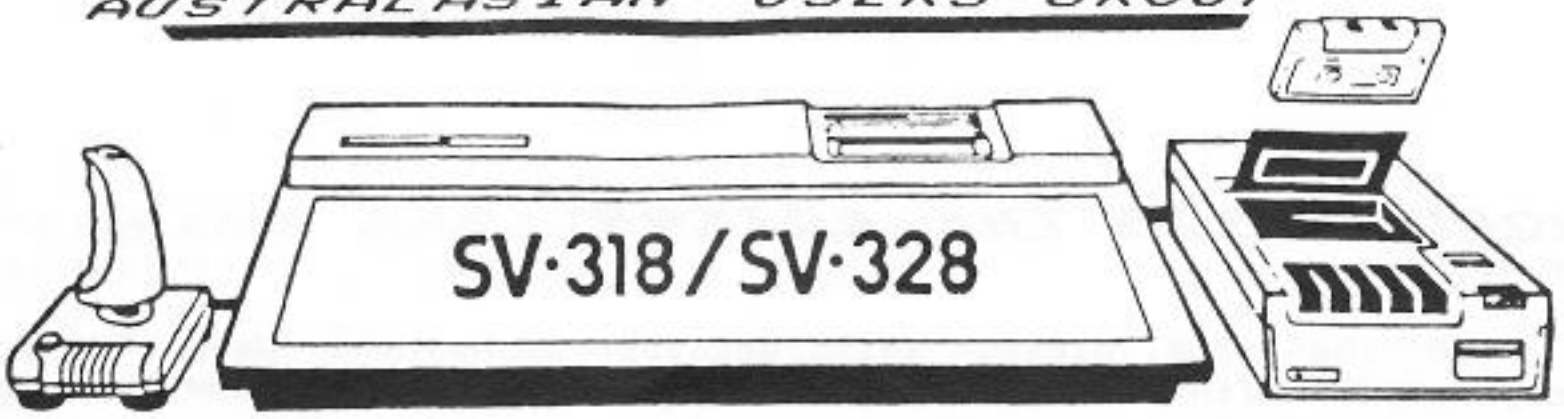
SPECTRAVIDEO™

AUSTRALASIAN USERS GROUP



News Letter

CONTENTS	PAGE

ISSUE No.
1 - 8
DATE
MAY - 1984

ALL CORRESPONDENCE TO:
S. A. U. G., P. O. BOX 191, SOUTH LAUNCESTON, TASMANIA, 7250. (003) 312648

MEMBERSHIP FEES	
AUSTRALIA	$15.00
OVERSEAS	$20.00
OVERSEAS AIRMAIL ..	$25.50

SPECTRAVIDEO™
AUSTRALASIAN USERS GROUP

WORDS FROM THE EDITOR

by ME

On the 17th, 18th, 19th and 20th of May the User Group in conjunction with M.T.C JESSUPS were part of a Towards 2000 Expo held at The Albert Hall in Launceston. I am very proud to announce that the Spectrvideo stand won the Most Outstanding Exhibitors Award. Well done Bruce. We created much interest in the Spectravideo Computers and I extend a big HELLO to all the new members who signed up for the group over those four days. For those not able to attend the expo, the competition we had included Apple/Macintosh, Eliza, Dick Smith Cat and Challenger, Tandy, Digital plus various other assortments. From all accounts the expo was a great success to all who were exhibitors and the patrons who graced the expo itself. The group are definitely going to be there next year and hopefully bigger and better at that.

On the interesting news side all you 328 owners who have 318 manuals can now take them back to their retailers, and have them exchanged for the new SV 328 manual which I'm pleased to say is error free. The manuals will be replaced at no charge.

A new keyboard for the SV 318 is now available, I've seen one and it appears to be the same as the 328 keyboard. So all you 318 owners who are sick of the rubber keypad can now upgrade to a fullstroke keyboard.

It seems also that a new Super Expander with two built in disk drives is to be released in a few weeks, I'll keep you all posted next issue if more details are available.

Also it's possible to do an onboard memory upgrade on an SV 318 to give it the same memory as the 328. This will probably cost about $80.00. It is not yet certain if the person concerned will do the conversions himself or publish the details. We hope to have the answer by the next newsletter.

SPECTRAVIDEO™
AUSTRALASIAN USERS GROUP

EXPLORING BASIC Pt. 4

by L.A. Dunning

This month, I give some detail to the Modes of Display used by the Spectravideo and some software support given to it. You may have been wondering how basic kept record of the TMS 99ØØ registers if they are write only and not read?

The answer is that it uses several mirror locations in high memory to do so. Not all registers are noted however, only registers Ø, 1 and 7. FE3CH copies register Ø, FAØ7H copies register 1. FAØAH, FAØBH, FAØCH are pidgeon holes for foreground, background and border colours respectfully. Other registers are set to default values and don't change unless another SCREEN type is set up, so all driver routines in basic are designed around these values. The values of each register (and equivalent VRAM locations) are listed in Table 1.

TABLE 1

SCREEN (Mode)	Ø Text)	1 (Grph II)	2 (Multi)
REGISTER			
2 NTB	ØØ (ØØØØ)	Ø6 (18ØØ)	Ø2 (Ø8ØØ)
3 COLB	NU	FF (3FCØ)	NU
4 PGB	Ø1 (Ø8ØØ)	ØØ (ØØØØ)	ØØ (ØØØØ)
5 SAB	NU	36 (1BØØ)	36 (1BØØ)
6 SPGB	NU	Ø7 (38ØØ)	Ø7 (38ØØ)

NU = Not used in that mode

TEXT MODE (SCREEN Ø)

The Display is in text mode when bit 4 of register 1 is set to 1 and bit 3 of the same register and bit 1 of register Ø are set to Ø. In this mode the screen is divided into 4Ø positions across by 24 positions down. Each position is represented by a single byte in the Name Table (NTB) which is 96Ø bytes long, so each position could be any of 256 patterns. The Pattern Generator Table (PGB) contains a definition of 256 Patterns, each 8 bytes long, forming a block of 2Ø48 bytes. The NTB and PGB have a one-to-one correspondence so that a value in the NTB will indicate which 8 byte pattern will be displayed. Figure 1 illustrates this relationship.

Only the six most significant bits of the pattern byte are displayed. The top 4 bits of register 7 define the text colour, while the bottom 4 define the background. When SCREEN Ø is initialised, basic sets up the proper addresses and colours, then clears the screen by zeroing all bytes on the NTB. The 256 patterns are then loaded from a table and in compressed form into the PGB. This table starts at 4198H with the compressed code for SPACE and ends at 4551H with the codes for "i" in Suzuki. Listing 1 is a simple program to demonstrate this coding.

The location of the first video byte of any pattern in PGB is equal to 2Ø48 + (P x 8) where P is the pattern number. The patterns are equivalent to the following ASCII characters:

PATTERNS	ASCII (CHR$)	
Ø-95	32-127	ASCII Character Set
96-191	32-127	Inverse ASCII
192-255	16Ø-223	Graphics Characters

Nothing stops a user from modifying any of these characters. Basic uses pattern 191 as the cursor character. When the cursor is on, it redifines that pattern as an inverse or composite of the pattern underneath the cursor and puts the cursor value at that location. The X axis value of the cursor is stored at FAØ4H, the Y axis value at FAØ3H.

There are two other memory locations of interest to the user. F543H stores the length of each screen line. Storing a vaiue less than 4Ø here will stop the far left column from being affected by PRINT statements (ala WIDTH 4Ø); a value less than 39 will protect a right portion of the screen; a value greater than 4Ø produces unpredictable results. FAØ6H stores a value used scrolling the screen; the number of lines affected by a scroll (as counting from the top line down) is equal to V - 232, where V is the value stored at FAØ6 (Ø counts as 256). This is normally used to protect the KEY prompts at the bottom of the display.

Basic also allows some manipulation of the screen by using PRINT followed by CHR$(27) - ESCAPE. The character values after the CHR$(27) determine the affect of the print as listed below:

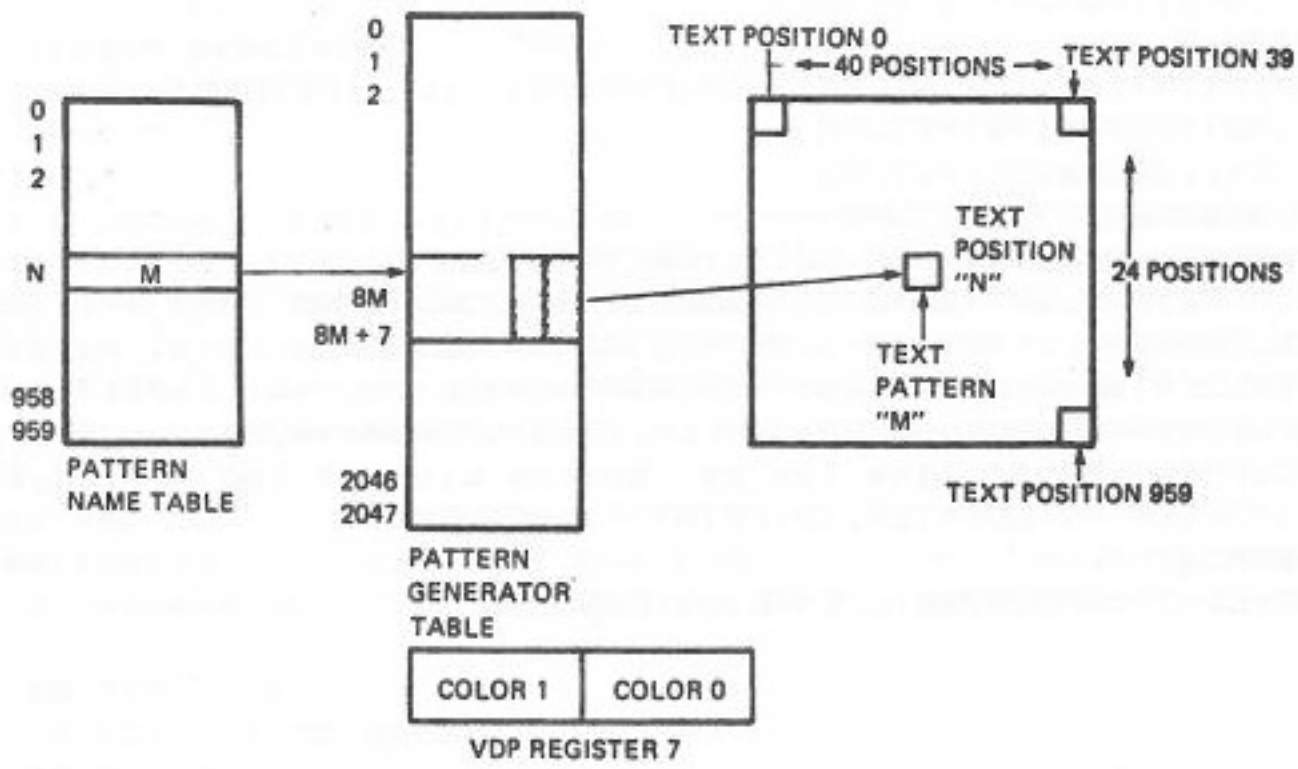
"A"	Moves Cursor UP	"L"	Scrolls Screen DOWN(2)
"B"	Moves Cursor DOWN	"M"	Scrolls Screen UP(2)
"C"	Moves Cursor RIGHT	"Y"yx	Position Cursor(3)
"D"	Moves Cursor LEFT	"j"	Same as "E"
"E"	CLS + Homes Cursor	"l"	Clears line
"H"	Homes Cursor	"p"	Turns on REVERSE ASC
"J"	Clears to end of Screen(1)	"q"	Turns off REVERSE ASC
"x4"	Set Cursor BLOCK	"y4"	Set Cursor INSERT
"x5"	Turns Cursor OFF	"y5"	Turn Cursor ON

(1) Will CLS Screen from last PRINT position.
(2) Will scroll in direction, top of scroll is row used in last PRINT or LOCATE statement, bottom is determined by FAØ6H.
(3) y and x are ASC coded characters for Y and X locations, add 2ØH (32) to each value to produce resultant characters.

Listing 2 demonstrates using the above knowledge. The following USR calls might prove handy as well; 3541H will set up SCREEN Ø; 3B86H turns off the Prompts; 3B9FH turns on the Key Prompts; 3498H restores the KEY buffer to original contents; 37ØAH will CLS the screen; 3584H will restore pattern values. Next month I will describe how SCREEN 1 & 2 work.

Remember last issues listing of the SCREEN Ø Designer? Well a routine that displayed the Keys and input File Names used a "bug" in the SPRITE$( ) statement coding. There is only 2K of VRAM available for the Sprite Pattern Generator Table (SPGB), when sprites are 8 x 8 this gives an available 256 different patterns, when sprites are 16 x 16 this gives only 64 different patterns. What happens to the other 192 patterns? They do not exist but the coding still works!

Using A$=SPRITE$(X) and SPRITE$(X)=A$ when the Sprite size is 16 x 16 and X is over 63, will get and put values from/to the screen. Each pattern is still 32 bytes long, so in SCREEN Ø it affects a 32 x 8 pixel rectangle; in SCREEN 1 it affects an 8 x 8 point rectangle. When reading a screen sprite however, allowance must be made that the initial address used is one less than the address used to write the sprite. This is I believe due to the autoincrementation used by the VDP. The values to use are SCREEN Ø:64 - 93; SCREEN 1:64 - 255; SCREEN 2:64 - 111. How do you get Sprites in SCREEN Ø? Simple, go to SCREEN 1 or 2 first with values 2 or 3, then switch back to SCREEN Ø. There is one limitation however, you can't dump an on screen sprite to another on screen sprite directly; SPRITE$(65)=SPRITE$(77) will not work. A$=SPRITE$(77) : SPRITE$(65)=A$ will work however.

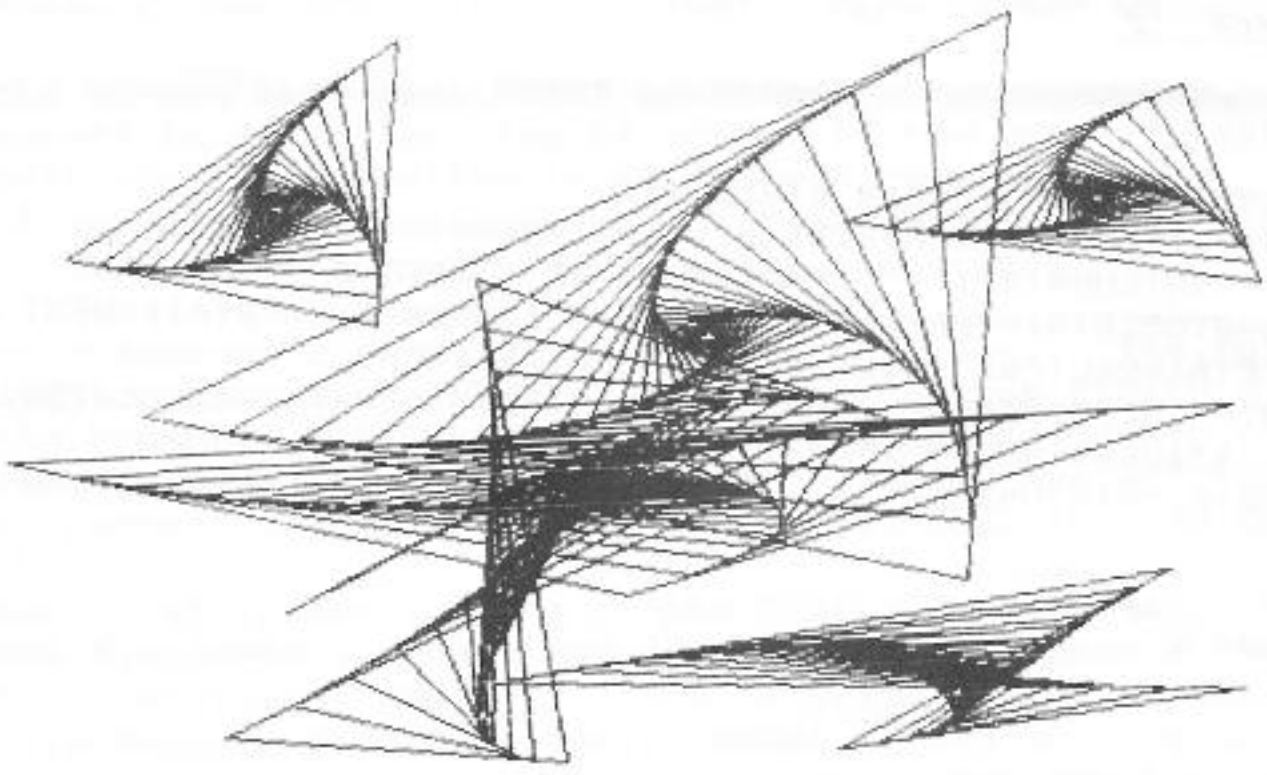


LISTING 1

```
1Ø CLS:CLEAR2ØØØ:SCREENØ,Ø:DIM B(6),P(8)
2Ø DEF FNB$(X)=STRING$(8-LEN(BIN$(X)),"Ø")+BIN$(X)
3Ø LOCATE5,5:PRINTCHR$(27)"1";:INPUT"PATTEN NUMBER Ø - 158";PN
4Ø B$="":FORA=ØTO5:B(A)=PEEK(&H4198+PN*6+A):B$=B$+FNB$(B(A)):NEXT
5Ø FORA=ØTO7:P(A)=VAL("&B"+MID$(B$,A*6+1,6))*4:NEXT
6Ø LOCATE5,1Ø:PRINT"BYTES       PATTEN":FORA=ØTO7:IFA<6THENLOCATEØ,12+A:PRINTUSIN
G"\   \ \       \";HEX$(&H4198+PN*6+A),FNB$(B(A));
7Ø LOCATE16,12+A:PRINTUSING"\       \ #";FNB$(P(A)),A;:NEXT:GOTO3Ø
```

LISTING 2

```
10 CLS:WIDTH40:SCREEN0,0:LOCATE,,0:CLEAR2000:PX=&HFA04:PY=&HFA03:PL=&HF543:SP=&H
FA06:C$=CHR$(27):SD$=C$+"L":SU$=C$+"M":CL$=C$+"1":J=RND(-TIME)
20 STOPON:ONSTOPGOSUB240:ONERRORGOTO230:KEYON:ONKEYGOSUB70,80,90,100,110,120,130
,140,150,160
30 DEF FNR(X)=FIX(RND(X)*X+1):FORA=1TO10:D$(A)=STRING$(40,34+A):NEXT
40 LOCATE0,10,0:PRINTC$"l";:INPUT" Line Length, Bottom Line, Top";LL,BS,TP:IFLL>
40ORLL<10ORBS>23ORBS<5ORTP<0OR(TP>BS-2)GOTO40ELSECLS
50 GOSUB170
60 GOTO60
70 LOCATE0,TP:PRINTSU$;:LOCATE0,BS-2:PRINTLEFT$(D$(FNR(10)),LL);:RETURN
80 LL=LL+(LL>10):GOSUB170:RETURN
90 BS=BS+(BS>4):GOSUB170:RETURN
100 TP=TP-(TP<BS-3):GOSUB170:RETURN
110 CLS:GOSUB170:RETURN
120 LOCATE0,TP:PRINTSD$;:PRINTLEFT$(D$(FNR(10)),LL);:RETURN
130 LL=LL-(LL<40):GOSUB170:RETURN
140 BS=BS-(BS<24):GOSUB170:RETURN
150 TP=TP+(TP>0):GOSUB170:RETURN
160 FORA=1TOFNR(10):PRINTD$(FNR(10));:NEXT:RETURN
170 POKEPL,40:POKESP,0:SV=(BS+232)MOD256:IFBS=23GOTO200
180 IFTP>0THENLOCATE-(LL<40),TP-1:PRINTSTRING$(LL,210);
190 IFBS<23THENLOCATE0-(LL<40),BS:PRINTSTRING$(LL+(LL=40),210);
200 IFLL<40THENFORA=TPTOBS:VPOKEA*40+LL+1,242:VPOKEA*40,242:NEXT
210 LOCATE0,23:PRINTUSING" Line len ##  Bottom Line ## Top ##";LL,BS,TP;
220 POKEPL,LL:POKESP,SV:LOCATE0,TP:PRINT"";:RETURN
230 PRINTERL;ERR:STOP
240 TP=0:BS=23:LL=39:GOSUB220:LOCATE,,1:END
```



PROGRAMMING HINT

by M.J. Tyeson.

It appears that most people don't realise that when you first turn your computer on for a programming session Spectravideo Basic initialises all variables to be double precision. This is great if you want to do complex mathematical equations that require 16 decimal places of accuracy. This, however has one big drawback, that being it slows program excecution times markedly. This ruins the effect of games written in basic. So if you don't need all that accuracy and be honest who ever does? You can speed things up by defining your variables at the beginning of the program. There are four types of variables:

1) Double precision (DBL)
2) Single precision (SNG)
3) Integer (INT)
4) Strings (STR)

I won't bother about strings at the moment, I will leave that up to you to fiddle with. To define variables at the beginning of the program you use the DEF <vartype> [<exp>[,[-<exp>]]...].
to put that in laymans terms the following are some examples.

DEFINT A-Z Defines all variables to be integers
DEFSNG A,M-O Defines A,M,N,O to be single precision

And so on.

The following program using the TIME function shows you the time differences between INTEGER, SINGLE, and DOUBLE precision mathematics.

```
20 TIME = 0
30 FOR I=1 TO 20000
40 X = I
50 NEXT I
60 PRINT "Time taken was "TIME/60" seconds."
```

Run the program with one of the following line 10's and compare the three times taken

1)

1.

1)

Using
places and
hell of alo

SPECTRAVIDEO™

AUSTRALASIAN USERS GROUP

INTERESTING FACTS ABOUT SV

I received some interesting bits and pieces about Spectravideo marketing around the world as compared with Australia. They are listed below.

In the U.S.A. and the U.K. the SV 328 comes with the original 48K ROM. This ROM contains the Word Processor and Networking Facility. In Australia a cartridge has just been released (at extra cost) with these facilities. WHY??

In U.S.A. and U.K. the Colecovision Adapter is sold at an equivalent price to here but with two SV joysticks.

An adapter similar to the Colecovision Adapter will be required to run MSX cartridges. This adapter will be an additional cost to a machine that is advertised as MSX compatible. The new machines we believe will be manufactured with the standard larger MSX slot instead of the small SV slot.

The quality of recent SV software releases is very very poor and when asked about high quality games we only get promises. How much longer do we have to wait?

The Spectravideo Basic is almost MSX standard but there are small differences and no matter how small they are it does not make it MSX compatible. Some examples are:-

Our Cassette baud rate will not allow us to read MSX tapes.

The MSX screen comand looks like this

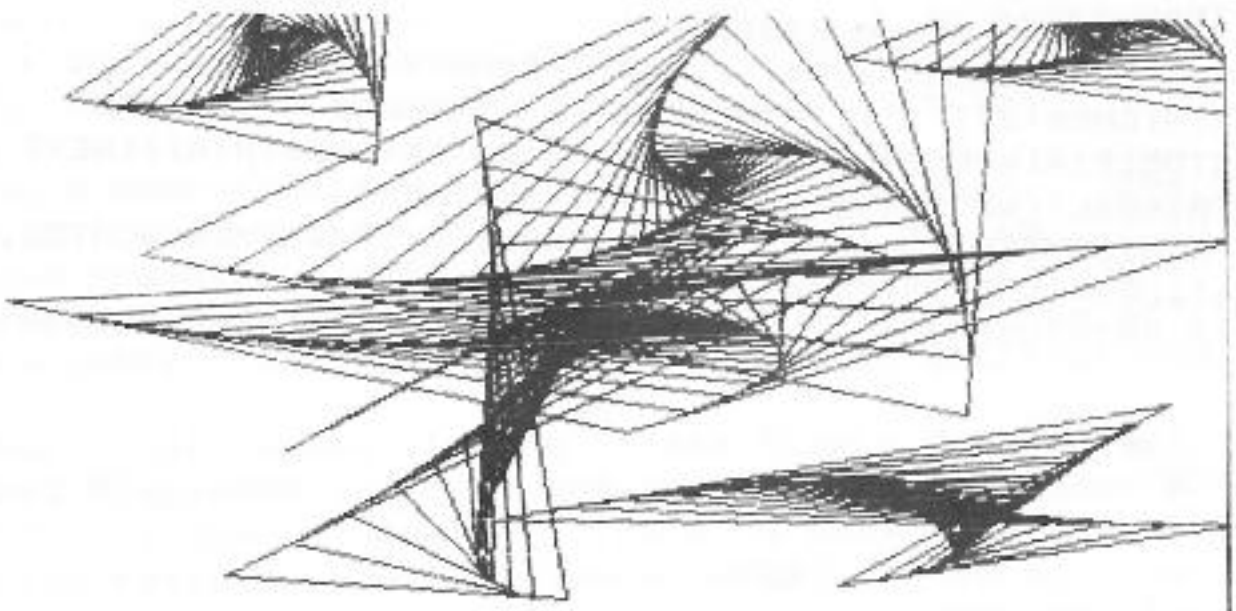
```
SCREEN (mode),(sprite size),(key click switch),
       (cassete baud rate),(printer option)
```

MSX basic also allows access to screen 3 the Multicolour mode

MSX Motor on/off command works differently to ours.

And there are more which will only depress you if I list them.

Thankyou to the people who brought the information to our attention



PLANET-X

by C. Ryan

A short but actually very difficult game to master. You are the pilot of a space shuttle that must save your home planet. It is your job to dodge through the speeding meteors (using hyperspace by pressing the SPACE BAR or TRIGGER at any time if you are really stuck) and land on one of the three landing bays. Once you land the meteors will change direction and you must fly to the top left hand side of the screen and dock with the satellite moon that will destroy the alien battlestation (pyramid). With sound effects and good graphics.

This is definitely some of the best graphics and sound effects I have seen. ED.

```
10 '
20 '
30 '        Planet-x:1984
40 '        By Caspar Ryan
50 '        Uses 5K bytes Approx.
60 '
70 '          15/12/83
80 '
90 CLEAR 100
100 PLAY"v5o614e-05r64l64gr64gl12b-14ar64af","o414gr64l64o3e-r64e-l12ge-14e-r64e
-","v5o314b-r64l64b-r64b-r64b-o4e-o314b-c-b-"
110 COLOR 15,1,1:SCREEN 2:FOR Y=1 TO 12
120 FOR T=1 TO 8:R(T)=2+INT(RND(6)*13):NEXT T:LOCATE 30,50:COLOR R(1),1:PRINT"P"
;:COLOR R(2),1:PRINT"L";:COLOR R(3),1:PRINT"A";:COLOR R(4),1:PRINT"N";:COLOR R(5
),1:PRINT"E";:COLOR R(6),1:PRINT"T";:COLOR R(7),1:PRINT"-";:COLOR R(8),1:PRINT"X
":NEXT Y
130 SCREEN 1,2
140 PLAY"v6o514e-r64l64gr64gl12b-14ar64af","o414gr64l64o3e-r64e-l12ge-14e-r64e-"
,"v6o314b-r64l64b-r64b-r64b-o4e-o314b-c-b-"
150 STOP ON:ON STOP GOSUB 870:F=0:RESTORE 880:FOR I=1 TO 8:READ C:Z$=Z$+CHR$(C):
NEXT:SPRITE$(1)=Z$
160 FOR H=1 TO 8:READ PB:S$=S$+CHR$(PB):NEXT H:SPRITE$(A)=S$
170 DEFINT A-Z:ON SPRITE GOSUB 460
180 ONSTOPGOSUB870:STOPON:CLICKOFF
190 ON INTERVAL=100GOSUB 910
200 CLS:FOR T=1 TO 80:Q=INT(RND(1)*240):P=INT(RND(2)*168):PSET(Q,P),15:NEXT
210 CIRCLE(60,30),25,11,,,1.25:CIRCLE(152,22),8,7,,,1.2:CIRCLE(240,10),45,3,,,1.
2
220 LINE(1,175)-(20,155),8:LINE(20,155)-(40,175),8:LINE(40,175)-(50,175),8:LINE(
50,175)-(60,165),8:LINE(60,165)-(70,175),8:LINE(70,175)-(120,175),8:LINE(120,175
)-(160,135),8
230 LINE(120,175)-(160,190),8:LINE(159,135)-(159,189),8:LINE(160,135)-(160,190),
9:LINE(120,175)-(160,190),8:LINE(160,135)-(200,175),9
240 LINE(159,190)-(200,175),9:LINE(200,175)-(210,165),8:LINE(210,165)-(220,175),
8
```

```
250 LINE(225,170)-(230,175),8:LINE(230,175)-(255,175),8
260 PAINT(60,30),11:PAINT(152,22),7:PAINT(240,10),3:PAINT(165,170),9:PAINT(150,1
50),8
270 CIRCLE(245,10),4,1:CIRCLE(250,25),3,1:CIRCLE(220,5),3,1:CIRCLE(225,28),4,1:C
IRCLE(250,45),4,1:CIRCLE(240,35),3,1:CIRCLE(235,45),3,1:CIRCLE(220,40),2,1:CIRCL
E(224,14),3,1:CIRCLE(235,20),4,1:CIRCLE(235,5),2,1:CIRCLE(210,12),4,1:CIRCLE(213
,24),2,1
280 CIRCLE(64,40),5,1:PSET(64,40),1:LINE(45,175)-(45,183),15:LINE(40,175)-(50,17
5),15:LINE(95,175)-(95,183),15:LINE(90,175)-(100,175),15:LINE(242.5,175)-(242.5,
183),15:LINE(237.5,175)-(247.5,175),15
290 PLAY"v6T255L603CFA04CR6L603A04L2CR":PLAY"v6T255L604CFA05CR6L604A05L2C":LOCAT
E 90,1:PRINT"PLANET-X":SPRITE ON:INTERVAL ON
300 Q=128:P=10:PUT SPRITE 1,(Q,P),15,A:FOR T=1 TO 2500:NEXT T
310 R=R+4:PUT SPRITE 13,(R,168),15,1:X=X-3:PUT SPRITE 0,(X+128,79),14,1:X=X-3:PU
T SPRITE 2,(X+60,70),14,1:X=X-3:PUT SPRITE 3,(X,75),14,1:M=M+4:PUT SPRITE 4,(M,9
5),15,1:M=M+4:PUT SPRITE 5,(M+128,90),15,1:M=M+4:PUT SPRITE 15,(M+60,100),15,1
320 C=C-5:PUT SPRITE 6,(C,116),14,1:C=C-5:PUT SPRITE 7,(C+90,112),15,1:D=D+4:PUT
 SPRITE 8,(D,140),14,1:D=D+4:PUT SPRITE 9,(D+68,130),15,1:D=D+4:PUT SPRITE 10,(D
+155,138),14,1
330 K=K+4:PUT SPRITE 14,(K+100,168),15,1:W=W-6:PUT SPRITE 16,(W+251,150),15,1:P=
P+1:PUT SPRITE 1,(Q,P),15,A
340 IF STRIG(0)OR STRIG(1)THEN BEEP:F=F+30:LET Q=5+INT(245*RND(-TIME)):LET P=INT
(165*RND(-TIME)):PUT SPRITE 1,(Q,P),15,A:IF F>400 THEN 440
350 JS=STICK(0)+STICK(1)
360 IF JS=1 THEN P=P-0:PUT SPRITE 1,(Q,P),15,A:F=F+1:IF F>400 THEN 440
370 IF JS=3 THEN Q=Q+1:PUT SPRITE 1,(Q,P),15,A:F=F+3:IF F>400 THEN 440
380 IF JS=7 THEN Q=Q-1:PUT SPRITE 1,(Q,P),15,A:F=F+3:IF F>400 THEN 440
390 IF JS=5 THEN P=P+2:PUT SPRITE 1,(Q,P),15,A:F=F+2:IF F>400 THEN 440
400 IF P=166 OR P=167 THEN 540
410 IF P>175 THEN 450
420 SP=SP+3:PUT SPRITE 25,(SP,10),12,A
430 GOTO 310
440 SPRITE OFF:LOCATE 65,95:PRINT"FUEL SUPPLY DEPLETED!":FOR T=P TO 165:P=P+1:PU
T SPRITE 1,(Q,P),15,A:NEXTT:GOTO 460
450 LOCATE 25,95:PRINT"YOU HIT THE GROUND AT HIGH SPEED!!"
460 PUT SPRITE 1,(Q,P),8,A:SOUND 7,8
470 INTERVAL OFF:FOR Y=1 TO 10:CIRCLE(Q,P),Y,9:SOUND 6,15:SOUND 7,7:SOUND 8,16:S
OUND 9,16:SOUND 10,16:SOUND 10,16:SOUND 12,16:SOUND 13,0:NEXT Y
480 LOCATE 65,75:PRINT"PLANET-X IS DESTROYED!"
490 SOUND 7,16:FOR EX=1 TO 16:SOUND 7,16-EX:COLOR,,15:FOR Z=1 TO 20:NEXT:COLOR,,
1:FOR Z=1 TO 5:NEXT Z,EX
500 SOUND 7,1:SOUND 12,255:SOUND 11,255:SOUND 13,14:FOR EX=1 TO 16:SOUND 8,19:FO
R X=1 TO 20:COLOR,,15:FOR Z=1 TO 20:NEXT:COLOR,,1:FOR Z=1 TO 5:NEXT Z,EX
510 SOUND 7,8:PLAY"v7T110O3L4DR64DL64R64DL4R64DR64FL60R64EL6R64ER64L6DR64DR64C#R
64D"
520 LINE(160,135)-(240,15),5:LINE(160,135)-(250,20),5:LINE(250,20)-(240,15),5
530 FOR U=1 TO 60:CIRCLE(240,10),U,13:SOUND7,10:SOUND9,16-UMOD16:NEXT:FOR T=1 TO
 2000:NEXT:SOUND9,0:GOTO 810
540 IF Q=40 OR Q=41 OR Q=42 OR Q=90 OR Q=91 OR Q=92 OR Q=238 OR Q=239 OR Q=240 T
HEN 560
550 GOTO 460
560 INTERVAL OFF:RESTORE 900:FOR I=1TO32:READ TA:TA$=TA$+CHR$(TA):NEXT I:SPRITE$
(19)=TA$
570 'play""
```

```
580 SPRITE OFF:FOR ZZ=255 TO Q+10 STEP -1:PUT SPRITE 30,(ZZ,175),14,19:FOR I=1 T
O 40:NEXT I:NEXT ZZ:SPRITE ON
590 PLAY"AAADEEDFDFDF":INTERVAL ON
600 FOR T=1 TO 2000:NEXT T
610 P=P-1:PUT SPRITE 1,(Q,P),15,A
620 IF P=39 OR P=40 OR P=41 THEN 740
630 IF STRIG(0) OR STRIG(1) THEN BEEP:LET Q=5+INT(245*RND(-TIME)):LET P=1+INT(16
0*RND(-TIME)):PUT SPRITE 1,(Q,P),15,A:F=F+30:IF F>400 THEN 440
640 AD=STICK(0)+STICK(1)
650 IF AD=5 THEN P=P+0:PUT SPRITE 1,(Q,P),15,A:F=F+2:IF F>400 THEN 440
660 IF AD=7 THEN Q=Q-1:PUT SPRITE 1,(Q,P),15,A:F=F+2:IF F>400 THEN 440
670 IF AD=3 THEN Q=Q+1:PUT SPRITE 1,(Q,P),15,A:IF P=40 THEN GOTO 740:F=F+2:IF F>
400 THEN 440
680 IF AD=1 THEN P=P-3:PUT SPRITE 1,(Q,P),15,A:F=F+5:IF F>400 THEN 440
690 X=X+3:PUT SPRITE 0,(X+128,70),14:X=X+3:PUT SPRITE 2,(X+60,70),15:X=X+3:PUT S
PRITE 3,(X,70),15:M=M-4:PUT SPRITE 4,(M,90),14:M=M-4:PUT SPRITE 5,(M+60,90),15:M
=M-4:PUT SPRITE 6,(M+128,90),15
700 C=C+3:PUT SPRITE 7,(C,110),14:C=C+3:PUT SPRITE 8,(C+90,118),14:C=C+3:PUT SPR
ITE 9,(C+170,110),14:D=D-3:PUT SPRITE 10,(D,130),15:D=D-3:PUT SPRITE 11,(D+110,1
38),15:D=D-4:PUT SPRITE 12,(D+120,130),15
710 K=K+4:PUT SPRITE 16,(K+100,160),14,1:W=W+4:PUT SPRITE 15,(W+190,160),15:KK=K
K-1:PUT SPRITE 13,(KK,60),15,1:PUT SPRITE 14,(KK+100,60),15,1
720 SP=SP-3:PUT SPRITE 25,(SP,10),12,A
730 GOTO 610
740 IF Q=59 OR Q=60 OR Q=61 THEN 760 ELSE SPRITE OFF:FOR Q=1 TO 400:PUT SPRITE 1
,STEP(4,-1),15,A:NEXT:LOCATE 50,60:PRINT"YOU WERE SUCKED INTO SPACE":GOTO 470
750 IF Q=60 THEN 760
760 INTERVAL OFF:LOCATE 80,80:PRINT"YOUR WORLD IS SAVED!"
770 LOCATE 80,90:PRINT"CONGRATULATIONS"
780 LINE(64,40)-(160,135),6:LINE(140,150)-(150,160),1:LINE(150,160)-(170,150),1:
LINE(170,150)-(160,130),1:LINE(160,130)-(140,150),1:PAINT(155,155),1:A=A+1000:SC
=A-F
790 LOCATE 60,65:PRINT"YOUR SCORE IS:"SC:FOR T=1 TO 3000:NEXT T:RUN 150
800 FOR J=1 TO 1500:NEXT:GOTO 810
810 CLS
820 SCREEN 1:LOCATE 90,94:PRINT"TRY AGAIN(Y/N)"
830 AC$=INPUT$(1)
840 IF AC$="Y" OR AC$="y"THEN RUN 150
850 IF AC$="N" OR AC$="n"THEN END
860 GOTO 830
870 CLEAR:SOUND 8,0:RUN 150
880 DATA60,255,127,254,252,127,127,44,16,56,16,84,84,124,84,238,60
890 DATA 16,56,16,84,84,124,84,238,60
900 DATA 0,0,0,0,0,130,254,135,1,0,1,255,145,68,36,31,0,0,0,4,200,208,88,248,224
,196,248,255,17,66,68,248
910 PLAY"v10t255o2l64fgfg"
920 RETURN
```

SV 807 SWITCHES

In answer to many queries about switch positions on the SV 807 64K RAM card the following was supplied by Mr. K. Beattie.

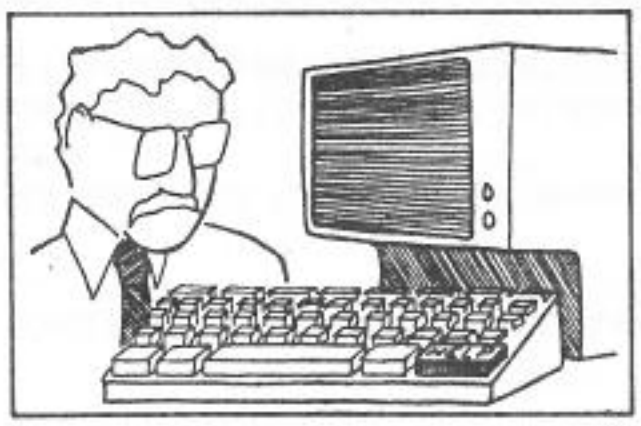
SV 807	SV 318	SV 328
S1	ON/OFF	OFF
S2	ON/OFF	ON/OFF
S3	ON/OFF	ON/OFF
S4	ON/OFF	ON/OFF
S5	ON (")/OFF	OFF
S6	OFF	OFF

Note: 1). Only two 'ON' is allowed set switch to left (on position)
2). (") can be selected only if SV 803 16K RAM cartridge is not used.

PIANO PROGRAM

by W. Szapirko

The following is a fun little program that will display a piano keyboard on the screen and allow you to play simple tunes on the keyboard. The graphics could lend themselves to use in a much more complex program if you have the time and energy to put into it.



```
10 REM ***By W.Szapirko ***
20 ' This program turns the computer
30 ' into a COMPUTER PIANO
40 ' To play on it you use the keyboard
50 ' that is drawn on the screen
60 ' To change sounds you have to press
70 ' space and type in commands (music
80 ' micro language)and enter it
90 ' some of the commands are
100 ' L64M300S8
110 ' L64M500
120 ' To find some more use the manual
130 '
140 '
150  DIM RE$(99)
160 STRIG(0) ON
170 CLICK OFF
180  COLOR 1,4,15
190 SCREEN 1,2
200 LINE(16,24)-(239,90),15,BF
210 LINE(47,96)-(207,162),15,BF
220 FOR T=23 TO 215 STEP 16
230 D=D+1
240 IF D=3 OR D=7 OR D=10 THEN 260
250 LINE(T+3,24)-(T+13,66),1,BF
260 LINE (T+8,24)-(T+8,90),1
270 NEXT T
280 D=0
290 FOR T=58 TO 189 STEP 16
300 D=D+1:IF D=3 OR D=7 OR D=10 THEN 320
310 LINE (T,96)-(T+10,138),1,BF
320 LINE (T+5,96)-(T+5,162),1
330 NEXT T
340 FOR T=21 TO 231 STEP 16
350 READ F$
360 DATA 7,Q,W,E,R,T,Y,U,I,O,P,[,],\
370 LOCATE T,78:PRINT F$
380 NEXT T
390 COLOR 15
400 FOR T=29 TO 235 STEP 16
410 READ F$
420 IF F$="3" OR F$="7" OR F$="0" THEN 450
430 DATA 1,2,3,4,5,6,7,8,9,0,-,=,←
440 LOCATE T,56:PRINT F$
450 NEXT T
460 FOR T=61 TO 200 STEP 16
470 READ F$
480 IF F$="F" OR F$="K"  THEN 500
490 LOCATE T,128:PRINT F$
500 NEXT T
510 DATA S,D,F,G,H,J,K,L,":"
520 COLOR 1
530 FOR T=53 TO 212 STEP 16
540 READ F$
550 LOCATE T,150:PRINT F$
560 DATA Z,X,C,V,B,N,M,",",.,/
```

```
570 NEXT T
580 LOCATE 23,7:PRINT "COMPUTER PIANO
590 COLOR 15
600 ON STRIG GOSUB 750
610 DIM M$(200)
620 FOR T=1 TO 41
630 READ F,D$
640 M$(F)=D$
650 NEXT T
660 DATA 9,03C,81,03D,87,03E,69,03F,82,03G,84,03A,89,03B,85,04C,73,04D,79,04E,80
,04F,91,04G,93,04A,92,04B
670 DATA 49,03C#,50,03D#,52,03F#,53,03G#,54,03A#,56,04C#,57,04D#,45,04F#,61,04G#
,8,04A#
680 DATA 83,05C#,68,05D#,71,05F#,72,05G#,74,05A#,76,06C#,58,06D#
690 DATA 90,05C,88,05D,67,05E,86,05F,66,05G,78,05A,77,05B,44,06C,46,06D,47,06E
700 I$=INPUT$(1):IF I$="" THEN 700
710 I=ASC(I$):PL$=H$+M$(I)+"T255"
720 PLAY"M9095T255S8"+PL$
730 IF RE=1 THEN RE$(R)=RE$(R)+M$(I)
740 GOTO 700
750 H$="":CLICK ON
760 COLOR ,,1
770 LINE (56,175)-(250,190),4,BF
780 LOCATE 56,175:PRINT ">";
790 FOR T=1 TO 20
800 B$=INPUT$(1):IF B$=CHR$(13)THEN 830
810 H$=H$+B$:PRINT B$;
820 NEXT T
830 CLICK OFF:COLOR,,15:RETURN
```

PUT SPRITE Command

by S. Colgan

Another incomplete documented function:

The PUT SPRITE command has a STEP option.
It moves sprite from the current graphics cursor position.
What I mean is:

```
PUT SPRITE 1,STEP(1,1),12,1
PUT SPRITE,SPRITE PLANE,STEP(X INCREMENT,Y INCREMENT),COLOR,SPRITE
```

Each time this line is excecuted the sprite will move across one position left and one position down.

* If given negative values it moves up and left
* If given zero values it stays still

In this command the increments refer to the current graphics cursor position. I hope you can make some sense of this (it does work) and it's alot faster than PUT SPRITE 1,(128,96),12,1 etc.

SPECTRAVIDEO™
AUSTRALASIAN USERS GROUP

ADHEX a program review

by J. van Staveren

Adhex is a utility whose function is to dump, find and substitute data in memory. Adhex is initiated by typing ADHEX and may optionally be followed by a filename e.g.

```
ADHEX
ADHEX MYPROG.COM
```

are both valid.

Adhex is self relocating to reside under the CCP (Console Command Processor), if a filename was given it will then load this into the TPA (Transient Program Area) at 100H, should the file be longer than available memory it will stop loading just short of the location occupied by Adhex, the remainder may be loaded restarting at 100H using the L command. Adhex will display a MENU and prompt for input allowing the usual CP/M line editing functions.

Control functions are -	^S	Stop/Start display
	^R	Restart menu
	^C	Reboot system

The commands are -

Dstart	end	Dump in Hexadecimal and ASCII
Fstart	end	Find ASCII, prompts for an ASCII string, it will search memory within the limits given for and display all occurences of that string.
Hstart	end	Same as Find but in hexadecimal.
X		Toggles menu off/on.
L		Continue loading file restarting at 100H.
Sstart		Substitute, displays the address and its contents, at this point there is a choice of 5 actions:-

1) a carriage return will advance to the next location.
2) a minus (-) will back up to the last location.
3) typing a hexadecimal value will replace the present value and advance to the next location.
4) a double quote followed by a string of ASCII characters will replace the values starting at the present location.
5) a full stop (.) will return you to the menu.

There are 2 commands not mentioned in the menu, they are:-

G	Executes the program at 100H.
Estart	Executes the program at start.

The G comand will not work if the program loaded didn't have a ,COM extension, however the E command will. Adhex will intercept a warm start to regain control. The only danger of this command is that if the program ovarlays Adhex there is nowhere to go on a warm start, a re-set is then the only solution. However it hasn't happened to me yet using Vers. 2.1.