The Ottawa T.I.99/4A Users' Group

VOLUME 7 NUMBER 03.....MARCH 1988

DON'T FORGET THE MEETING -- MARCH 1, 1988

P.O. BOX 2144,STATION D,OTTAWA
*** ONTARIO,CANADA K1P 5W3 ***

COMING EVENTS

March Meeting:	March 1, 1988 7:30 p.m.	Merivale High School
TI FEST: It's Almost Here!	March 5, 1988 9:00 a.m. - 5:30 p.m.	Merivale High School Mark it on your calendar and plan to attend! Contact Jane Laflamme for more information or to offer your help.
Software Contest:	Deadline - March 1988 meeting	Contact Bill Sponchia for more information.
Beginners Assembly:	March 16, 1988 7:30 p.m.	Dick Piché's home Contact Bill Sponchia for further information.

Newsletter Deadline: March 15, 1988

EDITOR'S NOTES
from
Ruth O'Neill

Well, the even of the year is almost here. The 1988 Canadian TI-FEST is happening on March 5th, and it promises to be a huge sucess. Jane Laflamme has lined up a number of speakers for that day, and although I would like to attend all of the talks, it just isn't possible. I'm sure that will be so for others as well, so I'd like to have small summaries of each for the newsletter. I'll be looking for volunteers to report on talks that interest them -- not necessarily formally. All I need is a few points about the talk, and perhaps a personal comment or two. I'll be circulating a list for people to sign up, but if you don't see it and are interested, please contact me at 234-8050. This could be a big favour to all of those who just couldn't make it to a talk, perhaps because of working to make the FEST a success for our visitors.

As the editor of the newsletter, I am in the best position to hear any feedback the readers have. When someone enjoys a particular article and tells me so, I try to pass that on to the author, but it just isn't the same as hearing praise firsthand. A number of our writers have had articles reprinted in other newsletters, which is a clear indication of the merit of their work, and Computer Shopper praised our newsletter for it's "20 pages of pure information", which also reflects favorably on our writers. My point, finally, is this: if you read an article here and enjoy it, tell the author. How else will they know? And if they know that they are appreciated, they will write more. There would be no newsletter without them. Of course, if you would like to join this crack team of writers, there's always room, and you would be welcomed with open arms.

Once more, thanks go to Lucie Dorais for this month's cover art.

The President's Two Cents' Worth
by Michael Taylor

Our third annual TI-Fest is upon us, and while all of us wait with some apprehension for its success, none do so more, I'm sure, than its coordinator, Jane Laflamme. Hard work and planning do not alone guarantee a good show; but add vision, and a good measure of cooperation from our members, and you have the recipe for success. Jane is a tireless worker; she has been planning this Fest since her previous stint as co-ordinator, two years ago; she has vision; and she has elicited the cooperation of many members.

Still, we the members must not fail her, the Fest, or the TI community, in these last critical minutes, with success so close. If no one has contacted you directly, please offer your services for a few hours. We need people to set up and take down tables - not a glamorous job, but an essential one. Perhaps more importantly, though, we need people to attend the show - paid admittances! Talk the Fest up with people you know who are letting their TI languish. Bring them along. Bring your family along. It is important that the distributors and the developers see a positive interest in their products. You do not have to buy anything (except your admission - not expensive), just browse; but you are going to see an exciting piece of software, or a real bargain in the used equipment sale, that you can not pass up, and you are going to kick yourself if you haven't brought along some funds for that eventuality.

On a different note, on behalf of the club, I want to thank Stephen Bridgett for the tremendous effort he has put in as Library Chairman. Stephen was the driving force that produced the recent library catalogue - a comprehensive listing of all our club software - but it is not just for this that he deserves recognition. He has worked hard on many fronts to make the library a strong and useful asset. Now he would like a rest. Stephen will remain active with the library, but he is passing on the reins to Dave Morrison. Thank you, Stephen, for a most commendable job. Welcome, Dave.

When I single out an individual for your attention, I feel badly for not mentioning the others whose efforts contribute to the success of our users' group. The club survives and grows due to the work of many members. It is not feasible to mention them all in any one column.

BROWSING THE LIBRARY
--with **STEPHEN BRIDGETT**

Dave Morrison now has the library under his control. He and I are working toward the Fest, and there is a lot to do. This year we will have the complete library on hand as well as two systems for copying diskettes. Users will be able to purchase a copy of any disk in the library, as well as the catalogue on disk itself. Steve McWatty, who was our Fareware representative at the Fest last year, will be on hand too. His knowledge of the Fareware packages will be a big help.

The Fest promises to be a success and I want to thank all those of you who received my pleading telephone calls for systems and general support. Let's put the icing on the cake, and come on out Saturday the 5TH. There will be many, many fine folks from places far and not so far away. They have lots to show us and lots of questions, too. It's a great day to be a TI owner and even better to be a member of the Ottawa Group. Let's show some Ottawa hospitality, and Club enthusiasm. All those who participate are guaranteed a good day. Remember too, the few days before the Fest are the critical ones. If you can help out at all, Jane is waiting by the phone to get you involved. Phone 837-1719, and your club will take on far more meaning for you than you had previously imagined.

If you doubt it, just ask any of the fine folks that have helped me out as librarian over the last two years. I don't want to miss a one; there is danger in starting, but I want them all to know how much the club appreciates their past and present help.

Thank you: Bob Boone, Lucie Dorais, Art Green, Bob Lanoy, Frank Munro, Dan McCormick, Steven McWatty, Al Palmer, Bill Sponchia, Peter Arpin, Bernard Vanden, Berry Minuk, Henri Monat, Ruth O'Neill, Dick Piché, and Charles Earl. Apologies to anyone overlooked.

One of the giants of our Club is a guy of endless generosity and computer smarts. Art Green's software has won him admiration from the entire TI community, yet there are many of us who have not seen or used his software. This month the disk offered is Art's 'RAG Utilities, ver 4'. You won't find a program written by Art that doesn't run or doesn't have Docs included.

RAGREENV4 Free 177 Used 183

Filename	Size	Type/No.	12	P
BASIC-XR	26	PROGRAM		U
DISKCOPY	13	PROGRAM		U
DISKCTLG	29	PROGRAM		U
DISKINIT	12	PROGRAM		U
DOC-UTIL	22	DIS/VAR	80	U
DSKTOCAS	25	PROGRAM		U
DSKTOCAT	9	PROGRAM		U
LOAD	6	PROGRAM		U
PRTSETUP	14	PROGRAM		U
SETUP-1	2	DIS/VAR	80	U
SETUP-2	2	DIS/VAR	80	U
TERMINAL	21	PROGRAM		U

File names are nearly self-explanatory. The first program provides a clean, neatly re-written listing of your Basic or XBasic programs with a cross reference variable table. It's a must for Basic and XBasic fans. Print-set-up is a dream. I've been using it since I got my printer. Pick up this disk and you'll be reaching for it time and again.

Next month, Dave Morrison will be your new author on this column. Please give him your support. The library can't exist without your feedback.

See you at the Monthly,.....Stephen.

GENEVE UPDATE (As at Feb. 16/88)
by Jack Adams

Myarc has just recently mailed out, to those of us who have sent in our Geneve WARRANTY REGISTRATION forms, three diskettes of updated software.

Diskette 1 - MYARC-DOS Version 1.01
2 - MYARC GPL Interpreter Version 0.99
3 - Cartridge saver; Multiplan Upgrade; MYWORD Processor Version 1.10

Apparently, no further upgrades are to be expected for Multiplan or Myword Processor. On testing MYARC-DOS, I have noticed that DSK5 cannot be formatted and therefore cannot be used unless you are working from GPL. In the GPL mode of operation, DSK5 is automatically formatted as double sided 720 sectors. Version 0.98 did not do this.

As promised, here is a brief review of MYARC's MYART.

This software is loaded directly from MYARC-DOS by simply entering MYART at the A> prompt for DSK1. The version in review is 1.0. Something that will help the first user immensely is a template that fits over the function keys. This template contains all the commands required from the key-board and the mouse and is available from Canaria Data Incorporated, 264 Weber St. W, Kitchener, Ontario, N2H 4A6.

I am writing this as a first user myself. I have tried TI-ARTIST previously and find MYART superior in ease of handling and in artwork in general. Much finer work can be expected of MYART. There are two resolution modes, 256 and 512. You should note here that if you start a picture in either mode, you cannot transfer one to the other without losing all your work. A picture saved in one mode will not load into the other mode from disk. Also note that if the diskette becomes full, there is no warning that this is occurring. You may end up with only part of your work saved. The saving routine stops when the diskette is full and no warning is given. It's best to check your diskette directory before using it.

512 MODE

512 mode offers finer detail and ability to mix your own colours from the basic Red, Green, and Blue combinations. You may also choose from a palette of 16 colours that will be displayed at the bottom of the screen. The art work achievable with 512 is truly amazing. With zoom and control over the speed of movement of the icon, fine detail can be displayed easily . The pixel shape in this mode is a vertical rectangle; consequently, any circles drawn will be elliptical. Unfortunately there is no way to correct this.

256 MODE

When MYART is first loaded, it is automatically in this mode of operation. This mode is identified by the fact that a multicoloured bar appears at the bottom of the screen. This bar contains 256 colours which are not all displayed at one time. The colour bar can, however, be moved left or right along the bottom of the screen by use of either the mouse or the arrow keys. The behaviour of the colour bar gives the appearance that it is on an endless drum. The pixel shape in this mode is square and the circles drawn are nearly circular (actually a little like horizontal ellipses).

IN GENERAL

There are 15 commands in all. A command is enacted whenever the first letter (usually) of the command is typed from the key-board. These commands will produce Boxes, Rectangles, Circles and Straight lines and allow free-hand sketches in any colour. They will also allow you to gain access to a Help File, to Load and Save, Format a disk, Cut and Paste, Type Text, Fill a space with a chosen colour, and provide several levels of Zoom.

By depressing CONTROL plus a letter, it is a simple matter to toggle between 512 and 256 mode, rotate text in 90-degree steps, clear the screen, display a disk directory, change the icon colour, and print what is on the screen. When printing the screen you have two sizes available, and you can print a picture in shades of grey or in outline only. (Sorry, I don't think colour printing is possible.)

The mouse has three buttons. One button toggles the colour palette, another controls the levels of zoom and erases the most recently produced line or product (depressing it a second time brings it back), and the third enables colour change and engages the activity desired on the screen. The speed of movement of the icon can be controlled in five stages by simply pressing the number keys 1 to 5. This is important, particularly when working on details while in zoom mode.

All in all, I feel that this is a super product that is available locally for about $200 from any of the Geneve dealers listed in this bulletin.

"Questions & Answers"
by Bill Sponchia

Last month, I answered the question about whether it was possible to send printer command codes to the printer from Multiplan. I had stated that there was nothing in the manual about this but that I had heard that it could be done. Well, good news - I found out that this can be done. This information comes from Denis Deny, a former member (he has since deserted and gone "elsewhere").

The method involves using a disk sector editor and putting the information directly into the saved file. I know that it works, because I actually did it. The steps I used follow:

1. It is best to work with a freshly initialized diskette so that there is not a lot of other junk on it to confuse matters.

2. Load Multiplan,then type 10 A's into cell R1C1. This will just make it easier to find later. Save this template onto the blank diskette.

3. Using a disk sector editor (I used Disk + Aid), locate where this cell is saved. When I did it, it was on sector 24.

4. Edit the sector by typing in the printer codes you wish in place of the A's. You must input the change in Hexcode and for each "A" not used, replace with >20.
   For example, to set up condensed printing, you must input "0F"; double strike - 1B 47; etc.

5. Rewrite this sector to the diskette, and you have your Multiplan file which sends printer command codes.

Well, enough said on that topic, now onto a different subject.

Q - I changed the 12Mhz in my TI99/4A for a 14.3Mhz and it does increase the speed as expected, but when I try to send something to the RS232 I get garbage. Is it a problem of software or hardware? J.P. Dubois, Quebec

A - From Dave Hamm of Greenwood, Nova Scotia comes the answer. (Note: he says that this comes "off the top of his head" as he doesn't have his schematics with him - but just listening to him tells me that this is more than just a guess). The problem is hardware. When using the serial port of the RS232 card, the baud rate is determined by using the Mhz of the computer. Because you increased the Mhz, it calculated out an unusual baud rate (one that the machine doesn't recognize) and therefore the output is garbage. As for a solution required -- I'm sorry, but that would require a hardware change in the ROM in the RS232. Let us know if you figure out a way. This problem occurs only when you use your serial port. The parallel port of the RS232 should work okay.

COMMENTARY ON FEST '88

I would like to take this opportunity to welcome all fellow TIers and GENEVE users who are visiting Canada's Capital on this our Third Annual National TI-FEST.

If this is the first time you have seen our Newsletter, we might mention that we are very proud of it. We have been very fortunate to have several great editors, each one making their own special mark on the newsletter and building on the previous editors ideas, to the point that Computer Shopper named it Newsletter of the month in their February 1988 issue. Congratulations to Ruth O'Neill and the many contributors. If you would like to see more of our Newsletters, we invite you to join our group. For the low, low price of $20, you will receive ten news-packed issues for a year, or for $15, you will receive seven for the balance of this year.

This brings me to point out that The TI-99/4A National Association of Canada (NUAC) has amalgamated with The Ottawa TI-99/4A Users' Group. The Ottawa group will now serve as a uniting force for the TIers across Canada (and other parts of the world). We invite your group and individuals to share news items covering their groups or particular interests, and also articles and reviews on new software, fairware, hardware, or any other items of interest to the 99er and GENEVE users.

Back to the FEST - we in Ottawa attempted to invite as many supporters of the TI and GENEVE as we could, and give you as much up-to-date information as possible, with the added advantage of meeting and socializing with other individuals and groups. Perhaps your group might like to host the FEST next year... Ottawa would give as much support as possible. Think about it.... suggest it to your own group. You will receive more advantages by doing so than you could ever imagine!

To you great Ottawa members who gave me the support to make this enormous project become a reality, many of whom have gone above and beyond the call of duty - THANK YOU!!

We hope that you (will/did) enjoy the FEST, and remember, only YOU can keep our computers alive and well, with YOUR support of the dealers, software writers, hardware producers, and of each other! Ottawa has given you one means. Now it is up to you to go home and keep the spirit alive!

Jane Laflamme
1988 FEST Co-ordinator

PREPARE FOR THE USED EQUIPMENT SALE:
notes from Lucie Dorais

Once again, we will have a Rummage Sale at the FEST, and I would like to make some suggestions, hoping that you will understand and follow them:

(1) Please, mark every piece of software and hardware CLEARLY with your name and maximum price wanted.
(2) In order to sell quickly, make the software more desirable with some packaging; use the original one if you still have it, or use Ziploc (or any sandwich) bags.
(3) To save us precious time, please prepare a list in advance, in duplicate (one for you, one for us), following this example. If you don't want us to discount the item at some point during the day, please write "NO" in the "Minimum price" column.

NAME : _____

ITEM name	MAXIMUM price	MINIMUM price	SOLD for
1 - _____	$_____	$_____	$_____
2 - _____	$_____	$_____	$_____
etc.			

EXPANSION PORT INTERFACING: Part 4
By David Caron

In last month's newsletter I published a pinout and a memory map for the TMS 9901 Interface chip. This month I will explain how to use it. You will need the diagrams from the last month, so if you do not have them, pull out the February newsletter.
There are five assembly operations which let you access the TMS 9901 through the CRU lines (Communications Register Unit). I have been told that the term CRU refers only to the three lines, CRUIN, CRUOUT, and CRUCLK, although Unit seems to refer to something specific, such as a chip. I personally believe the reason you cannot assume that the 9901 is the CRU is because you can use conventional logic chips in place of the 9901 if you only wish to perform an elementary function such as turning a CRU address on and off (as on a ramdisk, since it has no 9901).

The five assembly operations are: LDCR, SBO, SBZ, STCR, and TB. A detailed explanation of each can be found in the Editor Assembler manual. For now, I will just give a short description of each. All the operations require that you give the CRU base address in register 12 under the following format:

REGISTER TWELVE:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
NOT USED			CRU addresses which select a particular 9901 in the expansion box. If bit 3 is zero then the console 9901 is automatically enabled only.					NOT USED		S4	S3	S2	S1	S0	NOT USED
										five bit address which select the particular CRU bit on the 9901 chip					

A detailed memory bit-map of CRU address(A3-A7) can be found on page 407 in the editor assembler manual.

LDCR @@BUFF,X transfers X number of bytes from @@BUFF (starting with the least significant byte) to the TMS 9901 starting at the address in R12.
STCR @@BUFF,X same as above, except that the data is transferred from the 9901 to @@BUFF.
SBO X Sets the CRU bit indicated in R12 +X to 1 or high.
SBZ X Sets the CRU bit indicated in R12 +X to 0 or low.
TB X Reads the CRU bit indicated in R12 +X and sets or resets the equal status bit according to the value (high or low) read.

For our purposes, we will use only SBO, SBZ and TB for simplicity, and we will assume that the CRU base address is >1000, which would mean a value of >1000 in R12.

CRU BIT 0 is the control bit: when it is set low (SBZ 0), CRU bits 1-15 will function as interrupts; when it is set high (SBO 0), CRU bits 1-14 will function as the countdown register for the clock.

CRU BITS 1-15 are the Interrupt/clock mode bits. When they are in the interrupt mode, the interrupt request pin (pin 11) goes low if any of the 15 interrupt pins are forced low. The bit number for that pin also gets sent out on the INT code lines (IC 0-3). If two or more of the interrupt pins are forced low at the same time, the one with the lowest CRU bit number takes priority and its CRU bit number is sent out on the interrupt code lines. By the way, the TI-99/4A computer system does not use the interrupt code lines, so all the 9901 can tell the CPU is that an interrupt has occurred on one of the 15 interrupt lines. The above assumes that every interrupt line has been enabled. Yes, you can turn on and off any or every one of the 15 interrupt inputs individually. To turn off CRUBIT 1, all you do is SBZ 1 which will set the MASK bit for interrupt one to zero. When that pin (pin 17) is forced low, (INTREQ) and IC 0-3 will remain high (they are normally high unless an interrupt occurs), but the register for CRU bit 1 will emulate what is happening to the interrupt bit, so TB 1 will tell you whether that pin (pin 17) is high or low. (It will function as an input port when the mask is turned on or off). To restore operation of pin 17 simply do SBO 1 and (INTREQ) will go low again. Then pin 17 (CRU BIT 1) is forced low, and IC0, IC1 and IC3 will go low with IC4 high to indicate that CRU

BIT one is the highest priority interrupt pin that has been forced low. The above will work for all 15 interrupt pins. Is this not great? But I have only explained one third of the features of the 9901.

If CRU bit 0 is set high (SBO 0), the CRU bits 1 to 14 will function as registers for the clock. When writing to these registers, the clock will reset after every write operation. Although I can not find out how the clock counts down, let's assume that CRU bit 14 is the least significant bit and CRU bit 1 is the most significant bit. Once you have written the start numbers to the registers, the clock will countdown. When it reaches zero, the interrupt request line will go low to indicate that the clock is "ringing". The clock will immediately start counting down again from the number that you placed in the clock register; however, the INTREQ line will stay low until you write a one into the CRU bit 3 in INTERRUPT MODE. In other words, to "hit the snooze button" you must return to the interrupt mode with SBZ 0 and execute a SBO 3 or SBZ 3. Incidentally, it would be a good idea to set all the interrupt mask bits to zero (to disable all interrupts) while accessing the 14 bit clock register. (Perhaps that should have been mentioned earlier). While the clock is in mode SBO 0 you can read the present countdown time in CRU bits 1 to 14 with TB 1-14 (fourteen TB), TB 1, TB 2, TB 3, etc. After the countdown reaches zero, your initial value will reappear in the 14 bit clock register and the clock will decrement the register again to zero. The register will decrement at the rate of 349 ms if the 9901 is used with a 9904 clock generator. So if you programmed the 14 bit clock register with all ones, the timer will ring in 349 X 2 14 or 349 X 16384 or 5718016 ms which is 5718.016 s or 95.300267 minutes or 1.5883378 hours. Petty neat, eh! To turn off the clock altogether, simply set the 14-bit clock register to zero.

If you wanted to keep track of real time, but also used the disk drive extensively, you could simply set the clock register with 51 and know that it would ring in precisely 17.799s (0.349 X 51). Of course, you would have to turn off all the interrupts so that the VDP interrupt would not make INTREQ low and fool the computer into thinking that 18 seconds had already passed. When I refer to the VDP interrupt, I mean the console 9901 which causes an interrupt every 1/60 of a second because the VDP Vertical Snyc line forces CRU bit 2 low every 1/60 of a second. Therefore, CRU bit 2 (pin 18) would have to be disabled. CRU bit 1 is the external interrupt input, meaning that pin 17 on the console 9901 is connected to pin 4 on the expansion port, which I am assuming connects to the INTREQ (pin 11) on other 9901's in the expansion interface.

For timing control, you would be better off using the console 9901's clock and disabling its CRU bit 1 rather than using an external 9901's clock and having its INTREQ triggering the console's 9901's CRU bit 1 to cause the INTREQ line on the console 9901 to go low. This happens to be the only line that is connected to the CPU INTREQ. When the CPU's INTREQ (pin 32) goes low, the CPU will stop whatever it is doing and execute its regular interrupt routines such as checking for sprite motion, automatic sound generation, quit key, screen blankout timer, etc. However, the CPU can be convinced to ignore this INTREQ by the command LIMI 0. A description of this command is found in the Editor Assembler. If you have a copy of the TI-99/4A console technical data book, you will see how the console 9901 is linked to the CPU.

The other way of detecting whether a 9901's clock is "ringing" is to simply check CRU bit 15 (TB 15) in the clock mode (SBO 1). TB 15 will tell you the condition of the INTREQ (pin 11) for that chip. This should work for all the 9901's, including those in the expansion box. You could have all the 9901's timers counting down at the same time if those timers were not used for system purposes. Remember, though, the interrupt inputs to those 9901's must be disabled, or you will not know weather the INTREQ was triggered by the clock or interrupts, and do not forget to turn it off with (SBO 3) or (SBZ 3) in the interrupt mode.

Upon turning on the computer, RST1 is held low for a short period of time to reset the 9901 interface chip. When this is done, all interrupts will be disabled as well as the clock, IC 0-3 will go low (not high) (I suspect that IC 0-3 will go high if any of the interrupts are enabled), INTREQ will be set low, and all the I-O registers will be set to inputs. (Next month I will explain how they work). There is also a software reset for the 9901 (RST2), which is done by SBZ 15 while in the clock mode. So the two software commands that are equivalent to forcing RST1 low (pin 1) are SBO 0 and SBZ 15 (RST2). Both ways have the same effect on the 9901. If you have any questions or do not quite understand this complicated interface chip, feel free to call me at 745-4618.

VDP Tutorial
by Hal Tonkin

This article is an attempt to remove some of the mystery surrounding the VDP. It tries to explain in simple terms, with examples, how the VDP operates. One of the problems with the existing documentation in the Editor/Assembler manual is that it is a reference manual, not a teaching one. Thus it describes everything at once about the VDP, and ends up being very confusing, so let's look at it piece by piece, and function by function.

The first thing to note is that the VDP actually consists of two parts. The first part is the 9918A Video Display Processor (VDP), and the second part is the VDP memory.

The VDP is like a second CPU whose purpose is to display the contents of the VDP memory on the screen in one of 4 modes, and to control the generation and display of sprites on the screen.

The VDP memory consists of 16K (16384) bytes and is accessed by both the 9918A VDP and the 9900 CPU. The VDP memory contains several tables that tell the VDP what to put on the screen. However, these tables do not take up all of the VDP memory, and so the remainder of the VDP memory may be used by CPU programs to store other information. Some examples of this other data are sound lists, basic programs, and file I/O control information and buffers. As long as the display information required by the VDP is not overwritten by the CPU programs, all is well. Otherwise, you end up with some strange displays!

Let's now ignore all the CPU stuff in the VDP memory that is not used to create a screen display, and concentrate on how the VDP uses the VDP memory to create a display. We will also post-pone a discussion of SPRITES until later.

The VDP has the following four modes of displaying data:

1) Graphics. 24 rows of 32 characters (8 X 8 pixels)
2) Text. 24 rows of 40 characters (6 X 8 pixels)
3) Multi-Colour 48 rows of 64 squares (4 X 4 pixels)
4) Bit-Map 192 rows of 256 pixels

Some examples of where these modes are used are:

a) Basic and Extended Basic use only graphics mode.
b) C starts out in text mode, but can switch to any other.
c) Assembler starts out in graphics mode, but can switch to any other.
d) TI-ARTIST operates in bit-map mode.
e) The TI Editor operates in text mode.

The VDP is controlled by seven 8-bit registers. The contents of these registers will be described as the different features of the VDP are described. The display mode is selected by bits in registers zero and one as follows:

MODE	BIT 6	BIT 3	BIT 4
GRAPHICS	0	0	0
TEXT	0	1	0
MULTI-COLOUR	0	0	1
BIT-MAP	1	0	0

Care must be taken when altering these bits, because of the function of the other bits in these registers. Since these are write-only registers, you must know what you want (or previously had) in the other bits as well before selecting a new mode.

The VDP memory is accessed by reading and writing to CPU memory. There are pre-defined assembler routines that allow you to easily read and write to the VDP registers and VDP memory.

GRAPHICS MODE

Graphics mode is the default mode of the system. In this mode, there are 24 rows of 32 characters displayed on the screen. This gives a total of 768 character positions. The VDP processor requires three tables in the VDP memory to tell it what to put on the screen. The three tables are the Screen Image table, The Pattern Descriptor Table, and the Colour table.

Introducing...

TELCO

THE EXCITING NEW TERMINAL EMULATOR FOR THE TI99/4A & GENEVE 9640

BY CHARLES EARL

FEATURES:

MENU DRIVEN with context-sensitive HELP SCREENS

FULL-FEATURED AUTODIALER - sets baud rate, parity, terminal, etc. for each number dialed, and redials multiple numbers in sequence

3 DIFFERENT TERMINAL EMULATIONS available: ADM3A, ANSI, & D410

OPTIONAL STATUS LINE with elapsed time clock

LOGGING OR SCREEN DUMP to any device

8K REVIEW BUFFER - not destroyed by file transfers

USES GENEVE 80-COLUMN MODE, emulates 80 columns on TI with scrolling windows or window lock

COMPATIBLE WITH HRD, MYARC 512K RAMDISK, and CORCOMP RAMDISK

ASCII & XMODEM TRANSFERS, PRINT SPOOLER, and much more!

Telco is being marketed as USER-SUPPORTED software. Look for it on a BBS near you, or to order directly, send $20.00 to:

Charles Earl
34 McLeod Street
Ottawa, Ontario
Canada
K2P 0Z5

See you at the FEST on March 5th!!

The Screen Image table is 768 bytes long. Each byte of this table corresponds to one of the character positions on the screen. The bytes in this table do not contain the character to display, but a value from 0 to 255 that, when multiplied by 8, gives an index into the Pattern Table. The Pattern table is 2048 bytes long. Each 8 bytes describe the pixel pattern for the character to be displayed. For example, if the 5th byte of the Screen Image table contained the value 65, then the VDP adds the value 65 times 8 (= 520) to the start of the Pattern Descriptor table to get the pixel pattern to display on the screen at row 1 column 5. When the system is started, the default pattern Descriptor table is loaded, and the pattern at location 520 of the Pattern Descriptor Table shows up on the screen as the letter 'A'. The rest of the ASCII character patterns are also loaded at their respective positions in this table. In Basic, when the statement CALL CHAR(character-code, pattern-identifier) is made, the character identifier is multiplied by 8, and the pattern-identifier is inserted at this location of the Pattern Descriptor table. The third table required is the Colour table. It is 32 bytes long, and describes the foreground and background colours for the characters. There are 256 possible different patterns, but the patterns are divided into 8 character groups, giving 32 groups of 8 characters. Thus, character patterns 0 to 7 are in the first group, 8 to 15 are in the second group, etc. The 32 bytes of the Colour Table correspond to these 32 groups, and specify the foreground (upper 4 bits), and background (lower 4 bits) colours of the pattern that is displayed. If we use our previous example of byte 5 of the Screen Image table containing the value 65, then 65 is in the 8th group (64 to 71 of characters. If the 8th byte of the colour table contained the value F4 (hex), then the character would appear on the screen as a white letter 'A' on a dark blue background (foreground value 15, and background value 4). In Basic you are allowed to redefine only characters 32 to 156 and they have redefined the group numbers, but when you use the CALL COLOR(CHARACTER-SET,FOREGROUND-COLOUR,BACKGROUND) statement, it is the Colour Table that you are modifying.

VDP registers 2, 3, and 4 tell the VDP where these three tables are located in the VDP memory. They contain a value, that when multiplied by a constant give the base address of the table. These are the constant multipliers, and the default values of these tables when in graphics mode.

REGISTER	DESCRIPTION	DEFAULT CONTENTS	MULTIPLIER	DEFAULT BASE
2	Screen Image	0 (hex)	400 (hex)	000 (hex)
3	Colour	C (hex)	40 (hex)	380 (hex)
4	Pattern Descriptor	1 (hex)	800 (hex)	800 (hex)

That's enough theory for now. Try the following assembler program which shows how to write to the three tables to put a character on the screen.

```
* DEMONSTRATION OF GRAPHICS MODE ACCESS TO THE
* SCREEN IMAGE TABLE
* PATTERN DESCRIPTOR TABLE
* COLOUR TABLE

REF VSBW  * VIDEO SINGLE BYTE WRITE
REF VMBW  * VIDEO MULTIPLE BYTE WRITE

DEF START * PROGRAM ENTRY POINT

IMAGE  EQU  >000   * BASE OF IMAGE TABLE
PATRN  EQU  >800   * BASE OF PATTERN TABLE
COLOR  EQU  >380   * BASE OF COLOUR TABLE
PAT128 EQU  >400   * OFFSET FOR CHAR 128 ( 128*8)
CHR128 EQU  >8000  * CHARACTER CODE 128 IN UPPER BYTE
COL128 EQU  16     * COLOUR GROUP FOR CHAR 128
ROWCOL EQU  5      * CHARACTER POSITION IN IMAGE TABLE

START
LWPI MYREGS     * LOCAL WORKSPACE REGISTERS
LI   R0,PATRN   * R0 = BASE ADDRESS OF PATTERN TABLE
AI   R0,PAT128  * ADD OFFSET FOR CHAR 128 PATTERN
LI   R1,BLOB    * R1 = ADDRESS OF PATTERN TO LOAD
LI   R2,8       * NUMBER OF BYTES TO LOAD
BLWP @VMBW      * WRITE TO PATTERN TABLE
*
* SET THE CHARACTER COLOUR TO BE WHITE ON BLACK
*
```

```
LI   R0,COLOR  * R0 = ADDRESS OF COLOUR TABLE
AI   R0,COL128 * ADD OFFSET FOR CHAR 128 COLOUR GROUP
LI   R1,>F000  * R1 = CHAR TO WRITE IN UPPER BYTE
BLWP @VSBW     * WRITE TO COLOUR TABLE
*
* WRITE THE CHARACTER 128 TO THE SCREEN IMAGE TABLE
*
LI   R0,IMAGE  * R0 = BASE ADDRESS OF IMAGE TABLE
AI   R0,ROWCOL * ADD OFFSET FOR ROW 1 COL 5
LI   R1,CHR128 * CHAR 128 IN UPPER BYTE
BLWP @VSBW     * WRITE TO SCREEN IMAGE TABLE
LOOP
LIMI 2         * ENABLE INTERRUPTS
JMP  LOOP      * TO ALLOW FCN QUIT
*
* DATA
*
MYREGS BSS  32      * LOCAL REGISTER WORKSPACE
BLOB   DATA >FF99,>9999,>9999,>FFFF   * PATTERN
END
```

Good luck with this and next time we will look at how to display in Text Mode.

References: 1) Editor/Assembler Texas Instruments Home Computer
2) TMS 9918A Video Display Processor

Interfacing Programs for the Geneve 9640
by Charles Earl

This article will discuss the interfacing of TI programs to run under the Geneve 9640 GPL environment. The GPL environment simulates the TI-99/4a operating system and sets the 9640 to run in TI mode.

Although most programs will run on the 9640 computer, there are a few exceptions. The cause of this incompatibility is the kscan routine. Any program which is to run successfully on the 9640 must not contain any direct cru addressing for keypresses. On occasion, programmers have done this to increase the speed of their program, but with the 9640's increased speed this is not required.

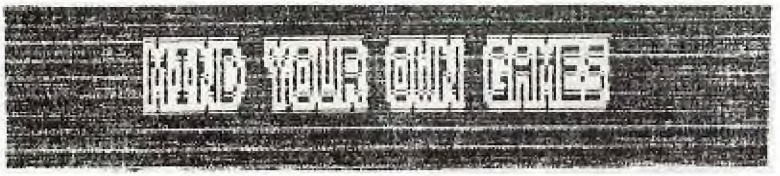
To have your program detect the 9640, check the value of address >0006. This value should be >0308 on a 9640. This is useful for making your program run under both environments without having to write multiple versions.

The display modes on the 9640 are quite complex with the exception of 40 and 80 column text mode displays. The 9640 has 128K of VDP memory. To stay compatible this memory is divided into 8 16K blocks. Paging is done by setting vdp write only register 14 to the 16k block number ranging from 0 to 7. Normally this will be set to 0. This register is not available on the TI. Any attempt to access it will crash the computer.

The 9640 80-column mode is very similar to TI's 40-column text mode, and the 9640's 40-column mode is identical to the TI's. To set the 9640 to 80 column mode the following vdp write only registers must be set as follows:

Register 0: On the TI this register is set to 0 for text mode. On the 9640 this register is set to >04 for 80-column text.
Register 1: For both the TI and 9640 set this register to >F0.
Register 2: This register is the base address for the screen image table times >400. With the 9640 in 80-column mode, the base address is multiplied by >1000. As well, the bits have been shifted left 2 positions and the 2 least significant bits are set to 1. For example to have an 80-column screen at vdp address >0000 a value of >03 will be loaded into this register. For vdp address >1000 a value of >07 will be loaded instead.

Not only does the 9640 have 80 columns to display, but it can also display up to 26 and a half lines. The half line is the top four pixel rows of each character on the 27th line. This is changed by pressing Ctrl-Shift-Alt. Because of this I find that it is more convenient to use the second 16k vdp block for displaying text. Fewer internal changes are required if the program you are working on uses the VDP.



GAMEAUCRATS

Henri Monat

The Gameaucrats column this month is relatively short (and hopefully sweet). It deals with 2 of the mind games I used to play most, both of which are available through the club's library: Backgammon and Mastermind.

BACKGAMMON

However popular this game is, I had never played Backgammon before I got this program 6 months ago. I had a very rough idea of what the game was about, but knew nothing about the rules. I decided to apply the "Learning by Doing" principle: I tried to learn the rules simply by playing against the computer, and it worked. I am now fully acquainted with the game and I never had to read anything about it. I hope that gives you an idea of how friendly this program is.

The program is in Extended Basic. It has been written by Dennis Webber and was published in the defunct Home Computer Magazine. It is a little slow to execute, but not to the point of being overly tedious. The graphics are nice for an XB program and the interaction is good.

To play is quite easy: press the Enter key to roll the dice; then press the number corresponding to the "square" from which and where to you want your counters to be moved along the board. The computer does all the work for you. It even gives on an ongoing basis your points and those of the computer. Whoever has fewer points is ahead of the other.

One point to remember in playing: because there are 24 "squares" on the board, squares 0, 1 or 2 must be followed by the Enter key or be entered by pressing 00,01 or 02. It took me a while to figure that out.

Finally, I must warn you: the computer is a bad looser. When it wins, it sends me a sarcastic music to remind me that I lost; when I win though, there is no winning tune for me, except for a FLAT NOTE at the bottom of the screen: "Do you want to play again?" Even worse, sometimes when I am about to win for more than 100 points, the program crashes with a "bad value" error... (Confidentially, when I loose for more than ... points, I just pull the plug).

In summary, this program has provided and is still providing me a good, enjoyable time, and I recommend it. Experienced players, however, would benefit from finding an Assembly version of the Backgammon program: it would run faster and its algorithm would likely be more powerful.

MASTERMIND

Have you ever played Mastermind against another player? How boring it is when it is your turn to be the dummy! With this program, the computer is always the dummy.

The version I use is in French. It has been written in Assembly language by Jean-Luc Bazanegue and was published in the French 99 Magazine, in December 1983.

You have 10 chances to find up to 10 colours chosen at random by the computer. You must not only find the right colours but also the right order each of them were placed at random by the computer in 5 different squares. Press the number corresponding to the right colour for each square and the computer gives you how many colours you have found and how many are in the right place. The computer even points out to you the errors you have made compared to previous trials. It also gives you the solution after your last trial, or at any time if you press FCTN 7.

This program is the one I favour for this kind of game. The graphics are appealing and I like the immediate response I receive when I press a key. This program is certainly to be recommended to all mind games addicts.

FAST EXTENDED BASIC

LUCIE DORAIS

Two great events happen in March: TI-FEST and SPRING. I wished to do a little column on music, but was unable to find the score for Vivaldi's "Spring"; anyway, Tex is not very good at imitating violins. So I looked into my music books and found a little piece called "Spring Song", by Schulz (I don't know more about him, sorry).

```
100 REM SPRING SONG (SCHULZ) / L. Dorais, Jan. 1988
110 REM > Title Display routine by David Fink
120 CALL CLEAR :: CALL SCREEN(4):: CALL MAGNIFY(2) ::
A$="SPRING SONG"
130 S=(6-INT(LEN(A$)/2))*16+32
140 R=(6-INT(LEN(A$)/2))*16+1
150 I=1 :: FOR C=S TO ((LEN(A$)-1)*16)+S STEP 16
160 CALL SPRITE(#I,ASC(SEG$(A$,I,1)),I+5,R,C)
170 R=R+16 :: I=I+1 :: NEXT C
180 DISPLAY AT(3,27):"BY": :TAB(23);"SCHULZ"
190 LLB=123 :: LC=131 :: LD=147 :: LFS=185 :: LG=196 :: LB=247 ::
C=262
200 D=294 :: FS=370 :: G=392 :: A=440 :: B=494 :: HC=523 ::
HD=587 :: HE=659 :: DUM=1475
210 T=600 :: TD=T/4*3 :: V1=1 :: V2=5
220 GOTO 230 :: CALL CHAR :: CALL COLOR :: CALL SPRITE ::
CALL SOUND :: CALL MOTION :: CALL DELSPRITE :: !@P-
230 A$="0092D6FEFE7C3810" :: CALL CHAR(97,A$,104,A$,112,A$) ::
CALL CHAR(120,"10101092D6FE7C10",128,"FFFFFFFFFFFFFFFF")
240 CALL COLOR(9,16,4,10,14,4,11,5,4,12,13,4)
250 CALL SPRITE(#13,128,4,161,28,#14,128,4,161,60,#15,128,4,161,92)
260 DISPLAY AT(21,3):"a   h   p":"  x   x   x"

270 CS(TD,HD,V1)                    |   580 CS(T/2,D,V1,LD,5)
280 CS(T/4,B,V1)                    |   590 CS(T/4,40000,V1)
290 CS(T/2,G,V1,LB,V2,LG,V2)        |   600 CALL MOTION(#15,1,0)
300 CS(T/2,G,V1)                    |   610 CS(T/4,FS,V1)
310 CS(T/2,A,V1,C,V2,LD,V2)         |   620 CS(T/4,G,V1)
320 CS(T/4,A,V1)                    |   630 CS(T/4,A,V1)
330 CS(T/4,B,V1)                    |   640 CS(T/4,B,V1)
340 CS(T,G,V1,LB,V2,LG,V2)          |   650 CS(T/2,HC,V1,D,V2,LFS,V2)
350 CS(T/4,40000,V1)                |   660 CS(T/2,A,V1)
360 CALL MOTION(#13,1,0)            |   670 CS(T/2,B,V1,D,V2,LG,V2)
370 CS(T/4,HD,V1)                   |   680 CS(T/2,G,V1)
380 CS(T/4,B,V1)                    |   690 CS(T/2,A,V1,FS,V2,LD,V2)
390 CS(T/4,HD,V1)                   |   700 CS(T/2,D,V1,LD,5)
400 CS(T/4,B,V1)                    |   710 CS(T/4,40000,V1)
410 CS(T/2,G,V1,LB,V2,LG,V2)        |   720 CS(TD,B,V1,G,V1)
420 CS(T/2,G,V1)                    |   730 CS(T/4,HC,V1,A,V1)
430 CS(T/2,A,V1,C,V2,LD,V2)         |   740 CS(T/2,HD,V1,B,V1,LG,V2)
440 CS(T/4,A,V1)                    |   750 CS(T/2,HD,V1,B,V1,LG,V2)
450 CS(T/4,B,V1)                    |   760 CS(TD,B,V1,G,V1)
460 CS(T,G,V1,LB,V2,LG,V2)          |   770 CS(T/4,HC,V1,A,V1)
470 CS(T/4,40000,V1)                |   780 CS(T,HD,V1,B,V1,LG,V2)
480 CALL MOTION(#14,1,0)            |   790 CS(T/4,40000,V1)
490 CS(T/4,FS,V1)                   |   800 CS(T/3,HE,V1,LC,V2)
500 CS(T/4,G,V1)                    |   810 CS(T/3,HC,V1)
510 CS(T/4,A,V1)                    |   820 CS(T/3,A,V1)
520 CS(T/4,B,V1)                    |   830 CS(T/3,HD,V1,LLB,V2)
530 CS(T/2,HC,V1,D,V2,LFS,V2)       |   840 CS(T/3,B,V1)
540 CS(T/2,A,V1)                    |   850 CS(T/3,G,V1)
550 CS(T/2,B,V1,D,V2,LG,V2)         |   860 CS(T/3,HC,V1,LD,V2)
560 CS(T/2,G,V1)                    |   870 CS(T/3,A,V1)
570 CS(T/2,A,V1,FS,V2,LD,V2)        |   880 CS(T/3,FS,V1)

890 CS(T*1.2,G,V1,1475,30,1475,30,-4,1)
900 DISPLAY AT(24,3):"AGAIN?  Y" :: CALL DELSPRITE(#13,#14,#15)
910 ACCEPT AT(24,11)VALIDATE("YN")SIZE(-1):A$ :: IF A$="N" THEN END
920 DISPLAY AT(24,1):"" :: GOTO 250
```

Extended Basic does not add anything to the CALL SOUND statement of BASIC, except that you could use multiple statements lines. I decided not to, because single CALLS are easier to read. And to type?? Well, what about using the REDO function? Just type the first CALL SOUND, line 270, and REDO it as you need; but don't forget to change the line number! (In the program listing, I wrote all the CALL SOUNDS as CS: you will need to type the full statement. I did that only for page setting purposes). A tip: lines 610-710 are an exact duplicate of lines 490-590, so just change the line numbers when you redo them. And if you have Triton's SXB, well, just do "COPY 490-590,610,10", and presto! all the lines are copied in one statement (10 is for the increment).

The nice title routine comes from our friend David Fink, who used it to fill the page of his renewal letter; thank you Dave! It is in lines 120 to 170, and came as a two-line (multiple statements) routine. You can use it with any program, but the title has to be 12 characters or less to work. To get all the letters in the same color, just give a single color value to the sprites instead of "I+5" in line 160.

I decided to put most of the values into variables at the beginning of the program (lines 190-200 are for the notes, line 210 for duration and volume). Each time Tex encounters a numeric value, it first puts it into a memory address before using it; using variables does it once only, and the program runs a bit faster and smoother.

The names for the note variables are suggested by Raymond J. Herold in the book "TI-99/4A Sound and Graphics" (Compute!), and follow the names of the notes on a piano: C to B for middle range, LC to LB and LLA-LLB for lower ranges; upper ranges are of course HC to HB, then HHC to... whatever you want. The "S", as in "LFS", means the note is a sharp; a flat would be "F", as in "BF", but we don't use any here. The variable "DUM" stands for DUMMY in line 890. If you look at the listing, you will find some note frequencies as "40000", which I did not put into variables for aesthetic reasons (to better visualize the score): they are "mute" notes (inaudible), and serve to stop the previous CALL SOUND; they correspond to the ends of the "slurs" in the musical score, and perform a short pause in the music.

Putting the duration of the quarter note into a variable "T" makes it much easier to control the tempo; the eighth note is T/2, the sixteenth T/4. At the end of the piece, the score had some "slur 3" ("triolets" in French, i.e. three notes in the same time as the basic quarter note), so we have T/3. To make the tempo slower, change T to a higher value, and do the opposite to make it faster. I think 600 is very appropriate to lift our spirits out of the winter blues. TD is for the dotted eighths, equivalent to three sixteenth notes; I made it into a variable because of the long calculation, for fear of slowing down the music.

I did not play too much with the volumes: all the Treble clef, (G), i.e. what is played with the right hand, is loud (V1=1), while the Bass (F) clef, played with the left hand, is a bit softer (V2=5). You can of course play with those values to add some "feeling" to Tex's playing.

Before leaving the score itself, a note on line 890: what does it do? Well, it plays a note which has a frequency lower than 110; in this case, the G just below it, with a frequency of 98. I got the tip (and the values) from a little tutorial program by Tom Moran, "Teach Music"; I quote the appropriate screen from this program: "These low notes are created by using noise # -4 in combination with specific frequencies. Here is the program list:

```
100 DATA 1475,1293,1227,1105,990,957,840,735 110 FOR I=1 TO 8 :: READ T :: CALL
SOUND(1000,T,30, T,30,T,30,T,30,-4,1) :: NEXT I "
```

Now, don't ask me what these values mean! The program plays down the scale below the lowest A (110). I was able to change the first value with the note and volume I needed in Treble clef, with no audible effect.

You should know about line 220. Lines 230 to 260 start some very crude animation: three crocuses grow slowly. In line 230, we define three characters (in three sets, to get three colors) as the same flower; char. 120 is the green stem. The three flowers are then "displayed" in line 260, a much faster way than with CALL HCHARs.

To simulate the growing, we use a "mask" sprite, character 128, defined as a solid block. We need three sprites, numbered from 13 to 15, because the title uses a maximum of 12 sprites. Being the same color as the screen (4), they are "invisible", but all there; being on top of the crocuses, they hide them until put in motion during the playing of the music - a velocity of one, the lowest possible, seemed most appropriate for nature's rhythm. I put the CALL MOTIONs following the "mute" notes to simplify, but you are welcome to change their location. Now, you may ask, how can a single character sprite hide flowers made of two characters vertically placed? Remember, in line 120, we magnify all the sprites to "2", which makes them 4x4, so our flowers are conveniently covered.

At the end of this musical masterpiece, you are asked if you want to play it again; note that even if we display a line without a SIZE statement, the letters in the title are not erased: this is because the sprites are on top of the text screen, and are thus not affected by this statement. The three "invisible" sprites are deleted, so that they don't come down far enough to "hide" the displayed text and our full-grown flowers while Tex waits for you to press a key.

SPRING BONUS!

Did you know that you can ACCEPT strings that are longer than the screen width (28 char.)? Easy, and they can be as long as 255 characters, while INPUT/LINPUT will take only 138 to 140. The only problem is this: it is always on line 24, but you MUST NOT specify it. However, you can qualify it as usual with BEEP, VALIDATE, etc. Try the following, and enter a text until Tex tells you to stop:

```
100 PRINT "INPUT:" :: INPUT A$ :: PRINT LEN(A$)
110 PRINT
120 PRINT "ACCEPT:" :: ACCEPT B$ :: PRINT LEN(B$)
```

TI BASIC continued from February
by **Steven Shaw**

IF...THEN...ELSE

TI BASIC may appear to be slightly limited in its use of IF...THEN compared to some other computers. TI do however allow the ELSE alternative.
The problem arises because TI insist that you use the construction only to transfer to another line. You cannot add commands such as:
IF X=B THEN B=C
to do this you need Extended Basic.

However, TI BASIC does have 'relational operators' which will often help you out of this problem.
The instruction IF X=1 THEN 100 is acted upon by the computer only if the expression (x=1) is TRUE.
A TRUE expression is treated by the computer as having a value of -1, while a FALSE expression is treated as having a value of 0.

The IF..THEN structure does not require an expression to evaluate to 0 or -1 however, and you may use a variable on its own to perform a line transfer:

IF X<>0 THEN 100 will transfer to line 100 if the variable X has any value.

IF X THEN 100 will have exactly the same effect but use less memory.

It is possible in TI BASIC to build up a set of expressions in an IF...THEN line, which will simulate OR and AND, and if you are careful, you may go well beyond OR and AND.

Each expression to be evaluated MUST appear in brackets.

For example:
IF (A=1)+(B=10) THEN 100
If both A=1 and B=10, then the sum of the two expressions is -2 (-1 plus -1), and as the result is not zero, the transfer to line 100 will take place.
If only A=1, but B=5, then the sum will be (-1+0) or -1, and the transfer will still take place.
If A=3 and B=5, then the sum is (0+0) or 0, and the transfer to line 100 will not occur.

What we have then is a way of saying
IF A=1 OR B=10 THEN 100 (don't type this line in).

Using a different mathematical operation, the multiply or *:
IF (A=3)*(B=2) THEN 100
When A=3 and B=2, the calculation is (-1 * -1) or +1.
The result is none zero and the line transfer takes place.
When A=3 and B=1, the calculation is (-1 * 0) or 0
The result is zero so no transfer will occur.
What we now have is a way of saying,in TI Basic:
IF A=3 AND B=2 THEN 100.

Provided you are careful to always know the possible results of the various expressions and mathematical operations, you may build up some very powerful IF...THEN commands, which will save you many lines of tedious programming.

This however is not all you can do with relational expressions.
How about trying to program IF A=5 THEN B=6 ELSE B=0 in TI Basic?
B=-6*(A=5) has exactly this result. If (A=5) then the calculation is B=-6*-1, or B=6. If A does not equal five, the calculation is B=-6*0, or B=0.

This is fairly advanced programming, but if you need this sort of power, it is there for you to use. All you need is to keep track of the possible values of the variables you use.

To whet your appetite: IF (X=1)+(Y=1)+(Z=1)=-2 THEN 100
This interesting group of expressions will transfer to line 100 if any two of the three bracketed expressions is true. What this fairly short line says to the computer is:
If any two of X Y & Z are equal to one then...

HOTLINE NUMBERS

The executive has expressed a desire to assist all members should you have some problems or questions, want to do some library swapping or borrow a book. This will be the place to look. Listed here are the members of the executive, committee heads, and others in the group willing to help in their specialized areas. Of course, if you wish to be placed on the list, just give me a call. I know there is a lot of expertise within our Group, so I hope to add to this list. Please respect normal hours unless you specifically know that someone doesn't mind a call at 3am, or use the BBS to leave a message at 738-0617, 24 hours a day, 7 days a week.

MICHAEL TAYLOR....PRESIDENT.....831-0143

JANE LAFLAMME.....VICE-PRESIDENT.....(H) 837-1719 or (W) 745-2225

PETER ARPIN.....TREASURER AND SYSOP.....523-0017

JOHN O'CONNOR.....SECRETARY.....833-2626

BILL SPONCHIA.....PAST PRESIDENT, SOFTWARE CONTEST.....523-0878

RUTH O'NEILL.....NEWSLETTER EDITOR.....234-8050

TONY HOPKINS.....ADVERTISING.....746-4463

STEVEN BRIDGETT...LIBRARY CHAIRMAN.....521-3631

HENRI MONAT.....ARCHIVES.....824-0941

LUCIE DORAIS.....MEMBERSHIPS.....232-0393

ART GREEN.....ASSEMBLY HELP.....837-1955

DICK PICHÉ.....TECH.....521-8667

CLUB BBS.....SET MODEM TO 8N1.....738-0617

FROM

P.O. BOX 2144,STATION D,OTTAWA
*** ONTARIO,CANADA K1P 5W3 ***