(065) 8802 Ottawa

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

VOLUME 7 NUMBER 02.....FEBRUARY 1988

DON'T FORGET THE MEETING -- FEBRUARY 2, 1988

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

COMING EVENTS

February Meeting:	February 2, 1988 7:30 p.m.	Merivale High School
How To Run FastTerm:	February 10, 1988 7:30 p.m.	34 McLeod Street Contact Bill Sponchia for further information.
Beginners Assembly:	February 17, 1988 7:30 p.m.	Dick Piché's home Contact Bill Sponchia for further information.
TI FEST:	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.
Newsletter Deadline:	February 15, 1988	Important this time, because of the FEST.

EDITOR'S NOTES
from
Ruth O'Neill

This month, Lucie continues the saga of the frog-couple as they make plans for the Canadian TI-FEST. Those involved in FEST preparations are getting busier by the moment, and still need help. It looks as if we will have a great show this year, but you can help make it better still.

Have you renewed your membership yet? If not, rush those cheques off to Lucie so you don't miss any newsletters, because this is the last one that will be sent out on last year's fees. The executive decided, hoping to bring in more renewals, to keep the fees this year at $20.00, even though the club spends considerably more than that per member on the newsletter alone. Then of course you have access to the club's excellent, lively BBS, the extensive library of software, the archives, and all of the workshops and other such activities. Unfortunately, it is only by maintaining a large membership that we can afford to offer all of this to you at the current level. We need you!!

The Ottawa TI99/4A Users' Group would like all of those interested to know that the role of NUAC is now being fulfilled by the club. Keeping the two entities separate created too many administrative complications, and seemed to be unnecessary, given that many of the people involved in one were involved in the other. We plan to continue to serve as a link between the many clubs throughout the country, helping to support a strong national TI community.

One item that should concern all of us is the fate of Picasso Publisher, a commercial program from Australia that was, unfortunately, incorrectly passed out by a user as shareware. It spread rapidly, and now many copies are in use with little compensation to the author of the program. I learned of this through one of Jim Peterson's columns in the L.A. 99er newsletter, and thought that the information should be passed on. Please, if you know of someone or a club who has it in their library under the mistaken impression that it may be distributed freely, please let them know, so that they can either destroy their copy or purchase the program.

From time to time, we receive a cheque for DM1000 made out to Bruce Caron, who is no longer in Ottawa. Since the club is in charge of the administration of the program, all cheques or money orders for DM1000 should be made out to The Ottawa TI99/4A Users' Group.

The February meeting will feature a demonstration of various modifications to the PEB by Dick Piché, as well as a "How to Run" workshop on VERMENU.

BROWSING THE LIBRARY
--with STEPHEN BRIDGETT

Few things in life work as quickly as the 'rumour mill', however inaccurately. On that note, it seems appropriate that I dispel all rumours with respect to the library and me. In the near future, the library will pass from my hands to Dave Morrison's. This will happen when Dave feels comfortable with the operation and will see a co-operative effort for some time; at least up to and through the Fest. The library demands a lot of attention which I do not feel, as a result of career aspirations, that I can continue to give. With Dave's permission I should like to continue to assist with the library in the position of correspondent. This is not a situation where I am abandoning TEX, or at all disgruntled. After the Fest I shall have had a 2 year 'kick at the can'. Enough said!

Again on correspondence, for personal reasons Sandra Touchette has had to put aside that job. Sandra remains a keen TI'er and wants to become active within the club in the future.

Several opportunities to associate with out-of-town members have provided a feeling for the tremendous camaraderie and loyalty amongst TI'ers. If your letter has not yet been answered, don't give up. New software has come in from far afield. It's amazing not only what IS in the library, but also what isn't. If you live a hundred or more miles from Ottawa, I'll bet you have a program which needs to be placed in the library. Why not write us? No one comes out short when they deal with and contribute to the success of the library.

Both the telephone survey and the documentation program are on track. The survey is expected to be completed with data base correlation by the Spring. It has proven to be a big job, but I'm sure the club and especially the executive will benefit from the information. The docs program needs a bit more participation. What do you say folks?

Henri Monat has concluded the first edition of the Newsletter archives. The disk is available through the library and tabulates each article ever written in the club's newsletter which is of an informative nature. Such things as these library articles or the Chairman's articles, which are of a 'chatty' nature, are of course not included. The data base used is PRBASE, also available through the library. There has been a wealth of info published in our newsletter and you may find the archives a valuable resource.

The disk of The month in Feb. will be Sorgan. This disk was not distributed in Jan. For details of this excellent music program see your Jan. newsletter.

The Fest needs you! ....The Library needs you!

The executive depends on a financially successful Fest to balance the books. Let's all circle March 5 and lend a hand to Jane to make it work. A little bit goes a long way - why, just look how far we've come already with the computer that nobody wanted!

Please contact Jane at: 613-837-1719 or 613-745-2225

See you in February.

"How to Run...."
by Bill Sponchia

Well, we had our first seminar - PR BASE - and if it is any indication of what's to come, the trouble of doing this will be well worth it. My thanks go out to Henri Monat not only for his instruction but also for the use of his house. In actual tact, it was so good that we decided to extend the seminar to a second day; the date is not yet set.

We have two more sessions coming up. The first will be held at the General Meeting on February 2nd and will be on the installation and use of VerMenu 7.1. The instructor will be Steve Bridgett.

The second one will be a double-billing because it will feature both FastTerm and TEXLINK (Ottawa's own BBS). Please note the following information:

Instructor: Charles Earl
Date: Wednesday, February 10th
Time: 7:30 PM
Location: 34 McLeod Street, Ottawa

Let Charles or myself know if you are planning to attend.

For those who have given me additional suggestions please, hold tight, and I'll try to do the best I can.

As a secondary output of these seminars I am attempting to put together a document on each one held. This will be as close to duplicating the seminar as possible and is intended for people unable to attend the actual seminar; however, it might also be a good reminder for those who did get to go.

"Questions & Answers"
by Bill Sponchia

Often enquiries about specific aspects of a program give indications of a more underlying problem. When this seems to be the case, I will attempt to not only answer the specific question, but also try to rectify the deeper problem.

A letter was received from a user in Texas who had specific problems concerning Multiplan. A personal reply was sent off answering these questions, but I also felt that there was a general indication of underlying problems or misconceptions. I have taken the liberty of both restating his questions in a more general nature and also of including questions to address the other problems.

Q1 - Should Multiplan be used whenever possible?

A1 - No. Multiplan is a spreadsheet, which by definition is useful for manipulation of numbers. Therefore, when numeric information must be manipulated the spreadsheet is one useful tool. However, if you are considering using Multiplan as a text holder (no calculations required) because it comes in ready-made columns (or any other similar reason) then I think you should reconsider. There is much better software to use (the most obvious is the word processor).

Q2 - With Multiplan, can numbers and letters be entered into the same cell in such a way as to allow for the mathematical manipulation of the numbers?

A2 - Sorry, but the answer is No again. A cell can only contain either "text" or "values", but not both. "Values" of course can be mathematically manipulated and must be numeric (or formulae which give numeric results.) "Text", on the other hand, can be any characters (letters, digits, comma, period, etc) and they cannot be mathematically manipulated. A digit character input as "text" is recognized only as "text" and not as a "value".

Q3 - How can Multiplan be sped up?

A3 - There are two way. The first is to be sure to turn the "recalculation" feature off. This is done by going to the "Options" and setting "Recalc" to "No". This should be done not only when inputting data to the sheet but also when you are assembling the sheet (eg - setting up the formulae, etc.) When all the data is entered, then you can get the sheet recalculated by pressing "FCTN 8".
WARNING - When told to save the sheet, if it has not been recalculated the computer will automatically do so before saving; however, when told to print the sheet, the spreadsheet as shown in memory will be printed. If the sheet has not been recalculated beforehand, the printed spreadsheet will be wrong. A good habit to get into is always doing a recalculation (FCTN 8) before printing.
The second way to increase the speed is to keep the spreadsheet as small as possible. This doesn't mean that you should fragment a spreadsheet thoughtlessly - some just have to be large - but it does mean that consideration should be given to dividing up large sheets. For example, if you have one large sheet showing "Revenues" and "Expenses", then you should consider making two small sheets: one for Revenues and one for Expenses. The extra time spent setting up two sheets instead of one will be more than offset by the savings in recalculation time.

Q4 - Can printer command codes (double-strike, line-spacing, etc) be sent to the printer from Multiplan?

A4 - After rereading the manual (more carefully than I had done before) I could find no way to send the codes. The few times that I can remember needing this feature, I remember that I set up the printer with a small BASIC program before going to Multiplan. I have since learned (heard) that there is a possible way, but unfortunately I didn't have time to verify the fact or even how to do it. I'll find out more this month and report on this next time. That's all for this time around. Thanks - see you. PS to Dave Caron - got your message. I'm enquiring about it because, needless to say, I don't have a clue what you are talking about, but thanks anyway, hope I can be of help.

GENEVE UPDATE (As at Jan.17/87)
by Jack Adams

So far the new year has brought Geneve users the following things :

1. MDOS Version 1.0 (already distributed prior to Christmas)
2. MDOS Version 1.1 (not yet assessed in Ottawa)
3. MY-WORD Version 1.1 (being used at this moment)
4. VIDEO CHESS fix (assessment not available)
5. MULTIPLAN final upgrade in files MPDATA and MPINTR
6. MYARC DMIII Version 2.1 (not yet assessed)
7. MYARC EXTENDED BASIC II Version 2.11
8. MYARC GPL INTERPRETER Version 0.98
9. MYARC MY-ART Version 1.0 (to be featured in next issue)

A few pointers for items 1. to 8. :

1. The commands CONFIGSYS and MODE have been dropped. The following commands have been modified: COPY, FORMAT, PROMPT, RAMDISK, SPEED, TIMODE, Scrolling the Command Stack, Creating a Batch File.

3. MY-WORD can be loaded either from option 5 of E/A program via the GPL interpreter using DSK(n).MW or from GENUTIL6 via GPL. In both cases, speed 3 is used. If you use GENUTIL6 you must rename the MW file to MYWORD. If you have loaded MYWORD into your Horizon ramdisk, be careful to name this disk MYWORD and during the loading process, remove any diskettes in your other drives if they are also named MYWORD and DO NOT contain any of the actual word processor files. Note that the lights of your real drives will momentarily come on while the word processor loader is searching for the disk MYWORD. It will make 2 passes before loading is complete. This version of MYWORD appears to be free of all bugs.

5. MULTIPLAN has been upgraded so that the program no longer has to go back to the disk drive containing the program (DRIVE 1) in order handle the various commands. Once the multiplan operating files have been loaded, all commands are handled directly from the memory in the computer. THERE IS ONLY ONE EXCEPTION: the HELP file must reside in DSK1 owing to the fact that this file takes up too much memory.

6. MYARC DMIII Version 2.1 will load using the GPL interpreter (use speed 5) and Editor/Assembler option 3 using the command DSK(n).DM. It may be used to manipulate all RAMDISK and floppy drives while in GPL mode, and is capable of one-pass copying of up to a 96K file and/or diskette. 18 sectors is the default setting in the formatting mode. REQUIRES MYARC DISK CONTROLLER CARD FOR OPERATION.

7. MYARC EXTENDED BASIC II Version 2.11 will run at speed 5 of the GPL. It provides for 32, 40, and 80 columns as well as many new graphics commands. To load, use option 5 of the Editor/Assembler program to call DSK1.BASIC. It will load and run most standard TI-Basic and Extended Basic programs. Be careful to call for TIMODE before loading GPL in order to allot adequate space for program operation.

SUPER EXTENDED BASIC IS HERE!
by Lucie Dorais

I got my new cartridge last night (from Triton, via Bob Boone), and I cannot resist showing you a little bit of what you can do with it. It is not a great effort, but the deadline for articles was two days ago...

This is not a review; MICROpendium did a superb one in its Sept. 1987 issue, and I refer you to it. Here is a short MENU program in XB and in SXB, so that you can compare both. The new CALLs I use are only a few (and not the most exciting ones) of all the goodies packed in by Craig Miller and his team.

XB

```
100 CALL CLEAR :: CALL SCREEN(
5) :: FOR X=1 TO 8 :: CALL COL
OR(X,16,5) :: NEXT X
110 DISPLAY AT(3,13):"MENU"
120 FOR X=8 TO 14 STEP 2 :: DI
SPLAY AT(X,8):STR$(X/2-3) :: N
EXT X
130 DISPLAY AT(17,8):"Q- RETUR
N TO XB"
140 CALL KEY(0,K,S) :: IF S=0
THEN 140 ELSE IF K=81 THEN END
145 IF K<49 OR K>52 THEN 140
150 LINE=2*(K-48)+6 :: CALL HC
HAR(LINE,8,42)
160 CALL SCREEN(8) :: FOR X=1
TO 8 :: CALL COLOR(X,2,8) :: N
EXT X
170 ON K-48 GOTO 180,190,200,2
10
180 RUN "DSK1.PGM1"
190 RUN "DSK1.PGM2"
200 RUN "DSK1.PGM3"
210 RUN "DSK1.PGM4"
```

SXB

```
100 CALL CLEAR :: CALL SCREEN(
5) :: CALL COLORS(16,5)
110 DISPLAY AT(3,13):"MENU"
120 FOR X=8 TO 14 STEP 2 :: DI
SPLAY AT(X,8):STR$(X/2-3)&"- P
ROGRAM "&STR$(X-2-3) :: NEXT X
130 DISPLAY AT(17,8):"Q- RETUR
N TO XB"
140 CALL KEYS("1234Q",P) :: IF
P=5 THEN END
150 ADR=32*(2*P+5)+7 :: CALL P
OKEV(ADR,138)
160 CALL SCREEN(8) :: CALL COL
ORS(2,8)
170 CALL RUNPROG("DSK1.PRG"&ST
R$(P))
```

Notes on some lines above:

140: P reads the position of the key in the string
150: ADR is the address to poke to; value is ASCII plus the XB bias of 96

Just to whet your appetite, here is a list of the new CALLS not used above: ALL (fills screen with one char.), BEEP, BYE, CAT("DSKx"), CHIMES, CLOCK (+CLKOFF), CLSALL (close all files), GOSPRT (+STSPRT: coordinates sprite movement), HONK, NEW, PEEKG + POKEG (to read GRAM and GROM), PEEKV, QUITOFF (+QUITON), SCROFF (+SCRON, to erase or display a whole screen).

Some CALLs are old statements but with a new twist: you can now use variables for line numbers with CALL RESTORE, GOSUB, GOTO. You can also detect whether or not the SHIFT, FCTN and CTRL keys have been pressed. And last but not least, with CALL DRAWNPLOT you have access to a whole range of high resolution graphic commands!

If that is not enough to make you desperately want this new module (currently avaiable for $85.00 CDN), let me tell you that it adds new editing features to control the cursor; you can also COPY, DEL or MOVE program lines, and the RES is much more flexible. A dream! And you can TRACE to the printer, and control the line width of your LISTs, like I did to get the above listings.

Texlink is EVEN Teaching
by Henri Monat

This article deals with many directives of assembly language, but mainly with the EVEN directive.

It all began on a Friday night. Reading messages in the "Programmers' Corner of our BBS (Texlink), I came across a message from Steve Bridgett. He had uploaded the program listed at the end of this article, and was asking for help in debugging it. The next message was from Charles Earl to Steve, stating that he had already fixed the bug by changing the cursor subroutine to one of his own.

For learning purposes, I wanted to know if there was a possibility of fixing that bug by keeping the same cursor subroutine. After 2 hours of hard work (no RamDisk), I found out that the program was working properly by moving the workspace instruction "WSREG BSS >20" right after the Ref/Def table, before the label PROMPT. Having found the bug (as I thought it was), I also noticed that the TEXT directive was 15 characters long and not 16 (see line LI R2 16) but noted that it had nothing to do with the bug problem. What is now past history proved that I was wrong with both of these assertions. I provocatively revealed on Texlink my findings to Steve; 24 hours later, there were about 30 messages on the BBS just on that point. The following is a summary of the discussion.

STEVE BRIDGETT 9:45 P.M.:

HA HA Excellent, it was a problem that I was sure you could solve for me. I have yet to download it but bet it concerns C **ANYBYTE, STATUS.** After Charles helped me, I looked at his solution and although his cursor routine was different than ours, the test he made was MOV **STATUS,STATUS.** I made this change and it worked. But, was sure that you had used the book's answer and with no problems. It drove me nuts.

STEVE BRIDGETT 10:30 P.M.:

I don't understand. Where does it say in the book that the workspace must be right after the Ref/Def table? In fact, the workspace in the book stands in the same place where they are in my program. There must be another answer.

STEVE BRIDGETT 11:00 P.M.:

Yes henri, I've figured it out. Since the prompt has 15 letters i.e. an odd number, therefore the assembler requires an EVEN directive. This forces the PC back to an even word boundary, i.e. the PC works on even bytes. Try moving the wsreg back after "NAME" prompt, but add the directive EVEN after "prompt". That will do it.

LLOYD GALENZOSKI:

I've added a few things to the STEVE PRGM. Have a look and see what you think. Playing with a program like this will be very valuable in teaching you how things can be put together in a variety of ways. Note that I've used a data statement to define the space for the NAME, instead of BSS. You could also use TEXT. The DATA statement initializes the space to zero each time the program is run. (Author's note: basically, LLoyd changed "NAME BS 30" FOR "NAME DATA 0,0,0, etc..., each 0 representing 2 bytes or 1 character.)

STEVE BRIDGETT:

Thanks LLoyd, this program is indeed a valuable learning tool. About the WSREG, I understood that it could be placed anywhere, then figured out that the directive EVEN after the odd length text directive was the problem, and not the location of the WSREG at all. When I placed EVEN after the TEXT, all was well. I assume that the reason the pgm worked by moving the WSREG up to the top was pure coincidence. Am I on track here?

CHARLES EARL:

Actually, I used MOVB, not MOV **STATUS,STATUS**. TEXT directives can also be put at the end of the program, just before the END directive.

STEVE BRIDGETT:

"... at the end..." makes sense, that way there is no chance of the program trying to execute the bytes as code or the PC getting out of boundary. Am I thinking straight?

LLOYD GALENZOSKI:

You're right - the moving of WSREG was probably a coincidence. Don't rely on that. As Charles said, it's a good idea to learn where to use the EVEN directive. After TEXT and after BYTE is always a good habit. You don't always need it, but when you least expect trouble is when you'll run into it.

CHARLES EARL:

Not placing an even directive after your text will cause problems with the code. The DATA directive does the same thing as a EVEN directive - it will set the address even before defining the data. The BSS, TEXT and BYTE will not change the address from odd to even. It is advisable to arrange the data so that the data block ends on an even address. Keep your BSS and DATA directives together and keep your TEXT and BYTE directives together.

LLOYD GALENZOSKI:

When using MOV **STATUS,STATUS**, or MOVB **STATUS,STATUS** the only difference I know of in the object code is that the former is C820837C837C and the latter is D820837C837C. The C820 being the MOV and the D820 the MOVB. If I'm out to lunch, Charles or Tom will surely point out any further differences.

CHARLES EARL:

You can't rely on the second byte of the STATUS register.

LLOYD GALENZOSKI:

I thought that CLR **STATUS** would look after the second byte. But, I agree that there may be circumstances where that may not work. If I understand, it is recommended to do a MOVB **STATUS,STATUS** to avoid problems.

TOM BENTLEY:

The big difference with MOV and MOVB is the number of bits that are being operated on. MOV moves 16 bits at a time and expects to reference its objects on a word boundary (even), while MOVB looks at 8 bits at a time with no boundary restriction. If you are referencing a register with MOVB then the 8 bits referenced will be the most significant 8 bits!!!

HENRI MONAT:

Do you put the EVEN directive at the end of each TEXT directive, or at the end of each labelled TEXT directive, or at the end of all TEXT directives?

LLOYD GALENZOSKI:

I keep my TEXT directives together and place the EVEN directive at the end. It stands alone in the second position, i.e. the first tab from the left margin.

TOM BENTLEY:

The reason that MOVB is used instead of MOV when looking at STATUS is that this is the GPL status BYTE and therefore you should only look at 8 bits of information. The other byte can have any value.

****

The precdeding gives, I hope, a clear picture of the discussions that took place. Most of all, it gives a good example of the kind of spirit and life there is on Texlink. The program listing follows:

```
          DEF       START,CURSOR
          REF       VSBW,VMBW,VMBR,KSCAN

PROMPT    TEXT  'ENTER FULL NAME'
KEYADR    EQU   >8374
KEYVAL    EQU   >8375
STATUS    EQU   >837C
ENTERV    BYTE  >0D
LEFTV     BYTE  >08
RITEV     BYTE  >09
ANYKEY    BYTE  >20
WSREG     BSS   >20
NAME      BSS   30

* ACCEPT DATA FROM KEYBOARD

START     LWPI  WSREG
          LI    R0,256          PUT UP PROMPT AT
          LI    R1,PROMPT       ROW9,COL 1
          LI    R2,16           NO OF CHAR OF PROMPT
          BLWP  VMBW
          LI    R0,288          ROW 10, COL 1
          LI    R10,30          LENGTH OF DATA
          BL    CURSOR         GET THE DATA

* READ DATA FROM VDP

          LI    R0,288          LOAD R0 WITH VDP RAM ADRS
          LI    R1,NAME         LOAD R1 WITH THE CPU RAM ADRS
          LI    R2,30           LOAD R2 WITH DATA LENGTH
          BLWP  VMBR           READ DATA FROM VDP

* WRITE DATA BACK TO BOTTOM LINE OF SCREEN

          LI    R0,734          LOAD R0 ADRS 1ST POSN LAST LINE
          LI    R1,NAME         LOAD R1 WITH ADRS OF DATA
          LI    R2,30           LOAD R2 WITH LENGTH OF DATA
          BLWP  VMBW

* ENABLE QUIT

          LIMI  2
          LIMI  0
          B     START           REDO


*       CURSOR SUBROUTINE

* AUTHOR'S NOTE: REFERENCE IS BEING MADE TO THE CURSOR SUBROUTINE
* DESCRIBED IN THE EXCELLENT BOOK OF RALPH MOLESWORTH "INTRODUCTION
* TO ASSEMBLY LANGUAGE FOR THE TI HOME COMPUTER", STEVE DAVIS
* PUBLISHING, PAGE 63.


          END   START
```

Trading Post

TI 99/4A FOR SALE

- PEB WITH INTERFACE CCA AND INTERFACE CABLE =200.00
- TI DISK CONTROLLER, SSSD FULL HT DISK WITH TI DISK MANAGER MODULE =100.00
- HORIZON RAM DISK (192) WITH EPROM ROS =180.00
- SPEECH SYNTHESIZE = 30.00
- 6 MODULE WIDGET BOX WITH RESET BUTTON = 20.00
- 2 WICO JOYSTICKS EA= 10.00
- CONSOLE BASE WITH ATARI JOYSTICK ADAPTER = 10.00
- 32K MEMORY FOR PEB = 75.00
- RS232 FOR PEB = 75.00
- SHUGART 455 1/2 HEIGHT DDDD DISK DRIVE =100.00 (WITH TI DISK MANAGER)
- SPARE KEYBOARD = 5.00
- SPARE INTERNAL POWER SUPPLY = 2.00
- TI 99/4A CONSOLE WITH RF MOD & POWER SUPPLY = 30.00
- MULTIPLAN = 30.00
- TI EXTENDED BASIC = 50.00
- EDITOR/ASSEMBLER = 30.00
- ASSORTED CHILDREN'S MODULES = ASK
- LOGO I = 20.00
- ADVENTURE MODULE WITH 5 DISKS = 15.00
- CHISHOLM TRAIL = 3.00
- VIDEO CHESS MODULE = 8.00
- CASSETTE CABLE = 5.00
- MANY ASSEMBLER GAMES ON DISK PER DISK = 3.00

BUY IT ALL OR BY THE PIECE.

!!CALL RALPH ROMANS
257-1765 (h) 596-7370 (w)

EXPANSION PORT INTERFACING: Part 3.

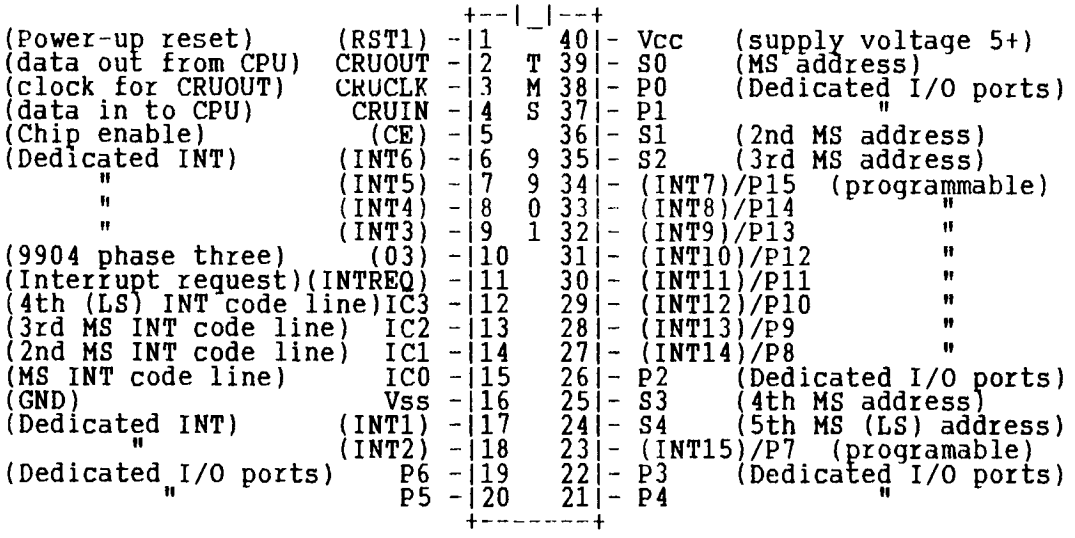
by David Caron

You may have noticed that I skipped a month. The delay was due to the difficulty in obtaining specifications for the: TMS 9901 Programmable Systems Interface, which you will see is an excellent chip for interfacing the 99/4A with the outside world.

Specific features of this chip are:
-6 Dedicated Interrupt Input lines. (can also be used as inputs)
-7 Dedicated I/O ports
-9 Ports Programmable as Interrupts.
-A programmable clock that can be set to countdown starting at a certain number (0 to 16383) and will "ring" at 0 .

Following is a pinout of the 40-pin 9901 chip and a table on how the software relates to the hardware pins. In the next article, I will go into detail about how it works and how to program it from assembly (Basic or Extended basic have no routines to directly access the cru lines).

BRACKETS () INDICATE THAT THE SIGNAL IS ACTIVE LOW



REGISTER BIT-MAP

CRU BIT address (S0-S5)	INT code lines (IC0-IC3)	CRU read data interrupt/clock mode	CRU write data interrupt/clock mode	Binary state and notes
0	DON'T CARE	CONTROL BIT	CONTROL BIT	-(0)Interrupt
1	1	(INT1)/CLK1	MASK 1/CLK1	\(1)Clock
2	2	(INT2)/CLK2	MASK 2/CLK2	
3	3	(INT3)/CLK3	MASK 3/CLK3	MASK
4	4	(INT4)/CLK4	MASK 4/CLK4	(0) disable
5	5	(INT5)/CLK5	MASK 5/CLK5	interrupt
6	6	(INT6)/CLK6	MASK 6/CLK6	(1) enable
7	7	(INT7)/CLK7	MASK 7/CLK7	interrupt
8	8	(INT8)/CLK8	MASK 8/CLK8	
9	9	(INT9)/CLK9	MASK 9/CLK9	CLK1 to CLK14
10	10	(INT10)/CLK10	MASK 10/CLK10	are registers
11	11	(INT11)/CLK11	MASK 11/CLK11	that indicate
12	12	(INT12)/CLK12	MASK 12/CLK12	the countdown
13	13	(INT13)/CLK13	MASK 13/CLK13	number.
14	14	(INT14)/CLK14	MASK 14/CLK14	
15	15	(INT15/INTREQ)	MASK 15/(RST2)	-(0) software
16	16	P0 (input)	P0 (output)	| reset
17	16	P1 (input)	P1 (output)	
18	16	P2 (input)	P2 (output)	P0-P15 are
19	16	P3 (input)	P3 (output)	I/O data
20	16	P4 (input)	P4 (output)	registers.
21	16	P5 (input)	P5 (output)	
22	16	P6 (input)	P6 (output)	They can be
23	16	P7 (input)	P7 (output)	made outputs
24	16	P8 (input)	P8 (output)	by simply
25	16	P9 (input)	P9 (output)	writing to
26	16	P10 (input)	P10 (output)	them and
27	16	P11 (input)	P11 (output)	inputs by
28	16	P12 (input)	P12 (output)	simply
29	16	P13 (input)	P13 (output)	reading from
30	16	P14 (input)	P14 (output)	them.
31	16	P15 (input)	P15 (output)	

FAST EXTENDED BASIC

LUCIE DORAIS

There are many labeler programs around; so why another one? For those times when you quickly need just a no-nonsense label. So I wrote QUICKLABEL, that also prints ENVELOPES; and, unlike any other, it is BILINGUAL!

```
100 REM ** QUICKLABEL ** Lucie Dorais ** Nov 86/Jan 88
110 REM == init ==
120 DIM L$(5):: CALL CLEAR :: CALL SCREEN(14):: FOR X=0 TO 14 ::
    CALL COLOR(X,16,14):: NEXT X
130 CALL CHARPAT(64,A$,92,B$,123,C$,125,D$):: CALL CHAR(137,A$,138,
    B$,139,C$,140,D$)
140 N,INFO=1 :: ML$="*** LABELS ***" :: ME$="** ENVELOPES **" ::
    MS$=ML$ :: SL$="" :: SE$=RPT$(" ",31):: S$=SL$ ::
    EF$=RPT$(" ",140)
150 GOTO 160 :: ENV,K,S,Y :: CALL KEY :: CALL HCHAR :: CALL VCHAR
    :: !@P-
160 CALL CHAR(128,"00FFFF0000000000000000000000FFFF006060606060606060
    6060606060606060")! borders
170 CALL CHAR(132,"003F7F606060606000FCFE0606060606060606060606FEFC00
    606060606073F00")! corners
180 CALL CHAR(136,"000010207E2010",142,"82BAA2B2A2BA82FE") !bck/exp
190 CALL CHAR(64,"00201038447C4444",92,"0000003C40403C08",123,"0008
    107C4078407C",125,"0020107C4078407C")! French Epson
195 REM == draw screen ==
200 CALL HCHAR(5,2,132):: CALL HCHAR(5,3,128,28):: CALL HCHAR(5,31,
    133):: CALL VCHAR(6,31,131,5)
210 CALL HCHAR(11,31,134):: CALL HCHAR(11,3,129,28):: CALL
    HCHAR(11,2,135):: CALL VCHAR(6,2,130,5)
220 CALL HCHAR(1,2,132):: CALL HCHAR(1,3,128,5):: CALL
    HCHAR(1,8,133):: CALL HCHAR(2,8,131)
230 CALL HCHAR(3,8,134):: CALL HCHAR(3,3,129,5):: CALL
    HCHAR(3,2,135):: CALL HCHAR(2,2,130)
240 DISPLAY AT(2,1)SIZE(4):"N= 1" :: DISPLAY AT(23,1):
    RPT$(CHR$(128),28)
245 REM == main menu ==
250 DISPLAY AT(6,1):ER$ ! erase label on screen
260 IF INFO=0 THEN 280 ELSE DISPLAY AT(14,4):" 1 - ENTER DATA"
    :"    2 - D> MY NAME":"    3 - D> OTTAWA TI UG"
270 DISPLAY AT(17,5):"4 - RESET NUMBER":"    5 - Redo LAST DATA"
    :"    6 - Toggle ENV/LAB":"    7 - Quit" :: INFO=0
280 CALL KEY(0,K,S):: IF S=0 OR K<49 OR K>55 THEN 290 ELSE 300
290 DISPLAY AT(2,11):"" :: DISPLAY AT(2,11):MS$ :: GOTO 280 ! flash
300 ON K-48 GOTO 370,310,320,330,340,350,480
305 REM == choices 2-7 ==
310 RESTORE 500 :: GOTO 360
320 RESTORE 510 :: GOTO 360
330 ACCEPT AT(2,4)SIZE(2):N :: GOTO 250
340 FOR X=1 TO 5 :: DISPLAY AT(5+X,1):L$(X):: NEXT X :: GOTO 410
350 IF ENV=0 THEN ENV=1 :: N=1 :: MS$=ME$ :: S$=SE$ :: GOTO 250
    ELSE ENV=0 :: MS$=ML$ :: S$=SL$ :: GOTO 250
360 FOR X=1 TO 5 :: READ L$(X):: DISPLAY AT(5+X,1):L$(X):: NEXT X
    :: GOTO 450
365 REM == enter data ==
370 INFO=1 :: DISPLAY AT(14,1):"   expanded: CTRL N ["&CHR$(142)
    &"]": :"   caract}res fran\ais:":""
380 DISPLAY AT(18,5):" @ = "&CHR$(137)&"     \ = "&CHR$(138)
    "     { = "&CHR$(139)&"     } = "&CHR$(140)
390 DISPLAY AT(20,4):"^ = vowel [CTRL H] ^"
400 FOR X=1 TO 5 :: ACCEPT AT(5+X,1)SIZE(-28):L$(X):: NEXT X
410 DISPLAY AT(24,1)BEEP:"Press PROCEED, REDO or BACK"
420 CALL KEY(0,K,S):: IF S=0 OR(K<>6 AND K<>12 AND K<>15)THEN 420
    ELSE DISPLAY AT(24,1):""
430 IF K=6 THEN 400 ELSE IF K=15 THEN 250
440 REM == print ==
450 OPEN #1:"PIO" :: PRINT #1:CHR$(27)&"R"&CHR$(1);
460 FOR X=1 TO N :: FOR Y=1 TO 5 :: PRINT #1:S$&"    "&L$(Y) ::
    NEXT Y :: PRINT #1:"" :: NEXT X :: PRINT #1:CHR$(27)&"@"
```

```
470 CLOSE #1 :: DISPLAY AT(24,1):"" :: GOTO 250
480 END !@P+
490 REM == data ==
500 DATA MY NAME,4000 My Address,"CITY, Prov.",H0H 0H0,""
510 DATA OTTAWA TI99/4A USERS' GROUP,P.O. BOX 2144,STATION D,
   "OTTAWA, Ont.",K1P 5W3
```

You start in label mode, with the printing counter N set to one. The screen shows an empty label, and a smaller one tells you the value of N. You will also see the main menu (plus 7 - Quit):

1 - ENTER DATA	3 - D> OTTAW UG	5 - Redo LAST DATA
2 - D> MY NAME	4 - RESET NU	6 - Toggle ENV/LAB

ENTER DATA allows you to manually enter the label. There is information about the French characters and how to type them (see below). You can print double-width by preceding your line with CTRL N, redesigned as an "E" in a square; try it for the postal code! You can also enter any other printer code, such as underlining; just remember that ESCAPE is CTRL . (period). Unfortunately, it cannot be shown properly on the screen (see below), but it will be sent to the printer. When you are finished, you may PROCEED to print the label, REDO your data, or go BACK to the main menu.

The two lines preceded by D> are for automatic printing of DATA lines. (You can change them, of course - lines 500-510. Remember that data containing commas need the quotation marks.) RESET NUMBER is for changing the N variable; I kept it completely independent of what you print for more flexibility and easier programming. REDO LAST DATA: if you find a mistake only on the printed label or envelope, there is no need to re-enter all the data. Just press 5, then REDO at the prompt, to make your correction.

If you press 6, you go into envelope mode. It works exactly the same, but the "label" is printed around the middle of the envelope. Just position it accordingly. To get sharper printing, open the back flap before inserting the envelope, and align the point with the centre of the roll. To give you time to change to the next envelope, the counter is automatically reset to one.

With that feature, you can get rid of labels altogether: print your return address (a data line) in the upper left corner in "label mode", then advance your paper and print the address of your mother in "envelope mode". Or use the "Ottawa UG" data to write to the librarian or send in your renewal...

SOME PROGRAMMING INFORMATION:

Line 120 changes the colors. Set 0, undocumented by TI, controls char. 24-31, and therefore the cursor (ASCII 30). In line 130, we CALL CHARPAT the characters that will be redefined for the French characters and we CALL CHAR them to new ones (to be able to show them on the info screen).

Line 140 is our main initialization line. Since N and INFO are both initialized to 1, we can group them in one statement. After redefining the variables for label and envelope mode, we initialize "working variables" MS$ (message) and S$ (print spacing) to the default label mode.

Pre-scan is put off in line 150, after Tex has encountered a bundle of CALLS and variables, then put on again in line 480, after the END and just before the DATA. We then draw the screen; since we need the full 28-character line to enter our data, we cannot DISPLAY the graphics. Only CALL CHAR can use columns 2 and 31 of the screen.

When you type the menu lines (260-270), include all spaces, even the starting ones. Although less "mnemonic" than letters ("Press N to Reset Number"), a number menu is much easier and shorter to code and to change. You can error-check the CALL KEY very easily, and use the ON GOTO statement for branching (line 300). Letter choices would have needed a long list of IFs...

When you use the DATA statements for choices 2 and 3 (branch to lines 310 and 320), you must RESTORE to the right DATA line. Otherwise Tex reads the DATA sequentially, and goes OUT OF DATA very quickly.

Line 350 is a typical toggle: if ENV=0, we are in label mode, so we switch to envelope mode: message and spacing are changed accordingly, and N is reset to

1. If, on the other hand we are in envelope mode (ENV=1), the reverse is done, but we don't reset the counter.

Be very careful when you type lines 370-390, which print the information to get the French characters. In line 420, we scan the keyboard for some functions keys. Their value, different from the digit on the key, is found in the reference card that came with your computer: 6 for REDO, 12 for PROC'D, and 15 for BACK.

PRINTING IN FRENCH AN: THEN SOME:

Most printers speak more than one language. Epson has devised a clever way to do this: when you tell the printer to go into some language other than English, it redesigns some characters in the true ASCII range, i.e. with numbers below 128. Other manufacturers have since adopted that standard as is, e.g. Mannesmann-Tally, or only parts of it, like the Gemini (the TI printer is, of course, a true Epson in disguise).

You get some, you lose some: the @ (ASCII 64) becomes an "à", the \ (92) a "ç", the | (124) a "ù", and the tilde (~, 126) a diaeresis. (Check your dictionary - I cannot print it while in English mode...). No great harm, but you also lose the {}s (123 and 125) to the "é" and the "è"; more importantly, the []s (91 and 93) become some rather useless but cute characters. If you know some French, you will ask where the accents circonflexes (^) are. Better ask Mr. Epson... Maybe because you can use it with all vowels, he thought that was too much! Just type the letter, then a backspace (CTRL H), then the "^". The information on screen will remind you how to do it, and CTRL H will show as a left arrow on the screen.

If your printer is not Epson-compatible, refer to your manual to get the right codes for the characters in lines 130 and 190. (And you will need to change the info screen in lines 370-380.) Please note that I did not include the []s, totally useless, so we can use them both in labels and on the screen. As for the ù, how many times did you see it in an address???

To make Epson a true Parisian (or Québécois!), you must send a special code. My program does it automatically in line 450; CHR$(1) is the code for French (not the digit 1, which is ASCII 49). The printer is reset to defaults before closing (ESC&"@"). Check your printer manual to see what code you need. You could modify the program to ask whether you want to go French or not, then use a toggle variable like FR; if FR=0, go U.S.; if FR=1, go French, then use that value in line 450 as CHR$(FR). You can also have a menu à la TI-WRITER to get different values for each language. Have fun!

While reading your manual, also check the value for expanded (shift-out) printing. Epson uses ASCII 14, and I have redefined its Basic equivalent (142) in line 180. Backspace should be ASCII 8 (Basic 136) for all printers. If you wonder where I got those Basic values, just look at your reference card: the true ASCII numbers are called "Pascal Mode", and their equivalents are "Basic Mode". Why then can we not change the ESC char.? Because its Basic equivalent, 155, is outside the range of CALL CHAR in XB...

Some more control codes are not listed because they don't have a Pascal value. Here is my Valentine. Add them on your card: CTRL , (comma) is ASCII 128, CTRL \ is 187; the CTRL digits are 176 to 183 for CTRL 0-7 (do a quick program to check the CALL KEYS and print the key value... you'll see what I mean).

SPECIAL CORCOMP

Refer to my November column and try to get the color changes in line 120 even faster! And now, a special Valentine just for you: you can redefine char. 155 for ESCAPE by poking its value directly into VDP. Just add those two lines, and CTRL . will show as a big square. The string "00FE82BABABA82FE" is poked byte by byte, in decimal less the 96 XB bias: FE=254-96=158, 82=130-96=34, etc.).

```
191 B$=CHR$(160)&CHR$(158)&CHR$(34)&RPT$(CHR$(90),3)
    &CHR$(34)&CHR$(158)
192 DELETE "LD-CMDS" :: CALL LINK("VPOKE")(2008,96,B$)
```

```
-39  65       ASCII for B
-40   8       LENGTH OF LINE

-41   0       END OF LINE
-42  69       ASCII for E
-43   1       "1 letter follows"
-44 199       "String follows"
-45 184       CONTROL CODE FOR & (concatenation)
-46  36       ASCII for $
-47  68       ASCII for D
-48 190       CONTROL CODE for =
-49  36       ASCII for $
-50  67       ASCII for C
-51  10       LENGTH OF LINE
```

A lot can be learned of the machines operations in this manner: it is possible to see how the program is stored, and also possible to learn the control codes for the BASIC commands.
NOTE: Although you key in REM, only one byte has been used.
If you key in GOTO only one byte is used, but GO TO (with a space in between) uses 2 bytes because two command words are used.
Note that A$ takes up two bytes but that "E" takes up three: one to indicate'string',one to indicate how many letters, and then one for 'E' itself.
Note that '2' similarly occupies 3 bytes but that 'B' only uses one byte.
If program memory is scarce, replacing often used numbers with single letter variables can save a little memory. You will appreciate that the number '12345' will occupy 7 bytes! but a variable set to this value only occupies 1 bytes!
(NB: Each numeric variable used also occupies stack space, and will always use 8 bytes of stack space regardless of the number).
Although not shown here, the CALL routines occupy 1 byte for the word CALL, but for example COLOR takes up 7 bytes, as it is treated as an 'unquoted string' (command code 200). Because command code 200 is used instead of 199, you cannot use CALL A$. (A$ IS A QUOTED STRING).

You may use this procedure to thoroughly investigate the way your computer stores its programs.

The memory locations -52 onwards in this example hold the LINE INDEX. As program lines do not appear in memory in line number order, but rather in the order keyed in, the computer makes use of an index, which IS in line number order.
Each line used occupies 4 bytes for the index:

```
-52  226
-53  255
-54  100
-55    0
```

Locations -54 and -55 give the line number (100) and locations -52 and -53 give the location of that line,slightly coded!
The memory location is 255x256 + 226 (which is 65506)
Although the computer can address 64k, it does so by splitting it into + and - 32k. To convert this large result to a memory location we can use, we subtract 65535:
Location= 65506 - 65535 = -31, which is where the line commences.

As each program line takes up a minimum of 7 bytes:
Index=4, Line end=1, Line length=1, Single command=1.
it makes sense when memory is tight to use as few lines as possible: this is where you can make appropriate use of the facilities of Extended Basic.

Here is a short program to show how you can force a program to write itself.
EXTENDED BASIC with 32k RAM:
Key in in THIS order!:

```
100 GOTO 140
110 PRINT "!!!!!!!!!!!!!"
120 END
130 STOP
140 CALL INIT
150 CALL LOAD(-47,156,199,13,84,69,83,84,32,
    67,79,77,80,76,69,84,69)
160 GOTO 110
170 END
```

After you have keyed this in, LIST it, then RUN it and LIST it again. Note that CALL LOAD has been used to enter several values into memory at one go: the first value goes into -47, the second value to -46, the third to -45 and so on.
Although you can overwrite a program line in this manner, the new line must be as long (in internal storage) as the old one, unless you wish to rewrite the line index.....

(Line 110 above contains 13 exclamation marks).

Using the MINI MEMORY, you have access to the VDP ram, and you can use this facility to change the definition and colour of the cursor, or even to have a sprite or two running in TI BASIC...
Cursor Definition:
The cursor definition is held in VDP RAM 1008 to 1015 (eg 8 bytes). You are used to defining characters with sixteen hexadecimal characters, but your code is translated by the console to 8 bytes:
Each row of pixels can form a binary number of 8 bits, with the left most pixel having a value of 128. Thus a solid block of pixels in one row can be thought of as the binary number 11111111, which in decimal form is: 128+64+32+16+8+4+2+1 =255

For a block cursor, try:

```
100 CALL POKEV(1008,255,129,129,129,129,
    129,129,255)
110 INPUT A$
```

The cursor will retain its new definition until the console is reset by using NEW or loading a new program.

Cursor colour is defined in VDP RAM 783.
The foreground colour and background colour are contained in a single byte. To separate them you need to divide the byte into two nybbles...
First form the two required colours into binary numbers (binary 0 has a colour value of 1):
WHITE=16 =binary 15 = 1111
TRANSPARENT=1 = binary 0 = 0000
Thus a white cursor on a transparent background would be:
11110000 in binary, which is decimal 240.
To see if this works, try: CALL POKEV(783,240).

Although TI Basic does not recognise sprites, a small part of the memory used by sprites is free, and by placing values there we can place three sprites on screen (stationary):
The sprite definitions must be placed in VDP RAM 768, and are made up of 4 values. The sprite definition must terminate with 208.

The first value is the pixel row (up to 192)
The 2nd value is the pixel column (up to 255)
The third value is the character code (NB: ASCII CODE +96)
The fourth value is the colour code (-1)

Thus:
CALL POKEV(768,98,128,161,1,208)

OR FOR THREE SPRITES:
CALL POKEV(768,98,128,163,1,20,40,164,1
130,170,165,1,208)

The memory area dealing with sprite velocity is also clear, temporarily, but

is used by BASIC for the value stack: in normal operation the computer will remove your velocity data as it moves stack data around.
It IS possible to fool the computer though: the top of the stack can be pushed down out of the way by redefining some of the lower case characters (they are usually derived rather than defined, saving memory).
First use:

```
100 A$="F111"
110 FOR I=96 TO 120
120 CALL CHAR(I,A$)
130 NEXT I
```

Now you can move your sprite(s):
Velocity is to be placed in VDP RAM 1920-, and each sprite requires 4 bytes. The first two bytes are for row and column velocity (max 255) and the other two are for vdp use.

Having placed velocities in the correct VDP RAM, the computer must be instructed to move them! This is done by loading the number of sprites to be moved into CPU RAM -31878.

After the above memory relocation try:

```
200 CALL CLEAR
210 CALL POKEV(768,98,128,161,1,208)
220 CALL POKEV(1920,50,50)
230 CALL LOAD(-31878,1)
240 GOTO 240
```

Being able to use one or two sprites can permit some advanced graphic work in your TI Basic programs. Each sprite can be positioned to within one pixel on the screen, and can be moved one pixel at a time. (A standard character is 8x8 pixels).

GRAPHICS MODES:
THE VDP chip permits the use of 4 graphics modes, but only three are available with BASIC programs, and only one if you only have the console.
The standard mode is 32x24 characters. This is all that is available to you if you only have the console.
Some utility programs are available giving pseudo hi resolution graphics, which work by redefining characters, but they tend to be a little slow.

TEXT mode allows 40x24 characters.It requires a machine code program to allow you to use it (various utilities are commercially available).

MULTICOLOUR MODE divides each character into four blocks, and each block can be any colour.
To see multicolour mode,try the following:
(Requires Mini Memory or Extended Basic + 32k ram):

```
100 CALL INIT
110 CALL LOAD(-31788,204)
120 CALL KEY(0,A,B)
130 IF B<1 THEN 120
140 CALL HCHAR(1,1,45,200)
150 FOR Z=1 TO 57
160 FOR X=1 TO 14
170 PRINT CHR$(Z+30);
180 NEXT X
190 NEXT Z
200 CALL LOAD(-31788,224)
210 CALL KEY(0,A,B)
220 IF B<1 THEN 210
230 END
```

Enter RUN, then press any key to start the action. At program end,press another key to return to normal. If necessary switch off to return to normal!

HI RESOLUTION MODE allows pixel plotting, but with nearly 49000 pixel positions (plus colour information) there is no room to operate both this mode and the BASIC operating system. It is only possible in machine code programs- such as PARSEC.

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

BOB BOONE.....HARDWARE/SOFTWARE.....623-7841

DICK PICHE.....TECH.....521-8667

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

1988 RENEWAL $20.00

NEW MEMBERS $20.00

NAME _____

ADDRESS _____

CITY _____ PROVINCE/STATE _____

POSTAL CODE _____ TELEPHONE (____)_____

Please make check payable to the OTTAWA TI-99/4A USER'S GROUP and send it, along with this form, to the address shown on the cover page, or better still, bring both to a meeting.

FROM

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