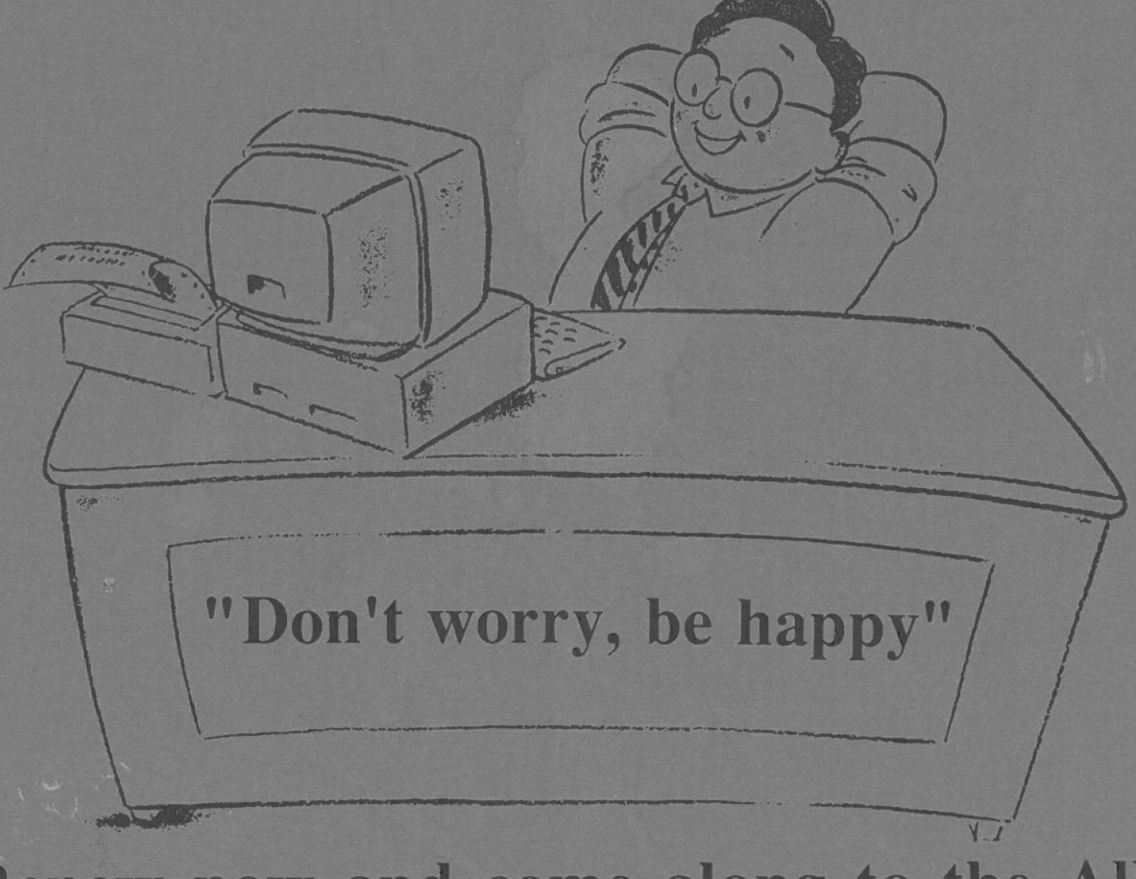
TISHUG NEWS DIGEST

Focusing on the TI99/4A Home Computer

Volume 10, Number 4 May, 1991

Registered by Australia Post - Publication No. NBH5933

It is our 10th Anniversary!



Renew now and come along to the All Day Tutorial to meet all your friends.

P.O. Box 214, Redfern, New South Wales, Australia, 2016 $3

TIsHUG (Australia) Ltd, based in
Sydney

TIsHUG News Digest

May 1991

All correspondence to:

P.O. Box 214
Redfern, NSW 2016
Australia

Membership and Subscriptions

Annual Family Dues	$30.00
Associate membership	$10.00
Overseas Airmail Dues	A$60.00
Overseas Surface Mail Dues	A$45.00

TIsHUG Sydney Meeting

The next meeting will start at 11.00 am on 4th of May at Ryde Infant School, Tucker Street, Ryde. A barbeque lunch and drinks will be provided at a small cost.

Printed by
The University of Wollongong
Printery

TIsHUG News Digest ISSN 0819-1984

Index

TIsHUG Fairware Author of the Month

The Fairware Author for this month is Clint Pulley for his c99 compiler and other utilities. c99 is a subset of the C language, which is used extensively in industry. It allows the power of assembler to be used in a high level language. The shop will have version REL4.0 of c99 for sale. All donations collected at the meeting and sent in will be mailed to him.

Subscriptions now due

Editor's Comment

by Bob Relyea

I finally thought that I had one of these buy, sway and sell days cracked! I left the house well and truly early and got to the venue in plenty of time, only to find numerous cars already parked out in front. When I got inside, however, there was little gear set out around the place like there usually is. Peter Schubert's hardware was the only exception. I looked in vain for the three hours that I was at the meeting but found little along the usual lines of gear for sale. Perhaps the holiday time had something to do with it? I did find it interesting to talk with several members who were asking me questions about Multiplan as I had decided to do a tutorial on it during the May meeting and had already written a short article (in this issue) to advertise the fact. That confirmed it! Hope to see you at the meeting where we can learn more about the TI and it's software. I think that the Page Pro experts are doing a great job of producing those great pictures. Keep it up boys. o

Co-ordinator's Report

by Dick Warburton

Due to unexpected technical difficulties, the Co-ordinator's report will not be in this issue. It should appear in next month's Newsletter.

Letter to the Editor

by Geoff Trott

I would like to put my views as Treasurer on the question of Life Members. I think that we need to consider carefully the consequences of making people Life Members of TIsHUG (Australia) Ltd. I do not wish to denigrate the existing Life Members in any way, or to say that other members should not be considered for Life Membership, but I just want to place some thoughts down for everyone to think about on this issue.

By definition, a Life Member is a member until he or she dies. That is perhaps the highest honour a club can bestow upon a member and is usually done after many years of service to the club given freely and without personal gain. In some organisations, a member can purchase Life Membership. It is expected that the club will last for many more years than the Life Member and it usually does not involve the club in any extra financial outlay (except for the purchased Life Membership).

Unfortunately, much as I hate to say it, our club has a limited life, which I imagine will be shorter than the life of our Life Members and each Life Member requires an expenditure by the club (at the present time) of about $30 per year for the magazine, envelopes and postage. Thus, at the moment there are five Life Members, which, when they were elected, composed about 2.5% of the total membership of the club. Now they form over 3% of the membership. If (or is it when) the membership falls to 50, they will be 10% of the membership and the paying members will have started to find them a financial burden, if a newsletter is still being produced. Of course, if more Life Members have been elected in the meantime, this situation will be worse.

There are things that can be done to help this potential problem, like not sending them a newsletter, which seems to be contrary to the spirit of having a Life Member. In fact I believe that it is rather ridiculous to have the category of Life Member in such a club as ours. There should be some other way of honouring members other than this Life Sentence, or perhaps the rights and privileges of Life Members need to be more carefully defined. In the end of the world scenario, can we ever wind up the club without treading on the rights of the Life Members? o

Treasurer's Report

by Geoff Trott

I have finally sent out cheques to those members who wanted their deposit paid on a monitor returned to them in cash. I am sorry for the delay but Percy's letter to me with the last of the names was lost by Australia Post, along with the bank statement for February. Usually the post is very reliable and we have not had any bad weather that I can recall. Oh well!

Our financial affairs are going well and will be further strengthened if everyone renews promptly. Look at the address label to see if your membership has expired and rush in your payment as soon as possible if it has.

Income for March	$1643.81
Payments in March	$3993.17
Excess of expenses over income for March	$2349.36

Multiplan and TI-Writer

At the Tutorial Day, by Bob Relyea

On the Tutorial Day on Saturday I intend on having my system available to give some tuition on the use of Multiplan and, if required, on Word-Processing.

I have had the Multiplan software for almost two years and until recently have used it just occasionally. Lately I have been 'flat out' doing some school work on it which has required that I become more proficient in certain areas. If you have been reading the Multiplan Tutorials by Herbert Schlesinger you have probably noticed have powerful the system is but, perhaps, intimidated by so much information. My tutorial will be geared to the level of those who attend. I will have some exercises ready to go through but will start with simple loading procedures and configuration. If you have a double disk drive/ramdisk set-up then I will show you how to configure Multiplan so that the work disk is other than DSK1 so you do not have to swap disks all the time.

Getting around the screen can be an initial problem if you have been used to Funnelweb as the key strokes are not the same. This can be overcome with some simple instruction. Cell formatting will be explained as well as all those options. Want to know how to turn your Multiplan spreadsheet into a DV/80 file for use with your word processor? It can be done with Multiplan itself inside of 60 seconds! What does 'Naming' involve and what can it be used for? And what about all those functions such as AVERAGE and STANDARD DEVIATION, how do they get used? Naming and using functions go hand-in-hand. Come to the tutorial day and find out how. We will also go over the building of a formula. There are at least two ways to do it so we will look at both.

I am sure that by the time the day arrives I will have thought of other things to include. I will be happy to go through anything that you want to the limits of my ability. I do not have all the answers but I am sure that will have a good learning experience.

See you there. Bob o

Auto-Loading Cassette Files

by D.N. Harris

The command RUN "CS1" can be used instead of END so that a tape of Extended BASIC programmes will load each other. EACH programme must have END changed to RUN "CS1". The effect is like RUN "LOAD" in a Disk collection but will not work unless used in the EXTENDED BASIC environment. Suppose the programme line is-

```
10000 END,      make it-        10000 RUN "CS1"
```

It starts a cassette load as soon as the programme ends. Thus you type RUN "CS1".

Then only one systems command needs to be taped, 'RUN "CS1" '. Remember to put the quotes around CS1, otherwise it is not valid device although for save CS1 you leave off the quotes, as well as for OLD CS1. For RUN "CS1" you must use the quotes both as a systems command and also as a statement in a programme.

As nobody seems to have noticed that an auto-loading Cassette operation is possible, I have taken the trouble to point it out. I hope this hint will be enthusiastically adopted and some export quality software tapes can be the result. It will work for TI BASIC running in the EXTENDED BASIC environment provided that the allowable range of Character Sets is employed and that the programme does not ask for "SPEECH", OUTPUT. Speech programmes should be kept separate on the other side of the tape, perhaps. o

Secretary's Notebook

by Terry Phillips

A fine and warm Saturday saw about 50 members at the April meeting attending the Buy/Swap/Sell day. It was noticeable on this occasion that not as much equipment was brought along for sale, and it also appeared, from my observations at least, that there was not a great deal of buying from what was available. Perhaps this is one more sign of the difficult economic times the country is currently going through.

Next months meeting promises to be a beauty with a whole host of activities planned throughout the day. It is, of course, a full day tutorial type meeting with a commencing time of 11am. Dick Warburton has organised BBQ facilities so you will be able to purchase some lunch and a drink for a nominal fee. Make sure you come along and get some good tuition from the willing band of instructors. I have been co-opted into giving a session on TI-BASE so I had better get the manual out and have a read before the big day.

A special demonstration of the BBS will be given at the May meeting. So if you are toying with the idea of signing up for the BBS then this will be a good opportunity to see it actually working.

If you like to borrow from the publications library then you will notice, in future, an improved service in making available local and overseas newsletter receipts. Previously the newsletters went via the Sysop, then the Editor and finally back to the library. Now they go direct to the library, with photo-copies doing the circuit. There is some good material in most of the newsletters we receive so you should check them out. Remember the publications library is available to ALL members at no cost.

Four new members have joined us over the past month and it is a big welcome to:

Gene Bohot - Chino California
Bryant Krause - Mira Loma California
Amalgamated Business Machines - Fyshwich, ACT
(presumably the Manager is the member)
Nirmal Hansra - Killara NSW

The former two joined after reading the article in Micropendium (that article by the way resulted in the signing up of 12 overseas members), while the latter two are as a result of console repairs through Percy Harrison.

As mentioned last month the majority of membership fees become due at the end of April. It was good to see 30 members making the commitment to stay with the group and paying their dues at the last meeting. My belief is that most will renew this year so that our membership base will stay around the the 180 mark.

May will mark the 10th anniversary of our group, it having been formed in May 1981. In those early days we first met at the home of our founder, Shane Andersen. By the way, where are you Shane? I have not heard from you or seen you for some time. Anyway, the numbers attending meetings soon got to be far too many to be accommodated at Shane's Home so we moved up to St John's Church Hall at Kings Cross where we stayed for a number of years prior to making the move to Woodstock Community Centre at Burwood. This latter venue is probably one of the most attractive in Sydney for group meetings but unfortunately we were not always able to get the number of rooms we required hence the decision was taken to move to the Ryde Infants School, our current "home".

Some highlights of the early days that still spring quickly to mind include:

1. The time when membership peaked at just over the 1000 mark. The committee had made a decision to purchase a daisy wheel printer when this number was reached so that a better quality newsdigest could be produced.

2. Several full day tutorials held at St. John's where members were encouraged to bring along and set up their computers. On at least two of these occasions I can well remember over 100 TI's all powered up - probably a world record and it must have played havoc with the electrical circuits in the hall.

3. Meeting days when it was not uncommon to sell over 150 tapes (not many members had disk drives in those days). The sad part about this was that most tapes were returned at the next meeting being unloadable. My how I hated that fast tape duplicator we purchased. My thoughts were that it would make a great anchor for a small boat.

4. The rapidness with which members made the commitment to expand their systems. By the end of 1986 tape sales had fallen to virtually zero and by now now we were selling over 100 copies of disk software at meetings.

5. The hardware projects locally developed by a great bunch of hardware gurus. This availability of cheap, home grown products has probably been the one single thing that has kept the club alive in its present healthy state over the past few years.

6. Lastly, but by no means least, the great bunch of people we have all become associated with by being members of this group. Some have left us for greener fields but the hard core element remains and importantly new members are still being recruited.

That is all for this month. I hope you can make it to the May meeting. o

TI-Writer File Keeping

by Paul E Scheidemantle, USA

To tell you the truth Once Upon A Time my Writer Files were a TOTAL DISASTER! I could not find anything without a long drawn out process. Sometimes spending hours going through all those disks really was a hassle. One day I told myself this has got to stop!

One of the first problems to overcome was FILE NAMES that were limited to 10 characters. So I decided that a menu program was in order. This way I could have file names that were as long as I wanted them to be. Also it would be great if all I had to do was change two letters of the MENU file name to load any file. So here is a simple method that I use for keeping my T.I. Writer Files from getting out of hand.

The 1st file of each disk will be called 00_WF_00. By making a menu file we can then have file descriptions longer than 10 characters because we read the menu instead of the disk directory. This is especially good if you have a TI Writer Loader other than the module.

We can save all the Menu files to a disk for quick searches.

Here are the easy steps to T.I. Writer File Keeping Mastery!!!

STEP 1

SETUP 2 DISKS AS FOLLOWS

1. File Disk #1 = FOR MENU FILES ONLY. Name this DISK ....

"WTRFILE_00"

We will save a copy of our menu's to this disk for easy access.

2. File Disk #2 = FIRST OF 99 FILE DISKS AVAILABLE. NAME THIS DISK ....

"WTRFILE_01

continued on page 31

TIsHUG Shop with Percy Harrison

Yes it is now available- the new EPROM RAM CARD AND KIT with the first two programmed Eproms sets - Funnelweb (3 Eproms) and TI Artist Plus (2 Eproms). For prices see the listing below, but there is a catch, as these programs are Commercial material they can only be sold to those members who can provide proof that they have already bought and paid the commercial price for the programs on disk. Alternatively they can purchase from the club the original commercial software, at the clubs commercial price, and then buy the programmed Eproms in which case they will receive both the Eprom set and the commercial disks and instruction manual. Additionally for every four Eproms (or part thereof) it will be necessary to purchase one 74LS08 IC. Finally to configure your Ramdisk you will need the latest version of ROS, the price of this is $12.00 and will be added on to the price that you pay for the first set of programmed Eproms that you purchase. Some members may have already purchased this ROS program at the club software price of $2.00 in which case they will be required to pay the shop the extra $10.00 as the shop inadvertently sold this programme at $2.00 not realising that it was Commercial Ware. Very sorry if this is confusing but hopefully we will get it sorted out without too many traumas.

PRICE LIST

5.25 in. DSDD Disks (Boxes of ten)	$6.50
5.25 in. HD Disks (Boxes of ten)	$13.00
3.5 in. DSDD Disks (Boxes of ten)	$12.00
3.5 in. SSDD Disk Drives	$20.00
5.25 in. DSDD Half Height Drive (New)	$65.00
5.25 in. DSDD Half Height Drive (Used)	$50.00
12 Volt AC Transformer	$3.50
13 Volt Arlec Transformer	$12.00
8.5, 17 Volt Transformer	$25.00
60 VA Transformer	$20.00
MFC Printed Circuit Board	$30.00
MFC Kit (Disk Controller)	$102.50
32K Kit for MFC	$26.50
PIO/RS232 (single port) Kit for MFC	$42.50
Combined 32K and PIO/RS232 Kit	$60.00
Music Kit with PCB	$65.00
32K Memory PC Board	$7.00
Eprom Ram PC Board	$45.00
Eprom Ramdisk Basic Kit	$35.00
Funnelweb Eprom Set (3 Eproms)	$36.00
TI Artist Eprom Set (2 Eproms)	$24.00
ROS Version 8.14	$12.00

(N.B. Must be purchased with first Eprom set)

COMMERCIAL SOFTWARE

Artoons SSSD	$12.00
Character Set Graphic Design Cataloguer	$6.00
Character Set Graphic Design I	$12.00
Character Set Graphic Design II	$10.00
Character Set Graphic Design III	$14.00
Display Master	$15.00
Genial Traveler (SSSD)	$6.00
Microdex I (SSSD)	$16.00
Microdex II (SSSD)	$11.00
Nuts and Bolts #1 (DSSD)	$6.00
Nuts and Bolts #1 (SSSD)	$7.00
Page Pro 99 version 1.6	$28.00
Page Pro Utilities	$17.00
Page Pro Applications #1	$2.00
Page Pro Templates (Vols 1 to 12)	$8.00/vol
Picasso Publisher Version 2.0	$14.00
Picasso Publisher Support Disks	$6.00
Picasso Applications Disk	$2.00
Rockrunner (SSSD)	$15.00
Spell It! (DSDD version)	$24.00
Spell It! (SSSD version)	$27.00
The Missing Link (TML)	$28.00
The Missing Link Companion Disk	$2.00
TI Artist Plus	$25.00
TI Base Vers 3.01 (SSSD)	$25.00
TI Sort SSSD	$15.00

Every month I seem to have some disappointing news and unfortunately this month is no exception as I regret to report that Australia Post has jacked up the price of postage on packages and have actually withdrawn the special rates they previously allowed for Disk and Video packs and the like so please refer to the new postage rate below before sending off the cheque for your mail order requirements.

Packaging and postage charges:

	Surface	Airmail
1 to 9 Disks	$3.00	3.70
10 to 20 Disks	$4.00	$4.90
TI Artist Plus	$3.00	$3.70
Display Master	$3.00	$3.70
TI Base	$3.00	$3.70
TI Sort	$3.00	$3.70
5.25 inch half-height drive (1.25 Kg)	refer to your local post office	

Typewriter 99

reviewed by Lou Amadio

The review program for this month TYPEWRITER.

TYPEWRITER, written by Jim Reiss and distributed by Asgard, is a nifty little program. It comes in both module and disk file format, although I only tried the module version. TYPEWRITER turns your TI-99/4A into an electronic typewriter and has two basic modes: LINE and TYPE.

Of the two modes, LINE is the more powerful as it offers text editing features. Nevertheless, I found the TYPE mode to be rather interesting as each character was immediately printed out on typing. This resulted in rapid output to the printer, but it means that you have to be a good typist as there is no room for error. The LINE mode is perhaps more practical and the one that most will use.

TYPEWRITER is easy and fast to use and is a practical alternative even if you have a disk system and TI-Writer.

The program supports both serial and parallel printers. The type and mode of printer is selected on first entering the program. Once set up for your printer, it is very easy to type out a short letter or note. The only real drawback with this program is that there is no "save" option. The program, however, serves its primary purpose well, that is it emulates an electronic typewriter.

The module comes with six pages of documentation. Everything that you need to know is explained in simple terms and is easy to refer to.

Initial setup includes the Left and Right margins as well as Tab settings for tabular information. An interesting feature is the margins line shows the full 80 columns available, including the cursor position, even though the screen operates in 40 column mode.

Some of the editing functions from TI-Writer are also available, for example: Insert, Delete and Window over. Window allows you to see any text which exceeds the normal 40 column screen width. During typing, Window operates automatically and, if in Word Wrap mode, any word which does not fit on one line is automatically transferred to the next line. There is a maximum of six lines on display at any one time. This allows a small buffer, although editing is limited to the current line. Pressing <ENTER> will send the line to the printer.

Other options include: Keyclick, Centre, Justify, Bold, Underline, Spacing and Next Tab, Screen Colours, Left and Right Margins.

continued on page 6

Techo Time

with Geoff Trott

My appeal in the last TND for the floppy disk controller chip from Western Digital (WD1773) caused a response from South Australia and Sydney. I thank all who responded, but particularly John Paine, who actually had one spare. It seems he had trouble obtaining this IC some years ago and eventually bought 5 from CorComp in USA and had one left. So now the Micro-expansion system is working again. I would still like to get hold of some spares, if anyone from the USA could help.

At the next meeting, the all day tutorial, I am planning to give a talk on the hardware, trying to explain how computers work with examples from our computer. I shall try and start from the very basics so that anyone with an enquiring mind can follow what I will be saing. I hope that people who venture into the talk will ask questions until they do understand what is going on. We held a session a few weeks ago with Dick, Lou and Alf which showed me some of the problems people have in trying to understand the operation of the hardware.

I would now like to talk about floppy disks for a while. This will probably take more room than is available in this issue, as I want to tell you about how data is stored on floppy disks and what interlace means. I have done some experiments with disks formatted with different interlace patterns and measured the time taken to perform various disk functions with particular interlace patterns. In the process I have also exercised various disk manager programs and the results are quite interesting. I have not completed all the tests but if I do not start writing you shall never read any of the results.

Formatting of Floppy Disks

Floppy disks spin at 300 rpm. This means that each revolution takes 1/5 second or there are 5 revolutions per second. The data is stored on floppy disks in circular tracks. The normal floppy disk has 40 tracks. Each track is completely circular so that to move from one track to another the read head must be moved the correct distance towards the centre of the disk or away from the centre of the disk. In most systems, the first track is at the outside of the disk and the last track is closest to the centre of the disk. For double sided disk drives, there is one read head on the bottom of the disk and a second head on the top of the disk and they move together. In this case, the first track is on the outside of the bottom of the disk with the next 39 tracks also on the bottom moving towards the centre of the disk. Track 41 is at the centre on the top of the disk with the last track on the outside and on the top of the disk. This provides compatibility between single sided and double sided drives, in that a single sided drive can at least read the first half of a double sided disk.

Now each track is divided up into regions called sectors (arcs would have been a better choice of words). For our system, single density uses 9 sectors in each track while double density uses 18 sectors per track (with 16 sectors per track possible with Myarc controllers only). This difference is obtained by changing the type of modulation used to record the data and is too technical to get into here (even if I knew the details!). It is sufficient to say here (I hope), that both single density and double density use up all the space on the track for the sectors they use. That is, the 9 sectors in single density are spread evenly around the track with only a small gap between each one as are the 18 sectors in double density.

So the disk is set up into tracks, the tracks are divided up into sectors and the data are stored in the sectors. In our case, 256 bytes are stored in each sector in serial form, one bit at a time. It would be nice if we could read or write one single bit or even one byte at a time. This is not possible as it is too difficult to locate such a small item of data as the disk keeps moving around. To re-write data back to an existing spot on the disk is like trying to put together an audio tape from various sources without stopping the tape moving. Something is bound to be over-written. So each sector is surrounded by some space filled with special characters for synchronization of the bits into bytes (another problem) and both reading and writing is done one sector at a time (minimum transfer). This allows a sector to be written without upsetting the ones around it. Formatting the disk writes on all the tracks dummy or empty sectors with all their surrounding special characters. Part of these special characters contain the actual address of the sector which enables the sector to be identified while other bytes are used for an error checking code.

At this point writer's block has come over me and I am having trouble thinking how to continue. Let me try and summarize what I have tried to say so far. Data are stored on floppy disk in files. Files are broken up into pieces that will fit in sectors of 256 bytes. Sectors are spaced evenly in circular tracks on the disk. There are usually 40 tracks on the side of a disk. Depending on the particular disk drive, both sides of a disk can be used for storage. Depending on the particular disk controller, either 9 (single density) or 18 (double density, 16 is also possible with Myarc disk controllers in double density) sectors are laid out on each track. Each sector is surrounded by other bytes (not part of the data) which are used for synchronization, error checking and also for the sector address. Now I want to talk about where on the track sectors are put.

Near the centre of the disk there is a hole punched in the disk with holes in the jacket to enable this hole to be used. In the disk drive is an opto-coupler sensor which will give out a pulse once each revolution as long as the disk itself is turning. This signal is called the index signal and can be used for many things like determining whether there is a disk in the drive correctly loaded. It is also used to give a reference point for the sectors to be laid out on the disk. The disk controller waits for the index signal and then starts looking at the data on the disk for the first sector information. Or when formatting the disk, the first sector for that track is written after the index signal is found. In fact, all the data for a track, including all the synchronization, error codes and address bytes are set out in a buffer and written out in one hit for the whole track when formatting a disk. This makes the actual formatting quite fast as each track is written in a single revolution of the disk. We can work out how long that will take. For single sided disks, there are 40 tracks which should take 40 revolutions of the disk at 5 revolutions a second making 8 seconds no matter what density is used. In practice, it takes 17 seconds, which looks like 2 revolutions per track. This makes sense, as it will take a bit of time after writing one track to set up the data for the next track and move to the next track, by which time the index hole would have passed by and so it would have to wait one revolution for the index hole to come around again. The time taken to move between adjacent tracks is the head step time and is at most 0.02 seconds, which will be significant when timing is very tight, especially with double density.

So, formatting a disk is done quickly no matter what the format used, but using the disk depends much more on the actual pattern used for the order of the sectors. Consider the usual next step in the format process, verification of the disk. This is the process of reading each sector, in sector address order and checking that the error codes indicate that no error was found in the data. This checking is done by the disk controller but it requires data to be read into a buffer one sector at a time. Consider what would happen with sectors placed one after the other on the track in sequential address order (0, 1, 2, 3, 4, 5, 6, 7, 8 for single density). Sector 0 is read, the controller is checked for errors and the sector number is written to the screen. By that time the start of the next sector has passed under the read head and the program has to wait one revolution for sector 1 to be read. This means that it will take one revolution to read each sector which means it will take 360/5 = 72 seconds to verify all the sectors on a single sided single density disk (288 seconds for DSDD disk). Now if the sectors were placed on the disk such that there was a gap before the next sequential sector appeared, there might be time to do all the processing required and to read it without a wait. For example, the order (0, 5, 1, 6, 2, 7, 3, 8, 4) has consecutive sectors separated by one sector which gives 1/45 second (1/90 for double density) for the processing of the data and to get back for the next sector. This is called an interlace of the sectors and this example is an interlace of 2. It is possible to have an interlace from 1 to one less than the number of sectors on the track. Here are the possiblities for single density

Interlace	Sector Pattern
1	0, 1, 2, 3, 4, 5, 6, 7, 8
2	0, 5, 1, 6, 2, 7, 3, 8, 4
3	0, 3, 6, 1, 4, 7, 2, 5, 8
4	0, 7, 5, 3, 1, 8, 6, 4, 2
5	0, 2, 4, 6, 8, 1, 3, 5, 7
6	0, 3, 6, 2, 5, 8, 1, 4, 7
7	0, 4, 8, 3, 7, 2, 6, 1, 5
8	0, 8, 7, 6, 5, 4, 3, 2, 1

As shown in the table above, the interlace number tells us how far apart consecutive sectors are (remember that the track is circular so that the end of the table is next to the start). It also tell us how many rotations are required to read all the sectors in consecutive order. So the higher the interlace number the slower the verification time. However, if too low a number is used so that there is not enough time to get ready to read the next sector, another revolution is required. The other thing to note is that if the interlace number has a factor in common with the total number of sectors, the interlace is not constant between all sectors. Look at interlace 3. There are 2 sectors between most consecutive sectors except between sectors 2 - 3 and 5 - 6 which have 3 sectors. The same thing applies to interlace 6. This does not pose a problem as there is a bit more time available rather than less.

The time of verification can be worked out for each interlace pattern, assuming no extra revolutions required. These times can then be checked by experiment. This is a simple formula, as one revolution over each track of a 40 track disk will take 8 seconds. The number of revolutions required to read all sectors on a track in their sequential order is the interlace number. So the verify time for a single sided disk (single or double density) should be 8 times the interlace number in seconds. An interlace of 4 will take 32 seconds to verify the disk while an interlace of 7 will take 56 seconds. If the actual verify time is longer than this, it will be because the processing has taken too long and an extra revolution is required.

The other thing to ponder, is what happens when moving from the last sector of this track to the first sector in the next track. This involves moving the head as well as all the rest of the things that must be done so that at least the same amount of time is needed so that the same or more gap is needed. Interlace 1 is no problem as there is the same gap between sector 8 on one track and sector 0 on the next (no real gap). Interlace 2 also follows its pattern to the next sector. Interlace 3 has no gap to the next track so we would expect an extra revolution delay here (8 seconds for one side) if interlace 3 gave the fastest verify. Interlace 4, interlace 5, interlace 7 and interlace 8 all follow their patterns onto the next track. Interlace 6 has an interlace 4 gap to the next track. This variation in interlace has not been addressed in any special way in any of the disk managers, but there is one copy program I have read about (MCOPY is it?) which allows the effective interlace between tracks to be adjusted (they call it the skew). This may allow a smaller interlace to be used by giving more time for the movement between tracks. I will not dwell on this aspect as it is not readily available. It is clear (from the single density case) that an interlace number that shares a common factor with the number of sectors in the track is probably not going to give the best result, as there is usually a smaller gap when moving to the next track. This means that for double density with 18 sectors per track, interlace numbers 2, 3, 4, 6, 8, 9, 10, 12, 14, 15, 16 are going to have the potential for worse than normal skew problems.

I shall conclude the first part of this talk with a table of the double density interlaces as they should be, followed by tables of interlace patterns produced by the CorComp/AT/MiniPE disk controller using the CorComp disk manager. See if you can find any sensible interlace patterns amongst the CorComp patterns!

No.	Sector pattern
1	0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17
2	0,9,1,10,2,11,3,12,4,13,5,14,6,15,7,16,8,17
3	0,6,12,1,7,13,2,8,14,3,9,15,4,10,16,5,11,17
4	0,9,5,14,1,10,6,15,2,11,7,16,3,12,8,17,4,13
5	0,11,4,15,8,1,12,5,16,9,2,13,6,17,10,3,14,7
6	0,3,6,9,12,15,1,4,7,10,13,16,2,5,8,11,14,17
7	0,13,8,3,16,11,6,1,14,9,4,17,12,7,2,15,10,5
8	0,9,7,16,5,14,3,12,1,10,8,17,6,15,4,13,2,11
9	0,2,4,6,8,10,12,14,16,1,3,5,7,9,11,13,15,17
10	0,9,2,11,4,13,6,15,8,17,1,10,3,12,5,14,7,16
11	0,5,10,15,2,7,12,17,4,9,14,1,6,11,16,3,8,13
12	0,3,6,9,12,15,2,5,8,11,14,17,1,4,7,10,13,16
13	0,7,14,3,10,17,6,13,2,9,16,5,12,1,8,15,4,11
14	0,9,4,13,8,17,3,12,7,16,2,11,6,15,1,10,5,14
15	0,6,12,5,11,17,4,10,16,3,9,15,2,8,14,1,7,13
16	0,9,8,17,7,16,6,15,5,14,4,13,3,12,2,11,1,10
17	0,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1

CorComp Interlaces (from the disk controller manual)
Single Density

No.	Sector pattern
1	0, 1, 2, 3, 4, 5, 6, 7, 8
2	0, 2, 4, 6, 8, 1, 3, 5, 7
3	0, 3, 6, 1, 4, 7, 2, 5, 8
4	0, 4, 8, 3, 7, 2, 6, 1, 5
5	0, 5, 1, 6, 2, 7, 3, 8, 4
6	0, 6, 3, 1, 7, 4, 2, 8, 5
7	0, 7, 5, 3, 1, 8, 6, 4, 2
8	0, 6, 3, 8, 5, 2, 7, 4, 1
9	0, 5, 1, 6, 2, 7, 3, 8, 4
10	0, 4, 8, 3, 7, 2, 6, 1, 5

Double Density

No.	Sector pattern
1	0,2,4,6,8,10,12,14,16,1,3,5,7,9,11,13,15,17
2	0,3,6,9,12,15,1,4,7,10,13,16,2,5,8,11,14,17
3	0,4,8,12,16,2,6,10,14,1,5,9,13,17,3,7,11,15
4	0,5,10,15,2,7,12,17,4,9,14,1,6,11,16,3,8,13
5	0,6,12,1,7,13,2,8,14,3,9,15,4,10,16,5,11,17
6	0,7,14,3,10,17,6,13,2,9,16,5,12,1,8,15,4,11
7	0,8,16,6,14,4,12,2,10,1,9,17,7,15,5,13,3,11
8	0,9,1,10,2,11,3,12,4,13,5,14,6,15,7,16,8,17
9	0,10,2,12,4,14,6,16,8,1,11,3,13,5,15,7,17,9
10	0,11,4,15,8,1,12,5,16,9,2,13,6,17,10,3,14,7
11	0,12,6,1,13,7,2,14,8,3,15,9,4,16,10,5,17,11

o

continued from page 4

Keyclick creates a sound when a key is pressed.

Centre allows titles to be centered within the chosen margins.

Justify aligns the right margin on a line by line basis.

Bold and Underline work as expected (underlining is continuous). These operations can be used individually or mixed.

Spacing allows printed output to be double or triple spaced for drafting purposes.

Overall, I like this program for its simplicity, features and ease of use. It is a must for "console only" users for quick and easy output to a printer. o

Assembly and Trains

at the Tutorial Day, by Ross Mudie

The next assembly class will be at the TISHUG tutorial day on Saturday 4th May 1991. Remember that the meeting on this day will start at 11am. I will be commencing my assembly tutorial at about 1pm to allow some people who cannot attend in the morning to be present (this is a change from the previous arrangements). The small HO gauge model train set controlled via a TI99/4A computer will be set up at the tutorial day for everyone to look at, discuss and operate. The train set is controlled by an assembly language program which is implanted inside an extended basic program to make it easy to load from a cassette tape recorder. Another model train set is being discussed as a club project.

<u>LINK-IT 20</u> by Ross Mudie, 6th April 1991

There is still some confusion over the process of creating a linked assembly language program. I hope that this short article may help to ease some of that confusion. The following file is based on questions that I received at the TISHUG meeting on 6th April 1991.

Texas Instruments in the design of their powerful Extended Basic provided a way for extended basic to load and transfer program execution to an assembly language program. The LINKED assembly language program is supported by a number of utility programs from within the extended basic module. The use of these support utilities is similar, but not the same as pure assembly or assembly linked from TI BASIC. Unfortunately the documentation of the linking to assembly was incomplete, probably due to the timing of Texas Instruments' decision to cease manufacture of the TI99/4A.

The support utilities are made available by using CALL INIT in an extended basic program. This copies the support programs from extended basic to the low area of the expansion RAM, from hex 2000 to 24F3, with the linking DEF table at the top of the low RAM from hex 3FFF down. The DEF table provides a list of the program entry names and the actual start address. These names are the ones used in the CALL LINK which is the way of getting from extended basic to assembly.

The assembly program is easiest written as a source file using the TI TMS9900 Editor/Assembler programs or the Editor/Assembler in Funnelweb. The SOURCE file is created using the EDITOR program and must be saved to disk as a DISPLAY VARIABLE 80 file. Once a source file has been written and saved to disk the ASSEMBLER program can be run and this reads the instructions contained in the SOURCE file to create the OBJECT file.

The OBJECT file, once sucessfully created and debugged, will contain the machine language codes to perform the desired tasks in machine code at the maximum possible speed for the computer.

The OBJECT file is still however just a disk file and must be loaded into the appropriate area of memory and then be linked into to allow the program functions to be performed. This is where extended basic can come come into the picture because it can LOAD and LINK from extended basic to assembly. In the extended basic program, CALL INIT is required, FIRST to set up the necessary operating environment.

The assembly OBJECT file must then be loaded from disk, eg, CALL LOAD("DSK1.OBJECT"). I often use the file names DSK1.S for my source file and DSK1.O for my object file, but you can use any valid file name of one to ten characters after the device description (that is the "DSK1." part).

Once the object file is loaded the extended basic program can link to the assembly program with a CALL LINK. Eg, CALL LINK("ASPROG") could be valid as the link name can be from one to six characters in length. CALL LINK also provides for up to 16 parameters to be passed to or from the assembly program. These can be numeric values, simple numeric variables, numeric arrays, strings, string variables or string arrays. These parameters are called "ARGUMENTS" and are included in the CALL LINK after the LINK name. CALL LINK("ASPROG",34,N,H(),"String",S$,SA$()) could be a valid call link. The assembly program must be written to provide the appropriate handling of the each parameter and this is very versatile within the assembly programming environment.

There are some limitations in the assembly loader of extended basic. It is only designed to load object files which are UNCOMPRESSED and for relocatable code the size is limited to that which will fit in the LOW RAM from hex 2000 to 3FFF. This is 6924 bytes of memory space including the DEF table area and is not used by the non-assembly functions of extended basic.

When assembling your machine code the assembler asks questions about the file names and other options that you want to use for the assembly process. Here are typical questions, answers and reasons.

Question	Typical Answer	Comment
SOURCE FILE?	DSK1.S or DSK1.SOURCE	The source and object file names describe the device and the file
OBJECT FILE?	DSK1.O or DSK1.OBJECT or any other file name that you like.	name on the device. Choose your own file names within the rules.
LIST DEVICE?	PIO. or DSK1.LIST or just press ENTER if not required.	Again a valid device plus file name as required. Choose the file names so that they are easy to type. The "." is essential for PIO usage.
OPTIONS?	RLST or R	R is required if your source file uses R followed by a number for registers. L is required if a LIST DEVICE is specified, otherwise no list. S produces a symbol table in the listing. T fully spells out Text in the listing and can use lots of paper.

C is not a valid option for assembly linked from extended basic as the extended basic loader can not load compressed code.

In the source file if a * appears in the first column it is a remark (just like ! in extended basic) and the whole line is ignored by the assembler.

When assembling, the assembler will issue error messages for anything that it cannot understand (because it is invalid) or improperly constructed. You must note the errors then rectify the problems in the SOURCE file, resave the source file then try to assemble it again. The assembler can not check for errors in the way you write the program, it only checks for correct use of the language. Thus once you have managed to assemble the program without errors you still have a second hurdle to jump, that of making the program work as you intend it.

I hope that this further assists those who are having problems coming to grips with assembly. I can assure you that it is not an easy path, I learnt it the hard way too! There is only one way to achieve success with assembly and that is LOTS of CONSISTENT HARD WORK.

BBS users, please remember that I can help you with assembly problems between meetings, just upload your source file as mail to SYSOP with a description of the problem. o

TML Graphics Programs

part 2, compiled by Stephen Shaw, England

An earlier program printed patterns of circles using a simple modulus method of deciding if a pixel was on or not. The following program can produce much more complex patterns but again uses a modulus decision making method. The second simpler form uses variable values which produce a very regular pleasing pattern.

This program is to experiment with. Instead of using SIN(X)+SIN(Y) try other formulae! And let me have any good patterns - formulae and variable values. By all means set R higher, it just takes longer! And you can plot "to the right" or "to the bottom" by using say FOR I=50 TO R+50 and then when LINKing to PIXEL instead of I use I-49 for the plot. I especially like the patterns when B1=-5, B2=-10, G=66, A=9, and M=2.

What happens to a pattern when you halve or double the value of G and leave other variables unchanged? To see how amending G or M varies the pattern, try say putting in a new line:

```
205 A=INT(I/12)+2 or
205 M=INT(I/20)+2
```

Have fun, experiment, and share your discoveries! If you do not have the Missing Link (commercial) program, these programs are easily adopted for ANY program that allows BIT MAP (PIXEL) plotting!

```
95 ! PATTERN MAKING PROGRAM IN TI XB REQUIRES THE
   MISSING LINK.Can be re-written for other bit map
   languages/utilities.
96 ! from egg tile generator page 242 of Computers Chaos
   Pattern and Beauty by Clifford A Pickover.
97 ! for TI by S. Shaw 1990
98 !
100 RANDOMIZE
109 ! R= number of pixels down and across of pic.
110 R=110
118 ! Do not have to use INT function except for M.
119 ! b1,b2=phase shift of sine wave, minor adjustments
120 B1=INT(RND*5)+6*-1
130 B2=INT(RND*9)+12*-1
139 ! g= frequency of sine wave, main pattern
    determinator.
140 G=INT(RND140)+60
149 ! a=degree of disorder, low a=good order
150 A=INT(RND*6)+4
159 ! m=modulus, degree of pattern density- m low=more
    pixels on. Also affects pattern.
160 M=INT(RND*2)+2
170 CALL LINK("PRINT",20,160,STR$(B1)):: CALL LINK("PRIN
    T",40,160,STR$(B2)):: CALL LINK("PRINT",60,160,STR$
    (G))
180 CALL LINK("PRINT",80,160,STR$(A))::CALL LINK("PRINT"
    ,100,160,STR$(M))
190 REM
200 FOR I=1 TO R :: FOR J=1 TO R
210 X=B1+(G*I)
220 Y=B2+(G*J)
230 Z=A*(SIN(X)+SIN(Y))
240 C=INT(Z)
250 IF C/M=INT(C/M)THEN CALL LINK("PIXEL",I+10,J+10)
260 NEXT J :: NEXT I
270 CALL LINK("PRINT",172,20,"ANY KEY")
280 CALL KEY(5,Z,X):: IF X<1 THEN 280
290 GOTO 100
```

Or in a much simpler form:

```
110 R=110
120 B1=-6
130 B2=6
140 G=314
150 A=5
160 M=3
170 !
180 !
190 !
200 FOR I=1 TO R :: FOR J=1 TO R
210 X=B1+(G*I)
220 Y=B2+(G*J)
230 Z=A*(SIN(X)+SIN(Y))
240 C=INT(Z)
250 IF C/M=INT(C/M)THEN CALL LINK("PIXEL",I+10,J+10)
260 NEXT J :: NEXT I
270 CALL LINK("PRINT",172,20,"ANY KEY")
280 CALL KEY(5,Z,X):: IF X<1
THEN 280
290 GOTO 100
```

This next program is definitely fractal in nature but also fairly short, indeed I will first present it as a one liner - this requires THE MISSING LINK but can be translated easily to any other language giving pixel graphics:

```
     10 FOR X=1 TO 170 :: FOR Y=1 TO 180 ::
T=((X/32) 2)*Y/32 :: IF T AND 1 THEN CALL
LINK("PIXEL",X,Y) :: NEXT Y :: NEXT X :: GOTO 10
```

The final GOTO 10 is just to hold the image on screen. Instead of AND 1 try AND 2 or AND 4 or AND 8 - we are using the XB AND operator not as the usual LOGIC operator, but as a BINARY operator, which works ONLY on numbers up to 32700 odd. The /32 part is just to give us a reasonable pattern, by magnifying the image for any particular set of screen widths/depths- it keeps the numbers below 2^16 exceeding which will give you a numeric overflow.

Here is a more refined form of the program:

```
90 CALL LINK("CLEAR"):: CALL LINK("PRINT",1,1,"TOP")
100 CALL LINK("INPUT",1,31,TOP$,4,"1"):: TOP=VAL(TOP$)
105 CALL LINK("PRINT",11,1,"BOTTOM:")
110 CALL LINK("INPUT",11,61,BTM$,4,"200"):: BOTTOM=VAL
    (BTM$)
115 CALL LINK("PRINT",21,1,"LEFT")
120 CALL LINK("INPUT",21,35,L$,5,"1"):: LEFT=VAL(L$)
125 CALL LINK("PRINT",31,1,"RIGHT:")
130 CALL LINK("INPUT",31,51,R$,5,"200"):: RIGHT=VAL(R$)
140 VS=180/(BOTTOM-TOP):: HS=200/(RIGHT-LEFT)
150 CALL LINK("CLEAR")
160 FOR X=TOP TO BOTTOM STEP 1/VS :: FOR Y=LEFT TO RIGHT
    STEP 1/HS :: TEMP=INT(( (X/32) 2)*Y/32)
170 IF TEMP/2<>INT(TEMP/2)THEN CALL LINK("PIXEL",(X-TOP
    +1)*VS,(Y-LEFT+1)*HS)
180 NEXT Y :: NEXT X
190 CALL KEY(5,A,B):: IF B<1 THEN 190 ELSE CALL LINK("
CLEAR"):: RUN
```

In this version- still not too long is it? - we are checking whether the integer of a number is odd- this is the same as AND 1. This allows us to use numbers well over 2^16!!! The refined form allows you to input the number for the top, bottom left and right row, then scales the display to 180 x 200. The display is not very satisfactory for widths/depths of under 20. At a width/depth of 3000 the pattern is decidedly chaotic, using smaller widths you will find that the pattern is almost repetitive and grows smaller as the numbers mount. The very definition of a fractal pattern. ENJOY! o

continued from page 30

AT LEAST 116K of Assembly code. This still leaves ALL of memory expansion free plus the 16K of cartridge RAM free for other things or for Assembly routines for your GPL programs to link to (another 48K). That gives us a TOTAL program space of 106K plus 16K of VDP Ram for a total of 122K (128K with the Operating System area). Also with GPL you can EXPAND or modify existing Modules. And, last but certainly not least, GPL is the controlling language for our 4As, so now you make it do most anything you want! Start thinking about those changes you have wanted to make for the last 6 years, your chance is coming!!!

(Hmmmmm, let us see, a cold boot of an Assembly, GPL, Forth, XB or Basic program that would ....) o

TI-Bits Number 5

by Jim Swedlow, CA USA

[This article originally appeared in the User Group of Orange County, California ROM]

DISPLAY VARIABLE 80 FILES-MULTIPLAN AND TI WRITER

The DV80 file is TI'S workhorse. TI Writer uses this format. If you open a file without specifying a type <OPEN #1:"DSK1.MYFILE">, it will be DV80. Assembly language source code files are DV80. This month we will cover some interesting aspects of these files as they are used by TI Writer and Multiplan.

First, you can save a Multiplan spreadsheet as a DV80 file on disk. Then, later, you can use that DV80 file for printing or for merging into a TI Writer file. You choose Print and then File. You must be careful to use a different file name than the one you used to save your spreadsheet as, unlike Transfer Save, Multiplan does not warn you if you are about to overwrite an existing file.

Just as when printing on a printer, you can control the margins and page format with Print Margin. One of the items that Print Margin lets you set is Print Width. If you set this to a number greater than 80, Multiplan will write a DV80 file wherein each record is longer than 80 characters.

Should you attempt to read such a file with a BASIC program, your system will produce a strange error code and lock up. Apparently the folks at TI thought that a DV80 file could not have a record longer than 80 characters so their error handling language does not consider that possibility.

TI Writer, however, will read this illegal file. It will only input the first 80 characters in each record but it is just about the only way to access the file (another is a disk sector editor).

Incidentally, TI Writer is very forgiving when reading files. More than once I have used TI Writer on a file with a glitch that prevented me from reading it. First I loaded the file into the Text Editor and then I saved the file back to disk. This process can remove a glitch.

QUOTES OF THE MONTH

"For those who like this kind of a book, this is the kind of a book they will like".

---A book review by A. Lincoln

"Knowledge comes but wisdom lingers."

---Tennyson

TWO TI WRITER TIPS

The Formatter makes sure that you have two spaces after each period. This can cause such strange things as: Mr. Smith
1023 N. Fargo Street

These extra spaces jump off the page to the reader as simply wrong. The easiest way I have found to solve this is to use the ^ sign to control the spacing. Mr.^Smith will print with just one space as will 1023 N.^Fargo Street.

The other tip concerns writing a on-disk note using the Editor. You might be writing some documentation or a disk-letter. If you save your final version to disk using Save File, the last record will contain the margin and tab information and will cause the final print line to have strange characters. The solution is to save your final version using Print File. Just enter the Disk File name and your on disk file will have only your text.

WORD OF THE MONTH

AVATAR - the descent of a deity to earth in an incarnate form; the incarnation of a god; any embodiment or manifestation of an abstract quality, attitude, concept or principle in a person.

"[His] piano teacher . . . was reportedly an avatar of the romantic giants of the 19th century."

A TI RESOURCE

Looking for a program to do something but you cannot find one that meets you needs? The Amnion Helpline Free Access Library is one of the largest public domain collections for the TI. To quote: "send me a note telling me as specifically as possible what you want. . . . I will go through the library and extract the programs that seem to fit your needs and send them to you. Naturally, the more specific the request, the better I can help."

This is a non-profit effort. For more information and current prices, write to: Amnion Helpline, 116 Carl Street, San Francisco, CA 94117. Be sure and send a self addressed stamped envelope (SASE). They will send you general information about the Helpline and answer your inquiries. o

continued from page 26

BEGIN. I have been told by experts that more than one WHILE may be used in the sequence. I do not know, I never have found a need to try it.

Usually BEGIN ... UNTIL is sufficient for my work. Notice that RPTLP accomplishes essentially the same thing as UNTLP except that it counts down. I cannot think of a simple example requiring the full capability of WHILE.

The word RECUR? fetches (@) the value from the variable Q and checks for a zero value (0=), and if so leaves a True Flag for IF which calls ABORT. RECUR? is in turn called by MYSLP (MYSelfLooP), which uses the Forth recursive word MYSELF, a very convenient word that permits a word to re-execute itself. A Forth word may not call a word which is not yet completely defined, (its SMUDGE bit is not set --- later alligator!) as would be the case if one tried to include the name of a word in its own definition. MYSELF overcomes this limitation. There is however, no decisions made by MYSELF to exit the loop, a branching decision must be made by IF ... THEN. In this case that is done by RECUR?. One possible use of a MYSELF loop is my DIRectory word which will be presented next time. MYSLP is in turn called by MYLP which stores (!) the number 250 in the variable Q and executes HOME before calling MYSLP, which fetches (@) the value in Q, prints it, then decrements the value in Q by one before executing RECUR? and STOP?. All kind of dumb maybe, but they are only illustrations after all.

```
SCR #55
\ LOOP TESTING EGR 8/20/88
  FORGET IT : IT ;      0 VARIABLE Q
: STOP?  ?TERMINAL IF ABORT THEN ;
: DEMO CLS 0 0 AT -5 100 DO I . -5 +LOOP ;
: UNTLP HOME 0 BEGIN 1+ DUP . DUP 25 =
    UNTIL CR CR CR ." I DID IT ALL BY MYSELF" ;
: AGNLP HOME BEGIN ." THE ONLY WAY OUT OF "
    ." THIS LOOP IS TO HOLD "  ." 'FCT  4' "
    ." OR TURN OFF THE POWER OR " CR ." HIT THE "
    ." RESET BUTTON IF YOU HAVE ONE. " CR CR CR
    2 WAIT STOP? AGAIN ;
: RPTLP HOME 25 Q ! BEGIN Q @ DUP . 0 > WHILE
    -1 Q +! REPEAT 18 12 AT ." DONE!" CR ;
: RECUR? Q @ 0= IF ABORT THEN ;
: MYSLP Q @ . -1 Q +! RECUR? STOP? MYSELF ;
: MYLP 250 Q ! HOME MYSLP ;
```

C U next time, may the FORTH be with U o

XB tips Number 6

by Jim Swedlow, CA USA

[This article originally appeared in the User Group of Orange County, California ROM]

IF THEN ELSE - AGAIN: Another thing I found in the 'Teach Yourself XB' tutorial is how Extended BASIC matches ELSE's with IF's.

Each ELSE is paired with the last unmatched IF. For example:

```
     IF A THEN B :: IF C THEN D
     ELSE E ELSE F
```

In words: If A is true, do B and then test C. If C is true, do D. If C is false, do E. If A is false, do F.

DISK MENU PROGRAM: There were two errors in this program, both in line 210: first D$="DSK1."A$(A):: should have been D$="DSK1."&A$(A):: and, second, the HCHAR parameters should have been (5,1,32,32@20).

When I write a program, I generally start with something simple and then modify it until it does all that I want it to do. I'm working on a revised program that will delete files, print or display a disk catalog and run programs. Look for Version 2.0 soon.

LOADMAKER: This month's program is a Disk Menu program for those who do not have memory expansion.

A note about the origion of this program. It is similar in concept to POOR MAN's LOADER, which appeared in 99'er Magazine. Both read a disk's directory and then create a LOAD program on the disk in merge format. The difference between the two programs is excution. PML uses DEF statements while LOADMAKER uses subroutines.

Save this program as LOADMAKER. Initially and whenever programs are added or deleted, run LOADMAKER and then follow the on screen instructions to recapture your LOAD program.

As before, save your program on disk before running as some errors can cause your system to lock up. If the program that LOADMAKER writes is defective, use the following description to find the area of probable error and compare character by character to find the problem.

LINES 100-140 are the header.

LINE 150 sets up the video screen and does necessary intializing.

LINES 160-210 read the disk for programs. If no programs are found, line 210 stops excution. If there are more than 26 programs, the user is given a choice in lines 190-200 of either aborting or continuing.

LINE 220 closes the disk reading file and opens the file for writing the LOAD program (under the merge file name 'XXX').

LINES 230-240 create LOAD line 100 which is the sole header line.

LINES 250-260 write LOAD line 110 which clears the CRT and displays the top line of the screen.

LINES 270-280 make the LOAD lines that display the menu on the CRT. In line 270, the display format is set: 1 column in the center if less than 19 programs, 2 columns if more.

LINES 290-310 create a LOAD line that displays 'Press your choice' on CRT line 24.

LINES 320-350 make the CALL KEY line for selecting a program.

LINES 360-390 do a LOAD line that DISPLAY's 'Loading' followed by a GOTO statement that sends the program to the correct RUN line.

LINES 400-410 write the LOAD lines that display the name of the selected program and then RUN it.

LINE 420 saves the final line on disk as well as the end of file marker and then closes the disk file.

LINE 430 displays instructions for retrieving the LOAD program.

LINE 440 is a subroutine that increases the line number counter (B), prints the previous line on disk and displays the new line number on your CRT.

LINE 450 adds the line number in base 256 to the line.

LINE 460 is a subroutine that starts a new line and and then adds 'DISPLAY AT (F,G' to that line.

LINE 470 is a subroutine that adds the number F to a line.

Good luck!

```
100 ! LOADMAKER
110 ! VERSION XB.1.1
120 ! 29 DEC 84
130 ! BY JIM SWEDLOW
140 !
150 DISPLAY AT(10,10)ERASE ALL:"LOADMAKER": : : : : :
  "Initializing . . . ." :: @=1 :: DIM A$(26),A(26):: B$=
  CHR$(182)&CHR$(181)&CHR$(199):: B=100:: C$=CHR$(179)
160 OPEN #@:"DSK1.",INPUT ,INTERNAL,RELATIVE :: INPUT
  #@:A$(C),D,E,F :: DISPLAY AT(16,@):"Disk ";A$(C);" *
  Free";F :: A(C)=LEN(A$(C))
170 INPUT #@:D$,D,E,F :: IF D$="" THEN 210 ELSE IF
  ABS(D)<>5 OR D$="LOAD" THEN 170
180 C=C+@ :: IF C<27 THEN A$(C)=D$ :: A(C)=LEN(D$)::
  DISPLAY AT(17,@):"Reading:  ";D$ :: GOTO 170
190 DISPLAY AT(16,@)BEEP:"Your disk has more than 26
   programs.  Do you want to   proceed with the first 26
    programs? Press Y or N."
200 CALL KEY(3,F,D):: IF F=78 THEN CLOSE #@ :: STOP ELSE
  IF F<>89 THEN 200
210 IF C=0 THEN DISPLAY AT(16,@)BEEP:"No programs were
  found on   this disk." :: CLOSE #@ :: STOP
220 CLOSE #@ :: DISPLAY AT(16,@):"Disk reading
  completed": : : : : :: OPEN #@:"DSK1.XXX",VARIABLE
  163,DISPLAY ,OUTPUT
230 DISPLAY AT(16,@):"   Making LOAD line 100" :: D$=
  CHR$(0)&CHR$(100)&CHR$(131)&CHR$(32)&CHR$(80)&
  CHR$(82)&CHR$(79)&CHR$(71)&CHR$(82)&CHR$(65)&CHR$(77)
240 D$=D$&CHR$(32)&CHR$(76)&CHR$(79)&CHR$(65)&
  CHR$(68)&CHR$(69)&CHR$(82)
250 G,F=@ :: GOSUB 460 ::
  D$=D$&CHR$(182)&CHR$(239)&CHR$(236)
260 D$=D$&CHR$(181)&CHR$(199)&CHR$(7+A(0))&CHR$(68)
  &CHR$(105)&CHR$(115)&CHR$(107)&CHR$(32)&CHR$(42)&
  CHR$(32)&A$(0)
270 G=@-7*(C<19):: E=2 :: FOR D=@ TO C ::
  E=E-(D>C/2)*(G=@)*INT(C/2):: G=G+(D>C/2)*(G=@)*14 ::
  IF D=C AND G=15 AND C/2<>INT(C/2)THEN G=8
280 F=E+D :: GOSUB 460 :: D$=D$& B$&CHR$(A(D)+3)&
  CHR$(64+D)&CHR$(32)&CHR$(32)&A$(D) :: NEXT D
290 F=24 :: G=@ :: GOSUB 460 :: D$=
  D$&CHR$(182)&CHR$(238)&CHR$(181)&CHR$(199)&CHR$(17)
300 D$=D$&CHR$(80)&CHR$(114)&CHR$(101)&CHR$(115)&
  CHR$(115)&CHR$(32)&CHR$(121)&CHR$(111)&CHR$(117)
310 D$=D$&CHR$(114)&CHR$(32)&CHR$(99)&CHR$(104)&
  CHR$(111)&CHR$(105)&CHR$(99)&CHR$(101)
320 GOSUB 440 :: D$=D$&CHR$(157)&CHR$(200)&CHR$(3)&
  CHR$(75)&CHR$(69)&CHR$(89)&CHR$(183)
330 D$=D$&CHR$(200)&CHR$(@)&CHR$(51)&C$&CHR$(75)&C$&
  CHR$(83)&CHR$(182)&CHR$(130)
340 D$=D$&CHR$(132)&CHR$(75)&CHR$(191)&CHR$(200)&
  CHR$(2)&CHR$(54)&CHR$(53)&CHR$(186)&CHR$(75)&CHR$(192)
350 F=64+C :: GOSUB 470 :: D$=D$&CHR$(176)&CHR$(201)::
  GOSUB 450
360 F=24 :: GOSUB 460 :: D$=D$&B$&CHR$(7)&CHR$(76)&
  CHR$(111)&CHR$(97)&CHR$(100)&CHR$(105)&CHR$(110)&
  CHR$(103)
```

continued on page 18

A Look at Assembler

Subroutines: by Art Green, Ottawa Users Group

As we suggested last time, the BLWP method of calling subroutines may not always be as costly as it at first appears. Since the workspace belongs to the called subroutine it can have some useful values already "loaded" into the registers. As an aside, you must keep in mind that the registers are nothing special, they are just words of memory; the idea of registers is just an addressing mode which allows a 4 bit address (0 to 15) to make instructions shorter.

As well as having values pre-loaded into its registers, a subroutine need not allocate space for all 16 registers if they are not going to be used.

To show these concepts we will code a familiar subroutine, VMBW, first using internal linkage and then external linkage. We will then total up the cost of each method. Note that neither of the routines is done the way VMBW is coded in the Editor Assembler package; both are better. In addition, the external version is done "correctly" so that interrupts can be enabled in the mainline code and are disabled when using the VDP as they must be.

```
*
* VMBW Internal Linkage
*       No caller's registers changed.
*       Interrupts must be disabled.
*
*       LI   R0,Vaddr
*       LI   R1,Caddr
*       LI   R2,count
*       BL   @VMBW            Costs
*                        Storage Time
VMBW    MOV  R3,@S3         2     4     Save R3
        MOV  R2,@S2         2     4     Save R2
        MOV  R0,R3          1     3     R3=VDP address
        ORI  R3,>4000       2     3     Set VDP write bit
        MOVB R3,@>8C02      2     4     Set VDP address
        SWPB R3             1     2
        MOVB R3,@>8C02      2     4
        MOV  R1,R3          1     3     R2=CPU address
VMBW2   MOVB *R3+,@>8C00    2     5*    Byte to VDP
        DEC  R2             1     2*
        JGT  VMBW2          1     1*    Loop for all bytes
        MOV  @S3,R3         2     4     Restore R3
        MOV  @S2,R2         2     4     Restore R2
        RT                  1     2     Return to caller
S3      BSS  2              1     -     R3 save area
S2      BSS  2              1     -     R2 save area
*                         -----  -----
*          Total Cost     24     39 + 8 per byte

* VMBW External Linkage
*       No caller's registers changed.
*       Interrupts can be enabled or disabled.
*
*       LI   R0,Vaddr
*       LI   R1,Caddr
*       LI   R2,count
*       BLWP @VMBW            Costs
*                        Storage Time
        DEF  VMBW                        Define entry
VMBW    DATA SWSP-18,$+2    2     -     NOTE: Short WSP
        LIMI 0              2     2     Interrupts OFF
        MOV  R13,R12        1     3     R12->caller's R0
        MOV  *R12+,R11      1     4     R11 = VDP address
        ORI  R11,>4000      2     3     Set VDP write bit
        MOVB R11,*R10       1     4     Set VDP address
        SWPB R11            1     2
        MOVB R11,*R10       1     4
        MOV  *R12+,R11      1     4     R11=CPU address
        MOV  *R12,R12       1     4     R12=count
VMBW2   MOVB *R11+,*R9      1     5*    Byte to VDP
        DEC  R2             1     2*
        JGT  VMBW2          1     1*    Loop for all bytes
        RTWP                1     4     Return
SWSP    DATA >8C00,>8C02    2     -     R9, R10 Pre-loaded
        BSS  4              2     -     R11, R12 Work Regs
        BSS  6              3     -     R13 to R15 Linkage
*                         -----  -----
*          Total Cost     24     34 + 8 per byte
```

As you can see, even on this not very complex routine, we overcame the adverse storage and time cost of the BLWP linkage; and at the same time, added the "interrupt enabled" feature.

Next time we will continue our discussion of linkage, and look at entrance and exit code for main programs.

"It is impossible to make a program foolproof because fools are so ingenious." o

Putting it All Together #7

by Jim Peterson, Tigercub Software, USA

The hardest part of learning to program is not in learning what the various commands do - it is learning how to put them together to do what you want them to do! Key in this little subprogram, run it to see what it does, then read the explanation of how it does it.

ACCEPT AT with a negative SIZE will accept whatever is on the screen at the specified location, if Enter is pressed - but if you want to input something which is shorter than that default, you must be sure to erase the remaining characters. This can be a nuisance, especially when the previous input becomes the default. This subprogram displays and accepts the default if the Enter key is pressed, otherwise erases the default and accepts whatever is entered. However, extremely fast typing can cause the second character to be missed.

```
100 CALL CLEAR
110 A$="TESTING" :: D$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
:: CALL ACCEPT(18,1,A$,D$,-10,B$):: DISPLAY AT
(20,1):B$ :: GOTO 110
10000 SUB ACCEPT(R,C,A$,D$,L,B$):: DISPLAY AT(R,C):A$
10001 CALL KEY(0,K,S):: IF S=0 THEN 1001 ELSE IF K=13 TH
EN B$=A$ :: SUBEXIT ELSE IF LEN(D$)>0 AND POS(D$,CHR$(K)
,1)=0 THEN 1001
10002 DISPLAY AT(R,C):CHR$(K)&RPT$(" ",LEN(A$)-1)
10003 IF LEN(D$)=0 THEN ACCEPT AT(R,C+1)SIZE(L-1):B$ ELS
E ACCEPT AT(R,C+1)VALIDATE(D$)SIZE(L-1):B$
10004 B$=CHR$(K)&B$ :: SUBEND
```

Lines 100-110 are just a routine to demonstrate the subprogram. The screen is cleared, the default string is defined in A$ and the string of acceptable characters in D$. This validation string can be omitted but if present it must list the characters - you cannot specify UALPHA, DIGIT, etc. as you can with a normal ACCEPT AT. Then the subprogram is called, with parameters specifying the row, column, default string, validation string, maximum length of string; B$ is the variable to which the input will be assigned. When execution returns from the CALL, the input is displayed at row 20 and execution returns for another input.

In line 1000, the variables in the parameter list of the subprogram must be in exactly the same sequence as in the CALL, and all must be present. The default string is displayed at the specified row and column. Then CALL KEY is used to get the first key input. If S (status)=0 then no key has been pressed, so the line returns to itself. If the ASCII of the key pressed is 13 (the Enter key) the default string is assigned to the input and the subprogram exits.

If the length of the validation string is more than 0 (i.e., it is not a null string, characters have been assigned to it) and the position in the validation string of the character represented by the ASCII of the key, starting with the first character, is 0 (i.e., the character is not in the validation string) the line returns to itself. Otherwise, line 10002 displays, at

continued on page 18

Tips

Edited by Stephen Shaw, England

ON BACK-UPS AND FLIPPIES

If you already back-up your disks faithfully and have decided about flippies, skip this. Otherwise, read on.

BACK-UP's are essential. The first thing you do when you get a new program or disk is back it up. Do not run it, do not modify it, do not catalog it, do not list it - back it up.

Why? Simply put, disks go bad and disk drives eat disks. If your only copy goes bad, it could take days to weeks to get a replacement, depending on the source. Ever read the 'warranty' that comes with some software?

So, make your back-up first. If you buy a program, keep the master (with the maker's label) with your back-ups and use a working copy for every day.

Keep your back-ups and masters in a separate disk box away from your computer. That way you will not use them by mistake.

If a disk does go bad, make sure that your hardware is working by using another disk BEFORE using your master or back-up. Otherwise, you could destroy both copies!

Updating back-ups is vital. If you wrote the program, you probably will revise it more than once. If you get a single program, you will normally add it to an exiting disk. If you do not update your back-up, you will lose your new program if something goes wrong with your working disk.

A FLIPPY is a single sided disk that has been modified to act like two single sided disks. By adding three holes, you can put your disk in your drive upside down and record on the back. Instructions for making flippies can be found in the August, 1984 ROM.

Some folks, like Craig Miller, recommend against flippies. They argue that a disk is designed to turn one way and bad things happen when you flip it over and make it turn the opposite direction.

Others use flippies and claim that they have had no problems. One approach is to use the front as a working copy of one disk and the back as the back-up of another disk.

I compromise by only using flippies for back-up copies, thus reducing the number of back-up disks I need. I do not, however, use them for working disks. I have not had any problems.

===

ASC AND SEG$

A common coding for determining the ASCII value of the first character of a string is:

```
ASC(SEG$(A$,1,1))
```

This can be simplified by omitting the SEG$ function:

```
ASC(A$)
```

ASC always returns the ASCII value of the first character of the string. You will get an error if A$ is a null string (if A$="").

Suppose you want to lop off the first four characters of a string. You might do this:

```
A$=SEG$(A$,5,LEN(A$)-4)
```

Our 4A's, however, do not compare the third value in the SEG$ function (the length of the new string) to the length of the old string. Therefore, this works just as well:

```
A$=SEG$(A$,5,255)
```

Use 255 as that is the maximum length of a string variable. If the length of A$ is less than 5 - even if it is zero the new A$ will be a null string (but NO error).

===

POSition

One way to ask for a menu selection is to ask the user to input the first letter of his/her choice. Suppose that the options were <C>hange, <P>rint or <Q>uit. You might do this:

```
190 ACCEPT AT(10,10)SIZE(-1)
    VALIDATE("CPQ"):A$ ::
    IF A$="C" THEN 230 ELSE
    IF A$="P" THEN 340 ELSE
    IF A$="Q" THEN 980 ELSE 190
```

A simpler way would be to use POS:

```
190 ACCEPT AT(10,10)SIZE(-1)
    VALIDATE("CPQ"):A$ ::
    ON POS("CPQ",A$,1) GOTO
    230,340,960
```

If your user inputs a null, the POS function will return a value of 1 and control will transfer to the first line in the GOTO list. If this is a problem, do it this way:

```
190 ACCEPT AT(10,10)SIZE(-1)
    VALIDATE("CPQ"):A$ ::
    IF A$="" THEN 190 ELSE
    ON POS("CPQ",A$,1) GOTO
    230,340,960
```

(Source: a Tigerclub program)

===

DIM's and SUBPROGRAM's

Will this program work?

```
10 DIM A(5)
20 CALL SETUP(A())
30 SUB SETUP(B())
40 FOR I=1 TO 10
50 B(I)=I
60 NEXT I
70 SUBEND
```

Answer: It will crash in line 50 when I=6. Once A has been DIMensioned in line 10, the process of calling SETUP in line 30 transfers the DIMension to B within SETUP. In essence, there has been a DIM B(5) inside SETUP.

Note that a STOP is not needed at the end of line 20. Your 4A will not execute a SUBPROGRAM unless it is CALLed. o

continued from page 31

B) Load a file after you find it by simply doing the following:

* FCTN 9
* LF & "enter"
* Change the first 2 chars of the file name displayed!

NOTE: Be sure to put the right diskette in!!!

2. Go find those files!!!!
(Thanks TICO Topics UG, July, 1990) o

Multiplan Exercises #3

by Herbert Schlesinger, USA

FORMATTING

Bring the file " TAXES1 " to the screen.

a. Aligning Cell Data

The screen will show the spreadsheet after formatting.

1.Positon the cell pointer to R1C4.

2.Select Format from the command menu. A submenu is displayed:

```
FORMAT: Cells Default Options Width
```

3.Select Cells from this menu. Another submenu appears:

```
FORMAT cells: R1C4  align: (D) C G L R—
cd:(Def)Cont Exp Fix Gen Int $ * % -
# of decimals: 0
```

4. Press the colon (:) key, and type in R1C6. Room for this is provided. Then press CTRL A. Now we have:

```
FORMAT cells:R1C4:R1C6     align: (D) C G L R—
cd:(Def)Cont Exp Fix Gen Int $ * % -
# of decimals:0
```

5. Press C so that the menu pointer moves to the "center" option.

6. Press <ENTER>. The titles will shift to the center of the columns.

Other options from this submenu are:

Setting	Example	Description
D (Default)	What is there?	
C (Center)	Salary 22	Both Alpha and numbers are centered in the cell.
G (General)	Salary 22.55	Labels are left-aligned; Numbers are right-aligned.
L(eft)	Salary 123.45 22	Both labels and numbers are are left-aligned.
R(ight)	Salary 123.45 22	Both labels and numbers are are right-aligned.

To right-align the labels in column 1:

1.Place the cell pointer to R2C1. Select Format from the command line menu.

2.Select Cells:

```
FORMAT cells:    R2C1    align:    (D)    C G L R-
cd:(Def)Cont Exp Fix Gen Int $ * % - # of decimals:0
```

3.Press colon (:), type R9C1 and press Ctrl A to get this:

```
FORMAT cells:R2C1:R9C1    align:    (D)    C G L R-
cd:(Def)Cont Exp Fix Gen Int $ * %         -
# of decimals:0
```

4.Select R for right-alignment, and Press <ENTER>.

Using the same methods the numbers can be aligned on the sheet. But let's go further with the numbers.

DISPLAY FORMATING

Cell appearance can be formatted as well as the alignment:

1> Position cell pointer on the first number in the Column to be formated (R2C4).

2> Select FORMAT, select CELLS, type ":" to select a RANGE and then type in the last cell in the block (or column) to be formated (R12C5).

3> Press CTRL A to tab over, CTRL A again to get to the (cd) "format" code options, select "$" and then press <ENTER>

Column 6 contains percentages so we again place the cursor on the starting cell,(R2C6), select FORMAT, CELLS, press : and then type R12C6 press CTRL A (the TAB key) twice and then press (%). Next press CTRL A to move to where you are asked for the number of decimals. Enter "2" and then <ENTER>.

The options offered on the format code line:

```
format code:(Def)Cont Exp Fix Gen Int $ * %
# of decimals: 0
```

CODE	MEANING	DESCRIPTION
Def	Default	Displays cells D in default format. That is what is there when no formating is done. It also undoes previous formats.
Con	Continuous	Labels which are too long for a cell are extended into the cell(s) overwriting what is there. Used for putting long titles or labels on a spread-sheet.
Exp	Expotential	Numbers are displayed in scientific notation, like 1E+6. Use the "# of decimals" to set the number of decimals in this notation.
Fix	Fixed decimals	Numbers are rounded to a fixed number of decimal places as you select using "# of decimals".
Gen	General	Numbers are shown with as much accuracy as possible. Very large or small numbers are shown in scientific notation.
Int	Integer	Numbers are rounded to integers, no decimals.
$	Dollar	Numbers shown in monetary ($564.32) format.
*	Bar Graph	Displays a series of stars, no numbers. One star for each integer value in the cell. (If the value falls between 4.5 and 5.5 five stars are displayed.
%	Percent	Shows all numbers as percentage figures with the % sign. Use "# of decimals" to set the display.

Contrary to the general computer usage, commas can be displayed in the thousands place by following this instruction:

1. Select Format which will bring this up:

```
FORMAT: Cells Default Options Width
```

2. Spacebar twice, <ENTER>, or press " O " selecting Options which shows:

```
FORMAT OPTIONS commas:Yes(No)   formulas:Yes(No)
```

3. Typing "Y" will change the commas option to YES. Press <ENTER>.

Now your spreadsheet will have some cells containing #### in place of figures. This means there is not enough room to display what should be there. So we must widen the column, thus:

1. Place the pointer in the column to be widened. (Here C 4)
2. Press Format, and then select Width. We have:

FORMAT WIDTH in chars or d(efault):   d
Column 4 through 4

Change the " d " to 12; press <ENTER>. The resulting screen was saved as the file "TAXES2".

To change the width of several columns at the same time, use the same steps as above, but after entering the desired width press CTRL A and move to the next option: column. Enter a number here or pass by using CTRL A again taking you to through. Enter the desired column number and then press <ENTER>. Actually it is not necessary for the pointer to be in one of the columns you wish to narrow or widen but if you are, the column: option will default to that column. You may enter any column in either of the locations. Notice that these changes can not skip columns. The affected columns for each Format must be contiguous. If the letter "d" is entered for the width, the width will revert back to the original (default) setting of 10 spaces.

FORMULA DISPLAY:

Refer to the previous command line illustration: if, instead of Enter we had pressed CTRL A , formulas: Yes(No) would have been hylited. Pressing "Y" and then <ENTER> will cause some changes - all columns are widened to 20 spaces, and the formulas rather than their results are displayed on the screen.

To see this; use Transfer; Load " PRINT_SS " will show the tax table in this form.

CHANGING THE DEFAULTS:

Defaults are the settings previously given to each cell, row and column either by the program itself or by a change you may have made in the FORMAT DEFAULT: Cells Width

The options offered are exactly the same as the Format Cells you have previously used except the (Def) is missing. The default is hightlighted as you will see and changes are made by le or spacebar for alignment and, after CTRL A, in the format code. The default is in parentheses.

To change the width default, merely shift (CTRL A) to the width option and enter the number of spaces you want as a default. All columns will be changed EXCEPT those already set by the Format cells option.

RANGES and COPYING:

PUT " COMMISSION " on the screen.
A block of cells, in one column or in more than one column is called a range. We previously designated a range by typing a colon : after the starting cell and then typing in the last cell of the "range".

We are now going to draw a range:
1. Move cell pointer to R3C2, the top of the numbers to be formated.
2. Select FORMAT ; then select Cells ; Type in a colon :

You then have:

FORMAT cells:R3C2: _

Instead of typing another cell reference, press the down arrow key four times - R7C2 will then show after the colon. Press the CTRL A key twice; select the $ option and then press <ENTER>. The data in the range which you "drew" will show the dollar sign and two decimals.

What we did was "draw" the range of the cells to format. This works with any command requiring a range. Here are the rules:

1. Place the cell pointer on the top or upper left corner of the range you want to work with.
2. Select the command you wish to invoke.
3. When the command asks for a range of cells; press the " : " key.
4. Move the cell pointer to the bottom right or bottom cell of the range you want to use.
5. Press the <ENTER> key.

USING RELATIVE CELL REFERENCE:

Ranges are most convenient when used in formulas: to figure the commissions column on the screen we would have to use R3C2*R3C3 for row three and for row four the formula is R4C2*R4C3 and so on down the column. Here is how to do it by ranges:
a. Cell pointer on R3C4.
b. Press the = key.
c. Use the left arrow key to move the pointer to R3C2 (two cells left). Here is what we have:

VALUE: RC[-2] _ (Notice the [] brackets)

d. Press the * key (for multiply) and the cell pointer jumps back to the original cell; and the screen looks like this:

VALUE: RC[-2]* _

e. Move the cell pointer to R3C3 by pressing the Left arrow key once. Now we have this screen:

VALUE: RC[-2]*RC[-1] _

f. Press <ENTER>. The calculation then appears in R3C4.

The formula we are using, instead of R3C2*R3C3 is:

RC[-2]*RC[-1]

What this says is: This row (R), two colunms to the left (C[-2]) times this row (R) one column to the left (C[-1]). Now what we can do with this is copy the formula to the other rows. o

continued from page 29

```
FCTN-SHIFT-D - TOGGLE DUPLEX BETWEEN FULL AND HALF
FCTN-SHIFT-P - PRINT SCREEN IN FREEZE OR WINDOW-BACK
                                                MODE.
FCTN-SHIFT-T - ENABLES TRANSMISSION OF FILE BY TE2
                                            PROTOCOL.
FCTN-SHIFT-X - ENTERS XMODEM FILE TRANSFER MODE.
*
CURSOR CONTROL
 - USE ARROW KEYS OR CONTROL (K,S,I,J)
*
      UN-DOCUMENTED CONTROL KEYS ARE SAME AS TI-99/4(A)
                                   IN PASCAL SCAN MODE.
*
FCTN- = QUIT-LEAVE PROGRAM
CTRL- =  "
*
```

PROGRAM KNOWS ALL STANDARD ADM-3A CONTROL CODES AND ESCAPE SEQUENCES. PRINTER BUFFER TURNS OFF AND RINGS THE BELL WHEN IT GET FULL. LOG AUTOMATICALLY WRITES TO DISK WHEN IT GETS FULL. o

continued from page 20

```
REPLACE QTS WITH ANS
SELECT 1
REPLACE 3.RTOT WITH 3.RTOT - ANS
REPLACE 3.LDT WITH .DATE.
RETURN
*
* INVUPDT  Save as INVUPDT/C
* *******  Ver. 2.01 04/01/89
```
o

Jenny's Younger Set

Dear Jenny,

This is Crocodile Jones. I am writing to you about my part in the Full Day Tutorial coming up soon. I hope people will be able to come to Crocodile Jones' Adventure Trail.

Crocodile Jones

Dear Jenny,

This month I have two programs for you. I hope you like them.

Vincent Maker

```
100 RIGHT=0
110 WRONG=0
120 REM
130 REM BY VINCENT MAKER
140 REM FOR MELANIE.
150 CALL CLEAR
160 DISPLAY AT(3,7):"MUSIC QUIZ."
170 DISPLAY AT(5,7):"BY VINCENT MAKER."
180 DISPLAY AT(7,7):"NOVEMBER 1990."
190 FOR T=0 TO 500
200 NEXT T
210 DISPLAY AT(3,7):"1.  WHO SANG,""IT MUST HAVE BEEN
    LOVE""?"
220 DISPLAY AT(6,7):"A) ROXETTE"
230 DISPLAY AT(7,7):"B) JASON DONOVAN"
240 DISPLAY AT(8,7):"C) WILSON PHILLIPS"
250 DISPLAY AT(9,7):"D) THE BEACH BOYS."
260 PRINT "PRESS THE APPROPRIATE KEY."
270 CALL KEY(0,J,K)
280 IF K=0 THEN 270
290 IF J=65 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
300 DISPLAY AT(5,7)ERASE ALL:"2.  WHO SANG,""RUNAWAY
    HORSES""?"
310 DISPLAY AT(8,7):"A) LITTLE RIVER BAND"
320 DISPLAY AT(9,7):"B) BELINDA CARLISLE"
330 DISPLAY AT(10,7):"C) WILSON PHILLIPS"
340 DISPLAY AT(11,7):"D) THE D-GENERATION"
350 PRINT "PRESS THE RIGHT KEY."
360 CALL KEY(0,K,L)
370 IF L=0 THEN 360
380 IF K=66 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
390 DISPLAY AT(3,7)ERASE ALL:"3.  WHO SANG,""FIVE IN A
    ROW""?"
400 DISPLAY AT(5,7):"A) THE D-GENERATION"
410 DISPLAY AT(6,7):"B) JASON DONOVAN"
420 DISPLAY AT(7,7):"C) KYLIE MINOGUE"
430 DISPLAY AT(8,7):"D) WET WET WET"
440 PRINT "PRESS YOUR ANSWER."
450 CALL KEY(0,J,I)
460 IF I=0 THEN 450
470 IF J=65 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
480 DISPLAY AT(3,7)ERASE ALL:"4.  WHO SANG,""HOLD ON""?"
490 DISPLAY AT(5,1):"A) THE BEACH BOYS"
500 DISPLAY AT(6,1):"B) WILSON PHILLIPS"
510 DISPLAY AT(7,1):"C) THE NOTTING HILLBILLES"
520 DISPLAY AT(8,1):"D) THE CHANTOOZIES"
530 PRINT "PRESS THE ANSWER."
540 CALL KEY(0,H,G)
550 IF G=0 THEN 540
560 IF H=66 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
570 PR=RIGHT/40/1
580 PW=WRONG/40/1
590 IF RIGHT=0 THEN RESULT$="YOU DID PRETTY BAD.  0 OUT
    OF 4"
600 IF RIGHT=1 THEN RESULT$="YOU DIDN'T DO VERY WELL.  1
    OUT OF 4"
610 IF RIGHT=2 THEN RESULT$="YOU DID FAIRLY WELL."
620 IF RIGHT=3 THEN RESULT$="YOU DID GOOD.  3 OUT OF 4"
630 IF RIGHT=4 THEN RESULT$="YOU DID VERY WELL.  4 OUT
    OF 4"
635 PRINT RESULT$
636 PRINT
640 PRINT "YOU GOT ";PR;"% RIGHT AND     ";PW;"% WRONG."
650 PRINT
660 IF RIGHT>4 THEN 680
670 STOP
680 PRINT "DO YOU WANT ANOTHER GO(Y/N)?"
690 CALL KEY(0,H,J)
700 IF J=0 THEN 690
710 IF H=89 THEN 100 ELSE END
```

```
100 CALL CLEAR
110 REM BY VINCENT MAKER
120 INPUT "YOU ARE FOUND ON THE WIZARD'S PROPERTY.  HE
    DEMANDS AN EXCUSE-":A$
130 PRINT
140 PRINT "SORRY.  YOU ARE COMMITTED FOR TRIAL."
150 PRINT
160 INPUT "HOW DO YOU PLEAD ON TRESPASS?":PLEA$
170 IF PLEA$="GUILTY" THEN 330
180 INPUT "THE JUDGE WANTS TO KNOW IF YOU WERE THERE TO
    TRESPASS?":P$
190 IF P$="YES" THEN SENTENCE=2
200 INPUT "THE JUDGE WANTS TO KNOW IF YOU DAMAGED
    ANYTHING WHILE YOU WERE THERE?":S$
210 IF S$="YES" THEN SENTENCE=5+SENTENCE
220 PRINT
230 PRINT "IF YOU ARE FOUND NOT GUILTY YOU WILL RECEIVE
    A SENTENCE OF 0 YEARS"
240 PRINT "THE JUDGE SENTENCES YOU TO ";SENTENCE;"
    YEARS"
250 IF SENTENCE=0 THEN END
260 INPUT "DO YOU WISH TO APPEAL?":K$
270 IF K$="YES" THEN 300
280 PRINT "SEE YOU IN ";SENTENCE;" YEARS!!"
290 END
300 A=INT(RND*SENTENCE)
310 PRINT "THE JUDGE REDUCES YOUR SENTENCE BY ";A;"
    YEARS."
320 END
330 PRINT "YOU ARE FOUND GUILTY AND SENTENCED TO DEATH."
```

o

Spellbreaker part 5

Copyright 1985 Infocom
This walk through is by Scorpia, Copyright 1985.

OK! Blorple the "MAGIC" cube. Once inside, make sure you are holding the vellum scroll with Girgol and the knife. You might also want to save here, in case you make a mistake later on (being branded a "menace to society" is so terribly embarrassing!).

Go through the east exit. You will find yourself in an eerie throne room. Just wait a bit and the shadowy figure will appear on the throne. Keep waiting, and the figure will explain what this has all been about. However, you do not want to wait too long. Attack the figure. You will be frozen in place. The figure would have done this to you anyway, but timing is critical here. You must get yourself frozen as soon as possible, so the spell will wear off in time for you to stop your shadow self from completing its grand project.

So, there you are, rigid and unmoving. After making its explanations, the figure will take the cubes from you, and begin constructing the Hypercube, the most potent magical force ever. As it does so, the freeze spell slowly begins to wear off. Little by little, your body thaws out.

Now you must be careful here. The Girgol spell must be cast at the proper moment, or all is lost. No matter when you thaw out completely (and it absolutely has to be completely), keep waiting until the figure is just about to jump into the Hypercube. NOW, cast Girgol!

Time has stopped, but it will not be stopped for very long. Do not waste any moves here, you only have two. First, get the "MAGIC" cube from inside the Hypercube. It will be difficult, but you will manage it. Next, put the knife in the Hypercube.

Just as you do that, time resumes...but the figure had already been on its way into the cube, and it can not stop now. It is drawn to the center, but the "MAGIC" cube is no longer there! The whole cube begins to collapse on itself. You have saved the world (indeed, the universe)! However.....

However, there was a price for that. You have not only put an end to the nefarious plans of your shadow self, you have also put an end to all magic, as Lelboz tells you. And, considering what you learned from your shadow self, perhaps that is really the better way. Congratulations, Scientist.....and Spellbreaker! o

```
100 ! ****************
110 ! * CYBER-ABACUS *
120 ! ****************
130 ! COPYRIGHT 1985
140 ! EMERALD VALLEY PUBLISH
ING CO.
150 ! BY SCOTT WILLIAMS
160 ! HOME COMPUTER MAGAZINE
170 ! VERSION 5.5.1
180 ! TI EXTENDED BASIC
190 ON ERROR 880 :: ON WARNI
NG NEXT
200 DISPLAY AT(12,9)ERASE AL
L:"CYBER-ABACUS": : : : : :
: :"   PLACE ALPHA LOCK DOWN
": :"   PRESS ENTER TO START
" :: GOSUB 810
210 CALL CLEAR :: CALL SCREE
N(5)
220 DISPLAY AT(2,1):"ENTER P
RINTER PARAMETERS IF YOU WAN
T A PRINTOUT.": :"PRESS ENTE
R WITH NO":"PARAMETERS IF YO
U DON'T":"HAVE A PRINTER."
230 DISPLAY AT(10,1):"PRINTE
R:" :: ACCEPT AT(11,1)SIZE(2
8):PRNT$ :: IF PRNT$="" THEN
 P=0 ELSE P=1
240 CALL CHAR(35,"0000001F1F
181818000000F8F8181818181818
1F1F000000181818F8F8000000",
39,"181818181818181800000OFF
FF",126,"0")
250 VI$=" +  -  *  /  ^  C
Q  F  M  CM +M -M *M /M R  P
 "

260 FOR Z=65 TO 90 :: CALL C
HARPAT(Z,C$):: CALL CHAR(Z+3
2,C$):: NEXT Z
270 FOR Z=1 TO 8 :: CALL COL
OR(Z,2,8):: NEXT Z :: FOR Z=
9 TO 12 :: CALL COLOR(Z,16,5
):: NEXT Z
280 GOSUB 690
290 GOSUB 740
300 GOSUB 760 :: IF OP>5 THE
N GOSUB 330 :: GOTO 320
310 GOSUB 780 :: GOSUB 330
320 IF CF=1 THEN CF=0 :: GOT
O 290 ELSE IF DONE=1 THEN EN
D ELSE GOTO 300
330 ON OP GOSUB 340,350,360,
370,680,380,390,410,590,600,
610,620,630,640,660,670 :: R
ETURN
340 F1=F1+F2 :: L=3 :: NUM$=
STR$(F1):: GOSUB 700 :: RETU
RN
350 F1=F1-F2 :: L=3 :: NUM$=
STR$(F1):: GOSUB 700 :: RETU
RN
360 F1=F1*F2 :: L=3 :: NUM$=
STR$(F1):: GOSUB 700 :: RETU
RN
370 IF F2=0 THEN CALL SOUND(
400,110,0):: RETURN ELSE F1=
F1/F2 :: L=3 :: NUM$=STR$(F1
):: GOSUB 700 :: RETURN
380 L=12 :: NUM$="" :: GOSUB
 700 :: OP=32 :: GOSUB 710 :
: P$="CLEAR" :: GOSUB 820 ::
 L=3 :: GOSUB 700 :: CF=1 ::
 RETURN
390 CALL HCHAR(19,1,126,192)
:: DISPLAY AT(20,1):"want~to
~exit~y~or~n~~N";:: ACCEPT A
T(20,22)SIZE(-1)VALIDATE("YN
"):A$
400 IF A$="N" THEN RETURN EL
SE DONE=1 :: P$="EXIT PROGRA
M" :: GOSUB 820 :: RETURN
410 CALL HCHAR(19,1,126,192)
:: RESTORE 870 :: FOR Z=19 T
O 23 :: READ A$ :: DISPLAY A
T(Z,1):A$;:: NEXT Z
420 DISPLAY AT(24,1):"functi
on~~";
430 ACCEPT AT(24,10)SIZE(1)V
ALIDATE("ABCDEFGHIJKL"):A$ :
: IF A$="" THEN 430
440 ON ASC(A$)-64 GOSUB 450,
460,470,480,490,500,510,520,
530,540,550,570 :: L=3 :: NU
M$=STR$(F1):: GOSUB 700 :: R
ETURN
450 F1=-(F1):: P$="NEGATE" :
: GOSUB 820 :: RETURN
460 F1=ATN(F1):: P$="ARCTANG
ENT" :: GOSUB 820 :: RETURN
470 F1=COS(F1):: P$="COSINE"
 :: GOSUB 820 :: RETURN
480 IF F1<=294 THEN F1=EXP(F
1):: P$="EXPONENTIAL VALUE"
:: GOSUB 820 :: RETURN ELSE
CALL SOUND(400,110,0):: RETU
RN
490 F1=INT(F1):: P$="INTEGER
" :: GOSUB 820 :: RETURN
500 F1=LOG(F1):: P$="LOGARIT
HM" :: GOSUB 820 :: RETURN
510 F1=PI :: P$="VALUE OF PI
" :: GOSUB 820 :: RETURN
520 F1=SIN(F1):: P$="SINE" :
: GOSUB 820 :: RETURN
530 F1=SQR(F1):: P$="SQARE R
OOT" :: GOSUB 820 :: RETURN
540 IF F1<=15707963267 AND F
1>=-15707963269 THEN F1=TAN(
F1):: P$="TANGENT" :: GOSUB
820 :: RETURN ELSE CALL SOUN
D(400,110,0):: RETURN
550 IF F1>=-1 AND F1<=1 THEN
 F1=ATN(F1/SQR(1-F1*F1)):: P
$="INVERSE SINE (ARCSINE)" :
: GOSUB 820 :: RETURN
560 CALL SOUND(400,110,0)::
RETURN
570 IF F1>=-1 AND F1<=1 THEN
 F1=-ATN(F1/SQR(1-F1*F1))+PI
/2 :: P$="INVERSE COSINE (AR
CCOSINE)" :: GOSUB 820 :: RE
TURN
580 CALL SOUND(400,110,0)::
RETURN
590 MEM=F1 :: L=17 :: NUM$=S
TR$(MEM):: GOSUB 700 :: P$="
MOVE FIELD 1 TO MEMORY" :: G
OSUB 820 :: RETURN
600 MEM=0 :: L=17 :: NUM$=""
 :: GOSUB 700 :: P$="CLEAR M
EMORY" :: GOSUB 820 :: RETUR
N
610 F1=F1+MEM :: L=3 :: NUM$
=STR$(F1):: GOSUB 700 :: P$=
"ADD MEMORY" :: GOSUB 820 ::
 RETURN
620 F1=F1-MEM :: L=3 :: NUM$
=STR$(F1):: GOSUB 700 :: P$=
"SUBTRACT MEMORY" :: GOSUB 8
20 :: RETURN
630 F1=F1*MEM :: L=3 :: NUM$
=STR$(F1):: GOSUB 700 :: P$=
"MULTIPLY MEMORY" :: GOSUB 8
20 :: RETURN
640 IF MEM<>0 THEN F1=F1/MEM
 :: L=3 :: NUM$=STR$(F1):: G
OSUB 700 :: P$="DIVIDE BY ME
MORY" :: GOSUB 820 :: RETURN
650 CALL SOUND(400,110,0)::
RETURN
660 F1=MEM :: L=3 :: NUM$=ST
R$(F1):: GOSUB 700 :: P$="RE
CALL MEMORY" :: GOSUB 820 ::
 RETURN
670 IF PRNT$>"" THEN P=ABS(P
-1):: GOSUB 720 :: RETURN EL
SE CALL SOUND(400,110,0):: R
ETURN
680 F1=F1^F2 :: L=3 :: NUM$=
STR$(F1):: GOSUB 700 :: RETU
RN
690 CALL HCHAR(1,1,126,768):
: FOR Z=1 TO 15 :: READ X,Y,
A$ :: DISPLAY AT(X,Y):A$;::
NEXT Z :: GOSUB 720 :: RETUR
N
700 CALL HCHAR(L,4,32,18)::
DISPLAY AT(L,1+(18-LEN(NUM$)
))SIZE(LEN(NUM$)+1):NUM$ ::
P$=NUM$ :: GOSUB 820 :: RETU
RN
710 CALL HCHAR(7,4,OP):: RET
URN
720 DISPLAY AT(7,20)SIZE(7):
"printer" :: IF P=1 THEN DIS
PLAY AT(8,20)SIZE(7):"  ON"
ELSE DISPLAY AT(8,20)SIZE(7)
:"  OFF"
730 RETURN
740 CALL HCHAR(3,4,32,18)::
ACCEPT AT(3,2)VALIDATE(NUMER
IC)SIZE(16):NUM$ :: IF NUM$=
"" THEN F1=0 ELSE F1=VAL(NUM
$)
750 L=3 :: GOSUB 700 :: RETU
RN
760 ACCEPT AT(7,2)SIZE(2)VAL
IDATE("+-*/^CFMPRQ"):OP$ ::
IF OP$="" THEN 760 ELSE OP=I
NT(POS(VI$," "&OP$&" ",1)/3+
.7)
770 IF OP=0 THEN GOSUB 800 :
: GOTO 760 ELSE P$=OP$ :: GO
SUB 820 :: RETURN
780 CALL HCHAR(12,4,32,18)::
 ACCEPT AT(12,2)VALIDATE(NUM
ERIC)SIZE(16):NUM$ :: IF NUM
$="" THEN F2=0 ELSE F2=VAL(N
UM$)
790 L=12 :: GOSUB 700 :: RET
URN
800 CALL HCHAR(19,1,126,192)
:: DISPLAY AT(19,1):"use~+~-
~*~/~~or";:: DISPLAY AT(20,1
):"C~Q~F~M~CM~+M~-M~*M~/M~R~
P";:: RETURN
810 CALL KEY(0,K,S):: IF S=0
 THEN 810 ELSE RETURN
820 IF P=0 THEN RETURN ELSE
OPEN #3:PRNT$ :: PRINT #3:P$
 :: CLOSE #3 :: RETURN
830 DATA 1,1,field~a,2,1,#((
((((((((((((((($,3,1,'
                 ',4,1,%((((((((
(((((((((&
840 DATA 6,1,#(($,7,1,'  '~~
operator,8,1,%((&
850 DATA 10,1,field~b,11,1,#
(((((((((((((((((($,12,1,'
                   ',13,1,%((((
(((((((((((((&
860 DATA 15,1,memory,16,1,#(
(((((((((((((((((($,17,1,'
                  ',18,1,%(((((
((((((((((((&  ____ __  ___
870 DATA a_ neg___ f_ log___
~k_ asn,b_ atn___ g_ pi  __
1 _acs,c_ cos __ h _sin,d_ e
xp_  i sqr,e int   j ta
n
880 CALL ERR(EC,ET,ES,LN)::
NUM$="0" :: IF EC=74 THEN RE
TURN 900
890 IF EC=109 OR EC=130 THEN
 P=0 :: GOSUB 720 :: ON ERRO
R 880 :: RETURN NEXT
900 ON ERROR 880 :: IF LN=60
0 THEN 740 ELSE 780
```

Modem Revolution?

by Earl Raguse, California, USA

**MONOGAMY IS A MUST
WHEN THE MODEM IS THE MEDIUM**

For years now my computer-nut buddy Ian had been trying to sell me on modems. "You will love it", he said, "it is a whole new world!"

Now, modems are little devices that let your computer connect up with other computers through the phone lines. With a modem, your computer can link up with other machines that have modems, share programs, exchange data, and even merge with national and international networks.

"It is like a giant singles bar for computers," Ian boasted. "It is the new age of information. There are no limits to data transmission! Information is free, and mankind is liberated. It is the electronic revolution."

So I bought a modem and joined the revolution, but as usual, I was too late.

I immediately called up my friend, "Ian! I have finally got a modem. Hook up your computer. Lets upload and download, baby."

"Are you kidding?" he gasped. "I never connect with other computers anymore."

"But I have got some great new programs. Let's link up."

"No way, man! Who knows where those programs have been."

"But Ian," I protested, "what about the New Age of Information? What about the liberating electronic revolution?"

"Aww, c'mon. Where have you been? This is the 1980's. Have you not heard of bugs? Have you not heard of viruses?"

"Huh?"

"A computer bug is a program where some nasty person has stuck in instructions that make your computer do bad things. It might be simple, like flash 'Ha Ha' on the screen, or it might wipe out all your data. Some bugs can even crash your entire system.

"And some viruses are worse. Somebody can stick a line or two in a program's millions of commands that will not only do bad things, but will write itself onto other disks and programs. If it gets loose - onto the networks. A bad virus can get into any computer that hooks in. There are lots of bad viruses out there!"

"So..." I felt sick.

"...so you cannot be too careful. No hacker with any sense is going to let his computer hook on with just anyone."

"Not even for some quick data exchange?"

"Those are the worst," Ian said. "You are at risk for every virus in the book. After all, how much do you really know about the other computer? It might be the kink that goes on-line for any stray word processor with a wink and an access code."

"But the free flow of information," I cried.

"We have all had to change our habits, the free and easy days of the 1970's and early 80's are over, my friend. We have all had to adopt more responsible attitudes. Sure, we all used to link up on Saturday nights, but no more. In fact, now I only log on one system that I know is clean. Monogomy is fashionable."

"So how come I never heard of all this?"

Ian shrugged, "Search me. Did you not get a packet from the surgeon general?"

"My great aunt Mildred probably threw it away," I said with a groan. "So it is over. My poor TI can never join that wild scene of swinging computers?"

"Only at your own risk," Ian said. "Except well... you can buy a sort of buffer that identifies and catches program bugs as they come in. The exchange is not quite... as sensitive. But it does offer protection. In fact, it is uniformally recommended that no actively networking computer be without one."

"You cannot mean its come to this?"

"I am afraid so," said Ian, "computer condoms." o

continued from page 25

MACFLIX 22...back to the Future 2; Country Code; and two Hedgehogs.

a new series also in Macflix format:

MACPAINT1...NAGEL1, NAGEL2, NAGEL3, NAGEL4 (girls)
MACPAINT2...Cavegirl, Communications Pin, Garfield, Little Men cartoon, Mickey Mouse, Picard (ST-TNG)
MACPAINT3...CIRCUS, HAN SOLO, ESCHER WATERFALL, WOOD DUCKS
MACPAINT4...SHAWS100,SHAWS400(slightly distorted),Unicorn, Yoda.
MACPAINT5...Five pics of George Shaw! plus a Manchester Theatre
MACPAINT6...BARTON ARCADE and TOWN HALL, MANCHESTER; GEORGE; STARTREK TRIO

=============

Archive section- the following disks are in archived format, and as appropriate replace double disk sets previously offered, or may be "new" adds-each now SSSD:
DEBUGGER- the 1984 Navarone debugger plus Source Code for Navarone Bugfixer. No docs but similar to TI Debug or SBUG.
DM1000 3.7+4.0-withdrawn versions for historic purposes- withdrawn as they are liable to corrupt files or disks from time to time.
FAST TERM- now offered with source code on one disk.
SYSTEM DISK LOADER-now on one disk.
TE3-now with source code on one disk.

=============

TI BASE section- databases for use with TI Base(required):
TI*MES issues 1 to 26 (1000 entries) INDEX
USER GROUP DATABASE (partly out of date-lots of duplications) NAMES INDEX [requires double sided disk-but only counts as one disk]-lots of addresses, mainly American, associated with the TI, many well out of date!
UK MAGAZINE INDEX-All issues of TI LINES, TIHCUG, TI USER, PARCO, TIdings, and EAR from May88 to Dec89
MICROPENDIUM INDEX JAN89->
UK INDEX 2- TIMES from issue 27 and EAR from Jan90
>UTIL20 has SAVEXT, a utility which recovers a crashed XB program! PROVIDED you keep the PEB switched on. Very neat.
>GAMES 22: starts off with another American Monopoly for 2-6 players, and an interesting Pinball construction set by John Behnke, POWERBALL, ALL in XB, and the most effective pinball in XB I have seen.

MA2 was accidentally wiped and it NOW contains a lovely picture of a bird, in 256 colours if you have a 9938VDP, and 424 lines long, so you need XHi (Geneve or TI+9938) or SmArtCopy (TI+9918).

Best wishes!
Stephen. o

TI-Base Tutorial #9 (cont)

by Martin Smoley, North Coast 99ers USA

NOTE: The following text was left out of last month's TI-Base tutorial:

The next database is MSRET. This is a good example of how to use a database to store information that would have previously taken up LOCAL variable space. I have used all 17 fields. Even though I don't need them at this time, I will probably use them later. You will notice that I have mixed C)haracter fields with X-type fields in a normal database. This is a very important development. If you need to do a very special printout or you have paper size restriction etc., special control codes could be saved with individual data records to automatically change the printer settings for special fields. A simple example of this would be a database containing 100 names, which must be printed weekly for inventory. 90 of those names have 80 characters or less, and the other 10 names have from 80 to 110 characters. The standard form you print on has 85 spaces to print the names. You can include special control codes with certain names to change the print pitch, or micro-justification if you have a very expensive printer, and the names will fill the space perfectly every time. I am probably confusing you with ideas, so let's get back to the subject. MSRET is self explanatory. As you will see late, I use these fields to print labels and screen messages. There is only one record in this database, the one listed below. A more complicated system could use more records for different labels and different messages.

```
      Database:  MSRET
 1    FR ..... From:
 2    TO ..... To:
 3    NAME ...   Martin A. Smoley
 4    STREET .   6149 Bryson Drive
 5    CTSTZP .   Mentor, OH  44060
 6    CTN ....  ** CAUTION **
 7    CD .....     COMPUTER DISK
 8    DNB ....  DO NOT BEND OR FOLD
 9    DNX ....     DO NOT X-RAY
10    MS1 ....  *******
11    MS2 .... * ORDER *
12    MS3 .... * MORE  *
13    MS4 .... * DISKS *
14    CD1 .... 1B331800000000000000
15    CD2 .... 1B332400000000000000
16    CD3 .... 00000000000000000000
17    CD4 .... 00000000000000000000
```

The screen below is the structure of SLSREC. It will be used to save 3 items. The ID number of the member requesting disks, the date the disks were shipped and the quantity of disks that were shipped to that ID number, or member.

```
CREATED 04/01/89 CHANGED 04/05/89
FIELD  DESCRIPTOR  TYPE  WIDTH  DEC
  1      ID          C    007
  2      SDT         D    008
  3      QTS         N    004    00

        000 1 SLSREC  00000/00063
```

The SLSREC database should be created, but left empty. The CF will fill in the data automatically each time disks are shipped. TNAMES is opened last, and I hope , needs no further explanation. The statement REPLACE MORE WITH "Y" will get us into the WHILE loop. "I hope that most of this standard stuff is familiar to you by now." You are then asked to enter a number for the NM field you wish to FIND. If you enter a zero, all databases will be closed and the CF will be ended. That's a quick way out that I may modify later. If you enter a good number it will be found by TI-Base, the statement IF (NM = SEL2) will be true and DO DSK2.DSKNAP1 will be executed. DSHNAP1 is the Command File I created to display the name, if found, on the screen so you can decide if it is the name you want. However, I am going to leave that until next month's tutorial.

TI-Artist Instances To TI-Base

Wes Richardson has said that he would attempt to write a program to convert TI-Artist Instances into a format that can be imported into a TI-Base Database using the Convert function. Knowing Wes' past record I would eliminate the word attempt from that statement. The creation of this type of program will open up a new world to the TI-Base user. There are currently large quantities of graphics available to everyone. There are also many program arrund to change those graphics to TI-Artist Instances. A program to change Artist Instances to data in a TI-Base database would give TI-Base users more tools in this area than had ever been imagined before. If everything goes well, the conversion program will be published in the NorthCoast Newsletter in the next couple months. Hopefully that timing will bring the program out at the end of my mini series on graphics and exactly when you are ready for it.

Horizon RAMdisks

I must throw in a plug for Bud Mills. I forced myself to do this graphic series on a standard disk drive to experience the speed of the system. I must say that it is too slow, and too noisy. If you have a real need for a database system like TI-Base then you will probably put your TI99/4A through some heavy use. In that case a Horizon RAMdisk is the only way to go. Bud Mills has been unbelievably helpful and supportive to me for the whole time I have known him, and I hear the same story from other people. His RAMdisks are a top quality item, they are super fast compared to a normal disk drive and they do not make a sound. If you are interested, I recomend that you call Bud at (419) 385-5946 and get further information. o

continued from page 10

```
370 D$=D$&CHR$(130)&CHR$(155)&CHR$(75)&CHR$(194)&
  CHR$(200)&CHR$(2)&CHR$(54)&CHR$(52)&CHR$(134)
380 D=B :: FOR B=B+10 TO B+10*C STEP 10 :: IF B>D+10
  THEN D$=D$&C$
390 D$=D$&CHR$(201):: GOSUB 450 :: NEXT B :: B=D
400FOR D=@ TO C :: F=24 :: G=9 :: GOSUB 460 :: D$=D$&
  B$&CHR$(A(D))&A$(D)& CHR$(130)&CHR$(169)&CHR$(199)
410 D$=D$&CHR$(A(D)+5)&CHR$(68)&CHR$(83)&CHR$(75)&
  CHR$(49)&CHR$(46)&A$(D):: NEXT D
420 PRINT #@:D$&CHR$(0):CHR$(255)&CHR$(255):: CLOSE #@
430 DISPLAY AT(16,@)BEEP:"LOAD program made!": :"Enter
  the following commands": :" >NEW":" >MERGE
  DSK1.XXX":" >SAVE DSK1.LOAD":"  >RUN" :: STOP
440 B=B+10 :: PRINT #@:D$&CHR$(0):: D$="" :: DISPLAY
  AT(16,20):B
450 D$=D$&CHR$(INT(B/256))&CHR$(B-256*INT(B/256))::
  RETURN
460 GOSUB 440 :: D$=D$&CHR$(162)&CHR$(240)&CHR$(183)::
  GOSUB 470 :: D$=D$&C$ :: F=G
470 IF F<10 THEN D$=D$&CHR$(200)&CHR$(@)&CHR$(48+F)::
  RETURN ELSE D$=D$&CHR$(200)&CHR$(2)&CHR$(48+
  INT(F/10))&CHR$(48+F-10*INT(F/10)):: RETURN
```
o

continued from page 11

the specified location, the character of the keypress followed by enough blanks to erase the default string.

In line 10003, if the length of the validation string is 0 (it is an undefined null string) a normal ACCEPT AT accepts input in the first space following that first character, accepting up to one less than the specified maximum size, because that first character is not yet counted; but if the validation string was defined, a separate ACCEPT AT accepts and validates the input.

In line 10004, the character input by the CALL KEY is added to the accepted string, and the subprogram returns control to the main program. o

TI-Base Tutorial #9 pt 2

by Martin Smoley, North Coast 99ers USA

I hate to have to make a correction already, but I must. Because I have the update chips in my TI 99/4 Impact Printer, I was able to set my line spacing for graphics in 216ths of an inch. The original TI Impact Printer is restricted to line spacing in 72nds of an inch. I cannot go into it at this time, so if your graphic prints out in separate segments, check your printer manual for the proper codes.

The CF below is the last item I mentioned last month. It demonstrates a couple new tricks. One: Instead of using local space, we are displaying fields directly to specific screen locations. Two: Each time DSKNAP1 runs, it will display the current Running TOTal on disks, which is kept in slot 3. This total will be updated by the CF INVUPDT. Everything else is fairly simple. You are asked for the number of disks requested, or the number to be shipped. The answer is stored in ANS and if greater that zero DSKPRL1 is run. DSKPRL1 is the CF all our Graphics people will be interested in. It is very simple in its straight through nature, but complicated in its use of data and commands, which are accessed throughout the TI-Base system. NOTE: In an effort to help you make some sense out of this information I will reference past tutorials with the use of brackets. If you encounter a (TUT 8) for example, this means that a particular item, or idea, was covered in tutorial 8.

```
*                    DSKNAP1
    CLEAR
   WRITE 5,9,"    NAME FOUND"
   WRITE 9,3," Exp. Date    ",XP
   WRITE 12,3,FN
   WRITE 12,18,LN
   WRITE 14,3,SA
   WRITE 16,3,CT
   WRITE 16,22,ST
   WRITE 16,25,ZP
WRITE 20,1," Disks in stock = ",3.RTOT
  WRITE 22,1," Disks requested = "
 READ 22,22,ANS
WRITE 22,1,"                              "
IF ANS > 0
  DO DSK2.DSKPRL1
ENDIF
 CLEAR
RETURN
*
* DSKNAP1    Save as  DSKNAP1/C
* *******           03/31/89
```

The first thing done in DSKPRL1 is a check to see if our stock of disks has fallen below 50. If it has, the CF NOTE1 is run. NOTE1, which I listed on the next page, puts the message, "ORDER MORE DISKS", in the upper right corner of the screen. The message is stored in the Db MSRET, which we have opened in slot #4, (TUT 9). The first PRINT statement starts the interesting stuff. PRINT 4.CD2 means print whatever is in field CD2, in the Db we have opened in slot 4. Once again we are using MSRET, but this time the field is designated X-type, (TUT 9). The field contains 1B3324, and a bunch of zeros. TIB will interpret this as Hex code because of the X designation, and send it to the printer. The printer will see it as the Hex Code to change the line spacing to 36/216ths of an inch, (TUT 9).

```
*                            DSKPRL1
* Copyright Martin A. Smoley 1989
*
   IF 3.RTOT < 50
    DO DSK2.NOTE1
   ENDIF
     PRINT 4.CD2,(e),(E),4.TO,(Drft)
    REPLACE TEMP1 WITH "              ";
      | " Exp. Date " | XP
     PRINT (E),TEMP1
    REPLACE TEMP1 WITH TRIM(FN) | " ";
           | MI | " " | LN
     PRINT TEMP1
     PRINT SA
    REPLACE TEMP1 WITH TRIM(CT) | " ";
           | ST | ". " | ZP
     PRINT TEMP1
     PRINT (CR),(LF),4.CD1
*
   SELECT 5
   FIND "OHIO"
   PRINT (E),(e),4.FR,(Drft)
   PRINT (CR),(LF),(E)
   PRINT 5.GR1,(CR),5.GR1,4.NAME
   PRINT 5.GR2,(CR),5.GR2,(CR),(LF)
   PRINT 5.GR3,(CR),5.GR3,4.STREET
   PRINT 5.GR4,(CR),5.GR4,(CR),(LF)
   PRINT 5.GR5,(CR),5.GR5,4.CTSTZP
   PRINT (CR),(LF),(CR),(LF),(Drft)
*
   FIND "DISK"
  PRINT 4.CD1,(E),(e),4.CTN,(CR),(e),;
    4.CTN
   PRINT (CR),(LF),(E)
   PRINT 5.GR1,(CR),5.GR1,4.CD
   PRINT 5.GR2,(CR),5.GR2,(CR),(LF)
   PRINT 5.GR3,(CR),5.GR3,4.DNB
   PRINT 5.GR4,(CR),5.GR4,(CR),(LF)
   PRINT 5.GR5,(CR),5.GR5,4.DNX
   PRINT (CR),(LF),(CR),(LF),(Drft)
   SELECT 1
RETURN
*
* DSKPRL1    Save as  DSKPRL1/C
* *******           04/01/89
```

36/216ths inch amounts to normal line spacing. The zeros are used for fill and will have no affect on the printer. NOTE: zeros are the only value that will have no affect on the printer. (e) is my printer Control Code for one line Enlarged Print, and (E) for Emphasized print, [8.1]. All this is to print TO: in Enlarged and Emphasized mode, at normal line spacing. The (Drft) then changes the print type back to Draft Mode. All the work you just went through is to demonstrate the ability to use data, built in printer controls, and controls of your own from special data fields in the same statement. The REPLACE statement is standard. We are placing some words and the persons expiration date into TEMP1. The print line changes the printer to Emphasized, (E), and prints everything we placed in TEMP1. The rest of this is standard, down to the PRINT (CR),(LF),4.CD1. In the past I initialised a BLANK and then printed it for blank lines. (CR) and (LF) will do the same job with no need for memory space. 4.CD1 sets the line spacing to 24/216ths inch, for the graphics. "AH the Graphics." SELECT 5 means make slot 5 TIB's main slot for a while. This is necessary for the FIND function. Our graphics are located in the DB named GRF1, in slot 5, TIB-TUT 9. It is sorted on the field GRFNM which allows us to find the graphic data by it name, ie; OHIO. This technique would work quickly for a larger number of graphics and the names could be menu selectable. After OHIO is found we print From:, in the same manner as above, feed one line and reset Emphasized. The (E) will not affect the graphic, but will affect my name in 4.NAME. We then print 5.GR1, the first line of graphic, do a Carriage return with no line feed and print 5.GR1 over again. This is a Double Struck Graphic.

Then we print my name, 4.NAME, in normal Emphasized mode. The next line prints the next part of the graphic with no text after it. Here is an important note. The graphic data I have set up in GRF1 has no CR or LF attached. TIB does not hang CR, LF on the end of X type fields. Actually it is better that way. However, this means that you must send a CR and LF after each graphic. The LF is the one that really fires the graphic print. This need not be done if normal data is sent after the graphic, because it will have its own CR, LF on the end. If you compare the different graphic print lines, you will see what I mean. You should also go over TUT 9 a couple times. The last line in this group prints a couple blank lines to get to the next blank label. This is something you must remember when working with 15/16" labels. It is exactly one inch from the top line on a label, to where you print the top line on the next label. If you are working with 216ths inch increments, you must move 216/216ths to get to the next label. During the time we are printing the graphic label, we are feeding lines of 24/216ths each. This means we can feed 9 lines total to get to the next label, 9X24=216. If it does not work out that neatly, you can do a one time line feed at the end of the label to make up the difference. This value can be placed in an X-type field and used wherever you need it. I know this stuff sounds complicated and maybe even trivial, but if you are short even one or two 216ths, the printing on your label will slowly creep up until it is off the label. I have had this happen many times and you have probably had the same problem.

The next label prints out in the same manner, but the data is changed. Because of the FIND "DISK" statement, when GR1, GR2, etc. are printed, we now get the disk graphic instead of OHIO. The SELECT 1 statement points TIB back at slot 1 before the RETURN to the previous CF. It is then possible to find another name and do the whole thing again. One thing I would like to cover is INVUPDT. This small CF is in charge of keeping track of the disk supply and who got what on which day. The first step is to SELECT 2 and APPEND a BLANK. This means we are working with SLSREC, which is in slot 2, and we have added some space to it so we can save some data to that space. The next three lines save the ID number, the present date and the quantity of disks shipped, from the members data named on the mailing label, to the SLSREC Database. This data could be used later to find out how many disks were shipped to whom, on what dates.

That is about it for this month. The LIMA meeting date is approaching rapidly and I have not prepared my demo yet. Along with the pages IN TUT 9, I have included a TI KEY/CHARACTER CHART by JIM SWEDLW, I hope there will be room for it in the newsletter. I have been using this chart constantly to convert ASCII values to HEX. If you are working with the X-type fields, the only Hex you should need are >0 to >FF, or 0 to 255. This chart is for XBasic, but it really helps with everything we have discussed in the last couple tutorials.

NOTE: I intend to do a few more things with this graphics series, and the retrieval of information from multiple databases simultaneously. Keep an eye on this column for the next few months.

```
* Copyright Martin A. Smoley 1989
*                            NOTE1
WRITE 2,28,4.MS1
WRITE 4,28,4.MS2
WRITE 6,28,4.MS3
WRITE 8,28,4.MS4
WRITE 10,28,4.MS1
RETURN
* Print a message from the DataBase
* in slot 4.  This is CF  NOTE1/C
* DB in  slot 4  should be  MSRET
*
***********************************

* Copyright Martin A. Smoley 1989
*
*                          INVUPDT
SELECT 2
APPEND BLANK
REPLACE ID WITH 1.ID
REPLACE SDT WITH .DATE.
```

continued on page 14

Air Taxi by Don Shorock

reviewed by Jim Peterson, Ohio, USA

I have always wished that there were more educational programs, above the 2+2=? level, for our computer. And I have always thought that the best educational programs were those that took advantage of computer capabilities to entertain while teaching.

Also, I have always much preferred games that require me to exercise my mind, rather than depending on quick reaction or blind guessing. And, being a programmer, I admire efficient, memory-saving programming.

All that is why I was so very impressed by the new game, Air Taxi, recently released by Don Shorock. It is uniquely educational, very entertaining, and so compactly programmed that the basic version is available on cassette!

The game can be played alone, as it usually will be, or by up to 8 players. Don customises each game with the default names of whatever number of players you choose and with your home town as the starting point. Each player may select his own handicap level, ranging from A to Z for 6 to 81 cities, and his skill level ranging from 1 to 9 which determines the target size.

A black silhouette map of the entire United States and southern Canada is then displayed; the only features are the Great Lakes, Great Salt Lake, and the coast lines. You are randomly offered a destination to fly to. Since all your friends bum rides from you, and TI users are cheapskates (that is my comment, not Don's!), you are not even paid for your gas for this first trip. It may therefore pay you to refuse any offer to a distant destination - however, each refusal costs you $2.00.

When you accept an offer, you then use the S and D keys to set your initial flight direction, in 45 degree increments (i.e., north, northeast, east, etc.) and press Q. You hear the sound of the motor revving up, and a small cursor dot begins moving from your town in the direction you selected, while your gas gauge shows your fuel being used up. You can use the S and D keys to change direction. If you get close enough (depending on the skill level you selected) before your fuel runs out, the cursor will stop, the motor revs down, and you will be shown the cost of the fuel expended and your remaining bank balance. If your fuel runs out to soon, you will glide to the nearest airport and you must then set your direction from that point and try to reach your original destination. However, if you were too far from any airport when your gas tank ran dry, you will be returned to your home town and will be assessed repair costs.

Once you have reached your first destination and said goodbye to your freeloading friends, you will then be randomly offered fares, at prices depending on distance, from that point to another city. You have the option to refuse offers, at a cost of $2.00. If you can fly to that point with a minimum of maneuvering, the fare will more than cover the cost of fuel, and you will make money - plus an occasional tip.

There are too many other features to describe here. The program comes with four pages of printed documentation, and the disk version includes three additional files, which can be merged in, to add many more cities or to convert the program for use with a joystick. At the handicap and skill level K 7 which Don set for me as defaults, I found that I was able to stay ahead of the game by refusing most fares except coastal cities and then cruising along the coast until the airport radar picked me up and brought me in. Trying to find Kansas City or Cheyenne on that black silhouette map would be very difficult without consulting a regular map - and in doing so, you would learn a great deal about the relative location of cities.

This is a commercial program, not fairware, and it is customised for each purchaser. The price is $15 for the disk version, $20 for the cassette version. To get an order form, on which you can specify your own default options, write to Don Shorock, P.O. Box 501, Great Bend KS 67530. o

TI*MES Utility Catalogue part 2

compiled by Stephen Shaw, England

>PLUS! Two disks from Jack Sughrue which gained an A+ review in Micropendium. LOTS of docs. Seems to be IFfing transliterate files. IGNORE docs on Funlweb which are out of date. Offered purely for anyone who saw the review and wants it. No refunds! DO NOT ask me anything about these disks!

> POSTERS. Hard work for your printer with this one as pictures are produced from DV80 files. We will not mention Anna (!) but there is a nice Madonna (no, the original) and a lovely LONG Christmas poster as well as Love... and a puppy printing, no need for TI Writer.

>PR-BASE. Version 2.0: double sided capability and easier to use.---TWO DISKS PLEASE!!!! Please state version! At last, a Freeware Data Base program! This program is by William Warren and use 1 disk sector per record,of up to 250 characters spread between up to 32 fields. Many pages of documentation so TWO DISKS required. Three different ways to sort. Tabular printouts or mailing label format.

>PR BASE Vn 2.1- ASK FOR VN 2.1!!! Two disks. Not an official release. Amended by Mike Dodd to run on a Geneve, and also make data disks easier to copy - but at the cost of making them incompatible with Vn 2.0 data disks. Complex conversion instructions to change 2.0 data disks to 2.1 format. Use only XB load- the UTIL1 file appears incompatible with anything!

>PRK CALLS DEMOS: A variety of programs in Basic which REQUIRE the PRK or Stats modules or library disk MODUTIL. PRKCONVERT will convert PRK files into two files which a Basic program can use- PRKHEADER which is IV80 and PRKDATA which is IV(n). Sample PRKDATA and PRKHEADER are supplied for a mail list program, with basic files ADD/DATA to use the database, B/PRKPRINT and PRKLABEL to print out to printer or screen. PRKWRITE is an inverse conversion program and will change the PRKDATA and PRKHEADER files back into a PRK file.

>PRKADDRESS and READ/NAMES are similar Basic programs but use data files of IV135 format (not supplied). MAIL/LIST is a PRK file. I have added my own PRK utility, which transfers data from a PRK file to a DV80 file for TI Writer to use. Examination of these programs will show you how to use the extra calls.

>PERSONAL RECORD KEEPING/BASIC. The full module program in Basic! (As it is 20k, it is on disk as XB!) PLUS merge files for parts of the program, AND details of all those lovely new calls, INCLUDING THE LOW HEX CALLS never documented before! EG CALL >05 and how you can use them. From Jan Alexanderson, Sweden.

>PULSAR by Mike Amundsen. A collection of 9900 m/code routines to include in your OWN m/c programs. Documentation is in the source code. Files include dataset, float, graph1, ifset, init-ea, intmath, keyset, loopset, randset, scrn-io, and start-ea plus sample "complete" code: INPUT, RANDOM, READ. These routines let you write m/c using words familiar from Basic.

>PICASSO PUBLISHER Vn 1.1 by Arto Heino. A bit map drawing program with lots of plusses. Draw on a "screen" 336 x 480 pixels. 32 brushes. 32 textures. 5 icons. Six fonts. (Copyright on Vn 2.0 has now been claimed by Asgard, with EXCLUSIVE selling rights owned by TENEX. This version stated as public domain by Asgard.)

>QUICK RUN DEMO DISK 1: The official demo disk for the commercial program QUICK RUN, which takes "snap shots" of running programs, which then start so much faster when reloaded again.

>QUICK RUN DEMO DISK 2: I have made this one up- it includes the original program for comparison. The original took 127 seconds to load and run- the quick-run version takes just 25.5 seconds! This disk will demonstrate the effect!

>RAG CASSETTE LOADER for saving/loading memory image machine code programs (32k ram required). Optional suppression of normal double recording (doubles speed) and also compression for really fast speed.

>RAGMAC MACRO ASSEMBLER from RAG SOFTWARE (R A Green). =VERSION 7.0 THREE DISKS!!! A replacement for TI's ASSEMBLER, this package adds a macro facility - this is not a package for novices. Please do not seek technical advice from me!

Adds: macro facility, improved listing format, improved diagnostic format, cross reference listing, and can run from XB. Documentation on disk. A quoted string may use " OR "

COPY may use * as disk number- same disk will be used as Source disk.

Supplied set up for GEMINI printer - INSTALLATION PROGRAM supplied. Output to disk as well as printer.

Macro library supplied: Branch Equal, Branch Not Equal, IF word, IF byte, MOVE bytes long, SET vdp address, issue accept/reject tone, GET record, INPUT from keyboard, PUT record, PRINT to screen, OPEN DCB, CLOSE DCB, Data Control Block, Define PAB, Screen Control Block, and labels for PAB fields. CALL subroutines with parameter list, RCALL subroutines with parameters in registers, LDP load byte value into register, SETV set vdp address, SETSW set switch on/off, IFSW-test switch.

3 DISKS PLEASE!

>RAG UTILITIES. Vn 6. Produce a formatted listing of an XB program with cross reference of variables and statement numbers. Sector based disk copy. Disk initialiser. Program to copy all PROGRAM type files on a disk to cassette without constant intervention! Printer initialisation. Dump ASCII terminal 300/1200. Disk catalogue (255 disks). Shorten GK files. Print file.

>REBEL (TWO DISKS)- Lots of utilities and source code- Cassette Backup- copy a whole tape from CS1 to CS2-essentially an audio copy via the consoles audio circuits. Cassette Builder-uses a DV80 control file to copy several disk files onto one cassette automatically. CRU TESTER- both educational and diagnostic. DNAME-a m/c utility to incorporate into your own m/c programs, allows input of a disk name, checks validity and calculates length. Disk Buffer-allows LONG TI Basic programs to transfer from tape to disk, even when CALL FILES(1) fails to work. QUICKSAVE will change a DF80 file WRITTEN FOR XB into what looks like an XB program. SMALL LIST (also to be found on UTIL-21),allows you to list XB programs in 28 column format. SPEECH looks through the speech synth and speaks every word it can find.

>ROMSTADT LINKS 1...assembly routines to load using ExBas (CALL LOAD) and use with CALL LINKs, including a multi line accept at; instant change a range of colour sets; font amendment routines; a unique routine to GET the colours of colour sets; a routine which will change any upper case in a string to lower case; a routine to place character patterns 63 to 143 into a string array; a routine to place screen contents into a string array; and a routine to place up to five screens into low mem for instant access. There is also a LOAD program with imbedded machine code for that extra touch (for XB Programs or to read text files or even print a catalog!). FULL COMMENTED SOURCE CODE.

>ROMSTADT LINKS 2...More assembly routines to use from ExBas... two text screen dumps, single size double density (4 cycles/line) and double sided double density(5 cycles/line, omitting first and last screen columns. A routine to insert spaces between letters of text; four non-horizontal display routines; and a suite of eight routines to use 40 column mode. FULL COMMENTED SOURCE CODE.

>SHAMUS MAPS- for Picasso, four maps showing where extra lives are, where the keys for which door are. Includes an XB printer routine for Epson printers which is SLOW or use Picasso to print.

>SIDE*PRINT Vn 3.1 by Jim Swedlow. A program which will print MULTIPLAN files SIDEWAYS! ...

>SIDEWRITER Vn 2.1 by Mauro Tomietto. Not JUST a program to print TIW and Multiplan sideways... an excellent program that prints sideways- choices are: font style (can use with disk UTIL-8); four horizontal spacings between characters, micro vertical adjustment, and a disk directory. Beautifully written and includes Source code. Now you can print TIW documents with all sorts of character styles! I like this one.

>SmArtCopy by Alexander Hulpke, allows you to print TI ARTIST and also MYART format pictures on a TI99/4A. Magnify up to 999 times in each direction after clipping pic, and then glue together for wallpaper! SEE MYART FILE FOR MORE DETAIL.

>SORT + DUMP. Two programs only! REVISED MARCH 88. Now includes MEGASORT96 for the Geneve, and improved sort routines for the 4a.
Sort Experiment by J P Hoddie, sorts up to 1000 records in ANY type of disk file, with up to 8 sorting keys. NB: Memory image file and docs are revised. Source code is unrevised. DUMP by Wayne Stith is to read and display/print ANY memory location: GROM, ROM, VDP, DSR. Not a very neat display but usable and supplied WITH source code.

Note- Spectrum disks below will not run on consoles fitted with a modified GROM 0, the one which gives odd characters on early Atarisoft modules such as Picnic Paranoia.

>SPECTRUM 1: A program to pick up the graphics from the START of a Spectrum cassette. Spectrum specification colour pictures are then stored in a compact format. A utility program to display these pictures is included - on this and next two disks. Also a program to transfer from this format to TI ARTIST format. And provided the TI Artist pic conforms to Spectrum specification, to transfer from TIA to the more compact format used on these disks. Also title screens from Spectrum games: Cookie, Mugsy, Ad Astra, Jack and the Beanstalk, Combat Zone, Chequered Flag, Mr Wimpy, The Hobbit, Harrier Attack, Auto Mania, Pyjamerama.

>SPECTRUM 2: Title screens from: Horace and the Spiders, Nightshade, Spy Hunter, Bruce Lee, Saboteur, Daley Thompson's Super Test, Odin, Sam Fox Strip Poker, Molecule Man, Merlin Rack, Gladiator, and Rambo 2.

>SPECTRUM 3: Title screens from:
Bomb Jack, Cobra, ?, Spitfire 40, Asterix, Slap Fight, FTL Gargoyle, Wonder Boy, Hulk, Magic Land, Jet Pac and Scuba Dive.

>SPECTRUM 4: Screens from Target Renegade Rudy; Pssst; Raid over Moscow; Bubble Bobble; Driller; Thundercats; Coconut Capery; Knight Lore; two unidentified; and the RLE Tiger in COLOR (Spectrum spec).

>SPELL & SORT: Very simple utilities from Software Specialities Inc, copied by TI99/4A USERS GROUP with the express consent of the copyright owner. SPELL is a spell checker for DV80 files, while SORT is a powerful general purpose sorter, which can sort ANY file on several keys. Any length, variable or fixed. An intermediate file is created, which is always FIXED, and may be quite long if the input/output files are Variable! ALWAYS SPECIFY ONE KEY- if you leave all keys set at 0 it will only remove blank fields!

>STAR by Michael Riccio of COM-LINK Enterprises. A full range of CALL LINK utilities for TI Extended Basic. If you program in TI XB or would like to improve existing TI XB programs, you need this disk. 53 CALL LINK routines, including: screen save/load, bye, new, quiton, quitoff, charset without colour change, title screen large caps, true lower case set, chimes, flashing text, vdp peek and poke, screen display on and off, read and write array to screen, instant sprite start/stop, check to see if alpha lock is engaged or shift/control/ function keys are depressed, character copy, magnify, rotate, flip, mirror, invert, disk file protect/ unprotect, disk catalogue (NOT RAMDISK), file rename, read and write sectors, 40 column mode plus 40 column PRINT, string reverse, and change strings to all capitals. Phew! PLUS SAMPLE PROGRAMS!

>STATISTICS/BASIC. The entire Stats module in the form of a BASIC listing (files are XB) with parts as MERGE files, and full documentation on all the extra CALLS that the STATS module makes available to you in Basic. The actual stats calculations are PURE BASIC so you can extract whatever bits you want.

>STATISTICS/MC. FOUR DISKS PLEASE. At last, the statistics module is available on disk! Requires an understanding of stats! All module functions available. Descriptive docs on the disks but you must know your stats!

> SUPER BUGGER 2 (Dohlmann). SBUG and DEBUG put together, with 6 debugs and 9 enhancements. AND DOCUMENTATION running to 40 pages! Memory Dumps and Dissassemblies to disk or printer, with or without address. ExBas load can now handle a 6k file of yours! Change file name in use! Block transfer SLOWLY (for EPROM, 10 to 20ms per byte). Change screen colours. The A,G,I and V instructions of DEBUG have been omitted due to length considerations. This program occupies 8k and is supplied in 3 formats: Relocatable condensed format code for EdAs and MM. A version for EdAs or XB to load into >A000 to >BFFF. And a version to load with EdAs into >6000 to >7FFF, IF YOU HAVE IT, complete with GROM HEADER- this last version is for Morningstar 128k card, either SUPER SPACE or SUPER CART modules. Documentation on disk but a printed copy is available at low cost from the author. Details supplied.
The above is Vn 1.0 - Version 2 is now available direct from the author for US$13 and includes a 52 page printed manual. Changes include amending list device, screen colours. loading and saving memory image format files, string searches and GROM base change. Write: Edgar Dohmann, Route 5, Box 84, Alvin, Texas, USA, 77511

>SYSTEM DISK LOADER (TWO DISKS) by EP Rebel, includes source code. Produces a custom menu as UTIL1 and has loaders for XB and MM. Loads only memory image machine code programs. No options so some it loads, some it does not - Scrabble is OK but Superfly is not. An alternative to Joe Nollans disk. Also a suite of CALL LINKs for XB programmers, to play with the screen display-the links are BIRMIR, BITREV, BITTRN, DELETE, DEMO, DOWN, DUMMY, HELP, INSERT, INVERT, LEFT, MIRROR, RANDOM, RIGHT, ROTATE, ROTNEG, ROTPOS, SCROLL, SORT, SWAP, SWPMIR, SWPTRN, TURN and UP, which allow you to do pretty well everything with a screen display! Up down left and right move the whole screen image and also wrap around.

>TASS (Tri Artist Slide Show) Vn 1.0 A program which will read and display a mixed disk of Graphx, TI Artist and Draw- A-Bit 2 pictures automatically. Can sequence several drives. Includes a "lines" program too.

>TE2 PROGRAMS-1. A collection of programs for TE2 owners. Many featuring Rock the Robot, who teaches addition,the alphabet, counting, division and subtraction, as well as singing OLD McDONALD and reciting nursery rhymes. A word game DUNKMAN. An animated Gettysburg Address, a rendition of Daisy,Daisy... and a good laugh.

>TEXTLOADER+EA5LOAD...from Paragon. Vn 1.2, Jan 89. Hard disk compatible. The textloader is something many have tried to do for years... now its done. Run TEXTLOADER and a DV80 text file is read into the console JUST as though you had typed it in. You can quickly load a program on disk as text, OR feed in a string of command mode instructions (do both together!). The EA5LOADER loads machine code memory image programs using XB, and comes complete with SOURCE code.

>TI ARTIST BITS. This disk contains a CARTOONKIT

by Tim O'Neill, composed of a set of instances and several sets of slides, together several 7 and 8 bit high fonts. This disk requires TI ARTIST Vn 2.0 to be of use to you!!

TI ARTIST INSTANCES-lots of clipart. Largest instance is 12 sectors.

>MFART1, 42 files
>MFART2,3,4 and 5, all 43 files each.
>MFART6, 41 files.
>MFART7, 14 files (113 sectors used).

>TI BASE TUTORIAL BY SMOLEY. FOUR DISKS. Nine tutorial articles written Nov 88 to Apr 89, with sample databases and command files.

>TI BASE TUTORIAL BY GASKILL. Covers Version 1 only.

>TI BASE USER GROUP DATA BASE: BY ANDI WISE. A sample application of TI Base with command files for you to inspect.

>TI Editor Assembler: To load and run from Extended Basic. Editor Assembler module NOT needed. MANUAL NOT INCLUDED and not available from me! Disk includes a French dissassembler an Italian utility program. This disk is known as BEAXS. Revision 1/86: To operate with any disk controller.

>TI Editor Assembler PLUS TI Writer, on one disk, to load with MINI MEMORY module (32k ram still required!!). Program from Belgium, but text in English. MANUALS not included and not available.

>TI MATH. Combination of the two TI disks MATH LIBRARY and ELECTRICAL ENGINEERING LIBRARY with programs in XB for base conversions, primes, hyperbolics, ordinary differential equations, matrices, fourier, simultaneous equations, PLL, Smith Chart, Filters and Root Locus. NB: ONLY ONE DISK!!!!

>TI SINGS from TRIO+ SOFTWARE by Barb Berg, now Freeware. Enables you to make your computer sing, using TE2 module and speech synth. Plenty of helpful docs on speech creation and several sample tunes.

>TI TEST DISKS (Two disks) A suite of programs to test your system, released to public domain by TI in 1986! Tests most parts of the TI system including the P-code card. Not suitable for testing third party peripherals. NB: Requires use of MINI MEMORY MODULE. Some tests also use TI XB plus 32k ram. INCLUDES SOURCE CODE for the machine code programs.

>EXTENDED DOCS FOR TEST DISKS- not offered by TI, and no room on the above two disks, a US user group has kindly made available a more full and technical set of docs for the above disks. Not essential but may be interesting.

>TI WRITER VERSION 4.3 By Art Green. Rewritten editor and formatter- editor MUCH faster for move, copy, replace, delete, and new Formatter commands. Special menu retaining loaders for minimem and super space. Will load from Funlweb but crashes on exit.

>TONY_MCG1: An article by Tony McGovern on writing efficient machine code, Plus modified EDITOR files for Funlweb, for use with an AVPC card, to give true 80 column display. Also works on a Geneve with 80 columns, or with a Mechatronic card. (Jul 89 issue) plus an 80 column version of QD for an 80 column directory display.

>TONY_MCG2. From Tony, a review of the Dijit AVPC card, and a modified ROS for the HRD, which is more friendly to Funlweb, and if also required if you have an AVPC card. Also a modified ED file for Funlweb 4.13 which offers a TAB function with CTRL Z. From elsewhere, a review of Fortran99, an article on making your joysticks work regardless of the alpha lock, and a long article on fault finding and correction in the console, which assumes you have a circuit diagram and a test meter.

>TONY_MCG3. Articles on DSRlinks and also on interfacing your machine code programs to Funlweb.

>TRIVIA 99er by Robert Wessler. Comparable to the various trivia BOOKS not to the game!!! Can handle an inordinate number of questions (supplied with480 to start you off), and consists of: Quiz program, File creator, File editor, File Printer, and specimen files of questions (and answers). The computer operator decides whether the answer given (verbally) is close enough to the answer stored by the computer in deciding on scoring. The computer acts in a similar way to the books, but selects the files and questions at random, keeps tabs of the scores, and you can write your own questions and answers in. If you like trivia games you will probably like this program. Could be used with youngsters by creating suitable files.

TI-PEWRITER PLUS NAME-IT. A word processor which allows cassette input/output, and a mailing list utility (disk only) for use with it. No docs but see sample files. Not hard to work out.

>UNIVERSAL DISASSEMBLER by Rene LeBlanc. Vn 2.3 Written in FORTH, the disk can be loaded from Ed/As or Mini Memory module, or from any TI Forth by using COLD. This powerful program can disassemble machine code ON DISK in any format: DF80 compacted or uncompacted, or memory image. It can also disassemble the contents of Mini Memory ram, disassemble console rom, and dump console rom and VDP Ram. Disk utilities are included to trace the sectors to be disassembled, included hex and ASCII search, and file analysis using the disk directory. Not quite perfect but a welcome addition to the utility collection. In tests I found this to be the most reliable disassembler, when handling DF80 disk files.

>UTILITIES-1: Inc: Disk initialisation, 28 column listing, 2 and 4 column printing, disk catalogue, a program to extract a routine from a larger program, a program to slash the zeros... even when typing in a program! and lots more goodies.

>UTILITIES-2: A number of disk catalguing utilities, including a machine code utility you can CALL LINK to in your ExBas programs, and return to YOUR program after use... machine code TEXT ONLY screen dumps ... and a utility to remove the automatic start on some DF80 machine code programs.

> UTILITIES 3: Machine code disk information Manager by Don Cook,similar to Disk Fixer, but also able to transfer m/c program files from disk to cassette. Disk Analyser by Ed Dohlmann. Another disk fixer, works in a similar manner to DEBUG, and includes many DEBUG commands. DISASSEMBLER by TI and DISKO by TI - the original disk fixer program!. Also a suite of fast disk copying programs for E/A and ExBas. NB:Most programs on this disk are for Ed/As or Mini Mem + 32k. Full documentation on disk.Disassembler for Mini Memory.

> UTILITIES 4: Ed Dohlmanns Sprite Editor, programs to Compact and Uncompact DF80 machine code files, a file reader, music writer, sound effects demo, librarian program, a word processor for MM or XB!, an electronic typewriter, article filer, and disk cataloguer. and others.

> UTILITIES 5: Two interrupt driven routines for XB module- a clock and a disk cataloguer you can call up just by pressing SHIFT and FCTN, An XB loader for machine code memory image files from cassette. A simulated machine code monitor, using 3 digit decimal words (this is one for you he-men hackers!) and Disk Manager 99, a machine code disk manager by Mike Dodd, which you use from your Basic/XB programs by CALL LINKs. DM99 is for EdAs, MM or XB modules. At Mike's request, DM99 is also available separately free of charge in return for a blank disk and return post and packing. (No extra packing added if you select this option!).

>UTILITIES 6. A turbo-load for INFOCOM Adventure disks, for EdAs or XB! Super Disk Cataloguer by Duke and

Beeker, each catalogue file can hold details of up to 1200 files. Super fast displays and sorting, very useful. Tiny calendar printer, sort demo, "sky at night (and day)" map, a special FLIPPY cataloguer, a program to extract routines from programs, and the piece de resistance, a DV80 to MULTIPLAN conversion program, with inbuilt disk sector writer to produce the odd hybrid files Multiplan needs. a Minimem disassembler.

>UTILITIES 7. Another full disk with 28 files. Programs include TWO to transfer machine code (written for XB) to be formatted as either a series of CALL LOADS (Program: ACE) or as a PROGRAM file with hidden code (Program: SYSTEX). Also the best text to XB program converter (XLATE) and a program to store screen contents in a machine code program, ready for subsequent use with a super fast CALL LINK. A program to convert small TI Artist Instances to TI Writer Formatter files, and a sample file (Snoopy). And a machine code interrupt driven clock for XB+32k, AMENDED FOR UK 50HZ MAINS!!! and also a SPEAKING CLOCK program (SP/CLOCK). AND STILL MORE!

>UTILITIES 8. A disk cataloguer to be called from your XB program with CALL LINK - and your screen display is restored after use. Several formats including a memory image file which can be saved to cassette. Several character sets in various formats for use in your programs. An XB disk copier for use with ONE disk drive, for SSSD disks only - very fast, only two passes. A cursor redefining utility. A 128k transfer program, and the FIRST Multiplan template to enter the list: a template using iteration to solve a second order differential equation -or which tells you what your new stable weight will be if you reduce your calorie intake.

>UTILITIES 9. A program to create disk menues, a mailing list program, two DV80 file reader programs with rapid scroll features, using 40 or 64 characters, a disk catalogue print program, a PRBASE utility program which produces graphs to assist you lay out PRBASE, and a program to convert ExBas graphics screens into TI ARTIST instances.

>UTILITIES 10. TI Disk Cat by Mack McCormick, a simple but FAST way to do a quick one-off list of all your disks, SORGAN, a fascinating sound synthesiser, and HYPHENATOR a program which allows you to insert hyphens into TI WRITER text before you run it through the Formatter option using FILL. You can also make very minor spelling changes! Then, for XB programmers, two machine code utilities to LINK to by Graham Marshall, one to find which element of a string array a particular string is in, and the other is a CALL COINC to indicate when a sprite is touching an ON pixel on the screen.

>UTILITIES 11 : CASSLOAD and CASSTRANS to enable you to move a machine code memory image file from disk to tape, and then to load and run it from tape with just XB and 32k. DISK HACKER PART 1 by Will McGovern, a disk analyser which reads FROM THE DISK for each track, track no, side no, sector no, sector length, CRC value, with results in decimal or hex; DISK AID by D M Thomson a sector reader utility with extras- the menu includes sector read, write, edit, move, compare, plus view CPU,GROM and VDP memory; plus a preliminary GPL disassembler by Paul Charlton (imperfect); SUPERSAVE by Erik Olson to convert most DF80 machine code files to memory image files- with a choice of what utilities to include, so most programs can be transferred (and then to tape with CASSTRANS!); and a machine code program to transfer an XB graphic screen to TI ARTIST format, and a few machine code sound to light demos...

>UTIL 12. TIW Utilities: an XB disk cataloguer which lists DV80 files first!, and two programs to count the number of words in a DV80 file, one m/c and one XB. Plus a program which provides sunrise/sunset times, and one called SOLAR for telescope owners; the TI disk manager Vn 2 now on disk, a revised m/c TI Artist Instance to XB transfer utility, NEATLIST- XB LOADING file, a revision of the NEATLIST disk program which will not fit on the original disk! Need NEATLIST disk for docs. Revision adds DV80 file output. Two specialised data bases- Address file/label printer (labels 3x8 on a sheet) and Ham Radio log. Also a two character high Large Character Set for your Basic programs. PLUS SUPERTRACE from Jim Peterson- an all-XB TRACE utility with output to printer and single step options. And a speech demo program which will give you the speech from MOONMINE at the press of a key!

[>UTIL 13. Withdrawn- John Birdwells DISK UTILITIES has now gone to Vn 4, on a disk of its own, listed elsewhere. The remaining programs which were on Util 13 have been moved to Util 17 and Util 19, listed on a later file]

>UTIL 14 : A DF80 editor which puts the right checksums in, a name and address database which stores 900 addresses in 39 disk sectors! (with source code), and source and object code and docs for S Michels joy-sketch program (for m/m but convertible) and his excellent screen scroll utility.

>UTILITIES 15: Archiver 2.1 by Barry Boone, to pack (and unpack) several files into a single file- keeps 'em all together. SNAP CALC, a 13x20 XB spreadsheet by Gary Strauss from HCM. TI Keys Vn 3.0 by Wes Johnson, instantly put up text on screen with CTRL 1 to 0 and A to Z, command mode or running; Prestel/Viditel Terminal Emulator (from Holland); XB by J P Hoddie- a machine code program to run with Funlwriter: its the same as RUN DSK1.LOAD so you do not have to quit to get back to XB; Tracker by Will McGovern, a track copy utility for owners of MYARC disk controllers; Unbasher by Barry Traver, (much revised March 1988) uncompacts those densely packed XB programs.

>UTILITIES 16: Several character sets in Source code, Object file and Merge file formats, a File reader,a Speech Tutor XB program, and M/COPY-(Vn 1.1)-> the program ALL disk owners should have! After you have repaired any fractured files using ordinary file copy, process your disk with MCOPY. If the disk has more than 32 files, MCOPY will place ALL the file descriptor blocks into a single disk area, vastly cutting down access time AND reducing drive wear. A must, especially for DD owners!

>UTILITIES 17: XBGC, a graphics program to translate from GRAPHX to CSGD and hence to TI ARTIST or PIO or MERGE format; 99-Calc, a small spreadsheet program, and a new Archiver (Vn 2.4 Jan 88) with a compression facility. A calendar program from MSP99 UG. CURSOR- a cursor redefinition utility.

>UTILITIES 18: One program only- CHARDES 5.2, a char/sprite design aid with a difference. LOTS of facilities, and fast to use. Can produce output as a MERGE format XB program! Save time!

>UTILITIES 19. (326 sectors used to date): Some machine code conversion routines from LA: object code to CALL LOAD, CALL LOAD to object code, recovery of code hidden in an XB program. A program to print graphs. And an XB utility to give you 8 strings at the touch of one key (in command mode). Marty Krolls machine code disassembler (formerly on the FastTerm disk), and INFOLISTER which will list the vocabulary for your Infocom adventures, from the GAME1 files. and SUPERMAIL, an address data base.

>UTILITIES 20: Utilities to allow MERGEing code in from tape (can be faster than disk MERGE!) and load/save to/from tape at double speed, also of XB programs up to 22k (32k req!); Character and Sprite Shape Maker, GiftLabel, and Procalc. Plus an XB (hidden code) version of Neatlister, by JP Hoddie. You will still need the NEATLIST disk for docs! And INSTANCE PRINER Vn2, which allows TI Artist instances to be printed in "correct" ratio- a circle looks like a circle! - a full screen instance takes up a full paper width- and print is very dense. A 40x24 Life universe at high speed, written in C by Mike Cavanagh.

>UTIL 21. Rewritten ARCHIVER program, now Version 3.03. Improvements include single step uncompress and

unpack (and vice versa). PRINT directory of compressed files! Directory includes total sectors used by compressed files! and archived file name. Plus... Assembly routines to LINK to from your XB programs- alpha lock key checker (neat); VPEEK,VPOKE,POKER from the Smart Programmer- read the commented source code for these!- plus a GPLLNK for XB use; another high speed tape loader; a bit-map utility source code for machine code programmers and an interrupt driven machine code routine for XB, to LIST programs to printer just 28 columns wide, just like on screen.

>UTIL22: A program to print cassette labels; a sector editor by Guy Boudreault; KwikFont, which is a quick (machine code!) character definer, with utility to transfer the characters to a CHARA1 file; a disk speed checker for Myarc controllers only, and two Logo utilities- one to make Logo procedures AUTOSTART- no more searching for start names! and one to print out the definitions of tiles and characters.(The LOGO utilities are in machine code and modify the original Logo files). Machine code utilities to use in your XB programs to quickly restore the definitions of lower case letters or to use the title screen BIG letters. Also supporting DSR/GPLLNK routine you can use in your own m/c utilities for XB. One line programs to catalogue a disk, display a DV80 file and count words, and THE BEST utility to reduce the size of a TI ARTIST picture, SQUEEZER, which is SUPERB. And a little XB program to print a years calendar on a single page. And a 6 memory calculator!

>UTIL 23. Not full yet but we have both disk and cassette versions of TI CALC in XB by Schmalhofer, slow but usable; plus a utility to print PICASSO 85 sector DV80 pictures without Picasso in XB, plus two machine code utilities for use with XB to print TI Artist pictures (ARTCOP and ARSMA by Hulpke) in quad density (ESC Z) in two heights- 65mm and 34mm,and in widths any multiple of 27mm (for true proportion use multiple 4 for the large size and multiple 2 for the small size). If you first use Squeezer on Util22 you can print out direct a TI Artist picture just 17mm x 27mm!!! DIY stamps, ideal for detailed labels.

>T-SHELL, by Travis Watford (remember RLE!) this is a super embedded machine code ExBas program that gives you a background environment- for XB programmers who would like -from the XB command line- while programming to catalog a disk to screen or printer, read a text file on screen or printer, sweep a disk, copy or rename a file, and protect or unprotect a file, without having to load a program which will destroy their XB program....; a routine by Bud Wright to be used in ExBas which changes lower case letters in a string to upper case (the opposite of a Romstadt routine above); and SPEECODER by Michael Zapf of Germany-a complex (!) utility for programming speech using the speech synth, enabling you to examine existing phrases in the synth, or in modules, and change the pitches, volumes and sounds, resaving the results to disk for later reuse, including in data statements in XB programs.

>X. This disk MUST not be used where children may be present nor should access be allowed to the disk. Animated pictures which although pythonesque may give offence. It is your choice. XB load.

>XB*TOOLS Vn 1.2 by Jim Swedlow. A group of programs to help you write in Extended Basic. They act on files saved in MERGE format to produce a reference list of variables, line transfers, subprogram calls, DATA and DIM lines; remove rem lines; join lines together; change the names of up to ten variables at a time to names you specify; replace variable and user subprogram names with one or two letter names, delete, keep or resequence a part of a program, move a block of lines, combine DATA lines ...

>XDP by Craig Sheehan, TWO DISKS- including 42 pages of docs and memory map. A utility to extend your XB programming, by adding 20 "LINK"s, which give you 12 extra charsets, 98 extra definable characters, 32/40 columns, windowing, pseudo-hi-res plotting, scroll and screen dump. Accept and Display have 14 sub commands ,

continued on page 29

Rambles

by Stephen Shaw, England

>UTIL24 amongst other goodies has...: the PIO printer test from TI*MES which in a running XB program tests to see if the printer is connected and ready, without hanging the program up; a program to print PAGE PRO saved pages with a denser print; a GLOBAL DISK SEARCH utility which will search a specified disk of DV80 files looking for a specific word (or phrase) and when it finds it in a record, provides a print out of the complete record (or records- eg lines!) with file name and line number(s). And a turbo copy program for TI disk controllers-note the warning & do not use to initialise double sided disks! It TRACK copies rather than file or sector copies. And a utility to merge the 40 column command files produced by TI BASE- especially useful for Version 3- the 40 column editor only allows 50 line files but the database program can use longer ones. ALSO...

The Miami User Groups BOOT utility-this is another "environment" shell, giving you a menu selection with disk catalogue and text read/print capability. It is very very neat-this version is dated 12.2.89... also a dedicated videotape database is added. (The BOOT environment now resides in my DSK1 with T Shell while Funlweb hides in disk 2, giving a very neat system to work with).

>TI*MES PROGRAMS- short programs and utilities from times-here is the file list with TI*MES issue number following the file name: AUTOGRAPH ?, BOMBER 29, CHANGECURS 21, CHURCHBELL 12, CLEARALL 26, COL/COMB 16, CORNWIPE 22, DEF/NSUB 26, DEFAULT 26, EQUATIONS 24, FIND/LAST 26, FLASHDATA 23, P 21 SEE ALSO UPSIDEDOWN, FLIP/DEMO 21, FONTMAKER 16, FRACMY ? FOR MYARC XB, GARBAGECOL 20, GET/KEY 25, HSCROLL 25, IKEDA ?, JBMGR ? FOR JBM103 GRAPHICS UTILITY, KALKULATOR 27, KEY/CHECK 14, KEYDISPLAY 17, LABELS 24, LET/SPRITE 7, LOW/UPCASE 16, NOISE 21 trick program!, NUM/COLOR 19, OLC ?, PRK/DV80 27 FOR PRK OR STATS MODULES USING TI BASIC, PRTCHK +, PRTCHK/1 +, PRTCHK/A +, PRTCHK/B +, PRTCO all 25, PUTAT/1 25, READ-D/80 24, RJBM ? FOR JBM103 UTILITY, RMXB ? FOR MYARC XB, SCRNCDEM/X 25|, SCRNCOLR/X 25|, SL/CALCU 27, SPRITEMOVE 25, SQUIRMY 20, ST$REPLACE 25, STAR(MXB) ?, SHOWCASE 25, TISAVECHAR 25, TIWRITER 21 CHANGES V2 DV80 FILES TO V1 COMPATIBLE, TRAFFICCOP 16 game, UPSIDEDOWN 21 see flip above, VALCALLKEY 23, WONKAPILL 25 TI basic game, XB/TRICK 25 note the name -list before running!

>TI*MES TEXT- three years of DV80 files from TI*MES in archived format (requires Bob Boones ARCHIVER to use-see Util 21). REQUIRES FIVE SSSD DISKS.

>FUNLWEB 4.30 - 40 COLUMN VERSION ! TWO DISKS BUT PARTLY ARCHIVED! to make it fit! Minor amendment to start up which may now be direct to UL or to DR, with space bar defeating this and going to main menu. The 80 column 4.3 has a major upgrade to DR not possible in 40 cols and comes as a single disk supplement to the 40 column version (eg 40 col=2 disks, 80 col=3 disks).

RLE28...Cairofont(lots of small pics to be clipped); Mrs Goggins, Hedgehog, Mermaid, Dogs, Pets, Old phone, Postman Pat.

TI RUNNER SCREENS A- in TI Artist format, the screens from the game, disk A contains screens 1 to 13. Shows exit ladders and different types of brickwork! May help you complete a screen you are stuck on.

TI RUNNER SCREENS B- screens 14 to 26 as above
TI RUNNER SCREENS C- screens 27 to 40 as above
TI RUNNER SCREENS D- screens 40 to 50 as above (some in colour)

MACFLIX DISKS- require you to own PIX PRO commercial program:

MACFLIX 21...Fievel01 and 02 (lovely mice); Make It Sew (Star Trek TNG); Shaw3 (you know who); TI

continued on page 17

Beginning Forth - part 5

by Earl Raguse, UGOC, CA USA

LOOPS

Before we continue with loops, I must rectify an oversight from Beginning Forth #3. On Screen #60, I used the conditional words IF ... ELSE ... THEN without comment, I hope you read the TIFM and were not thrown by these words. The manual is fairly clear on how these work, but repetition is always good, so I will do it. The word IF looks for a Flag on the stack. Any non-zero value is considered True and a zero is False. If the Flag is True, the words between IF and ELSE (or THEN if ELSE is not included) are executed, then control moves to the words following THEN. The word ENDIF may be used instead of THEN. If the Flag is False, the words between ELSE and THEN are executed, then control passes to the words following THEN. ELSE is an optional word, if not included, when IF senses a False Flag the words following THEN are executed. We will be using these words very often, so be sure to get a good handle on them.

In Beginning Forth #4, I said that the loop limit L must be larger than the index I, and that is true if you want to loop. However, if the limit L is smaller than the index I, the words between DO and LOOP will execute once, since the limit test is not made until LOOP is executed. Like BASIC, Forth has the capability to specify the increment for I. In this case Forth uses the word +LOOP instead of LOOP, and the increment(+/-) is placed just ahead of +LOOP. We can write, for example,

```
: DEMOLP CLS 0 0 AT -5 100 DO I .
   -5 +LOOP ;
```

DEMOLP will clear the CRT and print the numbers in descending order, by 5's, from 100 to 0 in a string across the top of the CRT. The CLS 0 0 AT clears the screen and places the cursor at the upper left corner of the CRT. This series of words is often defined as the word HOME, see my UFW's. Notice that that L=-5 and I=100, so my previous statement about relative sizes of L and I must be amended. The words DO I . just print the value of the index I as before. The new thing here is -5 +LOOP. The word +LOOP works just like LOOP except the instead of incrementing I by one, it increments (decrements) I by the value just preceding +LOOP, in this case -5. Control is returned to just after the DO, if the increment is positive, and the index is less than the limit; or the increment is negative, and the index is greater than the limit.

Usually we wish to execute a series of words in a definite loop a specific number of times set by the Index and the Limit, unless an event occurs to change our minds. In BASIC, we can escape a FOR NEXT loop with a GOTO statement. Forth has an escape word called LEAVE, which causes termination of the loop by setting the Limit equal to the current Index value, all words are executed normally until LOOP or +LOOP is executed. For example:

```
: JMPOUT HOME 100 1 DO I .
     I 10 - 0= IF LEAVE THEN LOOP ;
```

The word JMPOUT would normally print the numbers from 1 to 99, but the test conditions I 10 - 0= IF LEAVE THEN will cause the loop to be exited when I = 10. LEAVE works equally well with LOOP or +LOOP.

LOOP and +LOOP make what are termed definite loops. There are other Forth words for making indefinite loops when one has no way of knowing how many loops are required, but wishes to exit the loop whenever a certain testable condition becomes true. These other Forth loop words are BEGIN, UNTIL, WHILE, REPEAT and AGAIN. There is another looping word, MYSELF, which I will also discuss. The words AGAIN and MYSELF make infinite loops, and require a special means, other than shutting down the computer, to exit them. Screen #55 shows some fairly stupid, but illustrative, examples, for the use of these words.

I told you before, I would provide a simple way to avoid using up memory and getting the 'Not Unique' message every time you reloaded a screen. Forth provides a word for clearing memory of words no longer wanted or required. That word is FORGET followed by the word you wish to purge from memory. This is a mixed bag of blessings however, because Forth not only purges the dictionary of the specified word but ALL words defined thereafter. This insures that no word, which used the purged word, is left in memory.

On Screen #55 you will find the words FORGET IT : IT ;, which forgets and redefines the word IT. Before loading this screen, you must have previously defined IT, thusly : IT ;, else you will get an error message IT ?. I do this as the last thing on Screen #3 when I boot Forth, this leaves IT on top of the dictionary. Forgetting IT and redefining IT does nothing unless a new set of words have been put on the dictionary on top of IT, as happens when you load a screen. This little technique insures that you do not accumulate multiple versions of a screen in memory. Only the last version of a word executes when you use it, but old versions can take up a lot of memory. Why, you say, would I load the same screen twice or oftener? Well, if you are like me, you make errors or the words do not do what you expect. Then you have to re-edit and re-load the screen.

The word Q is defined as a variable for later use.

FCTN 4 is not active in Forth unless one makes it so, and because there is some danger of locking up the computer in an endless loop if an error is made, I have provided the word STOP?. STOP? calls the resident word ?TERMINAL which checks the input stream for a keyboard press of FCTN 4, if found, it leaves a true flag on the stack, else a false flag. You must provide the IF ... THEN to test for this and provide the appropriate instructions, if FCTN 4 is executed. In this case I have just executed ABORT, which restores Forth to the immediate mode with an empty stack.

DEMOLP is the loop word we have already mentioned. UNTLP (UNTil LooP) calls HOME, a UFW, then makes an an indefinite loop created by BEGIN ... UNTIL. BEGIN, more or less, just marks a place to return to. UNTIL checks for a True Flag on the stack. If found, it passes control to the word following UNTIL, if the flag is False, UNTIL returns execution to the word following the place marked by BEGIN. In between BEGIN and UNTIL, you may have any legal Forth words which will be executed each pass through the loop. These words should leave a flag on the stack for UNTIL.

In this case the value on the the stack, initially zero, is incremented by (1+), (DUP)licated and printed (.), then (DUP)ed again and compared (=) with 25, if equal a True Flag is left for UNTIL, which is followed by three carriage returns (CR) and a message to be printed. If not equal to 25 (False Flag) the loop is executed again.

The word AGNLP (AGaiNLooP) uses the words BEGIN ... AGAIN. BEGIN works just like before, but AGAIN mindlessly returns execution to the place marked by BEGIN forever. I am not sure I know a good use for this word. The Forth INTERPRETER is an infinite loop, but I do not know if it uses AGAIN, more probably it uses MYSELF discussed below. The only ways out of AGNLP are explained in the loop. This loop includes another UFW, WAIT, which is a waiting loop that was discussed in Beginning Forth #4.

The next loop is RPTLP (RepeaTLooP) formed by BEGIN ... WHILE ... REPEAT. WHILE looks for a Flag on the stack, left by the words between BEGIN and WHILE, as long as the Flag is True the words between WHILE and REPEAT are executed, else execution passes to the words following REPEAT. REPEAT is similar to AGAIN in that it simply causes execution to return to the place marked by

continued on page 9

FastTerm Documentation

by Paul Charlton, Virginia, USA

Instructions for use:

1) You must have: EDITOR/ASSEMBLER, MINI-MEMORY, TI-WRITER, or EXTENDED BASIC.
2) You must have a Disk-Controller card.
3) You must have 32K of expansion memory.
4) To use the printer spooler feature you currently must have a PIO interface manufactured by TI,AXIOM (PARALLAX), CORCOMP, or MYARC. The spooler has been tested with these devices and might not work with others.

To load from EXTENDED BASIC

1) Turn computer off.
2) Place EXTENDED BASIC cartridge into the cartridge slot.
3) Turn computer on and go to EXTENDED BASIC.
4) Type: CALL INIT, then press <enter>.
5) Type: CALL LOAD("DSK1.FAST-LOAD").
6) Then: CALL LINK("TERM"). You may even do this from a program you have written (PHONE-DIALER, etc.).

To load with EDITOR/ASSEMBLER or TI-WRITER menu

1) Turn computer off.
2) Place EDITOR/ASSEMBLER or TI-WRITER cartridge in the cartridge slot.
3) Turn computer on and go to the cartridge main menu.
4) Select "RUN PROGRAM FILE" (5) from E/A or "UTILITY" (3) from TI-WRITER.
5) Press <enter> (NO FURTHER TYPING IS NECESSARY.)

To load from MINI-MEMORY

1) Turn computer off.
2) Place MINI-MEMORY cartridge into cartridge slot.
3) Turn computer on and go to MINI-MEMORY menu.
4) Press 3, re-initialize.
5) Press FUNCTION "6", proceed.
6) Press 1, "LOAD AND RUN".
7) Type: DSK1.FAST-LOAD. <enter>.

FAST-TERM now attempts to find the file DSK1.CHARA1, the TI-WRITER replacement character set. You are now in the program. The program has auto-repeat on all keys. Enter the name of a parameter file you have created with the program DEFAULT or hit <enter> to choose the following defaults:

1200 =baud rate for the modem.
EVEN =parity for the modem.
PIO/1 =printer spooler (PIO/1 ON TI RS232 CARD! (not CORCOMP).
RS232/1 =serial port to be used by modem.
FULL =duplex (auto-echo of keys you type is OFF).
OFF =log file is not on when you enter FAST-TERM.
OFF =printer spooler is off when you enter FAST-TERM.
OFF =there is no linefeed sent after an <ENTER> in file send.
WHITE =text colour.
DK. BLUE =screen colour.
40 =screen width.
DSK1.SESSION =file to LOG to.

--------------------COMMUNICATIONS----------------------

There are several parameters that need to be configured to work with the system you are calling: baud rate, parity, and duplex.

The baud rate must be set to be the same as what the other system expects, it must also be set to a rate which your modem can handle. The default baud rate is 1200. This is not compatible with a large number of modems. You must set your baud rate by pressing CONTROL <1>. You will go through a series of baud rates that FAST-TERM handles:

```
     1200--> 2400--> 4800--> 9600--> 19200--> 110-->
300--> 600--> 1200
```

The baud rates faster than 1200 baud are not yet useful in tele-communications, they are most useful to someone who wishes to use the TI as a hard-wired terminal for another computer. (There are some CP/M cards for the TI that connect to the TI through the serial port, FAST-TERM is very useful for those.)

The parity must be set also. Consult the service you will be using to determine what parity they expect. The service will also have a suggested number of data-bits to use. In FAST-TERM, data with EVEN or ODD parity is automatically SEVEN bits; data with NONE for parity is automatically EIGHT bits. Use CONTROL <3> to change the parity. Here is the sequence:

```
EVEN--> ODD--> NONE--> EVEN
```

There is also DUPLEX. This controls how characters you type will appear on your screen. FULL duplex means that any character you type must be sent back to you by the other system before you actually see it. HALF duplex means that the other system expects the terminal you are using (TI, with FAST-TERM) to automatically display the characters as you type them. There are very, very few systems around which expect you to use HALF duplex, FAST-TERM's default is FULL duplex. You may change the duplex by pressing FUNCTION SHIFT <D>, all at the same time.

-----------------------HARDCOPY-------------------------

SPOOLER : FAST-TERM provides a printer spooler feature. This means that if you have a printer you may instruct FAST-TERM to send everything you will see on your screen to the printer as it is displayed on your screen. The spooler allows data to be displayed on your screen faster than the printer can print it, the printer is allowed to be as much as FOUR-THOUSAND characters behind what you see on your screen. If the printer does manage to get Four-Thousand behind, the incoming data will no longer be sent to the printer and FAST-TERM will give you a warning message and a bell to tell you that the printer is very far behind. The spooler enables you to have a complete print-out of your whole session with the other system. Spooler is enabled by pressing CONTROL <2>. You may disable the spooler by pressing CONTROL <2> again. When you disable the spooler no more incoming data will be sent to the printer; data that the printer had gotten behind on will still continue to print.

SCREEN-DUMP : FAST-TERM also provides a screen-dump feature. If you have a printer you may print a copy of the data which is on the screen. Press FUNCTION <0> to freeze the screen (explained later) then press FUNCTION SHIFT <P>, all at the same time. All of the characters on the screen will be placed into the spooler, to be printed when the printer catches up. You must leave this freeze mode before you will see the rest of what the other system was sending you (press FUNCTION <0> again).

------------------------HARDWARE------------------------

You must tell FAST-TERM how you have your hardware attached.

MODEM: Tell FAST-TERM which serial port your modem is attached to by pressing CONTROL <4>. Pressing this key takes you through a list of ports available for the modem. Stop when you get to the port to which your modem is attached.

PRINTER: Tell FAST-TERM which port your printer is attached to by pressing CONTROL <6>, this takes you though a list of ports available for your printer. Stop when you get to the right one. The default is the PIO port on the TI RS232 interface. If you have a CORCOMP serial interface and you have a printer for which you send data to PIO, you must select CORCOMP PIO from the list. NOT PIO! If you have an AXIOM PARALLAX interface, you must choose AXIOM PIO for the printer to work. People with serial interfaces built by yet another manufacturer should try all of the available choices to find one which works. It is possible that your interface IS NOT compatible with FAST-TERM. For such people, I suggest that you LOG to your printer (LOGging is explained later). Set your printer baud rate (serial printers only) by pressing CONTROL <7>. This works in the same way as setting the modem baud rate. Set your printer parity (serial printers only) by pressing CONTROL <5>. This works in the same was as setting the modem parity.

----------------SENDING AND RECEIVING FILES-------------

RECEIVING ASCII files (text that you can read). You can receive an ASCII file by LOGging to a file. Press FUNCTION <B>, this closes any previous file you were logging to and asks you for a new filename to log to. To log data you must give a filename. Then, at the point at which you want data to start going into the file, you must press FUNCTION <.> (period). All data will now go into the log file. Press FUNCTION <.> again when you get to the end of the data you want to save. All of the data is now in memory, to write it to disk you must press FUNCTION <B> again. This writes the file to disk, closes it, then asks you for a new filename, enter a blank line if you do not want another log. If at some point in the transfer the memory gets full FAST-TERM will empty the buffer to disk then continue where it left off, still logging to the file. With a proper setup, through the DEFAULT program written in TI BASIC, no data will be lost as FAST-TERM writes the log to the file. For people whose serial interfaces do not allow them to use a printer with FAST-TERM, you should log to "PIO" to get a hardcopy of your session. The log file is in a DISPLAY, VARIABLE 80 format. You may clear the LOG's memory buffer at any time by pressing FUNCTION <Y>.

SENDING ASCII files. To send an ASCII file you must first press FUNCTION N> and give the name of the file which you wish to send. Press FUNCTION <,> (comma) to start sending the file. You will be asked of you want to send the file line-by-line. If you decide to send the file line-by-line you must press the spacebar everytime you want to send another line from the file. Press FUNCTION <4> to stop sending from the file. If you choose not to send the file line-by-line the file will be sent as quickly as the other system can handle it. The other system may send FAST-TERM a CONTROL <S> when FAST-TERM gets ahead of what the other system can handle, and send a CONTROL <Q> to FAST-TERM when it has caught up. (XOFF/XON handshaking) When you have sent enough of the file you may hit FUNCTION <4> to stop. Files you send may be in either DISPLAY, FIXED 80 or DISPLAY, VARIABLE 80 format on your disk.

Another option you may use when sending the file is to add a line-feed after every carriage return you send to the other system. This is useful when you sending a file to someone else's terminal for them to read. This feature is enabled by pressing FUNCTION <J>. It is disabled by pressing FUNCTION <J> again.

---------------------- TE2 protocol --------------------

Sending files using the TE2 protocol. Press FUNCTION SHIFT <T>, all at the same time. This enables transmission of files with the TE2 protocol. Pressing those keys again enables ASCII send. Press FUNCTION <N> and give the name of the file which you wish to send. Press FUNCTION <,> (comma) to send the file. You will be kept informed of the progress of the file transfer by an on-screen report of current block number, record number, and number of retries. A bell will ring when the transfer is complete. You may abort the transfer at any time by pressing FUNCTION <4>.

Receiving files using the TE2 protocol. First press FUNCTION <N> and give the name of the file which you want data to go into. THEN tell the other system to start sending you the file. You are given a report of the transfer's progress. Press FUNCTION <4> at any time to abort the transfer.

--------------------- XMODEM protocol ------------------

Sending files using the XMODEM protocol. Press FUNCTION <N> and give the name of the file you wish to send. Tell the other system to prepare to receive a file. Once the other system is ready, press FUNCTION SHIFT <X>, all at the same time. Press <S> or <ENTER> to <S>end file. The rest of the transfer is automatic. The transfer will be aborted if at any time the number of retries for one record exceeds ten. You may abort the transfer at any time by pressing FUNCTION <4>, note that the other system will become confused by your abort, and will take about 90 seconds or so to give you a prompt back. (Most other systems.)

Receiving files using the XMODEM protocol. First press FUNCTION <N> and give the name of the file you want data to go into. THEN tell the other system to start sending you the file. Now press FUNCTION SHIFT <X>, all at the same time. Press <R> for <R>eceive. You now need to choose which type of error checking you wish to use, CRC or CHECKSUM. CRC is more accurate than CHECKSUM, but not all systems support it (ask the sysop of that system about availability of CRC error checking). Note that if you choose CRC and the other system does not support it that you will be delayed for about 60 seconds. This is the time it takes XMODEM to automatically switch to CHECKSUM mode. Press FUNCTION <4> to abort. Note that the other systems aborts immediately as well.

XMODEM sends TI disk directory information along with the file. It expects to see this information on any file it receives so that it may place the data into a file easily usable on the TI. Reception of a file which does not include directory information results in data being placed into a DIS/FIX 128 file format. Also, any file with the characteristics of DIS/FIX 128 is sent WITHOUT the directory information. This results in all-around compatibility with files from other systems.

---------------------OTHER FEATURES--------------------

SCREEN CONTROL: you may set the width of the screen to either 40 or 80 characters. Press CONTROL <0> to toggle between 40 and 80 columns. Note that this operation does not clear the screen like other terminal programs. If you choose 80 column mode you will have to window from side-to-side to see all of the screen. Pressing FUNCTION <5> jumps you to the right, FUNCTION <L> jumps you to the left. You can set how far you jump by using the DEFAULT program which is written in TI BASIC.

COLORS : You may change the screen colour by pressing FUNCTION <8>, you may change the text colour by pressing FUNCTION <7>.

SCREEN FREEZE: you may pause display of incoming data by pressing FUNCTION <0>. This allows you to read what is on the screen without worrying about it scrolling of the top before you can read it. You unpause by pressing FUNCTION <0> again. All of the data that came in while you were paused is now printed on the screen.

WINDOW BACK: this feature allows you to look at data which has already scrolled off of your screen. First enter FREEZE mode by pressing FUNCTION <0>. Press the spacebar to quickly scan through the data, press §S> to go slowly through the data. You will see a SOLID BAR on the screen when you have caught up to where you were when you began. During window back print spooling and LOGging are suspended. You may re-enable them during window back mode in order to save "lost" data, data which had already scrolled off of the screen. You may also change things like screen width and colour while you are in window back mode. When you leave window back mode by pressing FUNCTION <0> everything that you did like change screen parameters and LOGging is returned to the state it was in when you entered window back mode.

TIMER: FAST-TERM has a timer you can use to display your connect time. It is constantly display in the upper-left corner of the screen when you enable it by pressing FUNCTION <K>. You may turn the timer off by pressing FUNCTION <K> again. Enabling the timer always resets it to zero. The timer is disabled if you have FAST-TERM running at greater than 1200 baud. NOTE THAT THIS IS NOT A TIME-OF-DAY CLOCK, IT IS ONLY AN ELAPSED TIME COUNTER.

DISK DIRECTORY: the directory of a floppy disk drive may be shown by pressing FUNCTION <9>, only the names of the files are shown.

FAST-TERM knows all of the ADM3A control codes and escape sequences.

```
ADM3A: control code        ]   function
       >07                 ]   bell
       >08                 ]   backspace(non-destructive)
       >0A                 ]   linefeed
       >0B                 ]   move up one line
       >0C                 ]   move forward one space
       >0D                 ]   carriage return
       >1A                 ]   clear screen
       >1E                 ]   home cursor
       >1B '=' R C         ]   moves cursor to row(R-32)
                             and column ( C-32 ) SPECIAL:
                               (not ADM3A, FAST-TERM only)
       >0E                 ]   turns off reverse video
       >0F                 ]   turns on reverse video
```

To set up a parameter file for use with FAST-TERM, go to BASIC and run the program DEFAULT. The program is self-explanatory and menu-driven. Use it to set up everything that you can set up with function keys within FAST-TERM, as well as some features that can only be changed by the parameter file.

```
********** CONTROL KEYS FOR TERMINAL PROGRAM **********

FCTN- 1 Transmits >7F (DELETE)
    - 2 Transmits >14 (control-T)
    - 3 Transmits >0E (control-N)
    - 4 Transmits BREAK TONE
        --> also "break" hung-up I/O funtions
    - 5 WINDOW RIGHT (LIKE EDITOR AND TI-WRITER)
*
    - J TOGGLE TRANSMISSION OF LINEFEED AFTER "CR" IN
        ASCII FILE SEND MODE
    - K SET THE TIMER TO ZERO.
    - L WINDOW LEFT  (LIKE WINDOW RIGHT BUT OTHER
                                          DIRECTION)
*
    - 6 Transmits >05 (control-E)
    - 7 CHANGE FOREGROUND TEXT COLOR
    - 8    "   BACKGROUND   "    "
    - 9 DISK DIRECTORY
    - 0 TOGGLE WINDOW BACK MODE (ON/OFF)
        --> SPACEBAR FOR FAST DISPLAY, "S" FOR SLOW
                                            DISPLAY
*
CTRL- 0 TOGGLE SCREEN WIDTH BETWEEN 80 AND 40 CHARS --!
                               DOES NOT CLEAR SCREEN!
    - 1 CHANGE COMMUNICATION SPEED (BAUD RATE)
    - 2 TOGGLE PRINT SPOOLER ON/OFF
    - 3 CHANGE PARITY (modem)
    - 4 CHANGE MODEM PORT
    - 5 CHANGE PRINTER PARITY.   (ONLY USEFUL FOR SERIAL
                                            PRINTER)
    - 6 CHANGE PRINTER PORT.
    - 7 CHANGE PRINTER BAUD RATE
*
FCTN- . TOGGLE LOG ON/OFF
    - N NAME THE DISK FILE TO USE FOR FILE SEND AND TE2
                                          TRANSFERS.
    - Y CLEAR THE LOG
    - B WRITES LOG TO DISK, GETS NAME FOR NEW LOG
    - , SEND DISK FILE TO OTHER SYSTEM.
```

continued on page 14

Newsletter update

by Bob Relyea

TIBUG (Brisbane), February, 1991: Editorial; 32k memory in console; Testing 32k memory; In the P.O. Box (Newsletter Updates); TI-Base Tutorial #1; Adding Graphics To Your Program by Garry J. Christensen; Emulation Management Utilities (distributed by Asgard Software) by Garry Christensen; Tips From the Tigercub # 26.

ATICC (Adelaide), February, 1991: Coordinator's Report; Editorial by Peter Giles; Configure Printer for Funnelweb; Take Five by Bob Clarke; Statistics for Nonstatisticians by A Burke Luitich; Programming the TI by Regena; Basic Training by Boyd Shugert; User Notes including Checksum Update; Getting the Most From Your Cassette System by Mickey Schmitt.

SPIRIT OF 99, February, 1991: Conni Calendar; Putting It All Together #8 by Jim Peterson; What's Hott by Irwin Hott; You Don't Have To Have It All by Jim Peterson; Notes On Laser Printing with the TI-99/4A; Manipulating DV/80 Files, Part 2 by Art Byers; Tips From The Tigercub #62; New Age/99 #9 by Jack Sughrue, TI-Base V3.0 Tutorials 19.1.1,2,&3 by Martin Smoley.

LA99ers TOPICS, February, 1991: Ramblin' Thoughts From The President; XBasic Miscellany #1 by Earl Raguse; Help--Call 911 by Earl Raguse; Arrays in Forth; TI-TAX; The Cracker Barrel by Chick De Marti; New Age/99 #13 by Jack Sughrue, The VCR Connection; The Magic of Jim Peterson; The Cuckoo's Egg by Clifford Stroll; TI 99er User Club News.

TI-FOCUS (Ontario), February, 1991: News and Views from CH99 by Tom Arnold; A Module For The Times by Randy Packham; A Dream Come True by Randy Packham; Precise Math by Jim Peterson; Multiplan Tutorial? by Tom Arnold; Recorded, Transcribed, Printed by W. Murrell; New Age/99 #11 by Jack Sughrue; Club Page by Tor Hansen.

THE PUG PERIPHERAL, February, 1991: As The Floppy Turns by Don McCalla; TI Express, Configuring Funnelweb-The Quick and Dirty Way; New Age/99 by Jack Sughrue; The Kiddie Corner by Sue Harper; PC PS in the PEB by John Willforth.

THE OTTAWA NEWSLETTER, February, 1991: Editor's Notes by Philip Harris; Ottawa TI-Fest-1991 by Bill Gard; Fast Extended Basic by Lucie Dorais; Under The Palm Tree by Lucie Dorais; Hotline Numbers.

UGOC ROM, February, 1991: From the President by Ben Hathway; 24k of Data Storage; President Greets Visitors to Fest West '91 by Ben Hathway; The Blinkin' Cursor by N. Armstrong; Membership Corner by Bill Nelson; 4A's Owner's Test by Earl Raguse; Unsung Heroes by Stan Corbin.

THE BOSTON COMPUTER SOCIETY, January, 1991: Listen by Justin Dowling; Chicago TI Faire Report by Gary Cox (long article); TI-Image-Maker; Saving A Lost File by D.L. Mahler.

TIdbits, February, 1991: President's Bit by Gary Cox; Program Bit by Gary Cox; In The News by Gary Cox; Computerized Translator by from Industrial Week; Learning by Marshal Ellis; Artist Cataloger Review by Robert Bruce; Calling Subroutines by John Hartweg; Writing a Review by Marshal Ellis; Editor's Notes by Marshal Ellis. o

continued from page 25

some of which also have multiple possibilities. For instance, you can, in one single command, ACCEPT at a screen location either a Y or N, the user does NOT need to press enter, and you can simultaneously put the "Y" etc into two string variables.... there is also GETSTRING which is a multiple GCHAR. o

GPL Disassembly

by Craig Miller, California, USA

Millers Graphics will be releasing a GPL Assembler in the first part of 86'. We thought you might like to see what a powerful and COMPACT language GPL code is. With the GRAM KRACKER and our GPL Assembler you will be able to write programs that can reside in the Module space and will be displayed on your Main Menu as a selection. GPL can also link to Assembly and BASIC programs! So you will have FULL use of the THREE built-in languages in our 4As (Basic, GPL and Assembly). Eat your hearts out all you Atari, Commodore, IBM and other computer owners!

```
*----------------------------------------------------*
*  Disassembly of part of the Editor/Assembler Module *
*        Starting at Grom >6069 thru >6132            *
*----------------------------------------------------*

>6069  MOVE  7 FROM G@REGDAT TO VR01         Load the Vdp
                                                registers
       CALL  CHKMEM                         Go check for
                                          memory expansion
                                          and load the (C)
                                            character data

       MOVE  16 FROM G@CURSOR TO V@>08F0     Load the box
                                          and solid cursor
                                                      data

* Put up the first Menu Screen

       ST    >7E,@SUBSTK               Initialize the Sub
                                     Return stack pointer
       DCLR  @ERRCODE                  Zero out A/L Error
                                           Code indicator
       DCLR  @GROMFLG                   Zero the Grom Flag
       ALL   SPACE                        Clear the screen
                                   with space characters

       FMT                         Start formatted screen
                                                   output
       ROW   2                                  At row 2
       COL   1                      At column 1 (note 0,
                                      0 is home position)
       HTEXT '* EDITOR/ASSEMBLER * '  Put up horizontal
                                                     text
       ROW+  2                     At current row plus 2
       COL   1                               At column 1
       HTEXT 'PRESS:'                   .
       ROW+  2                          .
       COL   2                          .
       HTEXT '1 TO EDIT'                . etc.
       ROW+  2                        Note: VTEXT, HCHAR,
                                          VCHAR are also
       COL   2                      allowed in a FMT, so is
       HTEXT '2     ASSEMBLE'     FOR xx - where xx equals
       ROW+  2                      the repeat loop counter
       COL   2
       HTEXT '3     LOAD AND RUN'
       ROW+  2
       COL   2
       HTEXT '4     RUN'
       ROW+  2
       COL   2
       HTEXT '5     RUN PROGRAM FILE'
       ROW+  6
       COL   2
       HTEXT >0A                  >0A is the (C) character
       HTEXT '1981 TEXAS INSTRUMENTS'
       FEND                         End the formatted screen
                                                     output
GETKY  SCAN                           Scan the keyboard for
                                                a key press
       BR    GETKY                  BR (Branch on Reset) no
                                           NEW key pressed

CEQ FCTN9,@KEY Was FCTN 9 (Back) Pressed
       BR    GETKY1                NO! check the other keys
       EXIT                          YES! Execute the Power
                                                 Up routine

GETKY1 SUB >31,@KEY Subtract >31 from the
                                         keycode (0 - ?)
CHE >05,@KEY If its now Higher than 4
                                             - wrong key
       BS    GETKY                So, go wait for another
                                               key press

CASE @KEY Otherwise if @KEY equals

       BR    EDIT        0 - goto Edit Menu
       BR    ASSEM       1 - goto Load Assembler Prompt
       BR    LODRUN      2 - goto Load and Run prompt
       BR    RUN         3 - goto Run Program prompt
       BR    RUNPRG      4 - goto Run Program File prompt
```

Notes: ---*

The above code only requires 202 bytes of memory and that includes 119 bytes of text! So that means the actual instruction code only uses 83 bytes of memory! There is not another language available for our 99/4As that is this COMPACT as GPL. And, when compared to Assembly, it is much easier to program in. This is THE Language that TI should have released to us in the first place!

1. Most instructions can work with bytes or words. The D in front of an instruction indicates a word operation. The first operand to is SOURCE and the second is the DESTINATION. ie: ST >03,@TEMP1 stores one byte with the value of 3 into location TEMP1.
2. The COND bit in the GPL Status register (>837C) is turned ON if the test is TRUE and OFF when FALSE. It is also turned on when a NEW key is pressed on a keyboard scan or when the result of certain instructions is zero.
3. BR = Branch On Reset... or Branch if the COND bit in the GPL Status register is OFF.
4. BS = Branch On Set..... or Branch if the COND bit in the GPL Status register is ON.
5. CASE is like ON X GOTO ..... except it starts at zero instead of 1 (Note: the COND bit is always turned OFF (reset) for a CASE or DCASE).
6. A CALL works like a GOSUB or Assembly's BL (Branch and Link).
7. 'ALL' fills the screen with the one byte character following the instruction. (That is right only 2 bytes to clear the screen!!!!).
8. MOVE is a very powerful GPL instruction. With it you can MOVE x number of bytes FROM any type of memory TO any type of memory You can also move bytes to the VDP Registers! The MOVE instruction only requires 6 to 7 bytes for its object code!
9. SCAN only requires 1 byte of object code!!! (SCAN = >03).

Speed Test: --*

We ran the old 1 to 10,000 timing test in GPL to see how it compares to the other languages and here is how it came out.

1. In an incrementing loop with a DCEQ (double Compare Equal) 6.8 seconds.
2. In a decrementing loop (no compare just BR (not zero)) 4.3 seconds.

As we have seen from previous tests this places third on the list.

1. Assembly - well under .5 second
2. Forth - approx 1.3 seconds
3. GPL - 4.3 to 6.8 seconds
4. Pascal - I think this is where it falls
4. XB - 33.9 seconds
5. Basic - weeks (just kidding)

Since its not as fast as Assembly or Forth you are probably wondering why we are so excited about GPL?! True, a CRAY 3 its not. However, it requires LESS THAN one half the space of Assembly code! With the Gram Kracker you have up to 58K of GPL program space (with 6K reserved for the Operating System), which would require

continued on page 6

Regional Group Reports

Meeting Summary For MAY

Banana Coast	12/05/91	Sawtell
Carlingford	15/05/91	Carlingford
Central Coast	11/05/91	Saratoga
Glebe	09/05/91	Glebe
Illawarra	13/05/91	Keiraville
Liverpool	10/05/91	
Northern Suburbs	23/05/91	
Sutherland	17/05/91	Jannali

BANANA COAST Regional Group
(Coffs Harbour area)

Regular meetings are held in the Sawtell Tennis Club on the second Sunday of the month at 2 pm sharp. For information on meetings of the Banana Coast group, contact Kevin Cox at 7 Dewing Close, Bayldon, telephone (066)53 2649, or John Ryan of Mullaway via the BBS, user name SARA, or telephone (066)54 1451.

CENTRAL COAST Regional Group

Regular meetings are normally held on the second Saturday of each month, 6.30pm at the home of John Goulton, 34 Mimosa Ave., Saratoga, (043)69 3990. Contact Russell Welham (043)92 4000.

GLEBE Regional Group

Regular meetings are normally on the Thursday evening following the first Saturday of the month, at 8pm at 43 Boyce St, Glebe. Contact Mike Slattery, (02)692 0559.

ILLAWARRA Regional Group

Regular meetings are normally on the second Monday of each month, except January, at 7.30pm, Keiraville Public School, Gipps Rd, Keiraville, opposite the Keiraville shopping centre. A variety of activities accompany our meetings. Contact Lou Amadio on (042)28 4906 for more information.

LIVERPOOL Regional Group

Regular meeting date is the Friday following the TIshug Sydney meeting at 7.30 pm. Contact Larry Saunders (02) 6447377 (home) or (02) 7598441 (work) for more information.

NORTHERN SUBURBS Regional Group

Regular meetings are held on the fourth Thursday of the month. If you want any information please ring Dennis Norman on (02)452 3920, or Dick Warburton on (02)918 8132.

Come and join in our fun. Dick Warburton.

SUTHERLAND REGIONAL GROUP

Regular monthly meetings are held at the home of Peter Young, 51 Jannali Ave., Jannali on the third Friday of each month at 7.30pm. unless otherwise advised. (02) 528 8775.

The format of each meeting is quite informal, with topics ranging from software reviews to hardware modifications with a fair sprinkling of purely social chatter in between.

Peter Young
Regional Co-ordinator

TIsHUG in Sydney

Monthly meetings start promptly at 2pm (except for full day tutorials) on the first Saturday of the month that is not part of a long weekend. They are held at the RYDE INFANTS SCHOOL, Tucker Street (Post Office end), Ryde. Regular items include news from the directors, the publications library, the shop, and demonstrations of monthly software.

The cut-off dates for submitting articles to the Editor for the TND are:

June	12 May
July	9 June
August	7 July

TIsHUG Meetings for Sydney, 1991

May
The first all day tutorial session. Topics to be offered include: hardware clinic to explain how the bits, bytes and words go together and to answer questions; word processing for beginners; basic Extended BASIC; RAMdisks with or without EPROMs; how to modify your RAMdisk to take EPROMs; games; adventuring hints; assembly class; copying software; and browse the library. There will be a barbeque lunch provided for a small cost.

June
Demonstrations of the latest software or hardware. Watch this space for details.

July
The second Buy, Swap and Sell day. Bring things for sale and also your money for the terrific bargains. You have to be early to beat Rolf to a bargain!

August
Demonstrations of the latest software or hardware. Watch this space for details.

September
Demonstrations of the latest software or hardware. Watch this space for details.

October
The third Buy, Swap and Sell day. Your last chance this year to get some money for Christmas presents or to get an early present for yourself.

November
The second all day tutorial session.

December
The Annual General Meeting followed by some festive eats and drinks. Make sure you attend and give your support to all the workers in the club. o

continued from page 3

STEP 2

SETUP A MENU PROGRAM AS FOLLOWS:

1. Name this file ... "00_WF_01"
   00 = Menu
   WF = Writer File
   01 = File Disk #1

2. Keep the menu file simple or as complex as you would like it!

3. Example of file:

   A) Line 1 = Menu File Name: "00_WF_01"
   B) Line 2 = blank space
   C) Lines 3 & 4 = Header:

   FILE NO.   DESCRIPTION

   D) Lines 5 - ? = Your Dictionary:

00_WF_01	Menu
01_WF_01	Call Loads
01_WF_01	Disk Info

   The above descriptions can be as long as you wish!!!!

4. Save to disk.

STEP 3

HINTS

1. After loading a menu file:

   A) You can search for a file by using the 'Find String' function.

continued on page 12

Renew Now