SEGA®

Computer

May/June
Issue
1988

The official magazine of the
SEGA User Club of New Zealand

Another Battle won for SEGA Computer

Published by MJH Software in association with Poseidon Software

Published bi-monthly by **MJH Software**
36 Colum Place Bucklands Beach
Auckland New Zealand
Telephone (09) 534-3379
in association with **Poseidon Software**
FREEPOST 243 P.O. Box 277
Tokoroa New Zealand
Telephone (0814) 67-105

- All contributions are welcome, but please include your name, address and telephone number.
- A question and answer page in the form of **Letters To The Editor** is provided and we will do our best to answer any questions about software or programming.
- It is preferable that programs be submitted on tape or disk in a listable form. (No copyright protection please). A listing is useful but don't worry if you aren't lucky enough to own a printer. Where required please include instructions on how to type in the program.
- Please check your programs thoroughly for errors and spelling mistakes before sending it to us. Please send updates if any errors are discovered, so we can publish corrections.
- All software programs received by the magazine becomes the property of **MJH Software** unless by prior arrangement. They are accepted on the basis that they are the original work of the author or required modification to run on a SEGA.
- All contributions are subject to approval by the editor and may be edited to suit the magazine style. Submitted programs will be returned on request
- Each issue two prizes of NZ$40 and NZ$20 for the two feature programs are awarded in the following categories:

  Category 1 - Games. Judged on playability and use of graphics and sound.
  Category 2 - (i) Utilities. Judged on usefulness.
  (ii) Reviews. Judged on overall presentation.

SEGA USER CLUB

MEMBERSHIP YEAR

Oct 1987 - Oct 1988

You automatically become a member of the club when you subscribe to the magazine and this qualifies you for special club benefits.

New Zealand Subscription: NZ$25 incl GST
Australia Subscription: A$26 Airmail

See inside cover at the back

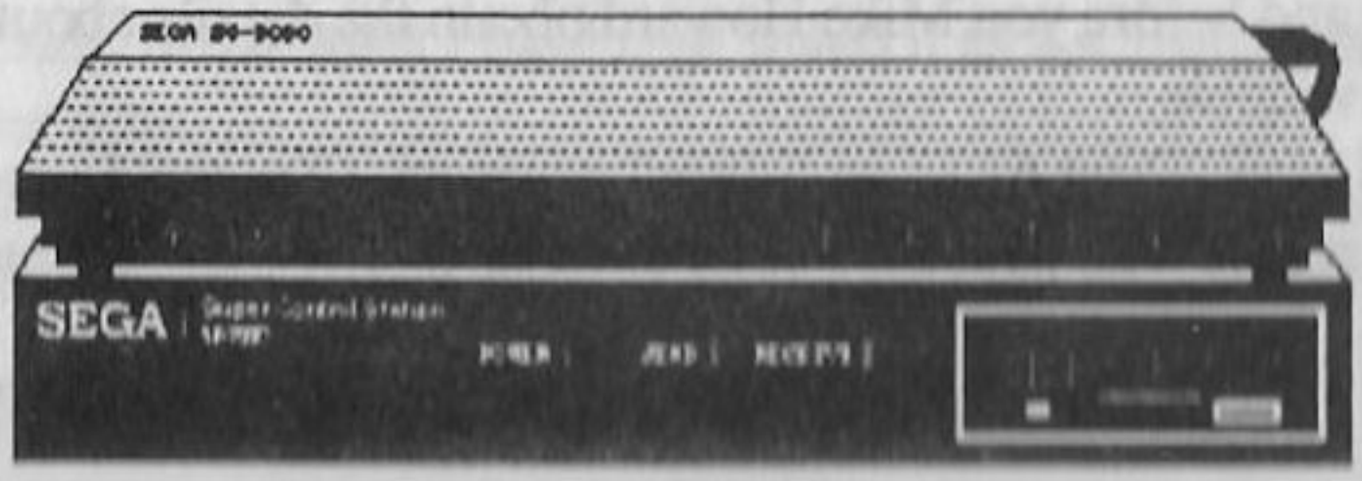
Welcome to the new look

SEGA®
Computer

The official magazine of the SEGA User Club of New Zealand

Issue 4 Contents

Cover illustration : I liked the last one so much, so I made up another.



Letters to the Editor

• This question and answer page is provided to help you. So send me some questions. Remember that you can ask about software or programming.

NOTES from the Editor

At last I've found a "Games Expert" who can answer all those questions concerning adventures such as "How do I get past the first room of ????". He is going to help me with all those questions I couldn't answer about games.

His name is Glen Mackie and he has completed most of the SEGA cassette and cartridge games. Glen will be able to answer most of your questions through the magazine or you can contact him by phone (any time within reason - no two o'clock in the morning calls please!) at (09) 832-5052

Dear Editor,

(a) Is it possible to scroll graphics and colours on the graphics screen? If so, could you possiibly write a program for it? I know that Print 64 can scroll graphics, but only in an upward direction and it leaves the colour behind.

John Dowman, Omaru

Editors reply

(a) See the machine code hints and tips section ...

Dear Editor,

Congratulations for the continued publishing of the magazine, it sure is good to have extra programs and language extensions available.

(a) Where do you and before you Mike Howard obtain the details about the ROM routines and are they are available.

(b) I have entered your "Print 64 as an Editor" program, (Xmas 87) and it crashes when I call &HED00, all seems well with the m/c statements and any comments would be helpful. (32K with LSV).

Continued on next page

(c) One routine that I would like to be able to use is to save individual strings to tape, as is available on most computers via "open" statements etc. I presume with enough knowledge of your LSV program and m/c programming this would be possible, but my efforts do not allow me to find individual strings in memory even with an Editor/Disassembler program.
(d) In one of the last old SEGA Magazines, you commented that you would further explain LSV to allow programmers to add their own routines, is this still possible?

Hope this is not to overwhelming.

Dave Coursey, Christchurch

Editors reply

(a) Mike Howard had access to a book that SEGA in Japan produced which listed all the routines. However I have never seen this (and never will most likely) and my information is obtained by working through the ROM and documenting everything myself.
(b) Remember that it requires Print 64, and make the alterations for Cartridge Basic, and all should work.
(c) If I can find the time, I will work on some more extended commands to save strings.
(d) See the machine code hints and tips section.

Dear Editor,

(a) I am dropping you a note at this time with a moan, I am having a 'griz'. When I put in for taped programmes out of the mags, it said/implied ALL!! Now the programmes should all be supplied along with each mag. I have now received three mags, along with their respective tapes, for which I am grateful for. However the first tape which was for the first two mags did not have two programmes in it!!! one out of each mag. QUOTE "..."
(b) There is also a programme missing from the last tape - "Mystery Program" on page 20. I gather that it is a picture of Garfield. I think that that stinks, we pay for the programmes on tape and then Mike decides to charge extra for some selected ones - not on ...

Geoff, Tokoroa

Editors reply!

(a) QUOTE "With each new magazine you receive in this subscription year, a tape will also be provided which contained all the programmes within that magazine."

To most people this IMPLIED that tapes would be provided from Issue 3 onwards (each new magazine) and in fact many people thought it was $20 per tape! The mere fact that you received a tape for Issue 1 and 2 was a bonus. By the way consider the cost of putting out 5 tapes a year, making up the cassettes and labels - there isn't much left over. The idea behind these tapes was to help pay for the magazine, not me, as I sure don't get anything for doing all the work!
(b) The "Mystery program" was left off the last tape and has already been added to the start of the next tape - my fault. But at least I have now saved it as a screen$ (using LSV), so it will draw up a lot faster now.
PS When I wrote this reply it was two o'clock in the morning and I was a little annoyed - I had to remove all the bad langauge!

Δ

Sorry couldn't fit in all the letters that arrived.

EDITORIAL

A serious computer error, resulted in the magazine being left unprintable, so I had to start again from scratch, otherwise the magazine would have been ready for the last week of May. The error was caused by using a Beta or test version of a program, which wasn't fully tested yet!

Only two more magazines to go before the current subscription ends. Did anyone notice that the last issue was a year late - someone forgot to change the year to 1988 on the front cover, I wonder who?

The feature program of this magazine is Shoot 'em Up by Grant Emms. (Delta Fighter, Tomb of Nozar, Deadly Jewel of Antark). He is helping with the magazine by writing the odd machine code game when he can. Thanks to him and all those who have sent in programs.

I started to write the Invaders game for the machine code section, but it was becoming more advanced than what I had expected and a lot of people learning M/C would have had trouble understanding it, so I have scrapped it for now. Instead of one large program, I have decided to work on small sections at a time - more on this in the next issue.

The next issue will be devoted mainlly to the hardware side of the SEGA and will contain information about keyboard scanning, the Video Display Processor and from the disk drive information on the RS-232, Centronics and Floppy Disk Controllers.

Anyway, back to the Man Logic cartoons again ...

EDITOR: Michael Hadrup

MAN LOGIC

Original Idea by Neil Bradley

News and Reviews

SEGA MASTER SYSTEM

Negoiations are underway (again) for the importation of this unit which is the #1 seller in the UK and US. It's looking good! An approximate retail price of $399 has been mentioned. But when it is available the unit will be sold thru Poseidon Software at a substantial discount and we will have another hire club for this unit.

Servicng of SEGA Computers

Brandt Corporation are continuing to service SEGA Computer's as part of Grandstand's original committments to SEGA Users. Note that they only have one service department which can service SEGA's, the one at their Auckland office.

Brandt Corporation
5 Bassandt Road
Phone (09) 590-369

Remember that Andrew at Poseidon Software is also servicing SEGA's at $20 per hour (compared to $50 per hour at Brandt). Send your computers to him via Poseidon Software.

"The Button" - Spike Suppressor

Andrew has also informed me that quite a few of the SEGA's he receives have blown chips, which are often due to mains borne spikes from fridges, stoves, even from the SEGA itself or the SP-400 when turned on. He has recommended "The Button" a spike suppressor as a possible solution.

"The Button" is a fully enclosed circuit, which is plugged into a 240V power socket at the computer and in Andrew's case he plugs it into the socket of the multi -board.

"The Button" can be obtained from various electrical outlets or from

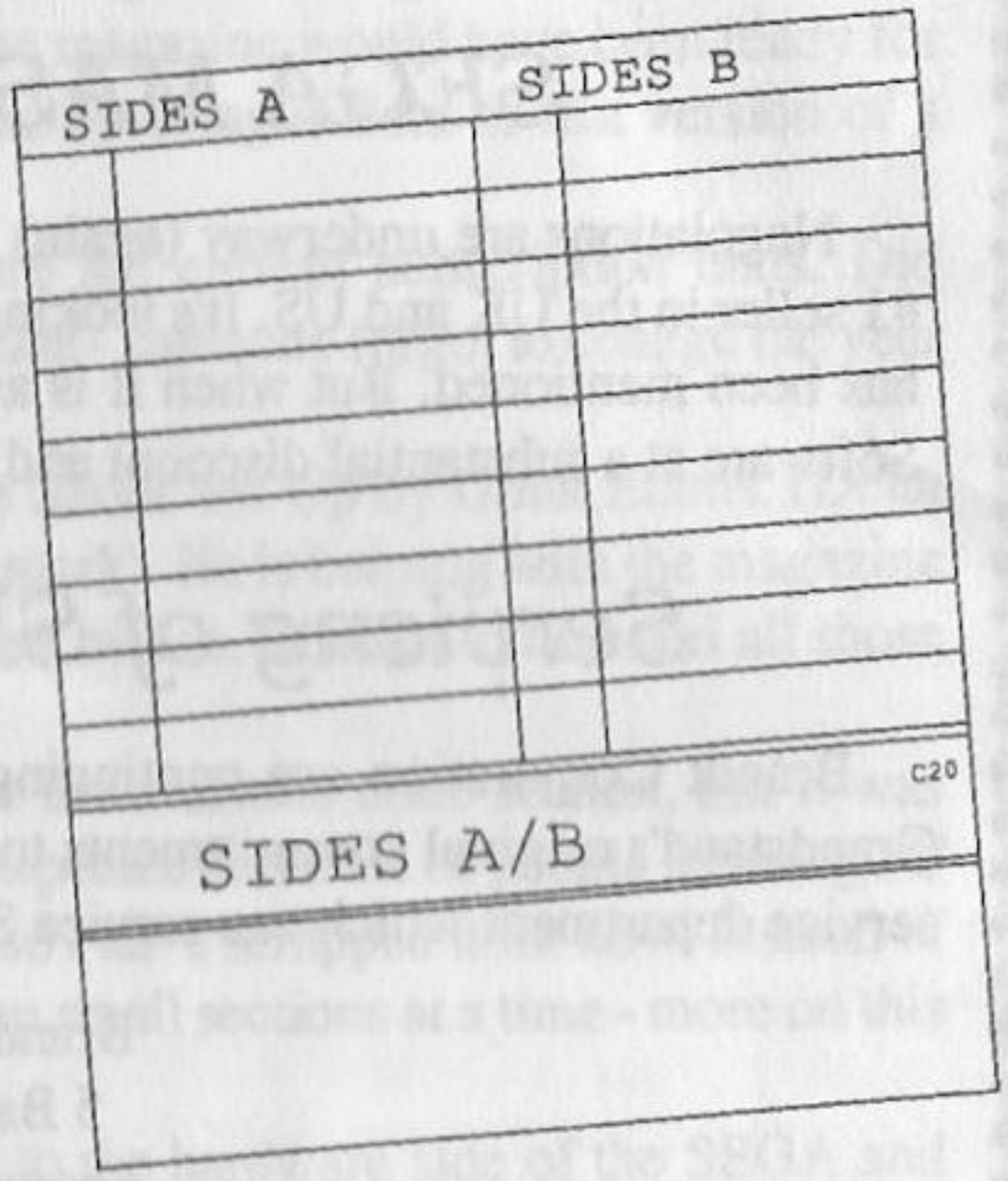
Jarvis Electronix Ltd
78 Riri Street
PO Box 683 Rotorua
Ph (073) 87-992
$28

Continued on page 19

Tape Covers from the SP-400

By Denver Scott

```
10 REM
20 REM        CASSETTE COVERS
30 REM        ON THE SP-400
40 REM
50 REM
60 REM        By DENVER SCOTT
70 REM
80 REM      (C) 1988 MJH Software
90 REM
100 REM
110 REM
120 REM Do not include this line
130 REM for Cartridge Basic
140 REM Selects the SP-400
150 REM on Disk Basic
160 REM
170 POKE&HAF6E,0
180 REM
190 REM
200 SCREEN1,1:COLOR15,1:CLS
210 PRINT"THIS PROGRAM PRINTS"
220 PRINT"CASSETTE COVERS ON THE"
230 PRINT"SP-400 PRINTER PLOTTER"
240 LPRINT"M0,-498"
250 LPRINT"I"
260 GOSUB690
270 RESTORE780:FORI=1TO13
280 READ A,B,C
290 LPRINT"D0,";A;",";B;",";A;",";C;",0,0,0"
300 NEXTI
310 LPRINT "S2"
320 CLS
330 PRINT "CHOOSE NAMES OF SIDE 1/2"
340 INPUT "SIDE 1   [ 8 CHARS] ";X$
350 INPUT "SIDE 2   [ 8 CHARS] ";Z$
360 GOSUB690
370 LPRINT "M19,460"
380 LPRINT "P";X$
390 LPRINT "H"
400 LPRINT "M259,460"
410 LPRINT "P";Z$
420 LPRINT "H"
430 LPRINT "M240,460"
```



$20 Prize
Program of the month

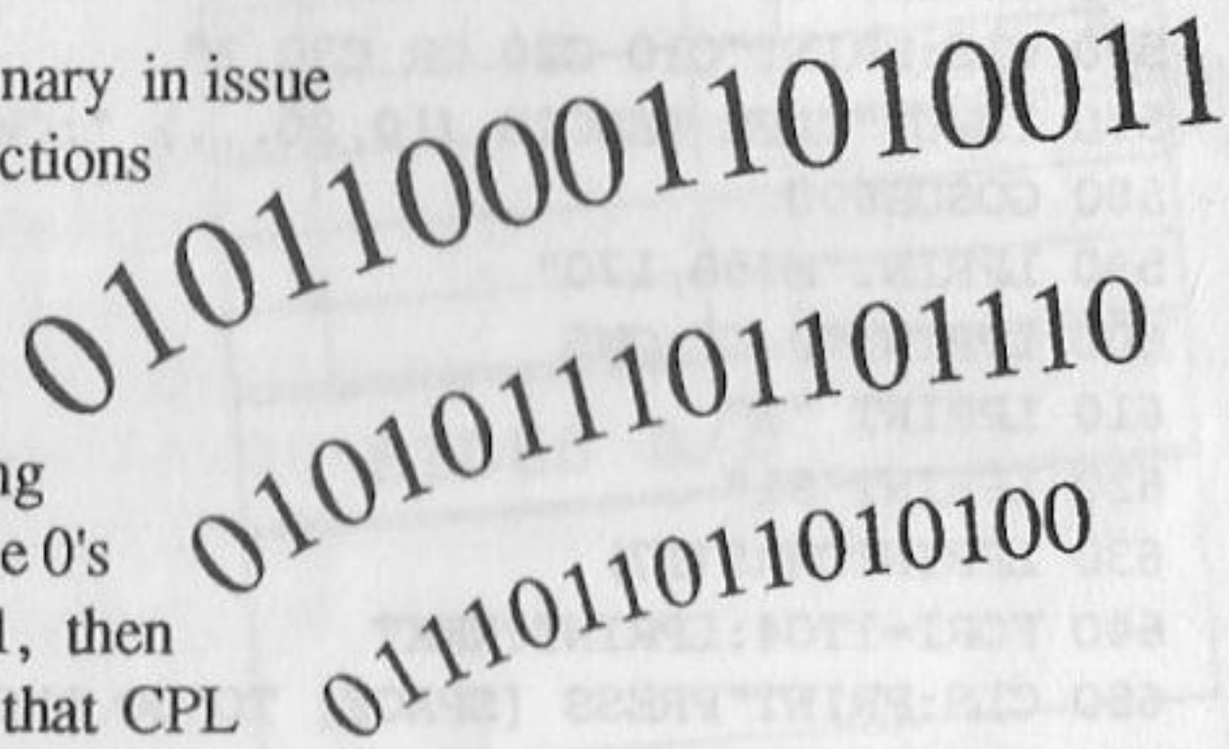
```
440 LPRINT "D240,460,240,190"
450 LPRINT "M290,460"
460 LPRINT "D290,460,290,190"
470 LPRINT "H"
480 LPRINT "M50,460"
490 LPRINT "D50,460,50,190"
500 LPRINT "H"
510 LPRINT "S3"
520 LPRINT "M50,136"
530 LPRINT"P";X$;"/";Z$
540 LPRINT "H"
550 LPRINT"S0"
560 CLS:PRINT"C10-C20 OR C30 ?"
570 INPUT"TAPE LENGTH (10,20...) ";QW$
580 GOSUB690
590 LPRINT"M450,170"
600 LPRINT"P C";QW$
610 LPRINT "H"
620 LPRINT"S1"
630 LPRINTCHR$(17)
640 FORI=1TO4:LPRINT:NEXT
650 CLS:PRINT"PRESS [SPACE] TO DO ANOTHER COVER"
660 IFINKEY$=""THEN660
670 IFINKEY$=" "THEN200
680 END
690 PRINT,,,,"0 - 1st pen colour"
700 PRINT"1 - 2nd pen colour"
710 PRINT"2 - 3rd pen colour"
720 PRINT"3 - 4th pen colour"
730 CURSOR0,12
740 INPUT"Select a colour ";N
750 IFN<0ORN>3THEN730
760 LPRINTCHR$(18);"C";N
770 RETURN
780 DATA 498,480,480
790 DATA 460,480,480
800 DATA 430,480,480
810 DATA 400,480,480
820 DATA 370,480,480
830 DATA 340,480,480
840 DATA 310,480,480
850 DATA 280,480,480
860 DATA 250,480,480
870 DATA 220,480,480
880 DATA 190,485,480
890 DATA 187,485,480
900 DATA 127,485,480
910 DATA 124,480,480
```

Δ

Machine Code Programming

By Michael Hadrup

BITs and BYTES



I've already discussed Counting in Binary in issue 1, so now lets concentrate on instructions which manipulate bits and bytes.

The most simplest of these is CPL which is short for ComPLement. What it does is to invert every bit of A. Inverting means changing all the 1's to 0's and all the 0's to 1's. Suppose A contained 00111001, then CPL will change it to 11000110. Note that CPL does not alter any of the flags.

Next we will look at AND, OR and XOR. These instructions always alter A, by comparing it bit by bit with some other value or register -such as

AND B, OR (HL), XOR 5.

AND	**OR**	**XOR**
0 and 0 gives 0	0 and 0 gives 0	0 and 0 gives 0
0 and 1 gives 0	0 and 1 gives 1	0 and 1 gives 1
1 and 0 gives 0	1 and 0 gives 1	1 and 0 gives 1
1 and 1 gives 1	1 and 1 gives 1	1 and 1 gives 0

If A contained 11000101 and B contained 01010110 then

AND B would give 01000100 as the new value of A.

OR B would give 11010111 as the new value of A.

XOR B would give 10010011 as the new value of A.

All you really have to remember are the following simple rules :

AND - if both are 1 then the result is 1 otherwise it is 0

OR - if both are 0 then the result is 0 otherwise it is 1

XOR - if both are the same (0 or 1) then the result is 0 otherwise it is 1

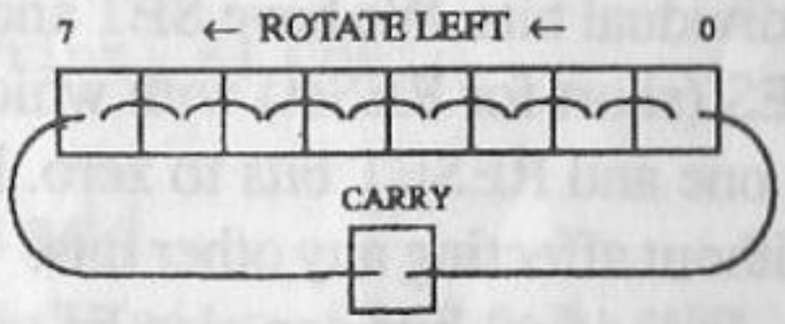
AND, OR and XOR alter the zero flag and always reset the carry flag. As a result of this AND A is often used to reset the carry flag or to test if A is zero.

Consider also XOR A. Verify that it is equivalent to LD A,0. You should use XOR A where ever possible as it is shorter (1 byte) and quicker than LD A,0.

ROTATING and SHIFTING

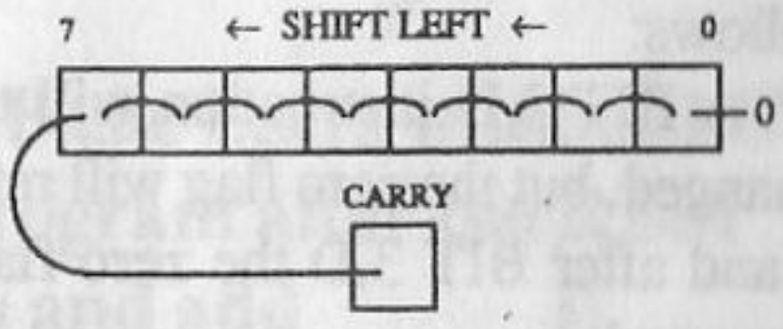
Another group of bitwise instructions are the shift and rotate instructions. They work on any register such as B ,C, D, E, H, L, (HL) and A. Two types of rotation are provided, eight or nine bit rotations where the ninth bit involves the carry flag.

Consider RL (Rotate Left) - such as RL B or RL (HL).



RL moves every bit left. What was once the leftmost bit moves into the carry, and what was once the carry moves into the rightmost bit. This is called a nine bit rotation .

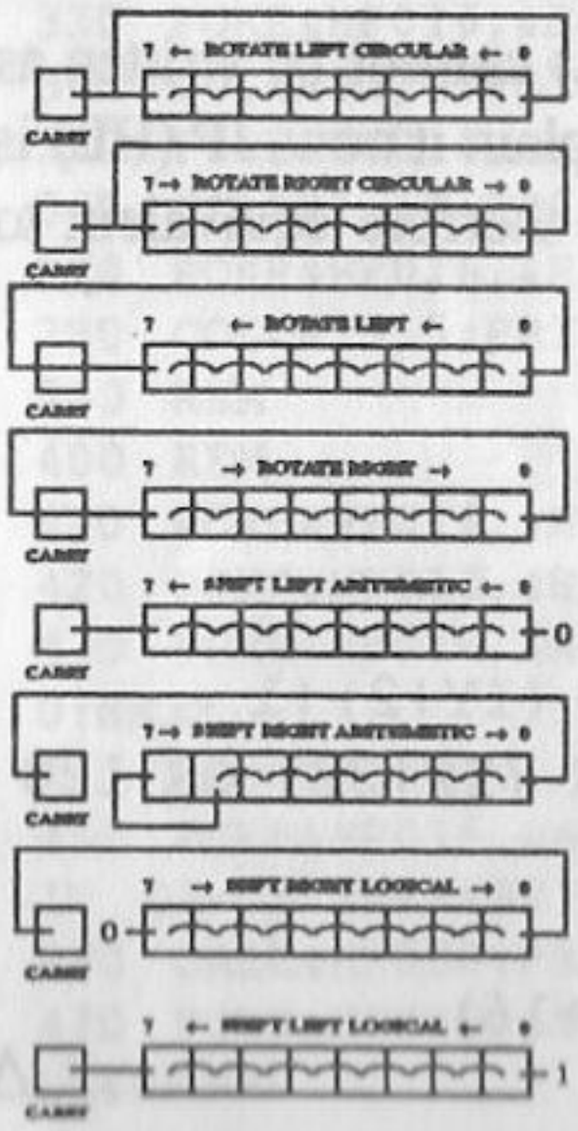
RR (Rotate Right) is similar, but it is to the right. RR moves every bit right. What was once the rightmost bit moves into the carry, and what was once the carry moves into the leftmost bit. This is also a nine bit rotation .



Next we have RLC and RRC (Rotate Left Circular and Rotate Right Circular). The C can also be considered as Rotate without Carry as when the bits are moved left or right, the bit which disappears off one end re-emerges one the other end. Carry gets missed out, although it aqquires a new value of the bit that swapped ends. This is called an eight bit rotation .

RL, RR, RLC and RRC change all of the flags as you'd expect them to. It is here that the strange PV flag is used- see Issue 3. Whenever a bitwise instrcution is used, the number of 1's is counted and the PV flag set is accordingly.

If we are using RL, RR, RLC or RRC on A and we only wish to change the carry flag, then we have a shorthand way of doing it. RLA, RRA, RLCA, and RRCA (note that the space is missing) will perform the same rotation but without changing any of flags except carry.



Next we have the shift instructions, which are very similar to the rotate instructions, except they are used for multiplying and dividing by two. The first is SLA (Shift Left Arithmetic) which multiplies by two. SLA A is in fact the same thing as ADD A,A.

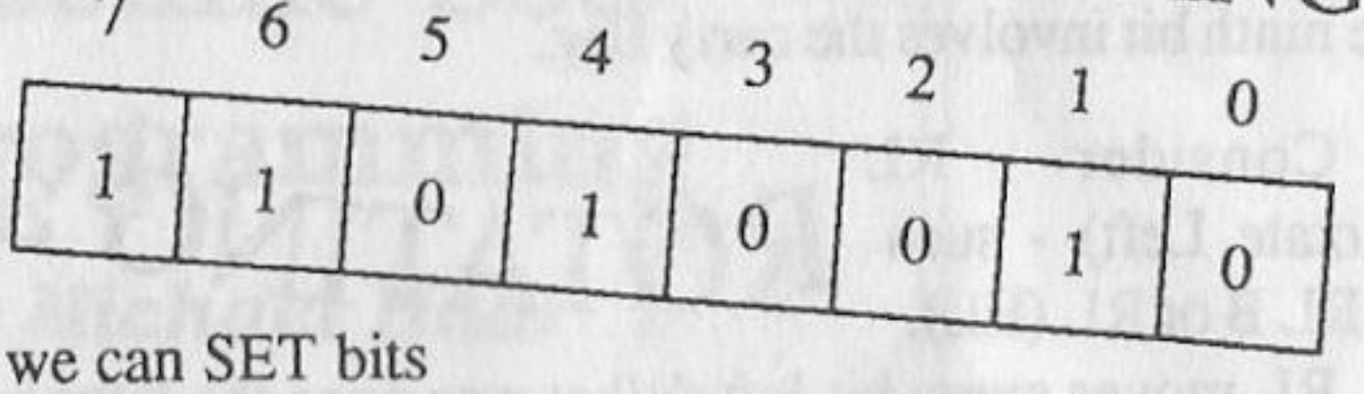
Next we have SRL (Shift Right Logical) which divides by two. It assumes that all 8 bit numbers are positive. In other words if B contained FC, it is not regared as -4 and SRL B would give 7E or 126.

However SRA (Shift Right Arithmetic) allows us to have negative numbers. If B contained FC or -4 then SRA B would give FE or -2.

SLL (Shift Left Logical) is a non-documented instruction and I have no idea what it is really used for!

SETTING and RESETTING

Remember when writting in binary, one byte is eight bits long. The leftmost being bit 7 and the rightmost bit 0.



Often in machine code we play around with a registers individual bits. We have SET and RES (short for RESet) with which we can SET bits to one and RESET bits to zero. For example SET 3,B would change bit 3 of B to one without affecting any other bits.

Individual bits can also be tested . There is an instruction called BIT which works as follows:

A BIT 3,D instruction will test bit three of register D. The actual value of D is not changed, but the zero flag will reflect the result of the test. BIT three of D can be either 0 or 1 and after BIT 3,D the zero flag will reflect this, so JR Z and CALL NZ will work as expected.

IX and IY - Index Registers

Index registers are used to access an item from within a table. The base address or start of the table (16 bit address) is placed in one of the index registers, such as LD IX,#F000. To access the first item add an offset of 0 using (IX+0) which can be simply written as (IX). To access the second item you would use (IX+1) and so on. The offset is also called a displacment and it is an 8 bit number in two complement form (see issue 2). ie. it can be between -128 and 127, so you can actually access the -1 item with (IX-1).

Any instruction which involves HL (except EX DE,HL, ADC HL,RR and SBC HL,RR) can be re-written with IX or IY and any instruction which invloves (HL) (see the exception below) can be re-written as (IX+d) or (IY+d) - where "d" is the displacement. The hex code of these instructions is DD for IX, or FD for IY followed by the hex code as if it was using HL. Note however that the displacement is always the third byte.

There is an exception to the (HL) rule. The instruction JP (HL) **cannot** be written as JP (IX+d). Since I haven't explained JP (HL) yet, I may as well explain it now. JP (HL) is equivalent to GOTO HL in Basic. So LD HL,#F000:JP (HL) is therfore equivalent to JP #F000.

```
DD2100F0    LD IX,#F000        IX=&HF000
DD360541    LD (IX+05),#41     POKE IX+5,&H41
FD5EFE      LD E,(IY-02)       E=PEEK (IY-2)
FD4302      INC (IY+02)        POKE IY+2,PEEK (IY+2)+1
DDCB20FE    SET 7,(IX+20)      POKE IX+32,PEEK (IX+32) OR 128
DD5E10      LD C,(IX+10)
0600        LD B,0
FD19        ADD IY,DE          IY=IY+PEEK (IX+16)
```

Δ

QUIZ *By Denver Scott*

Requires Print 64 from Issue 1

```
20  REM
30  RESTORE 1030
40  REM
50  FORN=&HED00TO&HED44
60  REM
70  READA$:POKEN,VAL("&H"+A$):NEXT
80  S=0
90  REM
100  REM   Set up windows
110  REM
120  DATA 2,2,55,7
130  DATA 2,10,55,15
140  DATA 2,18,55,21
150  REM
160  REM
170  RESTORE120
180  FORN=0TO2:READX,Y,X1,Y1
190  POKE&HF640+N*6,X
200  POKE&HF641+N*6,Y
210  POKE&HF642+N*6,X1
220  POKE&HF643+N*6,Y1
230  NEXT
240  POKE&HF017,&HF6
250  REM
260  REM Cls of each window
270  REM
280  SCREEN2,2:COLOR15,9,,1:CLS
290  CURSOR 30,5:PRINTCHR$(17);"QUESTION"
300  CURSOR 30,68:PRINT"RESPONSE"
310  CURSOR 30,133:PRINTCHR$(17);"YOURANSWER":PRINTCHR$(16)
320  CALL&HF000
330  POKE&HF016,&H40
340  COLOR15,1:PRINTCHR$(12);CHR$(1);
350  POKE&HF016,&H46
360  COLOR1,15:PRINTCHR$(12);CHR$(1);
370  POKE&HF016,&H4C
380  COLOR15,5:PRINTCHR$(12);CHR$(1);
390  REM
400  REM
410  POKE&HF016,&H40:CALL&HF000:PRINT"Thiss the question window"
420  POKE&HF016,&H46:PRINT"Thisshows the CPU's response to your answer"
430  POKE&HF016,&H4C:PRINT"Thisis where you type your answer":FORN=1TO100
0:NEXT
440  RESTORE2000
450  POKE&HF016,&H40:COLOR15,1:PRINTCHR$(12);CHR$(1);:READQ$,X$:
IF  Q$="*"THEN490
460  CALL&HF000:PRINT" ";Q$;"?
470  POKE&HF016,&H4C:COLOR15,5:PRINTCHR$(12):POKE&HF650,3:POKE&HF651,18:CA
LL&HED00
```

Disk Users add

```
10  LOADM"Print  64.Cde"
```

LSV Users add

```
10 *LOADC "Print 64.Cde",&HF000
```

Cartridge Basic Users without LSV add this program after the REM statements and add

```
10 CALL &H9F34
```

```
480  GOTO5000
490  FORM=0TO15:COLORM,1:PRINTCHR$(12)
500  PRINT:PRINTS;"  Correct";CHR$(5):NEXTM:FORN=110TO600STEP25:SOUND1,N,1
0:NEXT:SOUND0
510  IFINKEY$=""THEN510
520  END
1000  REM
1010  REM Machine code data
1020  REM
1030  DATA  DD,2A,16,F0,F3,CD,82,F2,FB,7E,5F,CD,24,ED,CD,E,F0,FE,D,C8,18,EE
1040  DATA  DD,2A,16,F0,D5,E5,F3,CD,43,F2,FB,E1,D1,C9
1050  DATA  36,7F,CD,1A,ED,6,A,CD,6A,56,A7,20,2,10,F8,F5,73,CD,1A,ED,F1,C0,
6,A,CD,6A,56,A7,C0,10,F9,18,DF
2000  DATA  What Nz city is closest to the home of Murray Ball,"?AK:GJF=
2010  DATA  Who played Tv's superman (Surname only),"J==N=K
2020  DATA  What is the main ingredient in glass,"K9F<
2030  DATA  What is a prestidigitator,"E9?A;A9F
2040  DATA  How many more men per team in Rugby than league,"*
2050  DATA  What is the longest river in the South island,";DML@9
2060  DATA  Which Nz city occupies the largest area,"E9F9C9M
2070  DATA  How many wives did Henry VIII have,".
2080  DATA  What is the Maori name for Egmont,"L9J9F9CA
2090  DATA  What number does the Roman numeral C  represent,")((
2100  DATA  What is the capital of Switzerland,":=JF
2110  DATA  What metal is Malaysia the worlds biggest producer of,"LAF
2120  DATA  What was Linus Yale's occupation,"DG;CKEAL@
2130  DATA  What mammal lives the longest,"E9F
2140  DATA  What number is at 6 o'clock in darts,"+
2150  DATA  How many minutes is a regulation soccer game,"1(
2160  DATA  *,*
5000  A$="":N=&HF800+PEEK(&HF640)
5010  IFPEEK(N)=32THENN=N+1:GOTO5010
5020  A=PEEK(N):IF A=0ORA=32THEN5050
5030  IF(A>96)AND(A<123)THENA=A-32
5040  A$=A$+CHR$(A):N=N+1:GOTO5020
5050  POKE&HF016,&H46:COLOR1,15:PRINTCHR$(12):POKE&HF64A,3:POKE&HF64B,10:P
RINTA$
5060  B$="":FORN=1TOLEN(X$):B$=B$+CHR$(ASC(MID$(X$,N,1))+8):NEXT
5070  IFA$=B$THEN5090
5080  PRINT"Wrong -  Answer ";B$;CHR$(5):FORN=600TO110STEP-25:SOUND1,N,10:N
EXT:SOUND0:GOTO450
5090  S=S+1:PRINT S;"   Correct";CHR$(5):FORN=110TO600STEP25:SOUND1,N,10:NE
XT:SOUND0:GOTO450
```

Δ

Machine Code Hints and Tips

Adding your own extended commands

The principle behind adding your own extended commands is in fact quite simple. If a Syntax error is found in a Basic program then the following routine would be used to give the error ...

Disk

```
LD A,31:JP #75D3

75D3    LD HL,(#9ED5)
75D6    JP (HL)
```

Cartridge

```
LD A,31;JP #2571

2571    LD HL,(#86E8)
2574    JP (HL)
```

and (#9ED5) or (#86E8) normally points to an error routine at `#748` or `#6D81` which returns to the text screen, prints the error defined by the number held in A and returns to the editor.

Most of you will know that if you typed "`10 *P2`" in a Basic program, a Syntax error would be generated. Therefore to add extra commands to Basic it is simply a matter of changing the ERROR Vector so that it points to your routine that looks for a Syntax error. Once you have identified that a Syntax Error occured, by checking what caused it, you can determine if one of your extended commands was used. Hence the "*", which causes a Syntax error, makes it easier to check. DE always points to a position in the Basic program which is currently being evaluated. So you look at (DE) for a "*".

Once you have found a Syntax Error and a "*", you check for the command "P" (using example from above), by checking for CHR$(80) (which is a "P"). If you have added lot of extended commands, then you must check for each one and if none are found give a Syntax Error.

Once a command has been identified, you can then check for parameters (the bits after it - if it has any). To do this you can check for characters or you can use some special routines which I will list later. If a parameter had only two possible values such as "1" and "2", then you could check for characters "1" and "2" as CHR$(49) and CHR$(50).

But if it could be any number between 0 and 255,then it would be easier to use a `CALL #4AFE` (`#4E39` for Cartridge Basic) to evaluate a number between -32768 and 32767 (a memory address in two's complement) and then use

`LD A,H:AND A:JP NZ,ERROR` to check that it is between 0 and 255.

(Note that if HL < 255 or #FF then H must be zero.)

If you have multiple parameters separated by comma's, then you must check for a comma CHR$(44) after each parameter.

Once you have checked for all the parameters, you must check for an end of statment character such as CHR$(13) or CHR$(58) which is a ":". Having done this you then return to the Basic interpreter with this character with a `JP #716` (`#6D4F` for Cartridge Basic).

Now to the example ...

Main Listing

```
40  X=&HF000
60  IFA$="*"THEN270
310  THIS LINE HAS AN ERROR!
1000  DATA FE1F203E
1010  DATA 1A13FEC13E1F2036
1020  DATA 1A13FE52C240F0
1030  DATA 1AFE0D2806
1040  DATA FE3A3E352023
1050  DATA D5215BF0CD174B
1060  DATA 2ACEAB
1070  DATA 235E2356EBCD9E7B
1080  DATA 3E0DCD4651
1090  DATA 3E0ACD4651
1100  DATA D11A13C31607
1110  DATA 3E35
1130  DATA F5
1140  DATA 2ABDAE
1150  DATA 7CB5280A
1160  DATA 3E0ACD4651
1170  DATA 545DCD6A47
1180  DATA F1C34807
1190  DATA .Line number not found in line
1200  DATA 2000
1210  DATA *
```

All LSV Users (incl Disk)

```
270  POKE&HFB7A,&H45:POKE&HFB7B,&H45
280  POKE&HFC05,240:POKE&HFC06,240
290  CALL&HFB67
300  POKE&HFC3D,14:POKE&HFC3E,240
1120  DATA C3B0FB
```

Non LSV Users

```
300  POKE&H9ED5,0:POKE&H9ED6,240
1120  DATA 000000
```

Add these listings to the updated MC Editor from issue 3 replacing lines where necessary.

The example program adds a new command *R. Often when Renum'ing a Basic program, it is annoying to get an Undefined Line Number Error. It doesn't tell you which line has the error, only that somewhere there is a GOTO to the wrong place.
Typing *R will tell you at which line contains the offending GOTO statement. The second part of this routine makes Basic list the offending line when there is an error, such as

```
260 THIS LINE HAS AN ERROR!
? Syntax error in 60
Ready
```

All Cartridge Users replace the following lines

```
1050  DATA D5215BF0CD524E
1060  DATA 2AEE93
1070  DATA 235E2356EBCD3A2B
1080  DATA 3E0DCDE454
1090  DATA 3E0ACDE454
1100  DATA D11A13C34F6D
1140  DATA 2A0597
1160  DATA 3E0ACDE454
1170  DATA 545DCD1F4B
1180  DATA F1C3816D
```

The M/C Routine - Disk Basic Version with LSV

```
F000 FE1F       CP 1F          Was the error "Syntax error"?
F002 203E       JR NZ,F042     If not then report the error
F004 1A         LD A,(DE)      Get the first character
F005 13         INC DE         Move to the next character
F006 FEC1       CP C1          Is it a "*"? - NOTE this is a token not CHR$(42).
F008 3E1F       LD A,1F        Set to Syntax Error
F00A 2036       JR NZ,F042     If not "*" then report error
F00C 1A         LD A,(DE)      Get second character
F00D 13         INC DE         Move to the next character
LSV enters here
F00E FE52       CP 52          Is it a "R"?
F010 C200F1     JP NZ,F040     If not "R" then report error or check for others
```

```
F013 1A          LD A,(DE)        Get third character
F014 FE0D        CP 0D            Is it a carriage return (end of line)?
F016 2806        JR Z,F01E        If it is CR then jump foward
F018 FE3A        CP 3A            Is it a ":" (end of statement)?
F01A 3E35        LD A,35          Set to Statement Parameter error
F01C 2023        JR NZ,F041       If not ":" then report error
F01E D5          PUSH DE          Store the position in program
F01F 215BF0      LD HL,F05B       Point HL to message
F022 CD174B      CALL 4B17        Print the message
F025 2ACEAB      LD HL,(ABCE)     Get the address of RENUM line
F028 23          INC HL           Skip the length byte
F029 5E          LD E,(HL)        Get the low byte of line number
F02A 23          INC HL           Move to high byte
F02B 56          LD D,(HL)        Get the high byte of line number
F02C EB          EX DE,HL         Switch line number to HL
F02D CD9E7B      CALL 7B9E        Print HL in decimal on current screen
F030 3E0D        LD A,0D          Set to Carriage Return
F032 CD4651      CALL 5146        Print it
F035 3E0A        LD A,0A          Set to Line Feed
F037 CD4651      CALL 5146        Print it
F03A D1          POP DE           Restore position in Basic program
F03B 1A          LD A,(DE)        Get end of line character
F03C 13          INC DE           Move to next character
F03D C31607      JP 0716          Return to Basic interpreter with character
F040 3E35        LD A,35          Set to Statement Parameter error
F042 C3B0FB      JP FBB0          Jump to LSV error routine. (allows for *ONE)
```

LSV returns here if error is to be reported When LSV is not used, this line is not included.

```
F045 F5          PUSH AF          Store the error number
F046 2ABDAE      LD HL,(AEBD)     Get execution line number
F049 7C          LD A,H           Is it zero?
F04A B5          OR L             (If any bit is 1 then result is non zero)
F04B 280A        JR Z,F057        If zero (direct command) then give error
F04D 3E0A        LD A,0A          Set to line feed
F04F CD4651      CALL 5146        Print it
F052 54          LD D,H           Set start and end line number to same value
F053 5D          LD E,L           (ie LIST 10-10)
F054 CD6A47      CALL 476A        Call list routine to list the line
F057 F1          POP AF           Restore the error number
F058 C34807      JP 0748          Report the error
```

Continued on next page

Chaining Programs on Disk

A lot of people have been asking how you can load and run a program from within another program on the disk drive. This is very easy to do. Just add aline such as the following anywhere in your program ...

```
CALL &H5D0:"FILENAME"
```

If your interested - the CALL is to a section of the RUN "Filename" command in Disk Basic. It calls the point just after the parameters are checked for. In fact a lot of routines can be called just after the parameters are checked for.

Extended Commands Continued

Here are some routines to help you with adding your own extended commands.
* Assume that DE is used as a pointer in a Basic program.

Disk Basic	Description	Cartridge Basic
#716	Return point after completion of a statement. A must holde end of line marker, so the routine can decide whether more commands are to be executed.	#6D4F
#843E	*Skip any spaces and put a char into A	#2310
#843D	*Skip one char and any spaces and put a char into A	#230F
#83F2	*Evaluates a numeric expression and as a decimal line number and places it in HL. (between 0 and 65535)	#22C4
#4AFE	*Evaluate a numeric expression as a number between -32768 and 32767 and places it in HL in two's complement form (Note this is the form of a memory address).	#4E39
#6838	*Evaluate a string expression and places result in string stack.	#2C76
#74EB	Copy string from stack to a string register. A holds 1 for string register 1 else string resgister 2	#2489
#83E1	Checks a range of values of A against min/max values in HL. The carry flag is reset if H<= A <=L.	#22B3
#8394	Converts character in A to uppercase if it was lower-case.	#2266
#83D8	Checks if A is in the range A...Z ie an uppercase letter Carry flag is reset if in range.	#22AA
#8436	Checks if A is in the range 0...9 ie a digit Carry flag is reset if in range.	#23Ø8
#4AEF	Check for CR or ":", if not found give syntax error, otherwise return.	#4E2A
#9A8F	Length of string in string resgister 1	#82A2
#9A90	Start of string register 1	#82A3
#9B8F	Length of string in string resgister 2	#83A2
#9B90	Start of string register 2	#83A3

Δ

Scroll Routines

Here are some scroll routines for you to look at. They scroll the graphics screen (pixels and colour) in each of the four directions 8 pixels at a time. Note that the routines are relocateable, but BUFFER (256 bytes maximum) must be replaced with an appropiate address. The values in brackets show the limits to poke in.

For example to scroll the whole screen, you could store the machine code at #F000 and let BUFFER = #F100 and make ...

TOP = 0
LEFT=0
WIDTH=32 sqaures (so poke in 8*32 = 256 ie poke in a 0)
BOTTOM=23

To scroll within a window such as (64,64)-(127,127) use
TOP=8
LEFT=8 (so poke in 64)
WIDTH=8 squares (so poke in 64) .
BOTTOM= 15

```
Hex codes    #=Hex otherwise decimal
F3    Up     DI
0EBE         LD C,#BE
1E??         LD E,TOP  (0..22)
CD09F0       CALL Up1
1E??         LD E,TOP+32  (32.54)
3E??  Up1    LD A,LEFT*8  (0,8..248)
D3BF         OUT (#BF),A
7B           LD A,E
3C           INC A
D3BF         OUT (#BF),A
21????       LD HL,BUFFER
06??         LD B,WIDTH*8  (0,8..248)
EDA2  Up2    INI
20FC         JR NZ,Up2
3E??         LD A,LEFT*8  (0,8..248)
D3BF         OUT (#BF),A
7B           LD A,E
F640         OR 64
D3BF         OUT (#BF),A
21????       LD HL,BUFFER
06??         LD B,WIDTH*8  (0,8..248)
EDA3  Up3    OUTI
20FC         JR NZ,Up3
1C           INC E
7B           LD A,E
E61F         AND 31
FE??         CP BOTTOM  (1..23)
20D5         JR NZ,Up1
C9           RET
```

```
Hex codes    #=Hex otherwise decimal
F3    Down   DI
0EBE         LD C,#BE
1E??         LD E,BOTTOM-1  (0..22)
CD3EF0       CALL Down1
1E??         LD E,BOTTOM+31  (32..54)
3E??  Down1  LD A,LEFT*8  (0,8..248)
D3BF         OUT (#BF),A
7B           LD A,E
D3BF         OUT (#BF),A
21????       LD HL,BUFFER
06??         LD B,WIDTH*8  (0,8..248)
EDA2  Down2  INI
20FC         JR NZ,Down2
3E??         LD A,LEFT*8  (0,8..248)
D3BF         OUT (#BF),A
7B           LD A,E
C641         ADD A,65
D3BF         OUT (#BF),A
21????       LD HL,BUFFER
06??         LD B,WIDTH*8  (0,8..248)
EDA3  Down3  OUTI
20FC         JR NZ,Down3
7B           LD A,E
1D           DEC E
E61F         AND 31
FE??         CP TOP
20D6         JR NZ,Down1
```

```
C9            RET
F3    Left    DI
0EBE          LD C,#BE
1E??          LD E,TOP (0.22)
CD70F0        CALL Left1
1E??          LD E,TOP+32 (32.54)
3E??  Left1   LD A,LEFT*8 (0,8..248)
D3BF          OUT (#BF),A
7B            LD A,E
D3BF          OUT (#BF),A
21????        LD HL,BUFFER
06??          LD B,WIDTH*8 (0,8..248)
EDA2  Left2   INI
20FC          JR NZ,Left2
D9            EXX
21????        LD HL,BUFFER+8
11????        LD DE,BUFFER
01????        LD BC,WIDTH*8-8
              (8,16..248)
EDB0          LDIR
D9            EXX
3E??          LD A,LEFT*8 (0,8..248)
D3BF          OUT (#BF),A
7B            LD A,E
F640          OR 64
D3BF          OUT (#BF),A
21????        LD HL,BUFFER
06??          LD B,WIDTH*8 (0,8..248)
EDA3  Left3   OUTI
20FC          JR NZ,Left3
7B            LD A,E
1C            INC E
E61F          AND 31
FE??          CP BOTTOM
20C9          JR NZ,Left1
```

```
C9            RET
F3    Right   DI
0EBE          LD C,#BE
1E??          LD E,TOP (0.22)
CDB1F0        CALL Right1
1E??          LD E,TOP+32 (32.54)
3E??  Right1  LD A,LEFT*8 (0,8..248)
D3BF          OUT (#BF),A
7B            LD A,E
D3BF          OUT (#BF),A
21????        LD HL,BUFFER
0600          LD B,WIDTH (0,8..248)
EDA2  Right2  INI
20FC          JR NZ,Right2
D9            EXX
21????        LD HL,BUFFER+WIDTH*8-9
              (7,15..247)
11????        LD DE,BUFFER+WIDTH*8-1
(15,23..255)
01????        LD BC,WIDTH*8-8 (8,16..248)
EDB8          LDDR
D9            EXX
3E??          LD A,LEFT*8 (0,8..248)
D3BF          OUT (#BF),A
7B            LD A,E
F640          OR 64
D3BF          OUT (#BF),A
21????        LD HL,BUFFER
06??          LD B,WIDTH*8 (8,16..248)
EDA3  Right3  OUTI
20FC          JR NZ,Right3
7B            LD A,E
1C            INC E
E61F          AND 31
FE??          CP BOTTOM
20C9          JR NZ,Right1
C9            RET
```

Δ

Telephone Caller

By Brendan Hallett

Type in the number and hold the handset to the TV speaker, then press CR and the number will be dialed. You may have to try the volume a bit before it works every time and remember it must be a push-button phone. Someone could also add a directory ...

```
10 ERASE:DIMT(10),T1(10)
20 RESTORE1090
30 FORN=1TO10:READT(N),T1(N):NEXT
100 CLS
110 CURSOR0,0:INPUT"Number ";A$
1000 FOR I=1TOLEN(A$)
1010 A=ASC(MID$(A$,I,1))-47
1020 IF(A<1)OR(A>10)THEN1050
1030 SOUND1,T(A),15
1040 SOUND2,T1(A),15
1050 FOR Q=1TO70:NEXT Q
1060 SOUND 0
1070 NEXT I
1080 GOTO 110
1090 DATA 941,1209
1100 DATA 697,1209
1110 DATA 697,1336
1120 DATA 697,1447
1130 DATA 770,1209
1140 DATA 770,1336
1150 DATA 770,1477
1160 DATA 852,1209
1170 DATA 852,1336
1180 DATA 852,1477
```

Δ

DOG *from Footrot Flats*

From David Martin of South Australia

```
10 FORN=0TO27
20 READA$:PATTERNS#N,A$
30 NEXT
40 SCREEN2,2:COLOR1,15,,15:CLS
50 POSITION(0,0):MAG1
55 RESTORE1000
60 FORN=0TO6
70 READX,Y
80 SPRITEN,(X,Y),4*N,1
90 NEXT
91 LINE(85,90)-(174,142),1,B
92 GOTO92
100 DATA 0000071806010000
110 DATA 00
120 DATA 0078860103874F7F
130 DATA 7F7F7F7F7F7F7F7F
140 DATA 0000000080808080
150 DATA 8080808080000000
160 DATA 00
170 DATA 00
180 DATA 0000000078FFFFDF
190 DATA 1820409884838060
200 DATA 000000037FFFE4DA
210 DATA 2E3B2C300000C0B8
220 DATA 3078FCFCDC884040
230 DATA 80F00906023B0204
240 DATA 00
250 DATA 0000C00000804000
260 DATA FFFFFEFEFE7F7F3F
270 DATA 3F7F7FFFFFFFFFFF
280 DATA 00000000000C0FFFF
290 DATA FFFFFFFFFFFFFFFF
300 DATA 000000000010FFFF
310 DATA FFFFFFFFFFFFFFFF
320 DATA 1F1F3EFCF8F8F0E0
330 DATA C0C0C1C2C2848408
335 DATA 0F09324040404080
340 DATA 8080800000000000
350 DATA F8408000000000000
360 DATA 00
1000 DATA 100,100,116,100,132,100,
148,100,108,116,124,116,140,116
```

PS David, I am slowly typing in your other programs, but you keep sending new versions of SEGACAD, and I have lost track of which version is which! - hopefully next issue

Δ

Reviews - Zippy Race

If you liked the arcade version then you'll love this because it's an exact copy with all the features of the original. Even the graphics and tunes are the same.

For those of you, who have never seen it - It's a Monaco GP style game, (birds eye view of the road), but with a motorbike in a race across America. The interesting thing about the race is that you race a motorbike and the other 99 racers drive cars!

As you race along roads and deserts trying to improve your ranking, you must also try to find fuel so you are not left stranded.

Another exact copy from the arcades.

Name: Zippy Race
Options: 1 Player
System: Cartridge - Poseidon Software Hire Club

Continued on page 23

Shoot 'Em Up *$40*

By Grant Emms

This is a fast game, where you have to shoot the alien ships coming towards you from the left of the screen. You can move your space-ship up and down with the cursor keys and you can fire missiles with a shift key.

If you're interested - the scrolling is achieved using the *Graphics Mode Name Table* - as described in the Machine Code Hints and Tips Section in issue 3 (Complex Screens). This is similar to the *Text Mode Name Table* except each byte represents a block of 8 x 8 pixels. (compared with 6 x 8). There are are 768 bytes in the *Name Table* and each corresponds to 8 bytes in the *Pattern Generator Table* and 8 bytes in the *Colour Table*. (This makes both tables 768x8 = 6144 bytes).

The screen is scrolled in three sections, at three different speeds (top contains stars, midlle mountains and the bottom a grid pattern) to give the impression of speed and distance. Of course since each byte in the *Name Table* corresponds to an 8 x 8 block the screen scrolls 8 pixels at a time making the scrolling a little jerky, but fast.

The subroutine which draws the background starts at line 180, so try drawing different backgrounds.

Typing in the program - 32K Cartridge Users without LSV

Type in the whole program save it, then type RUN it. If any errors are found fix them, save it and RUN the program again. Save the final version after all the bugs have been removed. To play - LOAD it and type RUN.

Typing in the program - Disk Basic Users

Type in the whole program and change line 20 to `20 LOADM "SHOOT'EM.M/C"`

Save it with `SAVE "SHOOT'EM.BAS"`, then type GOSUB 1000. If any errors are found fix them, save it and type GOSUB 1000 again. After all the bugs have been removed save the machine code with

```
SAVEM "SHOOT'EM.M/C",&HEC00,&HEFFF
```

To play type `RUN "SHOOT'EM.BAS"`

Typing in the program - LSV Users

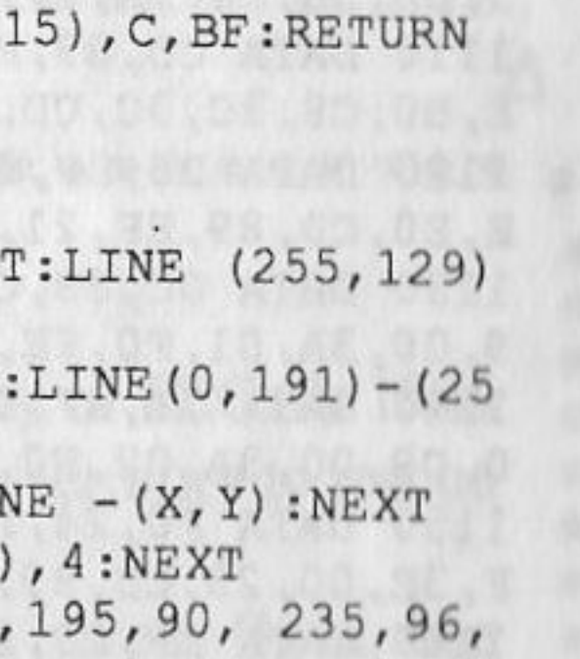
Type in the whole program and change line 20 to `20 *LOADC "SHOOT'EM.M/C"`

Save it with `*SAVE "SHOOT'EM.BAS",RUN10`, then type GOSUB 1000. If any errors are found fix them, save it and type GOSUB 1000 again. After all the bugs have been removed save the machine code after the main program with

```
*SAVEC "SHOOT'EM.M/C",&HEC00,&HEFFF
```

To play type `*LOAD "SHOOT'EM.BAS"` and allow the machine code to be loaded in from cassette.

```
1 REM    A SHOOT 'EM UP GAME
2 REM
3 REM       BY GRANT EMMS
4 REM
5 REM FOR 32K CARTRIDGE BASIC
6 REM       OR DISK BASIC
7 REM
8 REM May 1988
9 REM
10 CLS:CURSOR9,12:PRINT "PLEASE STAND BY"
20 LOADM"SHOOT'EM.M/C":REM GOSUB 1000:REM DISK BASIC users replace this
 line with ( LOADM"SHOOT'EM.M/C" ), LSV users replace with ( *LOADC"SHO
OT'EM.M/C" ).
30 GOSUB 340
40 GOSUB 180; DRAW SCREEN
50 SCREEN 2,2:C=13:GOSUB 160:CURSOR24,8:PRINT "PRESS SPACE TO START PLA
YING"
60 CALL&HEFE1:FOR I=0 TO19:IF INKEY$=" " THEN 80
70 NEXT :GOTO 60
80 GOSUB 170:C=2:GOSUB 160:CURSOR64,8:PRINT "SHOOT     'EM     UP  !!":
GOSUB 300:CALL &HEC00
90 FOR I=&H3B04 TO &H3B0C STEP 4:VPOKEI,192:NEXT
100 FOR I=15 TO 0 STEP -1:SOUND4,0,I:PT=INT((I)/4)*4+16:SPRITE0,,PT,4+I
NT(PT/4):FOR U=0 TO 1:NEXT U,I
110 VPOKE&H3B00,192
120 C=7:GOSUB 160:CURSOR110,8:PRINT "GAME OVER"
130 FOR I=0 TO 31:CALL&HEFE1:FOR U=0 TO 3:NEXTU,I
140 GOTO 50
150 REM  VARIOUS SUBROUTINES
160 FOR I=32TO63:VPOKE&H3800+I,I:NEXT :BLINE(0,8)-(255,15),C,BF:RETURN
170 FOR I=&HF019 TO &HF01D:POKEI,0:NEXT:RETURN
180 REM  DRAW SCREEN
190 SCREEN 2,1:CLS:COLOR,1,(0,0)-(255,191),1
200 FORX=32TO 240 STEP 32:LINE (X,128)-(X-32,191),8:NEXT:LINE (255,129)
-(224,191)
210 Y=128:FOR A=0 TO 10:Y=Y+A:LINE (0,Y)-(255,Y),6:NEXT:LINE(0,191)-(25
5,191)
220 RESTORE 250:PSET (0,120),5:FORI=1 TO 13:READ X,Y:LINE -(X,Y):NEXT
230 RESTORE 240:FORI=1 TO 5:READ A,B,C,D:LINE(A,B)-(C,D),4:NEXT
240 DATA 0,90,45,69, 45,69,97,88, 130,95,170,73, 170,73,195,90, 235,96,
255,90
250 DATA 30,100,35,80,70,115,90,84,93,98,108,68,150,122,155,102,160,120
,168,102,173,117,220,70,255,120
260 FOR I=0 TO 30:PSET(RND(1)*250,RND(1)*48+16),15:NEXT
270 COLOR14:FOR I=0 TO 9:CURSORI*8,0:PRINT CHR$(48+I):VPOKE&H3800+I,31:
NEXT
280 CURSOR96,0:PRINT "SCORE|":CURSOR22*8,0:PRINT "0":CURSOR23*8,0:PRINT
 "0"
290 RETURN
300 MAG1:SPRITE0,(&HE0,30),0,15:SPRITE3,,4,14
310 FOR I=&HF019 TO &HF01D:POKEI,0:NEXT
320 FOR I=&H3811 TO &H3815:VPOKEI,31:NEXT
330 RETURN
340 REM sprite data
350 RESTORE 370:FOR I=0 TO 31:READ A$:PATTERNS#I,A$:NEXT
360 RETURN
370 DATA 00000F33FF00030F,0000000000000000,0103FFFE03FCFEFE,00000000000
00000
```



```
380 DATA 0000000000003F00,0000000000000000,0000000020C020,00000000000
00000
390 DATA 0102073FC07F1F07,0400000000000000,8040E0FC03FEF8E0,20000000000
00000
400 DATA 000000603F3F183F,0000000000000000,00000000FCFE0000,00000000000
00000
410 DATA 1140024000009000,5002080000000000,0820450000020021,00124000000
00000
420 DATA 0144134F3C789E27,1402000000000000,0820C5FA1F0EFAA5,18800000000
00000
430 DATA 002004231FF86C97,2108000000000000,00800470FD0E9FFC,12200000000
00000
440 DATA 000000231FFF7F17,0100000000000000,00000070FCFEFFFC,10000000000
00000
1000 REM* DISK BASIC or CARTRIDGE BASIC WITH LSV: Enter lines 1000 onwa
rds as a seperate program, if you wish to LOAD the machine code as obje
ct data.
1010 REM TURN DATA INTO MACHINE CODE
1020 X=&HEC00:N=8*4
1030 RESTORE 1090:LN=1100
1040 FOR AD=X TO X+N*32-1 STEP 32
1050 T=0
1060 FOR I=0 TO 31:READ A$:P=VAL("&H"+A$):POKEAD+I,P:T=T+P:NEXT:READ A$
:IF T<>VAL("&H"+A$) THEN BEEP2:PRINT "MISTAKE IN LINE ";LN:END
1070 CURSOR0,0:PRINT HEX$(AD):LN=LN+10:NEXT
1080 RETURN
1090 DATA CD,9B,EF,CD,24,EC,CD,5D,EC,CD,95,EC,CD,B6,EC,CD,23,ED,CD,69,E
D,CD,11,EE,CD,6C,EE,CD,D4,EE,CD,2B,1681
1100 DATA EF,C3,03,EC,00,3E,92,D3,DF,00,3E,06,D3,DE,DB,DC,CB,77,20,11,2
1,00,3B,CD,7A,EF,FE,08,28,07,3D,3D,0E83
1110 DATA CD,89,EF,C9,00,3E,04,D3,DE,DB,DC,CB,6F,C0,21,00,3B,CD,7A,EF,F
E,B0,C8,3C,3C,CD,89,EF,C9,00,3E,92,1210
1120 DATA D3,DF,3E,06,D3,DE,DB,DD,CB,5F,C0,3A,01,F0,FE,00,C0,21,0D,3B,3
E,E0,CD,89,EF,21,00,3B,CD,7A,EF,21,10B1
1130 DATA 0C,3B,CD,89,EF,3E,01,32,01,F0,3E,01,32,10,F0,3E,20,32,11,F0,C
9,00,3A,01,F0,FE,00,C8,21,0D,3B,CD,0BE0
1140 DATA 7A,EF,D6,08,CD,89,EF,FE,00,C0,2B,3E,C0,CD,89,EF,3E,00,32,01,F
0,C9,00,3A,02,F0,FE,01,28,32,2A,04,0E95
1150 DATA F0,24,2C,7C,E6,7F,67,22,04,F0,7E,E6,3F,C6,10,21,04,3B,CD,89,E
F,3E,00,23,CD,89,EF,23,3E,08,CD,89,0E21
1160 DATA EF,23,3E,07,CD,89,EF,3E,01,32,02,F0,AF,32,03,F0,00,21,05,3B,C
D,7A,EF,C6,02,CD,89,EF,FE,00,20,03,0D98
1170 DATA 32,02,F0,3A,03,F0,C6,03,32,03,F0,21,04,3B,FA,1A,ED,CD,7A,EF,C
6,02,CD,89,EF,C9,CD,7A,EF,D6,02,CD,1087
1180 DATA 89,EF,C9,00,3A,06,F0,FE,01,28,2A,21,00,3B,CD,7A,EF,C6,10,F2,3
8,ED,D6,20,21,08,3B,CD,89,EF,23,3E,0E41
1190 DATA 00,CD,89,EF,23,3E,0C,CD,89,EF,23,3E,0A,CD,89,EF,3E,01,32,06,F
0,00,21,09,3B,CD,7A,EF,C6,04,CD,89,0DC9
1200 DATA EF,FE,00,20,03,32,06,F0,C9,00,3A,01,F0,FE,00,C8,3A,07,F0,FE,0
0,C2,C3,ED,21,04,3B,CD,7A,EF,57,23,0EA3
1210 DATA CD,7A,EF,5F,21,0C,3B,CD,7A,EF,C6,08,BA,38,34,D6,10,BA,30,2F,2
3,CD,7A,EF,C6,08,BB,38,26,D6,10,BB,0F07
1220 DATA 30,21,AF,32,01,F0,3E,C0,21,0C,3B,CD,89,EF,3E,08,32,07,F0,3E,0
2,32,10,F0,3E,10,32,11,F0,0E,03,CD,0B0E
1230 DATA A7,EF,C9,00,3A,08,F0,FE,00,C0,21,08,3B,CD,7A,EF,57,23,CD,7A,E
F,5F,21,0C,3B,CD,7A,EF,C6,08,BA,D8,0FF6
1240 DATA D6,10,BA,D0,23,CD,7A,EF,C6,08,BB,D8,D6,10,BB,D0,AF,32,01,F0,3
E,C0,21,0C,3B,CD,89,EF,3E,08,32,08,0F98
```

```
1250 DATA F0,3E,02,32,10,F0,3E,10,32,11,F0,0E,02,CD,A7,EF,C9,00,3A,07,F
0,FE,00,28,26,3D,32,07,F0,20,0E,21,0B51
1260 DATA 04,3B,3E,C0,CD,89,EF,AF,32,02,F0,18,12,E6,FE,CB,27,C6,10,21,0
6,3B,CD,89,EF,23,3E,09,CD,89,EF,00,0E81
1270 DATA 3A,08,F0,FE,00,C8,3D,32,08,F0,20,0D,21,08,3B,3E,C0,CD,89,EF,A
F,32,06,F0,C9,E6,FE,CB,27,C6,10,21,0EA0
1280 DATA 0A,3B,CD,89,EF,23,3E,09,CD,89,EF,C9,00,3A,10,F0,FE,00,C8,FE,0
2,CA,AF,EE,3A,11,F0,3D,32,11,F0,20,0F34
1290 DATA 08,32,10,F0,3E,FF,D3,7F,C9,0E,7F,FE,1F,20,0A,16,E7,ED,51,00,0
0,16,DF,ED,51,CB,3F,D0,ED,44,3D,E6,0F02
1300 DATA 0F,16,C0,ED,51,00,00,00,ED,79,C6,F0,ED,79,C9,3A,11,F0,3D,32,1
1,F0,20,08,32,10,F0,3E,FF,D3,7F,C9,0ECB
1310 DATA 0E,7F,FE,0F,20,04,16,E4,ED,51,CB,3F,C6,06,ED,44,3D,ED,79,C9,0
0,21,00,3B,CD,7A,EF,57,23,CD,7A,EF,0EAB
1320 DATA 5F,3A,07,F0,FE,00,20,20,21,04,3B,CD,7A,EF,C6,08,BA,38,15,D6,1
0,BA,30,10,23,CD,7A,EF,C6,08,BB,38,0D33
1330 DATA 07,D6,10,BB,30,02,E1,C9,00,3A,08,F0,FE,00,C0,21,08,3B,CD,7A,E
F,C6,08,BA,D8,D6,10,BA,D0,23,CD,7A,0F48
1340 DATA EF,C6,08,BB,D8,D6,10,BB,D0,E1,C9,00,3A,00,F0,E6,0F,20,03,C3,5
1,EF,E6,03,20,03,C3,47,EF,21,FF,3A,100F
1350 DATA 06,08,CD,5E,EF,18,12,21,FF,39,06,08,CD,5E,EF,18,08,21,FF,38,0
6,06,CD,5E,EF,21,00,F0,34,C9,00,C5,0C44
1360 DATA CD,7A,EF,4F,06,1F,2B,CD,7A,EF,23,CD,89,EF,2B,10,F5,79,CD,89,E
F,2B,C1,10,E6,C9,F3,7D,D3,BF,7C,D3,125D
1370 DATA BF,00,00,00,00,DB,BE,FB,C9,F5,F3,7D,D3,BF,7C,F6,40,D3,BF,00,0
0,00,F1,D3,BE,FB,C9,00,21,00,F0,06,10B4
1380 DATA 17,36,00,23,10,FB,C9,00,21,19,F0,7E,81,77,FE,0A,38,0F,D6,0A,7
7,23,7D,FE,1E,28,06,34,7E,FE,0A,30,0B5E
1390 DATA F1,00,11,1D,F0,21,11,38,1A,FE,00,20,08,1B,23,7B,FE,18,C8,18,F
3,1A,CD,89,EF,1B,23,7B,FE,18,C8,18,0C69
1400 DATA F4,21,20,38,CD,7A,EF,4F,06,1F,23,CD,7A,EF,2B,CD,89,EF,23,10,F
5,79,CD,89,EF,C9,FF,FF,FF,FF,FF,FF,1389
```

Δ

Choplifter

Yes, it's yet another arcade conversion. This time it's a copy of SEGA's 1985 arcade smash hit in which you must fly a helicopter into enemy territory to rescue trapped POW's.

There are 64 POW's trapped in huts behind enemy lines. You have to fly in, free them, then load them into your helicopter (equiped with machine gun and bombs) and return them to base. All this time you are dodging tanks, jets and satellites. If you can manage this you will advance to round 2, where you must rescue prisoners trapped in island huts off the coast of Japan.

The scrolling is good and the animation of your helicopter and the POW's is excellent - good copy of the original.

Name: Choplifter
Options: 1/2 Players
System: Cartridge - Poseidon Software Hire Club

Continued on next page

Wonderboy

This game has got to be my favourite, the colour and graphics are brillant. It is, (like almost everything else) an arcade conversion and the best one I've seen.

The game itself is about Wonderboy (a mini Tarzan) who's girlfriend has been taken by an evil demon. Armed with only an axe, he must jump, shoot and dodge deadly animals and bottomless pits to rescue her. You control Wonderboy in his quest to kill the demon and get his girl back. But the demon has many allies - as well as dodging cobras, spiders and wasps each round has a temple at the end of it, gaurded by either a Roc or Cyclops. The colour and detail of the temple is amazing.

Couple all this with the addictive music and playability of the game you'll see that Wonderboy is one of the best games available.

Name: Wonderboy
Options: 1/2 Players
System: Cartridge - Poseidon Software Hire Club

Championship Load Runner

If you liked Load Runner you'll love this. It is the most challenging game I've seen yet. Level 1 is better that the first five of Load Runner.

All the levels require quick thinking and good reactions. The levels are bigger than Load Runner and everyone of them has a name and a password.

You are given the password when you complete a level so that by typing in the password at the start you can begin from that level. For those of you who have never seen Load Runner to complete a level you must clear the screen of gold chests while dodging robot gaurds and then escape to the top of the screen.

If you are looking for a challange then this is the one to get - But it's advisiable to get Load Runner first.

Name: Championship Lode Runner
Options: 1/2 Players
System: Cartridge - Poseidon Software Hire Club

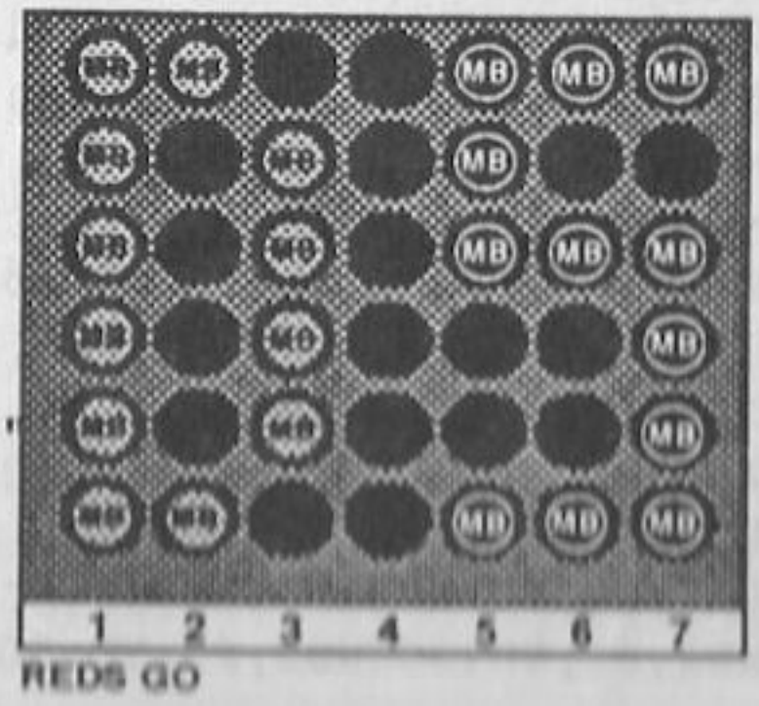
Δ

Connect Four

By Denver Scott

This is a two player version of the game by Milton Bradley
for 32K and Disk Users

```
1 REM Connect Four
2 REM
3 REM Written By
4 REM Denver Scott
5 REM
6 REM For MJH Software
7 REM
8 REM May 1988
9 REM
10 LX=&HE88E:VX=&H3B01
20 LY=&HE88F:VY=&H3B00
30 NM=&HE890:CR=&HE896
40 CL=&HE891:SU=&HE991
50 CM=&HE894
60 AD=&HE892:BD=&HED00
70 SB=&HE978:MS=&HE800
80 GOSUB780
90 SCREEN2,2
100 GOSUB350
110 M$="REDS GO"
120 GOSUB370
130 M=1
140 GOSUB390
150 POKE LX,X(COL)
160 POKE LY,Y(K):POKE NM,0
170 POKE CL,11
180 CALL MS
190 GOSUB480
200 REM
210 REM
220 REM
230 REM
240 GOSUB350
250 M$="BLACKS GO"
260 GOSUB370
270 M=2
280 GOSUB390
290 POKE LX,X(COL)
300 POKE LY,Y(K):POKE NM,4
310 POKE CL,11
320 CALL MS
330 GOSUB480
340 GOTO100
350 BLINE(0,181)-(255,191),,BF
360 RETURN
370 COLOR1:CURSOR16,182:PRINTM$:BEEP
380 RETURN
390 I$=INKEY$
400 IFI$<"1"ORI$>"7"THEN390
410 COL =VAL(I$)-1
420 POKE CM,COL
430 CALL SB
440 K=PEEK(CL)-1
450 IF K=-1THENBEEP1:BEEP0:GOTO390
460 POKE BD+COL+K*7,M
470 MAG1:RETURN
480 Y=VPEEK(VY):X=VPEEK(VX)
490 COLOR11:N=PEEK(NM)
500 CURSORX,Y
510 PRINTCHR$(N+200)
520 CURSORX+8,Y:PRINTCHR$(N+202)
530 CURSORX,Y+8:PRINTCHR$(N+201)
540 CURSORX+8,Y+8:PRINTCHR$(N+203)
550 CALL CR
560 IF PEEK(CL)>0THEN590
570 CHIPS=CHIPS+1:IFCHIPS>41THEN600
580 RETURN
590 GOSUB1410
600 SCREEN1,1:CLS
610 WHO=PEEK(CL)AND3
620 IF WHO=0THENWIN$="No-one !"
630 IF WHO=1THENWIN$=P1$
640 IF WHO=2THENWIN$=P2$
650 CURSOR4,10:PRINTWIN$;" Connected four"
660 BEEP
670 CURSOR4,15:PRINT"Press any key"
680 IFINKEY$=""THEN680
690 GOTO10
```



```
700 DATA00071F3F7F6EE4EA,EEEE7F7F3F1F0700,00C0F0F8FC8CB68E,B68EFCFCF8F0C0
00
710 DATA0007182040519B95,9191404020180700,00C0300804744A72,4A7204040830C0
00
720 DATA00,00,00,00
730 DATA *
740 REM
750 DATA16,20,191,170,16,170,191,180,212,22,237,178,210,20,239,180,1,1,1,
1
760 REM
770 DATA C,O,N,N,E,C,T,F,O,U,R
780 IF(PEEK(&HE800)XORPEEK(&HE801))<>210THENGOSUB1650

790 GOSUB1310
800 RESTORE700
810 N=0
820 READQ$:IF Q$="*"THEN850
830 PATTERNS#N,Q$:PATTERNC#N+200,Q$
840 N=N+1:GOTO820
850 SCREEN2,1:CLS
860 COLOR1,15,(0,0)-(255,191),15
870 COLOR11,1,(20,20)-(190,170)
880 RESTORE750
890 READA,B,C,D:IF A=1THEN920
900 LINE(A,B)-(C,D),1,B
910 GOTO890
920 ERASE
930 A=0
940 FORN=26TO170STEP24
950 X(A)=N
960 A=A+1:NEXT
970 A=0
980 FORN=25TO145STEP24
990 Y(A)=N
1000 A=A+1:NEXT
1010 N=0
1020 FOR B=32TO160STEP24
1030 CIRCLE(X(N)+6,B),10,11,1,0,1
1040 NEXT

1050 N=N+1:IFN>6THEN1070
1060 GOTO1020
1070 PAINT(30,21),11
1080 RESTORE770
1090 PRINTCHR$(17):COLOR1
1100 A=220:FORB=30TO90STEP10
1110 READA$:CURSORA,B:PRINTA$
1120 NEXTB
1130 FORB=120TO150STEP10
1140 READA$:CURSORA,B:PRINTA$
1150 NEXTB
1160 PRINTCHR$(16)
1170 N=0
1180 CURSORX(N),172:PRINTN+1
1190 N=N+1:IF N>6THEN1210
1200 GOTO1180
1210 CHIPS=0
1220 REM
1230 REM Call M/code routine to
1240 REM Blank board
1250 REM
1270 CALL SU
1280 REM
1290 REM
1300 RETURN
1310 SCREEN1,1:CLS

1320 CURSOR3,2:PRINT"C O N N E C T   F O U R"
1330 CURSOR2,5:PRINT"PLAYER  1 [RED]"
1340 CURSOR1,7:INPUT" ";P1$
1350 CURSOR2,10:PRINT"PLAYER  2 [BLACK]"
1360 CURSOR1,12:INPUT" ";P2$
1370 P1$=MID$(P1$,1,10)
1380 P2$=MID$(P2$,1,10)
1390 FORN=1TO200:NEXT:CLS:CURSOR5,10:PRINT"Please wait a mo'"
1400 RETURN
1410 RESTORE1460:GOSUB1500
1420 VO=12
1430 READ F2,F3,F1,D
1440 IFF2=0THENSOUND0:RETURN
1450 SOUND2,F2,VO:SOUND3,F3,VO:SOUND1,F1,VO:FORL=1TOD/5:NEXT:GOTO1430
```

```
1460 DATA 932,1864,165,30,784,1568,311,15,932,1864,311,15,932,1864,330,15
,784,1568,330,15,932,1864,330,15,784,1568,330,15
1470 DATA 698,1397,349,15,932,1864,349,15,1175,2350,349,15,1397,2794,349,
15,1397,2794,175,15,1175,2350,175,15
1480 DATA 932,1864,175,15,698,1397,175,15,784,1568,262,240,932,1864,262,2
40,1175,2350,175,120,1047,2094,175
1490 DATA 240,932,1864,175,120,932,1864,233,320,932,1864,175,320,932,1864
,117,720,0,0,0,0
1500 WH=PEEK(AD)+256*PEEK(AD+1)-60672
1510 A=WH MOD7
1520 B=INT(WH/7)
1530 X=X(A)
1540 Y=Y(B)
1550 F=(PEEK(CL)AND252)
1560 IF F=4THENX=X+4:X1=X+76:Y=Y+5:Y1=Y+2:GOTO1630
1570 IF F=8THENX=X+5:X1=X+2:Y=Y+4:Y1=Y+76:GOTO1630
1580 IF F=16THENX=X+4:Y=Y+4:X1=X+76:Y1=Y+76
1590 IF F=32THENX=X+6:Y=Y+4:X1=X-76:Y1=Y+76
1600 BLINE(X,Y)-(X1,Y1)
1610 BLINE(X,Y+1)-(X1,Y1+1)
1620 RETURN
1630 BLINE(X,Y)-(X1,Y1),,BF
1640 RETURN
1650 X=&HE800:RESTORE10000
1660 FORN=10000TO10120STEP10:C=0
1670 CURSOR0,0:PRINTHEX$(X)
1680 FORM=0TO31:READA$:POKEX,VAL("&H"+A$):C=C+PEEK(X):X=X+1:NEXTM
1690 READA$:IFC<>VAL("&H"+A$)THENBEEP2:PRINT"Error in line ";N:STOP
1700 NEXT:RETURN
10000 DATA F3,21,0,3B,CD,7A,E8,21,0,4,7D,D3,BE,0,0,0,7C,D3,BE,0,0,0,3A,90
,E8,D3,BE,0,0,0,3A,91,BCC
10010 DATA E8,D3,BE,0,0,0,FB,76,F3,21,1,3B,CD,85,E8,DB,BE,C6,3,21,8E,E8,B
E,30,D,21,1,3B,CD,7A,E8,D3,FC7
10020 DATA BE,0,0,0,18,E0,FB,76,F3,21,0,3B,CD,85,E8,DB,BE,C6,3,21,8F,E8,B
E,30,D,21,0,3B,CD,7A,E8,D3,F03
10030 DATA BE,0,0,0,18,E0,21,2,3B,CD,7A,E8,3E,8,D3,BE,0,0,0,AF,D3,BE,0,0,
0,C9,F5,7D,D3,BF,7C,F6,D99
10040 DATA 40,D3,BF,F1,C9,F5,7D,D3,BF,7C,D3,BF,F1,C9,0,0,0,0,0,0,0,ED,F3,
AF,32,91,E8,CD,AB,E8,D8,CD,1297
10050 DATA DC,E8,D8,CD,10,E9,D8,CD,44,E9,C9,21,0,ED,11,3,0,6,6,C5,6,4,C5,
6,3,7E,A7,28,15,E5,2C,BE,DF9
10060 DATA 20,F,10,FA,E1,C1,C1,22,92,E8,F6,4,32,91,E8,37,C9,E1,2C,C1,10,E
0,19,C1,10,D9,AF,C9,21,0,ED,11,FF5
10070 DATA 7,0,6,7,C5,6,3,C5,6,3,7E,A7,28,15,E5,19,BE,20,F,10,FA,E1,C1,C1
,22,92,E8,F6,8,32,91,E8,CAF
10080 DATA 37,C9,E1,19,C1,10,E0,7D,D6,14,6F,C1,10,D6,AF,C9,21,0,ED,11,8,0
,6,6,C5,6,4,C5,6,3,7E,A7,C95
10090 DATA 28,15,E5,19,BE,20,F,10,FA,E1,C1,C1,22,92,E8,F6,10,32,91,E8,37,
C9,E1,2C,C1,10,E0,7D,C6,3,6F,C1,1016
10100 DATA 10,D6,AF,C9,21,6,ED,11,6,0,6,3,C5,6,4,C5,6,3,7E,A7,28,15,E5,19
,BE,20,F,10,FA,E1,C1,C1,BE9
10110 DATA 22,92,E8,F6,20,32,91,E8,37,C9,E1,2D,C1,10,E0,7D,C6,B,6F,C1,10,
D6,AF,C9,F3,2A,94,E8,6,FF,4,7E,1118
10120 DATA A7,20,9,7D,C6,7,6F,78,FE,6,20,F2,78,32,91,E8,C9,21,0,ED,11,1,E
D,1,29,0,36,0,ED,B0,C9,FF,DD5
```

BOO BOO'S

The mystery program was left off the magazine tapes for the issue 3 and has already been added to the start of the magazine tape for this issue.

The VRAM to RAM and RAM to VRAM machine code routines listed in the Letters to the Editor, do actually work. However to make them work slightly better, add this line at the start of each routine to disable interrupts ...

```
START     F3        DI
```

Line 1600 of **Tank Battle** was printed incorrectly (a "7" became a "6") due to a communication error when sending the program from the SEGA to the Macintosh. It should have read

```
1600 DATA 01070207100F080F010F020740078007800F0800F090F190F7
10FE00FE00FE00FE00,093E
```

The instructions for typing in the program were a little bit ambiguous.

On page 23

All Cartridge Basic Users (incl LSV Users) add

```
1480 FORN=0TO63:A=PEEK(&H4020+N)
1520 C=&H1800+PEEK(&H3FA0+N)*8
```

On page 24

Cartridge Basic Users wihout LSV add

```
75 CLS:PRINT:PRINT"LOAD":PRINTCHR$(11);
```

and of course a few spelling mistakes. Δ

In the next issue

Information on keyboard scanning,
Video Display Processor
and from the disk drive information about
RS-232, centronics and Floppy Disk Controllers

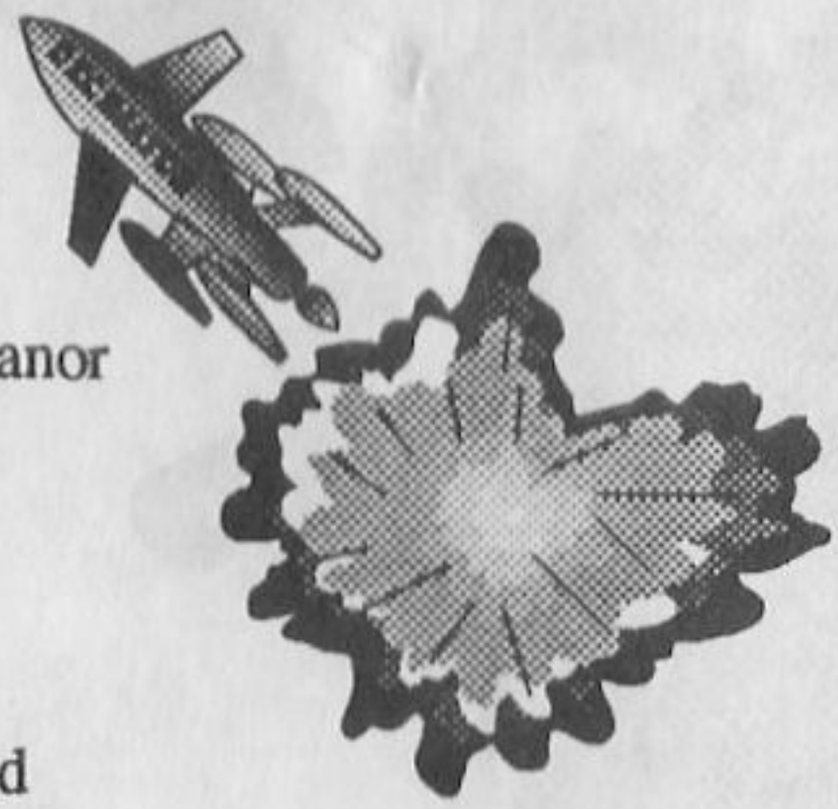
More on machine code programming and hints

Due out about July

Poseidon Software

NZ SEGA DISTRIBUTORS

FREEPOST 243
P.O. BOX 277
TOKOROA
NEW ZEALAND



Decimator	*32K only*	$27.95
Poker	*32K only*	$24.95
Delta Fighter	*32K only*	$24.95
Burgular Bill	*16K / 32K*	$17.95
Carverns of Karanor	*16K / 32K*	$19.00
Vortex Blaster	*32K only*	$12.00
Aerobat	*32K only*	$19.95
Orb of Power	*32K only*	$12.00
Castle of Fear	*32K only*	$12.00
Castaway	*32K only*	$12.00
Burgular Bill and Caverns of Karanor	*on one disk*	$50.00
Michael Howard's - More than 50 Programs		$5.00
Book and Tape - Teach Yourself BASIC Programming		$6.00
LSV, Print 64 and Pattern Paint	*Disk / Tape versions*	$15.00
Magazine programs on cassette		$20.00
Machine Code Summary Sheets		$1.00

	Australia	**New Zealand**	
Magazine subscription (6 issues)	A$26	NZ$25	GST incl

Name _____ **Phone** _____

Address _____

_____

Item	**No. of items**	**$ Total**

Add Postage **$2.50**

Payment by (circle one) **Total Enclosed $** _____

Cheque Cash Bankcard Visa

Orders over $20.00 only

Credit Card No._ _____

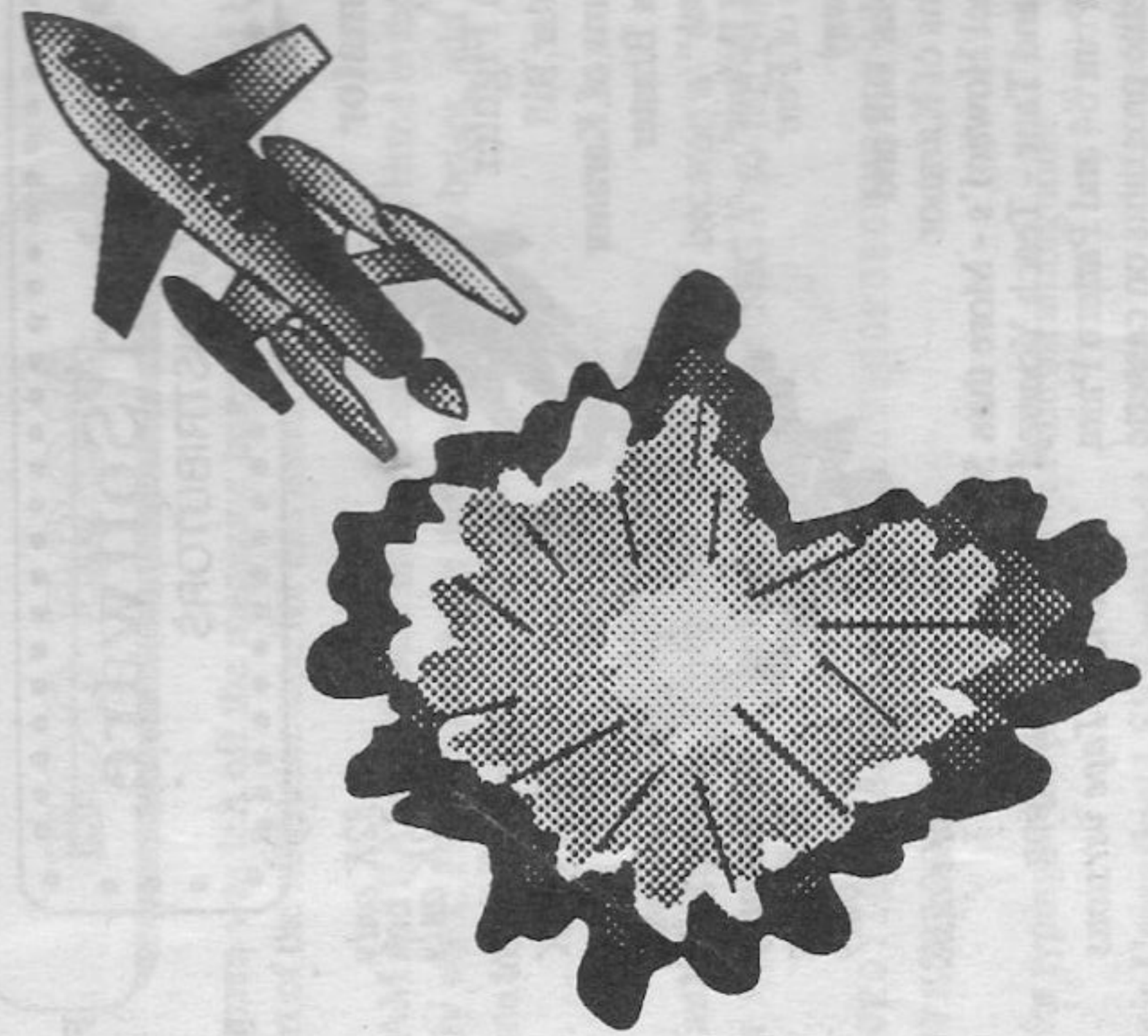
Signed _____ Expires / /

VISA

BANKCARD Welcome here

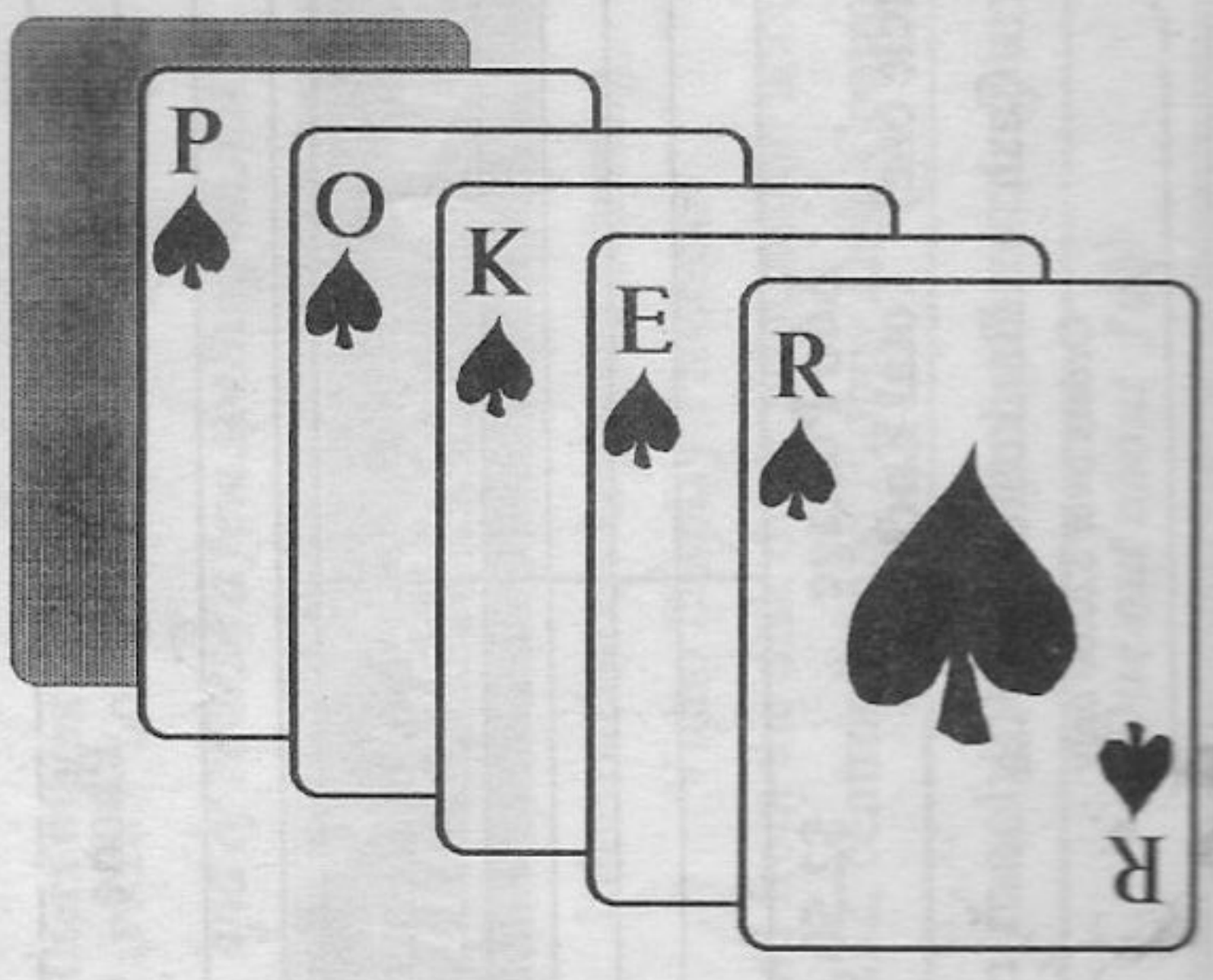
DECIMATOR

32K only



Written by
Michael Boyd
For Poseidon Software

POKER



Written by T. R. Speirs
for Poseidon Software