HOCUS

Home Computer
Users Spotlight

a monthly publication of the
Milwaukee Area 99/4 Users Group

<<< JULY 1985 >>>

Official Group Address
4122 No. Glenway
Wauwatosa WI 53222

Officers :

President.....	Jerry Trinkl	327-0170
Vice-President...	James Schroeder	264-4735
Treasurer.....	Phil Norton	462-8954
Secretary.....	Jim Steinhardt	475-9028
Librarians.....	E.J. VonDerEhe	549-0593
	Fred Fabian	327-3618
Newsletter.....	Gene Hitz	453-0499
Assembly SIG.....	Jim Vincent	782-9353
PASCAL Info.....	Gene Hitz &	Jim Vincent

Membership in the Milwaukee Area 99/4 Users Group is open to all interested in using, playing with, learning about or programming in the still-kicking ### Texas Instruments 99/4(A) Home Computer ###

Annual Dues...Individuals - $10.00
Families - $15.00

Meeting dates are the SECOND SATURDAY each month in the lower level of WAUWATOSA SAVINGS & LOAN 7500 W. State Street 1:00 til 4:00 P.M.

PLEASE NOTE

September meeting will be on the THIRD SATURDAY

BASIC COMPILER NOW AVAILABLE

The Compiler has arrived and is available only to members for $10.00. Extended Basic with 32K memory expansion required.

Computer T-Shirts available	$ 5.00
Stand alone RS232 interface	$50.00
Paraprint Parallel interface	$50.00
Gorilla Dot Matrix Printer	$70.00
TI 99/4 Computer Console	$40.00

ASSEMBLY AUTORUN PROGRAMS

by W. A. Molander
Suncoast Beeper Newsletter

Many Assebly Language programs are written so that they automatically execute when loaded. In Assembly language source code this is accomplished by entering the program name in the operand field after the END statement (pg 234 E/A manual) and assembling the program. The program would then execute, beginning at the memory address associated with the Program Name Label. The program name is the same as used in E/A menu option 3 (LoadRun) or option 4 (Run). The reverse is also true that by deleting this operand from the END statement in the source code and reassembling the program, it then won't automatically execute when loaded. The pros and cons of this feature include:

AUTORUN EXECUTION SIMPLIFICATION
EASE OF LOADING MULTIPLE PROGRAMS
AUTORUN DEBUGGING INHIBITED

When source code is not available, object code can still be edited to alter this feature. The following assembly language source code program will be used to illustrate how to edit this feature in UNCOMPRESSED object code only:

```
      DEF   TEST
      REF   VMBW
TX    TEXT  'TRIAL'
TEST  LI    R0,365
      LI    R1,TX
      LI    R2,5
      BLWP  @VMBW
      JMP   TEST
      END   TEST
```


CONT. NEXT PAGE

Assembling the above source code with the E/A using the normal (UNCOMPRESSED: option produces the following object code file.

```
00018       A0000B5452B44941B4C00B0200B00
16DB0201C0000B0202B00057F35AF                    0001
A0012B0420B0000B10F77FB9BF                       0002
200067FED1F                                      0003
50006TEST  30014VMBW  7FADAF                     0004
:       99/4 A5                                  0005
```

The last record of any object code file (line no. 0005) begins with a colon (:) and is followed by the 99/4 A5 identification code (pg 238 E/A manual). Preceding this are the program's external REFERENCES & DEFINITIONS (line 0004)

The AUTORUN feature is contained in the preceding record record (line 0003). This sequence of records is identical for all object code files in either format.

The AUTORUN record format is: TXXXX7YYYYF

The record will contain no other information except line no.

The TAG T designates that XXXX is the Entry Point of the program (see pgs 238, 307 & 309 E/A manual for further info The TAG 7 designates that YYYY is the checksum (pg 229 E/A) for the object code line. F is the end of record marker.

In the above example the relocatable Entry Point is >0006 (line 3 or 5). If the program was loaded beginning at >A000 then the actual memory address of the Entry Point is >A000 + >0006 = >A006.

The checksum >FED1 is the two's complement of 303 arrived at by adding the ASCII value of each character in 2006E7 [2=>32 (50) + 0=>30 (48) + 0=>30 (48) + 0=>30 (48) + 6=>36 (54) + 7=>37 (55) = 303]

Line 4 is the program's REF/DEF record. Tag 5 designates that 0006TEST is a DEFINITION with a relocatable address of 0006 and a symbol of TEST. If line 3 is eliminated then the program can be executed by using the Program Name 'TEST'.

DELETING AUTORUN

To eliminate automatic execution delete the AUTORUN line from the object code file and resave the file. To execute, this saved program will now require the Program Name, the symbol in the REF/DEF record with the same memory address. Autorun can be restored by re-entering the record in the object code file at the exact same line number position and with the same format and content that was deleted.

These changes can best be made with the EDITOR of the E/A program. As the procedure being described applies only to UNCOMPRESSED object code files, be sure the format complies. The screen should be almost filled with ALPHANUMERICS, if there are only two spaces between object code tags (B's etc) and most are blank except for recognizable text, it is a COMPRESSED object code program and cannot be altered. Scroll to the last lines of the program to see the line sequence described above and proceed with the desired changes. DISKO TI-WRITER and DISASSEMBLER can also be used in the same way.

ADDING AUTORUN

Adding Autorun to an object code program is similar to replacing a deleted one except the Entry Point desired must be chosen and the checksum calculated. The checksum will be ignored if the Tag is changed from a 7 to an 8. (E/A pg 241)

The most likely choice is the address of the Program Name used to run the program. In the above example by selecting 50006TEST as the entry address then line 3 would be created and inserted into the object code file as shown. By ignoring the checksum the record 2000680000F could be entered as the record of line 3.

Of the 14 Object Code Tags used by the TMS 9900, 5 relate to Absolute addresses and 5 to Relocatable addresses, pg 240 Programs with absolute addresses use the AORG directive and are placed in a specific memory location. The only effect on the above example would be that Tags 2 & 5 become 1 & 6.

The simplified procedure presented here will work in many cases however complications can arise. These may involve the absence of a DEFINITION in an AUTORUN program or a program too long to fit into the E/A memory. Solutions to problems like this, a list of cautions, and how to alter COMPRESSED object code will be covered in a subsequent article.

PASCAL PROGRAM "DV80TOPAS"

JAMES G. SCHROEDER 06/23/85 MILWAUKEE AREA 99/4 USERS GROUP

THIS PROGRAM IS A AID TO CONVERT DISPLAY VARIABLE 80 FILES TO A PASCAL TEXT FILE. CAUTION MUST BE USED THAT THE DISPLAY FILE STARTS ON AN EVEN NUMBERED SECTOR AND THAT IT IS NOT FRACTURED. WHEN SELECTING AN OUTPUT FILENAME BE SURE TO INCLUDE .TEXT IN YOUR FILENAME OR THE FILE BE OF DATA TYPE.

NOTE : WHEN YOU FIRST BOOT YOUR SYSTEM HAVE THIS PROGRAM ON A DISK IN VOL.#4 YOUR TEXT FILE WILL BE SAVED ON THIS DISK. PUT A SECOND PASCAL DISK IN VOL.#5: SO THE SYSTEM WILL CONFIRM VOL.#5 IS ONLINE. THEN YOU CAN REMOVE THE PASCAL DISK FROM VOL.#5 AND PUT YOUR D/V 80 TEXT FILE IN VOL.#5.

```
{**************************************************************}
{ DEFINITION : A DISPLAY VARIABLE 80 FILE FOR THIS PROGRAM     }
{              IS A FILE THAT WAS LOADED IN THE EDIT ASSEMBLER }
{              EDITOR AND SAVED BACK TO DISK.                  }
{     REASON : TO REMOVE ALL CONTROL CHACTERS FROM TEXT        }
{**************************************************************}

LABEL 90,100,110,120;

CONST ENDSEC = 255 ;

VAR BLOCKNUMBER,COUNT,COUNT1,SECTOR,K,LEN,STRNGLEN :INTEGER;
    PAUSE : CHAR ;
    FILENAME : STRING;
    F : INTERACTIVE ;
    BUFFER1:PACKED ARRAY[0..511] OF CHAR;
    BUFFER2: PACKED ARRAY[0..79] OF CHAR;

BEGIN
      PAGE(OUTPUT);
      WRITELN('CONVERT DV80 TO PASCAL TEXT');
      WRITELN('INSERT DV80 FILE IN DRIVE 2');
      WRITELN('INSERT PASCAL DISK IN DRIVE 1');
      WRITELN('NOTE : ***********************************');
      WRITELN('       *  DISPLAY FILE MUST START AT    *');
      WRITELN('       *  A EVEN SECTOR .               *');
      WRITELN('       *  FILE CANNOT BE FRACTURED.     *');
      WRITELN('       *  NUMBER OF SECTORS IS THE      *');
      WRITELN('       *  NUMBER USED BY DISK MANAGER.  *');
      WRITELN('       ***********************************');
      WRITELN;
      WRITELN('PRESS ANY KEY TO CONTINUE');
      READ(PAUSE);
      PAGE(OUTPUT);
      WRITELN('ENTER FILE NAME FOR OUTPUT');
      WRITELN('EXAMPLE : ******************');
      WRITELN('          *  #4:FILE.TEXT  *');
      WRITELN('          ******************');
      WRITELN;
      READLN(FILENAME);
      REWRITE(F,FILENAME);
      WRITELN('ENTER STARTING SECTOR NUMBER DECIMAL');
      READLN(BLOCKNUMBER);
      BLOCKNUMBER := BLOCKNUMBER DIV 2 ;
      WRITELN('ENTER NUMBER OF SECTORS IN FILE');
      READLN(COUNT);
      COUNT := COUNT-1;
      COUNT1 := 1 ;
 90 : STRNGLEN := 0;
      SECTOR  := 0;
      UNITREAD(5,BUFFER1,511,BLOCKNUMBER,0);
      BLOCKNUMBER := BLOCKNUMBER +1 ;
100 : LEN := ORD(BUFFER1[STRNGLEN]);
      IF LEN = ENDSEC THEN GOTO 110 ;
      MOVERIGHT(BUFFER1[STRNGLEN+1],BUFFER2[0],LEN);
      WRITELN(F,BUFFER2:LEN);
      STRNGLEN := LEN+STRNGLEN+1;
 GOTO 100 ;
 110 : IF COUNT1=COUNT THEN GOTO 120;
      COUNT1 := COUNT1 +1 ;
      IF SECTOR=1 THEN GOTO 90 ;
      SECTOR := 1;
      STRNGLEN := 256 ;
 GOTO 100;
 120 : CLOSE(F,LOCK)
END.
```

SOME FORMATTER TRICKS/ Jean Wilcox

Some time back, Mr. Molander very kindly gave me the information I needed for bringing a Basic or X-Basic program into the word processor, for which many thanks. The way to handle this is to list your program in a file format that can be read by TI- Writer, i.e., LIST "DSK1.program name". Then, using the Text Editor, the program can be loaded into your text buffer, either first or following a given line number of existing text. (A few rude souls nearby suggested it might be to my advantage to read the manual, and I fully intend to. Not today, of course, but sometime very soon). I did encounter a couple of small difficulties and made some gigantic messes before I figured out how to handle the situation. For one thing, if you are planning to print through the Formatter, as I generally do, the programs you draw up out of file will do some really strange things if you forget to use the transliteration commands before printing.

Looking at the screen, I assumed that what I saw was what I would get. Not so. It's easy to forget that the exponent sign for math is also the Required Space sign for text; the "at" key is handy to use for a variable name, but causes the printer to double-strike; the ampersand, the symbol for concatenation of strings, is used to underscore in word processing. A large proportion of programs will use either the exponent or the ampersand, or both, so it's a good idea to transliterate these before you attempt to print them out, unless you want to duplicate my goofs.

Here's something else that Texas Instruments never told us about the T-I Writer. Every time I think I have found all the things that will give trouble printing through the Formatter, I find another one. The newest character to add to the list is the asterisk. Assuming it is to be followed by a space or letter, (or group of spaces or letters), you can print asterisks all day long. You won't have any problems with A*B, 5*C, or Hello*****Stranger. But you will be confounded if you attempt to print one followed directly by a number. As an example, if you need "A*123", what you will get will be "A3".

Part of my wasted time was spent trying to find out why the miserable thing was doing what it was doing. I finally located the one place in the manual where an asterisk is mentioned as having a function, rather than as just another character to be printed. It's on pages 111-113, listed under Alternate Input, Mail Merge Option. (How many of you send out form letters!) If you feel like getting really technical about the thing, read the part about Define Prompt, too. It's on the same group of pages. Even after finding this I still didn't immediately associate the Mail Merge with the problem I was having, since the command necessary to carry out the job in a form letter is described as "*n*", an asterisk sandwich with a number in the middle. So I just quit worrying about the Why of the situation and started in working on the What To Do About It.

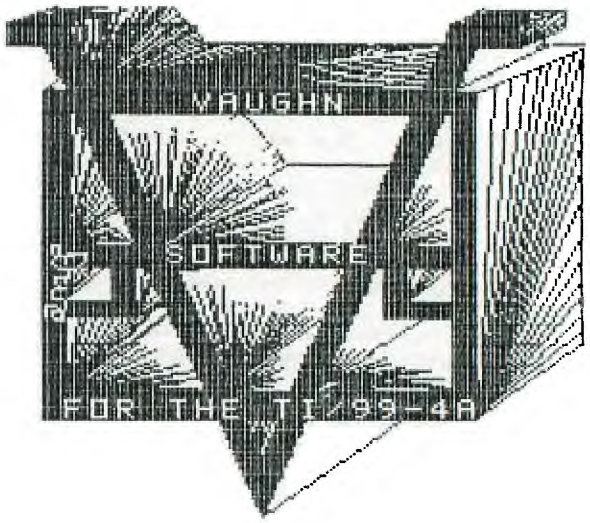
I must not be too swift because it took another hour of typing all sorts of strange stuff containing asterisks in odd configurations to realize that I could always get what I wanted if a space immediately follows the asterisk. In a formula it will look lop-sided, (since the space is printed, too), so, as a dedicated neat freak, I'm typing a space before and after it. It's just as easy to remember that as the other, and the results look as though it were what you had intended all along.

A day or two later, Irene called to say she had information on this from the head guru, Guy-Stefan Romano. He explained that it was indeed the Mail Merge Option that was the culprit. It seems that when the TI-Writer encounters an *, it looks for one or two numbers for the Value File needed for the form letter,

then proceeds to strip out the asterisk and the numbers following it. His solution was to print two asterisks, followed by two dummy numbers, one space, and then the figure you want printed. This works well, but you're going all the way around your elbow to get to your thumb. Why not just put a space fore and aft and carry on?

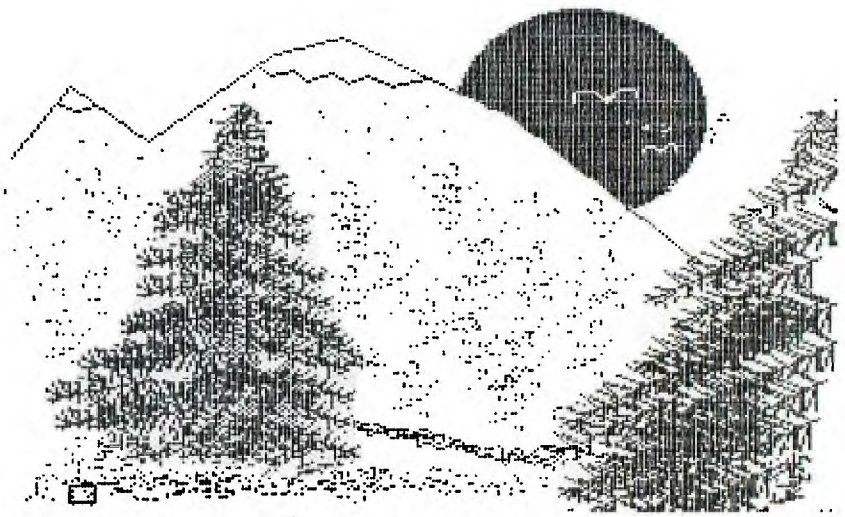
Another thing that caused me trouble also springs from from my neatness fetish. When the various program lines are formatted they get drawn all up in a knot....if there's room on a line of type to print something, it will get printed, whether you want it there or not. The obvious answer to this, of course, is to add a carriage return at the end of each program line. "Enter" is supposed to do this, but for me it's undependable. If the line is short it might work, then again it might not. I found that if I enter the CR symbol by using CTRL "New Paragraph" I was in business. This causes a blank line to be inserted after each existing line, but they are easily deleted. There are probably dozens of ways, all shorter, simpler, and more efficient, to accomplish these things, but they work for me. So until I get around to reading the book, this is the way I'm going to do it.

BITMAC disk media software was designed for use with the TI-99/4A home computer. It is written in TMS9900 machine language for the utmost in speed and program function. The program provides bit precision graphics generation and editing. Some of the features are:



Line, rectangle, circle, copy section, mirror, rotate, reverse video, free hand draw, 9 brush sizes, 16 colors, bit "on" color, bit "off" color, screen color, color test area, 40 column text, text on text, text on graphics, 16 color text, upper and lower case, 4 direction bit scrolling, dump to printer (two sizes), save to disk, boolean graphics enhancement, "LIFE" graphics enhancement, second computer input, X Y vector reporting, monochrome and color monitor support, trackball support, single bit erase, single bit placement, block erase, erase colors, erase all, fill, enlarge, reduce, "slide show", "oops" function.

BITMAC is icon driven and is simple to use. Included are extensive documentation, an example coprocessor program and one year warranty.



REQUIRED EQUIPMENT: Extended basic, Mini Memory or Editor Assembler module, a display monitor, joysticks, TI-99/4A computer, memory expansion and a disk drive system. THE PROGRAM MAY NOT BE COMPATABLE WITH SOME VERSIONS OF THE MYARC DISK CONTROLLER.

OPTIONAL EQUIPMENT: TI, Gemini or Epson printer, RS232 card, trackball, up to 5 disk drives (limited by the disk controller), second computer (any make) with RS232 interface and cable.

To Order: Send check or money order for $29.95 plus $2.00 shipping and handling to:

VAUGHN SOFTWARE
5460 Harlan #84
Arvada, CO 80002

For other inquiries please include a self addressed stamped envelope.

BITMAC

By Vaughn Software

For the

TI-99/4A COMPUTER

TIPS FROM THE TIGERCUB

#22

TIGERCUB SOFTWARE
156 Collingwood Ave.
Columbus, OH 43213

The entire contents of Tips from the Tigercub Nos. 1 through 14, with more added, are now available as a full disk of 50 programs, routines and files for just $15.00 postpaid!

Nuts Bolts is a diskfull of 100 (that's right, 100!) XBasic utility subprograms in MERGE format, ready for you to merge into your own programs. Contents include 13 type fonts, 14 text display routines, 12 sorts and shuffles, 9 data saving and reading routines, 9 wipes, 8 pauses, 6 music, 2 protection, etc., and now also a tutorial on using subprograms, all for just $19.95 postpaid!

And I have about 140 other absolutely original programs in Basic and XBasic at only $3.00 each!(plus $1.50 per order for casette, packing and postage, or $3.00 for diskette, PPM) Some users groups charge their members that much for public domain programs! I will send you my descriptive catalog for a dollar, which you can then deduct from your first order.

This challenge was printed in Tips #21 -

```
100!The Unprintable Unkeyabl
e Program!
110!To shuffle the numbers 1
 to 255 into a random sequen
ce without duplication
120!The strings contain the
ASCII characters 1 to 127 an
d 128 to 255
130!Most of the ASCII charac
ters below 32 or above 159 c
annot be input from the keyb
oard
140!So how was this program
programmed?
150 M$="
          !"#$%&'()*+,-./0
123456789:;<=>?@ABCDEFGHIJKL
MNOPQRSTUVWXYZ[\]^_`abcdefgh
ijklmnopqrstuvwxyz{|}~"
160 M2$="



                    "
170 M$=M$&M2$
180 L=LEN(M$):: RANDOMIZE ::
 X=INT(L*RND+1):: N=ASC(SEG$
(M$,X,1)):: M$=SEG$(M$,1,X-1
)&SEG$(M$,X+1,LEN(M$))
190 PRINT N;:: IF LEN(M$)=0
THEN STOP ELSE 180
```

And here is the answer - It was written by a program that writes a program! key this in and run it to create a MERGE format disk file. Then type NEW, then type MERGE DSK1.LONGSTRING and you will have a RUNable program consisting of lines 150-170 of the puzzle!

```
100 OPEN #1:"DSK1.LONGSTRING
",VARIABLE 163
110 LN=100 :: GOSUB 190 :: A
$=L$&"M$"&CHR$(190)
120 FOR J=1 TO 127 :: C$=C$&
CHR$(J):: NEXT J :: A$=A$&CH
R$(199)&CHR$(127)&C$&CHR$(0)
130 PRINT #1:A$
140 GOSUB 190 :: B$=L$&"M2$"
&CHR$(190)
150 FOR J=128 TO 255 :: D$=D
$&CHR$(J):: NEXT J :: B$=B$&
CHR$(199)&CHR$(128)&D$&CHR$(
0)
160 PRINT #1:B$
170 GOSUB 190 :: F$=L$&"M$"&
CHR$(190)&"M$"&CHR$(184)&"M2
$"&CHR$(0)
180 PRINT #1:F$ :: PRINT #1:
CHR$(255)&CHR$(255):: CLOSE
#1 :: END
190 L$=CHR$(INT(LN/256))&CHR
$(LN-256*INT(LN/256)):: LN=L
N+10 :: RETURN
```

Now type in the remaining lines, and you will have a speeded-up version of the Tigercub Scramble which was published in Tips #10. It is still not as fast as the CALL PEEK versions but is much more useful because you can modify it to scramble a sequence of any length anywhere between 1 and 255. For example, to shuffle the numbers 100 to 150 into a random sequence without duplication, just add a line 175 M$=SEG$(M$,100,50).

The method of writing a "program that writes a program" was fully explained by John Clulow in the 99er magazine Vol. 1 Nos. 3 and 4. It is a little-used but very valuable technique.

For instance, Tips#9 contained the following routine to turn the alphabet upside-down.

```
100 FOR CH=33 TO 127 :: CALL
 CHARPAT(CH,CH$):: FOR J=1 T
O 16 STEP 2 :: X$=SEG$(CH$,J
,2)&X$ :: NEXT J :: CALL CHA
R(CH,X$):: X$="" :: NEXT CH
110 INPUT A$ :: GOTO 110
```

The only trouble with that is that it takes about 50 seconds to run. Try this instead -

```
100 FOR CH=33 TO 127 :: CALL
 CHARPAT(CH,CH$):: FOR J=1 T
O 16 STEP 2 :: X$=SEG$(CH$,J
,2)&X$ :: NEXT J :: CALL WRI
TE(CH,X$):: X$="" :: NEXT CH
1000 SUB WRITE(CH,X$):: IF F
LAG=1 THEN 1010 :: FLAG=1 ::
 OPEN #1:"DSK1.WRITE",OUTPUT
,DISPLAY ,VARIABLE 163 :: LN
=3000 :: GOSUB 3000
1010 X=X+1 :: L$=L$&CHR$(200
)&CHR$(16)&X$ :: IF X<5 AND
CH<127 THEN L$=L$&CHR$(179):
: SUBEXIT
1020 X=0 :: PRINT #1:L$&CHR$
(0):: L$="" :: IF CH=127 THE
N 1030 :: GOSUB 3000 :: SUBE
XIT
1030 PRINT #1:CHR$(255)&CHR$
(255):: CLOSE #1 :: GOTO 301
0
3000 L1=INT(LN/256):: L2=LN-
256*L1 :: L$=CHR$(L1)&CHR$(L
2)&CHR$(147):: LN=LN+10 :: R
ETURN
3010 SUBEND
```

RUN that, type NEW, then MERGE DSK1.WRITE, and you will have a program consisting of DATA statements containing the hex codes for all the upside-down characters. Add a line 100 FOR CH=33 TO 127 :: READ CH$ :: CALL CHAR(CH,CH$):: NEXT CH, and you can turn everything upside-down in only 12 seconds.

Someone sent me a classified ad, clipped from an unknown publication, which read -

TI-WRITER COMPANION. Loaded with ingenious ways to make your TI-Writer more effective. Well written. Send $2.50 to Dr. Bill Browning, 7541 Jersey Avenue North, Brooklyn Park, MN 55428. Money back guarantee.

I sent off my money and have just received 29 pages, 3-hole punched, loaded with useful and ingenious tips and ideas for getting more out of TI-Writer. I recommend it - it's worth twice the money and then some!

The K-Town newsletter recently published a utility routine that is so useful that I want to pass it on to everyone. If a program is not resequenced after it is modified, this will compare

it with the original and prepare a MERGE format file of all the changes, for the use of others to update their copy.

```
100 !%%%%%%%%%%%%%%%%%%%%%%%%
110 !%  COMPARE PROGRAM      %
120 !     by Mike Dodd       %
130 !%%%%%%%%%%%%%%%%%%%%%%%%
131 ! In k-Town 99'er V.2 #1
      April 1985
140 !Version 85.0406.1XB
      Requires disk drive.
      Compares two programs,
      gives list of all diff-
      erences.
150 !SAVE old program in
 MERGE format (SAVE DSK1.(ol
 dfilename),MERGE). SAVE up-
 dated program in MERGE for-
 mat(SAVE DSK1.(newfilename)
,MERGE)
160 !RUN this program, answe
 r prompts for OLD FILE name,
 NEW FILE name, and a differ
ent OUTPUT FILE name.
170 !When finished, type NEW
, then MERGE DSK1.(outputfil
ename) and ENTER
180 !Can be MERGED into othe
r copies of OLD program to
 update them
190 DEF @(@$)=ASC(SEG$(@$,1,
1))*256+ASC(SEG$(@$,2,1))
200 A$=CHR$(255)&CHR$(255)::
 DISPLAY AT(1,1)ERASE ALL:"O
LD    FILE:": :"NEW    FILE:
": :"OUTPUT FILE:"
210 ACCEPT AT(1,13)BEEP:B$ :
: ACCEPT AT(3,13)BEEP:C$ ::
ACCEPT AT(5,13)BEEP:D$ :: OP
EN #1:B$,INPUT ,VARIABLE 163
220 OPEN #2:C$,INPUT ,VARIAB
LE 163 :: OPEN #3:D$,OUTPUT,
VARIABLE 163
230 LINPUT #1:@$ :: LINPUT #
2:E$ :: F$=SEG$(@$,1,2):: G$
=SEG$(E$,1,2):: A=@(F$):: B=
@(G$)
240 IF F$=A$ AND G$=A$ THEN
CLOSE #1 :: CLOSE #2 :: PRIN
T #3:A$ :: CLOSE #3 :: STOP
250 IF B>A THEN PRINT #3:F$&
CHR$(131)&" **DELETED LINE *
*"&CHR$(0):: LINPUT #1 :: @$
 :: F$=SEG$(@$,1,2):: A=@(F$
):: GOTO 240
260 IF A>B THEN PRINT #3:E$
:: LINPUT #2:E$ :: G$=SEG$(E
$,1,2):: B=@(G$):: GOTO 240
270 IF @$<>E$ THEN PRINT #3:
E$
280 GOTO 230
```

Thanks to some ideas from Joyce Corker, I have made some more improvements to the Tigercub Menuloader, and I have used the above utility routine to list all the changes made since it was published in Tips#15.

```
100 !by A. Kludge/M. Gordon/
T. Boisseau/J. Peterson/etc.
modified in Tips #22
102 OPTION BASE 1 :: DIM PG$
(127),VV(127),VX(127):: GOTO
 110
105 @,A,A$,B,C,D$,FLAG,I,J,K
,KD,KK,N$,NN,P$,PG$(),Q$,S,S
T,T$(),TT,VT,VV(),VX(),W$,X,
X$,K2,S2
106 CALL INIT :: CALL LOAD :
: CALL LINK :: CALL PEEK ::
CALL KEY :: CALL SCREEN :: C
ALL COLOR :: CALL CLEAR :: C
ALL VCHAR :: CALL SOUND :: !
@P-
150 ! **DELETED LINE **
160 T$(1)="d/f" :: T$(2)="d/
v" :: T$(3)="1/f" :: T$(4)="
1/v" :: T$(5)="pro" :: ON WA
RNING NEXT
170 IMAGE ###
180 DISPLAY AT(1,4):"TIGERCU
B MENU LOADER"
210 D$="DSK1." :: OPEN #1:D$
,INPUT ,RELATIVE,INTERNAL ::
 INPUT #1:N$,A,J,K :: DISPLA
Y AT(1,2)SIZE(27):SEG$(D$,1,
4)&" - Diskname= "&N$;
230 FOR X=1 TO 127 :: IF X/2
0<>INT(X/20)THEN 260
240 DISPLAY AT(24,1):"Type c
hoice or 0 for more 0" :: AC
CEPT AT(24,27)VALIDATE(DIGIT
)SIZE(-3):K :: IF K=0 THEN 2
50 :: IF VV(K)<>5 THEN 411 :
: IF K>0 AND K<NN+1 THEN 420
 ELSE 240
290 DISPLAY AT(X+4,2):USING
170:NN :: DISPLAY AT(X+4,6):
P$ :: PG$(NN)=P$ :: DISPLAY
AT(X+4,18):USING 170:J :: DI
SPLAY AT(X+4,22):T$(ABS(A))
291 VV(NN)=ABS(A):: VX(NN)=A
BS(B)
295 X$="  "&STR$(B):: DISPLA
Y AT(X+4,26):SEG$(X$,LEN(X$)
-2,3):: VT=VT+J
350 DISPLAY AT(X+6,1):"  C
hoice?" :: ACCEPT AT(X+6,16)
SIZE(3)VALIDATE(DIGIT):K ::
IF K<>NN AND K<>NN+1 THEN 41
0
410 IF K<1 OR K>127 OR LEN(P
G$(K))=0 THEN 320
411 IF VV(K)=5 OR(VV(K)=4 AN
D VX(K)=254)THEN 420
412 ON ERROR 417 :: CALL CLE
AR :: OPEN #2:D$&PG$(K):: CA
LL SCREEN(16)
413 LINPUT #2:W$ :: IF EOF(2
)THEN 416 :: PRINT W$
414 CALL KEY(0,K,S):: IF S=0
 THEN 413
415 CALL KEY(0,K2,S2):: IF S
2<1 THEN 415 ELSE 413
416 CLOSE #1 :: CLOSE #2 ::
END
417 DISPLAY AT(12,10):"UNLIS
TABLE" :: CALL SOUND(200,110
,0):: RETURN 400
430 ON ERROR 417 :: CALL INI
T :: CALL PEEK(-31952,A,B)::
 CALL PEEK(A*256+B-65534,A,B
):: C=A*256+B-65534 :: A$=D$
&PG$(K):: CALL LOAD(C,LEN(A$
))
```

The Menu Loader will now list up to 127 programs and files, showing the number of sectors in each and the file type, record type and record length of each file. It will stop at the end of each page, and continue on a default value of 0, or will stop for selection when any key is pressed. It gives disk name, number of sectors used and available. It adds up sectors actually used and gives a warning if all sectors are not accounted for. It will load and run any program which can be loaded from Extended Basic, displaying the program being loaded. It will delete any program or file, after first displaying the filename and requesting verification. It will list any listable file to the screen, pausing on any key input, and can be very easily modified to list to a printer. If a file is not listable, it will inform you so, and restart the menu selection. It has the pre-scan option to speed it up.

Fairly often, the disk directory will lose track of one or a few sectors during the process of loading records, even though the Disk Manager showed all 358 were initialized. That's why I put the checking routine in the Menu Loader. The figure shown as "used" is actually 358 minus the number of sectors still available, and is checked against the total sectors of all files.

The loss of a few sectors is no serious matter, but once in a great while you may notice that the "available" and "used" sector quantities have obviously been reversed. I have found that this is a signal that the disk is about to go haywire and you had best back it up immediately!

Programs and files are loaded in the first available sector, and continued in the next available sector. If a number of small files are deleted from a disk, and a long file is then loaded, it may thus be fractured into many parts. If you have a work disk on which you continually add and delete files of various lengths, it will become badly fractured. This can cause disk errors, and it also badly overworks your drive. It is a good idea to recopy your work disk occasionally - file by file, not sector by sector with a quick copier.

MEMORY FULL! - Jim Peterson

```
50 !
   ! S P A C E - P I R A T E
   !
-------------------------------
100 Q=600 :: CALL SCREEN(11)
:: CALL CHAR(120,"FF241800FF
DB5AFF"):: CALL COLOR(12,16,
7):: CALL HCHAR(1,1,120,768)
-------------------------------
110 DISPLAY AT(8,8)SIZE(14):
" SPACE-PIRATE " :: INPUT "
USING JOYSTICKS??(Y/N) ":J$
-------------------------------
120 DISPLAY AT(3,1):"OBJECT:
": :" DESTROY ENEMY SHIPS FO
R      POINTS...AND FUEL TO P
OWER  YOUR OWN SPACECRAFT."
-------------------------------
130 DISPLAY AT(16,1):" RADAR
 WILL LOCK ON TARGET   MOMEN
TARILY,":" THEN PRESS ANY KE
Y TO FIRE." :: DISPLAY AT(23
,1):"NOW PRESS ANY KEY TO BE
GIN !"
-------------------------------
140 IF J$="Y" THEN 160
-------------------------------
150 DISPLAY AT(10,1):" USE A
RROW KEYS TO AIM AND  CENTE
R YOUR TARGET IN THE  CROSS
-HAIRS." :: GOTO 170
-------------------------------
160 DISPLAY AT(10,1):" CENTE
R YOUR TARGET AT THE  CROSS
HAIRS !"
-------------------------------
170 CALL KEY(0,F,G):: IF G<1
 THEN 170
-------------------------------
180 CALL CLEAR :: CALL SCREE
N(2)
-------------------------------
190 DISPLAY AT(1,1):"POINTS
0" :: DISPLAY AT(1,18):"FUEL
 100" :: DISPLAY AT(24,1):"
NO. OF TORPEDOS LEFT 50" ::
RANDOMIZE :: FOR H=1 TO 8 ::
 CALL COLOR(H,16,2):: NEXT H
 :: CALL MAGNIFY(3)
-------------------------------
200 CALL CHAR(36,"0103030001
0160ED6001010003030100008080
0000000C7C0C000000808",40,"8
303200109030707C703080120038
300848008002080C6D6C68020000
88082")
-------------------------------
210 CALL CHAR(100,"00201F3FC
CCC7C7C3FFF3E0B100000000004F
8FCCFCF3232FCFC101008",104,"
80404020100C0F7FFFEFE3FC7E38
2800010202040830F0FEFFF7C73F
7E1C14")
-------------------------------
220 CALL CHAR(108,"040804020
10F9FDCFFCF3C01030F0C0020102
040B0F0F93BFFF3C180C0F03",11
2,"C07F3FFF:E1F0F0F0F0F3F2A0
0150015(3FFFEEE50F0E0E0E0E0F
8A80050005")
-------------------------------
230 CALL CHAR(96,"202010100F
0281F3FFFF0F093071F0000040408
0EFF10F8FCFF0F0FC9E0F8",140,
"00000000000000010100000000
0000000000000000000808")
-------------------------------
240 CALL CHAR(116,"070C18306
040414140603018OC070000E0301
80C0602828202060C1830E",140,
"00000000000001010000000000
0000000000000000080800")
-------------------------------
250 CALL CHAR(120,"000304081
1222121201008040300000000C02
0100804848404081020C",136,"0
0000000000010202010000000000
0000000000008040408")
-------------------------------
260 CALL CHAR(124,"000003040
810111110E0403000000000000C
0201008888081020C",128,"000
00003060C09090C0603000000000
0000000C0603090903060C")
-------------------------------
270 CALL CHAR(132,"000000000
1020505020100000000000000000
0008040A0A0408")
-------------------------------
280 Z,Y=1 :: FOR I=3 TO 14 :
: CALL SPRITE(#I,140,16,INT(
256*RND+1),INT(256*RND+1),Z,
Y):: NEXT I
-------------------------------
290 CALL SPRITE(#1,36,7,90,1
20):: C=0 :: D=100 :: E=50
-------------------------------
300 L=INT(192*RND)+1 :: M=IN
T(256*RND)+1
-------------------------------
310 P=INT(5*RND)+1 :: R=INT(
2.5*P+1):: P=92+4*P
-------------------------------
320 CALL SPRITE(#2,110,R,L,M
):: CALL SOUND(1,500,0)
-------------------------------
330 FOR T=132 TO 116 STEP -4
 :: CALL PATTERN(#2,T):: CAL
L SOUND(1,1610-10*T,0):: NEX
T T :: CALL PATTERN(#2,P)
-------------------------------
340 IF J$<>"Y" THEN 360
-------------------------------
350 CALL JOYST(1,K,S):: IF K
=0 AND S=0 THEN 355 ELSE Y=K
 :: Z=-S
-------------------------------
355 CALL COINC(#1,#2,8,W)::
IF W=0 THEN 465 ELSE 370
-------------------------------
360 CALL KEY(0,U,V):: CALL C
OINC(#1,#2,8,W):: IF W=0 THE
N 430
-------------------------------
370 CALL MOTION(#2,0,0):: CA
LL LOCATE(#2,90,120):: DISPL
AY AT(24,1):"*RADAR LOCKED O
N TO TARGET*"
-------------------------------
380 FOR A=0 TO 5 :: CALL KEY
(1,U,V):: IF V<>0 THEN 510 E
LSE CALL SOUND(100,700,0)
-------------------------------
390 NEXT A
-------------------------------
400 CALL MOTION(#2,INT(100*R
ND),INT(100*RND))
-------------------------------
410 DISPLAY AT(24,1):" NO. O
F TORPEDOS LEFT= ";E
-------------------------------
430 IF U=83 THEN Y=-4 :: Z=0
-------------------------------
440 IF U=68 THEN Y=4 :: Z=0
-------------------------------
450 IF U=69 THEN Z=-4 :: Y=0
-------------------------------
460 IF U=88 THEN Z=4 :: Y=0
-------------------------------
465 D=D-1 :: DISPLAY AT(1,23
):D :: IF D=0 THEN 590
-------------------------------
470 CALL POSITION(#2,L,M)::
IF L<10 OR L>180 OR M<10 OR
M>246 THEN 740
-------------------------------
480 L=INT(8-9*RND):: M=INT(8
*RND)+1 :: CALL MOTION(#2,L-
M,M-L):: N=INT(50*RND)+1 ::
IF N=1 THEN 660 :: IF N=2 TH
EN 700
-------------------------------
490 FOR N=2 TO 14 :: CALL MO
TION(#N,-Z,-Y):: NEXT N
-------------------------------
500 IF J$<>"Y" THEN 360 ELSE
 350
-------------------------------
510 E=E-1 :: DISPLAY AT(24,1
):" NO. OF TORPEDOS LEFT= ";
E :: CALL SPRITE(#15,116,12,
190,119,-45,0)
-------------------------------
520 FOR N=116 TO 140 STEP 4
:: CALL SOUND 20,-7,0):: CAL
L PATTERN(#15,N):: NEXT N
-------------------------------
530 CALL DELSPRITE(#15):: CA
LL COINC(#1,#2,10,N):: IF N=
0 THEN IF E=0 THEN 590 ELSE
IF J$<>"Y" THEN 360 ELSE 350
-------------------------------
540 FOR N=0 TO 5 :: CALL SOU
ND(30,-5,N):: NEXT N :: CALL
 PATTERN(#2,40):: CALL COLOR
(#2,16):: CALL COLOR(#2,7)::
 CALL COLOR(#2,16)
-------------------------------
550 DISPLAY AT(24,1):"
 YOU GOT HIM!" :: CALL COLO
R(#2,1):: CALL COLOR(#2,7)::
 CALL COLOR(#2,16):: CALL DE
LSPRITE(#2):: FOR N=1 TO 200
 :: NEXT N
-------------------------------
560 DISPLAY AT(24,1):" NO. O
F TORPEDOS LEFT= ";E :: IF E
=0 THEN 590 ELSE 580
-------------------------------
570 CALL SOUND(-1000,-7,1)::
 CALL PATTERN(#2,99):: CALL
SCORE -10,-5,5)
-------------------------------
580 C=C+R+10-A :: D=D+R-A ::
 DISPLAY AT(1,8)SIZE(4):C ::
 DISPLAY AT(1,23):D :: GOTO
300
-------------------------------
590 CALL MOTION(#1,0,0,#2,0,
0):: DISPLAY AT(24,1):" NO.
OF TORPEDOS LEFT= ";E
-------------------------------
600 DISPLAY AT(9,10):"GAME-O
VER" :: IF E=0 THEN DISPLAY
AT(11,7):"OUT OF TORPEDOS" E
LSE IF D=0 THEN DISPLAY AT(1
1,9):"OUT OF FUEL"
-------------------------------
610 FOR N=1 TO 200 :: NEXT N
 :: IF C>X THEN X=C
-------------------------------
620 DISPLAY AT(14,1):"  TODA
Y'S HIGH SCORE ";X :: DISPLA
Y AT(16,4):"ALL-TIME HIGH...
.";Q
-------------------------------
630 IF X>Q THEN Q=X :: DISPL
AY AT(18,7):"WOW !!  WOW !!"
: :"   A NEW ALL-TIME HIGH !
!    PLEASE CHANGE VARIABLE
'Q'  IN LINE 100 TO..";Q
-------------------------------
650 DISPLAY AT(24,1):"TYPE "
"Y"" TO PLAY AGAIN" :: CALL
KEY(0,U,V):: IF U=89 THEN 18
0 ELSE IF V<1 THEN 650 ELSE
END
-------------------------------
660 IF RND<.5 THEN N=0 ELSE
N=-1
-------------------------------
670 CALL MOTION(#2,N,0):: FO
R N=124 TO 136 STEP 4
-------------------------------
680 CALL PATTERN(#2,N):: CAL
L SOUND(10,500-2*N,0):: NEXT
 N
-------------------------------
690 CALL SOUND(100,110,0)::
DISPLAY AT(24,1):"SHIP WARPE
D OUT OF SECTOR!!" :: CALL D
ELSPRITE(#2):: FOR N=1 TO 30
0 :: NEXT N :: DISPLAY AT(24
,1):" NO. OF TORPEDOS LEFT=
";E :: GOTO 300
-------------------------------
700 CALL MOTION(#2,INT(40*RN
D),INT(40*RND))
-------------------------------
710 FOR N=124 TO 136 STEP 4
:: CALL PATTERN(#2,N):: CALL
 SOUND(20,500-2*N,2):: NEXT
N
-------------------------------
720 FOR N=136 TO 124 STEP -4
 :: CALL PATTERN(#2,N):: CAL
L SOUND(20,500-2*N,3):: NEXT
 N
-------------------------------
730 CALL PATTERN(#2,P):: GOT
O 490
-------------------------------
740 CALL DELSPRITE(#2):: GOT
O 300
-------------------------------
```

REPRINT

****

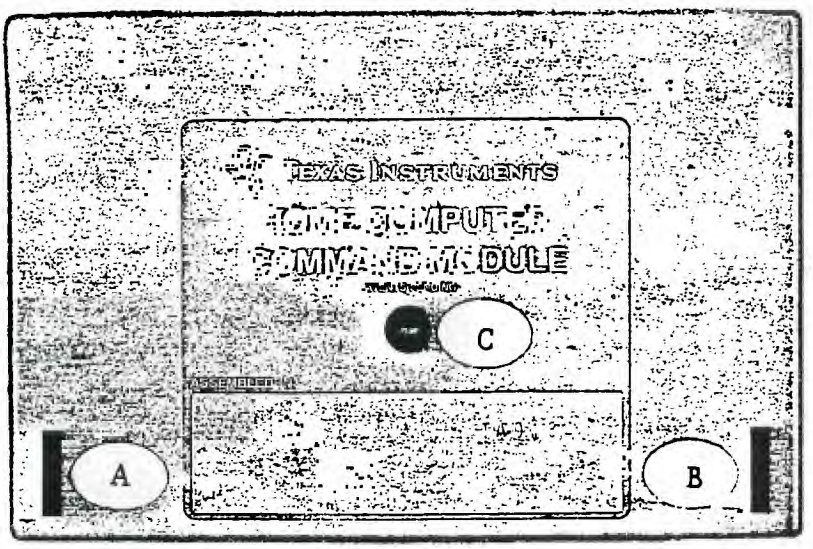
*CIUG:0485 CARTRIDGE CLEANING - by Ron Rutledge *

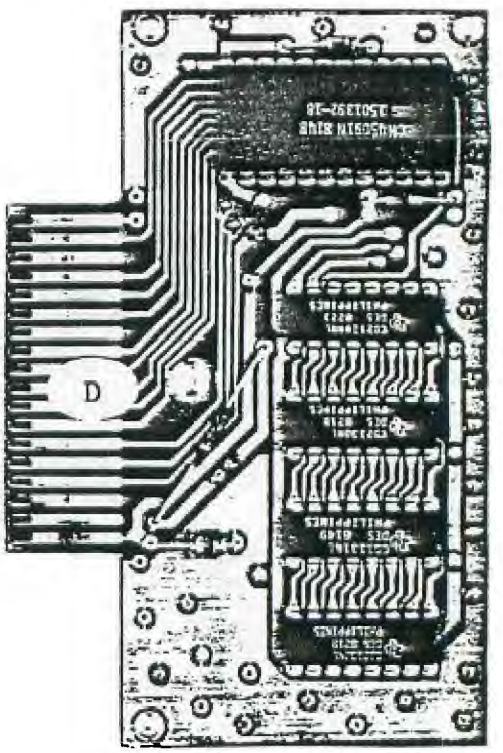
****

Dirty contcts can screw-up any electrical device and the 4A is not an exception. The only place you are fairly likely to run into this problem is in using command modules. Both the module contacts and the port itself can become dirty but cleaning the port itself is a big job as you have to disassemble the console. The good news is that cleaning the cartridge will almost always suffice and can be done quickly without any special tools or cleaners. All you need is a regular screwdriver, some sort of rag, a standard pencil eraser, and in some cases a medium phillips screwdriver.

Remove the screw from "C" if there is one. Then pry the clips in slots "A" and "B" outward to pop open the cartridge. If there is a clip in "C" pry it back after "A" and "B" are loose. If it should bend off don't worry, it won't affect the performance of your module.

The module board can now be removed. Do this carefully and note how the spring-loaded "door" is assembled if there is one so that you can put it back together if it pops out. Once you have the board removed take your rag (a kleenex will work but something cloth is much better) and rub off any residue from the contacts, shown as "D". Remember to do the contacts on both sides if that particular module has them. Once the worst is removed take any soft rubber eraser and "erase" the contacts until they become dry, clean and shiny. You need to do only about the outer half of the contacts as that is more than ever gets used (you can see the scratch marks in the picture below). Once this is done simply put the cartridge back together and go. Some symptoms of dirty contacts are the console locking-up, strange errors where no occured before, etc (my XB cartridge giving me a syntax error when there was non for example). Don't jump to clean a cartridge on your first error, it could be alot of things like static, not having the module in tight, or a number of other things. But if you find you have a continuing problem cleaning the contacts is quick and free and may correct what was wrong.

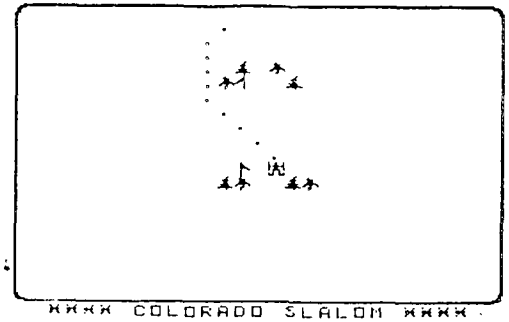




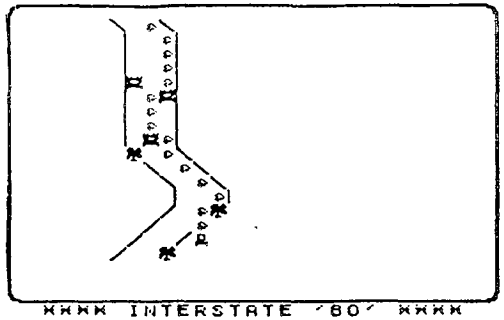
PROGRAM INNOVATORS

ARCADE ACTION SOFTWARE

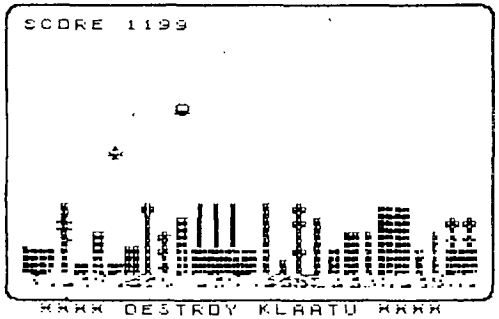
Arcade Action Software
2007 N. 71
Wauwatosa, WI 53213



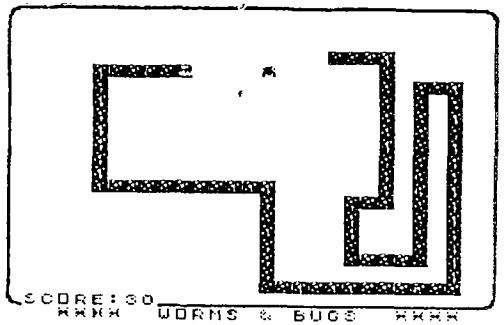
**** COLORADO SLALOM ****



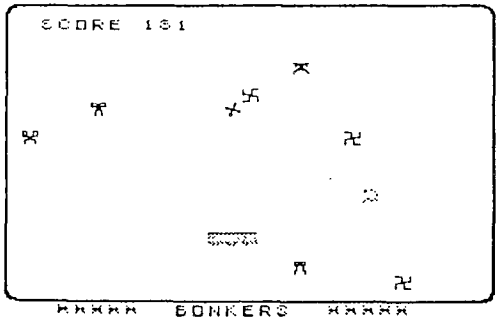
**** INTERSTATE '80' ****



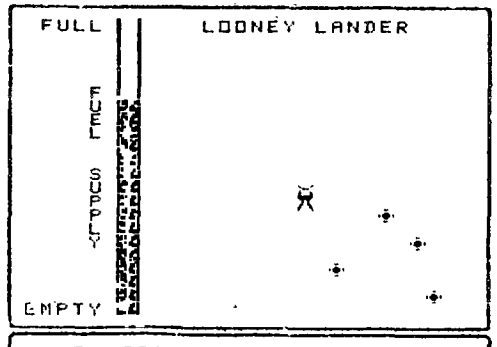
**** DESTROY KLAATU ****

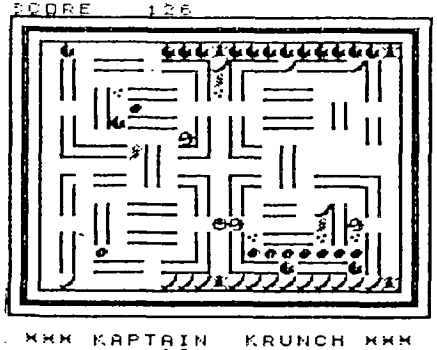


**** WORMS & BUGS ****

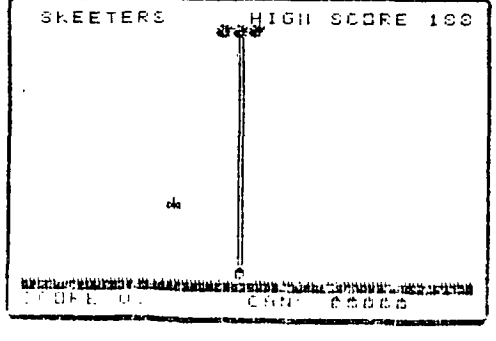


****** BONKERS ******





*** KAPTAIN KRUNCH ***



T.I. BASIC

DESTROY KLATU
laser-blast the empire
evade the drone defenders

COLORADO SLALOM
ski the giant slalom
don't collide with the trees

INTERSTATE '80'
cross country on the interstate
watch out for the semis

BONKERS II
catch the crazy bonkers
in your little bonker-net

ZOMBIES GALORE
the zombies will get you
if you don't watch out

KAPTAIN KRUNCH
race against the munchkins
thru the maze of fruit

CATZ N' MOUSE
while the cat's away
the mouse will play

ROBOTHELLO
play othello against the
grandmaster computer

BUGS & WORMS
gobble up the elusive bugs
with your ever-expanding worm

LOONEY-LANDER
try to land softly on the moon
dodging space quarks

MISSILE TRACKER
defend U.S. cities from
ICBM missile attack

DON'T FENCE ME IN
2 players try to trap
each other...& GOMOKU

BOMBSCARE
defuse the hidden bomb
before we all blow up

DIET RITE
diet evaluator, calorie count
& D.M.R. analyzer

HOME VALUE
appraisal, amortization,
& rental comparison

SKEETERS & SKEETERS
spray the pesky skeeters
cuz skeeters byte

PRICES:

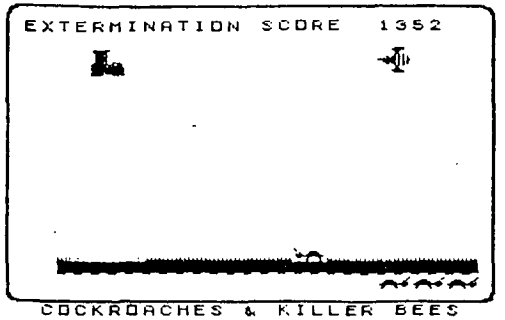
Cassette tape version,
First selection.....$
Each additional one...$
Disk version.....add $

PROGRAM INNOVATORS

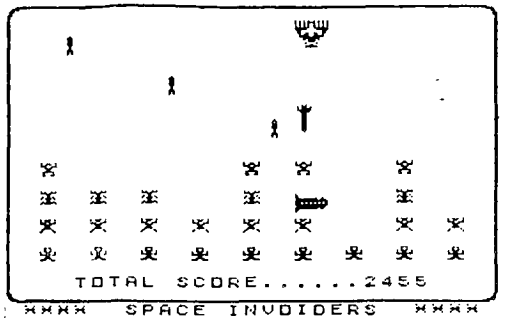
ARCADE ACTION SOFTWARE

Arcade Action Software
2007 N. 71
Wauwatosa, WI 53213

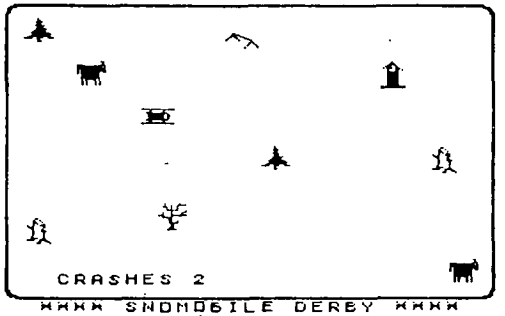
EXTENDED BASIC



COCKROACHES & KILLER BEES

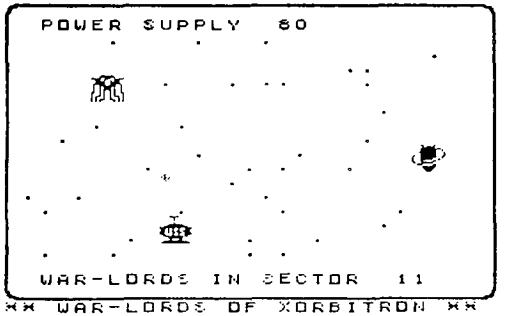


XXXX SPACE INVDIDERS XXXX

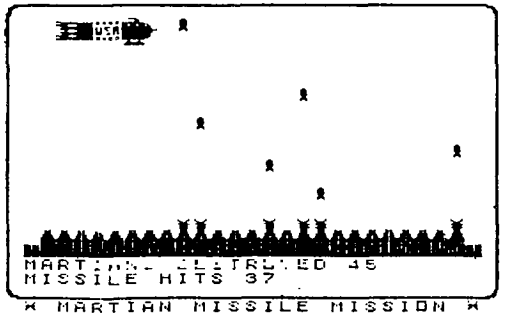


XXXX SNOMOBILE DERBY XXXX

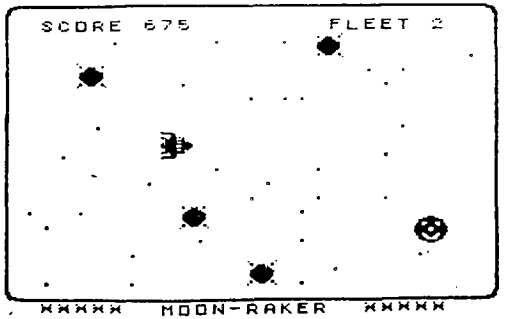
XXX KLINGON ENCOUNTERS XXX



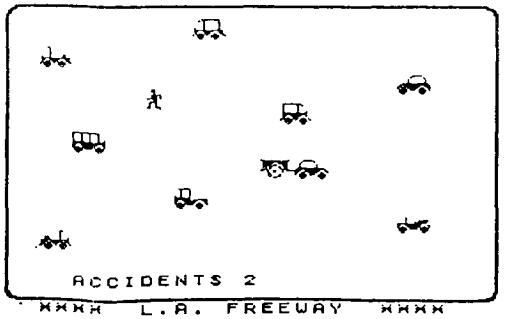
XX WAR-LORDS OF XORBITRON XX



X MARTIAN MISSILE MISSION X



XXXXX MOON-RAKER XXXXX



XXXX L.A. FREEWAY XXXX

MOONRAKER
operate moon shuttle
thru meteorite storm

GALACTIC GUARDIAN
destroy lizzard aliens
infiltrating our galaxy

SNOMOBILE DERBY
cross country race on snomobile
thru hazardous fields

L.A. FREEWAY
cross the L.A. freeway
during rush hour

WARLORDS OF XOBITRON
stop the alien invasion
evade deadly H-bombs

KLINGON ENCOUNTERS
control space sattelites
under alien attack

ROCKET-DOCKET
try to dock your shuttle
at the space station

MARTIAN MISSILE MISSION
blast out the missile
launcher sites

ICBM INTERCEPT
destroy ICBMs raining
down upon U.S. cities

SPACE INVDIDERS
wipe out multiple waves
of alien invaders

BULL-FROG
hop frogs across traffic,
ice, river & quick-sand

ROCKY MT. SKI RUN
ski down the icy slopes
of the Rocky Mts.

COCKROACHES & KILLER BEES
step on all the cockroaches
kick the killer bees

AIR-RAID
defend our country
against mid-night air blitz

KAMIKAZE SQUAD
stop kamikaze squad attack
before they score

SPACE COMBATTLE
2 player games
battle over Europe or outer space

MEDIEVAL SIEGE
2 player game
cannon blast castles to rubble

BAMMON-MASTER
play backgammon against
the master computer

PRICES:

Cassette tape version,
First selection.....$
Each additional one...$
Disk version.....add $