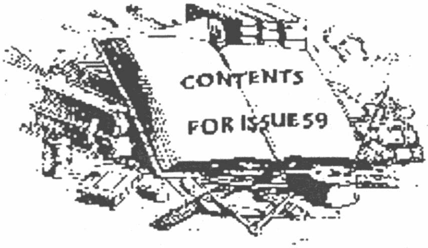
TI*MES

Issue 59 Winter 1997

The Magazine of the Texas Instruments TI-99/4A User Group UK

Committee Members

Chairman / BBS....	Trevor Stevens	Notts
Vice Chairman....	Mark Wills	Shrops
General Secretary/ Co-Editor..........	.Richard Twyning	Notts
Co-Editor.........	Ian Pare	Notts
Treasurer...........	Alan Rutherford	Cheshire
Disk Librarian.....	Stephen Shaw	Cheshire
Module and Cassette Librarian.......	Francesco Lama	Oxon
Hardware......	Ross Bennett	Cheshire



Editorial 4

BBC to Texas Sound Converter. 5

From the Chairmans Chair. . Trevor Stevens 6

Ski-ing Listing 17

News & Reviews Richard Twyning 20

TI Music Listing "The Brook". 23

USA News & Update. . Richard Speed 26

Woodstock's Christmas Present Listing. 32

In A Maze Listing. 39

Hi-Res Plotter Listing 44

More TI Tunes 47

Disclaimer

The views expressed in this magazine are those of the individual authors and not necessarily those of the Editor or the Group.

Editorial E-Mail : TI99@infoserv.demon.co.uk

EDITORIAL

Yes folks, here's the Winter issue of TI*MES (59), and its still winter of 1997. My God, I hear you cry in dis-belief, there can't be much doing in this issue ! Well thats were you would be wrong. Mr. Chairman updates us with some States side news and continues his series of articles on TI Assembler. Mr. General Secretary gives us the low down on the Britsh escapades at TI-TREF in Holland. Also this issue contains lots of listings to keep you busy over the festive season, if you find your selves bored with the re-runs of James Bond and Terry Wogan's reviews of the year.

As were now fast approaching the end of the 20th Century, and preparing to move forward into the next, it seemed appropriate to mention the current state of the membership numbers (which is also reflected in the Graphs in Richard's article). Having talked to Ross recently, he informed me that the current membership is now down to less than 70, which came as a big surprise to me, as I was previously informed that there were around 100 members.

At the last AGM this issue was raised, and a plan was put into place to try and do something about this, with certain individuals elected to take responsibility for publicity and awareness of the group. I ask only this question... if nothing is being done to try and find new members and the membership numbers continue to fall... at what point does the feasability of this magazine come into question ? Food for thought... comments please !

Can I make a formal request to the Group executive to appoint someone who will be resonsible for publicity. There has to be other people out there in the UK waiting to be found.

Well, all it leaves for me to say on behalf of Richard Twyning and myself is have a great Christmas and a Happy New Year.

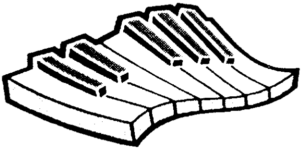
Ian Pare (Co-Editor)

Merry Christmas

BBC to Texas Sound Convertor

The table below converts BBC Micro sound values to their corresponding Texas sound Values. This allows music listings written for the BBC to be converted to the Texas.

Octave	BBC		Texas
1	0	A#	117
	1	B	123
	5	C	131
	9	C#	139
	13	D	147
	17	D#	156
	21	E	165
	25	F	175
	29	F#	185
	33	G	196
	37	G#	208
	41	A	220
2	45	A#	233
	49	B	247
	53	C	262
	57	C#	277
	61	D	294
	65	D#	311
	69	E	330
	73	F	349
	77	F#	370
	81	G	392
	85	G#	415
	89	A	440
3	93	A#	466
	97	B	494
	101	C	523
	105	C#	554
	109	D	587
	113	D#	622
	117	E	659
	121	F	698
	125	F#	740
	129	G	784
	133	G#	831
	137	A	880
4	141	A#	932
	145	B	988
	149	C	1047
	153	C#	1109
	157	D	1175
	161	D#	1245
	165	E	1319
	169	F	1397
	173	F#	1480
	177	G	1568
	181	G#	1661
	185	A	1760
5	189	A#	1865
	193	B	1976
	197	C	2093
	201	C#	2217
	205	D	2349
	209	D#	2489
	213	E	2637
	217	F	2794
	221	F#	2960
	225	G	3136
	229	G#	3322
	233	A	3520
	237	A#	3729
	241	B	3951
	245	C	4186
	249	C#	4435
	253	D	4699



FROM THE CHAIRMANS CHAIR

By T.STEVENS

Well the year has nearly finished and another TI season comes to an end and the next one starts. Not bad for a computer that was said to be dead. Texas Instruments themselves have said as much because the company have allowed TI user group SW99UG (Southwest 99ers User Group) to hold the 1998 Fest WEST in their LUBBROCK factory. The show will be held on Saturday 14th February 1998 from 8am to 6pm. There will be stalls and Demos, talks and displays. The show will be split between the LUBBROCK factory and the nearby Sheraton Four Points Hotel. If any one out their wants more details they can contact me and I will send you a copy of the documentation I have.

All went well with the trip to Holland. As you may have learned in this issue I went with Richard Twyning and Paul who is Richards University mate. We went across to Holland in my car, which is a Ford Granada, so we had some real comfort to travel long distances. On getting off the boat we travelled about two hours and passed through France and got into Belgium. We slept in the car at a place called Gent, which was as far as I was prepared to go on the first leg of the trip. We got up in the morning and went into Holland. We stopped for breakfast at the Dutch boarder at a McDonalds. Strange really as they were cheaper and were giving more for your money than the other Dutch Café. After filling our batteries we set off towards our final destination which was UTRECT. The hotel was good and the food fine. Most of the TI folks arrived for about 6pm that day which was the Friday. From then on it was a learning curve to meet new friends. I have brought back some nice memories after chatting with these people. Satrday was the BIG day and more ideas and thoughts were exchanged between us. That evening, the English contingent, which consisted of us three, and Ross and Christine BENNETT, who had arrived by plane on the Friday, went out to Amsterdam on the train. Should I say this but they run better than our trains for time and frequency even throughout the night. We found a lovely Restaurant and all had a lovely meal. We then went out and saw the SIGHTS!!! The next day saw the farewells and we set off home at about 11.30am. We made good time and arrived in Calais at about 4pm. The three of us did some essential buying from the local wholesale wine markets and got on the boat for 5.30pm. After landing in Dover 90mins later we headed home. To say the least I am well stocked up with contacts and booze, as result of the trip to the land of Tulips and Dykes.

To give you some idea of what we saw at the show I will now describe to you what I saw which was out of the normal. The best for me was how the Germans love to play with the TI. There was one lad there who was using his TI to run a laser light show. He had the patterns running on the wall of the room he was in. He had pre-programmed the movements of the laser senders and the shapes then appeared. He had a teddy bear and spirograph type shapes whizzing around on the wall.

He was also using his TI to run TIBASE, which in my opinion was not well programmed, but he was producing a definitive database of all TI Users. So he got me included on it. Then there was Mr Stock Broker. His son together with Oliver ARNOLD had produced this really cleaver program, which reads data via the serial port from a teletext television, and reads the data off the stock market pages. It then puts them in day by day into the computer and you can analyse the trading figures to se which ones to buy and sell, if you are that way inclined. The only BIG down side is that you need a Geneve and a hard drive to run the program as it is so BIG! However if you saw it you would be amazed at what it could do. I brought back a leaflet about it and how you can use the TELETEXT feature on your TI/994a. I will now type out the leaflet for you, so here goes.

The TELETEXT-Card for the TI or Geneve.

The Teletext card is a new hardware project for the TI/994a and the Geneve. The card is an external device, which is connected to the RS232 interface. On the other side you need a CVBS signal from a Television or something else. This can include a satellite receiver. The software is mostly written in C99 and mixed with assembly language. This software controls the decoder chip (SAA5246A/PE) via the RS232. The software needs 32k of memory to run, a Diskdrive and a program file loader like Editor/Assembler. All the hardware is tested before you get the device however there is a back up service in case things do go wrong. The device is copyright by Oliver ARNOLD however the software is Fairware.

On power up to your computer you need to use a program called TEST80 or TEST40 depending on you system type (80 col 40 col) this program gives you info on the CVBS signal received. If all is well then you load a program called Menu 80 or Menu 40 and this loads in the data from the CVBS signal.

When you have loaded the Menu program and obtained your data you can do the following. Search for pages equally. Dump pages to printer. Save them to a device like a Disk Drive. Show memory status. Show moving index, date, time, and channel. Find subpages. The program allows you to access all of these by simple key presses.

The Teletext Card has 8k of memory inside of it for saving 8 pages of information. From the program you also have full access to this memory. To get a page you have to transfer 1KB of data via the RS232. And convert it into ASCII codes. This takes less than one second to do. During the search process you see the index moving like on your TV so the Time etc is updated. Full Graphic are supported but if you store pages these graphics get stripped off. The program uses TEXT mode for the display screen. In this mode it is not possible to display more than two colours. If however you have an 80-col device, you will see two pages at once. This is just an overview of the documentation but it gives you some idea what it does. If you are interested then write to OLIVER ARNOLD Implerstr. 8 81371 Munchen Germany. I can also give you a copy (Oliver said I could) of the documentation that I have on the project. You can also see this if you went to the Workshop in Sandbach.

Well that's the chit chat out of the way. We will now move onto some computing which I know we all love to do in various languages. We continue with the Machine code Tutor for the Minimemory which we have been doing for some time now. I mentioned some time back that this would be lengthy and quite deep but I hope you all understand what is going on. So here we go.

Mini Mem Tutor

Saving Memory

Well this is a very important thing in the Minimemory as the system only allows space for program of about 4k. However this memory is also used up for the Symbol Table which put labels into it. So every label you use, you use up valuable memory space. When you start your program you are defaulted to the location 7D00 the symbol table starts at the lower position of 7CD8, thus allowing for 9 labels in your program. However if you use more then the symbol table will over write the start of your program. So planning of labels is essential. So if you only use two labels you can move your program back into the symbol area and take advantage of the extra bit of memory. You can do this by counting the number of labels you use, then add one dummy; this is because the computer adds an extra one as a matter of course. Multiply this figure by four (4) because each entry is four bytes. Convert to hexadecimal so you can add or subtract from the symbol table, and this will give you exact start location.

So assume we have 14 labels the calculation would look like this

(14+1)*4=60   60 = >3C
So 7CD8 + >003C = >7D14
The location 7D14 being your start address.
If you program only had say three labels then it works the same.
(3 + 1) * 4 = 16
16 = >10
>7CD8 + >0010 = >7CE8

Labels use up a lot of valuable space and can be avoided in several ways.

TIP 1

Do not use the EQU directive. Instead of giving a memory location or routine a label, branch to the memory location directly. Code one is the memory eater, Code two the simple most economical method.

```
1. 7D00    N3    EQU >6034
   7D00          BLWP @N3
2. 7D00
```

If you know where you are going to place a subroutine, instead of branching to a label and assigning it later, branch directly to the starting memory address of the routine.

```
1. 7D00          BL @CC
      .
      .
      .
   7EF8   CC     CLR R0

2. 7D00          BL @>7EF8
      .
      .
      .
   7EF8          CLR R0
```

Again the second method does the same thing but is much more of a lean machine.

The same method can be used with the jump and branch instruction.

JUMP

```
1. 7D00          JMP S9
      .
      .
   7EF8   S9     LI R7,5
```

```
2. 7D00          JMP>7D50  (NO @NEEDED)
     .
     .
   7D50          LI R7,5
```

BRANCH

```
1. 7D00          B @PQ
     .
     .
    7E50  PQ     CLR R1

2. 7D00          B @>7E50
     .
     .
   7E50          CLR R1
```

You can also refer directly to memory location when using a DATA table or text if you know where it is in your program, or will be in memory.

```
1. 7D00          LI R0,300
   7D04          LI R1,TX
   7D08          LI R2,5
   7D0C          BLWP @>6028
     .
     .
   7F30   TX     TEXT 'LABEL'

2. 7D00          LI R0,300
   7D04          LI R1,>7F30
   7D08          LI R2,5
   7D0C          BLWP @>6028
     .
     .
   7F30          TEXT 'LABEL'
```

THE $ SIGN

The assembler predefines the dollar sign($) to mean "the current memory location." This is a great help when your program is jumping around in memory without using labels. For instance, both these instructions mean the same.

```
1. 7D00           JMP$
2. 7D00   NQ      JMP NQ
```

The first example means to jump to the current memory location, which happens to be >7D00. The second, which also creates an endless loop, does the same, but uses a label and more memory to do the same task.

The dollar sign can also allow you to jump locations say six bytes forward in memory. You do this like the following example.

```
7D00      JMP $+6      (THIS PASSES CONTROL TO >7D06)
```

You can subtract as well or move back.

```
7D0A      JMP $-4      (THIS PASSES  CONTROL BACK TO >7D06)
```

PROGRAM EXECUTION

When you have finished writing your assembly program there are three ways you can execute or run your program. The first and most immediate is to END your program then exit the assembler, select Easy Bug and use the (E) command, this is followed by the Hex address of the program start. We have used this method to test our examples, or I hope you have.

The second way is to add a name and the position into the REF DEF table. (Table of REFerences and DEFinitions) and executes the program like in LINES. You then run this from the Mini Memory Menu. (I will show you how to do this in a bit)

The third way also requires the name and start position of the program adding to the REF DEF table. To call it from Basic you have to use the CALL LINK subroutine, which must follow the following Syntax. CALL LINK("program name") The program name must be the same one that is in the REF DEF TABLE. Care must be taken when linking Basic and assembly programs. We will discuss this later in the series. To test the theory try this, load into Mini Memory the LINES program from Tape. In command mode (No program lines) Type CALL LINK("LINES") and then press ENTER.

ADDING THE PROGRAM NAME

This operation is fairly simple. (He SAYS!) The REF DEF table starts at memory location >7FFF and grows backwards to >7000. That means that it occupies the very last area of your memory. Each entry is only eight (8) Bytes long thus several program names can be added to the table. The program name only uses six (6) bytes of memory and the starting address only two.

When you load the assembler, the entry for the NEW option of the assembler occupies addresses >7FF8 to >7FFF. The OLD option occupies addresses >7FF0 to >7FF7. LINES is located at >7FEF to >7FE8

You can add the name and position of your program before the entry for LINES from >7FEF to >7FE8. You can also use the entry of LINES as your program will over write some of the LINES program anyway. You must remember that your program must not be longer than the place where you will add the name and position of it in the table, or you will overwrite your own program.

Two memory addresses tell you the Last Free Address of the Module (LFAM) and First Free Address of the module (FFAM). These addresses are (FFAM >701C) (LFAM >701E) These programs tell you where your program finished (FFAM) (The next free address after your program) and the LFAM (the place in memory where your program name and position are loaded).

To check this load up LINES from Tape and choose the NEW option of the Assembler.

Then Type.

```
7D00 045B          AORG >701C
```

To get the FFAM

You will see displayed on your screen

701C 7FB2

This value (>7FB2) is the FFAM, the first free address after the program LINES. Press ENTER to get to the next memory location, >701E, where the LFAM is stored. Now you should see:

7D00 7FE8

This means that the last free position **BEFORE** the REF/DEF Table entry for lines is >7FE7. The entry for LINES begins at >7FE8

When you finished writing your program, you must update these values. Use AORG to get to the correct memory locations and DATA to place the correct values there. Remember that your program has to leave eight bytes for the name and starting address of your program. When you have updated the values in >701C and >701E, you can proceed to add your program entry to the REF/DEF table.

The steps to change the FFAM and LFAM are:

Use AORG to get to >701C, where the FFAM is stored

Use DATA to change the value there to the new first free address after your program is finished.

Use Data to change the value in >701E to the place in the REF/DEF table where you will add the entry for your program. To use the same entry as line use >7FE8. For the entry before LINES, use >7FE0. For each new entry, subtract eight bytes from the previous one. Check that that the place where you will make your entry for your program (the value you place in >701E) is at least seven bytes or more than the value in >701C, or you will not be able to add you entry as there will not be enough room.

Once we have updated the FFAM and LFAM addresses in memory, you can then add the name and starting position of your program to the REF/DEF table. To do this, use the AORG directive to get to the place where you want to make the entry for your little program. (This should be the same location you stored earlier in >701E.) Once there, use the TEXT directive to place the name of your program. The name can be one to six characters long. If it is shorter than six you MUST pad out with blanks In other words the entry must have six positions filled for a valid entry. Then Use the DATA directive to add the starting position of your program into memory. If your program starts say at >7D03, enter DATA >7D03. If the program's first instruction is labelled, with say B6 you could start with this DATA B6.

AN EXAMPLE

It is always easier to understand something if you have an example in front of you. So let's do it! We will write a very short program and add a program name.

```
7D00        LWPI >70B8
7D04        LI R0,298
7D08        LI R1, TX
7D0C        LI R2,8
7D10        BLWP @>6028
7D14        JMP $

7D16  TX  TEXT ‘TI-99/4A’

7D1E        AORG >701C        Jump to the FFAM
701C        DATA >7D1E        Set new FFAM
701E        DATA >7FE0   Set new FLAM for the first entry in REF/DEF
7020        AORG >7FE0        Jump to the location 7FE0
7FE0        TEXT  `TI-99 `    NAME = TI99  (note the blank)
7FE6        DATA >7D00        Start of our program

7FE8        END
```

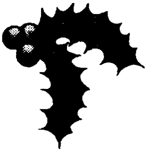
Now run the program using the RUN option of the MINI Memory menu. You do not have to leave a blank space as the computer does this for you. So type TI99 enter and the program will run. The program will not work with a CALL LINK in basic however we will discuss this later in the series and show you how this is done. Lets stick to the basics first.

You have now completed the basics of Assembly programming and you have been through all the directives and instructions. You have seen how to write a program in a simple form. From now on in we start to go in depth with assembly programming. If you do not understand anything just re read and practice.

Next time we will be looking at the controls of your computer i.e. KEYBOARD and the like so to next time have fun.

Memory Very Full Fctn/QUIT

Happy Christmas and New Year to you all out
there in TI LAND.



WANTED TI-99/4A SURPLUS

If you have any equipment you no longer need or is faulty, Please let our hardware Guru Know. Fair Prices paid. Also Repairs at no charge for group members.

Please do not throw anything away !!

Ross Bennett, 20 Oak Avenue, Romiley, Stockport, SK6 4DN.
TEL :0161-430-7298 Evenings/Weekends,
FAX :0161-483-4516 (24-Hour)

TI SKI-ING

A seasonal listing for the un-expanded TI

```
10 CALL CLEAR
20 PRINT TAB(10);"SKI-ING": : : : :
: :"DO YO WANT INSTRUCTIONS(Y/N)":
: : : : :
 : : : : : : : :
30 CALL KEY(0,K,S)
40 IF S=0 THEN 30
50 IF K=89 THEN 110
60 IF K<>89 THEN 120
100 CALL CLEAR
105 GOTO 120
110 GOTO 940
120 CALL CLEAR
130 CALL SCREEN(5)
140 CALL COLOR(1,3,16)
150 P=3
160 RESTORE
170 CALL
CHAR(104,"0C0C191E180809FE")
180 CALL
CHAR(112,"101038387C7CFE10")
190 CALL
CHAR(120,"0C3C7C3C0C040404")
200 CALL
CHAR(128,"383810FEBABA286C")
210 CALL
CHAR(129,"38BA92FE3838286C")
220 CALL
CHAR(105,"30309878181090FF")
230 CALL COLOR(11,3,16)
240 CALL VCHAR(1,1,112,22)
250 CALL VCHAR(1,32,112,22)
260 CALL HCHAR(1,1,112,32)
270 CALL HCHAR(22,1,112,32)
280 FOR N=0 TO 41
290 READ A,B,C
300 FOR M=A TO B
310 CALL HCHAR(C+1,M+1,112)
320 NEXT M
330 NEXT N
```

```
340 CALL COLOR(12,9,16)
350 FOR N=1 TO 31
360 READ A,B
370 CALL HCHAR(A+1,B+1,120)
380 NEXT N
390 CALL COLOR(5,2,16)
400 CALL HCHAR(5,6,32)
410 CALL HCHAR(5,6,32)
420 FOR N=5 TO 7
430 CALL COLOR(N,2,16)
440 NEXT N
450 CALL HCHAR(2,2,83)
460 CALL HCHAR(3,2,84)
470 CALL HCHAR(4,2,65)
480 CALL HCHAR(5,2,82)
490 CALL HCHAR(6,2,84)
500 CALL HCHAR(16,31,70)
510 CALL HCHAR(17,31,73)
520 CALL HCHAR(18,31,78)
530 CALL HCHAR(19,31,73)
540 CALL HCHAR(20,31,83)
550 CALL HCHAR(21,31,72)
560 E=104
570 X=3
580 Y=3
590 A=0
600 B=0
610 CALL KEY(0,K,S)
620 IF K=83 THEN 800
630 IF K=68 THEN 840
640 IF K=88 THEN 880
650 IF K=69 THEN 910
660 X=X+A
670 Y=Y+B
680 CALL COLOR(10,2,16)
690 CALL COLOR(13,2,16)
700 CALL GCHAR(X,Y,C)
710 IF C=112 THEN 1760
720 IF C=120 THEN 2100
730 IF
(C=70)+(C=72)+(C=78)+(C=83)+(C=
72)THEN 1470
740 CALL HCHAR(X,Y,E)
750 CALL SOUND(200,-5,9)
760 CALL HCHAR(X,Y,32)
770 CALL HCHAR(22,9,128,21)
780 CALL HCHAR(22,9,129,21)
```

Ski-ing Listing

```
790 GOTO 610
800 B=-1
810 A=0
820 E=105
830 GOTO 630
840 B=1
850 A=0
860 E=104
870 GOTO 640
880 B=0
890 A=1
900 GOTO 650
910 B=0
920 A=-1
930 GOTO 660
940 CALL SCREEN(5)
950 FOR N=1 TO 9
960 CALL COLOR(N,5,16)
970 NEXT N
980 CALL CHAR(35,"3C4C99A1A199423C")
990 CALL CLEAR
1000 K$="99'ER SKIING BY "
1010 Z=2
1020 GOSUB 1350
1030 K$="£ DAVID KIMBERLIN-WYER "
1040 Z=4
1050 GOSUB 1350
1060 K$="WELCOME TO SKIING.MAKE YOUR
"
1070 Z=6
1080 GOSUB 1350
1090 K$="WAY DOWN THE COURSE USING
THE "
1100 Z=8
1110 GOSUB 1350
1120 K$="CURSOR KEYS TO DODGE THE
FLAGS "
1130 Z=10
1140 GOSUB 1350
1150 K$="AND FOREST TO REACH THE
FINISH "
1160 Z=12
1170 GOSUB 1350
1180 K$="AND COLOURFUL CELEBRATION.
"
1190 Z=14
1200 GOSUB 1350
1210 K$="BUT BE WARNED YOU ONLY "
1220 Z=17
1230 GOSUB 1350
1240 K$="HAVE 3 LIVES. "
1250 Z=19
1260 GOSUB 1350
1270 FOR N=262 TO 312 STEP 8
1280 CALL SOUND(100,N,0)
1290 CALL SOUND(100,N-60,0)
1300 NEXT N
1310 FOR M=1 TO 600
```

```
1320 NEXT M
1330 CALL CLEAR
1340 GOTO 120
1350 V=16-INT(LEN(K$)/2)
1360 FOR I=1 TO LEN(K$)
1370 CD=ASC(SEG$(K$,I,1))
1380 CALL HCHAR(Z,V+I,CD)
1390 NEXT I
1400 RETURN
1410 DATA
5,30,1,6,8,2,13,30,2,25,30,3,28,30,
4,27,30,5,1,3,6,9,10,6,22,23,6,29,3
0,6,1,23,7
1420 DATA
30,31,7,1,7,8,12,23,8,30,31,8,1,6,9
,17,22,9,1,4,9,30,31,9,1,1,10,30,31
,10,1,2,11
1430 DATA
29,30,11,1,1,12,10,11,12,29,30,12,1
,1,13,9,14,13,22,30,13,1,1,14,5,15,
14,21,30,14
1440 DATA
1,1,14,1,1,15,5,30,15,1,1,16,6,30,1
6,1,2,17,1,3,18,19,24,18,1,6,19,1,3
0,20
1450 DATA
3,8,3,5,5,5,5,8,2,11,4,11,4,17,6,17
,3,22,5,22,7,28,9,28,9,25,12,23,10,
23,11,19
1460 DATA
13,19,10,15,12,15,8,10,10,10,10,7,1
2,17,13,2,13,4,16,4,16,2,17,9,19,12
,17,12,17,18
1470 FOR N=110 TO 210 STEP 8
1480 CALL SCREEN(INT(RND*15)+1)
1490 CALL COLOR(11,3,16)
1500 CALL COLOR(10,2,16)
1510 CALL COLOR(13,2,16)
1520 CALL SOUND(100,N,0)
1530 CALL SOUND(100,N+5,0)
1540 NEXT N
1550 CALL CLEAR
1560 CALL COLOR(1,12,9)
1570 CALL
CHAR(33,"FFFFFFFFFFFFFFFF")
1580 FOR X=8 TO 14
1590 CALL HCHAR(X,9,33)
1600 CALL HCHAR(X,25,33)
1610 NEXT X
1620 CALL HCHAR(8,9,33,16)
1630 CALL HCHAR(14,9,33,16)
1640 FOR N=4 TO 9
1650 CALL COLOR(N,2,9)
1660 NEXT N
1670 K$=" WE HAVE "
1680 Z=9
1690 GOSUB 1350
1700 K$=" A WINNER "
```

```
1710 Z=11
1720 GOSUB 1350
1730 INPUT "DO YOU WANT TO PLAY
AGAIN":T$
1740 IF T$="Y" THEN 100
1750 STOP
1760 CALL HCHAR(X,Y,E)
1770 FOR N=170 TO 220 STEP 8
1780 CALL SOUND(100,N,0)
1790 CALL SOUND(100,N-60,0)
1800 NEXT N
1810 FOR N=5 TO 9
1820 CALL COLOR(N,2,16)
1830 NEXT N
1840 K$="BAD LUCK YOU HIT A TREE "
1850 Z=24
1860 GOSUB 1350
1870 FOR M=110 TO 155 STEP 8
1880 CALL SOUND(100,M,0)
1890 NEXT M
1900 CALL HCHAR(24,1,32,32)
1910 K$="LOOSE ANOTHER LIFE "
1920 Z=24
1930 GOSUB 1350
1940 FOR N=110 TO 210 STEP 8
1950 CALL SOUND(100,N,0)
1960 NEXT N
1970 CALL HCHAR(24,1,32,32)
1980 P=P-1
1990 IF P=0 THEN 2290
2000 E=104
2010 CALL HCHAR(X,Y,112)
2020 GOTO 570
2030 CALL HCHAR(X,Y,E)
2040 FOR N=250 TO 110 STEP -8
2050 CALL SOUND(100,N,0)
2060 NEXT N
2070 FOR N=5 TO 9
2080 CALL COLOR(N,2,16)
2090 NEXT N
2100 K$=" HA HA YOU HIT A FLAG "
2110 Z=24
2120 GOSUB 1350
2130 FOR N=262 TO 110 STEP -8
2140 CALL SOUND(100,N,0)
2150 NEXT N
2160 CALL HCHAR(24,1,32,32)
2170 K$="LOOSE ANOTHER LIFE "
2180 Z=24
2190 GOSUB 1350
2200 FOR N=196 TO 130 STEP -8
2210 CALL SOUND(100,N,0)
2220 NEXT N
2230 CALL HCHAR(24,1,32,32)
2240 P=P-1
2250 IF P=0 THEN 2290
2260 CALL HCHAR(X,9,120)
2270 E=104
2280 GOTO 570
2290 CALL CLEAR
2300 CALL SCREEN(2)
2310 CALL COLOR(1,16,1)
2320 FOR N=5 TO 9
2330 CALL COLOR(N,9,16)
2340 NEXT N
2350 K$="GUESS YOU RAN "
2360 Z=10
2370 GOSUB 1350
2380 K$="OUT OF LIVES "
2390 Z=12
2400 GOSUB 1350
2410 PRINT "DO YOU WANT TO PLAY
AGAIN": :TAB(12);"(Y/N)": : : : :
2420 CALL KEY(0,K,S)
2430 IF S=0 THEN 2420
2440 IF K=89 THEN 100
2450 IF K<>89 THEN 2460
2460 END
```

Richard Twyning (General Secretary) -

News and Views

DV80 files on the BBS, and in particular, uploaded TI*MES articles, I've just realized that it would be nice if Trevor could keep an eye on the TI*MES articles area, and once a new file comes in, he could convert it for us. Un-ARK it if it's archived, and convert any DV80's / DF80's into standard PC text files.

When we access the BBS then, to download any articles, we can be fairly sure that they are already converted, and don't need excessive playing with, to get them into the mag!!

I'll have to mention it to him.

At the TI-Tref, I'm hoping to meet up with Micheal Becker, and you will find out whether I do or not later in this article! I have heard from Francesco that Michael is definitely going to the show, but unfortunately, Francesco can't make it.

I want to talk to him not just about the SCSI card, but about what software is currently available for the GENEVE to allow it to use both the SCSI card, and the MYARC HFDCC. Is all the software complete for the HFDCC so I can use my 1.44Meg Floppy drive as a full 1.44 Meg floppy drivee, and will the HFDCC work in the same box as the SCSI card, and do I still need my TI floppy controller in the box to boot up from, and can I use the full set of commands with the SCSI card, so I'll be able to copy every file from a floppy disk in DSK1 by typing COPY A:* in MDOS (assuming that there's an ASSIGN in my AUTOEXEC file as follows: ASSIGN A=DSK1:).

Will I ever be able to use floppy disks with the SCSI card by using the FC1 card, and will this give me the claimed 3.5 Megs of (useable) disk space on a 2.88 Meg floppy, and will it read PC disks as well, as was promised?

All mysteries waiting to be solved, and something that I am hoping will be solved at the show.

It was rumoured that Don Walden from CECURE would be there, but people said that they couldn't see it happening. It would be nice to hope that some Americans make it over, and bring some new wondrous hardware! I seem to have fallen behind with things over the last couple of years, and Ian told me that I seem to have been on a steady decline in my enthusiasm for the TI over the past year.

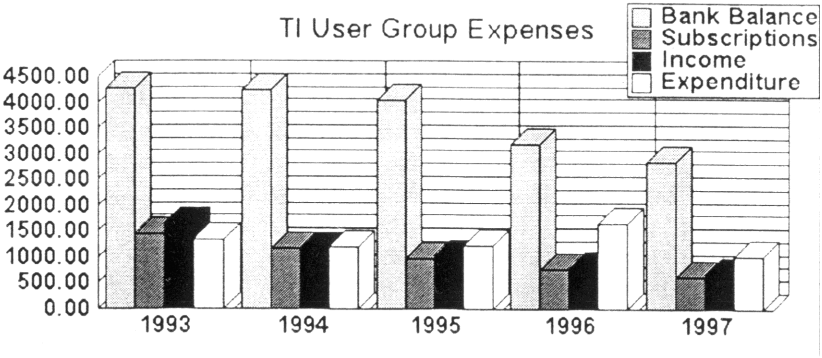
It's months now since the GENEVE was powered up. I can still remember the night that I first met Trevor when he came to my place with Gary. I had the GENEVE up and running with my HFDCC, and my 40Meg ST251-1 hard disk. Everything was clean and smooth. I could come home at night, and flip a couple of switches and my GENEVE would boot up in a few seconds from hard disk.

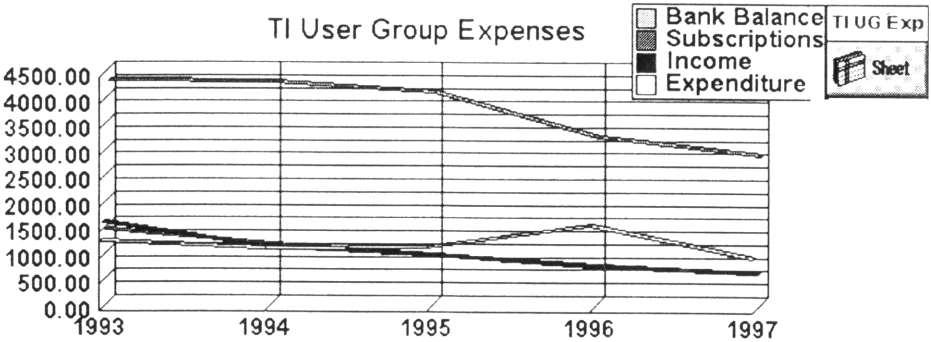
I think the reason I stopped using it was because I'd tried to copy more software onto the drive, and a BIG BUG in the version of MYARC Disk Manager 5 I was using, meant that it didn't stop writing when the drive became full! It somehow looped back on itself to a different section of the drive and started overwriting part of the directory structure! It's a good job I did a backup onto floppy. I still have the floppy backup, but I still haven't got around to sitting down properly and transferring it onto my PC so that I'll be able to have it written onto CD-ROM, unless the Germans have sussed out how to run a CD-ROM writer on the SCSI card!

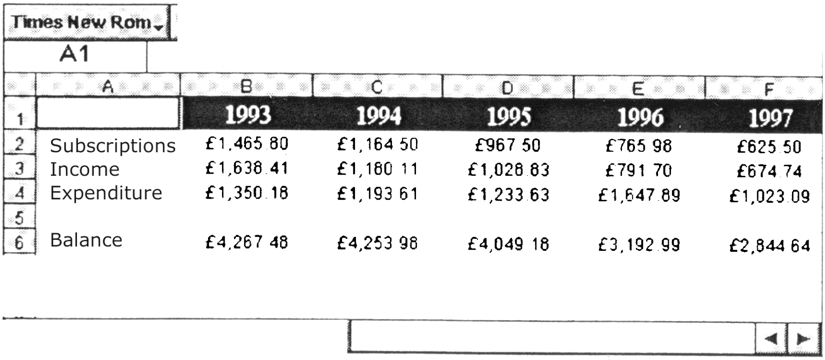
When I heard about the SCSI card I decided that it was what I needed to sort out my filing system, and I had always planned to get something like the Optical Floppy drive to give me a VERY reliable backup, but the rest is history, and it's still not running!

In a way, I'm frightened to attempt setting up my system fully, in case anything doesn't work! I just need to get over this, and get some guaranteed software that will work with the GENEVE, the SCSI card, and also the HFDCC, then I can get my system working again and then make a start on my long term programming projects.

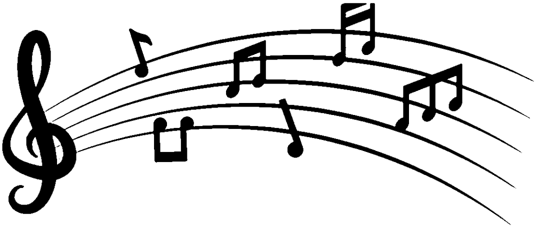
Unfortunately, the disks that Berry Harmsen sent me must have been in either double density, or 80-track, or even both, because they were unreadable on my old Panasonic 40 Track drive.







	A	B	C	D	E	F
1		1993	1994	1995	1996	1997
2	Subscriptions	£1,465.80	£1,164.50	£967.50	£765.98	£625.50
3	Income	£1,638.41	£1,180.11	£1,028.83	£791.70	£674.74
4	Expenditure	£1,350.18	£1,193.61	£1,233.63	£1,647.89	£1,023.09
5						
6	Balance	£4,267.48	£4,253.98	£4,049.18	£3,192.99	£2,844.64



TI MUSIC

“THE BROOK”

By Ian Pare

This was a program that I wrote to play some music that was originally printed in Acorn User for the BBC micro. It was quite easy to do once I had used to sound convertor (printed in this issue) to transform BBC sound values to TI Sound values.

```
100 CALL CHAR(42,"FFFFFFFFFFFFFFFF")
110 CALL CLEAR :: CALL SCREEN(2):: FOR A=2 TO 12 :: CALL
COLOR(A,8,1):: NEXT A
120 CALL COLOR(2,13,1)
130 DISPLAY AT(7,10):"THE BROOK"
140 DISPLAY AT(10,8):"(C) Acorn User"
150 CALL HCHAR(5,8,42,18)
160 CALL HCHAR(12,8,42,18)
170 CALL VCHAR(5,8,42,8):: CALL VCHAR(5,25,42,8)
180 DISPLAY AT(22,2):"  TEXAS INSTRUMENTS 1984"
181 CALL HCHAR(1,1,42,32):: CALL HCHAR(24,1,42,32):: CALL
VCHAR(1,1,42,24):: CALL VCHAR(1,32,42,24)
190 FOR Z=1 TO 151
200 READ A,B,C
210 CALL COLOR(2,INT(RND*12)+4,1)
220 CALL SOUND(C*9.2,A,0,B,0)
```

```
230 NEXT Z
240 CALL CLEAR :: END
250 DATA 784,784,50,1047,659,25,1047,523,25,1047,659,25,1047,523,
         25,784,659,25,784,523,25,784,659,25,784,523,25

260 DATA 880,698,25,784,523,25,880,698,25,1047,523,25,784,659,25,
         784,523,25,784,659,25,784,523,25

270 DATA 880,698,25,880,523,25,880,698,25,880,523,25,988,587,25,
         988,523,25,988,587,25

280 DATA 988,523,25,1047,659,25,1047,523,25,1047,659,25,1047,
         523,25,784,659,25,784,523,25

290 DATA 784,659,25,784,523,25,1047,659,25,1047,523,25,1047,523,
         25,1047,523,25,784,659,25

300 DATA 784,523,25,784,659,25,784,523,25

310 DATA 880,698,25,784,523,25,880,698,25,1047,523,25,784,659,
         25,784,523,25,784,659,25

320 DATA 784,523,25,880,698,25,880,523,25,880,698,25,880,523,
         25,988,587,25,988,523,25

330 DATA 988,587,25,988,523,25,1047,659,25,1047,523,25,1047,
         659,25,1047,523,25,1047,659,25

340 DATA 1047,1047,75

350 DATA 988,587,25,988,523,25,1175,698,25,1175,523,25,1047,
         659,25,1047,523,25,1047,659,25

360 DATA 1047,523,25,988,587,25,988,523,25,1175,698,25,1175,
         523,25,1047,659,25,1047,523,25

370 DATA 1047,659,25,1047,523,25,880,587,25,784,587,25,880,
         587,25,988,587,25

380 DATA 880,587,25,988,587,25,1047,587,25,880,587,25

390 DATA 988,784,25,1047,784,25,988,784,25,988,698,25,784,659,
         25,784,587,25,1047,659,25

400 DATA 1047,523,25,1047,659,25,1047,523,25,784,659,25,784,523,
         25,784,659,25,784,523,25

410 DATA 880,698,25,784,523,25,880,587,25,1047,523,25,784,659,25,
         784,523,25,784,659,25

420 DATA 784,523,25

430 DATA 880,698,25,880,523,25,880,698,25,880,523,25,988,587,25,
         988,523,25,988,587,25

440 DATA 988,523,25,1047,659,25,1047,523,25,1047,659,25,1047,523,
         25,1047,659,25,1047,523,25
```

```
450 DATA 784,659,25,784,523,25,880,698,25,1047,523,25,880,698,25,
         1047,523,25,988,587,25

460 DATA 988,523,25,988,587,25,988,523,25

470 DATA 1047,659,25,1047,523,25,1047,659,25,1047,523,25,1047,659,
         25,1047,523,25,784,659,25

480 DATA 784,523,25,880,698,25,1047,523,25,880,698,25,1047,523,25,
         988,698,25,1047,698,25

490 DATA 1175,698,25,988,698,25,1175,698,100,1047,659,100
```

NEXT ISSUE

TI*MES 60

SPRING

1998

We Want your input...

USA News & Update

from Richard Speed, the man with his finger on the pulse of TI life in the U.S. of A.

Hello once again from the unacceptable face of real-world computing. My apologies if this issue's instalment is a tad short, but TI*MES is getting to be like a London Transport bus – you wait all day for one, and then a whole bunch come at once!

Seriously though, having been editor myself for a while, I can fully understand Ian's problems with balancing job, family and magazine all at once!

I must also apologise for the lack of listings this time around. I am planning to include them in a proposed 6-weekly leaflet to arrive in between TI*MES, so you won't get bored in between issues!

So, what news do I have this month (November) – well, as some of you may be aware, that stronghold of TI publishing, MICROpendium has changed format to the same size as TI*MES due to mounting publication expenses. Like all TI organisations it is suffering from falling subscriptions, and I would therefore strongly advise you subscribe. Short of the Internet it really is our only source of up-to-date information (and is now published bi-monthly.)

A year's subscription (6 issues) costs $40 airmail or $52 for surface mail. MICROpendium accepts MasterCard (Access), Visa or international money orders. MICROpendium can be contacted via email at **Error! Bookmark not defined.**, telephoned at (US) 512 255-1512 or written to at PO Box 1343, Round Rock, TX 78680, USA.

Plug done, and now to some news. As many of you probably know, I am a keen PC99 enthusiast and am pleased to report the Stage 5 has now gone into Alpha testing.

"Thanks again to the efforts of Dr. Eric Bray of Philadelphia, the developers of PC99 have been granted formal permission by Mr. Lou Phillips, former chief executive officer of Myarc Inc of New Jersey, to use Myarc's copyrighted code stored in ROM in the Myarc MEXP-1 32/128/512K memory expansion card.

In addition, the developers of PC99 have been granted permission to distribute Myarc Extended Basic II. This is disk-based software that extends many of the capabilities of TI Extended Basic. Included with the distribution will be the corresponding manuals for the 512K card and XB II in Adobe Acrobat format.

The PC99 developers, and CaDD Electronics, which distributes PC99, would like to thank publicly Dr. Eric Bray, for acting on our behalf, and Mr. Lou Phillips for granting this permission.

The next release of PC99 will allow for the following memory configurations:

1. No expansion memory
2. TI 32K
3. AMS (128K, 256K, 512K, 1024K)
4. Myarc 512K (with ROM that supports XBII, or ROM that doesn't)

These can be selected through the PC99 configuration program, cfg.exe.

The AMS emulation is out for alpha testing at present. The Myarc card supports: CALL RDTEST, CALL PART, CALL EMDK, and CALL RDDIR. This means you can set up a RAMdisk with, e.g. CALL PART(128,384,0) and then do a CALL RDDIR.

PC99 will also detect that a Myarc RAMdisk is present, and will save the contents in a DOS file, and restore them the next time PC99 runs."

So there you have it, another version in the pipeline with yet more features for PC-bound TI-ers. Speaking of Myarc, I was able to source the circuit diagrams required by Ross from the US (via the mail list I mentioned last time) and learnt some interesting Myarc related news. You may remember that Cecure were now sole Myarc distributors and had even begun to remanufacture certain parts (even the Geneve.) It seems that the TI business has now been transferred to a name familiar to all of us in the TI community. Below is the press release:

"As of October 22, 1997, Myarc software/hardware upgrades, repair, and distribution will be managed by Tim Tesch.

I spoke at length with Don Walden about Myarc's future. After some discussion, he accepted my offer to assume responsibility for all Myarc-related aspects of Cecure Electronic Inc. Don will maintain responsibility for all remaining aspects of CEI.

It may take a few weeks to go through the existing paperwork and equipment, and I am not going to make this my second full-time job. However, I will put forth the effort to complete outstanding requests as well as make way for some new ideas Don and I have for Myarc as well as begin to accept new requests.

I ask that ALL PEOPLE with outstanding Myarc requests (Geneve, HFDCC, Floppy Controller, SOFTWARE, etc.) contact me as soon as possible. Doing so will allow me to estimate the workload for the next few weeks as well as evaluate those requests which may have to be denied in the interest of the customer.

Future plans may include a Myarc-only line which would be answered on the day(s)/evening(s) I am at Cecure. A set time window would be established so as to not interfere with "real" work.

Email correspondence for MYARC questions should go to

"ttesch_myarc@juno.com"

This account is active as of 10/22/97 and will be the focal point for all online inquiries. Snail mail may be sent to my home address until a suitable PO Box can be obtained.

Tim Tesch
1856 Dixie Road
Port Washington, WI 53074

Please DO NOT send equipment to my home address!

Thank you,

Tim"

So any of you with Myarc related queries now know the guy to hassle!

Some of you with Web access may be interested in trying out Bill Gaskill's web page at http://www.gj.net/~lucky7. However, it is under construction, and there is no TImeline on it. He is building a repository of module information and you might want to check it out.

Now onto the slightly thorny issue of the SCSI card. Many of you will know of our co-editor's trials and tribulations in getting this so and so to work and it seems the story has taken a new twist. It appears that Michael Becker in Germany has produced a SCSI card

WHT Dan O'Neill made the following comments in a TI forum:

"These cards are cloned DIRECTLY from the WHT SCSI card (at Michael's admission) and are DIRECTLY IN VIOLATION OF US AND INTERNATIONAL COPYRIGHT LAWS.

I have offered Michael a way out of the legal bind through licensing, but he has not responded to our offer. If he does not withdraw the cards from the US and German markets I may consider litigation. If the license fees are paid, then I would be happy to have him as a LEGAL competitor.

I have evidence, and proof of his admissions, and even a faxed request for support from one of his customers in Germany.

I did not want to have to bring this to the open forum, but my emails go un-answered, and the issue is not resolved.

I would appreciate any support from the community in resolving this problem.

Don O'Neil"

Michael's response is interesting (copied verbatim):

"Fact is:

- Our card was produced because we have bought (and payed!) a lot of cards from Bud Mills. But we got only one and without DSR!

2023 note: Bud Mills was a DISTRIBUTOR of the WHT SCSI card and not a copyright owner. Bud passed away in 2016 but left many satisfied customers.

- After I know Don O'Neill as the source of WHT-cards, I have told him the whole story and asked him for an license-agreement.
- I have an agreement with Don: We have stopped our cards and make only a sell-out of our 20 produced cards.
- The card is not for sale in USA and not for sale in the future! We only legalise our cards inside our community as "WHT-like" cards by paying a license fee to Don and his DSR-author for every card, which is sold (max. 20, actual 10)!
- CECURE has only the right to produce our HSGPL-card, nothing more!
- I want not to steel anybodys work! The reason, because we have made the whole story was described in an earlier email and described above too. For over four years, we have not had any source to buy a legal WHT-card, because we does not know Don
  So we have made our cards ourselves.

I was absolutely perplexed in the early morning today, when the discussion starts! The ASCSI from snug is not on the market!"

So, a very strange development. I hope that by printing both sides of the argument verbatim, any rumours members may have heard can now be well and truly quashed!

Lastly some good news. Plans for FestWest 98 (reported last issue) at the Lubbock TI facility are rolling along, with two more important speakers lined up. The first, Bill Gaskill, will need no introduction as the de facto TI historian and the second, Lee Kitchen was the Manufacturing Engineering Manager. Lee was involved in the Home Computers line from day one and is going to give insights into the development, design, and manufacturing of the TI-99/4, TI-99/4A and TI-99/8. I do hope at least one member of our group is able to justify a 'holiday' to Lubbock for him or herself next February. A full report in the spring issue would be most appreciated!

Anyway, apologies again for the short copy – I ran out of time! I hope you those who went to Sandbach enjoyed it, and in the next issue (Spring) I hope to be able to bring you more news of FestWest 98 and some of those long overdue listings!

Until next issue, or the next pamphlet, Merry Christmas and a Happy New Year!

WOODSTOCK'S CHRISTMAS PRESENT.

by Ray Kazmer

Stephen Shaw last issue, reported on Ray Kazmer who passed away in September. We thought as a tribute, we would print the seasonal listing of "Woodstock's Christmas Present", to show people just what this guy could do.

If you haven't seen this program running before, then get your digits on the keyboard and tap away.

We apologise for the format of the listing, but this was how it came out of Richard's computer.

Cont....

```
100 CALL CLEAR :: CALL CHARSET :: CALL CHAR(91,"0000FF"):: CALL SCREEN(2):: DISP
LAY AT(1,9):"** NOTICE **":TAB(12);"[[[[[[":"  The main character in this"
110 DISPLAY AT(4,1):"program speaks in a languageunknown to humans. Sentenceshe
speaks, though different,will sound identical to you!     [[[[[ [[[[[[[[[    [[[
"
120 DISPLAY AT(9,1):"  Fortunately, any 99/4A CANunderstand him, and will actas
your interpreter.":;:"  Whenever he ""speaks"", yourTI will INSTANTLY translate"
130 DISPLAY AT(24,1):: DISPLAY AT(15,1):"his ""words"" into ""human"" foryou, th
en WINDOW them in, atthe bottom of your screen.":;:"  If you miss reading any of
"
140 DISPLAY AT(20,1):"his ""words"" because he spoketoo fast, DON'T DESPAIR, you
will RUN this program again![[[[     [[[[           [[[[[ (Press ANYTHING to conti
nue)"
150 T=T+1 :: IF T>1 THEN 160 ELSE CALL SOUND(150,1393,0):: CALL SCREEN(4)
160 CALL KEY(0,L,I):: IF I=0 THEN 120 ELSE T=60 :: CALL CLEAR :: CALL SCREEN(2)
170 DISPLAY AT(1,1):"This program ISN'T freeware,     it is simply FREE!":;:"* A
 1987 CHRISTMAS PRESENT *":"to 99'ers, everywhere, from:"
180 DISPLAY AT(6,12):"[[[[[[[[[[":TAB(10);"RAY KAZMER":"     13225 AZORES AVE.,"
:"   SYLMAR, CA. 91342  USA" :: CALL D(75)
190 T=T-(.5):: DISPLAY AT(9,25):"A":;:;:"BUT, if you LIKE my gift, ORlearn a few
 ""tricks"" from it"
200 DISPLAY AT(14,1):"OR if you just want to ""GET INTO A CHRISTMAS SPIRIT"" and
you WANT to send me a coupleof bucks ($2.00 IS OKAY!!) Iwould GRATEFULLY accept
it!"
210 DISPLAY AT(19,7):"[[[[[[[[[[":"Press any key to start....OR":;:" THE PROGRAM
 AUTO-STARTS IN" :: IF T>59 THEN CALL SOUND(150,1393,0):: CALL SCREEN(11)
220 CALL KEY(0,L,I):: IF I<>0 THEN CALL SCREEN(2):: GOTO 230 ELSE IF T<1 THEN CA
LL SCREEN(2):: GOTO 230 ELSE DISPLAY AT(24,5):"EXACTLY ";INT(T);" SECONDS!" :: G
OTO 190
230 CALL CHAR(132,"60909FF020202010080404040506040300000080403F070F1E3C78F060404
08",140,"000000102FCE0F0783C1E0F060202010609F90F04040408102020 20A06020C")
240 CALL CLEAR :: R=13 :: CALL SCREEN(7):: GOSUB 1620 :: CALL CHARPAT(63,A$,39,B
$,46,C$,44,D$,45,Y$)
250 CALL CHAR(34,A$,35,B$,64,C$,42,D$):: CALL MAGNIFY(3):: DISPLAY AT(2,4):"IT I
S CHRISTMAS EVE@@@@" :: R=5 :: C=1175 :: CALL SPRITE(#1,132,11,25,76):: GOSUB 14
00
260 A$="0E03077EFEFFFF7F0101030301010107C852F4B8BEF8ECA290C0C0C0E05848E7" :: B$=
"0E03077EFEFFFF7F013F1F03114B3C06C852F4B8BEF8ECA280FCF8C084E91E2" :: R=7 :: C=13
19
270 CALL SPRITE(#2,140,11,25,106):: GOSUB 1400 :: R=5 :: C=1175 :: CALL SPRITE(#
3,132,11,25,136):: GOSUB 1400
280 C=988 :: CALL SPRITE(#4,140,11,25,166):: GOSUB 1400 :: CALL CHAR(96,"A9FF642
",97,"00000080C3C3EFFF",56,"0000000003C7EFF")
290 DISPLAY AT(7,1):"     GOOD OL# CHARLIE BROWN AND HIS FAMILY* HAVE GONE TOGRA
NDMA#S HOUSE TO SPEND THENIGHT AND OPEN GIFTS IN THE MORNING@"
300 DISPLAY AT(13,1):"(THEY TOOK SNOOPY WITH THEM)" :: F$="0E03077EFEFFFF7F011F0
F0301010107C852F4B8BEF8ECA280FCF8C0E05848E7"
310 CALL CHAR(57,RPT$("F",16),58,"03070F1F1F3F7FFFC0E0F0F8F8FCFEFF",50,"F8F8F8F8
FCFCFCFC",136,"7FC0FFFE5F231F03",98,"00C0F0F0F8FCFEFE")
320 CALL CHAR(51,"FCFEFEFEFEFEFFFF",99,"FEFEFCFCF8F8C",106,"48A289E6FFFFFFFF",11
2,"AFFB766E7E7A6E6E",113,"6C3C3C2C28382838")
330 CALL CHAR(104,"FFFFFFFFFFFFFF00",105,RPT$("F",16),48,"00808080C0C0C0C",49,"E
0E0E0E0F0F0F0F",114,"383830301010101"):: C=1175 :: CALL PATTERN(#1,140):: GOSUB
1400
340 R$="090A0C19270707FF77030103050D193190503098E4E0E0FFEEC080C0A0B0988C" :: S$=
"885030292707070F1F3371E1010101011 10A0C94E4E0E0F0F8CC8E878080808"
350 CALL CHAR(52,"0000000000000003F3F3F3F3F3F3F0000000000000000F8F8F8F8F8F8
F8",121,"E07CE080F860FCC0")
360 R=7 :: C=1319 :: CALL PATTERN(#2,132):: GOSUB 1400 :: R=5 :: C=1175 :: CALL
PATTERN(#3,140):: GOSUB 1400 :: C=988 :: CALL PATTERN(#4,132):: GOSUB 1400
370 CALL CHAR(122,"031F060F7F073E07",137,"FE03FF3BFA84F80040C0403C0203023C000000
000000F0F8")
380 CALL CHAR(120,"7FAADF",123,"E850E",128,"0102050D0A122428",129,"030386643C0C0
8F8",60,"E0B0D8EC75FBFF791D3971FF7F01010103050B17AFDEFF9F8E8080FCFC80808")
390 P$="2154014411A40144201242142 10A41104412241042 2401A410A40194401441" :: O$="0
70F1F1F1F1F0F07070F1F3F3F3F1F0FF8FC5E5E5EFEFCF8F8FCFEFFFFFFFFEFC"
400 DISPLAY AT(16,1):"    BUT SNOOPY DIDN#T FORGETABOUT HIS FRIEND* WOODSTOCK*AN
D HAS LEFT A GIFT FOR HIM@"
410 G$="04789721031F3F017FFFFFFE7E070931603CD288C0F8FC80A2ECF8BEB8F45248" :: H$=
"0478972103030301 7FFFFFFE7E070931603CD288C0C0C080A2ECF8BEB8F45248"
420 J$="0020405060C0AE3F3F2EA0406050402000925479 3EFC67FCE7FCFCFAF9F9F87" :: K$="
0E1F9F9F5F3F3FE73FE63F7C9E2A490004024A66626574FCFC7563666A4204"
430 L$="0E1F9F9F5F3F3FE73FE63F7C9E2A490004020A06020574FCFC7503060A0204" :: M$="1
24A2F1D7D8F3745013F1F03114B3C068C90E07E7FFFFFFE80FCF8C084E91E2"
440 N$="124A2F1D7D8F3745010303031 14B3C068C90E07E7FFFFFFE80C0C0C084E91E2" :: I$="
0020425666C6AE3F3F2EA64666524020009254793EFC67FCE7FCFCFAF9F9F87"
450 CALL CHAR(130,"070C9366040C12E3",115,"685878505010101"):: C=1760 :: CALL PAT
TERN(#1,132):: GOSUB 1400 :: R=5 :: CALL PATTERN(#2,140):: GOSUB 1400 :: C=1480
460 CALL PATTERN(#3,132):: GOSUB 1400 :: GOSUB 1430 :: CALL D(200):: DISPLAY AT(
21,3):"THERE#S JUST ONE@@@@TINY":;:"   @@@LITTLE@@@@PROBLEM@@@@"
470 C=1568 :: CALL PATTERN(#4,140):: GOSUB 1400 :: R=7 :: CALL PATTERN(#1,140)::
 GOSUB 1400 :: GOSUB 1430 :: R=3 :: C=1175 :: CALL PATTERN(#2,132):: GOSUB 1400
480 T$="E0B0D8EC75FBFF791E060C181818100003050B17AFDEFF9E0C" :: C$="3C7F7F7F3F0F0
703013F1F03010101073C20FEB978FCFAF290FCF8C0E05040E"
490 U$="CA2A9EFB18789C0F033B1F032197790C3878F0E0C0C0C0C0700005C5E5FFE202" :: CAL
L CHAR(107,"0B1B17376F6E7E3C"):: Q$=RPT$("0",26)&"3F3F3F"&RPT$("0",26)&"FCFCFC"
500 V$="E3B29A9EC0B777F7000000000606601F81C264797AACDEDB191F1F0E0E0C889" :: W$="
3777F7F794948080000 0C000000000000080C0C0C0C0C0C0000000C"
510 X$="3C047F9D1E3F5F4F093F1F03070A02073CFEFEFCF8F0E0C010FEF8C0808080E" :: E$="
3C7F7F7F3F0F0703010103030101 01073C20FEB978FCFAF290D0C0C0E05848E7"
520 CALL D(400):: GOSUB 1590 :: CALL DELSPRITE(ALL):: CALL CLEAR :: CALL SCREEN(
2):: R=2 :: GOSUB 1620
530 CALL CHAR(132,"FF00FFBFD9E7FFFF1F0000000000000 0FF00FFC3FFFF1FFFF1",140,"0000
000000D0D070F0D0D07060202020F8B8B8F8D0D0D07060602020202")
540 DISPLAY AT(12,4):"AND HERE HE COMES NOW!" :: CALL SOUND(700,110,0):: CALL SO
UND(200,123,0):: CALL SOUND(425,131,0):: CALL SOUND(300,110,0)
550 CALL COLOR(3,2,2,4,2,2):: CALL HCHAR(22,8,56,25):: CALL HCHAR(3,24,130):: FO
R I=13 TO 21 STEP 2 :: CALL HCHAR(I,1,104,7)
560 CALL HCHAR(I+1,1,105,7):: NEXT I :: CALL HCHAR(22,1,96,7):: CALL HCHAR(22,8,
59):: CALL HCHAR(5,32,123):: CALL HCHAR(4,32,128)
570 CALL HCHAR(18,1,106,7):: CALL HCHAR(12,1,97,5):: CALL HCHAR(13,8,49):: CALL
HCHAR(14,8,50):: CALL HCHAR(14,7,98):: CALL HCHAR(15,8,51)
580 CALL HCHAR(16,8,98):: CALL HCHAR(16,9,120):: CALL HCHAR(17,9,49):: CALL HCHA
R(17,8,99):: CALL HCHAR(13,7,99):: CALL HCHAR(15,7,99)
590 CALL HCHAR(20,8,114):: CALL HCHAR(18,9,114):: CALL HCHAR(16,9,48):: CALL HCH
AR(6,31,115):: CALL HCHAR(23,1,57,64):: CALL HCHAR(22,25,57)
600 CALL HCHAR(22,26,59):: CALL HCHAR(22,27,58):: CALL HCHAR(22,30,59):: CALL HC
HAR(21,25,56):: CALL HCHAR(19,8,113):: CALL HCHAR(22,24,58)
610 CALL VCHAR(1,28,121,21):: CALL VCHAR(1,29,122,21):: CALL HCHAR(22,28,96,2)::
 CALL HCHAR(8,21,120,4):: CALL HCHAR(8,25,123)
620 CALL HCHAR(7,25,128):: CALL HCHAR(7,26,123):: CALL HCHAR(6,26,128):: CALL HC
HAR(6,27,123):: CALL HCHAR(5,27,128):: CALL HCHAR(8,20,130)
630 CALL HCHAR(8,23,129):: CALL HCHAR(10,27,129):: CALL HCHAR(1,26,130):: CALL H
CHAR(2,25,129):: CALL HCHAR(1,27,120):: CALL HCHAR(3,25,114)
640 CALL HCHAR(11,27,115):: CALL HCHAR(6,30,120):: CALL HCHAR(6,31,123):: CALL H
CHAR(5,31,128):: CALL HCHAR(7,21,136):: CALL HCHAR(7,24,137)
```

```
650 CALL HCHAR(18,8,112):: CALL
SPRITE(#1,132,1,49,169,#2,60,1,73,42,#4,52,1,73,
43,#5,140,1,60,155)
660 DISPLAY AT(12,4)SIZE(22):"ab0" :: CALL SCREEN(5):: CALL COLOR(3,16,2,4,16,2,
9,9,16,10,9,2,11,8,2,12,7,2,13,7,2,14,11,2)
670 CALL COLOR(#1,11,#2,14,#4,4,#5,8):: C,L,T=1 :: R=20 :: GOSUB 1340 :: L=0
680 MA$=" LOOKOUT! LOOKOUT! GANGWAY!" :: L=2 :: GOSUB 1340 :: CALL SOUND(10,-5,6
):: GOSUB 1430 :: CALL SOUND(10,-5,6):: CALL SOUND(10,-5,6):: GOSUB 1430 :: CALL
 D(100)
690 CALL SOUND(10,-5,3):: GOSUB 1430 :: CALL SOUND(10,-5,3):: T=T+1 :: IF T<12 T
HEN 690 ELSE GOSUB 1390 :: GOSUB 1390
700 GOSUB 1390 :: T=1 :: GOSUB 1380 :: CALL SPRITE(#3,36,12,R,C,0,5):: I=4 :: GO
SUB 1330 :: T=I :: GOSUB 1390 :: CALL PATTERN(#3,44)
710 IF T=10 OR T=12 THEN GOSUB 1450 ELSE IF T=9 OR T=11 THEN GOSUB 1460
720 CALL PATTERN(#3,36):: GOSUB 1390 :: CALL PATTERN(#3,44):: T=T+1 :: IF T<12 T
HEN 710 ELSE GOSUB 1450 :: CALL MOTION(#3,4,3):: GOSUB 1390
730 CALL PATTERN(#3,116):: GOSUB 1390 :: CALL PATTERN(#3,108):: GOSUB 1390 :: T=
T+1 :: IF T<18 THEN 730 ELSE CALL MOTION(#3,-4,4):: GOSUB 1390 :: CALL PATTERN(#
3,92):: GOSUB 1390 :: T=1
740 CALL PATTERN(#3,100):: GOSUB 1390 :: CALL PATTERN(#3,92):: GOSUB 1390 :: T=T
+1 :: IF T<10 THEN 740 ELSE CALL MOTION(#3,0,8):: GOSUB 1390 :: T=0 :: I=2 :: GO
SUB 1330
750 CALL SPRITE(#3,36,12,12,174,0,0):: GOSUB 1710 :: CALL LOCATE(#3,12,168):: CA
LL D(250):: CALL PATTERN(#3,44):: GOSUB 1390 :: CALL CHAR(36,A$,92,F$,108,R$)
760 T=1 :: CALL SPRITE(#3,108,12,40,169):: CALL SOUND(200,-7,0):: CALL MOTION(#5
,23,0):: CALL D(340):: CALL DELSPRITE(#5):: CALL SOUND(10,-5,6)
770 CALL CHAR(140,"82542854285482"&RPT$("0",50),44,"0C275F0D1D3F7FFFFD7903030101
0107E080F0C8C0C0A09080C0C0C0E05848E7")
780 CALL CHAR(116,K$,40,G$):: CALL SPRITE(#3,36,12,38,167):: CALL SOUND(100,440,
5):: CALL D(300):: MA$="      A PERFECT LANDING!" :: GOSUB 1340
790 CALL SPRITE(#3,92,12,36,167):: CALL D(300):: GOSUB 1390 :: CALL SPRITE(#3,36
,12,33,155):: CALL D(330):: CALL PATTERN(#3,44):: CALL D(250):: CALL PATTERN(#3,
36)
800 MA$="  SNOOPY LEFT ME A PRESENT!" :: GOSUB 1340
810 CALL PATTERN(#3,92):: MA$=" WHAT A SURPRISE! I#LL BRING" :: MB$="  IT UP HER
E AND OPEN IT!" :: GOSUB 1350 :: T=1 :: CALL CHAR(36,H$,44,I$):: A=4 :: I=48
820 CALL SOUND(20,-6,4):: CALL SPRITE(#3,36,12,30,151,4,-1):: CALL SOUND(10,-5,5
):: GOSUB 1720 :: GOSUB 1390 :: CALL PATTERN(#3,40):: GOSUB 1390
830 CALL PATTERN(#3,36):: GOSUB 1390 :: CALL PATTERN(#3,40):: GOSUB 1390 :: T=T+
1 :: IF T<12 THEN 830 ELSE GOSUB 1480 :: CALL MOTION(#3,0,-3):: GOSUB 1390
840 CALL PATTERN(#3,44):: GOSUB 1390 :: CALL PATTERN(#3,124):: GOSUB 1390 :: CAL
L PATTERN(#3,116):: GOSUB 1390 :: CALL PATTERN(#3,36):: GOSUB 1390 :: T=T+1 :: I
F T<20 THEN 840 ELSE CALL MOTION(#3,-6,0)
850 CALL PATTERN(#3,124):: GOSUB 1390 :: CALL PATTERN(#3,92):: GOSUB 1390 :: T=T
+1 :: IF T<24 THEN 850 ELSE CALL POSITION(#3,R,C):: GOSUB 1490
860 CALL PATTERN(#3,44):: CALL D(250):: CALL PATTERN(#3,36):: A=4 :: GOSUB 1480
:: MA$="WHAT THE HECK SHOT ME DOWN""!" :: GOSUB 1340 :: CALL CHAR(92,F$):: CALL
PATTERN(#3,44)
870 CALL D(150):: MA$="OH NO!! A KILLER SNOWFLAKE!!" :: MB$="         [ [[[[[[ [[
[[[[[[[" :: GOSUB 1350
880 CALL PATTERN(#3,92):: MA$=" THE GREATEST TERROR A NIGHT" :: MB$="CAN HOLD FO
R A GUY LIKE ME!" :: GOSUB 1350 :: CALL PATTERN(#3,44)
890 CALL INIT :: CALL LOAD(-31878,18):: A=8 :: I=97 :: GOSUB 1470 :: I=218 :: GO
SUB 1470 :: I=213 :: GOSUB 1470 :: I=41 :: GOSUB 1470 :: I=35 :: GOSUB 1470
900 I=215 :: GOSUB 1470 :: CALL PATTERN(#3,36):: I=54 :: GOSUB 1470 :: I=209 ::
GOSUB 1470 :: CALL PATTERN(#3,44):: I=228 :: GOSUB 1470 :: I=68 :: GOSUB 1470
910 CALL PATTERN(#3,92):: MA$="I HAVE TO TAKE IT TO MY NEST" :: MB$="THRU THIS R
AGING BLIZZARD""!" :: GOSUB 1350 :: CALL DELSPRITE(#5)
920 CALL PATTERN(#3,44):: CALL D(300):: CALL PATTERN(#3,36):: MA$="I#M NO CHICKE
```

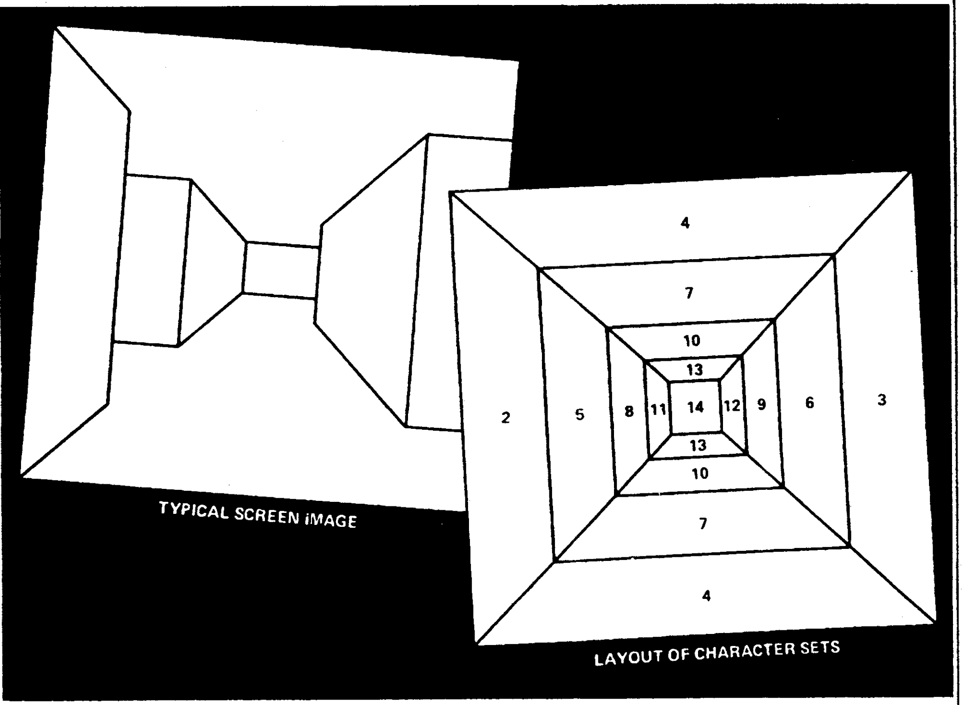
```
N SO HERE I GO!" :: MB$="        [[[[[[[" :: I,L=2 :: GOSUB 1350 :: T=0
930 R=123 :: C=159 :: CALL PATTERN(#3,92):: GOSUB 1600 :: GOSUB 1380 :: CALL MOT
ION(#3,-4,-7):: GOSUB 1330 :: GOSUB 1490 :: MA$="          . RATS!!" :: L=2 :: GO
SUB 1340 :: CALL D(250):: GOSUB 1380 :: CALL PATTERN(#3,44)
940 MA$="HERE COMES A LAUNCH WINDOW!!" :: GOSUB 1340 :: CALL PATTERN(#3,36)
950 MA$="    AND I AM OUTTA HERE!" :: MB$="             [[[[[" :: L=2 :: GOSUB 1
350 :: CALL PATTERN(#3,36):: GOSUB 1600 :: GOSUB 1380 :: T=0 :: I=4 :: CALL MOTI
ON(#3,-4,-4):: GOSUB 1330 :: I=5
960 CALL PATTERN(#3,36):: GOSUB 1390 :: CALL PATTERN(#3,44):: GOSUB 1390 :: T=T+
1 :: IF T<I THEN 960 ELSE T=0 :: GOSUB 1330
970 CALL MOTION(#3,0,-5):: I=1 :: GOSUB 1330 :: CALL SPRITE(#3,116,12,9,39,0,0):
: GOSUB 1390 :: CALL PATTERN(#3,92)
980 GOSUB 1390 :: CALL PATTERN(#3,36):: GOSUB 1390 :: CALL PATTERN(#3,44):: GOSU
B 1390 :: CALL D(200):: CALL BOUNCE(127):: CALL BOUNCE(85)
990 CALL BOUNCE(45):: CALL SPRITE(#3,108,12,64,42,0,0):: CALL SOUND(100,-7,0)::
CALL D(300):: GOSUB 1720 :: CALL SPRITE(#3,92,12,62,42)
1000 CALL SOUND(100,440,0):: CALL D(300):: MA$="ANOTHER PERFECT LANDING! THE" ::
 MB$="RED BARON#S GREEN WITH ENVY!" :: GOSUB 1350 :: CALL CHAR(36,E$,44,C$,40,H$
):: T=0
1010 CALL PATTERN(#3,36):: MA$="           EEEEK!" :: L=2 :: GOSUB 1340 :: CALL
POSITION(#3,R,C):: CALL D(150):: GOSUB 1380 :: GOSUB 1410 :: GOSUB 1420
1020 R=R-3 :: GOSUB 1390 :: CALL PATTERN(#3,36):: GOSUB 1390 :: CALL PATTERN(#3,
44):: GOSUB 1440 :: T=T+1 :: IF T<10 THEN 1020 ELSE T=0
1030 R=R-3 :: C=C+3 :: GOSUB 1390 :: CALL PATTERN(#3,36):: GOSUB 1390 :: CALL PA
TTERN(#3,44):: GOSUB 1440 :: T=T+1 :: IF T<6 THEN 1030
1040 R=R-3 :: GOSUB 1410 :: GOSUB 1420 :: GOSUB 1410 :: GOSUB 1420 :: CALL LOCAT
E(#3,11,176):: GOSUB 1710
1050 CALL SPRITE(#3,124,12,11,171):: CALL SOUND(200,-6,6):: CALL CHAR(60,T$,40,Q
$):: CALL SOUND(125,-6,0):: CALL SPRITE(#4,40,4,33,173)
1060 CALL SOUND(200,-7,0):: CALL CHAR(52,Q$):: CALL PATTERN(#4,52):: CALL CHAR(4
0,H$)
1070 CALL LOCATE(#2,37,176):: CALL SOUND(50,-5,0):: CALL PATTERN(#3,40):: CALL D
(300):: CALL LOCATE(#3,25,172):: GOSUB 1710 :: CALL LOCATE(#3,10,163)
1080 CALL D(300):: CALL SPRITE(#3,108,12,40,159):: CALL SOUND(200,-7,0):: CALL D
(400):: CALL CHAR(124,N$):: CALL SPRITE(#3,124,12,38,160)
1090 CALL SOUND(100,440,0):: CALL D(200):: MA$="ASK ANY KITE FLYER@@@@@TREES" ::
 MB$="  ARE TREACHEROUS BEASTS!!" :: GOSUB 1350
1100 GOSUB 1690 :: CALL D(200):: CALL LOCATE(#3,34,164):: CALL SOUND(-100,-5,9):
: CALL LOCATE(#2,38,166):: CALL SOUND(-100,-5,7)
1110 CALL LOCATE(#3,30,150,#2,38,160):: CALL SOUND(-100,-5,5):: CALL COLOR(#4,7)
:: CALL LOCATE(#2,50,145):: CALL SOUND(-100,-5,0)
1120 CALL SPRITE(#8,96,4,55,150):: CALL SOUND(200,-5,0):: CALL LOCATE(#3,37,160)
:: A=2
1130 CALL D(250):: GOSUB 1700 :: MA$="  I AM THE WORLD#S FASTEST" :: MB$="CHRIST
MAS PRESENT UNWRAPPER!" :: GOSUB 1350 :: GOSUB 1690
1140 CALL D(250):: GOSUB 1700 :: MA$="A METAL BOX@@@@WITH A HANDLE" :: MB$=" STI
CKING OUT OF IT#S SIDE!" :: GOSUB 1350 :: GOSUB 1690 :: CALL D(200):: GOSUB 1700
1150 MA$="  I#LL GIVE IT A CRANK@@@@" :: GOSUB 1340 :: GOSUB 1690 :: GOSUB 1640
:: GOSUB 1700 :: MA$="WHATAYA KNOW!@@@A MUSIC BOX!" :: GOSUB 1350
1160 GOSUB 1690 :: CALL CHAR(108,"000000000F7C1C1000000D52A000000000000000008080
0000")
1170 C=1109 :: GOSUB 1650 :: C=1175 :: GOSUB 1650 :: GOSUB 1650 :: C=1319 :: GOS
UB 1650 :: GOSUB 1650 :: C=1568 :: GOSUB 1650 :: C=1175 :: GOSUB 1650 :: GOSUB 1
680 :: A=2 :: CALL D(200):: GOSUB 1640 :: CALL CHAR(100,V$,36,X$):: C=1865 :: GO
SUB 1650
1180 CALL SOUND(-35,-5,0):: CALL CHAR(54,"07070707070707070707070707FCFCFC"):: C
ALL HCHAR(6,23,138)
```

```
1190 CALL HCHAR(5,23,138):: CALL HCHAR(4,23,138):: CALL SPRITE(#3,36,12,37,160)
1200 FOR I=1 TO 30 STEP 5 :: CALL SPRITE(#6,108,16,5,170,#7,100,10,15,169)
1210 CALL SOUND(-1,550,I,555,I):: CALL LOCATE(#6,20,170,#7,5,169):: CALL SOUND(-
1,410,I,415,I):: NEXT I :: CALL LOCATE(#6,15,170,#7,10,169)
1220 CALL HCHAR(3,22,139,2):: CALL D(100):: CALL CHAR(40,W$,100,"E3B29A9EC080000
0"):: MA$="  EEEEEEEEEEEEEEEEEEEEEEEK!"
1230 CALL PATTERN(#6,40):: CALL HCHAR(4,22,107):: CALL SOUND(800,-3,0):: CALL CH
AR(108,R$):: CALL D(100)
1240 R=33 :: C=126 :: CALL CHAR(64,Y$,116,U$):: MB$="  [[[[[[[[[[[[[[[[[[[[[[["
:: CALL SPRITE(#3,116,12,R,C)
1250 L=2 :: FOR I=1 TO 30 :: CALL SOUND(30,2300,0):: NEXT I :: GOSUB 1360 :: GOS
UB 1500
1260 GOSUB 1630 :: MA$="I WAS WRONG ABOUT SNOWFLAKES" :: GOSUB 1340 :: GOSUB 163
0 :: MA$="A GARFIELD@IN@A@BOX IS A LOT" :: MB$="MORE SCARY TO A GUY LIKE ME!" ::
 GOSUB 1350
1270 R=5 :: C=440 :: DISPLAY AT(11,9)SIZE(16):"MERRY CHRISTMAS" :: GOSUB 1400 ::
 CALL PATTERN(#3,44):: CALL D(50):: C=523 :: DISPLAY AT(13,11)SIZE(11):"EVERYBOD
Y!!"
1280 GOSUB 1400 :: CALL D(100):: C=698 :: R=3 :: DISPLAY AT(16,9)SIZE(15):"FROM
RAY KAZMER" :: GOSUB 1400 :: CALL PATTERN(#3,36):: GOSUB 1630
1290 CALL CHAR(124,B$):: R=150 :: C=188 :: L=1 :: MA$=" AND A VERY HAPPY NEW YEA
R!" :: MB$="      [[[[ [[[[[" :: FOR I=1 TO 3
1300 CALL LOCATE(#3,R-6,C):: CALL PATTERN(#3,124):: GOSUB 1350 :: CALL LOCATE(#3
,R,C):: CALL PATTERN(#3,36):: NEXT I :: L=2 :: GOSUB 1360
1310 CALL D(500):: R=6 :: C=2217 :: GOSUB 1400 :: R=8 :: C=1661 :: GOSUB 1400 ::
 C=1397 :: GOSUB 1400 :: R=5 :: C=1661 :: GOSUB 1400 :: R=8 :: C=1480 :: GOSUB 1
400 :: R=5 :: C=1245 :: GOSUB 1400 :: R=2 :: C=1109 :: GOSUB 1400 :: CALL D(900)
1320 CALL LOAD(-31804,0,36)
1330 CALL PATTERN(#3,124):: GOSUB 1390 :: CALL PATTERN(#3,116):: GOSUB 1390 :: C
ALL PATTERN(#3,100):: GOSUB 1390 :: CALL PATTERN(#3,36):: GOSUB 1390 :: T=T+1 ::
 IF T<I THEN 1330 ELSE T=0 :: RETURN
1340 MB$=RPT$(" ",28)
1350 CALL SOUND(50,2300,0):: GOSUB 1430 :: CALL SOUND(50,1700,0):: GOSUB 1430 ::
 CALL SOUND(100,2300,0):: CALL SOUND(50,1950,0):: IF L=1 THEN RETURN
1360 DISPLAY AT(24,1):MA$ :: DISPLAY AT(23,1):MA$:MB$ :: IF L=2 THEN RETURN ELSE
 CALL D(600)
1370 IF MB$<>" " THEN CALL D(250)
1380 DISPLAY AT(23,1):RPT$("9",28);MA$ :: CALL HCHAR(24,3,57,28):: L=0 :: RETURN
1390 CALL SOUND(10,-5,0):: GOSUB 1430 :: CALL SOUND(10,-5,0):: RETURN
1400 FOR I=0 TO 28 STEP R :: CALL SOUND(-50,C,I,C*3,I,C*6,I):: CALL SOUND(-200,C
,I+2,C*3,I+2,C*6,I+2):: CALL D(1):: NEXT I :: RETURN
1410 C=C+3 :: GOSUB 1390 :: CALL PATTERN(#3,124):: GOSUB 1390 :: CALL PATTERN(#3
,92):: GOSUB 1440 :: T=T+1 :: IF T<7 THEN 1410 ELSE T=0 :: RETURN
1420 C=C+3 :: GOSUB 1390 :: CALL PATTERN(#3,36):: GOSUB 1390 :: CALL PATTERN(#3,
44):: GOSUB 1440 :: T=T+1 :: IF T<8 THEN 1420 ELSE T=0 :: RETURN
1430 CALL SOUND(-1,4E4,30):: RETURN
1440 CALL LOCATE(#3,R,C,#2,R+8,C,#4,R+8,C):: RETURN
1450 CALL POSITION(#3,R,C):: CALL SPRITE(#3,36,12,R-8,C,0,4):: GOTO 1390
1460 CALL POSITION(#3,R,C):: CALL SPRITE(#3,36,12,R+8,C,0,4):: GOTO 1390
1470 CALL D(210)
1480 A=A+1 :: CALL SPRITE(#A,140,16,240,I,3,2):: RETURN
1490 CALL SOUND(100,-7,0):: CALL DELSPRITE(#A)
1500 CALL SPRITE(#3,108,12,R,C,0,0):: CALL CHAR(36,A$,44,E$,92,O$,116,S$,124,P$)
:: IF L=0 THEN CALL CHAR(40,B$)ELSE IF L=2 THEN CALL D(50):: GOSUB 1380
1510 CALL SPRITE(#3,116,12,R,C):: CALL SOUND(-50,2500-(R*10),0):: CALL PATTERN(#
3,108):: R=R+8 :: IF R<168 THEN 1510
1520 R=158 :: CALL SPRITE(#3,124,16,R,C,8,0):: CALL SOUND(50,-6,0):: CALL D(175)
:: CALL DELSPRITE(#3)
```

```
1530 CALL D(300):: CALL SPRITE(#3,92,16,163,C):: CALL SOUND(100,440,5):: CALL D(
300):: R=159
1540 CALL LOCATE(#3,R,C):: CALL SOUND(20,-5,4):: C=C+4 :: GOSUB 1430 :: IF C<188
 THEN 1540 ELSE IF C>188 THEN C=188
1550 CALL LOCATE(#3,R,C):: CALL SOUND(20,-5,4):: R=R-2 :: GOSUB 1430 :: IF R>150
 THEN 1550 ELSE IF R<150 THEN R=150
1560 T=1 :: R=145
1570 CALL LOCATE(#3,150,192):: CALL SOUND(-1,-7,9):: CALL SPRITE(#5,124,16,R,200
):: R=R+5 :: CALL LOCATE(#3,150,184,#5,R,170):: T=T+1 :: IF T<6 THEN 1570
1580 CALL DELSPRITE(#5):: CALL SPRITE(#3,36,12,150,188):: RETURN
1590 CALL CHAR(100,J$,108,K$):: GOTO 1610
1600 CALL CHAR(40,F$):: CALL PATTERN(#3,40)
1610 CALL CHAR(36,G$,44,H$,92,I$,116,L$,124,M$):: RETURN
1620 FOR I=1 TO 8 :: CALL COLOR(I,16,R):: NEXT I :: RETURN
1630 CALL D(200):: CALL PATTERN(#3,44):: CALL D(200):: CALL PATTERN(#3,36):: RET
URN
1640 C=1175 :: GOSUB 1650 :: GOSUB 1650 :: C=1319 :: GOSUB 1650 :: GOSUB 1650 ::
 C=1568 :: GOSUB 1650 :: C=1865 :: GOSUB 1650 :: C=1568 :: GOSUB 1650 :: C=1175
:: GOSUB 1650 :: CALL D(200):: RETURN
1650 A=3-A :: ON A GOSUB 1670,1680
1660 FOR R=0 TO 20 STEP 10 :: CALL SOUND(-15,C,R):: NEXT R :: GOSUB 1430 :: CALL
 D(50):: RETURN
1670 CALL LOCATE(#3,39,166):: RETURN
1680 CALL LOCATE(#3,37,164):: RETURN
1690 CALL CHAR(124,"124A2F1F7F8F3745"):: RETURN
1700 CALL CHAR(124,"124A2F1D7D8F3745"):: RETURN
1710 CALL SOUND(130,110,0,-4,0):: RETURN
1720 CALL CHAR(92,N$):: RETURN
1730 SUB BOUNCE(R):: CALL LOCATE(#3,60,42):: CALL MOTION(#3,-R,0):: GOSUB 1710 :
: CALL MOTION(#3,0,0):: CALL D(70):: CALL MOTION(#3,R-20,0):: SUBEND
1740 SUB D(T):: FOR L=1 TO T*1.2 :: NEXT L :: SUBEND
```

IN A MAZE

G. Anderson's adaptable program will a-maze you for hours.



TYPICAL SCREEN IMAGE

LAYOUT OF CHARACTER SETS

This game has been coded in TI Extended Basic and runs on the TI99/4A in just under 16K. The game is easy, all the player has to do is escape from a simple maze, sounds easy?

When the program is run initially a simple set of instructions on the use of the keyboard control is displayed and when the player is ready he can continue by pressing any key. The program then scans the screen to read the maze and on completion, the maze will be displayed showing the exits and the player's position.

After a short delay the screen will clear and the game can begin. A three dimensional image is then shown of your position. You will see that the maze is not so simple! If the player becomes hopelessly lost he can use the help option by pressing the H key. This will redisplay the maze with the player's position. However it will also add 10 steps to your score.

In theory it's possible to emerge from the maze in under 16 steps, but if you get that good then why not change the layout of the maze and see just how difficult it can become (modify lines 840-1080). As a tip, the direction in which the player faces is towards the top of the screen at the start of the game. If this becomes too easy then change line 630 to DIR = INT(RND * 4) + 1 and this will return a random direction. This game shows a method of creating a maze game on any micro. This method simply uses a two dimensional matrix (array) in which values of 0 and 1 represent pathways. All that any program has to do is to scan the area around the player to detect the presence of walls and pathways. All that is then required is to add the desired graphics.

This game uses many TI graphics commands, which like most micros are unique, and could

IN A MAZE

cause problems when converting the code to run on other computers. It may be easier to redesign your own graphics than to convert these used here.

Before I explain the commands used it may be wise to explain the method used to draw the corridors. Firstly the TI99/4A is said to use a very slow form of BASIC so the problem was how to redraw the corridor quickly. This was overcome on the Texas by drawing the corridor once only and then simply changing the colour sets used for each of the three elements of the corridor i.e. walls and floors. Using this method of patterning, which is very simple on the Texas (CALL COLOR), the speed of execution can be brought up to an acceptable level. It should also be said for those trying to convert this game to TI Basic that the sprites used are there only to clip the doors to give a clearer image and could be safely omitted.

Apart from the graphics commands the basic syntax of the game can remain the same on any micro. Listed here are some of the non-standard commands used within this program.

List of Variables

S	Character set number.
MAZE$(.)	Matrix to store corridor locations.
DIR	Direction of player, 1 = up, 2 = right, 3 = down, 4 = left.
YP,XP	Vertical and horizontal position of player.
ST	Number of steps taken.
L	Length of corridor in front of player.
CF	Presence of wall flag.
FO	Presence of exit flag.

HOW IT RUNS

100-360	Introduction and instructions.
370	Randomize.
380-420	Character codes used.
470-480	Set up maze.
490-560	Scan maze and store values in matrix.
570-590	Locate player starting position.
600	Display map and player.
610-640	Draw and paint corridor.
650-730	Scan keyboard for instructions.
740-800	Check for legal move, exit and execute instruction.
810-830	Help routine (display map and player).
840-1080	Draw map subroutine.
1090-1510	Subroutine to colour corridor.
1520-1870	Subroutine to draw corridor.
1880-1940	Subroutine to display player's position.
1950-2010	Program termination.

HINTS ON CONVERSION

CALL CLEAR	Clears entire screen.
CALL CHAR(X,A$)	Redefines characters. X represents the character code. A$ defining code in hexdecimal.
CALL SCREEN(X)	Paints screen to desired colour (2 represents black).
CALL KEY(Ø,K,S)	Detects keyboard input. K is the ASCII code of key depressed. S is the keyboard status Ø if no key is depressed.
CALL COLOUR(A,B,C)	Defines the foreground and background colours of the desired colour set. A is the character set and B,C are the foreground and background colours.
CALL SPRITE	Creates and positions the sprite.

(X,A,B,C,D)	X = Sprite number 1 to 28.
	A = Character code number of pattern required.
	B = Foreground colour of sprite.
	C = Vertical position 1 to 194.
	D = Horizontal position 1 to 256.

CALL DELSPRITE(ALL)	Deletes all sprites from screen.
CALL CHARSET	Resets all characters to their standard shape.

The code has been written so that the maze defined in lines 840 to 1080 can be changed by the player. This means that almost any maze can be constructed and the program will still run. However, the exits shown as a ' (grave Character code 96) symbol should be left in the same positions and no corridor should be longer than 4 steps.

PROGRAM LISTING

```
100 REM ********************
110 REM ****            ****
120 REM **** IN A MAZE  ****
130 REM ****            ****
140 REM ********************
150 REM *** G ANDERSON   ***
160 REM ***  5-2-82      ***
170 REM ********************
180 CALL CLEAR
190 DISPLAY AT(5,9):"I N  A  M A
 Z E"
200 DISPLAY AT(6,9):"===========
===="
210 DISPLAY AT(8,1):"THE OBJECT
OF THIS GAME IS"
220 DISPLAY AT(9,1):"TO ESCAPE F
ROM A MAZE IN THE"
230 DISPLAY AT(10,1):"LEAST POSS
IBLE NUMBER OF "
240 DISPLAY AT(11,1):"MOVES. THE
 DIFFERANCE IS "
250 DISPLAY AT(12,1):"THAT YOU A
RE IN THE MAZE."
260 DISPLAY AT(14,1):"TO MOVE FO
RWARD PRESS E."
270 DISPLAY AT(15,1):"TO TURN US
```

```
E THE ARROW KEYS."
280 DISPLAY AT(16,1):"THE H KEY
CAN BE USED FOR "
290 DISPLAY AT(17,1):"HELP BY SH
OWING THE MAZE "
300 DISPLAY AT(18,1):"AND YOUR P
OSITION."
310 DISPLAY AT(19,1):"HOWEVER, I
T WILL ADD 10"
320 DISPLAY AT(20,1):"STEP TO YO
UR SCORE."
330 DISPLAY AT(22,1):"THE DOORS
ARE SHOWN IN "
340 DISPLAY AT(23,1):"GREEN & TH
E EXIT IN BLACK"
350 DISPLAY AT(24,1):"  PRESS AN
Y KEY TO START"
360 CALL KEY(0,K,S):: IF S=0 THE
N 360
370 CALL CLEAR :: RANDOMIZE
380 C$(1)="FFFFFFFFFFFFFFFF"
390 C$(2)="80C0E0F0F8FCFEFF"
400 C$(3)="0103070F1F3F7FFF"
410 C$(4)="FF7F3F1F0F070301"
420 C$(5)="FFFEFCF8F0E0C080"
430 REM ****LOAD CHARS*****
440 FOR S=2 TO 14 :: FOR T=1 TO
5
450 CALL CHAR(23+8*S+T,C$(T))
460 NEXT T :: NEXT S
470 CALL SCREEN(2)
480 GOSUB 840
490 REM ****LOAD ARREY*****
500 DIM MAZE$(21,29)
510 FOR Y=1 TO 21 :: FOR X=0 TO
29
520 CALL GCHAR(Y,X+2,D)
530 IF D=72 OR D=96 THEN MAZE$(Y
,X)="1" ELSE MAZE$(Y,X)="0"
540 NEXT X :: NEXT Y
550 FOR Y=0 TO 21 :: MAZE$(Y,0)=
"0" :: NEXT Y
560 FOR X=0 TO 29 :: MAZE$(0,X)=
"0" :: NEXT X
570 REM FIND POSITION OF PLAYER
AT RANDOM
580 YP=INT(RND*20)+1 :: XP=INT(R
ND*6)+11
590 IF MAZE$(YP,XP)="0" THEN 580
600 CALL COLOR(6,14,1):: GOSUB 8
40
610 CALL CLEAR :: CALL COLOR(6,1
,1,7,1,1)
620 GOSUB 1520
630 DIR=1
640 GOSUB 1090
650 REM *******START*******
660 CALL KEY(0,K,S):: IF S=0 THE
N 660
670 IF K=72 THEN 810
680 IF K=69 THEN 740
690 IF K=83 THEN DIR=DIR-1
700 IF K=68 THEN DIR=DIR+1
710 IF DIR=0 THEN DIR=4
720 IF DIR=5 THEN DIR=1
730 GOSUB 1090 :: GOTO 660
740 IF DIR=1 AND MAZE$(YP-1,XP)=
"1" THEN YP=YP-1 :: GOTO 780
750 IF DIR=2 AND MAZE$(YP,XP+1)=
"1" THEN XP=XP+1 :: GOTO 780
760 IF DIR=3 AND MAZE$(YP+1,XP)=
"1" THEN YP=YP+1 :: GOTO 780
770 IF DIR=4 AND MAZE$(YP,XP-1)=
"1" THEN XP=XP-1 :: GOTO 780
780 ST=ST+1 :: GOSUB 1090
790 IF YP=10 AND XP<3 AND DIR=4
THEN 1950
800 IF YP=10 AND XP>19 AND DIR=2
 THEN 1950 ELSE GOTO 660
810 REM ********HELP*******
820 ST=ST+10
830 CALL COLOR(6,13,1):: GOSUB 8
40 :: CALL CLEAR :: GOSUB 1520 :
: GOSUB 1090 :: GOTO 660
840 REM ******DRAW MAP*****
850 CALL CLEAR :: CALL DELSPRITE
(ALL)
860 DISPLAY AT(1,1):"   H HHHH
  HHHH"
870 DISPLAY AT(2,1):"H HHHH  H
HHH H    HHH"
880 DISPLAY AT(3,1):"H  H   HHHH
    HHHH H HHH H"
890 DISPLAY AT(4,1):"H HHHH H  H
HH  H  HHH H HHH"
900 DISPLAY AT(5,1):"HHH  HHH  H
 H     H HHHH  H"
910 DISPLAY AT(6,1):"     H HHHH
    HHHH  H H  HH"
920 DISPLAY AT(7,1):"   HHHH  H
   H H   HHHH   H"
930 DISPLAY AT(8,1):"   H  H  HH
H  HHHH    H HHHH"
940 DISPLAY AT(9,1):" HHHH HHHH
HHHH H   HHH  H"
950 DISPLAY AT(10,1):" H H   H
 H H      H   HHH "
960 DISPLAY AT(11,1):" H     H
 HHHH  H  H HHH H"
970 DISPLAY AT(12,1):" HHHH  HHH
  H H  HHHH H HHH"
980 DISPLAY AT(13,1):"  H HHH  H
```

IN A MAZE

```
HHH HHHH    H H HH"
990 DISPLAY AT(14,1):" HHHH H  H
     H   HHHH    H"
1000 DISPLAY AT(15,1):"HH H  HHH
H  HHHH HHH H   HHH"
1010 DISPLAY AT(16,1):" HHHH  H
   H    H HHH HHH H"
1020 DISPLAY AT(17,1):"HH  HHHH
 H H  HHHH H  H HH"
1030 DISPLAY AT(18,1):"H H  H
 HHHH  H   HHHH H"
1040 DISPLAY AT(19,1):"HHH  HHHH
 H  HHHH  HH  HH  H"
1050 DISPLAY AT(20,1):"  HHHH  H
HH  H  HHHH    HHHH"
1060 IF YP=0 THEN 1080
1070 GOSUB 1880
1080 RETURN
1090 REM *****DRAW DOORS****
1100 CF=0 :: L=0 :: S=-1 :: FD=0
1110 IF YP=10 AND XP<3 AND DIR=4
 THEN FD=1
1120 IF YP=10 AND XP>24 AND DIR=
2 THEN FD=1
1130 ON DIR GOTO 1140,1410,1230,
1320
1140 REM *****UP********
1150 IF MAZE$(YP-L,XP)="0" THEN
CF=1
1160 S=S+3 :: IF S=14 THEN 1500
1170 IF CF=1 AND FD=0 THEN CALL
COLOR(S,5,5,S+1,5,5,S+2,5,5):: C
ALL COLOR(#S-1,1,#S+12,1,#S,1,#S
+13,1):: GOTO 1160
1180 IF CF=1 AND FD=1 THEN CALL
COLOR(S,2,1,S+1,2,1,S+2,2,1):: C
ALL COLOR(#S-1,1,#S+12,1,#S,1,#S
+13,1):: GOTO 1160
1190 IF MAZE$(YP-L,XP-1)="1" THE
N CALL COLOR(S,13,15):: CALL COL
OR(#S-1,15,#S+12,15)ELSE CALL CO
LOR(S,6,15):: CALL COLOR(#S-1,1,
#S+12,1)
1200 IF MAZE$(YP-L,XP+1)="1" THE
N CALL COLOR(S+1,13,15):: CALL C
OLOR(#S,15,#S+13,15)ELSE CALL CO
LOR(S+1,6,15):: CALL COLOR(#S,1,
#S+13,1)
1210 CALL COLOR(S+2,15,1)
1220 L=L+1 :: GOTO 1150
1230 REM *****DOWN********
1240 IF MAZE$(YP+L,XP)="0" THEN
CF=1
1250 S=S+3 :: IF S=14 THEN 1500
1260 IF CF=1 AND FD=0 THEN CALL
COLOR(S,5,5,S+1,5,5,S+2,5,5):: C
ALL COLOR(#S-1,1,#S+12,1,#S,1,#S
+13,1):: GOTO 1250
1270 IF CF=1 AND FD=1 THEN CALL
COLOR(S,2,1,S+1,2,1,S+2,2,1):: C
ALL COLOR(#S-1,1,#S+12,1,#S,1,#S
+13,1):: GOTO 1250
1280 IF MAZE$(YP+L,XP+1)="1" THE
N CALL COLOR(S,13,15):: CALL COL
OR(#S-1,15,#S+12,15)ELSE CALL CO
LOR(S,6,15):: CALL COLOR(#S-1,1,
#S+12,1)
1290 IF MAZE$(YP+L,XP-1)="1" THE
N CALL COLOR(S+1,13,15):: CALL C
OLOR(#S,15,#S+13,15)ELSE CALL CO
LOR(S+1,6,15):: CALL COLOR(#S,1,
#S+13,1)
1300 CALL COLOR(S+2,15,1)
1310 L=L+1 :: GOTO 1240
1320 REM *****LEFT********
1330 IF MAZE$(YP,XP-L)="0" THEN
CF=1
1340 S=S+3 :: IF S=14 THEN 1500
1350 IF CF=1 AND FD=0 THEN CALL
COLOR(S,5,5,S+1,5,5,S+2,5,5):: C
ALL COLOR(#S-1,1,#S+12,1,#S,1,#S
+13,1):: GOTO 1340
1360 IF CF=1 AND FD=1 THEN CALL
COLOR(S,2,1,S+1,2,1,S+2,2,1):: C
ALL COLOR(#S-1,1,#S+12,1,#S,1,#S
+13,1):: GOTO 1340
1370 IF MAZE$(YP+1,XP-L)="1" THE
N CALL COLOR(S,13,15):: CALL COL
OR(#S-1,15,#S+12,15)ELSE CALL CO
LOR(S,6,15):: CALL COLOR(#S-1,1,
#S+12,1)
1380 IF MAZE$(YP-1,XP-L)="1" THE
N CALL COLOR(S+1,13,15):: CALL C
OLOR(#S,15,#S+13,15)ELSE CALL CO
LOR(S+1,6,15):: CALL COLOR(#S,1,
#S+13,1)
1390 CALL COLOR(S+2,15,1)
1400 L=L+1 :: GOTO 1330
1410 REM *****RIGHT*******
1420 IF MAZE$(YP,XP+L)="0" THEN
CF=1
1430 S=S+3 :: IF S=14 THEN 1500
1440 IF CF=1 AND FD=0 THEN CALL
COLOR(S,5,5,S+1,5,5,S+2,5,5):: C
ALL COLOR(#S-1,1,#S+12,1,#S,1,#S
+13,1):: GOTO 1430
1450 IF CF=1 AND FD=1 THEN CALL
COLOR(S,2,1,S+1,2,1,S+2,2,1):: C
ALL COLOR(#S-1,1,#S+12,1,#S,1,#S
+13,1):: GOTO 1430
1460 IF MAZE$(YP-1,XP+L)="1" THE
N CALL COLOR(S,13,15):: CALL COL
```

```
OR(#S-1,15,#S+12,12)ELSE CALL CO
LOR(S,6,15):: CALL COLOR(#S-1,1,
#S+12,1)
1470 IF MAZE$(YP+1,XP+L)="1" THE
N CALL COLOR(S+1,13,15):: CALL C
OLOR(#S,15,#S+13,15)ELSE CALL CO
LOR(S+1,6,15):: CALL COLOR(#S,1,
#S+13,1)
1480 CALL COLOR(S+2,15,1)
1490 L=L+1 :: GOTO 1420
1500 IF FO=1 THEN CALL COLOR(14,
2,1)ELSE CALL COLOR(14,5,1)
1510 RETURN
1520 REM *******DRAW*******
1530 CALL CLEAR
1540 FOR TT=2 TO 14 :: CALL COLO
R(TT,1,1):: NEXT TT
1550 DISPLAY AT(2,5):")888888888
888888888882"
1560 DISPLAY AT(3,5):"()88888888
888888888820"
1570 DISPLAY AT(4,5):"(()8888888
888888888200"
1580 DISPLAY AT(5,5):"((()888888
888888882000"
1590 DISPLAY AT(6,5):"((((APPPPP
PPPPPPPJ0000"
1600 DISPLAY AT(7,5):"((((@APPPP
PPPPPPJH0000"
1610 DISPLAY AT(8,5):"((((@@APPP
PPPPPJHH0000"
1620 DISPLAY AT(9,5):"((((@@@Yhh
hhhhbHHH0000"
1630 DISPLAY AT(10,5):"((((@@@XY
hhhhb`HHH0000"
1640 DISPLAY AT(11,5):"((((@@@XX
q";CHR$(128);CHR$(128);"z``HHH00
00"
1650 DISPLAY AT(12,5):"((((@@@XX
p";CHR$(136);CHR$(136);"x``HHH00
00"
1660 DISPLAY AT(13,5):"((((@@@XX
p";CHR$(136);CHR$(136);"x``HHH00
00"
1670 DISPLAY AT(14,5):"((((@@@XX
t";CHR$(128);CHR$(128);"{``HHH00
00",
1680 DISPLAY AT(15,5):"((((@@@X\
hhhhc`HHH0000"
1690 DISPLAY AT(16,5):"((((@@@\h
hhhhhcHHH0000"
1700 DISPLAY AT(17,5):"((((@@DPP
PPPPPPKHH0000"
1710 DISPLAY AT(18,5):"((((@DPPP
PPPPPPPKH0000"
1720 DISPLAY AT(19,5):"((((DPPPP
PPPPPPPPK0000"
1730 DISPLAY AT(20,5):"(((,88888
8888888883000"
1740 DISPLAY AT(21,5):"((,888888
888888888300"
1750 DISPLAY AT(22,5):"(,8888888
8888888888830"
1760 DISPLAY AT(23,5):",88888888
888888888883"
1770 CALL MAGNIFY(4)
1780 CALL CHAR(32,RPT$("F",64)):
: CALL CHAR(36,RPT$("F",32)&RPT$
("0",32))
1790 CALL SPRITE(#1,32,1,153,49)
:: CALL SPRITE(#14,32,1,9,49)
1800 CALL SPRITE(#2,32,1,153,193
):: CALL SPRITE(#15,32,1,9,193)
1810 CALL SPRITE(#4,32,1,129,81)
:: CALL SPRITE(#17,32,1,33,81)
1820 CALL SPRITE(#5,32,1,129,161
):: CALL SPRITE(#18,32,1,33,161)
1830 CALL SPRITE(#7,32,1,113,105
):: CALL SPRITE(#20,32,1,49,105)
1840 CALL SPRITE(#8,32,1,113,138
):: CALL SPRITE(#21,32,1,49,138)
1850 CALL SPRITE(#10,36,1,105,12
0):: CALL SPRITE(#23,36,1,57,120
)
1860 CALL SPRITE(#11,36,1,105,13
8):: CALL SPRITE(#24,36,1,57,138
)
1870 RETURN
1880 REM ***DISPLAY POSITION****
*****
1890 DISPLAY AT(YP,XP)SIZE(1):CH
R$(80)
1900 FOR T=1 TO 50
1910 FOR T1=1 TO 5 :: CALL COLOR
(7,2,1,9,5,1):: NEXT T1
1920 FOR T2=1 TO 5 :: CALL COLOR
(7,9,1,9,5,1):: NEXT T2
1930 NEXT T
1940 RETURN
1950 REM ****END SEQENCE****
1960 CALL CLEAR :: CALL DELSPRIT
E(ALL):: CALL SCREEN(3):: CALL C
HARSET
1970 DISPLAY AT(5,10):"WELL DONE
"
1980 DISPLAY AT(7,3):"YOU DID IT
 IN ";ST;" MOVES"
1990 DISPLAY AT(10,2):"TO PLAY A
GAIN PLEASE RERUN"
2000 DISPLAY AT(12,8):"THE PROGR
AMME"
2010 END
```

Hi-Res Plotter

The TI99/4A lacks Hires commands. Or it did until J Murray came up with this invaluable program to tackle the problem.

The Texas Instruments TI99/4A home computer is described as having a screen resolution of 256 x 192 pixels. However, TI BASIC lacks commands to control the screen in a 'high resolution' mode, such as PLOT, DRAW and MOVE, which makes drawing graphs very difficult. The only way it is possible to create a high resolution picture on the TI is by defining an eight by eight pixel character and then positioning it on the screen by using CALL HCHAR in the usual way.

If it is necessary to display a graph, it can become very tedious to have to define and calculate the position of many characters to do this. The best solution is to create a program to handle all the defining and positioning of characters for you and here it is!

The program is designed to be used as a sub-routine in conjunction with a program of your own. The method of operation is quite simple, the main program only needs to supply two parameters to the subroutine. They are the x and y coordinates of a point on a screen of resolution 256 x 192 with 0¯x¯255 and 0¨y¯191. The value of x is stored in the variable X and the value of y is stored in the variable Y. Then a GOSUB to line 28 is performed and a single point is plotted on the screen. To unplot, a GOSUB to line 18 must be performed.

There are two more subroutines provided, one is to save the finished picture onto tape, the other is to load it back again. However, when the picture is loaded it cannot be seen, so it must be printed out. The picture is stored on tape as a string of characters (as used to define a character in CALL CHAR (128,"0") followed by its x position and its y position on screen (where 1≤x≤32 and 1≤y≤24). To print out the picture from tape just use a simple loop like this:

```
110 FOR A=1 TO
      CH_MAX-33
120 CALL HCHAR
      (POSN(A,1),PO
      SN(A,2), A+32)
130 NEXT A
```

By adding or subtracting a constant to POSN(A,1) or POSN(A,2) your picture may be moved around the screen, perhaps it may be inserted between text which makes reference to a graph etc. An example of a graph drawing program might be as follows NB the origin of the screen is in the bottom left corner.

```
110 CALL SCREEN(8)
120 PI=4*ATN(1)
130 FOR N=1 TO 64
140 Y=80+20*SIN
      (N/32*PI)
150 X=N+100
160 GCSUB 28
170 NEXT N
```

This will print out a sine graph near the middle of the screen. If you replace line 150 in the above program with X=100+20*COS(N/32*PI) it will draw a circle.

variables used

BIN$ ()	Contains 4 bit binary representation of the numbers 0-15.

ROW()
COL()
	Contains data about current character being altered.

CHAR$ ()	Contains shape of character being altered in the form of a sixteen character string.
POSN()	Contains the x and y coordinates of the character being altered.
A,	General purpose loop.
ID$	Contains powers of 2 from 0-3 in the same order as is used to identify a character in its shape definition.
CH_MAX	The highest character code to be used next.
HEX$	Contains hexadecimal digits from 0-F in ascending order.
COM$	Contains 4 bit binary representation of part of character to be altered.
COLUMN	The horizontal position being considered in the character being altered.
V	The decimal value of the part of the hexadecimal string that defines the character, currently being considered.
IN$	The hexadecimal equivalent of V which needs to be inserted into the correct part of the character definition string to alter the shape of the character on screen.
CHAR	The character code of the character currently being altered.
YPOS	The vertical position of the character being altered on the screen.
XPOS	The horizontal position of the character being altered on the screen.
Y	The position on a 256 x 192 size screen of the dot being plotted in the vertical direction.
X	The position on a 256 x 192 size screen of the dot being plotted in the horizontal direction.
YSTORE XSTORE	Used to prevent X and Y being corrupted from the main program by the subroutine
SEG	The position in the character definition string of the data that need to be altered.
C$	The data in the character definition string that need to be altered.

how it runs

Line	Effect
1-17	Clear screen and initialise variables
18-227	Subroutine to UNPLOT point
28-45	Subroutine to PLOT point
46-54	Check validity of position on screen
55-61	Calculate positions on screen and store in X and Y
75-82	Calculate what data are in the definition string of character at appropriate point on the screen and store their positions
84-91	Subroutine to save a finished picture on tape
91-101	Subroutine to load a picture in the computer
102-	Main Program area

hints on conversion

This program is designed to provide a high resolution plotting system for a micro which does not have one. However, the micro that it is run on must have some form of character set re-definition. The program is not really suitable for conversion to other micros as it was designed specifically for use on the TI. It would be a better idea to study how the program works and then try to write one for your own micro.

The TI's screen is 32 characters wide by 24 lines high and the origin is in the top left corner. Characters on the TI are defined in a string of sixteen hexadecimal digits. e.g. CALL CHAR (128, "0123456789ABCDEF")

program listing

```
1 REM  HIGH RES. PLOTTER
2 REM  BY J.S.MURRAY
3 CALL CLEAR
4 OPTION BASE 1
5 DIM BIN$(16),COL(127),ROW(127)
,CHAR$(127),POSN(127,2)
6 FOR A=1 TO 127
7 CHAR$(A)="0000000000000000"
8 NEXT A
9 ID$="84218421"
10 CH_MAX=33
11 CHAR=CH_MAX
12 HEX$="0123456789ABCDEF"
13 RESTORE 83
14 FOR A=1 TO 16
15 READ BIN$(A)
16 NEXT A
17 GOTO 102
18 REM  UNPLOT X,Y
```

Hi-Res Plotter

program listing

```
19 GOSUB 46
20 COM$=BIN$(V+1)
21 COLUMN=COL(CHAR-32)
22 IF COLUMN<5 THEN 24
23 COLUMN=COLUMN-4
24 IF SEG$(COM$,COLUMN,1)="0" TH
EN 27
25 V=V-VAL(SEG$(ID$,COL(CHAR-32)
,1))
26 GOTO 36
27 RETURN
28 REM  PLOT X,Y
29 GOSUB 46
30 COM$=BIN$(V+1)
31 COLUMN=COL(CHAR-32)
32 IF COLUMN<5 THEN 34
33 COLUMN=COLUMN-4
34 IF SEG$(COM$,COLUMN,1)="1" TH
EN 43
35 V=V+VAL(SEG$(ID$,COL(CHAR-32)
,1))
36 IN$=SEG$(HEX$,V+1,1)
37 CHAR$(CHAR-32)=SEG$(CHAR$(CHA
R-32),1,SEG-1)&IN$&SEG$(CHAR$(CH
AR-32),SEG+1,16-SEG)
38 IF CHAR$(CHAR-32)<>"000000000
0000000" THEN 41
39 CALL HCHAR(YPOS,XPOS,32)
40 GOTO 43
41 CALL CHAR(CHAR,CHAR$(CHAR-32)
)
42 CALL HCHAR(YPOS,XPOS,CHAR)
43 Y=YSTORE
44 X=XSTORE
45 RETURN
46 REM  CALCULATE V
47 IF (X>=0)*(X<=255)THEN 51
48 CALL SOUND(100,225,0)
49 PRINT "* X PLOT OUT OF RANGE =";X
50 STOP
51 IF (Y>=0)*(Y<=191)THEN 55
52 CALL SOUND(100,225,0)
53 PRINT "Y PLOT OUT OF RANGE =" ;Y
54 STOP
55 YSTORE=Y
56 XSTORE=X
57 Y=192-INT(Y+0.5)
58 X=INT(X+0.5)+1
59 YPOS=INT((Y-1)/8)+1
60 XPOS=INT((X-1)/8)+1
61 CALL GCHAR(YPOS,XPOS,CHAR)
62 IF CHAR<>32 THEN 75
63 IF CH_MAX<>160 THEN 73
64 CH_MAX=CH_MAX-1
65 FOR A=1 TO 127
66 IF CHAR$(A)<>"0000000000000000"
THEN 69
67 CHAR=32+A
68 GOTO 75
69 NEXT A
70 CALL HCHAR(24,16,30)
71 CHAR=CH_MAX
72 RETURN
73 CHAR=CH_MAX
74 CH_MAX=CH_MAX+1
75 ROW(CHAR-32)=8*((Y-1)/8-INT((
Y-1)/8))+1
76 COL(CHAR-32)=8*((X-1)/8-INT((
X-1)/8))+1
77 POSN(CHAR-32,1)=YPOS
78 POSN(CHAR-32,2)=XPOS
79 SEG=2*(ROW(CHAR-32)-1)+1+INT(
(COL(CHAR-32)-1)/4)
80 C$=SEG$(CHAR$(CHAR-32),SEG,1)
81 V=POS(HEX$,C$,1)-1
82 RETURN
83 DATA 0000,0001,0010,0011,0100
,0101,0110,0111,1000,1001,1010,1
011,1100,1101,1110,1111
84 REM  SAVE
85 OPEN #1:"CS1",SEQUENTIAL,INTE
RNAL,OUTPUT,FIXED
86 PRINT #1:CH_MAX
87 FOR A=1 TO CH_MAX-33
88 PRINT #1:CHAR$(A),POSN(A,1),P
OSN(A,2)
89 NEXT A
90 CLOSE #1
91 RETURN
92 REM  LOAD
93 OPEN #1:"CS1",SEQUENTIAL,INTE
RNAL,INPUT ,FIXED
94 INPUT #1:CH_MAX
95 FOR A=1 TO CH_MAX-33
96 INPUT #1:CHAR$(A),POSN(A,1),P
OSN(A,2)
97 CALL CHAR(A+32,CHAR$(A))
98 NEXT A
99 CLOSE #1
100 CHAR=CH_MAX
101 RETURN
102 REM  REST OF PROGRAM
110 CALL SCREEN(8)
120 PI=4*ATN(1)
130 FOR N=0 TO 64
140 Y=80+20*SIN(N/32*PI)
150 X=N+100
160 GOSUB 28
170 NEXT N
180 PRINT "DONE"
190 GOTO 190
```

MORE TI MUSIC

Texas Tunes from 99'er Magazine

```
10 ON BREAK STOP
100 CALL CLEAR :: CALL SCREEN(2)
110 CALL CHAR(42,RPT$("F",16))
120 FOR A=1 TO 12 :: CALL COLOR(A,12,1):: NEXT A
130 DISPLAY AT(7,9):" TEXAS TUNES"
140 DISPLAY AT(10,8):"(C) 99'er 1984"
150 DISPLAY AT(22,2):"  TEXAS INSTRUMENTS 1984"
160 CALL HCHAR(5,8,42,18):: CALL HCHAR(12,8,42,1
170 CALL VCHAR(5,8,42,8):: CALL VCHAR(5,26,42,8)
180 CALL HCHAR(1,1,42,32):: CALL HCHAR(24,1,42,32):: CALL VCHAR(1,1,42,24):: CAL
L VCHAR(1,32,42,24)
190 RESTORE 230
200 READ A,B,C :: IF A=0 THEN 190
210 CALL SOUND(-2000,A,5,B,5,C,5):: CALL KEY(0,K,S):: IF S<>0 THEN END
220 CALL COLOR(2,INT(RND*12)+3,1):: GOTO 200

230 DATA 392,311,233,392,311,233,392,311,233,523,370,220,523,370,220,523,370,220

240 DATA 466,392,233,466,392,311,466,392,294,622,415,262,622,415,262,622,415,262

250 DATA 622,392,233,587,349,233,523,311,233,466,294,175,466,294,175,440,262,175

260 DATA 523,311,175,523,311,175,523,311,175,466,294,233,466,294,233,466,294,233

270 DATA 392,311,233,392,311,233,392,311,233,523,370,220,523,370,220,523,370,220

280 DATA 466,392,233,466,392,233,392,311,233,253,349,220,523,349,220,523,349,220

290 DATA 698,466,294,587,349,233,622,392,233,523,311,233,523,311,220,587,311,196

300 DATA 523,311,175,523,311,196,523,311,220,466,294,233,466,294,233,466,294,233

310 DATA 466,415,294,587,415,233,523,415,262,466,415,294,523,415,311,466,415,294

320 DATA 466,392,311,622,392,311,523,370,311,466,392,311,466,392,311,466,392,311

330 DATA 466,415,294,587,415,233,523,415,262,466,415,294,523,415,311,466,415,294

340 DATA 466,392,311,622,392,311,440,370,311,466,392,311,466,392,311,466,392,311

350 DATA 622,392,262,622,392,262,622,392,262,622,349,196,622,349,220,587,349,247

360 DATA 523,311,208,523,311,208,253,311,208,494,311,208,494,311,208,494,311,208

370 DATA 466,392,233,466,392,233,466,392,311,466,415,294,523,415,294,466,415,294

380 DATA 466,392,311,466,392,311,466,370,311,466,415,294,253,415,294,466,415,294

390 DATA 622,392,277,622,392,277,622,392,277,622,415,262,622,415,262,622,415,247

400 DATA 622,392,233,622,392,233,622,392,233,622,392,233,622,392,233,622,392,233
,0,0,0
```

```
2 CALL COLOR(2,8,1)
5 ON BREAK STOP
10 CALL CLEAR :: CALL SCREEN(2):: FOR A=1 TO 12 :: CALL COLOR(A,8,1):: NEXT A
11 CALL CHAR(42,RPT$("0",16))
20 DISPLAY AT(8,8):" ON THE LAKE"
21 DISPLAY AT(10,8):" Acorn  User"
25 DISPLAY AT(22,4):"TEXAS INSTRUMENTS 1984"
32 CALL COLOR(2,8,8)
35 CALL HCHAR(6,9,42,15):: CALL HCHAR(12,9,42,15):: CALL VCHAR(6,9,42,7):: CALL
VCHAR(6,23,42,7)
90 CALL HCHAR(1,1,42,32):: CALL HCHAR(24,1,42,32):: CALL VCHAR(1,1,42,24):: CALL
 VCHAR(1,32,42,24)
100 X=2^(1/12):: DU=50
110 O=0 :: D=0
120 D=D+1
130 RESTORE 260
140 FOR N=1 TO 44
150 READ A,B,C :: CALL SOUND(C*DU,A*X^O,0,B*X^O,0)
155 CALL COLOR(2,1,INT(RND*12)+4)
160 NEXT N
170 ON D GOTO 180,190,200,210,220
180 O=-12 :: DU=-50 :: GOTO 120
190 O=0 :: DU=50 :: GOTO 120
200 O=-6 :: DU=-50 :: GOTO 120
210 O=0 :: DU=50 :: GOTO 120
220 END
230 REM
240 REM
250 REM

260 DATA 1319,523,5,1175,523,5,1047,523,5,1568,494,5,1568,440,5,1568,392,5,
     1319,523,5,1175,523,5

270 DATA 1047,523,5,1568,494,5,1568,440,5,1568,392,5,1319,523,5,1175,523,
     5,1047,523,5

280 DATA 1397,587,5,1319,587,5,1397,523,5,1568,494,5,1397,494,5,1319,523,
     5,1175,392,5

290 DATA 1175,440,5,494,494,5,1319,523,5,1175,523,5,1047,523,5,1568,494,5,
     1568,440,5,1568,392,5

300 DATA 1319,523,5,1175,523,5,1047,523,5,1568,494,5,1568,440,5,1568,392,5,
     1319,523,5,1397,523,5

310 DATA 1319,523,5,1175,440,5,1319,440,5,1175,440,5,1047,523,15,1047,523,10
```