(063) 8805
Winnipeg

WINNIPEG

Newsletter

MAY

The Winnipeg 99/4A User Group is a non-profit organization formed by computer hobbyists for users of the Texas Instruments 99/4A Home Computer and compatables. The content of this publication does not necessarily represent the view of the Winnipeg 99/4A user Group.

Next General Meeting - Date : May 29th 1988
Time : 1:00 P.M.
Place : 1334 Aikins St.

Current Executive

President:	Sheldon Itscovich	633-0835
Treasurer:	Bill Quinn	837-7758
Newsletter Editor	Darrell Wyatt	339-9742
Co - Editor	Marie	339-9742
Technical Advisor	Rick Lumdsen	253-0794
Publisher:	Peter Gould	889-5505
Publisher:	Sheldon Itscovich	633-0835
Publisher:	Dave Wood	895-7067
Publisher:	Bill Quinn	837-7758

Mailing Address:

NEWSLETTER EDITOR
1334 Aikins St.
WINNIPEG, MANITOBA
CANADA, R2V 2C7

LETTER FROM THE EDITOR
by Darrell Wyatt

It is always good to see the amount of activity that the TI can generate. As each month goes by, more and more seems to be happening all the time and on an orphan computer this is indeed good to see.

This month saw the setting up of two new members with PEB boxes which are apparently beginning to become scarce. Also Rick's supplies started to come in causing a flurry of buying and selling of hardware and software. I myself have been Rick's biggest customer what with buying the TI Artist Borders disk, latest font disk, Certificate'99 vers.2, Towers,Grampacker,Certificate'99 Companion disk.EZ Keys, Remind Me, The Printer's Apprentice and font disk, and many more. Horizon ramdisks were the biggest hardware item and I believe 3 or 4 cards were purchased this month. Rick also has the Geneve in stock as well now so.........dig deep!

It was good to see that one of our long time members and excellent supporters, Hank Derkson has joined the Edmonton group and they are certainly correct in saying that their gain is our loss. Glad to see you keeping it up Hank.

Unfortunately the amount of newsletters coming in is decreasing and I hope that this is not an omen. I can readily sympathize with the problems of getting the mailing done though as we certainly have our hands full with the cost here as well. If anyone has not been getting our newsletter regularly, please let us know and we will certainly try to remedy it. Jim McLaren out of Sudbury is trying to standardize a mailing label format with PRBase and hopfully I will get the system up and running soon. By the way Jim, Rick says that the problem you have with the extra character could be a printer problem...he had the same thing happen but with different programs on his old printer.

Sorry to digress but getting back to the mailing problem, one thing that I will be trying starting this month is a split newsletter. Certain articles that I have received through the exchange are well worth reprinting for the benefit of the group but because they are already being circulated it is pointless to keep sending groups their own articles back so I will have on occasion a separate addendum for our local group only. This will keep the mailing weight down to a minimum and save needless expense.

Once again, Rick and I are willing to give tutorials or advice on any programs and/or languages etc., so if you wish to go over anything let us know.

Well I guess that time has come...it is 4 AM. and I want to play with my toys for a bit so I better sign off. Happy TI'ing and I'll talk to you next month.

HELPFUL HINTS
by Rick Lumsdem

Another month and as summer fast appproaches thoughts of fishing and golfing are creeping into the minds of many of our members. Just keep in mind that the old TI works just as well in the summer as the winter and even works out on the patio so don't pack it away too tightly.

To start things off this month, I finally got our donation off to Tony and Will McGovern for their Funnelweb program. Total donations amounted to $50 and the reciept for the money order sent will be on Darrell's bulletin board at the next meeting if you would like to see it. Perhaps in the near future we will try the same approach for some of the other Shareware programs many of us are using.

A few weeks back and then at the last meeting I have had requests on explaining how the Editor Assembler module is to be used and how to load the various programs. I tried a short tutorial on the cartridge, but it is a lot to absorb in such a short meeting so I will attempt ot put it down on paper here so you will all have a reference in the future.

In short, the E/A package is meant to be used as a development system for the 4A to write and assemble programs in Assembler format. First I will regress a bit here and try to clear up exactly the difference between the interpretted languages (such as TI BASIC and XBASIC) and Assembled languages. There is also the Compiled languages but that I will leave for another time.

Think of your computer as a vast bank of sitches all hooked together. To get the current to flow where you want it, you must turn the appropriate switches on and off. These switches can be thought of having just two positions, either on or off. Now think of the switch as being on representing a "1" and the off as a "0". This is the "native language" of a computer which is known as machine code and it is just the BASE 2 or binary number system. A string of 00110100 could actually be an instruction to the computer because it turns on and off various "switches". Trying to write a program using this binary system would be terribly slow and error prone although it has been done. Remember the old keypunch cards?? Programmers soon realized that to be productive in any form, there must be an easier way to enter a program into the computer and so the first ASSEMBLERS were developed. The Editor and Assembler programs allowed the programmers to enter their code in a language that was easier understood by people. The EDITOR is just like the name implies, a text editor much like TI-Writer, that allows you to build up a SOURCE file. A source file consists of a long list of commands that are in a form known as Mnemonics. An example is BLWP. This mnemonic is a shortened form of Branch and Link with Workspace

Pointer. An assembler programmer must first learn to use all these commands and an understanding of the computer's operation is a virtual necissity as well. After he enters his "SOURCE" code into the editor, he then saves this file to disk. This file is just a large collection of commands in ASCII format and in this form is NOT runable. It first must be assembled with the ASSEMBLER. The ASSEMBLER reads in the SOURCE file and converts or ASSEMBLES the code. Basically it is a program that takes the ASCII source file (which in this form is meaningless to the computer) and chnages it to binary code (1's and 0's) that is understandable by the CPU. The ASSEMBLED file is saved back to disk and is known as the "OBJECT" code. Because this OBJECT code is in the computer's native bianry language, it will execute very fast. This may all sound complicated but asfter awhile it will begin to make a bit of sense.

Now, just a short explanation of the interpretted languages and their advantages and disadvantages. BASIC programs are entered directly into the computer with no need for a text editor. When you finish typing it in, you just type run and watch for errors. The problem is, is that the command "PRINT" is not in a form the computer can understand and must be INTERPRETTED. The built in BASIC INTERPERTTER handles all this translation, but only ONE LINE AT A TIME. Here in lies the difference. Each time you run a BASIC program, the interprtter must translate each line as it is executed into something the CPU can understand. With the E/A version, you type in the code, assemble it, and it is already in the proper format requiring no translation. Although BASIC is much easier to understand and write, it executes very slowly when compared to it's Assembler equivalent. On the other hand, the E/A version is still tedious to write, and if an error is made it will not generally show until it is assembled and then one must examine the source code, find the error and correct it, then re-assemble. Assembler programs also require much more detail in what you want the CPU to do. In simpler terms, let's compare it to telling someone to open a door. The BASIC version would be "please open that door" and hopefully the person would respond. The ASSEMBLER version would be "put left foot forward, right foot forward, move right arm to door knob, turn knob to left, push door outward....etc." Quite a bit of work huh?? Well suffice it to say that what you lose in this kind of tedium you will more than gain back in speed and control of your computer.

OK, now onto the TI Editor Assembler module and associated programs.

When you first plug in your E/A cartridge and selct the E/A option form the menu, you will then be greeted by another menu with 5 choices. I'll go through each of these individually. Remember that the disk that came with your module must be in drive #1 to load some of the options. The first option is the EDITOR and is selected by choosing #1. This will take you to another screen with 5 more options. You may choose either option #1-LOAD or option #2-EDIT from this menu but the E/A disk marked PART A must be in drive #1 to load the editor. The difference between these two options is that #1-LOAD will load the editor and then you will be into the editor screen ready to begin writing your code. Optio #2-EDIT will also load the editor but then will prompt you for a filename. The filename you can enter here can be one of your own source codes you had previously wrote or someone else's code that you may wish to look at. Once in the Editor you will immediately notice the similarity to the TI-Writer Editor. That is because it is basically a Text Editor with the proper tabs set for entering source codes, word-wrap turned off and a way of saving files that does not save the control codes that TI-Writer does.

Once you have entered or modified your code, you must save it back to disk or you will loose it when you exit. Pressing FCTN 9 will return you to the second screen with #3-SAVE as the third option. Pressing 3 will prompt you for a new filename to save you code to disk. The Editor saves source code files in the same format as TI-Writer which is Display/Variable 80 or as it shows up on a disk directory DIS/VAR 80. The other two options on the second menu are fairly obvious. The 4th option, PRINT, first prompts you for a filename and then will print the DIS/VAR 80 file specified out to the device you entered. Most likely a printer. The last option is PURGE and will ask you if you are sue and then will clear out your text buffer erasing any code you may have in there at the time. Make sure you save anything to disk that you have been working on before uisng this option.

Now a couple taps on the FCTN 9 button and you will be back to the first E/A screen. Option #2 is ASSEMBLER. Again you must have the E/A disk PART A in drive one and when you tap option #2 for ASSEMBLER it will ask if you want to load assembler. Press "Y" and it will load the program form disk. The assembler program will then prompt you for the SOURCE filename. You must enter the filename you were working on in the EDITOR here or any other source file you may wish to assemble. Then it will ask for an OBJECT filename. This is the filename your newly assembled code will be stored under. If you are using onl:y 1 drive make sure you have plenty of room for both the source and object code and do not use the same name for you object code as your source or it will overwrite it. This is not a problem with two drives in general. DO NOT USE your E/A disk to assemble programs. Once you have loaded the assembler you may remove the E/A disk form the drive. Now you will be prompted for a list device. You may enter any valid device name here and while the assembler is working it will list out the Source and Object code to the specified device for use in debugging your code. The device can be a printer, drive, or even the RS232. This is usually only used for debugging purposes so you may not want to enter any device name. In this case just hit enter for the next step. Now the program will ask you for OPTIOS. There are a number of options and if you decide to start using ASSEMBLY it would be best to look at your E/A manual. In short there are two that are used most often. The first is the "R" option and is used almost every time you use the assembler. It names your registers for you in the assembled program. (Don't ask please!!!!) The second is used occasionally and is known as "C" for compressed. If you enter this option, your assembled code will take up alot less space on disk. You can load a DIS/FIX 80 file into your editor to look at it and a compressed file is easily recognized by the large number of capital "B's" in the code. The disadvantage to compressing a file means it cannot be loaded into the Extended Basic enviroment. Normally you can load and link Assembler code to XB with the CALL INIT :: CALL LOAD("DSK1.filename") :: CALL LINK("startname") command, but if it has been assembled using the "C" option this will not work form XB. It still can be loaded with the E/A module and that is the next item to be discussed. Once the last option is entered Assembly begins and will report any errors it finds on the screen or list device.

Now back to the main screen again... Option #3-Load and Run. This is where you can run your assembled programs from. When you press 3 the E/A will prompt you for a filename. Enter the disk number and filename in the usual format (e.g.DSK1.GAME) and the file will be loaded. This loader will load DIS/FIX 80 files or in other words the program you just finished assembling. Now comes the fun part. When the author of the program wrote *his* source code he had the option of making it auto-start or not. If he did make it auto-start, after you enter the filename, once the program has started it will begin running. If he did not, and there are reasons for this, then once the file is laoded you are returned to the filename prompt. Just hit enter and the next prompt will be PROGRAM NAME? This is the start name of the program and it is entered with no disk # prefix. The problem arises if you do not know the start name you cannot get the program running. There are ways of finding the start name and hopefully I'll get into that when I go into the FUNNELWEB program at a later date. One other *pecularity* of this option #3 is that you can load in multiple files. Let's say you had a huge source code that totally filled your text buffer but it was not the whole code. You could save this out to disk, purge your buffer and start again where you left off. Now when you assemble the program you must assemble the two parts seperate and you end up with two object codes that are virtually the same program in two parts.(there is a method of tying the two codes together called a copy directive when assembling, but I don't know why it is not used in cases like this) So if you have this problem with two object codes you must load it in the following manner. Choose option #3-Load and Run and then enter the DSK1.part1 of the program. When the cursor returns enter the next part in the same format using the second filename. e.g. DSK1.part2. Once that has loaded, strike enter with no filename or drive and then enter the start name. Confusing??? You bet but it is not often used so don't get too uptight about it.

OK.... Now to option #4-RUN... This is rarely used unless you accidently hit FCTN 9 after you have loaded an Option #3 programm and before you enmtered the startname. This will allow you to enter a startname without reloading the file from disk. In a Mini-Memory situation, because the program is battery backed in the cartridge, you may only need a startname to get the program running. Simple huh?? Not used much though.

Option #5-Run Program FIle. More explanations.... So far we have learned that an assembled object code is saved out to disk in a DIS/FIX 80 format. The problem with this format is that it is very slow loading because it is read in 80 chars at a time and then sent out to the prroper area of memory. A long file can take quite awhile to load. TI to the rescue!!! In their infinite wisdom they included a program on the E/A disk called SAVE which will take a properly set-up DIS/FIX 80 program and convert it to what is called PROGRAM IMAGE. This shows up on a disk directory as the same as a XB or BASIC program with just a simple PROGRAM under the file type column. I did mention a few newsletters back on how to distiguish these files from Basic and XB programs so I won't go into that here. The SAVE program will take the DF/80 program and save it out in 8K blocks. This shows up as files that are 33 sectors long on the disk. The TI is set up to work on 8K blocks of memory so this is the reason the SAVE program is designed to format it's programs in the 33 sector or 8K method. One other item, when the SAVE program operates, on a file larger than 8K it will automatically start a new file after the first 8K is re-saved to disk. This second and subsequent files will have the last digit in their filenames incremented by one in the ASCII table. Now to load one of these files, all that is required is to press #5 for Run Program File and then enter DSK1.program and the E/A loader will handle loading all the associated files.

There is a lot more to the Assembly and E/A module but this should help get you started. In future months I'll go into some of the E/A loaders, their pitfalls and some other hints but that's about all I have room for now.

GENEVE CORNER

by Darrell Wyatt

User Notes

Chart is key to TI-Artist

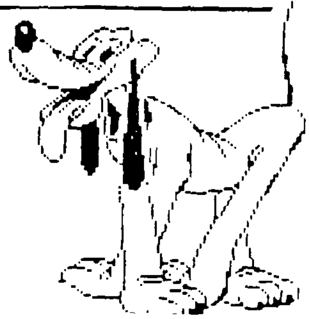
This comes from Jon Hodges, of Richardson, Texas. He writes:

I would like to respond to the Feedback letter from Elton Schooling regarding TI-Artist V2.01. If he looks on the third page of his manual he will see a "Key Layout" that describes the keys that require FCTN to access the designated operation. This is the only place in the manual that even hints at the use of several keys, and it does not do so clearly. They key combination he wants to slow cursor speed is FCTN semicolon.

Last year I taught TI-Artist at a Mini-SIG for our user group and made up the enclosed chart.

We hope readers who use TI-Artist will find this chart to be a handy reference.

Do-It-Yourself Instructions
1. Empty parts on clean workspace.
2. Wire components according to schematic diagram.
3. Plug in and turn on.
4. Write programs.



ARTIST FUNCTIONS

SYMBOL	FUNCTION	TYPE	USE OF ENTER/FIRE	COMMENTS
D	Draw	cmd	start/stop	hold fire down
P	Point	cmd	place	
L	Line	cmd	begin/end	
K	K-Line	cmd	begin/intermediate	D to exit
R	Rays	cmd	center/start/stop	D to exit
F	Fill	cmd	do	SPACE to abort
V	Frame	cmd	lst corner, opp. corner	
X	Box	cmd	lst corner, opp. corner	fills w/pattern
O	Circle	cmd	center/edge	
Q	Disk	cmd	center/edge	fills w/pattern
CTRL-A	Clear Image	cmd		leaves color, pattern
H	Hor/Vert	cmd	begin/end	
N	Swap	cmd	new color/old color	pick new color, place on old color.
I	Invert	mode		negative image
E	AlphaNumeric	cmd	Begin lower left/stop	does not use fonts, use CTRL x for width, FCTN x for height
CTRL-B	Clear Color	cmd		leaves pattern
S	Store	menu		load/save/index picture
Z	Zoom	mode	select window	move with cursor cmds
M	Mirror	mode		4 reflections
C	Hard Copy	menu		
FCTN-.	Plot/Erase	mode		
FCTN-,	Foregnd Color	cmd	change to next color	
FCTN-;	Cursor Speed	mode		fast/slow toggle switch
F icon	Foreground/ Background	mode		color chosen will be fore- ground or background
P icon	Pattern in use	mode	display next pattern	only "P" is solid
- icon	Color Cursor	mode		use P to clean up borders

ENHANCEMENT FUNCTIONS

M	Move w/o Color		select top left/bottom right/place	T to check
N	Move w/Color		select top left/bottom right/place	T to check
C	Copy w/o Color		select top left/bottom right/place	T to check
D	Copy w/Color		select top left/bottom right/place	T to check
A	AlphaNumeric	menu	enter text/place text	SPACE to exit, T to check
	Use a Slide		select/place	SPACE to exit
S	Slides	menu		
	Define		pick box/define slide	SPACE to exit
	Erase		pick slide	SPACE to exit
	Rotate		pick slide	SPACE to exit
	Load Slide file			__S added to filename
	Save Slide file			__S added to filename
	Load Instance		enter name/place instance	T to check __I added to filename
	Save Instance		enter name/select top lft/bottom right	__I added to filename

SPACE BAR is used to toggle between the MENU screen and artwork. It can also be used to abort any function except ALPHA, STORE, SAVE/LOAD, or COPY.
Keyboard cursor movement: Horizontal/Vertical—S/E/D/X DIAGONAL—W/R/C/Z

Newsletter Editor
1334 Aikins St.
Winnipeg Manitoba
Canada R2V 2C7

Edmonton 99ers
P.O. Box 11983
Edmonton Alberta
T5J 3L1