SOUTH BAY TI USERS GROUP

BITS & BYTES NEWSLETTER

Mailing Address: P.O. BOX 23447 SAN JOSE, CALIFORNIA 95153-3447

*** MARCH 1988 ***

PRESIDENTS MESSAGE
by Mike Ewell

The demo at the March meeting will be on using Archiver v2.4 or EGBS, the disk editor program from Canada. Which demo to use will be decided! by the members at the March meeting. If time permits, both could be done. Both programs are in the club library and are well worth having. I use them both from the 99/4A mode of the 9640.

EGBS is the fastest "Diskfixer" type of program that I have used so far. The string searches are easy to use as is the "reading" of the disk with the ASCII option.

Archiver V-2.4 is the latest improvement on the all assembly program to have both packing and squeezing functions. Almost all of the programs that I will be downloading in the future will be both archived and compressed. This is the future for all programs to be put into the club library, so now is the time to learn how to use it!!

Thanks to Kevin Daberkow for the demo on PRBASE at the last meeting. This data base program is also in the club library. There are two versions, 2.0 and 2.1! V2.1 was modified for use on both the 99/4A and the 9640. Files saved with 2.0 can be modified for use on 2.1 with the instructions that are in the notes. You still need the docs from 2.0. In both PRBASE versions, you must name both disks PRBASE. Also, PRBASE will not accept DSDD. In fact?, it doesn't like DD at all, but it will accept DS formats.

[]

The March SBTIUG meeting will be held at 7:13 P.M.

THURSDAY, MARCH 3, 1988

The meeting will be held in the Saratoga Public Library. The library is located at 13650 Saratoga Avenue. From 280(680), take the Saratoga Avenue exit SOUTH. The library will be on your left, just past the Fruitvale intersection. This is about four miles from the 280 exit.

[]

I will be attending TI XPO 88, formerly the TI FEST WEST, in Las Vegas this 27th and 28th of February.

The members need! to give the club officers more input as to what they want done at the meetings. It does not make sense to demo something that only two or three vocal! people want, but not demo something twenty? silent!!!! people would like.

Please ask Bill what YOU can do for YOUR newsletter.
Please ask Pat what YOU can do for YOUR newsletter.
Please ask Bill what YOU can do for YOUR newsletter.
Please ask Pat what YOU can do for YOUR newsletter.

X NOTICE XX NOTICE XX NOTICE XX NOTICE XX NOTICE XX NOTICE X

APRIL AND MAY MEETINGS

The APRIL meeting will be held on 4-7-87 which is the first Thursday.

The May meeting will be held on 5-5-88 which is the first Thursday.

The first Thursday will continue to be the first choice for our meetings, but please read the dates carefully to avoid a wasted trip or missing a meeting.

X NOTICE XX NOTICE XX NOTICE XX NOTICE XX NOTICE XX NOTICE X

SBTIUG CLUB OFFICERS AND OTHERS

PRESIDENT.............MIKE EWELL................408/370-7988
VICE-PRESIDENT........JOHN WENTE................408/559-1680
TREASURER.............KEVIN DABERKOW............408/281-7435
SECRETARY.............NORMAN KNUDSEN............408/267-6193
LIBRARIAN.............HELMUT FUCHS..............408/263-8339
EDITOR................BILL SCHULT...............408/446-1182
*ASSISTANT...........PAT MICETICH..............408/227-6212
*ASSISTANT...........How about YOU!!!..........408/???-????
SBTIUG BBS............24-HRS/DAY?...............408/258-3679
*SYSOP...............KEITH FELIX...............408/258-6678
*REMOTE SYSOP........RON SPRINGER..............408/225-8670
CALTEX BBS............FRANK CONTINOLA...........408/578-6264
PUBLIC RELATIONS......DON APTE..................408/629-0725
ARCHIVER/NEWSLETTERS..HOY COLE..............SEE AT MEETING

SBTIUG GENERAL MEETING 4 FEBRUARY, 1988

by Norm Knudsen

The February meeting of the SBTIUG was held at the Saratoga Public Library and was called to order at 7:25 PM. There were 18 members in attendance.

President Mike Ewell first called for a treasurer's report from our NEW treasurer, Kevin Daberkow who proclaimed a starting balance of $503.48. He added, however, that the cost of the last newsletter is still outstanding.

The librarian's report was next. Helmut Fuchs announced that printed copies of the documentation for PR Base w\\e available. (It is also available on disk if you wish to print your own.) A major revision to FUN-L-WEB, Version 4.0 is now in the library. A new version of Fast Term is also available.

Our PR man, Don Apte, had a few announcements on upcoming swap meets and computer faires as usual. He also mentioned that Federated was clearing out Atari software including some TI 99 modules.

With the Business segment af the meeting concise and to the point, we then adjourned to watch Kevin present his demo on PR-Base.

TREASURERS REPORT

by Kevin Daberkow

PLEASE look at your mailing label to see if some color has been added. If your membership expiration date has been high-lighted in RED, this is your last issue until you renew. If you membership expiration date is in YELLOW, then you should renew at the March meeting.

>> THE DUES ARE $15 PER YEAR <<

NOTE: Your membership expiration date can be found on the last line of your mailing label.

If any of the information on your label needs to be changed, please let me know. Call me at (408) 281-7435 or write to me at the following address:

SBTIUG - Treasurer
P.O. Box 23447
San Jose, CA 95153-3447

There were no renewals during the month of February.

This left a balance of $503.48 in our treasury at the end of February.

EDITORS RAMBLINGS

by Bill Schult

As the new newsletter editor, I wish to continue producing the high quality newslwtters that have been produced by my predecessors. In order to do this I am going to need the contributions from the members of the group.

I am not an accomplished writer, nor am I an expert on the TI-99/4A computer (or any other computer for that matter).I therefore will need some help in producing the newsletter. Help in the form of articles, reviews of products, programs, repair or modification articles of the computer or perpherals.

These articles will be accepted in any form on disk, since it is an easy matter to reformat them to fit the newsletter format. However it will be appreciated if they are in the 60 column format. In short, what I am asking for are articles that are informative and interesting to the readers of our newsletter.

Articles in future issues will include articles on RAMDISKS, C-LANGUAGE, EDITOR-ASSEMBLER USAGE and generally articles that I hope will be of interest to the readers of our newslwtter.

COMPUTER SWAP
Saturday, March 5, 1988
COW PALACE
10 AM to 5 PM
admission:$5.00-children under 12 free
ALL PRODUCTS AVAILABLE FOR IMMEDIATE
AT BARGAIN PRICES!

NEW DISKS IN OUR LIBRARY.

ARCHIVER 2.4...utility to decompress archived files.

FUNNELWRITER 4.0...new text editor version.

GBS...disk sector editor.

GAMES... no description given.

MENU CREATOR...no description given.

downloaded from The Source
TI-99/GENEVE Archiver Format - By Al Beard

1 General

This is a text file describing the archiver format popularized by Barry Traver. It is furthur extended by Huffman encoding techniques, into a squeezing archiver.

An archiver is a program, which reads a collection of (usually related) other programs, and combines them into a single file called an archive. Sometime later, the programs can be recovered from this single file in the original file formats.

The advantage of an archiver is that only a single file need be posted on a network instead of multiple files. A user need only be concerned that the single file was downloaded, and be assured that this single file contains all necessary elements to run a program (e.g. source, objects, executables, documentation files, readme files, etc.).

A second advantage of an archiver is that the archiver can be made "smart", i.e. it can take advantage of repitition of characters in files to "squeeze" or "crunch" the files into an archive, making the resulting archive usually some 30 to 60% smaller than the original files contained. This is done without loss of information. The "squeezed" archive is shorter to upload/download, saving the user 30 to 60% on his downloading fees. Also, moderators on networks and bulletin board systems encourage file compression as immediately 30 to 60% more mass storage becomes available, with no increase in costs! A squeezing archiver also makes sense to a user interested in archiving a number of infrequently used files (such as the source for a program after it has been compiled).

As a matter of fact, the only disadvantage to archiving is that the archive/dearchiving process tends to be time-consuming (2 to 15 minutes to unpack an archive!. But, since this is done offline, on an essentially free (your) system, this usually isn't very significant.

2 The TRAVER Algorithm/Format

Barry Traver introduced the most popular archiver for the TI-99 and Geneve. It is written in Extended BASIC, is quite a professional program, and demonstrates the capabilities of Extended BASIC. A single assembly language subroutine was used, which allowed sector reads and writes by accessing a Device Service Routine (DSR) in the TI Disk controller card. The TRAVER archiver is FAIRWARE, for those people who subscribe to the TRAVelER, the program is free.

Barry picked a simple approach to the archiving algorithm, one that shows an indepth knowledge of the internal TI-99 disk file structure, and uses that structure to the archiver's advantage.

Files are packed in a archive very close to the same way they are on the TI-99 disk. The archive itself consists of a fixed/ display/128 file (probably since BASIC can't open a fixed/display/ 256 byte file).

2.1 File Packing Algorithm

The basic algorithm for the archiver works something like this: a. The file descriptor index record is read which points to all of the one sector file header records.

b. The user is prompted for which files on the disk are to be compressed.

c. Each desired file header record is read, the unused bytes are stripped (including the reserved expansion bytes, and the data chain pointer blocks) leaving a "mini" file descriptor record which consists of the following:

- o 10 character file name
- o File Status Flags
- o Maximum Number Records/Sector or AU
- o Total Number of Sectors Used
- o End of File Offset
- o Logical Record Length
- o Number of Fixed Records or Number Sectors Used

These 18 byte mini file headers are packed fourteen to two 128 byte records (for a total of 252 bytes). The remaining unused four bytes per sector contain either zeroes (meaning this is NOT the last header sector) or the characters END! (meaning this is the last header sector).

d. Following the header section of the archive is the data section. Each 256 byte sector of the file is packed as two 128 byte records.

2.2 File Unpacking Algorithm

The file unpacking algorithm is quite straightforward. The user is solicited for the file or the files to be "unpacked". These are read from the archive file headers, and: a. The data section for each file is located using the "total number of sectors used" field in the mini-file descriptor records.

b. The file is read from the archive, and written to the disk as a DISPLAY/FIXED/128 file.

c. After the file has been successfully written to the disk, the disk is searched for the file header and the file header is overwritten by the information contained in mini-file descriptor.

3 The BEARD Huffman Squeezing FORMAT

An extension to the TRAVER archiver format was released by myself in September 1987. This extension provides capability for squeezing the archive using HUFFMAN encoding techniques originally developed by R.Greenlaw under CP/M. This CP/M method was ported from CP/M to the IBM PC, and then to the Amiga, and finally to the TI-99.

Software sources were obtained in the public domain from BYTE INFORMATION EXCHANGE.

Thanks should be offered at this point for the inputs provided by Dr. Jerry Coffey, who helped me work out the final bugs in the squeezing archiver, especially for helping me make it backwards compatible to the Traver Archive.

The program is written in FORTRAN, so to use it you need to either own a copy of FORTRAN, or get a copy of the public domain module 99STAND.ARC (for version 2.0) or FORTSA (for version 3.1 and above).

The program maintains backwards compatibility to the TRAVER format, it can unpacked either squeezed or unsqueezed archives, or a combination of them. The basic format of the archive is the same, a header section consisting of "Traver-like" mini file headers, followed by a data section of squeezed and (possibly) unsqueezed data files.

3.1 File Analysis

Before a file can be squeezed, it must be analyzed. This is performed automatically by the squeezing archiver as a first pass on the file.

The analysis portion basically consists of reading every sector in the file to be compressed, and counting the occurances of each of the possible 256 data byte values. Once all of the file has been read, and all of the occurance counts have been collected, then a sort and binary tree is created containing the optimal Huffman codes for the file.

3.2 File Squeezing Algorithm

Once the file has been analyzed, and a binary tree created, then the squeezing part of the archiver is executed. This portion first writes a prologue to the data file consisting of the archival type, the sector length of the original file, the number of binary tree nodes, and the binary tree, followed by variable length bit strings representing the Huffman encoded data.

Following the squeeze operation, the reduction in the file size is tested. If the resulting file is larger than the original file, then the file is "re-packed", using the normal unsqueezed TRAVER format.

If a reduction is detected in the file size, then the file header is modified. First, the file status flag is OR'd with a '20'X, indicating an archived file. The Total Number of Sectors used field is changed to the Total Number of 128 byte records used.

The actual data file contains the following information:

o One word representing the archiving method. This is currently set to a 1, since this is the first squeezing method to be implemented. Future methods should set this to a different unique number representing the different squeezing technique.

o One word representing the original value for the Total Number of Sectors. Note that this was changed in the mini-file header to be the number of 128 byte records.

o One word representing the number of nodes in the binary huffman tree (this word starts the particular encoding scheme, whereas the first two words must be understood by all archivers).

o The nodes of the binary tree, first the left child, and then the right child, repeated for the NUMNODES.

o The squeezed archive data, terminated by a special "end-of-file" character.

3.3 File Unsqueezing Algorithm

The file unsqueezing algorithm is similar to the "non-squeezing" file unpacking algorithm. The only key is to recognize that this is a squeezed file (by the special file status bit OR'd '20'X), and to recognize that the total sectors used is actually the total records used if this is a squeezed file.

If this is not a squeezed file, than the unpacking algorithm is identical to the TRAVER format.

One small difference can occur between an unpacked file (via the Traver method) and a squeezed file (via the Beard method). The final sector of a file is usually only partially used, and the squeezing archiver takes advantage of this fact to only save the in use partial amount of the final sector. The archiver "zero- fills" the remainder of the sector. The inconsistency is that the final sector can be different (in its unused portion) than the original file. This should not cause any problems in using the file (unless some tricks have been played with the DSR), but would show as a difference on a sector by sector compare.

4 Improvements

Many improvements to the implemented squeezing archiver are possible and encouraged. The following is a list of possible improvement areas:

a. The TI-99 disk format is quite wasteful in that it does not split a record (either fixed or variable) over a sector boundry. Therefore, a DISPLAY/FIXED/129 file will have 127 bytes/sector of unused space! Variable files are similar, if the next record to be stored cannot fit in the current sector, then the record is terminated and the next record is started. Since the squeezing archive is actually a stream of bits, this unused space could easily be skipped in the archive.

b. The bit manipulation routines in this program are in FORTRAN, therefore are slower than what is achievable with Assembly.

A good deal of the lower level stuff could be streamlined.

c. It would be useful if the archiver would recognize that a file could not be squeezed at the "analysis" stage rather than after it has been squeezed and written.

d. Different, and better squeezing algorithms are available. It might be possible to borrow from the IBM PC archiver, and do the analysis in various formats, and pick the compression method that yielded the best result.

e. There is currently no support for hard disks, or reading/ writing sectors to/from a RAM disk. (the reason being that I don't have a hard disk, and I haven't figured out the RAM disk yet).

5 Archive FORMAT Recap

a. Archive consists of two sections packed into a 128 byte fixed length record file:

c Header Section

o Data Section

b. The header section consists of "mini-file headers" packed 14 to a record. Each "mini-file header" is a stripped version of a standard TI-99 file header, and contains the following:

byte nos	Contents		
0-9	10 Character (MAX) file name. Unused characters are space characters.		
10	File Status Flags:		
	Bit No	ON=1	OFF=0
>00 Dis/Fix	0	Program File	Data File
>01 Program	1	Internal	Display
>02 Int/Fix	2	Reserved	
>80 Dis/Var	3	Write Prot	No Write Prot
>82 Int/Var	4	Reserved	
	#5	Squeezed	Unsqueezed
# Non-standard meaning			
	6	Reserved	
	7	Variable Len	Fixed Len
11	Maximum Number of Records/Sector or AU		

12-13 Total Number of Sectors Used (unsqueezed archive) or total Number of 128 byte Records Used (squeezed archive)
14 End of File Offset
15 Logical Record Length
16-17 Number of Fixed Length Records or Number of Sectors Used by Variable Length Records
The last 128 byte header record contains the characters "END!" in the last four bytes. Unused header record slots (to fill out a 128 byte record) contain zeroes.
c. The data section follows the header section, and contains a number of data files pointed to by the header section. There is no special seperation between the data sections in the data section itself, it is merely a collection of 128 byte records.
The unsqueezed data section contains an even number of 128 byte records for each sector in the input file.
The squeezed data section has a more complex format, but still consists of 128 byte records. Actually, the data section can be thought of as a continuous "stream" of data, organized as follows: word 0 - Set to a "1", to indicate the compression method.
word 1 - The original value for the Total Number of Sectors
word 2 - The number of nodes in the binary tree
words 3 to numnodes*2+2 - The binary tree remaining - Variable bit data for the Huffman Encoding.

MORE FLEXIBILITY IN USING PRBASE

by Ken Woodcock
reprinted from TIDEWATER TI USERS GROUP

Ever since I got version 2.1 of PRBASE from Mike Dodd (the version that works on both the TI and GENEVE) I hae been urging everyone to convert their previous version data disks to this format. Now there is an additional reson to. If you have ever wished to be able to access your PRBASE data files from a BASIC-XB program, now you can

As I was perjing a stack of newsletters from user groups in our exchange program, I came across an article in the NOV 87 WEST PENN 99'ers which caught my eye (their newsletter always has something of interest). This article was entitled "TRANSFERING PRBASE FILES" by Doug Bootee, CIN-DAY user group his article described a meathod he had devised to create a D/F 128 "SHELL" file on another disk and then transfer the PRBASE records to it using the XBASIC PRBUTIL program by John Johnson. I tried his method and it worked fine... But it seemed like a pain th have to keep a separate disk. Then last night while watching the 49'ers beat the Browns I got this inspiration! Why not build the "SHELL"file around the data on the PRBASE data disk?? Yeah, that's it! After all, Mike Couture had just given us a fine discussion on disk sector formats at the last meeting, plus.

I had just received version 4.0 of "DISK UILITIES" by John Birdwell (a great program- ya'll send him some $$). This would not be possible with the 2.0 or earlier data disks because sector 0 and 1 are used as part of the "HEADER" section in those versions but Mke Dodd moved everything down two sectors to allow for a standard disk header. so we can put the file pointer onto sector 1: the problem is where to put the FILE DESCRIPTOR RECORD (hereafter called the FDR). The FDR consumes one complete sector and contains all the informaion about the file such as name, type,record length, location, size etc. It can exist anywhere on the disk- the pointer in sector I tells the disk controller w;ere to find it. If you are running single density (TI controller). The best location is probably the last sector on the disk-359 (>167) for single sided or 719 (>2CF) for double sided. This will mean you'll have to give up one more record in addition to the 2 that Mike Dodd took leaving 347 maximum for SSSD and 707 max for DSSD. Well as the wise man said... you must give up something to get something. If your are fortunate enoug; to own a double density disk controller, you don't have to give up anything. In fact you can even take back the 2 records that Mike Dodd apropriated! In this case format the disk DSDD, then put the FDR at sector 722 (>2D2). This is the next sector following the space reserved for the 710 (max) records. The rest of the disk space will be available for normal program storage after we "doctor up" sector 0 to prevent overwriting the PRBASE data. OK lets get started. We will have to use a sector editor to setup sector 0,1 and the FDR (I highly recommend DISK UTILITIES). Starting here, put an F in each position- all the way to the end if the disk is not DSDD. For a DSDD disk only go through sector 146 (>92). Write this editaed secror then go to sector 1. In the first 4 positions put the sector number of the FDR (0167 for SSSD 02CF for DSSD/SSDD 02D2 for DSDD). Write this sector to disk. Now all that remains is to create the FDR. The easiest way to get most of the information for this sector is to run this program using another freshly initialized disk of the same format.

OPEN #1:"DISK1.XXX",RELATIVE 346,FIXED 128::PRINT #1:REC 346:"X"::CLOSE #1 substitute 706 for 346 (2 places) for 2SSD or SSDD disks. For DSDD juse 709. (relative files start with record 0) Now you can copy the FDR from this disk (it should be at sector 2) to the disk you are setting up. Remember to copy it to the appropriae sector 359 (>167) for SSSD, 719 (>2CF) forDSSD/SSDD or 722 (>2D2) for DSDD. Now all that remains is to modify 3 bytes on this FDR. These are 28, 29, and 30 (>1C, >1D, >1E) DON'T FORGET THAT THE FIRST BYTE IS ZERO!

SSSD-0C A0 15 DSSD/SSDD- 0C 20 2C DSDD- 0C 60 2C

If this seems like a lot of work it only has to be done once. Thereafter you can copy the three sectors to your other data disks. Besides, athink of all you'll learn from the experience! If you're really not in the mood for learning, send me a disk initialized in the desired format with a reusable mailer and return postae and I'll do it for you. (If double density, 18 sector/track only)

Ken Woodcock 4701 Atterbury street Norfolk Va. 23513

BURGLAR ALARM

Rick Lumsden Winnipeg, Manitoba Canada

The program listed allows you to use your spare TI console as a burglar alarm with very little investment except for a bit of time.

The actual program is very simple and can be modified to suit your own particular needs. This particular version has a lot of statements that allow you to see what is going on in the program while running a demonstration, However they can be removed quite easily with no effect on the operation of the program. Just a few cautions though. Understand the program first before making any drastic changes. The other preraution is not to use your perimeter loop on the same joystick "direction" as the entry keyswitch. (eg. if you use the UP position for the keyswitch do not use this direction for the perimeter loop even if it is opposite joysticks) The program is set to use the UP position of joystick 2 for the entry keyswitch and the DOWN position of joystick 1 for the perimeter loop. It is also possible to use the other joystick directions (with appropriate program mods) to have more than one loop. Remember, this program will run as a standalone routine but is intended to be modified or totally rewritten by yourself to suit your particular job. The intention of this program was to be as simple as possible and not require any peripherals or modules. Most of us have a second console so here is a good use for it other than a paper weight.

To set up the alarm you will need the following:

1. TI console

2. Normally open magnetic or pushbuton switches for each door on the perimeter loop. (Radio Shack #49-495 or #49-497) with changes to the program(using the fire buttons and other positions) you may add other protection loops but you must insure that you have one switch per loop when using the normally closed switches. You may use as many switches of the normally open version on the loop as you wish.

3. Entry keyswitch (Radio Shack #49-515) or a hidden SPST toggle switch.

4. An audio amplifier and speaker(s) (your stereo amplifier will work just fine but the alarm will only be sounded in the house)

5. A cable to hook the audio out port from the console to the amp (if you have a monitor cable these will work fine. Some are availale for the TI from Super Valu stores for $10.95

6. Joystick connector (Radio Shack #276-1538)

7. Hook-up wire.

To run a simple demonstration of the program you will need two joysticks and your TV monitor

First, you may want to set the dela;y variables in lines 150 and 160. Line 150 is the exit delay variable. This allows you time to leave the house after you turn on the keyswitch. If you mount the keyswitch outdoors, then set this variable to 1.

The variable in line 160 sets the entry delay. This one allows you time to enter your home and disarm the system with the keyswitch before the alarm sounds. Remember to set this one on the fast side because it also delays in the event of a break-in.

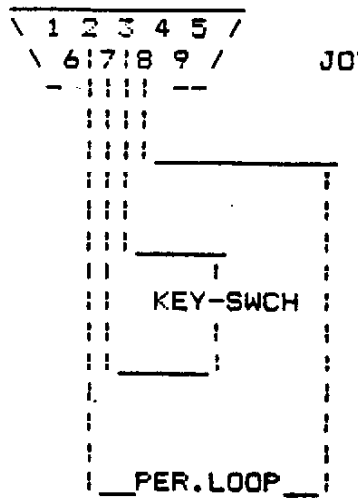
When you type RUN, the words "PLEASE REMOVE ALPHA LOCK" and "PRESS 'C' TO CONTINUE" appear. Follow the instructions and next comes "PERIMETER CHECK (Y/N)?". If you press "Y" the program jumps to line 700 and checks Joystick 1 for any openings in the protection loop. If an opening is found(such as J1 in the center position) the program sounds a warning and tells you to check and remedy the situation. Do this by moving J1 to the down position and holding it there. Now push the "R" key and the program goes back to line 310 and sounds the OK chime.

The word "UNARMED" appears and tells you that the system is now ready for input from the keyswitch. When you turn the keyswitch on (by holding J2 in the UP position and J1 in the DOWN position) the program goes to the exit delay loop. This loop allows you to leave your home without triggering the system. Once this times out the program begins looping and checking each of the joysticks for a change in state.

If J1 suddenly becomes open and program moves to the entry delay loop. This delay allows you to enter your home and disarm the system with the keyswitch without setting it off. If the timer times out (eg. break-in) the program now sounds the alarm. You can simulate this by letting J1 return to the center position. Even if you were to close the door now it is too late, the timer is running down and the only way to stop it is to disarm the system.

Any number of changes and additions can be made to the program limited only by your imaginagion and your requirements. The intention of this routine was to give you an idea of what is possible and also to be as simple as possible. There are also heat detectors available that work on the normally open and normally closed switch principles so a fire alarm can also be added.

Try it out and if you have any questions or ideas on how to improve the program I would be interested in hearing from you.



JOYSTICK PORT

PINOUTS

1 No Conn.
2 Right Grd.
3 Up
4 Fire Buttons
5 Left
6 No Conn.
7 Left Grd.
8 Down
9 Right

Rick Lumsden
18 Corton Place
Winnipeg, Manitoba
Canada
R2N 1W6

```
100 REM BURGLAR ALARM PROGRAM
110 REM FOR THE TI HOME COMPUTER
120 REM A PUBLIC DOMAIN PROGRAM
130 REM WRITTEN BY R.A.  LUMSDEN WINNIPEG,MANITOBA CANADA
140 REM 85/11 HUG-TIBBS
150 ENDEL=1000
160 EXDEL=1000
170 SKIPD=1
180 CALL CLEAR
190 PRINT"PLEASE REMOVE ALPHA LOCK"
200 PRINT
210 PRINT
220 PRINT"PRESS 'C' TO CONTINUE"
230 CALL KEY(3,K,N)
240 IF N=0 THEN 230
250 IF K<>67 THEN 230
260 CALL CLEAR
270 PRINT"PERIMETER CHECK (Y/N)?"
280 CALL KEY(3,L,T)
290 IF T=0 THEN 280
300 IF L=89 THEN 730
310 IF L<>78 THEN 280
320 CALL SOUND(1000,440,0,33,0,5)
330 CALL CLEAR
340 PRINT "UNARMED"
350 CALL JOYST(2,X,Y)
360 IF Y<>4 THEN 350
370 IF SKIPD>1 THEN 400
380 GOSUB 660
390 SKIPD=SKIPD+1
400 CALL JOYST(1,A,B)
410 IF B=-4 THEN 350
420 CALL CLEAR
430 PRINT "ALARM TRIPPED"
440 PRINT
450 PRINT"ENTRY DELAY INITIATED"
460 FOR ENTRDEL=1 TO ENDEL
470 NEXT ENTRDEL
480 CALL JOYST(2,X,Y)
490 IF Y=0 THEN 330
500 FOR LOOP=1 TO 5
510 FOR SIREN=700 TO 900 STEP 10
520 CALL SOUND(-99,SIREN,0)
530 NEXT SIREN
540 FOR SIREN=900 TO 700 STEP -12
550 CALL SOUND(-99,SIREN,0)
560 NEXT SIREN
570 NEXT LOOP
580 CALL CLEAR
590 PRINT"ALERT!!!!!!!"
600 PRINT
610 PRINT
620 PRINT"ALARM TRIPPED"
630 PRINT
640 PRINT "PLEASE RESET"
650 END
660 CALL CLEAR
670 PRINT "EXIT DELAY INITIATION"
680 FOR DELAY =1 TO EXDEL
690 NEXT DELAY
700 CALL CLEAR
710 PRINT "ARMED"
720 RETURN
730 CALL CLEAR
740 CALL JOYST(1,A,B)
750 IF B=-4 THEN 320
760 CALL SOUND(1000,-2,0)
770 PRINT "BREAK IN PERIMETER CIRCUIT"
780 PRINT
790 PRINT "PLEASE CHECK"
800 PRINT
810 PRINT "PRESS 'R' TO RECHECK"
820 CALL KEY(3,K,S)
830 IF S=0 A THEN 820
840 IF K=82 THEN 730
850 IF K<>82 THEN 820
860 END
```