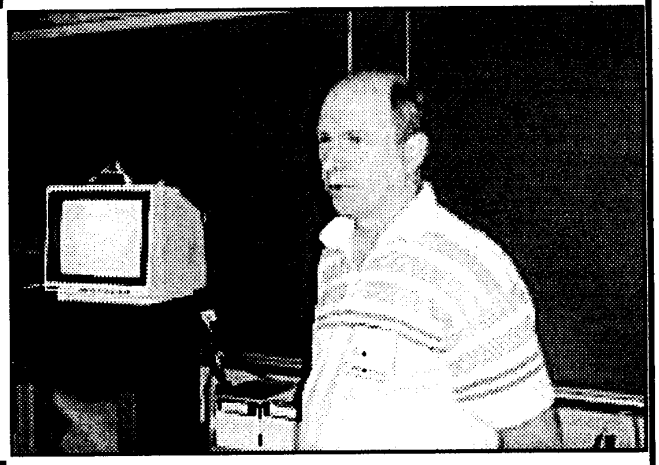
At left, Mark Van Coppenole and Mike Wright of CaDD Electronics. Below, Bob Carmany gives a presentation. Bottom, Jim Krych presents the Jim Peterson Hardware Award to the SouthWest Ninety-Niners, represented by Matt Mathews.

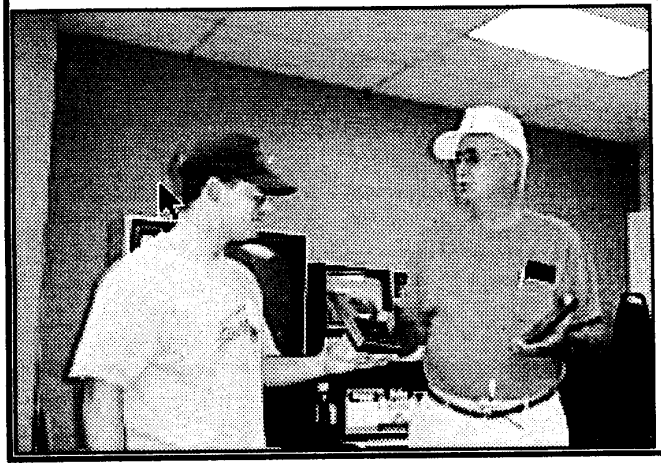
Cover and inside photos from the MUG Web page

REVIEWS

Simplicity

PC99 Stage 4





ALSO

Having fun with c99

Extended XBASICs

Plans for a Lubbock Fest West

and more!

CONTENTS

MICROpendium

MICROpendium (ISSN 10432299) is published bimonthly for $35 per year by Burns-Koloen Communications Inc., 502 Windsor Rd., Round Rock, TX 78664-7639. Periodical postage paid at Round Rock, Texas. POSTMASTER: Send address changes to MICROpendium, P.O. Box 1343, Round Rock, TX 78680-1343.

Display advertising deadlines and rates are available on request.

All correspondence should be mailed to MICROpendium at P.O. Box 1343, Round Rock, TX 78680. We cannot take responsibility for unsolicited manuscripts but will give consideration to anything sent to the above address. Manuscripts will be returned only if a self-addressed stamped envelope is included.

Foreign subscriptions are $40.25 (Mexico); $42.50 (Canada); $40.00, surface mail to other countries; $52 airmail to other countries.

All editions of MICROpendium are mailed from the Round Rock (Texas) Post Office.
Mailing address: P.O. Box 1343, Round Rock, TX 78680.
Telephone & FAX: (512) 255-1512
Delphi TI NET: MICROpendium
Internet E-mail: jkoloen@io.com
Home page: http://www.io.com/~jkoloen/
John Koloen.....Publisher
Larua Burns.....Editor

*READ THIS

Here are some tips to help you when entering programs from MICROpendium:
1. Most BASIC and Extended BASIC programs are run through Checksum, which places the numbers that follow exclamation points at the end of each program line. Do not enter these numbers or exclamation points. Checksum is available on disk from MICROpendium for $4.
2. Long Extended BASIC lines are entered by inputting until the screen stops accepting characters, pressing Enter, pressing FCTN REDO, cursoring to the end of the line and continuing input.

COMPETITION COMPUTER 800-471-1600

(Nationwide & Canada ORDERS ONLY)

SUMMER DISK SALE:
USED DISK BLOWOUT 10 FOR $2.50(NO DOCS)+$1 SHIPPING
ASSORTED COMMERCIAL/SHAREWARE/PUBLIC DOMAIN
ANY OF THE FOLLOWING TITTLES FOR $5+$1 SHIPPING:

ARTIST CATALOGER
ARTIST PRINT SHOP
BACKSTEINE
50 BACKSTEINE SCREENS
CARD/FLYER GRAPHICS
CARDS FOR ALL OCCASIONS
CERTIFICATE 99 GRAPHIC COMPANION
CERTIFICATE 99 COMPANION PLUS
CERTIFICATE 99 COMPANION #2
COLOR BANNERS FOR COLOR DOT MATRIX
COLOR CARD
COLOR FLYER
CSGD GRAPHICS
FORMSHOP
GIANT ARTIST POSTERS
JIFFY FLYER
JIFFY CARD
JOY PAINT 99
JOY PAINT PAL
THE LIVING TOMB
MINE FIELD
PICTURE IT
WAR ZONE
WAR ZONE II
YUCAN BUSINESS MANAGER

INFOCOM COMPLETE WITH ALL GOODIES/CLUES $20 EACH
ALL ORIGINAL PACKAGING LIMITED SUPPLY SHIPPING INCLUDED
PLANETFALL
DEADLINE
CUTTHROATS
SUSPENDED

COLLECTORS CORNER:
TOMY TUTOR CONSOLE WITH XEROXED MANUAL $40
TOMY TUTOR JOYSTICK $10
TOMY TUTOR MANUAL ONLY(XEROX) $5
CONTROL DATA CONSOLES $40
CONTROL DATA POWER SUPPLIES $10
CONTROL DATA PE BOXES $40
CONTROL DATA CARDS FOR PEB ALSO AVAILABLE
COMPLETE CONTROL DATA SYSTEM(ALL OF ABOVE PLUS 32K,DRIVE
DISK CONTROLLER,FLEX CABLE,CONSOLE,POWER SUPPLY $150
WE HAVE A GOOD SELECTION OF ORIGINAL CONTROL DATA SOFTWARE
TI 99/4 RF MODULATOR HANDBUILT BY TI WITH ORIGINAL MANUAL $20

TI CDROM OWNERS:
WE NEED TO HEAR
FROM YOU AS TO
WHAT YOU WOULD
LIKE TO HAVE IN
THE NEXT ISSUE!

AS A MICROSOFT OFFICE 97
OWNER I STILL USED MY TI
TO DO THIS AD!

BUDGET MINDED IBM OWNERS NOTE: YOU CAN HAVE YOUR TI LIVE ON
INSIDE YOUR IBM FOR AS LITTLE AS $49 AND WE WILL THROW IN OUR
CD ROM FOR A PACKAGE PRICE OF ONLY $79 +$3 SHIPPING
PC99 VERSION 4 FULL IS ALSO AVAILABLE FOR ALL YOU HACKERS FOR $99
OR $130 WHEN BUNDLED WITH OUR CD ROM

COLLECTORS OF CARTRIDGES: DO YOU HAVE A GRAMCRACKER FILE OF SOME
SCARCE CARTRIDGE THAT YOU WOULD LIKE TO BE ABLE TO RUN ON AN ORDINARY
TI CONSOLE? OUR UNIVERSAL GROM EMULATOR CARTRIDGE IS THE ANSWER!!
WE ARE COLLECTING GROM CODE FOR UNRELEASED OR EUROPE ONLY RELEASE CARTS
AND PURSUEING APPROPRIATE PERMISSION TO RELEASE THESE CARTS BUT WE NEED
THAT GROM FILE! LETS TALK. WANTED:TOMY TUTOR CARTS

COMPETITION COMPUTER 1.800.471.1600 OR 415.697.1108
SEND SELF ADDRESSED STAMPED LONG ENVELOPE FOR OUR CATALOG
350 MARCELLA WAY,MILLBRAE CA 94030

COMMENTS

Congratulations, Peterson winners

Congratulations go to this year's winners of the Jim Peterson awards. They are:

Geneve 9640.....Tim Tesch
TI Community.....Tom Wills
TI Hardware .....Southwest 99ers User Group
TI Software.....Bruce Harrison

These selections were excellent.

MUG LOOKED INTERESTING

There seemed to be a lot going on at this year's MUG conference, as you'll see in the write-ups included this month. Congratulations are in order to Charlie Good and the Lima user group for putting on another successful show.

Plans for next year's Fest West in Lubbock are well under way. As you'll see later in this issue, Tom Wills is thinking big, reserving 100 rooms at one hotel while scouting out others. (So, what can you do in Lubbock after the fair? You can visit the Buddy Holly museum, for starters.)

PC99 STAGE 4 HAS ARRIVED

Charlie Good reviews the latest version of PC99 this month, and as far as I am concerned his high marks are well-deserved. I've been using version 4 for a month or so and find that it is absolutely terrific. I've tried emulators for a bunch of different machines (Commodore, Atari, etc.) and none of them even comes close to what Mike Wright and CaDD Electronics have accomplished with this program, or more accurately, collection of programs. Most other emulators are either crippled by lack of complete development or users are abandoned with no support. To the contrary, CaDD provides an excellent, fully developed product and excellent support after the sale. I've used V9T9, but development of it stopped well short of completion. If you're serious about emulating a TI99/4A on a PC, there's really only one choice. But it's a good one.

—JK

FEEDBACK

Awards appreciated

The SouthWest Ninety Niners User Group and I, Tom Wills, wish to express our heartfelt "Thank You" to the entire TI community for the Jim Peterson Awards we received.

As president of the SouthWest Ninety Niners User Group, I am very proud to extend the "Thank You" on behalf of the entire SW99UG membership. Our award for the hardware category is one that is indeed shared by each and every member of this fine user group. Every member was involved in one way or another in the production of the Super AMS card. Either as designers, investors (by putting up the money so we could financially start up production), assemblers, testers, salespeople or even as end users. From each and every member comes a heartfelt "Thank You!"

I personally wish to express my heartfelt "Thank You" to the members of the TI Community for voting me the Jim Peterson Community Award. I am proud to have been awarded the award named after such a fine person as Jim Peterson. Having personally know Jim Peterson from his attendance at the Wisconsin 99er Computer Council's TI Fairs that were held in Milwaukee, this award means so much more to me.

Tom Wills
Tucson, Arizona

Second thoughts on Juno problems

I am a former TI99/4A user and loyal subscriber of MICROpendium. I am still a member of the Dallas TI Home Computer Users Group (Ditty Hug). I was very flattered to see my Ditty Hug newsletter article in your last issue.

I would like to correct inaccuracies in that Juno article.

All of the problems reported with Juno were caused by my expensive brand-new U.S. Robotics Sportster Modem. In the past I had owned seven different modems — none of which ever gave me a moment's trouble. So when I bought a brand-new modem, that expensive, and installed a new shareware application (Juno), I assumed my problems were with Juno.

It is true Juno has had some start-up pains, mainly with their phone line provider. Each change in phone providers has resulted in more rapid response. Unfortunately, the 1-800 service has been discontinued. This has eliminated my daughter (North Texas) and an old friend in East Texas. I am a tense Internet user and can report that e-mail on Juno is vastly superior to Netscape's e-mail.

It is true there is advertising on Juno. It is colorful, well done and I think unobtrusive. Messages go through rapidly and accurately. Juno does not support e-mail at-

(See Page 5)

FEEDBACK

(Continued from Page 4)

tachments. This may be a plus. At the present there is a message on my Internet e-mail which cannot be downloaded nor deleted. This renders the Netscape e-mail option useless.

I still have my non-functioning, expensive U.S. Robotics Modem. It has a five-year warranty, but because of the way they handle communications, it is impossible to get an acknowledgement from them about warranty repairs or replacements. They will not accept a returned product without a "RMA" number and repeated attempts to contact U.S. Robotics to get a "RMA" number have met with failure. I remember precious little of this type of business practice in the 4A world.

Ditty Hug is alive and well. Alas, most of our meetings are social. We meet at the Infomart, usually on the second Saturday (it varies). You and all other loyal 4A aficionados are invited to join us — call (817) 267-5987 to verify a meeting date. We had 23 loyal members and guests at our June meeting.

Tom Hall
Euless, Texas

Getting on line

I've looked forward to the last year's issues of our mag. They are real life savers. When confidence wanes, I turn to MICROpendium pages. I do hope you can keep the good work up even in these lean times

Since I wrote you last year I have taken on board another hard drive which was wired up for me at our AGM in Derby May 10 by our group technician Ross Bennett. Modems were distributed by our chairman Trevor Stevens, in the hope that we could at least get on line to our bulletin board, which now runs 24 hours every day, but sadly I have not been able to get mine to work, but we have a workshop coming up in the autumn at Sandbach, which is only a few miles from where I live, so the user group chaps should be able to sort something out for me.

David H. Caine
Crewe, Cheshire, England

Thanks for award

This is an open letter of thanks to those whose confidence in my abilities led to my receiving the 1997 Jim Peterson Award for Software. I'm afraid I was a bit flustered during the presentation at Lima, and failed to say thanks to the many whose votes totaled up for the award. Special thanks also go to Jim Krych, whose efforts led to the establishment and continuation of the Peterson Awards, and to Jim Peterson's family for their endorsement of the awards.

During the last years of his life, Jim Peterson was both a friend and an inspiration to me. Jim and I would talk for hours on the phone, discussing every topic under the sun. We had similar views on many topics, including Windows for the PC, which we both detested. Many times during our phone conversations Jim would issue challenges to my assembly skills, as, for example, how could excess capacity in the program memory be used to store strings for Extended BASIC programs. This led to my public domain product called STOR-MOR, which allowed Jim and others to stash string variables in other than VDP space, and made possible some XB programs that otherwise would have run out of string memory (a.k.a. STACK SPACE). Others of my public domain products flowed in similar fashion from Jim's inspiration.

The Jim Peterson Achievement Award now hangs in a special place of honor in my computer room, as Jim Peterson's memory has a special place in my heart.

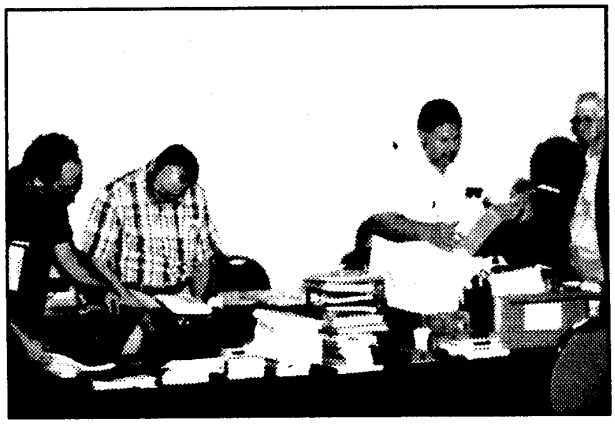
Bruce Harrison
Hyattsville, Maryland

Send your letters and comments to *MICROpendium Feedback*, P.O. Box 1343, Round Rock, TX 78680.

Photos from Fest West

photos by Lou Brody

Kyle Crichton of Competition Computer shows his CD-ROM and other products.



Tlers check out the goodies at April's Fest West.

Lima MUG Conference

New products featured, planning for future discussed

By GLENN BERNASEK

This article was downloaded from the World Wide Web — Ed.

The 1997 Multi-User Group Conference at Lima, Ohio was a "bitter/sweet" experience.

Seventy-six attendees had signed in by 2 p.m. (I know there were more because I saw familiar faces who hadn't signed the registry.) Our numbers are dropping slightly, but the TI99/4A and Geneve 9640 enthusiasm certainly isn't.

The first thing (and somewhat confusing) that greeted the people who came was a "give away" table set up by the Lima User Group. It turns out that they were cleaning house and everything had to go either to TIers or the dumpster. Charlie Good felt bad about the dumpster option, and he said he hoped everything would go to the TIers who attended. Needless to say, this put a "slight" crimp in the plans of attending TIers who were hoping to sell some of their equipment at the conference.

John Parken, membership chairman for the TI-Chips, brought a Canon BJC-620 color jet printer to be raffled off by the TI-Chips. The raffle was great success. We didn't make any money, but it didn't cost us very much either. (Darned near broke even, and everyone had a ball!)

I was able to receive consultation from Mike Wright of CaDD Electronics on the ins and outs of running the file transfer routines of PC99. Mike also stated that PC99, Stage IV will be available for an upgrade cost of $9.94. This is most definitely *not* vapor ware. We've had a fantastic amount of new developments for the TI99/4A and Geneve systems in the past year. CaDD Electronics' PC99 is just another gem in this development. Mike also said that he was thinking of writing a version of PC99 for the Super AMS card.

Bruce Harrison showed his software written for the Super AMS. Bruce demoed his "Slide Show" and "Video Titler." Both routines took advantage of the memory pages in Super AMS that allowed simulated animation and very fast screen title presentations. Bruce said he is interested in working on a Super AMS version of "Midi Master."

> Dave Szippl said the Internet would be a possible mechanism for keeping the TI community together. He said we could set up a world-wide Internet user group and be able to keep on communicating and sharing.

PLANNING FOR THE FUTURE

The Multi-User group officers' conference covered three very important points. First, Charlie Good said he is trying to get two students to take on offices in the Lima User Group for 1998. If he is successful, Lima will host the 1998 MUG. Charles stated that 1998 would probably be the last MUG the Lima group would host. I said that the TI-Chips would commit for 1999 if Lima confirmed, by this September, that they would host the 1998 show. At this time, Charlie made clear just what was happening at this MUG. The Lima group found it necessary to remove their equipment from campus storage and had also decided to close out their treasury. This, in effect, will put the Lima User Group on an informal organizational status. Charlie said he will continue to swap software and publish the "Bits 'n Bytes" newsletter when he is able.

The second point was brought up for discussion by Dave Szippl. Dave said the Internet would be a possible mechanism for keeping the TI community together. He said we could set up a world-wide Internet user group and be able to keep on communicating and sharing. The officers in attendance said the time has come to give this idea a chance. The technology is there. All we have to do is to log on and start talking.

Lastly, the idea of opening user groups up to other computer system platforms on an formal/informal basis seems to be gaining momentum. One group said they have been having "mixed" meetings with both the TI home systems and MS-DOS clones. They found that this was an excellent way to keep the user group viable and maintain friendships gained through the years. When asked if they had changed their by-laws to accommodate the new agenda, the answer was "no." The meetings continue as usual, but room is now made for MS-DOS discussion or demos. Another representative said their Faire is open to anyone who is willing to pay the table registration fee. (I distinctly got the impression that this change has been quietly occurring throughout the community.)

IMPRESSIVE DEMONSTRATIONS

The few demonstrations I was able to attend were impressive to say the least!

Andy Freuh put on a super, two-part presentation on how to use the Internet and how to set up an e-mail account. Andy uses Microsoft's Explorer, and used Rich Polivka's Web page as an example of what the Net has to offer. Andy handed out copies of a very comprehensive guide to the workings of the Internet. Super job ,Andy! One of the high points in Andy's demonstration was when John Bull posted a message from those at the show on Tom Wills' List Server. John signed the posting with the names of those at the demo. (It

(See Page7)

MUG —

(Continued from Page 6)

was great to see this posting when I got home that night.)

Ron Markus (Ramcharged Computers) showed the Pro-Stick joystick and game software he has available, while Jim Krych demoed the Video Turtle. The most amazing part of this demo was the clarity of the images on a "standard" S-Video TV screen. Detailed color pictures and 80 columns with *no* fuzziness or distortion. (To paraphrase a line form a well-known Christmas essay, "Yes, TIers, there is a working replacement for those hard-to-find RGB monitors. It's called the Video Turtle.")

PETERSON AWARD WINNERS

The winners of the 1997 Jim Peterson Achievement Awards follow:

TI Community Tom Wills
TI Software Bruce Harrison
TI Hardware SW99ers (Super AMS)
Geneve 9640 Tim Tesch

Nominations for the 1998 awards are now open.

Our gracious hosts, the Lima User Group, closed out the MUG in grand fashion with a catered, after the conference dinner which included stuffed chicken, potatoes, green beans, salad and dinner rolls. This was a gift from the Lima User Group Thank you, Lima! The TI99/4A and Geneve community will forever be in your debt. Thanks for the ride. It was wonderful!

Seminar tapes are available from Lima group

Videotapes of most of the seminars given at the May 23-24 Multi-Users Group are available for $10, according to Charles Good of the Lima 99/4A Users Group, the MUG's host. The two tapes cover about 10 hours of seminars, and the charge pays for the media and postage within the United States. Send requests to Good at P.O. Box 647, Venedocia, OH 45894.

A little gloomy, perhaps, but MUG was a success

By ANDY FRUEH

This article is from Bits, Bytes & Pixels, *the newsletter of the Lima, Ohio, 99/4A Users Group.*

On behalf of the Southwest 99ers User Group, Matt Matthews (right) accepts Jim Peterson award from Jim Krych.

Well, I had a *terrific* time. Keep in mind that this was my first TI fair of any kind since I left the community in 1992. There wasn't a huge crowd (the current estimate stands at around 70 to 75, I believe) but all of the "regulars" were there (RAMCharged Computers, Bud Mills, Bruce Harrison and many others).

That's part of what struck me as odd. When I went to the MUG Conference five years ago, the ration of vendors to buyers was small. There were many more "regular" people than vendors. This time out, it seemed that nearly everyone who attended also had a table. What also disturbed me was the apparent absence of families. I guess families have all moved on to Nintendo 64s and Macs or PCs?

Another gloomy aspect of this show (hey, let's get the gloom and doom out of the way *first!*) was the lack of anticipation. At previous shows, there was always the eager anticipation of the new hardware and software that would be released and/or demoed at the show. Not so this time. The feeling I got from talking to various people was that, after the AMS, every significant product that has ever been made for the TI has *been* made. Has the 4A's number been called?

I doubt it. There may never again be a new software product as significant as Funnelweb or hardware like the Geneve, TIM or AMS cards. But the *community itself* seems to be as tight and close as ever. Probably even *more* so with the advent of e-mail. As our community slowly shrinks, what we have left are the diehards. Those who either absolutely don't want to, or can't, get rid of the 4As. What surprised me was the large number of people who knew or recognized me.

FREE STUFF

One of the neatest things to me was the large quantity of free or almost free stuff available. I grabbed three consoles and had to pay for one of them. I grabbed a P-code card for $5! I also took a Disk Manager 2 cartridge (don't know exactly why, yet, though) and a Console Writer cart! I forgot to pick up a Navarone Widgit, though — can anyone sell me one? Oh, yeah, and to anyone who's offered me the TI-Writer and other manuals, I've got them now (free!). I've even picked up a TI-Forth Manual! (Gasp!) Now that I'm older, maybe it'll make sense to me.

UNUSUAL HARDWARE

There were three products for sale that really grabbed my attention. The HSGPL,

(See Page 8)

MUG —

(Continued from Page 7)

while I have no use for it and didn't understand it to a great extent, was one. The SCSI card at Bud Mills' table I also found exciting. I saw a P-Box with a Syquest drive in it! For those who are unfamiliar with them, Syquest cartridges are a lot like Zip cartridges for Zip drives. They are removable media that can store a *whole* lot more than 1.4 megs on a PC. This particular system was using the cartridge as a hard drive (that's how the 4A "saw" the drive — as a hard drive).

There was also the Video Turtle. I didn't go to many seminars, but I wanted to check this one out for you. My opinion? It's a high-quality device that does *exactly* what they said it would do. It's possible to send RGB video out to an S-video monitor. And it looks *wonderful*. I sometimes have trouble reading 80-column text on a monochrome monitor. And I'm 23 with good eyes! But I could actually read it on a 19- or 20-inch TV with *no problem*. And I was at least 6 to 8 feet away from the screen. Graphics were clear and sharp. The device is simple to use and the manual is clear enough for anyone to figure it out. It's literally as easy as plugging in a toaster.

My only complaint? The cost. The gadget is $129 and you'll need an S-video television. I assume you could use a VCR with S-video, but if you send that to a non-S-video TV, the picture won't improve at all (you're still seeing a regular television signal if you do that). While they announced that Tex-Comp, distributor of the Turtle, had 19-inch S-video TVs for something like $179 or so, that's still $300 worth of equipment. Now, put that in perspective. You can *not* buy a good (or even barely decent) 19-inch or even 17-inch monitor for $300. Heck, even 15-inch monitors usually go for just over that! And the Turtle/TV combo works with any device that sends out RGB signals, including Amigas, Sega Genesis, Apple IIgs and other machines. I think the product will be a success, but I'm not sure how many 99ers will contribute to that success.

My session on Internet resources available for the TI99/4A went pretty well. All of the Internet books I had brought were gone by the end of the seminar. If you need a copy, please let me know! If you can use Microsoft Word 6 format, I'll e-mail it to you in that format. Since I'm pretty much done with the book, you can reproduce it in whole or in part. Send me a donation for my trouble if you really want to, but I'm not formally asking for one. And yes, you can post the Word file on the Internet.

I received a very good response to both a survey and questions about an on-line user group. Fifteen or so wanted to join such a group!

I received a very good response to both a survey and questions about an on-line user group. Fifteen or so wanted to join such a group! So let's move on with this. E-mail me and let me know *when* you want to meet and *how* (irc? Brad's chat page?) and we'll get something started ASAP.

So, in summary, I think the MUG conference was a success. A little gloomy perhaps, but a success. I can't wait until I can attend another fair!

E-mail address for the author is andyfrueh@hotmail.com — Ed.

MUG seminar speakers

Bob Carmany — "A Program That Writes Programs With Token Codes."

Tony Knerr — Software demos.

Joe Simmons — Managing a church budget using Irving Crowley's "Checkbook Manager" software.

Dan Eicher — The CC40 and TI74, hardware and software demos.

Ron Markus — The Prostick II joystick, and Texaments products.

Mike Wright — PC99 demo, with some newly added features of this 99/4A PC emulator.

Bruce Harrison — Software demos.

Don Walden — Demo of TI BAR II.

Gerd Weissman — The HSGPL card.

Andy Frueh and John Bull — Internet resources relating to the 99/4A and Geneve, with actual on line demonstrations using a windows 95 PC and hopefully also a 99/4A.

Tim Tesch — Port, and CYA for the Geneve.

David Nieters — Demo of SCSI for the 99/4A and the 99/4A AT keyboard interface.

Jim Krych — The "Video Turtle", hardware which lets the 99/4A produce 80-column quality resolution on an S-video monitor.

Bud Mills — Horizon products

MUG —

Demos of PC99 other products catch attention

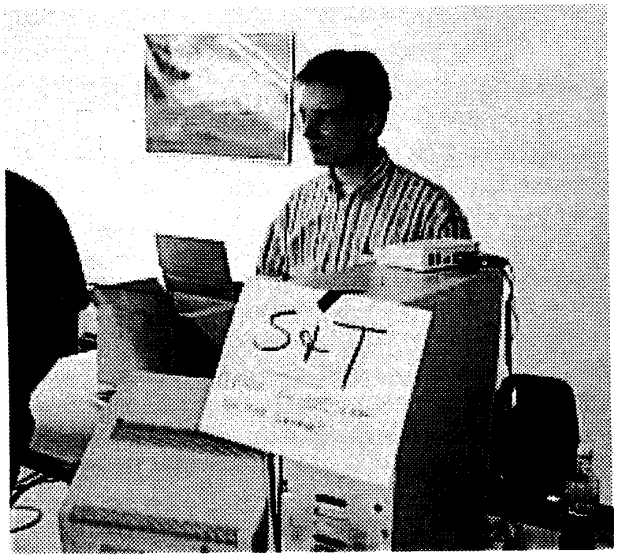
By DICK BEERY

This article comes from the June/July 1997 issue of Spirit of 99, *the newsletter of Central Ohio Ninety-Niners Inc.*

Even though the attendance was, as expected, not comparable to that of former years, those who did attend were serious and involved, and the panel of speakers and the other exhibitors, well-prepared and knowledgeable, made it seem as if no downsizing had occurred. From our C.O.N.N.I. group only Everett Wade and I attended.

We missed Bob Carmany's 8 a.m. lecture on "A Program That Writes Programs with Token Codes." We both did attend Tony Knerr's software demos, however. Labeled "Funnelweb '97," his product depends on the use of a Gramkracker or other P-GRAM device, which left, I felt, many people at a disadvantage. I know personally only a few people who ever purchased one of these devices. In an already-slim market, this restriction probably limited the number of potential buyers even further.

The same could be said for Bruce Harrison's software demos at 11:30 a.m. These latter depend upon the user's accessing the AMS card, a bank-switched memory card that replaces the 32K card in the P-box. I know even few people who have acquired, or plan to acquire, one of these. It seems only fair to note, though, that what must be involved is a sort of Catch-22. Software programmers want to, and should, support the efforts of those bringing what are truly wonderful hardware devices into our already-dwindling market. And perhaps there is the attraction of novelty. No longer just Extended BASIC or whatever. I am personally glad that they are doing so, but fear they will not meet with financial success as a result. Maybe their satisfaction is to be gained from breaking new ground and doing it well, whatever the financial result.



Tim Tesch, winner of the Jim Peterson award for the Geneve, stands beside his tower-based Geneve.

Groups with display tables

Bruce Harrison

Bud Mills Services

CaDD Electronics — sales of PC99 and TI module documentation on PC disks.

Cecure Electronics

CinDay User Group

Cleveland Area User Groups — (TI Chips and Northcoast groups).

Dan Eicher

David Connery — sales of monitors and other hardware

Hoosier User Group

Jerry Grams — used hardware.

John Bull — demo and sale of Contract Bridge

OSHTI User Group — Oshawa, Ontario Canada.

Ramcharged Computers — Asgard, Texaments, and Bodenmiller software.

S&T Software

Tony Knerr

Turtle Enterprises — RGB-to-S video adapters.

Western Horizon Technologies — sales of new SCSI cards and AT keyboard interfaces.

Neither of us witnessed the presentations by Joe Simmons, who spoke on "Managing a Church Budget" using Irvin Crowley's Checkbook Manager, nor Dan Marcus' on the Prostick II joystick and Texaments products, which sounded suspiciously like a presentation from last year's fair, nor Dan Eicher's talk on the CC40 and TI74.

PC99 UPGRADE

We both did attend, and listened closely, to Mike Wright's demonstration of the expanded PC99 (see MICROreviews column for review—Ed.), software that permits users of PCs to add a TI emulator in DOS. The major new thrust in version 4, which I purchased at the fair, is the addition of the Myarc disk controller. This allows the "disks" in PC99 (actually text files on the PC's hard drive) to be formatted double-sided, double-density, or 1,440 sectors, and also allows for use of special commands available only with that disk controller. One reason I like using such a

(See Page 10)

MUG —

(Continued from Page 9)

device is that it permits one to program in TI BASIC or Extended BASIC. While not much of a programmer, I enjoy doing it occasionally. One cannot do that comfortably with a PC without having to learn Q-BASIC or such.

Don Walden did a demo of TI Bar II, and Gerd Weissmann, one of the three visitors from Germany, demonstrated the HSGPL card. I regret having missed both those, as well as that of my friend Jim Krych on the "Video Turtle," hardware which lets the 99/4A produce 80-column quality resolution on an S video monitor. The Rev. John Bull demonstrated at his booth his program for playing Contract Bridge on the 4A. Tim Tesch demonstrated Port and CYA for the Geneve. I don't have one, so didn't attend. David Nieters demoed SCSI for the 4A and the 99/4A keyboard interface.

Everett attended that session, and says that the group was able to report real progress in making a SCSI drive available to users of the 4A, but that there are still bugs that are being addressed, and that the product is therefore not available at this time. Bud Mills also lectured — no topic was given in the program.

The second annual Jim Peterson awards went smoothly. The award for software development went to Bruce Harrison; Tim Tesch won for the 9640; Tom Wills won the community service award; and the SouthWest 99ers won the hardware award for the Super AMS card. Matt Mathews accepted the award for the group.

Kudos to Jim Krych, who, because of his deep respect and admiration for Jim Peterson, initiated the awards program last year. The plaques awarded each year are truly beautiful: the fine wood finishing and handsome brass plates inscribed with the recipients' names make these welcome rewards for fine achievements in their fields. Thanks very much, Jim!

The former officers' meeting, now open to any users group members present at the fair, featured lively discussion about e-mail and chat for 4A and Geneve users throughout the world. If a list server were to be used, writers could place their articles on it, and these would be then available to users groups for incorporation into their newsletters. Users groups represented at the meeting reported normal attendance of only 8 or 9 members per meeting, often fewer. One exception reported 15; Charlie Good's Lima group sometimes has only 2, and Charlie reported talking to himself at one recent session. It is hoped that participation on the Web will help reunite and bring together present and former TI users relishing the renewed contact with old friends. At least one person, Andy Frueh, a former Lima Group member, who had sold his TI equipment, has now acquired another system and is once again using it as well as a PC. Glenn Bernasek spoke excitedly of the "TI users entering an entirely new (e-mail and group chat) phase" in maintaining and expanding ties with other users.

The hosts provided two wonderful additions to this year's fair, probably never to be repeated: Coffee, tea, orange juice and rolls were provided free until nearly noon. Then at 6:30 p.m., those still present (about 50 people) were treated to a buffet

Very well attended, and possibly the most dramatic of all, was the presentation, in two parts, of actually being on the Web (a Pentium was used, together with a large projection screen). Immediately Following, the use of the 4A in getting and sending e-mail was demonstrated in another room. Andy Frueh conducted the Pentium session, and he and the Rev. John Bull led the 4A session. Much to think about here! One can subscribe to Brad Snyder's TI99 site (http://www.enter.net/~bsnyder/tichat/html) for access to the chat group already mentioned.

The hosts provided two wonderful additions to this year's fair, probably never to be repeated: Coffee, tea, orange juice and rolls were provided free until nearly noon. Then at 6:30 p.m., those still present (about 50 people) were treated to a buffet consisting of chicken, boiled potatoes, green beans, rolls and beverages. Greatly appreciated by all. Another fair is planned for next year, but whether it will take place in Lima or in Cleveland is not yet known.

Another feature, new this year, was that the program, and some pictures, could be accessed on the Web (http://www.bright.net/~cgood/mug1997.html), both before and following the conference. More than 20 pictures from the conference are available there. A novelty involved the "free" tables, loaded with books, cartridges, cables and even several P-boxes and a console. All free for the asking.

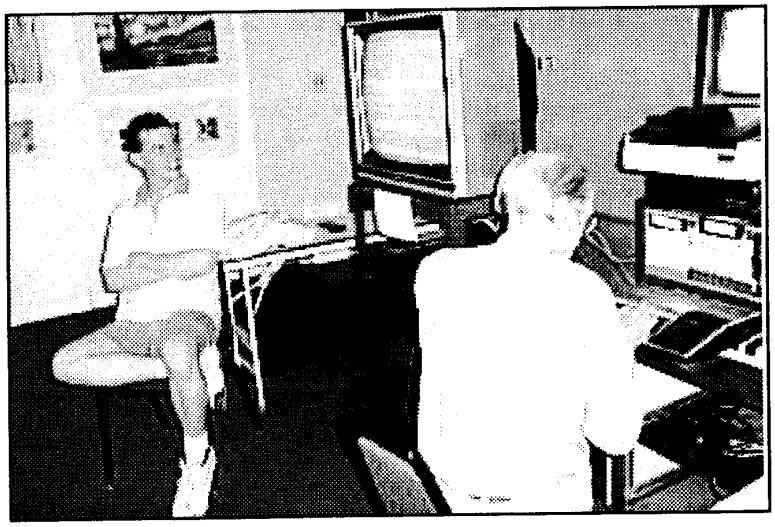
In closing, I have to say that I had not planned to attend this year's fair. I felt that it would be very depressing to see a deteriorated version of the great fairs I had attended previously. I did not decide to go until the Tuesday preceding the fair. I am *very* glad that I went. Hope to see more of you there next year.

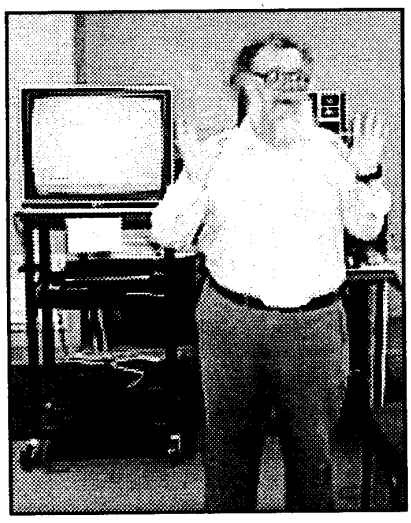
Nominations open for Peterson Award

Nominations are being taken for the 1998 Jim Peterson Achievement Awards, given in four categories: hardware, software, community and 9640. Deadline for nominations is Dec. 31.

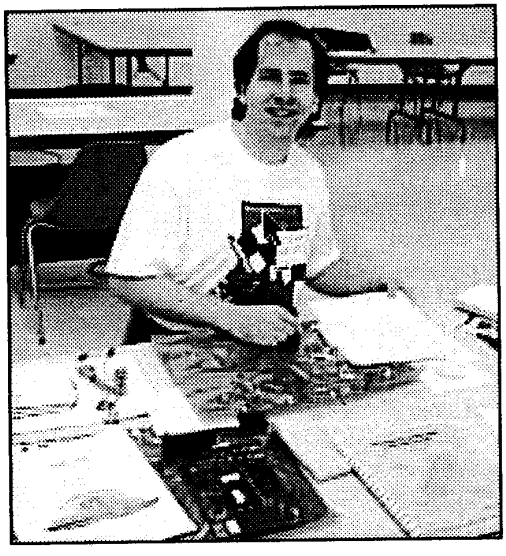
Anyone wishing to make a nomination can write Jim Krych, 3969 Clague Rd., North Olmsted, OH 44070 or e-mail ab453@cleveland.Freenet.Edu.

MUG 1997 in photos



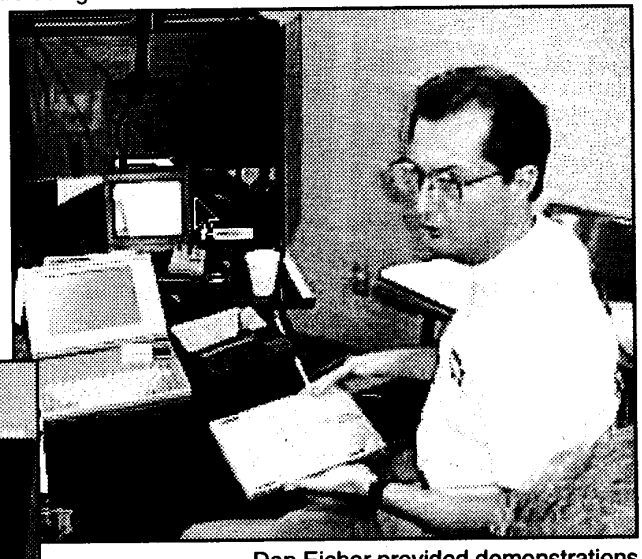


Bruce Harrison received a Jim Peterson award for contributions to TI software.



Andy Frueh and John Bull (above) gave an Internet demonstration.

David Nieters (right) gave a demonstration of the SCSI card using a TI99/4A.



Dan Eicher provided demonstrations of the CC-40 and TI-74.

Ron Markus of RAMcharged Computing displayed an array of wares for sale.

Beginning c99 — Part 5

Time for a little fun!

By VERN JENSEN

After going through some pretty boring (but important) stuff the last two issues, today we're going to do something fun. We're going to make a "screen saver" that simulates flying through space. In the process, you'll learn a number of things, such as how to add comments to your source code, how to make definitions, and how to use functions and arrays.. Who says work can't be fun?

To get started, start up the Editor and type in the program in today's sidebar. Save it as DSK2.SPACE;C, then compile, assemble, and run it. After seeing what the program does, continue reading below.

COMMENTS

Probably the first thing you'll notice about today's example is that it has lots of comments. In C, you can specify the beginning of a comment with the characters "/*" and the end of the comment with the characters "*/" (without quotes, of course). And although all the comments in today's example are only one line long, you are not limited to a single line. (Remember that the compiler ignores spaces.) The really great thing about this is that you can easily comment out half your program if you want, simply by adding the "/*" and "*/" characters to the beginning and end of the code you want to comment out.

For instance, as I was working on this program, I didn't like th way the star's initial position was calculated in NewStar(), so commented out all the if statements and wrote new code. Later, realized my new method wasn't that great, and it was easy to un comment the old code and start using it again. If I had previously deleted it rather than commenting it out, I would have had to typ it in from scratch.

There is just one problem with commenting out whole section of code like this, and that is that comments can not be nested. Fo instance, if I were to try to comment out the lines below as demon strated, it wouldn't work properly, because the compiler woul see the "end of comment" character at the end of the "/* Move lef */" comment, and assume the first comment was done:

```
/*
if (myKey == 'S')  /* Move left */
      hDelta = -5;
*/
```

Because the compiler ignores everything from the first "star comment" characters it sees to the first "end comment" character it sees, the code above would be turned into this:

```
      hDelta = -5;
*/
```

Obviously this is not what we wanted, so you have to be careful when commenting out entire sections of code to make sure that th code doesn't already have other comments within it. The only rea son I was able to comment out all the if statements in th NewStar() function while working on the program was because I hadn't added the other comments to the code yet.

DEFINITIONS

At the beginning of the program, following the #include state ments, you'll notice a series of #define statements, which allow you to assign a number to a name. When the program is compiled the names will be replaced with their corresponding numbers This means that using #defined names in your program is just as fast as using the actual numbers themselves. For instance, the statement:

```
if (a == 69)
is just as fast as
#define kUpKey 69
...
if (a == kUpKey)
```

since the second example is changed into the first during compila tion. Definitions provide a way of keeping your code much neater, since you can put all your definitions at the beginning of a pro gram, allowing you to easily change those values in the future if necessary, without having to hunt down all the places they are used in the program. For instance, in today's program, the defini tion kNumStars is used twice in the program. By using this defin ition instead of the actual number in the program, I can easily change the number of stars by changing the definition, rather than

(See Page 13)

BEGINNING c99 —

(Continued from Page 12)

having the computer hunt down all the places the number 15 is used in the program.

In case you're wondering why the definitions start with the letter k, it's just to keep them from conflicting with variable and function names. (A habit I picked up when programming on the Mac.) And, just like variable and function names, the c99 compiler only recognized the first 6 characters of a definition name. However, since definitions are replaced with their corresponding value *before* assembling, definitions *are* case sensitive — so if you #define kFColor 5 at the beginning of the program, and then use the name kfcolor later in the program, the compiler won't recognize it, and won't replace it with the number 5. This will result in an "Undefined Symbol" error when you try to assemble the program, because the assembler won't recognize the name kfcolor either, which should have been replaced with 5 by the compiler. To demonstrate this is the following program, which is a modified version of the test program we used in the first article in this series:

```
/************/
/* MYTEST;C */
/************/
#include "DSK2.GRF1;H"

#define CHR 65
#define myChr     66

main()
{
  int chr, keyCode, status;

  Grf1();
  Screen(14);

  chr=33;

  do
  {
    HChar(1,1,chr,768);
    HChar(12,1,CHR,32);
    HChar(13,1,mychr,32);
    chr++;

    if (chr >= 126)
      chr=33;

    keyCode = Key(0,&status);

  } while (status == 0);
}
```

You will notice two definitions at the beginning of the program. The first defines CHR as 65. However, this does not conflict with our variable of the same name, because the variable is in lower case. This means that if you want, you could put your definitions in upper-case in order to keep them from conflicting with variable names, rather of starting the definition name with k.

The second definition defines myChr as 66. Later in the program, mychr (lowercase) is used. If you try to compile and assemble the program the way it is written, you'll get an "Undefined Symbol" assembly error. I'd recommend doing this just for the experience, in case this actually happens to you when you're writing your own program. When you get an assembly error, load the file being assembled (in this case, MYTEST;S) in the Editor and take a look at the line number containing the error. If the file is too big to load in the Editor, you can still look at the line using the LINEHUNTER program, option 6 in Funnelweb's menu. Once you find the line, you'll see that the name MYCHR is causing the problem, and you can then search MYTEST;C for that name until you figure out which line contains the error.

It's things like this that gave me extra incentive to write this series. The c99 manual makes absolutely no mention that definitions are case sensitive, while variable and function names are not, which caused me a lot of frustration while working on Virus Attack. By sharing these and other tips with you, hopefully you can avoid the problems I encountered, and be able to spend your time writing code rather than hunting down bugs.

ARRAYS

In c99, you can have one or two dimensional arrays. Arrays can be of type char or int, and the first element of an array is 0, just like in XB (unless you call OPTION BASE 1 first). Defining an array is simple - just define a variable name as usual, placing the number of desired elements between brackets at the end of the variable name:

```
int myArray[10];
```

It is important to realize that this statement only creates 10 elements, meaning that since the first element is 0, the last element is 9. Thus, trying to access element 10 of this array will cause problems. To create a two-dimensional array, add another set of brackets, like so:

```
int myArray[10][10];
```

Note that this is not the "add a comma" method used by XB, so this statement would be wrong:

```
int myArray[10,10];
```

Accessing an array is straightforward; the following example shows how to set all elements of the array defined above to 0:

```
for (a=0; a<10; a++)
      for (b=0; b<10; b++)
            myArray[a][b] = 0;
```

In the SPACE;C example, our star array keeps track of 3 variables for each star - the x, y, and z coordinates. If you look after the array definition, you'll notice these statements:

```
#define kX 0
#define kY 1
#define kZ 2
```

These statements allow us to easily access each of the three variables for each star without having to remember which array element corresponds to the desired variable. For instance, to access the y variable of star 5, we can use the statement

```
      y = star[5][kY];
```

rather than having to remember this:

(See Page 14)

BEGINNING c99 —

(Continued from Page 13)

```
y = star[5][1];
```

The first statement is much clearer, since at a glance you can tell that the y variable of star 5 is being read, while the second statement is unclear. This also makes debugging much easier, since at a glance you can tell if you accidentally accessed the wrong element of an array.

FUNCTIONS

In today's program, you'll notice that very little actually happens in the main() function. Instead, it calls other functions that do most of the work. Functions are very similar to XB's subroutines. In XB, all you have to do to call a subroutine is precede the subroutine name with the word CALL. In c99, it's just as easy — all you have to do is name the function and put parentheses after the name. (Even if it has no parameters.) In fact, the main() function does nothing but call other functions - first functions to prepare for the animation, and then the MainLoop() function to run it.

Creating your own function is just about as easy as calling one. In fact, whether you know it or not, you've created a function each time you've typed in a c99 program, because main() is a function. The only difference between main() and the other functions in a program is that main() is automatically called when your program is first started.

Take just a minute to look at the SetUp() and MainLoop() functions. They're about the same as the main() function, as they have no parameters. I could have put the code for these functions inside the main() function, but chose to split it up into separate functions to make the code even easier to read. As you can see, all that needs to be done to create a function is to name it, follow it with parentheses containing the names of your parameters (if any), and then braces containing the code that makes up the function. Thus, to make an "empty" function that does nothing, you would write this:

```
Empty()
{
}
```

Adding parameters is easy as well. For an example of a function that has parameters, take a look at the GetRnd() function at the end of the program. Here you see that to add parameters, all one must do is place the names of the parameters between the parentheses, separated by commas. You also must tell the compiler what type of variables the parameters are (char or int) with a separate statement (or statements) following the function name. It is possible to have parameters of different types; for instance, one int and one char. This would be done like so:

```
MyFunc(key,number)
int key;
char number;
{
}
```

One important point to remember is that functions, unlike XB's subroutines, are given copies of the values passed to them as parameters. This means that if you pass variables as parameters to a function and the function changes its copy of those variables, the originals will remain unchanged. The same thing is possible in XB by placing parentheses around your variable names when passing them to a subroutine (see page 181 in your XB manual).

This may bring you to wonder how functions can return a value to the caller. There are two solutions: using pointers, and the return statement. We will cover pointers next issue, but the return statement is simple enough we can cover it now.

Normally, all the statements in a function are executed, and then control returns to the line after the calling statement. However, if the return statement is encountered, the function is exited early, just like SUBEXIT causes an early exit from an XB subroutine. However, c99's return statement has an advantage - it can optionally return a value to the caller. For instance, the following function squares a number, and returns the result:

```
Square(myNum)
int myNum;
{
      return myNum * myNum;
}
      We would call this function like this:
n = Square(n);
```

This would call the Square() function, passing the variable n as the parameter. The function would receive a copy of this variable, multiply it by itself, and return the result, which would then be assigned to the variable n. When calling a function that returns a value, you can optionally discard the value if you don't need it. For instance, we could call the function above with the statement "Square(n);", although that would be pretty pointless, since the whole purpose of the function is to return a number.

As I mentioned before, a function can not change a variable that is passed to it by modifying its own copy of the variable. To illustrate this, let us consider a different version of the Square function:

```
Square(myNum)
int myNum;
{
      myNum = myNum * myNum;
}
```

When this function is called, it receives a copy of the value passed to it and squares it, assigning the result to myNum. However, this is completely pointless, since changing myNum doesn't affect the original variable that was passed to the function.

In addition to returning the value of a variable, a function can also return a fixed number. For instance, this function returns 1 (true) if the requested character can be found on the screen, or 0 (false) if it can not:

```
FindChar(myChar)
char myChar;
{
      int r, c;

      for (r=1; r <= 24; r++)
            for (c=1; c <= 32; c++)
                  if (GChar(r,c) == myChar)
                        return 1;       /*
```

(See Page 15)

BEGINNING c99 —

(Continued from Page 14)

```
Found a match */

      return 0;    /* Loop ended - no match
found */
}
```

This example may be slightly confusing because of the call to GChar(), which is a function in the GRF1 library that returns the value of the character at the specified row and column of the screen. In this case, rather than assigning the returned value to a variable, we place the call to the GChar() function itself in the if statement, which compares the value returned by the GChar() function with myChar. This is a technique that may look quite foreign to XB programmers, but is used quite often by C programmers. Again, the fact that you can actually put a function call inside an if statement is yet another example of how flexible C is.

Something you should keep in mind when writing code that calls functions is that a single C statement for calling a function has to be translated into many assembly language statements when the program is compiled. The more parameters a function has, the more assembly language statements there will be. Because of this, writing code with many function calls can add considerably to the size of a program. In addition, you should avoid calling functions as much as possible when in a time-critical loop where speed is important.

That's why the MainLoop() function in today's example does all that is needed for the animation in one function — I didn't break that code up into several smaller functions. (Except for creating a new star, which only happens once a star reaches the edge of the screen.) However, I did take the liberty to make a separate SetUp() function, rather than placing that code in the main() function, since it makes the code clearer, and it isn't in a time-critical loop. Also, SetUp() doesn't have any parameters and is only called once, so doing this doesn't add much to the length of the program.

GLOBAL AND LOCAL VARIABLES

The C language has so much in common with Extended Basic that sometimes I wonder if parts of Extended Basic were modeled after C. The next lesson is one of those instances. As you may know, variables that are used in a subroutine in XB are "local" to that subroutine, meaning they can't be "seen" by code outside that subroutine. In fact, you can have variables in your subroutine that have the same name as variables used in the main part of your program, and they won't conflict.

It's just the same in C. Variables that are declared within a function are called "local" variables, since they are local to that function only, and can't be used by any other function. Variables that are declared outside of a function are called "global" variables, because any function can use them.

In today's program, we have many local variables and only two global variables. The global variables are created before the main function with this statement:

```
int   n, star[kNumStars][3];
```

This means that any function can access the n variable or the star array. However, variables that are declared within a function can only be used by that function. For instance, the variables x, y, and z which are declared in the MainLoop() function can only be used by that function. Local variables must be declared at the very beginning of a function (before any statements), and global variables should be declared at the beginning of the program (before any functions).

In standard C, the position of your global variables can affect their scope, so placing the declaration of the global variables n and star[][] after the MainLoop() function but before the NewStar() function would make those variables accessible only to the NewStar() function and those after it; they would be "hidden" from all the functions in front of the definition. I doubt c99 supports this, and there generally isn't a need for it anyway, so you should just stick to declaring your global variables at the beginning of the program unless you have tested the other method and know that it works.

Both global and local variables have a random value at the start of the program or function. However, you can assign an initial value to global variables. For instance, you could put this statement at the beginning of your program:

```
int   numLives=3, level=1;
```

This would start the player off with 3 lives on level 1. However, you must remember to reset these values if the player starts the game over again. You can not assign a value to a local variable when declaring it as you can to a global variable. The reason for this is that global variables are permanent, so they can be given a starting value, but local variables are temporary.

Local variables are called "auto" variables because they are created when a function is entered and disposed when the function is done. This means that you can not assign a value to a local variable and expect it to retain that value when the function exits and is then called again. For this reason, you should use global variables when you need a variable to retain its value even after a function exits, or if you need to share a variable between several functions. However, it's generally best to use local variables for "temporary" functions, such as cycling through a loop. Consider the following program:

```
int n;
main()
{
      Grf1();
      for (n=2; n<=16; n++)
            TextColor(n);
}

TextColor(color)
char color;
{
      for (n=4; n<16; n++)
            Color(n,color,1);
}
```

Here, the code in main() attempts to cycle through all colors, calling a separate function to change the text ascii characters to that color. However, since the same global variable is used in both

(See Page 16)

BEGINNING c99 —

(Continued from Page 15)

loops, the first loop won't function properly, since the value of n will change when TextColor() is called. This is one good reason why loops and other things which can use local variables should use them. And although the problem above is pretty obvious, in a real program, it often isn't, so using local variables when possible can help you avoid a nightmare of problems.

I mentioned above that local variables in one function can't be seen by other functions. While this is true, there is either a bug, or undocumented limitation in the c99 compiler that appears if one function calls another that uses a variable of the same name. In normal C, this shouldn't be a problem, but in c99 it is. In this case, the variable of the same name seems to be "shared" by the two functions, so changing its value in one will change its value in the other. Consider again our program above, this time modified to use local variables:

```
main()
{
      int n;
      Grf1();
      for (n=2; n<=16; n++)
            TextColor(n);
}

TextColor(color)
char color;
{
      int n;
      for (n=4; n<16; n++)
            Color(n,color,1);
}
```

While this would work in normal C just fine, it doesn't in c99. The solution is to use a different name in the second function, such as c instead of n, so the two variables don't conflict. In addition to keeping local variable names from conflicting with each other, you also need to keep your global variable names from conflicting with local variable names.

One technique commonly used nowadays it to place the letter g at the beginning of global variable names. However, on the TI, this uses up one of your six letters, making it a less desirable technique. A technique I prefer is to make sure my local variable names don't conflict with the global names, which I often do by putting the letters "my" at the beginning of the local variable names, such as "myCol" or "myRow", which would keep them from conflicting with global variables with the names "col" and "row".

ABOUT THE ACTUAL PROGRAM ITSELF

I've spent most of this article talking about basics of c99, as I should. However, a few of you may be wondering how the program works. Most of it is pretty self-explanatory. The stars are given a destination somewhere outside the boundaries of the monitor, and a depth; the greater this depth, the closer the stars appear to the center of the screen. Each frame, the depth is decremented, bringing each star closer to its destination. When the depth is equal to 1, the star has reached its destination outside of the screen.

What makes the code in MainLoop() a little harder to understand is the fact that I needed to be able to decrease the depth of each star (the kZ element of the star array) by .1 per frame, which is impossible in c99 because the variables can only contain whole numbers. The solution to overcoming this was to multiply the x, y, and z variables by 10 when the star is first created, and then to divide them by 10 when actually calculating the star's position on the screen. I could then decrement the depth variable by 1 each frame, which would effectively lower the actual depth by .1.

As many of you know, the screen has 256 visible pixels horizontally and 192 visible pixels vertically. This means that 128 is the middle pixel horizontally, and 96 is the middle pixel vertically. Therefore, when you see numbers like 960 or 1280 being used in the program, keep in mind this is just 96 or 128 being multiplied by 10.

One other interesting thing about this program is the following statement in the MainLoop() function:

```
star[n][kZ] = z = z - kSpeed;
```

You're probably wondering why it has two equal signs. Well, the fact of the matter is that c99 is very flexible, so you can put assignments anywhere you'd like, in as many places as you'd like. The key to remember is that statements such as these (unlike if statements) are processed from right to left, meaning that first kSpeed is subtracted from z, then this result is assigned to z, and then z is assigned to star[n][kZ]. Generally code like this is harder to read than it would be if broken up into separate statements, although it does have its place. The most common example would be when setting a number of variables to the same value:

```
a = b = c = d = 5;
```

One final thing I might mention about this program is that this time I called it SPACE/C rather than SPACE;C. To tell you the truth, using a slash is the standard naming convention for c99 filenames; a semicolon is the standard for assembly language programs. However, I found it easier to hit the semicolon key when typing a filename quickly, which is why I've used it up until now. Naturally, you can use whatever you like best, although you should be consistent.

IN CONCLUSION

Once again, we have a pretty long column. Hopefully it's been helpful to you on your journey from BASIC to c99, and should help keep you busy until the next issue. You now know enough to start making your own programs, which I suggest doing, as experimenting with the language is the best way to learn. Have fun!

SPACE/L

```
DSK2.SPACE/O
DSK2.CSUP
DSK2.GRF1
```

SPACE/C

```
/*******************************/
/* SPACE/C by Vern Jensen      */
/* Appeared in the July/August */
```

(See Page 17)

BEGINNING c99 —

(Continued from Page 16)

```
/* 1997 issue of MICROpendium. */
/*******************************/

#include "DSK2.GRF1;H"
#include "DSK2.RANDOM;C"

#define kNumStars  15  /* Can be any number from 1 to 32.
*/
#define kSpeed     1   /* Speed of the stars. (Higher is
faster.) */
#define kMaxDepth  10  /* Max starting distance of star
in space. */
#define kMinDepth  6   /* Min starting distance of star
in space. */

int  n, star[kNumStars][3];

#define kX 0  /* Meaningful names for the second */
#define kY 1  /* element of our star array.      */
#define kZ 2

main()
{
  Grf1();
  Screen(2);
  Randomize();

  SetUp();
  MainLoop();
}


SetUp()
{
  for (n=0; n < kNumStars; n++)
  {
      /* Create the star sprite. */
    Sprite(n,46,16,200,0);

      /* Set its starting position. */
    NewStar();
  }
}


MainLoop()
{
  int x,y,z;

  while (1)  /* Loop indefinately */
  {
    for (n=0; n < kNumStars; n++)
    {
        /* Convert 3D coordinates into 2D coordinates */
      x = star[n][kX];
      y = star[n][kY];
      z = star[n][kZ];

      if (y>960)  /* 960 = middle of screen vertically */
        y = ( (y-960)/z ) + 96;
      else
        y = 96 - (960-y)/z;

      if (x>1280) /* 1280 = middle of screen horizontal-
ly */
        x = ( (x-1280)/z ) + 128;
      else
        x = 128 - (1280-x)/z;


       /* Move the star closer to the viewer */
      star[n][kZ] = z = z - kSpeed;

       /* Reset star's position if it has moved off the
screen */
      if (x < 0 || x > 255 ||
          y < 0 || y > 191 || z <= 0)
      {
        NewStar();
        continue;  /* <--- Don't draw this star without
first          */
      }            /* converting its coordinates for the
new position. */


      SpLoct(n,y,x);  /* Move the sprite */
    }
  }
}


NewStar()
{
  int row, col, myNum;

  myNum = Rnd(4);


  if (myNum == 0)        /* Top */
  {
    row = GetRnd(-200,-50);
    col = GetRnd(-130,386);
  }
  else if (myNum == 1)   /* Bottom */
  {
    row = GetRnd(242,392);
    col = GetRnd(-130,386);
  }
  else if (myNum == 2)   /* Left */
  {
    row = GetRnd(-130,322);
    col = GetRnd(-200,-50);
  }
  else                   /* Right */
  {
    row = GetRnd(-130,322);
    col = Rnd(306,456);
  }


  star[n][kX] = col * 10;
  star[n][kY] = row * 10;
  star[n][kZ] = GetRnd(kMinDepth,kMaxDepth) * 10;
}


GetRnd(min,max)  /* Get a random number between min and
max, inclusive. */
int min,max;     /* Both positive and negative numbers can
be used.     */
{
  return Rnd(max-min)+min;
}
```

THE ART OF ASSEMBLY — PART 65

3X5=1X1

By BRUCE HARRISON

Talk about new math! The non-equation in today's subtitle is not a joke, but a magic formula that we actually used in two of our Assembly programs. The formula applies for the case of printing pictures from the TI screen (bit-map wise) on a dot matrix printer.

It all started back when we created our own drawing program. We wanted to have the printing function keep the proportions (or aspect ratio) the same on paper as it appears on the screen. Thus the image area had to be in exactly the same ratio height-to-width as the screen image area. Doing this means that circles on the screen are circles on the paper, and squares are squares, etc.

A COMMON DENOMINATOR

All of our dot matrix printers are of the 9-pin variety, so keep in mind that this discussion refers to those, and not the more modern 24-pin type. The printhead on such printers has nine pins arranged vertically, spaced 1/72nd of an inch apart. In their normal "graphics" mode, these printers make dots at 1/60th of an inch spacing across the paper. What was needed was some way to make the horizontal and vertical dimensions of each bit in the screen image as a square area on the paper. Of course this had to be done in a way to allow 192 by 256 pixels to fit on the 8 1/2 by 11 inch paper, too. The answer was to use the printer's double density graphics mode, in which it puts the dots at 120 per inch across.

THE SOLUTION

For each bit in the image, we fire three pins at the top of the printhead, then replicate that five times horizontally. Thus one pixel from the screen is 3/72nds in height and 5/120ths wide. Now get out your calculator and divide 3 by 72. Write that down, then divide 5 by 120. Eureka! Since these fractions are identical in value, our pixel will appear square on the paper. Now of course we have to see if that will fit on the paper. If each pixel in the height of the screen (width of the paper) occupies 5/120ths, and there are 192 such, that equals exactly 8 inches, so the height dimension is okay. Now along the length of the sheet, we need 256 pixels. Thus we get out the calculator again and multiply 3/72nds by 256. That comes out to 10.66666... (10 2/3) inches, so we're safe on the 11-inch paper length as well.

WHY DO THIS?

In case anybody missed it, we don't own TI-Artist. What we were seeking in our most recent experiment was a way to print TI-Artist picture files without having that program. We had another incentive to do this. Our friend Charles Kirkwood Jr. wrote complaining that he couldn't get his own TI-Artist pictures to print correctly. He sent along some examples, so we started experimenting. Our own drawing program can print its own pictures, but it can't handle all of the picture area of the TI-Artist pictures. (It leaves off 24 dot-rows.) What we wanted to make for our own use and for our friend was a very simple program that would both show the pictures on screen and send them to the printer.

BORROWED SOURCE CODE

As we often do, we borrowed from ourselves. We took some source code from our "picture show"program, to catalog a disk and produce a listing of the TI-Artist picture files that are on the selected disk. Now when the user selects one from that list, the program takes that picture from that disk and displays it on the screen.

To complete the program, then, we took the section of our drawing program that sends the pixels to the printer, modified it to use the entire TI-Artist picture, and added that to the selection source code from our picture show. We added one small thing in the beginning, so that when the program starts up, it will prompt for the printer's file name, with a default entry of PIO.CR. This way, most users will just have to press enter at the prompt, but for those whose printer is connected via RS-232, there's a way to "fix" that for the duration of the run. Next there's a prompt for a drive number, and that will accept a single keystroke in the range 1-9 or A-Z. Given that, the program catalogs the disk in the designated drive. If there's no such drive, or no disk in that drive, an error gets reported. If the catalog contains no TI-Artist Picture files, that will be reported. Given a disk that contains one or more picture files, the screen will show a list of the picture files on that disk, with a selection marker. The user then moves that marker to any file name and presses Enter.

The program accesses the selected drive and puts the selected picture on screen. The program now waits for a keystroke from the user. Pressing P or p on the keyboard will start the printing process, any other key will simply put back the list of files, with the marker where it was. We figured that P for Print was an easy enough mnemonic connection for most anybody. During printing the picture just remains on the screen. When printing finishes, the list comes back.

A RAVE REVIEW

When we sent an advance copy of this new product to Charles Kirkwood, we got a very enthusiastic response, and some examples of how TI-Artist printed his pictures. The pictures in question were all on a "railroad" theme, including a very nice rendition of an old-fashioned steam engine. Steam engines have large wheels on them, and those are supposed to be round! Charles made them so they are nicely round on the screen, but on the TI-Artist printouts they came out as ovals. On printouts made with our new program, the wheels are round, as they should be.

We have in our collection a couple of Ken Gilliland's "Disk Of..." products. These contain very nice TI-Artist pictures, including even a self-portrait of Ken. We tried printing these with our new program, and the results were just what we'd hoped for, in that the printed image nearly fills the sheet of paper, and looks exactly like the screen image.

BUT THEN AGAIN ...

Our little program is not a panacea for the TI-Artist users of the world. We have a collection of very colorful pictures from the Lima library, but none of them will print out as anything but "gar-

(See Page 19)

ART OF ASSEMBLY —

(Continued from Page 18)

bage" in black and white. Seems that the folks who created these took advantage of being able to manipulate the foreground and background colors to make their pictures very pretty on the screen, but this means that the pattern part is not a black-and-white rendering of the image content. Thus we recommend that our new product should only be used with the black and white type of picture, of the sort found in Notung's "Disk of..." series.

The other question that plagues users of dot matrix printers is of course the business of control codes. We used very simple control codes that are common to most models of the Epson and Star Micronics brands, and also most 9-pin Panasonic printers. Thus the program works correctly on our Star NX-1000, on our Gemini-10X, and on Charles' Epson RX-80. It will probably work just as well on the Epson FX series, and on many Panasonic models.

We have as usual released the program as public domain software, on a SS/SD disk called TIAPRINT. That's available through the Lima user group. The disk contains instructions, an Extended BASIC program to print the instructions, an XB LOAD program to run the Assembly program from XB, and two sample pictures kindly donated by Charles Kirkwood Jr. We hope you'll all get some pleasure from this.

TODAY'S SIDEBAR

In the sidebar is the "printing" part of the source code, with its subroutines and data sections, so you who are interested can follow just how we did this 3X5=1X1 trick. If you're using a 24-pin printer, you'll probably see that the same control sequences we've used are available on your printer, but that they perform differently. On the 9-pin models, for example, the ESC A n sets up for an n/72nds line feed, and when followed by >A, the paper advances by 3 72nds if n was 3.

For those with 24-pin printers, though, the ESC "A" sequence would result in a line feed measured in 60ths of an inch, not 72nds. Fortunately, our friend Harley Ryan sent us an extra copy of the manual for his Panasonic KX-P2123 24 pin model. It turns out that on 24-pin printers you can use ESC "+" to set line feed amounts in 360ths of an inch. A little simple math shows that 360 is exactly 5 times 72. Thus by multiplying our 3 by 5, we were able to create a version of the program that makes line feeds of 15/360ths, which reduces to 3/72nds, thus keeping our "square pixel" idea alive for the 24-pin printer. We sent copies of this new version out to Harley, and his test showed that it worked exactly as planned. It's also been tested on a 24-pin Epson by Earl Raguse, and worked perfectly there, too.

Thus there are new versions of our drawing program (DRAW24) and our TI-Artist printing program (TIAPRN24) which have been released through the Lima User Group for use by anyone having the 24-pin problem.

Next month, we'll discuss the addition of circles to our drawing program, by use of another Bresenham algorithm.

SIDEBAR65

```
* SIDEBAR 65
* PRINTING PORTION - JUST A FRAGMENT
* (NOT ALL SUBROUTINES ARE SHOWN)
* CODE BY Bruce Harrison
* PRINTER OUTPUT
* 15 JUN 1995
PRNBUF EQU  >3800          BUFFER IN VDP RAM
PRNOUT LI   R1,PPABDT      PRINTER PAB DATA
       BL   @FILOP         OPEN THE FILE
       JNE  PRNSET         IF SUCCESS, JUMP
       BL   @CLOSE         ELSE CLOSE
       BL   @BLNK          BLANK SCREEN
       BL   @SETGM         BACK TO GRAPHICS
       BL   @CLS           CLEAR SCREEN
       BL   @UNBLNK        UNBLANK
       LI   R1,PNAMSG      PRINTER NOT AVAILABLE
       BL   @ERRRPT        REPORT ERROR
       B    @SAVE0         THEN BACK TO MAIN PROGRAM
PRNSET LI   R5,32          32 COLUMNS OF PICTURE
       LI   R12,>1707      START AT LOWER RIGHT CORNER
PRNOTL MOV  R12,R3         PUT ADDRESS IN R3
       LI   R4,24          24 ROWS
       CLR  R14            R14=0
       LI   R10,TEMSTR     TEMPORARY STRING
PRNMIL LI   R6,8           8 BYTES
PRNINL MOVB @PIXBUF(R3),*R10+ GET A BYTE
       JEQ  PDEC3          IF ZERO, JUMP
       INC  R14            ELSE INCREMENT R14
PDEC3  DEC  R3             BACK ONE BYTE
       DEC  R6             DONE 8?
       JNE  PRNINL         IF NOT, JUMP
       AI   R3,-248        NEXT CHAR IN COLUMN
       DEC  R4             DONE 24?
       JNE  PRNMIL         IF NOT, REPEAT
       MOV  R14,R14        CHECK R14
       JNE  PRLD8          IF NOT ZERO, JUMP
       LI   R0,PRNBUF      LOAD PRINT BUFFER
       LI   R1,SPLF        ADVANCE 24/72NDS
       BL   @DISSTR        PLACE STRING
       BL   @PRNSND        SEND TO PRINTER
       JMP  PRDEC5         THEN JUMP AHEAD
PRLD8  LI   R13,8          8 BITS/BYTE
PLRTS  LI   R4,4           FOUR GROUPS
       LI   R0,PRNBUF      POINT AT BUFFER
       LI   R1,BGRSTR      BIT GRAPHICS CONTROL STRING
       BL   @DISSTR        PLACE IN BUFFER
       BL   @PRNSND        SEND TO PRINTER
       LI   R10,TEMSTR     POINT AT BYTES FOR 1 COLUMN
PRLBUF LI   R0,PRNBUF      PRINT BUFFER
       LI   R2,48          GROUP BY 48
PRMV1  MOVB *R10+,R1       GET A BYTE
       ANDI R1,>8000       MASK ONLY MSB
       SRA  R1,2           REPLICATE IN THREE MSBS
       BLWP @VSBW          WRITE THAT
       INC  R0             NEXT ADDR
       BLWP @VSBW          WRITE AGAIN
       INC  R0             NEXT ADDR
       BLWP @VSBW          WRITE AGAIN
```

(See Page 20)

ART OF ASSEMBLY —

(Continued from Page 19)

```
        INC  R0             NEXT ADDR
        BLWP @VSBW          WRITE AGAIN
        INC  R0             NEXT ADDR
        BLWP @VSBW          WRITE 5TH TIME
        INC  R0             NEXT ADDR
        DEC  R2             DONE 48?
        JNE  PRMV1          IF NOT, REPEAT
        LI   R2,240         240 BYTES IN BUFFER
        BL   @PRNSND        SEND THOSE TO PRINTER
        DEC  R4             DONE 4 GROUPS?
        JNE  PRLBUF         IF NOT, ANOTHER GROUP
PRCRLF  LI   R0,PRNBUF      PRINT BUFFER
        LI   R1,CRLSTR      CR/LF (3/72NDS)
        BL   @DISSTR        PLACE IN BUFFER
        BL   @PRNSND        SEND
        LI   R10,TEMSTR     COLUMN STORED
        LI   R2,192         192 BYTES
PRNSHL  MOVB *R10,R1        GET ONE
        SLA  R1,1           SHIFT LEFT BY ONE
        MOVB R1,*R10+       PUT BACK AND INC POINTER
        DEC  R2             DONE 192?
        JNE  PRNSHL         IF NOT, REPEAT
        DEC  R13            USED ALL 8 BITS?
        JNE  PLRTS          IF NOT, REPEAT FOR NEXT BIT
PRDEC5  AI   R12,8          NEXT COLUMN
        DEC  R5             DONE 32?
        JNE  PRNOTL         IF NOT, CONTINUE
        LI   R0,PRNBUF      POINT AT PRINT BUFFER
        LI   R1,FFSTR       FORM FEED
        BL   @DISSTR        PUT IN BUFFER
        BL   @PRNSND        PICTURE FINISHED
PRNCLS  MOV  @PABLOC,R0     PAB LOCATION
        MOVB @ONE,R1        CLOSE OPCODE
        BLWP @VSBW          WRITE THAT
        BL   @FILOP3        CLOSE THE FILE
        B    @SAVE0         BACK TO MAIN PROGRAM
*
PRNSND  MOVB @DELKEY,R1     WRITE OPCODE
        MOV  @PABLOC,R0     GET PAB LOCATION
        BLWP @VSBW          WRITE ONE BYTE
        AI   R0,5           ADD FIVE
        SWPB R2             LENGTH TO LEFT BYTE
        MOVB R2,R1          PUT IN R1
        BLWP @VSBW          WRITE LENGTH
        AI   R0,4           ADD 4
        MOV  R0,@>8356      AT DSR POINTER
        BLWP @DSRLNK        USE DSR LINK
        DATA 8              WRITE TO FILE
        JNE  PRNOK          IF NOT "EQUAL", NO ERROR
        BL   @BLNK          ELSE BLANK SCREEN
        BL   @SETGM         GO TO GRAPHICS MODE
        BL   @CLS           CLEAR THE SCREEN
        BL   @UNBLNK        UNBLANK
        LI   R1,PTMSTR      PRINT TERMINATED
        LI   R0,21*32+3     ROW 22, COL 4
        LI   R15,PRNCLS     LOAD RETURN ADDR
        JMP  ERRRP0         THEN JUMP TO ERROR TRAP
PRNOK   RT                  RETURN
*
ERRRPT  MOV  R11,R15        SAVE R11 IN R15
        LI   R0,21*32+4     ROW 22, COL 5
ERRRP0  BL   @DISSTR        DISPLAY STRING
        LI   R0,23*32+4     ROW 24, COL 5
        LI   R1,PAK         "PRESS ANY KEY"
        BL   @DISSTR        DISPLAY THAT
        BLWP @KSCAN         CHECK KEYBOARD
        BL   @KEY           THEN WAIT FOR KEYPRESS
        LI   R0,21*32       ROW 22, COL 1
        LI   R4,32*3        3 ROWS
        BL   @CLRFLD        CLEAR THE ERROR MESSAGE
        B    *R15           THEN RETURN
FILOP   MOV  @8(R1),R2      LENGTH INTO R2
        AI   R2,10          10 FIXED DATA
        MOV  @>8370,R0      HIGH VDP ADDR
        S    R2,R0          SUBTRACT LENGTH
        MOV  R0,@PABLOC     SAVE PAB VDP LOCATION
FILOP2  BLWP @VMBW          WRITE TO VDP
FILOP3  AI   R0,9           ADD 9 TO R0
        MOV  R0,@PABPNT     PUT AT >8356
        BLWP @DSRLNK        USE DSR LINK
        DATA 8              FOR FILE ACCESS
        RT
*
* DATA SECTION
*
TEMSTR  BSS  256
PNAMSG  BYTE 21
        TEXT 'PRINTER NOT AVAILABLE'
PRNNAM  BYTE 17
        TEXT 'PRINTER FILE NAME'
PTMSTR  BYTE 19
        TEXT 'PRINTING TERMINATED'
FFSTR   BYTE 3,12,27,'@'
SAVBYT  BYTE 6
LODBYT  BYTE 5
HEX80   BYTE >80
PPABDT  DATA >0012,PRNBUF,>FE00,0,>0006
        TEXT 'PIO.CR
BGRSTR  BYTE 4,27,'L',192,3
CRLSTR  BYTE 5,13,27,'A',3,10
SPLF    BYTE 5,13,27,'A',24,10
*
* ALTERNATE DATA VERSIONS FOR
* 24 PIN PRINTERS
* THE FOLLOWING TWO LINES
* ARE USED IN PLACE OF THEIR
* COUNTERPARTS FOR 24 PIN PROGRAMS
* DRAW24 AND TIAPRN24
*
CRLSTR  BYTE 5,13,27,'+',15,10
SPLF    BYTE 5,13,27,'+',120,10
```

Fest West '98

100 rooms booked at Lubbock hotel, web site organized

By TOM WILLS

Things are moving along with the planning of the "first" International TI Fest West-Lubbock on Feb. 14. Arrangements have been made with one hotel for a block of 100 rooms. It is the Sheraton Hotel and they have offered a rate of $54 for a single or double room (1 or 2 persons). Each additional person is $10. We are looking at another hotel so we can be assured of enough rooms. That price with be in the same range, except it will be for up to four people.

Airlines serving Lubbock include American Eagle, ASA the Delta Connection, Continental, Southwest and United Express.

One potential fly in the ointment, so to speak, might be transportation to the TI facility. The facility is just outside the city and about five miles from the Sheraton. This means we'll need to ask those driving to Lubbock, or those renting cars, to help out by taking a couple additional passengers to the facility. I really don't think this will be a real problem, but it is best to prepare now.

At a SouthWest Ninety Niners User Group meeting in early June, it was decided to try to make this Fest West less of a buy/sell fair, not that there won't be TI vendors there, but we want this to be a real "TI Experience!" Instead of just wandering around looking at what the vendors have, we want to have things happening in the main hall throughout the day(s). Much of this is still being formulated, but we hope to make this a fair never to be forgotten.

For more information, readers can access Tom Wills' web site (http://personal.riverusers.com/~twills/).—Ed.

You don't need a hard disk for files

Running floppies with PC99

BY ROGER PRICE

As I have been using PC99 from CaDD Electronics for a while I have figured out that you do not always have to change the disk path to load a program from the TI disk if the file is on a 1.44-megabyte floppy. There are two different ways to go about this.

1. Set up your paths as:

path for dsk1 a:\one.dsk
path for dsk2 a:\two.dsk
path for dsk3 a:\three.dsk

Name the first three TI "disk" files on the 1.44-mb floppy "one," "two," and "three." I am talking about the MS-DOS file name, not the name that is the TI "disk," which is the name of the disk you had on the original TI-99/4A floppy. These disk names count only as a matter of cataloging and for programs that load using a disk name, as in the Atari games like Mrs. Pac-Man. This way any floppy that you put in the floppy drive will automatically have the correct path to the first three disks on the floppy. If you forget and have the MS-DOS name something other than "one, two, or three," then you can still load the TI "disk" programs without changing the path with this second method.

With this setup you can run PC99 nearly the same as with the TI console and floppies.

2. When you put the TI "disk" files on the 1.44-mb floppy, make sure you know what file is the first one on the disk, the second, and the third. I do not mean the order that they appear as cataloged. I mean which file is copied to the floppy disk first, second, and third. I do this by using Explorer to drag and drop them to the floppy.

Now start up the PC99 program and put a disk in with the correct path names. I call this my startup disk. When you have reached the point of selecting either BASIC, E/A or Extended BASIC to load your program, remove the startup floppy from the floppy drive and put in any other floppy with the programs you want to load.

If the program you want to load is on the first file of the disk — for example, an Extended BASIC program called "game" — then type: OLD DSK1.GAME as you would normally. The PC99 program will look for the disk as directed by the path. When it does not find this "disk" it will default to the first file on the disk and proceed to look for the named program. The important thing is to know what "disk" files are first, second, and third and to know which file the program is in.

With this setup you can run PC99 nearly the same as with the TI console and floppies. I caution you to plan the disks to be downloaded from the TI floppy to the PC "disk" files carefully. You can mess up the directories on a floppy if you do too

(See Page 22)

PC99 AND FLOPPIES —

(Continued from Page 21)
much adding or deleting of programs or files from one "disk" to another on the 1.44-mb floppy.

Keep in mind that you can have five TI "disk" files on a 1.44-mb floppy, but these methods only work with the first three TI "disk" files. If you had four drives this would work with the first four files, the same being true with five drives, and so on.

Also, if you have a program that loads everything from drive one, then this "disk" must be named as disk "one.dsk" or must be the first MS-DOS "disk" file on the 1.44-mb floppy. Otherwise, the loading program will not find the program it wants to load. It's just the same as if you had the floppy in the wrong drive on the TI99/4A.

Extended Extended BASICs

Looking for extra power from XB? There's a lot to choose from

By BOB CARMANY

Years ago, when the 99/4A was first introduced, BASIC was not only the language of choice but the only programming language. Soon afterward, Extended BASIC was developed. Extended BASIC provided some enhanced commands and the ability to use multi-statement program lines. This made for more compact and faster running programs.

Of course, it wasn't long before some innovative programmers decided to take yet another step and take Extended BASIC to another level. Suddenly, there were a myriad of enhanced XBASIC extensions to choose from. They all had one characteristic in common. They were comprised of a series of assembly language routines stored in the lower 8K of memory expansion and accessed through a series of CALL LINKs. They came in a bewildering array of names: "STAR," "XDP," "XXB," "Amerisoft," and others. They were also generally V-E-R-Y slow to load from disk into memory. But they were all worth the effort.

Most of these offerings have a common programming thread running through them. There seemed to be "universal" functions that have been used since the first of these extensions appeared. They usually have some variant of DISPLAY AT and ACCEPT AT with increased string length allowed.

There are some sound utilities (ie. HONK, BEEP, and sometimes CHIMES). Most allow the user to access BYE and NEW from a running program and turn the screen off and on (SCROFF and SCRON) as well as disable the QUIT key. They also permit the user to get a return variable to indicate whether Alpha Lock, FCTN, CTRL, SHIFT, or other key has been pressed to let the program branch if it has been.

> The vast majority of the new commands appeared in one form or another in just about all of the early XBASIC extensions. Let's look at a couple of these extensions of the XBASIC environment and see what makes them special.

Another common feature is to be able to change all of the character sets' foreground and background colors with a single instruction — usually COLORS or a variant of CALL COLOR. There are usually routines to catalog a disk and sometimes perform other functions usually associated with a disk manager program. There are frequently routines to access VDP directly. They also generally have routines to start and stop all sprite motion simultaneously. It just depends how large the XBASIC extension program is as to which of these routines are present but in most cases all of them are available.

Unfortunately, the early extensions of XBASIC are simply that — an "add-on" to the regular XBASIC cartridge. The original commands that exist in the XBASIC cartridge are still there unchanged. The added functions were available through the somewhat cumbersome use of CALL LINK and a parameter list. It took a bit of concentration at times to remember exactly the order they had to be entered and what they were.

So the vast majority of the new commands appeared in one form or another in just about all of the early XBASIC extensions. What, then, makes each of them unique? Let's look at a couple of these extensions of the XBASIC environment and see what makes them special.

STAR (Super TI Assembly Routines) was written in 1986 by Michael Riccio. It loads a total of 53 assembly language routines into memory. Most of the routines are the more or less standard ones mentioned earlier. However, there are some unique routines. STAR has some interesting graphics commands that allow the user to rotate and invert characters as well as manipulate the magnification of sprites. In addition, STAR has routines to save and recall the contents of a screen including the colors to a peripheral device. All of the routines are accessed via a CALL LINK.

XXB (or Extended Extended BASIC) was distributed by Barry Traver with the routines written by several different authors. It loads some 45 new routines into memory. Once again, the routines are ac-

(See Page 23)

EXTENDED XBASICS —

(Continued from Page 22)

cessed by CALL LINKs. This extension makes use of the RAW (Read And Write) routines to access disks at the sector level. The user can directly read or write data to a specific disk sector. There is a routine for binary/hexadecimal conversion and a few other interesting routines. It is not quite as comprehensive as STAR. Still, it is an interesting utility to have around.

Amerisoft is one of the more specific-use XBASIC extensions. It is oriented toward graphic applications. It is based on the German APESOFT graphic commands. With it, you can draw circles, rectangles, ellipses, arcs, etc. and combine them into larger pictures. Windows can be created and text inserted to create graphs of all descriptions. All of these creations can either be saved to disk and subsequently recalled or printed to an appropriate printer. It is an ancestor of the more recent XBASIC extension, The Missing Link.

Most of these programs were direct extensions of the Extended BASIC environment (ie. allowing longer ACCEPT AT lines, etc.). Some added additional sprite and graphic commands and others added routines to allow the placement of characters more easily on the screen. But, they were rudimentary extensions at best. All of these enhanced XBASIC routines simply danced around the periphery of the real issue.

A real enhanced XBASIC was to wait until a substantive re-write of the entire XBASIC environment was finally finished in 1987 with the release of Super Extended BASIC. SXB was originally released as a series of expanded commands to be used with the Miller's Graphics GRAM Kracker. Released as a series of GRAM Kracker files with enhancements by Danny Michael, it was eventually further altered By D.C. Warren and released as a cartridge. The final product incorporated a series of printing routines from Quality 99 Software to print your graphics creations on a variety of printers.

Significantly, many of the standard XBASIC commands had been altered and upgraded. SXB allowed for the moving of program segments from one place to another, deleting program segments, and copying program segments — all without disturbing the rest of the XBASIC program.

In addition, SXB added some new CALLs of its own. There were sound control (BEEP, CHIMES, and HONK), key tests to return a numeric variable if a key was depressed (ALOCK, CTRL, and FCTN) and CALL GOSUB and CALL GOTO to allow for transferring program control to a subroutine or program line through the use of a numeric variable.

In all, SXB added some 30 new CALLS and modified eight commands from the original Extended BASIC. Its release as a cartridge eliminated the process of loading all of these routines from disk into a GRAM device. SXB remained the "ultimate" XBASIC extension for a number of years.

It is important to note that access to the new XBASIC commands is now via the use of a CALL rather than a CALL LINK (parameter,parameter). The extension had now become a true enhanced XBASIC.

RICHXB

The next advance was Rich Gilbertson's RICHXB, or RXB. It started out similarly to SXB and is still in the form of a series of GRAM files that require a GRAMulator, GRAM Kracker, or other GRAM device to load. The upcoming version is 1006 currently in beta-test. Version 1005 has some interesting features not the least of which is CALL USER. This XB CALL switches control to a user-created D/V 80 batch file that can perform a number of functions. The example used loads a program, resequences it, merges a second program, saves the altered program, and runs it. It leads to some interesting possibilities!

RICHXB is really the combination of the SXB module and the Editor/Assembler module with some extras thrown in for good measure. The E/A screen can be called up from the XBASIC environment and some of the functions accessed directly from XB in some cases. There have been some additions to the basic CHAR and COLOR routines to allow all of the characters to be defined at one time. There are other interesting features that allow for the swapping of character definitions (CALL SWAPCHAR) and character colors (CALL SWAPCOLOR). The user can also duplicate characters and character colors with a single command (CALL DUPCHAR and CALL DUPCOLOR). The list goes on and on with RICHXB.

The unreleased version 1006 is rumored to have some routines for the AMS memory card produced by the Southwest 99'ers that will facilitate its use. *(RICHXB is available as a cartridge from Competition Computer for $59.95.—Ed.)*

Each of these last two offerings in the evolution of XBASIC has many good features and a few not-so-good features. RICHXB has the annoying habit of taking control of all XB program loading — even from a menu program like my Quest menu or John Johnson's BOOT program. The result is a several second delay in program loading and execution. SXB doesn't have the array of commands that are found in RICHXB and I particularly miss the CALL USER function.

So, when all is said and done, which one of these XBASIC extensions is the best? I guess it depends on what you want it to do. I probably use SXB more on a day to day basis than RICHXB but I still drag out my copy of Amerisoft from time to time to produce graphs and other things that are difficult or impossible with the others.

The best solution by far is to have several of the XBASIC extensions on hand and available. That way, you can pick out the one that has the features that you need for a specific application. Contact your local User Group to get a copy. Most of the early extensions have been widely distributed and the last two are available as commercial software for a reasonable price (I got a SXB cartridge at the Lima Fair a couple of years ago for $10). Add any (or all) of these programs to your library — they are well worth it!

Extended BASIC

Summarizing disk contents, counting the characters in a text file

By W. LEONARD TAFFS

This is reprinted from the February 1997 newsletter of the SouthWest Ninety-Niners — Ed.

If you use Super Extended BASIC you know how convenient it is. Its CALL CAT ("DSKn.") allows you to catalog any disk to ascertain free and used sectors. With Super XB you can interrupt (with FCTN/4) any XB program you may be running, or interrupt your programming, to use this CALL (in COMMAND mode) for this information. This is very handy if you are working on a large program and you know you are getting close to filling up a disk. TI users who do not have Super XB may assume there is no choice but to exit their program and XB to load a cataloging program. There is an alternative.

DISK SPACE TIP FOR REGULAR EXTENDED BASIC USERS

You can program a few lines *without* having to exit Extended BASIC to load a cataloging program to find out available free sectors left on your disk. I've not seen these program lines in any program other than disk cataloging programs. It is surprising to me this idea has not been incorporated more often in programs, and that I didn't think of it years ago, myself.

Enter these lines in your program at whatever point it can be useful:

```
80 OPEN #1:"DSKn.",INPUT ,RE
LATIVE,INTERNAL
90 INPUT #1:Z$,J,K,Z :: CLOS
E #1 :: RETURN
....
....
....
250 GOSUB 80
260 DISPLAY AT(12,1):Z$;" ha
s";Z;"secs. Av."
```

The letter "n" designates the drive of your choice. Omit the GOSUB line 250 and the "RETURN" in line 90 if you do not intend to use this as a GOSUB routine. Don't forget to enter "GOSUB 80" in your program if you use it as a GOSUB. Keep in mind Z$,J,K,Z variables are set once the routine is activated. If you do not know if these variables are used elsewhere in the program, the use of these lines may reset these variables elsewhere in the program. To protect yourself from this possibility, use different variables in line 90 or try entering line 160 as:

```
260 DISPLAY AT(12,1):Z$;" ha
s";Z;"secs. Av." :: Z$="" ::
J,K,Z=0
```

If you use the above as a GOSUB routine, you can repeat GOSUB direction within the program as often as necessary to keep a running account of your disk space. This could be especially useful if you are running a program processing several large files being saved to disk and it is important to keep track of space available. The routine may be called up with a CALL KEY option at significant parts of your program with something like:

```
700 DISPLAY AT(24,1):"Check
Disk Space? (0/1)
710 CALL KEY(0,K,S):: IF S<1
 THEN 710 :: IF K=49 THEN GO
SUB 80
```

Responding with "1" to line 700's prompt will direct program to the GOSUB for displaying available space. Any other response to this prompt will return you to continuing to the next line of program.

If you have neither Super XB nor regular XB, you can use this routine in TI BASIC:

```
80 OPEN #1:"DSKn.",INPUT ,RE
LATIVE,INTERNAL
90 INPUT #5:Z$,J,K,Z
100 CLOSE #5
160 PRINT Z$;" Has";Z;"secs.
 Av."
```

Note once you have loaded a program using this routine you need to designate the correct drive number you will be using in line 80 *before* running the program.

The following lines are for users with multiple disk drives and/or RAMdisks, which summarize available sectors. Note the program repeats the routine of lines 130-150 for each drive. If you copy these lines you need only enter as many repetitions of the 130-150 routine as you have number of drives. Enter your drive numbers/letters in lines 130, 160, 190, 220, 250, 280, 310 and 340 where you see "n". enter "1" at "Use Printer? (0/1) " prompt for a printout.

CHECKDISKS

```
1 REM [CHECKDISKS] 12-4-96
By W. Leonard Taffs, SW99ers
 !025
2 !!131
3 ! To See FREE Sectors
    Dsk 6,7,8,9,A,B,C,D !102
4 !!131
100 CALL CLEAR :: DISPLAY AT
(3,2):"CHECK DISKS FOR FREE
SPACE": :"          [CHECKDISK
S]": :"By W.Leonard Taffs, S
W99ers" !115
110 INPUT "Use Printer? (0/1
) ":PR !084
120 PRINT :: INPUT "DATE: ":
DT$ :: PRINT :: IF PR THEN O
PEN #2:"PIO" :: PRINT #2:TAB
(8);"Disk Space Summary ";DT
$: :!060
130 OPEN #1:"DSK1.",INPUT ,I
NTERNAL,RELATIVE :: INPUT #1
:A$,A,B,C :: GOSUB 410 :: PR
INT A$;" has ";N3$;" FREE":
: :: CLOSE #1 !122
140 IF PR THEN PRINT #2:TAB(
5);A$;"has ";N3$;" Free Sect
ors." !047
150 GOSUB 390 !215
160 OPEN #1:"DSK2.",INPUT ,I
NTERNAL,RELATIVE :: INPUT #1
:A$,A,B,C :: GOSUB 410 :: PR
INT A$;" has ";N3$;" FREE":
: :: CLOSE #1 !123
170 IF PR THEN PRINT #2:TAB(
5);A$;"has ";N3$;" Free Sect
ors." !047
180 GOSUB 390 !215
190 OPEN #1:"DSK3.",INPUT ,I
NTERNAL,RELATIVE :: INPUT #1
:A$,A,B,C :: GOSUB 410 :: PR
INT A$;" has ";N3$;" FREE":
: :: CLOSE #1 !124
```

(See Page 25)

EXTENDED BASIC —

(Continued from Page 24)

```
200 IF PR THEN PRINT #2:TAB(
5);A$;"has ";N3$;" Free Sect
ors." !047
210 GOSUB 390 !215
220 OPEN #1:"DSK4.",INPUT ,I
NTERNAL,RELATIVE :: INPUT #1
:A$,A,B,C :: GOSUB 410 :: PR
INT A$;" has ";N3$;" FREE":
: :: CLOSE #1 !125
230 IF PR THEN PRINT #2:TAB(
5);A$;"has ";N3$;" Free Sect
ors." !047
240 GOSUB 390 !215
250 OPEN #1:"DSKn.",INPUT ,I
NTERNAL,RELATIVE :: INPUT #1
:A$,A,B,C :: GOSUB 410 :: PR
INT A$;" has ";N3$;" FREE":
: :: CLOSE #1 !183
260 IF PR THEN PRINT #2:TAB(
5);A$;"has ";N3$;" Free Sect
ors." !047
270 GOSUB 390 !215
280 OPEN #1:"DSKn.",INPUT ,I
NTERNAL,RELATIVE :: INPUT #1
:A$,A,B,C :: GOSUB 410 :: PR
INT A$;" has ";N3$;" FREE":
: :: CLOSE #1 !183
290 IF PR THEN PRINT #2:TAB(
5);A$;"has ";N3$;" Free Sect
ors." !047
300 GOSUB 390 !215
310 OPEN #1:"DSKn.",INPUT ,I
NTERNAL,RELATIVE :: INPUT #1
:A$,A,B,C :: GOSUB 410 :: PR
INT A$;" has ";N3$;" FREE":
: :: CLOSE #1 !183
320 IF PR THEN PRINT #2:TAB(
5);A$;"has ";N3$;" Free Sect
ors." !047
330 GOSUB 390 !215
340 OPEN #1:"DSKn.",INPUT ,I
NTERNAL,RELATIVE :: INPUT #1
:A$,A,B,C :: GOSUB 410 :: PR
INT A$;" has ";N3$;" FREE":
: :: CLOSE #1 !183
350 IF PR THEN PRINT #2:TAB(
5);A$;"has ";N3$;" Free Sect
ors." !047
360 GOSUB 390 !215
370 PRINT :"Tot Av. Space: "
;TOT :: IF PR THEN PRINT #2:
 :TAB(8);"Tot Av. Space: ";T
OT;" Sectors." !192
380 END !139
```

Some years ago when I first realized my sorted D/V 80 files were growing beyond ... the capacity of a disk, I wrote a rather labored and lengthy program which read the file and tabulated the number of records for each letter.

```
390 REM ** CLEAR VARIABLES *
* !194
400 TOT=TOT+B-(B-C):: A$=""
:: A,B,C=0 :: RETURN !000
410 REM ** ADJ STR$ & NUM **
 !004
420 N$="     " :: N=LEN(N$)::
 N2$=STR$(B-(B-C)):: N2=LEN(
N2$):: N3=N-N2 !152
430 N3$=SEG$(N$,1,N3)&N2$ !0
87
440 A1=LEN(A$):: A2$="
     " :: A2=LEN(A2$):: A3=A
2-A1 :: A$=A$&SEG$(A2$,1,A3)
!057
450 RETURN !136
```

For more drives than provided for in the above program, use lines 130-15 as model to copy for programming extra drives.

Some years ago when I first realized my sorted D/V 80 files were growing beyond not only buffer sizes of Writer programs but also larger than the capacity of a disk, I wrote a rather labored and lengthy program which read the file and tabulated the number of records for each letter. It required a sorted file, of course. I failed to find the final version of this program, though I came across some earlier versions of it. It was quite a long program and if the user wished to tabulate records that were not individually programmed into these earlier program lines, the user had to add more program lines. This program worked well for what it did but it was much longer than needed and limited to tabulating only those records referenced in existing program lines.

As my recent collation of Masterdisk D/V 80 files was now beginning to get so big, I needed this program once again. The earlier version was called something like *EEZY*FCLT or FILERDRCNT — I simply have forgotten. Anyway, here is a much, much simpler version which will do the same job in a few lines; 90 percent or more of the following lines are limply screen display, user inputs, etc. The gist of the program doing the work is in lines 190-280. The screen display was set for relatively short records (less than screen width). If this program were to be used for up to 80 char line lengths then you can adjust the "I" variable in line 200 and the increment in line 240 according to your record size.

Here is the program:

COUNTFILRC

```
1 REM [COUNTFILRC] 12/20/96
By W. Leonard Taffs, SW99ers
 !072
2 !!131
3 ! Program to read records
    in a sorted file and
    tabulate by first char.
    of each record. !124
4 !!131
100 CALL CLEAR :: DISPLAY AT
(1,2):"D/V80 FILE READER PRO
GRAM": :"By W. Leonard Taffs
, SW99ers" !099
110 DISPLAY AT(6,2):"Program
 tabulates the number": :"of
 file records by the first":
 :"character in each record
of" !216
120 DISPLAY AT(13,8):"a sort
ed file." !107
130 INPUT "Use Printer? (0/1
) ":PR :: PRINT :: INPUT "DA
TE: (optional) ":DT$ :: PRIN
T :: IF PR THEN OPEN #2:"PIO
" !146
140 INPUT "File to read: ":F
N$ :: PRINT :: INPUT "From D
isk #: ":DSC$ :: FN$="DSK"&D
SC$&"."&FN$ :: PRINT :FN$:,"
```

(See Page 26)

EXTENDED BASIC —

(Continued from Page 25)

```
O.K.? (0/1) ": : :: INPUT ""
:OK !042
150 INPUT "Make outfile? (0/
1) ":OF :: IF OF THEN PRINT
:: INPUT "Save by name: ":SV
$ :: PRINT :: INPUT "To Dsk#
: ":DSC2$ :: PRINT !043
160 IF OF THEN OF$="DSK"&DSC
2$&"."&SV$ :: PRINT OF$:,"O.
K.? (0/1)" :: INPUT "":OK2 :
: PRINT !200
170 PRINT :: INPUT "Press an
y key to continue":K$ :: CAL
L CLEAR !109
180 IF OF THEN OPEN #3:OF$,O
UTPUT !121
190 CALL CLEAR :: OPEN #1:FN
$,INPUT !155
200 I=2 !002
210 ON ERROR 290 :: LINPUT #
1:A$ :: A=ASC(A$):: CT=CT+1
:: KC=KC+1 :: IF KC=1 THEN S
T$=A$ :: ST=ASC(ST$)!189
220 DISPLAY AT(I,1):A$;A1;CT
 !128
230 IF A=ST THEN A1=A1+1 !05
5
240 IF A<>ST THEN DISPLAY AT
(22,1):"There are";A1;" Recs
 of ";CHR$(ST);" " :: ST$=A$
 :: I=I+2 :: IF I=20 THEN I=
2 !217
250 IF A<>ST THEN IF PR THEN
 PRINT #2:TAB(5);"There are"
;A1;" Recs of ";CHR$(ST);" C
T:";CT$(FN$,6,10)!147
260 IF OF THEN IF A<>ST THEN
 PRINT #3:CHR$(ST)&" had "&S
TR$(A1)&" Recs in File "&SEG
 !131
270 IF A<>ST THEN ST=ASC(ST$
):: A1=1 !192
280 GOTO 210 !033
290 REM ** ON ERR/EOF ** !08
1
300 PRINT : :"End File: [";F
N$;"]":"with ";CT;" Recs." :
: IF OF THEN PRINT :"OUTFILE
 was: ";OF$ !071
310 IF PR THEN PRINT #2: :TA
B(8);"End of File [";FN$;" w
ith ";CT;" Recs. ";DT$ :: IF
 OF THEN PRINT #2:TAB(8);"Ou
t File was: ";OF$ !159
320 IF OF THEN PRINT #3:"~ E
OF "&OF$&" File "&SEG$(FN$,6
,10)&" had "&STR$(CT)&" Recs
."&DT$ !207
330 ON ERROR 340 :: CLOSE #1
 !119
340 IF OF THEN CLOSE #3 !098
350 STOP !152
```

Just what does this program do? It compares the first character of each record to the previous and following one. When that character changes the previous group is tallied, and so on through the whole file. This provides a list of how many As, Bs, Cs, etc., there are. When you have finished reading all the files you are checking you can find how many As or other characters there are in all your files. This helps yo plan how much you can fit in one file o disk.

When the program is run there are tw counters shown to the right of each record The first will be a tally of how man records there are, beginning with the firs character of each record. The counter fur thest to the right is the file record count The program gives you the option of view ing the results on screen, sending them t printer or sending to an Output D/V 80 fil as well.

User response uses the "(0/1)" choic for No or Yes. Enter "0" (zero) for a NO o a "1" for a YES at these prompts. This pro gram beats my earlier version by miles. I will begin with the lowest (first characte of your sorted file) and continue to the end of the file. If you copy this program, afte you have done so, You might want to cre ate a short file in sorted order of some records to test if your copying is OK. I you have copied it correctly, the result should equal the test file you made. Once you know it is working OK you can try i on a "biggie" file. Hope you can use thi program!

If you have accumulated a massive number of any such character files, a need might arise to fine-tune this process a step further, such as how many ABs, ACs, ADs, AEs, etc., are there? One can use this same program if it is modified to compare the second character of each line.

Screen pager utility

Assembly routines let you save screens in Extended BASIC

By MICHAEL ST. VINCENT

How often have you wanted to look at part of a program as it runs or set up an initial instruction screen that could be stored and recalled in an instant? If you are familiar with the almost complete impossibility of doing this, especially in the Extended BASIC environment and want to get free of such limits, here is your answer: an assembly language subroutine that is short and uncomplicated.

Simple solutions to problems such as screen storage are often overlooked in favor of staying strictly in one programming language's environment. Most people are unfamiliar with the usefulness of having machine language routines take over chores that are much slower in BASIC. To store a screen in BASIC, for example, most programmers would use a GCHAR to read all of the screen and store the result in an array. Besides being slow and inefficient, a BASIC routine to do such a task would use large amounts of memory.

Enter the amazing and fast 9900 machine language routine! The screen, usually a set of rows and columns to a BASIC programmer, becomes only a set of memory locations. In this form, moving a copy of the screen becomes as simple as assigning the assembly equivalent of a few variables and a GOSUB. Operation of

(See Page 27)

SCREEN PAGER UTILITY —

(Continued from Page 26)

the subroutines is kept simple by having the computer do the calculating. The possible applications of these subprograms are limited only by the programmer's imagination.

HOW THE PROGRAM IS USED

The subroutines, once assembled, are some of the simplest to use. Loading the programs into memory is accomplished by using a CALL INIT command followed by a CALL LOAD("DSK1.PAGER/OBJ") command, assuming you named your program PAGER/OBJ. The routines are automatically stored in memory and are invisible until needed.

Four programs are loaded simultaneously for use in Extended BASIC. They are PGSAV1, PGSAV2, PGSHO1, AND PGSHO2. The SAV programs save everything on the screen at the instant they are called to pages 1 and 2, respectively. The SHO programs return the previously saved pages to the screen. All four programs are accessed by CALL LINK("pgname"), where "pgname" is one of the program names given above.

The amount of time spent by the programs is measured in microseconds. Using OLD, SAVE, MERGE, and NEW commands have no effect on the screens stored in memory. Thus, one could list a program, save a screen of the list, load a new program, and still be able to look at the listing of the old program. The only restrictions on the programs are that they store only the characters. Neither colors nor sprites are kept.

HOW THE PROGRAM WORKS

These assembly language programs use a simple system of setting up a block of CPU RAM to store pages. Once a screen is to be stored, the registers 0, 1, and 2 are loaded with the address of the screen map in VDP RAM (000), the address of the CPU RAM block, and the number of bytes to transfer (768 for the full screen). A simple BLWP (branch and link with workspace pointer) command links to another utility routine that does the actual transfer.

After the transfer is completed, the program uses the pseudo-opcode RT to reset the workspace pointer to the BASIC interpreter area from where it branched. At that point, the BASIC level program continues to execute.

n the source listing which

HOW TO ASSEMBLE AND INSTALL THIS PROGRAM

Using the Editor/Assembler package, type in the source listing which follows, exactly as shown. Spacing is important to ensure that the program will assemble properly. Once the program is type in (you don't need to copy the remarks that are preceded by an asterisk), save the source code under the file name "PAGER/SOU." Then load the assembler.

When asked for the source file name, give "DSKx.PAGER/SOU," and when asked for the object file name, give "DSKx.PAGER/OBJ." If you have a printer, give the device name at the prompt, otherwise hit Enter.

The options for assembly are "RSL" if you have given a printer device name, or "RS" if you haven't. The assembler should do its job within a few seconds and should print "0000 ERRORS" when it is done. If there are any errors during assembly, refer to the source listing and compare it to what you typed. As listed, the program assembles with no errors.

The amount of time spent by the programs is measured in microseconds. Using OLD, SAVE, MERGE, and NEW commands have no effect on the screens stored in memory. Thus, one could list a program, save a screen of the list, load a new program, and still be able to look at the listing of the old program.

If you want to use this program in BASIC with the Editor/Assembler module, change the lines to match this header:

```
*
      DEF  PGSAV1,PGSAV2,PGSHO1,PGSHO2      * NAME ROUTINES
      REF  VMBR,VMBW
*
SCRMAP EQU  >0000          * START OF SCREEN MAP ADDRESS
SCRCNT EQU  768            * NUMBER OF CHARACTERS IN MAP
  Operation of the program is the same as described for Extended
BASIC.
```

SCREEN PAGER UTILITY

```
* THE SCREEN PAGER UTILITY
* SOURCE CODE WRITTEN BY MICHAEL ST. VINCENT
* USED TO STORE UP TO 2 SCREENFULS FOR LATER USE
*
      DEF  PGSAV1,PGSAV2     * NAME ROUTINES
      DEF  PGSHO1,PGSHO2     * NAME ROUTINES
*
VMBW   EQU  >2024          * VDP WRITE ROUTINE
VMBR   EQU  >202C          * VDP READ ROUTINE
SCRMAP EQU  >0000          * START OF SCREEN MAP ADDRESS
SCRCNT EQU  768            * NUMBER OF CHARACTERS IN MAP
*
PAGE1  BSS  768            * STORAGE BUFFER 1
PAGE2  BSS  768            * STORAGE BUFFER 2
*
PGSAV1 LI   R1,PAGE1       * ACTIVATE BUFFER 1
       JMP  GOSAVE         * GOTO THE SAVE ROUTINE
PGSAV2 LI   R1,PAGE2       * ACTIVATE BUFFER 2
GOSAVE LI   R0,SCRMAP      * STARTING POINT TO READ FROM MAP
       LI   R2,SCRCNT      * NUMBER OF BYTES TO MOVE
       BLWP @VMBR          * "GOSUB" TO READ
       RT                  * RETURN TO BASIC
*
PGSHO1 LI   R1,PAGE1       * ACTIVATE BUFFER 1
       JMP  GOSHOW         * GOTO THE RESTORE ROUTINE
PGSHO2 LI   R1,PAGE2       * ACTIVATE BUFFER 2
GOSHOW LI   R0,SCRMAP      * STARTING POINT TO REPLACE MAP
       LI   R2,SCRCNT      * NUMBER OF BYTES TO MOVE
       BLWP @VMBW          * "GOSUB" TO WRITE BACK TO MAP
       RT                  * RETURN TO BASIC
*
       END                 * TELL ASSEMBLER TO STOP.
```

TI-Artist Plus

How to use color

We found this tutorial in the June/July 1997 issue of Spirit of 99, *the newsletter of the Central Ohio Ninety-Niners. — Ed.*

The purpose of this is to help users of TI-Artist learn how to use TI-Artist Plus learn how to use color to make better pictures.

Chris Bodenmiller demonstrated the use of color at the Multi User Group conference in May 1996 and I was impressed and this led me to look at the video of his demonstration and then try out some of the techniques. My intent was to make interesting pictures for my videotapes. Here is the first tutorial illustrating how to make some meaningful pictures.

My intent was to make interesting pictures for my videotapes. Here is the first tutorial illustrating how to make some meaningful pictures.

In our first trial we shall simply make a picture which has a title and an imported graphic.

Before loading TI-A+ (TI-Artist Plus) prepare a graphic Instance by using TIPS (TI Print Shop V1.8) or by using PIX PRO or some other conversion program. Let's say, for example, that we have a car graphic, which is saved to disk as DSK2.CAR_I. It is a D/V 80 file which simply is filled with numbers to represent the car. We don't need to know the file structure to be able to use it.

Now load TI-A+ and enter the Artist segment. (You can use the mouse or joystick or keys to move around in TI-A+.)

The first thing that we will do is create a colored background. We only have to do this once and then we can use it in all of our graphics.

From the Artist screen

1. Choose the background from the *top* row of colors. I choose the tan color, the second from the left on the top row. You will see a tan square to the left of the black cross-hair.

2. Press M or pick the Mirror Icon.

3. Select the Color Cursor by pressing the cross-hair while on the color bar.

4. Now go to the picture screen (space bar or the mouse buttons will take you there).

5. Press the mouse button or the joystick to activate the screen color background. You will see that there is a color bar put down on the screen and mirror on the other side. What you want to do is make the entire background this color. Even if you don't, the background color will be transparent. This is the default TI-A+ background. This is pretty useless; that's why we are making a colored background.

When the *entire* screen is the color that you have chosen (tan in my case), then save it to your device. Press "S" or select the Store icon. I save my background with the file name BLANK.

After you have saved the background color, escape from the Artist segment. In the Main Menu, select Enhancements (E).

1. When you get to the Enhancements screen select I for instance.

2. I then use I for Index and index the drive where I have stored my CAR_I file.

3. Load the car instance.

4. Use "T" to test where you want the picture of the car.

5. Then use the appropriate key or button to place the car on the background.

Exit the Enhancement menu and return to the Main Menu.

Now select F for Fonts.

1. Then F again when the new screen comes up.

2. Index the drive that your fonts are on and select one of the fonts that you like. Let's choose a larger font like PCSET1>

3. After the font has loaded, press the space bar to go back to the Edit screen.

4. Press "E" to write our message.

5. Type in "OUR VACATION"

6. Next use Function 9 to exit the editor and then the space bar to see the screen.

7. There will be a moving box of lines on the screen. Use "T" to test where we would like the message.

8. When you have it where you want, then press the fire-button or press Enter to

(See Page 29)

Quick and dirty summary

A. ARTIST SEGMENT
1. Pick B-color
2. "M" Mirror
3. Use color bar
4. Space bar to screen
5. Paint screen
6. "SAVE" Save to device

B. ENHANCEMENT SEGMENT
1. "I" for Instance
2. "I" to index
3. Load Instance
4. "T" test position
5. "FIRE" to insert

C. FONT SEGMENT
1. "F" for font
2. "L" load font
3. <Space bar> to editor
4. "E" to edit
5. "OUR VACATION"
6. Fctn 9 to exit and <Space bar>
7. "T" to test position
8. "FIRE" to leave message

D. ARTIST SEGMENT 2
1. Select B-color
2. Select F-color
3. "N" to swap
4. <Space bar> to pix
5. "FIRE" and band
6. "FIRE" to change
7. "S" to store

TI-ARTIST —

(Continued from Page 28)

insert the message. The message should be in black.

Again press space bar to get back to the editor screen. You could add more text, and/or use a different font, but for the purpose of this tutorial we have had enough.

Escape to the Main Menu.

Enter the Artist segment again.

What we can do to spice up the picture is to add some color.

1. Select the background color *first*.. Use the same one that you had. this is *very important.*

2. Select a foreground color. Let's pick a blue. You should now see that the cross-hair cursor is blue and the little box to the left is tan (or your selected background color).

3. Press the "N" key or select the Swap icon.

4. Go to the picture (use space bar).

5. Press the fire button or the mouse button just to the top and *left* of the word "OUR"; expand the box so that it takes up the entire words "OUR VACATION"

6. Then press the fire button again or the mouse button again.

Presto chango. The black title is now blue.

You can go back and change it to black, by selecting BLACK for the cursor (cross-hair). Or you can change one letter or word to another color. You can even do one letter at a time. However, you may get some bleeding because the color boxes are 8 pixels wide.

You can use the same procedure to change the car to a different color. Remember that you can only change pixels that are *on* to a different color. These pixels are *foreground* colors.

7. When you are finished, you can save your colored picture to disk using the STORE icon or "S" command.

The Printers Apprentice

Converting fonts

By RICK FELZIEN

The author of this article, dating from 1988, was a member of the West Jax 99ers.—Ed.

The Printer's Apprentice has many fine fonts and they are mostly near letter quality, but I personally wanted to do some decorative lettering without having to load in the TI-Artist program just for that purpose. So here is how to convert a font from Artist to TPA.

The first thing to do is load up the TI-Artist Enhancements program and then type all the letters of the font to the screen, leaving a little space between. For this practice session, let's use the script font or font 19 on the font disks I placed in the library. As I said, type the letters to the screen and then go to the main Artist program and save the screen as a picture.

Now we can load up the TPA disk and select Picture Editor from the main menu. Load in your picture saved from Artist and don't forget the "_P" suffix. When you enter the picture editor you will get a blank screen with a flashing cross cursor. Press CTRL(8) to get the Load/Save option menu which looks like this:

Filename Dir Load Save eXit

Select F)ilename and type in "DSKn.filename" and Enter, then select L)oad and Enter. This will load the file and place the letters on the screen. Now use CTRL (=), which puts you in Klipper mode, which is similar to the clipboard mode in Graphx. Here you are prompted for a file name which should be the name that you want to give to your font. You sill be asked, "Create a new font-file?(Y/N)" at which point you would respond with a "Y"; after the disk file is created you are placed back in the picture editor and are ready to start saving your letters to the fontfile.

First place the cursor at the upper left corner of your letter, the first being "A," and press FCTN(5) to place the marker at the cursor position, and then move the cursor to a clear area of the screen and press Enter. You will then see a prompt near the cursor for Char. Here you would enter the letter you are saving (in this case, "A"), then hit Enter again. You will now be prompted W or X, which means Write the letter or eXit without doing anything. Use W and the character will be saved to the fontfile. Do this for all the characters that you want to save to the fontfile and then exit the Picture Editor and load in the Character Editor.

When you enter the Character Editor you will see the following menu:

Edit Disk Print Convert Setup Help eXit

First select S)etup and enter S for single-height letters, then select D)isk, which will present this menu:

Filename Dir eXit

Enter F)ilename, and then enter the filename that you saved the font with, and then eXit to the main menu. You can now enter E)dit, which will place the cursor in the character editing area of the screen, which you will notice has a column of numbers at the left edge. These are the row numbers which aid in determining the height of the font, etc. You will also notice an active column counter in the upper center of the screen which keeps track of the cursor position column. Now you can begin editing your font. There are several things that you must do to set up the sizing of the characters of your font.

First use CTRL(9) to get to the menu on the right of the screen, which looks like this:

ASCII CHAR ASCII CODE CHAR WIDTH

ReadWriteExit

At the first prompt enter "A" and then just hit Enter for Code and Width for now. Enter "R" for read and the character will be displayed next to the column counter. There may be some garbage to the right of the character, as the clipper saves a 24x24 pixel area and may have saved a part of the next letter, but do not worry, this can be corrected.

(See Page 30)

FONTS —

(Continued from Page 29)

Now press CTRL(R) to copy the character to the editing area and then check to see if the top of the character is on row 1 and the left edge is in column 1. If not, delete rows and columns until it is in the proper place. You will now notice that the bottom of the letter is in row 13, so this means the font will be 13 rows high. Now if there is garbage at the right of your character, move the cursor to the left column of the garbage area and delete columns until it is gone.

Now move the cursor to the rightmost pixels of the character, and in this case you will see that it is 14 on the column counter, and use CTRL(9) again and leave the "A" at the first prompt, and at the second leave the character code. At the third prompt enter the width, which in this case is 14. Now, at the last prompt, enter "W" for write and it will write the changes and the value for the width, etc., for that character to the font file.

Now load in the lowercase "a" and copy it to the character editor section with CTRL(R), and make sure the lowest row of pixels is at row 13, like the capital "A" was. Now check where the top row of pixels is, in this case it should be 7, which is the LC capline, or height to which the lowercase characters rise. Since the font height is 13, the baseline, or line on which the letters sit, is 14. Now you can edit this letter and save it as you did the last one.

Now we must set up font height, so use CTRL(=) to enter font height control menu. Enter 13 for font height, 14 for baseline and 7 for LC capline. When you hit Enter at the last prompt, you will return to the editing area. When you save the next letter, then height information will be written to the disk.

After you have edited all of the characters of your fontfile, select Print from the main menu, and select W)riteindex. This writes a listing of the width and height values to the file. If this is not done after each editing operation, the spacing may not be right when you use the font for printing. If your saved file didn't contain characters such as colon or semicolon, you can create them and save them to the file while doing the editing process, and, by all means, create a space character for each font, sized according to the character sizes.

After you have created and saved your fontfile, you will naturally want to print it out to see how it looks, so now you can exit the Character Editor and load in the TPA Formatter.

Once you have loaded in the Formatter, selected V)ars from the main menu, and then enter (in my case "G" for printer type, yours may be different). Enter defaults for the selections, except for space character, which would be approximately 10, and 460 for the right margin.

Now enter the Jotter and Edit and enter all the letters in your font, and SaveF to the disk. At the main menu you will now see at the bottom of the screen, the following:
PrinterPIO.CR TxtfileDSK1.TEXT
FntfileDSK1.TYPER

If you have two drives you will want to change to the following: (the printer default is OK for most printers, if not, change it to match your printer specs). Hit "B" for buffer, and change to DSK2.SCRIPT for Fntfile. Then hit "G" for Go, and the formatter will print your fontfile to the printer.

If you want to create an Over/Under strike (high resolution) file from your fontfile, you can do so by entering the Character Editor and selecting C)onvert from the main menu, and then you will be prompted for a filename for this file. (Mike McCann uses "OU" before the names to be able to distinguish the Over/Under strike fonts.) I recommend that you do the same to avoid confusion. After naming the file the program will automatically create an Over/Under strike font from the Single/Strike font that you created before.

I hope that this article not only helped you to create a new font for the TPA program, but helped also to let you become a little familiar with this powerful publishing program.

Simplicity

A 'little masterpiece' in Extended BASIC

By CAL ZANELLA

(This appeared in the April 1994 newsletter of the Pomona Valley 99ers — Ed.)

Author: Bill Gaskill, 2310 Cypress Court, Grand Junction, CO 81506.

Several years ago, I embarked on a search for the ultimate personal financial manager software package for my hard-working TI99/4A. After several months of using three different packages, I settled on what I felt was the most powerful and complete package that I could find at that time. Many of you may remember this software package. It is Personal Auditor by Bill Gaskill.

Well, I have never looked back, because to this day I am still using these programs to keep track of my financial position. In my humble opinion, Personal Auditor had, and even to this day has, no rivals. The only complaint I had at the time was that the learning curve for Personal Auditor was quite extensive. However, my determination won out and I am now very comfortable using all the fine features of this masterpiece.

Well, this review is not about Personal Auditor! But it is a review of a software package that is another "little" masterpiece from Bill Gaskill.

The title of this masterpiece is "Simplicity" and it well lives up to its name. Like most of all Bill's software, this one was written in Extended BASIC. It is a modular package and works seamlessly and is very friendly.

The program boots up through an Extended BASIC load programs and presents a clean double-windowed options menu. The left window is titled FINANCE and the right window is titled UTILITIES. There are seven options available in each respective window. The options are as below:

FINANCE	UTILITIES
Business	Help File
Checkbook	Mail list
Estate Plan	Options
Investments	Planner
Loans	Tool Box
Retirement	Word Proc
System File	eXit

That is the beauty of this program. It gives you plenty of useful software to begin with, yet leaves room for additional user installed options as well.

As you can see, there is much to select from. Pressing the capitalized letter of any of the listed options will load in a submenu of choices that are pertinent to the area of finance or utility that was selected. For example, selecting Business (B) from the main menu would bring up a submenu that would present three choices.

1. Depreciation schedule.
2. Effective interest rate
3.
4.
5.
6.
7.
8.
9. return to main menu

As you will notice, option numbers 3-8 are blank. That is the beauty of this program. It gives you plenty of useful software to begin with, yet leaves room for additional user installed options as well. You really don't expect to have everything done for ya, do ya?

The Mail List, Checkbook, Planner and Word Processor submenus are complete full-featured programs of a Simplicity nature, i.e., they will do exactly what they advertise they will do, quickly and simply! Mail List allows the creation of a mail database and allows printing to labels or tabular reports. Checkbook has more than the usual features of checkbook programs and allows check reconciliation and the import of Personal Auditor files. Planner is a monthly reminders program and allows you to edit and print out a monthly calendar of reminders. The Word Processor is a somewhat limited editor (60 lines max) for whipping out single-page notes or short letters.

Simplicity gives you access to a host of other capabilities such as depreciation schedules, effective interest rates, ability to track mutual fund performance, print loan amortization mortgage loan rates, IRA (individual retirement account) projections, net worth statement, etc.

The Help file on the main menu explains just about all you will need to know to get the system going and the System file submenu will configure your floppy drive, printer, screen colors and set your beginning bank balance for the Checkbook. There is also on-line help available at most submenus by typing H or Function-7. These help screens are on the disk, as D/V 80 files, so, should all else fail, you can print them out through your favorite word processor or through the Simplicity word processor.

The Tool box submenu will set you catalog floppies and hard disks.

The hardware requirements for use of all features of this package are fairly minimal. You will need a trusty ol' TI99/4A, 32K memory expansion, one disk drive, Extended BASIC cartridge and a printer. This is definitely a lot of useful software on a single floppy disk.

Now, a few words of explanation!. I acquired the Simplicity disk purely by chance along with several other disks that I had requested from Frank Aylstock, president of the Brea Users Group. So, prior to sending this article of review to several users groups for publication in

(See Page 32)

SIMPLICITY —

(Continued from Page 31)

their newsletters, I thought it best to call Bill Gaskill personally to inform him of my intentions. My discussion with Bill revealed that he had decided to abandon this project due to a lack of interest on the part of the TI community! A classic case of "use it or lose it." Had enough interest been shown in this endeavor, I believe it would have blossomed into a single-unit package that would have met most all of your financial needs.

Bill did confirm that Simplicity has been thrown into the public domain pot. My suggestion, then, is for you to try it. If you find it useful and use it, I suggest that you have the simple courtesy of sending the author a few rubles for all the effort that he expended. Do it! Encourage survival!

MIDI-Master 99 gets upgrade

Bruce Harrison has made an improved version of MIDI-Master 99. This version is for users without AMS and precedes his planned version to work with the Super AMS Card.

The new version includes its own MIDI Album program and is said to be compatible with all source and program files created by or for older versions. According to Harrison, this new version, V2.5Z, features better ACCEPT fields, in which FCTN-1 deletes a character, FCTN-2 puts it in insert mode and FCTN-3 clears the entire accept field.

Harrison notes, "Even the cursor blinking has been improved, so that it now blinks at the same 'normal' rate on either TI or Geneve. (In older versions, the blink on TI was painfully slow."

Harrison feels that the biggest improvement is in saving music in memory image (program type) files.

Harrison feels that the biggest improvement is in saving music in memory image (program type) files. In older versions, the program always created such files in groups of three, totaling 100 sectors, regardless of the length of the music. V2.5Z saves only what's needed, depending on the music in memory. Short musical numbers may need only one file of 12 or 13 sectors, and that's all that V2.5Z will create. An improved version of MIDI Album is included on the disk as well, Harrison says. The album's random play has been changed so that each selected file plays only once, but all get played in random order.

This new version is available only to current owners of MIDI-Master 99 on sending $1 for media and shipping to Harrison at 5705 40th Place, Hyattsville, MD 20781.

1997 TI FAIRS

APRIL

TI-Faire, April 26. Hotel Starkenburger-Hof, Heppenheim, Germany. Contact Volker Papst, Heppenheim, Phone: +49-6252-2903 or Michael Becker, Mannheim, Phone: +49-621-722735

MAY

TI Users Group U.K Annual Group Meeting, May 10, Princess Anne St. Johns Ambulance Training Centre, Trinity Street, Derby, England. Contact Trevor Stevens, 249 Southwell Rd. East, Rainworth, Notts., NG21 0BN, UK, telephone 01623 793077, or BBS 01623 491282 (Friday 7 p.m. to Sunday 10 p.m. only, times are United Kingdom times).

Multi Users Group Conference, May 23-24, Ohio State University, Lima Campus. Contact Charles Good, P.O. Box 447, Venedocia, OH 45894. Phone (419) 667-3131. Preferred e-mail address good.6@osu.edu.

OCTOBER

12th International TI-Tref, Oct. 24-26, Ibis Hotel, Utrecht, the Netherlands. Contact Berry Harmsen, 1e Oosterparkstraat 141 E, 1091 GZ Amsterdam, the Netherlands. Phone: 00 31 20 6941047.

1998 TI FAIRS

FEBRUARY

Fest West, Feb. 14. Texas Instruments facility, Lubbock, Texas. Contact SouthWest Ninety-Niers, P.O. Box 17831, Tucson, AZ 84781 or access Tom Wills' web site (http://personal.riverusers.com/~twills/).

This TI event listing is a permanent feature of MICROpendium. User groups and others planning events for TI/Geneve users may send information for inclusion in this standing column. Send information to MICROpendium Fairs, P.O. Box 1343, Round Rock, TX 78680.

MICRO-REVIEWS

PC99 Stage 4 — practically perfect in every way

By CHARLES GOOD

The times they are-a-changing. Soon all schools will be hooked up to the Internet and all school children will have lots of experience using computers. Almost all areas of employment include using computers in one way or another. You can purchase computer software at Wal-Mart, Sears and other department stores as well as at computer stores. And almost none of this relates to the TI99/4A computer any more, or to the original IBM PC or 286 or 386 computers either.

Technology marches on, and many TIers who don't want to be left behind now also own powerful 486 or Pentium IBM compatible computers. Why do these PC owners still use their TI or Geneve? Because TI equipment still works well and they believe in the theory that if something isn't broken why try to fix it? Because years' worth of all your important financial, name and address and text file data are already in TI99/4A format on TI99/4A disks. Because they are familiar with and feel comfortable with TI99/4A software. Because existing TI software is so easy to customize, either by changing the BASIC code or by using a sector editor. Because they grew up with and still love playing those great 99/4A games. I remember in the middle 1980s when it was the dream of many TIers to have some sort of an upgraded system that would run both IBM compatible and TI99/4A software. There was talk of coprocessor machines with both a 9900 and an 8088 CPU. Then came the Miller Graphics interface box, which was not quite worthless. Well TIers, the time has finally arrived! The ultimate 99/4A upgrade system is here. It is called PC99 and consists of two disks you install on a 486 or Pentium, making the IBM behave *exactly* like a super-fast 99/4A. You can now easily transfer all you TI99/4A software to the hard drive of your favorite modern IBM-compatible computer and run this software on your IBM just as easily and probably a lot faster than on your trusty old 99/4A.

What's new in the latest v4 of PC99 is emulation of the Myarc floppy disk controller. This gives you 267 percent of the previous amount of on line TI software and for this reason I urge all current PC99 owners to upgrade.

I reviewed v2 of PC99 by CaDD Electronics several years ago. At that time, although I didn't say so in my review, I considered PC99 to be so slow that it was practically useless using the IBM-compatible computer I owned at the time. PC99 has changed so much since then that it is really a new product and deserves a second and much better review. What's new in the latest v4 of PC99 is emulation of the Myarc floppy disk controller. This gives you 267 percent of the previous amount of on line TI software and for this reason I urge all current PC99 owners to upgrade. The two most important things that have changed in PC99 since my original review are that PC99 is now much faster, and that I now have a Pentium 75 desktop and a Pentium 100 laptop on which to run PC99. In my original review I only had a 386-40 computer, and the old version of PC99 ran very very slowly on such a machine. On my Pentium computers I have to slow down the speed of PC99 in order to make it run at 99/4A-like speed. The PC99 doc recommends a 486-66 or faster computer for reasonably fast emulation. I've tried the current version on a 386-40 and it is kinda slow, but still usable. PC99 is a DOS program that will run out of DOS or from within a DOS window under Windows 95. It does not like running out of a DOS window under Windows 3.1. Minimum hardware requirements are a 386 computer with DOS 5 or higher, 8 megs of rAM, VGA video and 10 meg of free hard drive space. A Sound Blaster or similar sound card is optional. PC99 uses your IBM's printer port as the TI's PIO, and your IBM's com 1 and/or 2 ports as the TI's RS-232/1 and/or RS-232/2 port. If your IBM has "game" ports you can connect joysticks to these ports and have them emulate TI joysticks. If not, the action of TI joysticks can be obtained by pressing special keys on the IBM keyboard.

PC99 emulates the following TI system: console, two joysticks, 32K memory card, RS-232 card with one PIO and two RS-232 ports, Myarc floppy disk controller with four DSDD drives, TI floppy disk controller (in case you don't like the Myarc FDC), Editor/Assembler cartridge, Extended BASIC cartridge, Tombstone City cartridge, PC99 doc and Myarc's FDC doc. You also get the following 99/4A software: Myarc floppy disk manager, Archiver III and some other useful TI software. This 99/4A software comes on a PC99 "disk" PC file ready to use immediately within PC99. There are also a host of PC utility programs that let you move your files and disks between the PC and TI operating systems. That's a lot of TI equipment and it all fits onto two little IBM-compatible floppy disks. All this hardware, disk software, command modules and docs exist as IBM files on the two IBM-formatted disks. You also get a 99/4A disk with software to run on your real 99/4A.

The first thing you do after installing PC99 on your IBM's hard drive is run the cfg program to configure PC99 to your

(See Page 34)

MICROREVIEWS—

(Continued from Page 33)

system and preferences. If you have a mouse attached to com1 then you have to reconfigure the TI's RS-232 to com2. To get the full benefit of the TI's sound you have to tell cfg about your IBM's sound card. With a Sound Blaster-compatible sound card your IBM can emulate the full glory of the TI's three sound channels and one noise channel. Since my laptop has no sound card it can only produce channel 1 of the TI's sound through the little PC speaker. PC99 comes set up to use the TI disk controller card, but you will probably want to use cfg to change this to the Myarc floppy controller card. The TI controller card only gives you three DSSD drives while the Myarc card lets you use four DSDD drives. Although not obvious when running cfg, the easiest way to install the Myarc controller from within cfg is to select "Peripherals" and then select "Setup peripheral." If you have one or two joysticks attached to game ports on your IBM you need to use cfg to set these up to emulate the TI's joysticks. Otherwise you have to use ALT+F1-10 key presses to emulate joystick action. These key presses give you up/down left/right and fire for each of the two joysticks, but no diagonals. Real IBM joysticks will give you diagonals. There is no provision for using the IBM's mouse to emulate a TI joystick. In fact, the IBM mouse isn't used at all in PC99. There is lots of other stuff you can set up or altered using the cfg program, some of which can also be altered from within a running PC99 emulation session. After you have PC99 set up the way you like the main routine thing you will have to use cfg for is changing command modules. To change command modules you have to shut down PC99 emulation, run cfg to change the module, then restart PC99. This is, of course, just about the same thing you have to do with a real 99/4A system when changing command modules.

The next thing you will want to do after installing and setting up PC99 is to transfer your important TI software to your IBM to run under PC99. Unfortunately you can't just put a 99/4A-formatted disk into a floppy drive connected to your IBM and download the TI software. Folks have talked about this possibility for years, but it will probably never happen. There are three ways to move a disk of 99/4A software to your IBM for use with PC99, and by far the easiest of the three ways is to cable the two computers together. The cable you need can be purchased at almost any store that sells computer stuff. You need a "modem cable." I purchased mine at Wal-Mart for $7. My cable is 10 feet long, which is plenty long enough to connect two side by side desktop systems or to connect a desktop to a laptop. One end of the cable connects to the 99/4A's 25-pin RS-232 and the other to my laptop's 9-pin com1 port. If you have a mouse at com1 you will have to disconnect the mouse and boot your computer in DOS mode. After the TI and IBM are cabled together you start PC99 on the IBM. Then you run some software within PC99, some other software on your real 99/4A, put your 4A disk in one of the 4A's drives, and watch it move over the modem cable and be converted into a PC99 "disk" on your IBM. This is an easy one-step process. You can transfer SSSD DSSD or DSDD TI disks this way, and it only takes a few minutes per disk. The resulting PC99 "disk" is a file on your IBM hard drive that contains what is essentially a memory image of the 99/4A disk, with all the software data plus all of the TI disk's sector and track data. This ability to quickly transfer any disk based software from a 99/4A to PC99 on an IBM using only a commonly available cable and software that comes with PC99 is really outstanding! The only limitation to this system of disk transfers that I have been able to discover is that it doesn't work with a Geneve. You need to cable a real 99/4A system to your IBM. If you only have a Geneve then you must use one of the other two methods of converting 99/4A formatted disks to PC99 "disk" files on an IBM disk or hard drive.

If you have one or two joysticks attached to game ports on your IBM you need to use cfg to set these up to emulate the TI's joysticks. Otherwise you have to use ALT+F1-10 key presses to emulate joystick action.

If you have a Cor-Comp or Myarc disk controller on your 99/4A or Geneve then another way to transfer TI software from TI disks to your PC is by using the TI software "PC Transfer." This does not require cabling the PC and TI computers together and it will work with a Geneve. PC Transfer is available from TI dealers including the one listed at the end of this article, and requires a DSDD disk controller. First, you transfer your TI software onto a 360K PC disk on your TI system using PC Transfer and some software that comes with PC99. This is a two-step process. You then put the PC disk into your PC and perform several layers of file manipulation in order to create a "disk" you can use with PC99. This is a more complicated procedure than just sending the TI disk to the PC via a serial cable.

The third way to move your software from TI disks to PC disks is to use CaDD's disk duplication service. You send CaDD your SSSD or DSSD or DSDD disks, some money, some PC disks, and a paid return mailer and CaDD will do the work for you. The fee is $0.25 for each SSSD or DSSD disk, or each side of a flippy disk. I have been visited the CaDD world corporate headquarters and have seen the special TI cabled to a 486 PC that is specifically reserved for this service. The equipment is set up in a corner of a spare bedroom in Mike Wright's townhouse. Converting large numbers of 99/4A disks to PC99 "disk" files disks is *very* boring work and requires someone to sit in front of the cabled computers all the time to switch disks and restart the copy programs. The fee charged in part compensates Mike Wright for his time.

There are several ways to get a TI direc-

(See Page 35)

MICROREVIEWS —

(Continued from Page 34)

tory of your PC99 disks. Using PC99's Myarc controller you can from TI BASIC or Extended BASIC command mode type CALL DIR(x) to get a listing of the contents of drive x. You can also use the Myarc disk manager, Funnelweb, DM1000, DSKU or any other TI disk management software to get a directory from within a running PC99 session. Using software provided with PC99 you can get a directory of a PC99 "disk" from the DOS command line, and you can get such a directory while running cfg.

After you get your favorite TI disks converted to PC99 files on your PC you will want to set up your 99/4A-on-a-PC computer. With PC99's emulation of the Myarc controller you can have on line up to four 1440 sector "disks" of TI software. You can set up PC99 to act similarly to a real 99/4A that contains several Horizon RAMdisks. Have Boot or Funnelweb automatically come up when you select Extended BASIC from PC99's startup menu and have all your best TI software configured to run from menus out of drives DSK1 DSK2 and DSK3. Leave DSK4 for your working disk. You can switch this disk, or any PC99 disk, from within the emulator just as you can switch disks while running software on a real 99/4A. Everything is running off of your PC's hard drive at speeds equal to or faster than a real 99/4A with big Horizon RAMdisks.

How fast is PC99? It is my subjective opinion that on a 486-100 or a Pentium 75 PC99 runs at about normal 99/4A speed. On my Pentium 100 laptop PC99 definitely runs faster than a 99/4A and maybe faster than my Geneve. There is a way to slow PC99 down if necessary. Either from the cfg program or from within a running PC99 session you can specify a percentage of full speed. Slowing down PC99 is fun to try with games. You can slow down TI games to less than their normal speeds, which allows you to anticipate what will be happening and to achieve great scores as a result. On my Pentium 100 system I have to slow down many games because they run much too fast.

What about other command modules? PC99 can use module disk files created

How fast is PC99? It is my subjective opinion that on a 486-100 or a Pentium 75 PC99 runs at about normal 99/4A speed.

with the Gram Kracker, P-Gram, Gramulator, or other GRAM device. You send these TI disk files over a cable or via PC Transfer to your PC and then manipulate them into PC files that are recognized by PC99 as a command module. You change command modules using PC99's cfg program. If you don't have a GRAM device you can purchase the command module PC99 files directly from CaDD. Every command module ever sold or licensed by TI is available, including several that you probably have never seen. The price per module decreases rapidly with volume orders. Expect to pay $2.50-$4.50 per module. These are legal copies of TI's copyrighted code and TI gets a royalty payment for every set of module files sold.

Information strips, like those that come with command modules and are displayed above the number keys on the 99/4A, are available with PC99. These are displayed on the right of the PC99 screen on your PC's monitor. TI BASIC and E/A module information strips come with PC99. You can make your own custom information strips using any PC ASCII editor.

Keys on the IBM keyboard usually act as you would expect them to on a 99/4A keyboard. The IBM arrow keys act like the 99/4A's arrow keys. The AFT key acts like the 99/4A's FCTN key. You can also press the IBM's F1-F10 keys to get the same response as the 99/4A's FCTN/1-0 keys. There are a few 99/4A key presses, such as the CTRL/= used in Funnelweb's word processor, that you can do on a 99/4A keyboard but which aren't recognized by the IBM keyboard. You have to enter special keypress sequences to simulate these special 99/4A key presses, as explained in the PC99 doc. Since my laptop has no "game" ports I am forced to use ALT/F1-10 key presses in my IBM keyboard to emulate TI joysticks. At first I found these ALT key presses very unresponsive when playing joystick games. Later I discovered that there is a keyboard delay, or "k" value which can be set by cfg or from within a running PC99 session that will make the keyboard seem much more responsive at the expense of possibly missing some key presses. When playing joystick games I recommend resetting this "k" value to 3000 instead of the 1500 default value.

There are two versions of PC99. The "light" version costs $47 and comes with everything described above. The "full" version costs $94 and also includes an implementation of the 99/4A's "Review Module Library" feature, an emulation of the P-code card with the ability to run P-system software, and a marvelously extensive debugger screen. Review Module Library allows you to have several modules immediately available at the 99/4A's menu screen. You can page through the on-line modules and select the one you want. All GPL routines in all the on-line modules remain available, so that, for example, you can have theoretically TE II speech while running Extended BASIC. There are very severe limitations to the kinds of modules you can have on line because of the way TI designed this feature. Only one module with RAM can be among the group of on line modules. This makes "Review Module Library" a not very useful feature, which is why TI never actually used it in the products they sold.

The PC99 "full" version includes emulation of the P-code card and minimal P-system documentation. You turn on and off the P-code card from within cfg. You need to already own P-code-compatible software or TI's P-system assembler, linker and utilities in order to utilize the P-code card emulation. These software products and their extensive documentation are available from some 99/4A dealers but not from CaDD. You need to convert the software from TI disks to PC files.

(See Page 36

MICROREVIEWS —

(Continued from Page35)

Limited debugging information is available with the "light" version. The "full" version's debugger screen is marvelous! In the upper left corner you get a small version of the 99/4A screen, which is easily readable at VGA resolution and which is fully functional. The rest of the PC screen is filled with memory displays. You can watch these displays change as you run PC99, or you can stop PC99 and edit the displays. In edit mode you get a cursor you can move all over the screen using the arrow keys, entering memory values in either ASCII or hex. You can also set break points and do lots of other neat stuff. This is the most comprehensive 99/4A editor available anywhere. You can view and edit all areas of 99/4A memory from outside the 99/4A environment. If you are an assembly or GPL programmer then you need this debugger. If you are a mere mortal user interested in seeing how a running application affects memory then you also will find this debugger useful.

With the following exceptions, virtually *all* 99/4A software on disk or as command modules will run on PC99 *exactly* as it does on a real 99/4A. If you have a 486-100 or faster it will run at or above regular 99/4A speed. Here are the exceptions:

1 — You can't load from cassette. You can save to CS1 and examine the contents of the saved file in DOS, but you can't load the saved cassette program or file back into PC99.

2 — There is no speech. Software that expects to find the speech synthesizer will not lock up PC99, but you hear no spoken words.

3 — You can't run 99/4A software that has an 80-column mode such as 80-column Funnelweb. This is because PC99 emulates the 9918A video processor, not the 9938 or 9958 video processors used in 80-column 99/4A systems.

4 — No Geneve-specific software will run on PC99.

5-You can't run software that is specific to cards not yet emulated by PC99. This includes software for the AMS card and Myarc Extended BASIC for the Myarc memory expansion card.

I consider PC99 the greatest software to ever come along for owners of both 99/4A and IBM compatible computers. I can now run my 99/4A checkbook program, my 99/4A name and address database and manipulate all my important DV80 files on my IBM-compatible computer at my office and on my IBM compatible laptop and desktop at home. I urge all 99/4A+PC owners to purchase this product and I urge all who have already purchased earlier versions to upgrade to the latest version. The upgrade from the previous version costs only $7, which includes postage and media.

I consider PC99 the greatest software to ever come along for owners of both 99/4A and IBM compatible computers.

And now a word about the other emulator. There is a freeware (meaning no donation to the author is expected) 99/4A emulator available for downloading on the Internet called V9T9 which is very capable. You are supposed to be able to send software from a 99/4A to V9T9 using the type of serial cable I use with PC99. However, for whatever reason, I couldn't get V9T9 to transfer stuff from my 99/4A to my IBM a couple of years ago, so I gave up on the thing. More recently I have put a preconfigured version of V9T9 on my desktop and enjoy playing its command module games, but I still have problems moving my disk software over to V9T9. V9T9, unlike PC99, lets the mouse emulate TI joysticks, and on some (but not all) IBM-compatibles this mouse emulation works very well. Others have used V9T9 with varying degrees of success. It is supposed to allow you to convert your software from a 99/4A to V9T9 format on a PC by cabling the two computers together, and for some people this works OK. However, not all 99/4A software will run under V9T9 and some users report having trouble making the needed cable. Although V9T9 is free to use, the author now provides no support for his product. If you want help you have to ask other V9T9 users. There is always a discussion about V9T9 to be found on the Internet listserver that is devoted to the 99/4A. If you already have disks or individual 99/4A programs converted to V9T9 format PC99 includes utilities to convert these V9T9 files into PC99 disks.

Why purchase PC99 when V9T9 is free?

1 — PC99 runs everything. It is the better of the two products.

2 — It is easy to convert your 99/4A software to PC99 format without a custom-made cable.

3 — PC99 is supported. This support manifests itself in several important ways. First, there is a (not toll-free) telephone number you can call for assistance from the PC99 authors. Second, you can very inexpensively have all your 99/4A disks copied into PC99 format by paying for the CaDD Electronics disk duplication service. Third, a complete line of TI command modules and TI documentation (reviewed by me in the May/June 1997 MICROpendium) for use with PC99 is available from CaDD. Fourth, PC99 has in the past been regularly upgraded and made available to current owners at a very low cost, always less than $10 There is no guarantee that any additional upgrades will appear, but this is a distinct possibility. For example, CaDD is looking at the possibility of emulating the AMS card in a future upgrade.

PC99 is a very professional product with professional quality support. It is great!

ACCESS:

CaDD Electronics (source for PC99), 45 Centerville Dr., Salem, NH 03079. Phone (603) 895-0119 or (603) 893-1450, e-mail mjmw@xyvision.com.

Ramcharged Computer (source for PC Transfer), 6467 E. Vancey Dr., Brook Park, OH 44142. Phone (216) 243-1244.

Charles Good (your humble reviewer), P.O. Box 647, Venedocia, OH 45894, Phone (419) 667-3131. Preferred e-mail good.6@osu.edu.

USER NOTES

Mechatronic XB makes good use of CALL ALLSET

The following item was posted to the TI list server by Mike Wright of CaDD Electronics(mjmw@xyvision.com).

I am in the process of tweaking the Mechatronic Extended BASIC II Plus manual for release to PC99 customers in Adobe Acrobat format. We are pleased to announce that we will be able to offer this version of BASIC — which is based on TI Extended BASIC but offers many extras — as a shareware product to PC99 users, with the permission of Mechatronic GmbH of Germany. A full announcement regarding this will be made when the work is complete.

In the Mechatronic manual, I came across a tip that I don't think I've ever seen published:

If you've keyed in a long and complicated BASIC listing, with a large number of CALL CHARs, and then run the program, there are often "bad" graphics resulting from keying errors in the CALL CHARs. If you press BREAK the ASCII codes 32-96 will be reset to their standard values. This is because CHARSET was designed for the 99/4, which had no "lowercase." This makes it pretty hard to track down the errant character.

However, Mechatronic XB II Plus includes the command CALL ALLSET, which resets **all** codes (32-126) to their standard values. If you put this in a program at a strategic point, and then do a CALL KEY immediately afterwards, the CALL ALLSET will reset the characters in the graphic image to their standard values. If you point to the screen at the "bad" part of the graphic just before CALL ALLSET is executed, you will be able to find the standard ASCII code for that character. Then you just need to look in the listing for a CALL CHAR of that value and compare it to the original listing to find the error.

In standard TI Extended BASIC the above would still work to a certain degree using CALL CHARSET, but the lowercase characters would not be reset.

Extended BASIC doesn't have a bug

The following comes from Gary Bishop, of Marion, Iowa. He writes:

In response to Martin Zeddies program in March/April MICROpendium about an apparent XB bug, and Oliver D. Hebert's response in the May/June issue, I contend there is no bug, and XB is behaving very predictably and exactly as advertised. This may confuse the programmers that want the result to be P=1, but it can quickly be explained.

(See Page 38)

USER NOTES

(Continued from Page 37)

To start with, type in the program as indicated, omitting unnecessary comments as such:

```
140 D=0
150 IF D=0 THEN GOSUB 170 EL
SE Y=5 :: P=3
160 GOTO 180
170 P=1 :: RETURN
180 PRINT "P="&STR$(P)
190 STOP
```

Running this program produces a result of "P=3" which is exactly what I would expect. Here's why:

Refer to the XB manual page 157 for RETURN (with GOSUB) which states: "RETURN used with GOSUB transfers control back to the statement after the GOSUB..."

Now refer to page 38 which says "a double colon (::) is used to separate statements on a line."

Finally, look at page 94 under IF THEN [ELSE], the first example which is:

```
100 IF X=5 THEN GOSUB 300 ELS
E X=X+5
```

The description for this example is "when the subroutine is ended, control returns to the line following this line." Because we have used a statement separator in our original program, this should actually be interpreted as saying "when the subroutine is ended, control returns to the statement following the IF THEN [ELSE] statement, which is the P=3 assignment."

If the TRACE command is issued and our program run, the results may also confuse the unwary. The line number sequence as shown on the screen will be: <140><150><170><160><180><190> which seems to skip over line 150 for the RETURN and proceeds directly to line 160. However, the XB manual can help us out here also. Looking up TRACE, we find that it "causes each line number to be displayed on the screen BEFORE (emphasis is mine) the statements on that line are executed." I believe it was Jim Peterson who found out if the line number is displayed once, branched out and then returned, this line number will not be displayed again because it was already displayed. One sure fire way to untangle this is to rewrite the program eliminating the statement separators, so the TRACE command will show exactly what is going on. Add lines 155 and 175 as shown:

```
140 D=0
150 IF D=0 THEN GOSUB 170 EL
SE Y=5
155 P=3
160 GOTO 180
170 P=1
175 RETURN
180 PRINT "P="&STR$(P)
190 STOP
```

Line 150 will transfer to 170, which dutifully assigns the value of 1 to the variable P, but upon return, line 155 simply overwrites it with 3.

With our faith in XB firmly restored, I trust readers Zeddies and Hebert will no longer consider the previously noted operation a bug, but simply XB doing exactly what we asked of it. They just weren't asking it properly for the desired result.

Internet costs

The following item came from the Dallas 99 Interface newsletter.

Next time you pay your Internet service provider bill, give a thought to the following message posted by a TI99/4A user from Germany.

"As for connection costs, here are some numbers that I calculated, referring to users in Germany. After hearing that U.S. citizens do not pay for local calls (unbelievable!), here are our "local" connection costs:

If I were connected a whole day long, it would cost:

DM 77,76 ($46) per day from Monday to Friday;

DM 60,48 ($36) per day on Saturdays and Sundays.

Based on a month with five weekends during 30 days, that yields:
DM 2160,- ($1,270) per month."

TI-Writer punctuation problems and solutions

The following item has appeared in several TI newsletters.

Typists will often use two spaces after a punctuation mark ending a sentence. TI-Writer does things a little differently.

THE PERIOD — TI-Writer will always put two spaces after every period that is followed by a single space. This is fine if the period is at the end of a sentence, but what if you are using an abbreviation within a sentence? The formatter will put two spaces here also, even though you want only one. What you need to do in this case is use the required space symbol (^) after the period of an abbreviation. This will give you the desired single space when using the formatter. (A period followed by no space will appear as just that.)

THE EXCLAMATION AND QUESTION MARKS — In these cases, the formatter will not automatically give two spaces as it properly should. To make your document look correct, you will need to add one space and one required space symbol.

THE PERIOD AND DECIMALS — the formatter thinks that any line that begins with a period is a formatter command and will delete the whole line. If by chance your document contains a value such as .10 and wraparound caused by the formatter's Fill and Adjust commands put it at the beginning of the line, the whole line will disappear. To correct this, you could put a zero in front of your decimals (0.10).

ASTERISKS AND NUMBERS — If you are printing out of the formatter and your document contains an asterisk followed by two or more numeric digits, the asterisk and two digits will disappear. For example, A*256 becomes A6. What is happening here is that TI-Writer misinterprets the asterisk and two digits as an instruction to input data from a value file, as when using mail-merge. To correct this problem you will need to type two asterisks followed by two dummy numbers, then the actual digits. For example, type A**25256, instead of A*256.

REQUIRED SPACE — If you tie words together for the purpose of underlining (&) or overstriking (@) with the required space (^), the Fill and Adjust commands of the formatter will leave gaping blanks in your lines. If you tie too many together, the line will extend beyond the right margin. It would be better to put a separate & or @ in front of each word. Be sure to include the spaces between the words. If you want a ^ to appear in your text, you will need to transliterate it. Of course, the @ and & characters are typed twice in succession to get them to print.

USER NOTES

Formatter and Funnelweb v.5.01

The following is by Matt Matthews and appeared in the newsletter of the SouthWest Ninety-Niners. — Ed.

If you have a file longer than one page to be printed out and plan to use the Funnelweb v5.01 Formatter, you just may get a form feed between your pages and waste a bunch of fan-fold paper.

You can get around this by doing all your work, including right justification, from the FW v5.01 Editor using control codes and the new FW command CTRL-R.

Turn on your printer and set it to the IBM CHAR Set No. 2 by typing this command first: CTRL-U FCTN-R CTRL-U t1 followed immediately by a carriage return. This should be on the first line.

Compose your file in wordwrap mode and then justify by going to the beginning of each paragraph. From there, press FCTN-2 then press CTRL-R. The Aussies have done a fine job in both cutting down on paper waste as well as time.

You may put your control codes in your document as needed to make it look as good as you want it to look.

The boys Down Under would enjoy hearing from you, especially if you include a few bucks to show how much you appreciate their hard and dedicated work: Tony and Will McGovern, 215 Grinsell St., Katora, NSW 2289, Australia.

Cleaning up

The following item was written by Michael O'Dowd and appeared in the newsletter of the 9T9 User's Group.—Ed.

I decided it was time to clean one of my old single sided drives to see would it perform any better and I did not want to use one of the disk cleaners.

One of my extra drives is in a power box so I removed the four case screws and the bottom screws which were holding the disk drive in the box. Knowing how one can forget where a part goes when it comes time to put things together again, I made a sketch and notes of everything I did, no matter how small.

All those resistors and chips looked very formidable, so taking the bull by the tail, I removed the power plug and put a dot with a marker pen on the male and female parts of the plug. The four pins are marked but a magnifying glass is required to see the numbers properly. The dot saves time and when replacing parts, .I also removed the 34-pin cable from the edge connector, the other end goes to the disk controller card.

With the disk drive removed from the box I then removed the connections plugged into the rear of the board. There are several of them and I sketched and marked them as well. Then the two screws holding the board were removed and the board slid out from under two tags, and I gazed at the unknown.

Remembering my sea-going days when I cleaned the gyro compass parts with carbon tet (dangerous stuff) I decided to use alcohol. I cleaned the edge connectors with a soft rubber eraser and set the board on some aluminium foil to protect it from static. I bought some alcohol and cleaned around the worm gear mechanism being careful not to disturb the head, I used cotton swabs and changed them frequently and cleaned the head with a new swab dipped in the alcohol. Some Teflon lubricant was applied on moving parts and I blew and sucked out dirt with a vacumn and put everything back together and bingo to my surprise the drive functioned on all cylinders.

CLASSIFIED

FOR SALE

FOR SALE

TI99/4A computer, w/loaded P-box, 2 internal and 1 external disk drives, 3 other TI computers and 1 extra RS-232 card, modem, recorder.
1 Star sg 10 printer, 1 speech synthesizer, 1 Navarone cartridge holder and various cartridges.
Light pen, 3 sets of joysticks, 1 trackball.
Numerous programs on disk.
Books: MICROpendium in binders from 1985 to 1997. Operating manuals, TI-Logo II, Editor/Assembler, TI-Writer, Word Processor, Microsoft Multiplan, TI Extended BASIC, Guide to 99/4A TI/Games, SG/10, Intro Assembly Language, Best of 99er, Kids and the TI and other material.
All for $300.00 plus shipping. If interested call 315-393-6383 or e-mail OVERHOMER@bbs.tsf.com. v14n4

FOR SALE

TI99/4A computer with peripheral expansion system, disk drive and controller, RS-232 panel & 32K memory. Extended BASIC and 7 other cartridges with games. — $100. John W. Lauter, 9724 Maplehill Dr., Dallas, TX 75238, 214-341-8879. v14n4

FOR SALE

Got PC99. Don't need my extra new backup console with 15 game cartridges $95, SuperSpace 11 E/A and 32K memory with centipede and defender on disk. $50 765-664-6001. v14n4

FOR SALE

Signalman Mark XII (1200 baud) modem, $7; Commodore 1702 composite color monitor (works with TI), $35. Call 512-255-1512, email jkoloen@io.com.

Devoted to the TI99/4A since 1984

Subscription Fees

❑ 6 issues, USA, **$35** ❑ 6 issues, Mexico, **$40.25**

❑ 6 issues, Canada **$42.50** ❑ 6 issues, other countries surface mail, **$40.00**

❑ 6 issues, other countries, air mail, **$52.00**

Outside U.S., pay via postal or international money order or credit card; personal checks from non-U.S. banks will be returned.

Address Changes

Subscribers who move may have the delivery of their most recent issue(s) delayed unless MICROpendium is notified six weeks in advance of address changes. Please include your old address as it appears on your mailing label when making an address change.

Check each item ordered (or list on separate page) and enter total amount here:_____

Check/MO ❑ MasterCard ❑ VISA ❑ (check one)

Card No. _____

Expiration Date _____

(Minimum credit card order is $9)

Signature _____

(Required on credit card orders.)

No sales tax on magazine subscriptions. Texas residents add 7.75% sales tax on other items, including back issues and disk subscriptions. Credit card orders add 5%.

Name _____

Address _____

City _____

State _____ **ZIP** _____

The set of numbers at the top of your mailing label indicates the cover date of the last issue of your subscription.

Disks, Etc.

❑ **Back Issues**,$3.50 each to March 1996, later $6 each. List issues: _____

No price breaks on sets of back issues. Free shipping USA. Add $1. single issues to Canada/Mexico. Other foreign shipping75 cents single issue surface, $2.80 airmail. Write for foreign shipping on multiple copies.

OUT OF STOCK: Vols. 1, No. 1-2; Vol. 2, No. 1

❑ **MICROpendium Index** (2 SSSD disks, 1984-1992), Extended BASIC required $6.00

❑ **MICROpendium Index II** (9 SSSD disks — 1 for each year — 1984-1992), XB required $30.00

❑ **MICROpendium Index II with MICROdex 99** (11 SSSD disks), XB required $35.00

❑ **MICROdex 99** (for use with MP Index II, 2 SSSD disks), XB required $10.00

❑ **MICROpendium Index II** annual disks ordered separately (1 disk per year, 1984-1992); each $6.00

MICROdex 99, by Bill Gaskill, is a collection of programs that allow users of MP Index II to modify their index entries, as well as add entries. MICROdex 99 supports many other functions, including file merging, deletion of purged records, record counting and file browsing.

GENEVE DISKS (SSSD unless specified)

❑ **MDOS 2.21** (req. DSSD or larger (for floppy & hard drive systems) $4.00

❑ **GPL 1.5** $4.00

❑ **Myarc Disk Manager 1.50**.....$4.00

❑ **Myarc BASIC 3.0**.....$4.00

❑ **MY-Word V1.21** $4.00

❑ **Menu 80** (specify floppy or hard disk versions(s); includes SETCOLR, SHOWCOLOR, FIND, XUTILS, REMIND.....$4.00

GENEVE PUBLIC DOMAIN DISKS

These disks consists of public domain programs available from bulletin boards. If ordering DSDD, specify whether Myarc or CorComp.

	SSSD	DSSD	DSDD
❑ **Series 1**	$9.00	$7.00	$5.00
❑ **Series 2**	$9.00	$7.00	$5.00
❑ **Series 3**	$9.00	$7.00	$5.00
❑ **Series 4**	$9.00	$7.00	$5.00
❑ **Series 5**	$9.00	$7.00	$5.00
❑ **Series 6**	$9.00	$7.00	$5.00

PERIODICAL