SVI & MSX

— SPECTRAVIDEO —

NEWSLETTER

REGISTERED BY AUSTRALIA POST PUBLICATION No. TBH 0917 CATEGORY "B"

ISSUE NO.	ANNUAL SUBSCRIPTION	DATE
3 - 3	AUSTRALIA ..... $20.00 OVERSEAS ..... $25.00 OVERSEAS AIRMAIL ... $30.00	DEC - 1985

CONTENTS

NEWSLETTER CORRESPONDENCE

S.A.U.G.,
P.O. BOX 191,
LAUNCESTON SOUTH,
TASMANIA, 7249.

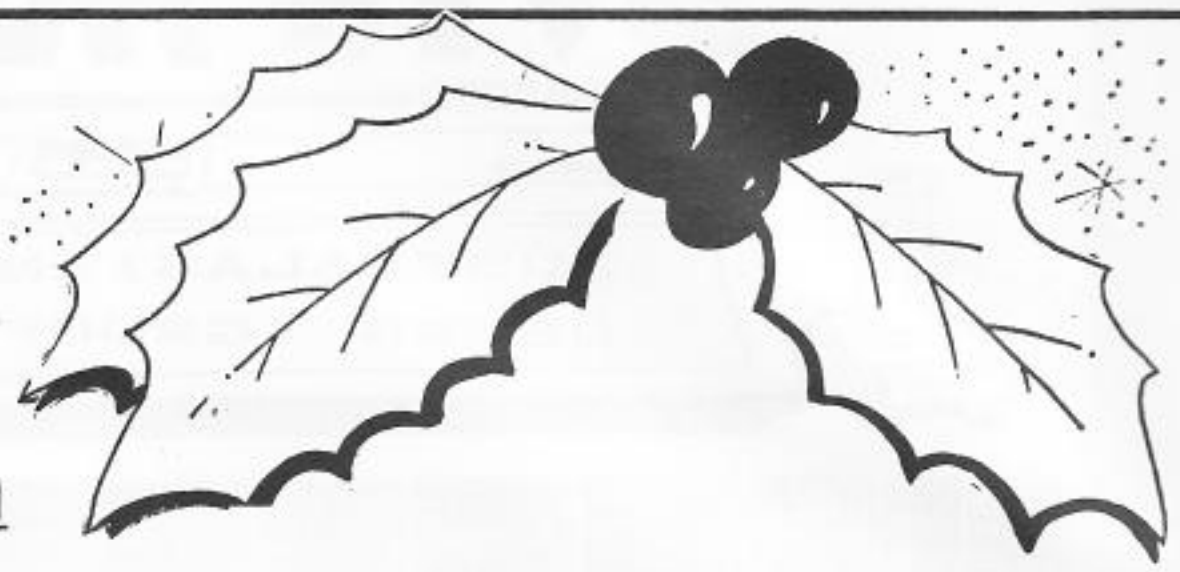
(003) 442493

LIBRARY CORRESPONDENCE

S.A.U.G. LIBRARY,
1 CONRAD AVENUE,
GEORGE TOWN,
TASMANIA, 7253.

(003) 822919

SVI & MSX SPECTRAVIDEO AUSTRALASIAN USERS GROUP



INTRODUCTION

Well Christmas is again with us. But I hope some of our members are enjoying it just a little bit more. We are pleased to announce the winners of the Great S.A.U.G. Software Competition.

1st Prize the SVI-728 M.S.X. Computer was won by Jon May for a great program DIAMOND MINE.

2nd Prize the SVI-318 Pack was won by Malcom Perrett for another great program COMPUTER VOLLEY BALL.

These two winning programs are featured in this months newsletter.

Minor prizes were also won by the following members.

Stereo Headphone Radio was won by G. Trevathan for his program MASTER MIND. This program will be printed next month.

2 Box of Datalife Disks was won by B. Goodman for his program DISK STARTUP. Again this program will be printed next month.

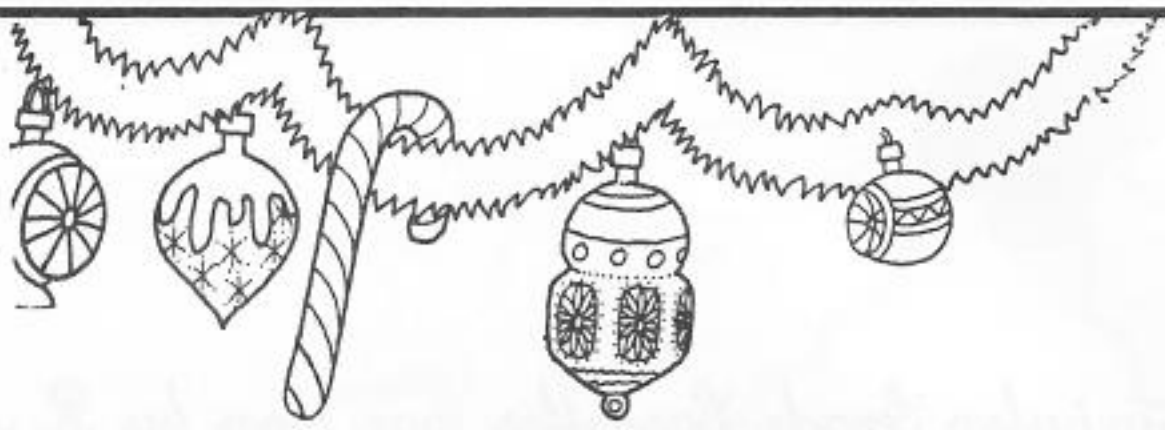
Box of Computer Grade Cassettes was won by R. Moore for his program JOUST. A note to Mrs. Moore, your Poem will appear in a newsletter soon also.

To all the other entrants many thanks for all your efforts and remember with a bit of luck your efforts may be printed in the newsletter in the comming months.

A very merry Christmas to you all and a great New Year.

See You all Next Year.

SVI & MSX
SPECTRAVIDEO
AUSTRALASIAN USERS GROUP



FIRST PRIZE WINNER

DIAMOND MINE

By. Jon May.

GAME DESCRIPTION

Guide "Willie the Rap-Dancer" around the haunted mines picking up as many diamonds as he can without being caught by the ghosts. You can only progress onto the next screen when all the diamonds have been collected.

Willie can move about platforms and ladders by using the arrow keys or the joystick in port 1. He can also move up and down on the green elevators by pressing the spacebar or trigger and pushing the joystick in the direction required. The ghost's movement is not restricted by anything. They can pass through the walls and floors.

Willie starts the game with three lives. The game is over when he has no lives left. He can, lose one life when :-

He is caught by a ghost,
He falls off a platform,
He is caught between two lifts when they move,
He rides a lift off the top of the screen,
He rides a lift off the bottom of the screen.

Each diamond is worth 50 points. There are ten completely different screens. Completing a screen is worth 200 points, except for the tenth, which is worth 1000 points.

To start the game from screen 1, press the spacebar. (You can also use the "cheat" facility to start from any screen by pressing that Number.)

IT IS POSSIBLE TO COMPLETE THE WHOLE GAME WITHOUT
LOSING A LIFE.

DIAMOND MINE

by : Jon May

This Program may be entered using the 'INPUT' program from Newsletter 2 - 2 (NOV. 84.) or The Year Book.

```
CJ      10 CLEAR1000:COLOR5,1,1:SCREEN1,2:SOUND3,10:COLOR7:LOCATE40,40:PRIN
           T"D I A M O N D":LOCATE41,40:PRINT"D I A M O N D":LOCATE40,41:PR
           INT"D I A M O N D":LOCATE41,41:PRINT"D I A M O N D"
DM      20 LOCATE52,80:PRINT"M I N E R":LOCATE53,80:PRINT"M I N E R":LOCATE
           53,81:PRINT"M I N E R":LOCATE52,81:PRINT"M I N E R"
FG      30 DRAW"c15s4bm80,128f16m80,168m64,144e16g16r32l8m80,128m72,144m80,
           168m88,144bm96,136e8bm88,128m94,118bm80,124u10bm72,128m66,118bm6
           4,136h8
MN      40 PLAY"v10t100l9o4ccco3b-b-a-a-b-b-ffb-b-a-a-b-b-o4ccco3b-b-a-a-b-b-
           o4ccco3b-b-o4cc","v12t100l9o5ccggfe-do4b-o5cde-dco4a-a-b-o5ccggfe
           -do4b-o5cde-dcc"
BL      50 ONERRORGOTO930
KB      60 DATA3,7,15,138,218,95,252,124,60,30,31,31,15,15,7,2,0,128,192,19
           6,204,252,254,254,254,240,248,252,248,240,192,0
LK      70 DATA3,7,15,10,154,223,92,252,127,63,31,31,15,15,15,12,0,128,196,
           204,220,254,240,224,224,240,240,248,248,224,128,0
GN      80 DATA0,0,0,12,30,6,30,0,13,125,122,6,12,11,27,57,0,0,0,12,30,6,30
           ,0,22,87,217,156,12,22,115,38
IB      90 DATA0,0,0,48,120,96,120,0,56,190,174,48,24,104,108,78,0,0,0,48,1
           20,96,120,0,104,218,187,57,48,104,206,100
BJ     100 DATA0,0,0,24,61,61,3,63,126,252,188,128,60,60,48,48,0,0,0,24,188
           ,188,192,252,126,127,61,1,60,60,12,12
BI     110 DATA"3513003153200200200243346643341440000445313000031326450054622
           2200000000240550055041111111111110914
AC     120 DATA"3531110000414501613333356100447770000000747115005341400006570
           0003516002000200015411140000000829
AF     130 DATA"5305606056046006000060006006000006310000005045006006006060000
           0060000600000310005660045600000719
BC     140 DATA"0000003530005060717000000020200060645750000000020006060517600
           0000000200606060070000050504007 28
AE     150 DATA"6666613113000006751700500073570505002754500050775300000024570
           000006753466666075701010114541139 9
AO     160 DATA"6151151000061151000000351300000020540005002611000000415160000
           0011111600051601516005106115001165
AE     170 DATA"5000600005006000606060606060606000060000006000006060006006060
           0060060000060060006060500006000050415
CG     180 DATA"3511115305430000340004300340000043340005511771510000344300000
           3400430053400004300451115146009 49
AO     190 DATA"3151015113200000000241531135140002002000351400451320000000024
           1111331114000022000015514415511053
BE     200 DATA"5011311315000020020006002007150000200200600020041500002000000
           6002000000000020000060004150000519","END"
FH     210 GOSUB220:G$(1)=S$:GOSUB220:G$(2)=S$:FORB=1TO6:GOSUB230:M$(B)=S$:
           NEXTB:GOTO240
BG     220 S$="":FORT=1TO32:READA:S$=S$+CHR$(A):NEXTT:RETURN
BL     230 S$="":FORT=1TO16:READA:S$=S$+CHR$(A):NEXTT:RETURN
KI     240 T=1:SF=1:LF=3:RESTORE110:DI$="":FORT=16TO1STEP-1:DI$=DI$+MID$(M$
           (1),T,1):NEXT
DF     250 COLOR1:LOCATE220,10:PRINT"■■":LINE(40,90)-(140,110),1,BF:LOCATE6
           3,94:COLORJM:PRINT"GAME OVER":LOCATE63,95:PRINT"GAME OVER":SPRIT
           E$(5)=M$(1):PUTSPRITE5,(215,5),6,5:COLOR15:LOCATE220,10:PRINT" 3
           "
```

```
OL    260 LOCATE200,40:PRINT"SCORE:":LOCATE200,41:PRINT"SCORE":LOCATE200,8
          0:PRINT"HIGH:":LOCATE200,81:LOCATE200,120:PRINT"SCREEN:":LOCATE2
          00,121:PRINT"SCREEN:":LOCATE200,81:PRINT"HIGH:":COLOR1:LOCATE200
          ,90:PRINT"■■■■■■■":LOCATE200,50:PRINT"■■■■■■":LOCATE200,130

KK    270 PRINT"■■■■■":IFSC>HSTHENHS=SC
DB    280 SC=0:COLOR15:LOCATE200,50:PRINTSC:LOCATE200,90:PRINTHS
CB    290 LOCATE200,150:COLOR5:PRINT"by JON":LOCATE224,160:PRINT"MAY":LINE
          (186,2)-(248,186),7,B:LINE(188,4)-(246,184),7,B:COLOR15:LOCATE20
          0,130:PRINT" 0":JM=15
EO    300 IFINKEY$<>""THENGOTO300ELSEMR$=""
AC    310 O=0:IFSTRIG(1)+STRIG(0)=-1THENREADA$:SF=1:GOTO330
IA    320 I$=INKEY$:IFI$><""THENRESTORE110:FORT=1TOVAL(I$):READA$:NEXTT:SF
          =VAL(I$):ELSEGOTO310
ON    330 PLAY"O4V8L5CCGGF+F+A-G":DK=0:LINE(0,0)-(180,190),1,BF:FL=1:FORO=
          1TO9:FORP=1TO10:Q=VAL(MID$(A$,FL,1)):FL=FL+1:B=O*20-20:A=P*16
AN    340 IFQ=1THENLINE(A,B+17)-(A+15,B+19),9,BF:GOTO410
CK    350 IFQ=2THENLINE(A,B)-(A+7,B+20),4,BF:PSET(A+1,B+1),1:FORT=1TO9:DRA
          W"C1S4R5bD2L5":NEXTT:GOTO410
DP    360 IFQ=3THENLINE(A,B+17)-(A+7,B+19),4,BF:LINE(A+8,B+17)-(A+15,B+19)
          ,9,BF:GOTO410
CE    370 IFQ=4THENLINE(A,B)-(A+7,B+16),4,BF:PSET(A+1,B+1),1:FORT=1TO9:DRA
          W"C1S4R5BD2L5":NEXTT:LINE(A,B+17)-(A+15,B+19),9,BF
GA    380 IFQ=5THENFORFD=1TOLEN(MR$)STEP2:IFVAL(MID$(MR$,FD,1))=P-1ANDVAL(
          MID$(MR$,FD+1,1))=OTHENQ=1:GOTO340ELSENEXTFD:LINE(A,B+17)-(A+15,
          B+19),9,BF:PSET(A+7,B+16),1:DRAW"c15s4e4h4g4f4":PAINT(A+7,B+14),
          15:GOTO410
CJ    390 IFQ=6THENLINE(A,B+17)-(A+15,B+19),12,BF:GOTO410
DL    400 IFQ=7THENLINE(A,B)-(A+7,B+19),4,BF:PSET(A+1,B+1),1:FORT=1TO10:DR
          AW"C1S4R5BD2L5":NEXTT:LINE(A+8,B+17)-(A+15,B+19),9,BF:GOTO410
BP    410 NEXTP,O:ND=VAL(MID$(A$,91,2)):IFPO=1THENND=JM
AJ    420 X=VAL(MID$(A$,93,1)):Y=VAL(RIGHT$(A$,1)):X=X*16:Y=Y*20-20
AC    430 PO=0:D=1:SPRITE$(1)=M$(1):SOUND1,200:SOUND7,&B11010O:SOUND11,40:
          SOUND8,16:SOUND10,8
KD    440 COLOR1:LOCATE200,130:PRINT"■■■":COLOR15:LOCATE200,130:PRINTSF
FI    450 J=161-X:M=190-Y:DJ=SGN(X-J):DM=0
EH    460 PUTSPRITE2,(J,M),15,2:PUTSPRITE3,(J,M-38),15,2
BC    470 ONSPRITEGOSUB740:SPRITEON
IE    480 PUTSPRITE1,(X,Y),15,1:SOUND3,0:
AL    490 I=STICK(0)+STICK(1)
BI    500 IFSTRIG(0)+STRIG(1)=-1THENGOTO870
CK    510 IFI=0THENGOTO610
DO    520 IFPOINT(X,Y+15)=4THENGOTO550
CM    530 IFI=3THENLD=3:X=X+4:GOTO590
CN    540 IFI=7THENLD=1:X=X-4:GOTO590
BB    550 IFI=3ANDPOINT(X+15,Y+18)>1THENLD=3:X=X+4:GOTO600
KA    560 IFI=7ANDPOINT(X-4,Y+18)>1THENLD=1:X=X-4:GOTO600
FL    570 IFI=5ANDX/8=INT(X/8)ANDPOINT(X,Y+17)=4THENY=Y+4:LD=5:GOTO600
FF    580 IFI=1ANDX/8=INT(X/8)ANDPOINT(X,Y+16)=4THENY=Y-4:LD=5:GOTO600
DG    590 DL=DL+.5:IFDL=1THENDL=0:GOTO600ELSEGOTO610
AI    600 SOUND8,12:IFD=LDTHEND=LD+1ELSED=LD
EF    610 SPRITE$(1)=M$(D):PUTSPRITE1,(X,Y),14,1:SOUND8,0
GL    620 SPRITE$(1)=M$(D):PUTSPRITE1,(X,Y),14,1
DO    630 IFPOINT(X+4,Y+18)=1THENGOTO740
HM    640 IFPOINT(X+4,Y+10)=15THENGOTO810
EC    650 IFX<J+4ANDX>J-6THENGOTO720
EM    660 IFY<M+4ANDY>M-6THENGOTO730
```

```
AH    670 J=J+DJ:M=M+DM:IFJ>180ORJ<0ORM<0ORM>190THENDJ=-DJ:DM=-DM:J=J+3*DJ
          :M=M+3*DM:GOTO670
AJ    680 KL=KL+.2:IFKL>2.9THENKL=1
DA    690 K=INT(KL):SPRITE$(2)=G$(KL)
DJ    700 PUTSPRITE2,(J,M),15,2:PUTSPRITE3,(190-J,M-38),15,2
AO    710 GOTO490
HG    720 DJ=0:DM=2.4*SGN(Y-M):GOTO670
GN    730 DM=0:DJ=2.4*SGN(X-J):GOTO670
BE    740 SPRITEOFF:SPRITE$(1)=DI$:SOUND7,&B111100
LO    750 Y=Y+2:IFPOINT(X+4,Y+13)>1THENDI=INT(RND(1)*3)-1:GOSUB770
HK    760 PUTSPRITE1,(X,Y),15,1:IFY>190THENGOTO780ELSEGOTO750
DA    770 PLAY"","V10O2S1M3000C":FORQ=-1TO-4STEP-1:Y=Y+Q:X=X+DI:PUTSPRITE1
          ,(X,Y),15,1:FORTT=1TO6:NEXTTT,Q:FORQ=1TO5:Y=Y+Q:X=X+DI:PUTSPRITE
          1,(X,Y),15,1:FORTT=1TO6:NEXTTT,Q:RETURN
AD    780 LF=LF-1:IFLF=0THENJM=7:GOTO240
CB    790 LOCATE220,10:COLOR1:PRINT"■■":LOCATE220,10:COLOR15:PRINTLF
BJ    800 IFDK=1THENJM=ND:PO=1:GOTO330ELSEGOTO420
CC    810 LINE(X-2,Y)-(X+10,Y+16),1,BF:ND=ND-1:SC=SC+50:MR$=MR$+RIGHT$(STR
          $(INT(X/16)-1),1)+RIGHT$(STR$(Y/20+1),1):GOSUB920
CM    820 SOUND9,12:SOUND7,&B110100:SOUND3,0:FORH=100TO1STEP-3:FORG=HTOH+2
          :SOUND2,G:NEXTG,H:SOUND9,0
BI    830 IFND=0THENGOTO850
AC    840 GOTO500
CI    850 SC=SC+200:GOSUB920
DC    860 MR$="":SF=SF+1:READA$:IFA$="END"THENSC=SC+800:GOSUB920:RESTORE11
          0:READA$:GOTO330ELSEGOTO330
AC    870 IFPOINT(X,Y+17)=12ANDPOINT(X+7,Y+17)=12THENGOTO880ELSEGOTO650
DD    880 IFI=1THENW=-4:B=-2:V=-5:ELSEIFI=5THENW=4:B=2:V=5ELSEGOTO650
HB    890 SPRITE$(1)=M$(1):LL=150:DK=1:A=INT(X/16)*16:
AM    900 IFPOINT(A+7,18+Y+W)=1ANDY>-3THENY=Y+W:SOUND9,10:SOUND2,Y*1.5:LIN
          E(A,Y+17)-(A+15,Y+19),12,BF:LINE(A,Y+(18-V))-(A+15,Y+(18-B)),1,B
          F:PUTSPRITE1,(X,Y),15,1:SOUND9,0
AK    910 GOTO650
JH    920 LOCATE200,50:COLOR1:PRINT"■■■■■":COLOR15:LOCATE200,50:COLOR15:PR
          INTSC:RETURN
AJ    930 Y=Y+10:RESUME740
END
```

UNDERSTANDING CP/M Pt-4

By S.W. McNamee

Well folks at long last here is the last installment in my series UNDERSTANDING CP/M. As I said in the first article this chapter will be concerned with the BIOS and its' peculiarities as installed on the SV-328.

First some generalities. All BIOS'S (BIOSES?, BIOI??, ?????) begin with a table of jumps with exactly 17 vectors in it. A vector by the way is just an address to jump to. These vectors point to routines in the BIOS that do all the low level I/O for the particular machine that it was written for and must be customised for that machine. I will not explain in detail what each routine does as this is well documented in Section 6.6 of the CP/M Operating System Manual. What I will do is take each jump in turn and discuss any special parts of it which the Spectravideo does a little differently to other machines. In 2.2 and 2.22 the BIOS jump table starts at E600H and is 51 bytes long (17 vectors) and appears thus for 2.22:

```
BOOT:     JP   EEEC
WBOOT:    JP   EF4C
CONST:    JP   E7A4
CONIN:    JP   E7AF
CONOUT:   JP   E7BA
LIST:     JP   E7D0
PUNCH:    JP   E7DB
READER:   JP   E7E6
HOME:     JP   EC78
SELDSK:   JP   EC5C
SETTRK:   JP   EC7A
SETSEC:   JP   EC7F
SETDMA:   JP   EC8E
READ:     JP   EC93
WRITE:    JP   ECD0
LISTST:   JP   E7C5
SECTRAN:  JP   EC58
```

The vectors are different for 2.2 due to its' shorter length but the principle is the same.

BOOT:

This is the routine that is jumped to immediately after the cold boot loader. I guess a brief word about the cold boot loader is in order. Most CP/M machines only have a very small ROM which is only responsible for reading in the first sector of the first track of a master disk in the A: drive. This sector contains code which switches out the ROM and then loads in the rest of the BIOS. The Spectravideo has a small routine in its' BASIC ROM that does this for us. If it finds that there is no disk in the drive (or no drive!) it jumps to BASIC.

Once the cold boot loader has loaded in the whole BIOS it jumps to the first vector in the jump table ... the BOOT routine. This section of code does things such as initialise any buffer areas, programme peripheral hardware (Such as RS-232 card or 80 Column card etc.) and print the sign on message. In the Spectravideo, calls are also made to the BASIC ROM to initialise the keyboard interrupt and the drivers for the 40 column display. The Spectravideo cold boot routine jumps to the warm boot routine when it is finished and it is this code that finishes loading in the rest of the system.

WBOOT:

This code is executed any time a Warm Boot is done. (Either by an executing program or by ^C from the keyboard.) It uses special sections of the BIOS to reload the BDOS and CPR and re-initialise the vectors at 0000H and 0005H. After it is finished the warm boot jumps to the CPR which then initialises the drive system, logs in the A: drive and prompts the user with the familiar 'A>'. A vector for this routine is stored at 0000H so that a warm boot will be done any time a program jumps to 0000H.

CONST:

This returns the status of the keyboard in A. The keyboard status routine in BASIC ROM at 3BH is actually called for this function. Since the BASIC ROM scans the keyboard on an interrupt driven basis the BIOS must keep this interrupt sequence going. Thus the BIOS switches in the BASIC ROM on every interrupt and calls its' interrupt handler.

CONIN:

Returns the next character from the keyboard buffer in A. The keyboard input routine in BASIC ROM at 3EH is actually called.

CONOUT:

Takes a byte in C and outputs it to the CONsole device. If the 80 column card is not being used this function calls the screen out vector in BASIC ROM at 18H.

LIST:

Takes a byte in C and outputs it to the LiST device.

PUNCH:

Takes a byte in C and outputs it to the PUNch device.

READER:

Returns the next byte from the ReaDeR device in A.

LISTST:

Returns the status of the LiST device in A.

The above routines share some common code in the Spectravideo versions and all implement the I/O byte concept. For a more involved discussion of this see page 138-139 of the CP/M manual. All these routines are of the form:

```
call    iobtst
dw      vect1
dw      vect2
dw      vect3
dw      vect4
```

The routine 'iobtst' tests the I/O byte and selects one of the four vectors following the call to it depending on the setting of the particular 2 bits in the I/O byte. It then branches to that vector.

HOME:
Sets the track counter to 0

SELDSK:
Takes a disk No. in C and selcts that disk. Returns the address of the DPH in HL.

SETTRK:
Sets the track counter to the value in C.

SETSEC:
Sets the sector counter to the value in C. Note that logical 128 byte sectors are used.

SETDMA:
Takes the address for the next 128 byte disk transfer in BC and sets the buffer.

READ:
Reads the 128 byte block from the previously selected disk, track and sector into the selected address.

WRITE:
Writes the 128 byte block from the selected address to the selected disk, track and sector.

SECTRAN:
Takes a sector number in BC (starting at 0) and returns a Physical sector number in HL. The spectravideo has all sector skewing physically placed on the disk at format time so no sector translation is needed. This routine simply increments the sector number by one and returns it in HL.

As well as these standard vectors the Spectravideo BIOS has a second jump table at E651H. These routines perform some extra duties for the system utilities and can be used by programers to their advantage.

E651 - Takes an address in HL and calls the BASIC ROM ... VERY useful.

E654 - This routine reads a block of PHYSICAL sectors from disk to a buffer. The call to this routine should be followed by a series of six bytes. It will check these bytes, read the selected block from disk and then return to the instruction immediately after the six bytes. In an assembly language program a call to this routine would look like this:

```
rdblck    equ  0e654h
          call rdblck
          dw   buffer
          db   sector
          db   track
          db   drive
          db   count
```

next:

'buffer' is the address to read the block to.
'sector' is the physical sector number to start reading from.
'track' is the track number
'drive' is the drive number (0 = drive A:).
'count' is the number of physical sectors to read. When the routine is finished it will return to the address labled 'next:'.
Note that track 0 is single density and has 18 128 byte sectors and all other tracks are double density and have 17 256 byte sectors. The routine allows for this and transfers an exact number of sectors whether they are single or double density. You should keep this in mind if you are reading from track 0 (as when accessing the system tracks).

E657 - This routine is exactly the same as the previous one except this one WRITES a block of memory to disk.

E65A - Calling this vector resets the disk timer.That is it keeps the drive motor on. It must be called every 10 seconds or so if you want the drive motor to stay on. Note that this routine is automatically called every time a sector is read from or written to disk. You only need to use this routine if you want the drive to stay on but do not want to access it for a while.

I will now talk a little bit about the 80 column card. This card uses a direct memory video RAM from F000H to F77FH, controlled by a 6545 CRT controller IC. The top left hand character is at F000 while the bottom right hand character is at F77F. The video RAM does not take up main memory however because it is bank switched. It is switched in when needed and then switched back out again. To switch in VRAM send 0FFH to port 58H and to switch it back out send 0 to the same port. The character set used is exactly the same as the colour display in BASIC. Ei The ASCII characters go from 0 - 5FH, the inverse ASCII is from 60H - BFH and the graphics characters start at C0H. The last characters in the set are the now familiar letters 'J.Suzuki'.

Three points are very important.

1.ALWAYS disable interrupts before switching in VRAM. You can enable them when you switch VRAM off. This is because the interrupt routine has a stack in VRAMs' address space, and also calls the BASIC ROM which also has stacks and vectors in the same address space.

2.NEVER switch in VRAM when your program is running in the area F000 - F800, and NEVER allow a program to CALL this area or JP to it.

3.NEVER locate your stack in this area when switching in VRAM.

If you ignore these rules your program will start running quite happily in VRAM with completely unpredictable results.

One more point concerning the 80 column card can be quite useful. There is a table in the BIOS used at cold boot to initialise the registers of the 6556 CRT controller. It is 16 bytes long, 1 byte for each of the first 16 registers which are the only ones used in the Spectravideo circuit configuration. In version 2.2 this table is at EB54H while in 2.22 it is at EC11H. Changing these values can allow you to customise the format of the 80 column display to a certain degree. For example although only 1920 bytes of VRAM are used there are actually 2048 bytes available. Changing the CRT register contents can allow you to display these hidden bytes. If you would like DATA sheets on the CRT controller chip send me a SAE and I will send you a copy.

My Address is:

S.W.McNamee
5/15 Stuckey Rd.
Clayfield 4011.

The last topic I will deal with is a way of modifying the CP/M system tracks of a disk. My preferred method is to use SYSGEN. Run SYSGEN as normal and when it asks for a destination drive press <ENTER>. A system disk will then be requested and you can reboot. Immediately after rebooting SAVE 51 CPM.COM (or some other appropriate file name. You will then have on disk a copy of the SYSGEN utility complete with a memory image of the operating system track. You can then run DDT and read in this file and modify it. When you have finished the modification just type G100 (from the DDT prompt) and the modified utility will start running. When it asks for the source drive hit <ENTER> to skip the read stage and then procede as normal specifying a destination drive. When finished a copy of the modified system tracks will be placed on the selected drive. The addresses of the various parts of the system in the memory image of the saved file are as follows:

BOOT LOADER	(128 bytes)	900H
CCP	(2048 bytes)	980H
BDOS	(3584 bytes)	1180H
BIOS	(2688 bytes)	1F80H

Although the cold boot loader only loads in 2688 bytes of BIOS there is actually room on the disk for 6528 bytes. To load in these sectors the cold boot loader sector needs to be changed and this is left as an exercise for the reader. All addresses in the BIOS are offset by C680H in the SYSGEN image. For example say you had version 2.22 and you wanted to modify the CRT register table at EC11. You would find this table at 2591H in the sysgen image and this is where you would make the change(2591H = EC11 - C680).

As noted earlier on the BIOS calls the BASIC ROM to perform some of its' functions. This means that you have to be careful in touching any RAM above F4FFH as this is BASICs' scratch pad area. However since CP/M only uses a small part of the BASIC ROM there are whole chunks of memory that are never touched and can be used as buffers etc. Their addresses and length follow:

F550 - F78F	575 bytes
F810 - F86F	95 bytes
F880 - F8DF	95 bytes
F900 - F97F	128 bytes
FB10 - FCDF	463 bytes

These are areas of RAM that I know to be safe from BASIC. There may be others but they will all be fairly small and not very useful. All RAM below F500 is safe from BASIC.

Well that just about wraps up this series of articles on CP/M. I hope you have enjoyed them and perhaps have gleaned a little bit of useful information from them. If any body would like to see some other topics discussed please write to the editor. If enough people show an interest in a particular subject I will endeavour to put 'fingers to keyboard' and churn out some words of wisdom???

P.S. The library has a file available which is a complete dissassemly of the Spectravideo BIOS. It can be modified and reassembled if desired, but is only partially documented. Both versions 2.2 and 2.22 are available.

SECOND PRIZE WINNER

COMPUTER VOLLEY BALL

By. Malcom A. Perrett.

THE GAME

The aim of the game is to be the first to score 10 points. This is done by getting the ball to bounce off the floor of your opponents side, for which you get one point and also get to serve the ball. If you hit the ball five times before it passes over the net your opponent is awarded one point and gets to serve the ball. The ball may bounce off any wall, the net or the ceiling but not of course the floor. Any Joystick can be used by following the instructions given at the beginning of the game.

You can manouver the ball by, when hitting the ball, moving the bat in the directions you want the ball to go but remember the ball has inertia which may cause it not to go exactly where you wanted it to go. You can play against another human or you can play against the computer under the alias of Specky Travideo.

THE PROGRAM

In this listing I have left most of the lines as they were at the time of writing in the hope of making it easier to understand. You may leave all of the REM's out when entering the program as there are no direct jumps to those lines.

The subroutine at line 118 is only used if the one player option is selected. It uses the speed, movement, direction and position of the ball to position bat #2. Lines 302 and 129 move the bat by adding the value in the arrays to the co-ordinates of the bats. The ball hitting a bat is detected by the sprite collision interrupt which jumps to line 102. This routine first calculates which bat is involved and then adjusts the speed and direction of the ball according to the bat's angle, movement and the ball's movement.

(I renumbered before listing sorry. ED.)

COMPUTER VOLLEY BALL

by : M.A. Perrett

This Program may be entered using the 'INPUT' program from Newsletter 2 - 2 (NOV. 84.) or The Year Book.

```
AD     10 GOTO 1750
CJ     20 ' ********************************
CK     30 ' *                              *
DM     40 ' *   Computer Volley ball       *
CM     50 ' *   ---------------------      *
IP     60 ' * Written by : M.A.Perrett     *
AE     70 ' * Date       : 02/09/85        *
CE     80 ' * Bytes used : 8.1 K Bytes     *
EK     90 ' * Written in : SVI BASIC       *
AK    100 ' *                              *
AJ    110 ' ********************************
AI    120 ' ----* Variable Definitions *----
GK    130 ' ----- Bat #1
HM    140 ' B1 : angle of bat
FL    150 ' C1 : hit ball counter
JL    160 ' D1 : direction from j'stick
FK    170 ' J1 : j'stick used
AG    180 ' N1$: name of player
BF    190 ' S1 : score
FN    200 ' X1 : X co-ordinate of bat
EO    210 ' Y1 : Y co=ordinate of bat
GN    220 ' ----- Bat #2
HN    230 ' B2 : angle of bat
FM    240 ' C2 : hit ball counter
JM    250 ' D2 : direction from j'stick
FJ    260 ' J2 : j'stick used
AH    270 ' N2$: name of player
BE    280 ' S2 : score
GH    290 ' X2 : X co-ordinate of bat
EM    300 ' Y2 : Y co=ordinate of bat
BE    310 ' ----- Ball
HN    320 '  X : X co-ordinate of ball
HO    330 '  Y : Y co-ordinate of ball
DI    340 ' XA : ball acceleration
DI    350 ' YA : ball acceleration
AC    360 ' ---- Literal Variables
JJ    370 ' A(): ball acceleration from bat
CP    380 ' B(): bat angle
IL    390 ' D(): ball movement from bat
F6    400 'XD(): j'stick / sprite movement
F6    410 'YD(): j'stick / sprite movement
FL    420 ' ----- Other
JE    430 '  K : multi purpose  integer
JC    440 '  J : multi purpose  integer
FF    450 ' A$ : multi purpose  string
AL    460 ' N$ : input player name
CG    470 ' OP : game option
CK    480 ' CB : computer bat change
EH    490 ' -----* Game Subroutines  *-----
NA    500 ' ---- hit ball (sprite int. trap)
BP    510 SPRITE OFF:IF X>126 THEN 590
```

```
DM    520 IF X>118ANDXA>0 THEN GOTO 590
IO    530 '   ** bat #1
EO    540 IF B1 THEN XA=XA+D(B1)
PL    550 XA=XA+SGN(XD(D1)):YA=-ABS(YA)-A(D1):Y=Y1-28
HL    560 SOUND 4,80:C1=C1+1:C2=0:IF C1>4 THENGOSUB 1200:RETURN 930' score
AC    570 GOTO 630
IK    580 '   ** bat #2
FB    590 IF B2 THEN XA=XA+D(B2)
PE    600 XA=XA+SGN(XD(D2)):YA=-ABS(YA)-A(D2):Y=Y2-28
HJ    610 SOUND 4,20:C2=C2+1:C1=0:IF C2>4 THENGOSUB 1200: RETURN 930' scor
          e
DM    620 ' ** bounce sound & cont
DM    630 SOUND5,1:SOUND10,10:GOSUB 850
BL    640 FOR K=0TO4:NEXT
AG    650 SOUND10,0:SPRITE ON:HF=-1
CA    660 RETURN
IA    670 ' ----- computer move bat #2
BB    680 D2=1:IF X>115 THEN 710
DC    690 IF X2<170 THEN X2=X2+2 ELSE IF X2>174 THEN X2=X2-4
AC    700 FOR K=1TO50:NEXT:GOTO 760
ED    710 IF X2<X OR (XA>5ANDY<104) THEN IFX2<234 THEN X2=X2+4:GOTO 730
AJ    720 IF X2>X OR (XA<-4ANDY<92) THEN IFX2>139 THEN X2=X2-4
FN    730 IF XA<4 AND YA<10 AND Y2>114 THEN Y2=Y2-4:D2=1 ELSE IF YA>2 AND
          Y2<144THEN Y2=Y2+4:D2=5
GM    740 IF RND(1)>.8 THENCB=INT(RND(1)*5):IFCB>2 OR C2>1 THEN CB=2
DG    750 IF B2<>CB THEN B2=CB:SOUND2,20:GOSUB 1140
CB    760 RETURN
GM    770 ' ----- player #2 move bat #2
EI    780 D2=STICK(J2):X2=X2+XD(D2):Y2=Y2+YD(D2)
EE    790 IF X2<135 OR X2>236 THEN X2=X2-XD(D2)
BL    800 IF Y2<70 OR Y2>148 THEN Y2=Y2-YD(D2)
EB    810 IF STRIG(J2) THEN IF B2<>B(D2) THEN B2=B(D2):SOUND 2,20:GOSUB 11
          40
BA    820 ' ----- put sprites
BK    830 PUT SPRITE1,(X1,Y1),5,B1
BN    840 PUT SPRITE2,(X2,Y2),9,B2
FK    850 PUT SPRITE3,(X,Y),11
CC    860 RETURN
AC    870 ' ----- reset variables
FE    880 X1=16:Y1=130:B1=1:D1=0:C1=0
BF    890 X2=225:Y2=130:B2=2:BC=2:D2=0:C2=0
AL    900 X=16:Y=50:XA=0:YA=0
CI    910 RETURN
IC    920 ' ----- score
EL    930 SPRITE OFF:IF X<116 THEN 1000
CB    940 '   ** player #1
DG    950 S1=S1+1:GOSUB880:X=16:GOSUB830
FN    960 LINE(98,32)-(80,24),1,BF:LOCATE 80,24:COLOR5:PRINT USING":##";S1
AC    970 IF S1<10 THEN GOSUB 1070 ELSE 1940
DA    980 GOTO1030
CH    990 '   ** player #2
DD   1000 S2=S2+1:GOSUB880:X=225:GOSUB830
EA   1010 LINE(206,32)-(188,24),1,BF:LOCATE 188,24:COLOR9:PRINT USING":##"
          ;S2
DC   1020 IF S2<10 THEN GOSUB 1070 ELSE 1940
DB   1030 FOR K=0TO150:NEXT :SPRITE ON
AB   1040 GOTO1510
```

```
EH   1050 ' -----*  Music & Sound    *-----
AC   1060 ' ---- echo 1
FB   1070 A$="132o4egab"
EF   1080 PLAY"v10;xa$;"
EA   1090 PLAY"v7;xa$;"
EN   1100 PLAY"v4;xa$;"
EP   1110 PLAY"v1;xa$;"
DJ   1120 RETURN '----
BE   1130 ' ---- bat change sound
EJ   1140 SOUND 3,2:SOUND9,16:SOUND 11,0:SOUND 12,4:SOUND 13,4
DM   1150 RETURN '----
DL   1160 ' ---- ball bounce sound
EF   1170 SOUND 3,4:SOUND9,16:SOUND 11,0:SOUND 12,1:SOUND 13,4
DP   1180 RETURN '----
KB   1190 ' ---- 5 hits message & music
JI   1200 Y=209:GOSUB 830:PLAY "v10o418e4o3ao4dc4o3a4"
GI   1210 IF C1=5 THEN B=20 ELSEB=148
BB   1220 LINE (B,48)-(B+88,72),12,BF
PD   1230 PLAY "v10o418e4o3ao4dc4o3a4"
CE   1240 FOR K=0TO4:IF KAND1 THEN C=1ELSE C=15
IC   1250 LOCATE B,56:COLOR C:PRINT "   FIVE HITS"
HP   1260 FOR J=1TO350:NEXT J,K
EE   1270 LINE (B,48)-(B+88,72),1,BF
BH   1280 RETURN
AG   1290 ' ---- Scherzo  by C.M.von Weber
LG   1300 PLAY"t210o518v8edco4bag+","t210v7o318cr8cr8cr8","t210v6o318er8er
          8er8"
IG   1310 PLAY"abo5cde4","o318cr8cr8cr8","o318er8er8er8"
IK   1320 PLAY "v9o5f2v8o4g+4","dr8dr8dr8","fr8fr8fr8"
IJ   1330 PLAY "v9o5f2v8o4g+4","dr8dr8dr8","fr8fr8fr8"
PA   1340 PLAY"18o5edco4bag+","18cr8cr8cr8","o318er8er8er8"
LM   1350 PLAY"abo5cdef","o318cr8cr8cr8","o318er8er8er8"
DN   1360 PLAY "v9o5g2v8o4b4","dr8dr8dr8","fr8fr8fr8"
IN   1370 PLAY "v9o5c2r4","c4o2c4r4","e4r2"
BG   1380 RETURN
BH   1390 ' ** More Scherzo
PC   1400 PLAY "t210v718o4;gabo5cd4","t210v818o2gbo3d","t210"
EP   1410 PLAY "v8o4bo5cdef4","18v8o3gabo4cd4"
BP   1420 PLAY "v9o5defga4","v9bo3bo4cde4"
AB   1430 PLAY "v10o5fedco4b4","v10o4deff+g"
AF   1440 PLAY "o5co4bagf+g","18v9o2cr8o3cr8cr8","18v9o3r4er8er8"
ED   1450 PLAY "abo5cdef","18cr8cr8cr8","18er8er8er8"
IJ   1460 PLAY "g2o4b4","dr8dr8dr8","fr8fr8fr8"
ND   1470 PLAY "o5c2r4","cr8o2cr4","er2"
BF   1480 RETURN
DM   1490 ' -----*    The Game       *-----
GP   1500 ' ---- move bat #1
BM   1510 D1=STICK(J1):X1=X1+XD(D1):Y1=Y1+YD(D1)
GM   1520 IF X1<8 OR X1>104 THEN X1=X1-XD(D1)
BN   1530 IF Y1<70OR Y1>148 THEN Y1=Y1-YD(D1)
AF   1540 IF STRIG(J1) THEN IF B1<>B(D1) THENB1=B(D1):SOUND 2,80:GOSUB 114
          0
GL   1550 ' ---- move bat #2
AJ   1560 ON OP GOSUB 680,780
BK   1570 ' ---- move ball
EK   1580 YA=YA+1:Y=Y+YA:X=X+XA
AG   1590 IF YA>13 THEN YA=13
```

```
DG    1600 IF Y<93 THEN 1660 ELSE IF  X<115 OR X>128 THEN1660
CN    1610 '   ** hit net
FJ    1620 IF Y>94+YA THEN XA=-(XA-XA\2):IF XA<0THEN X=112:GOTO 1660 ELSE X
         =130:GOTO 1660
BI    1630 IF (XA>-1ANDX>118) OR (XA<1ANDX<123) THEN Y=94:YA=-1
EM    1640 IF XA=0 THEN XA=1
DD    1650 '  ** bounderies
FB    1660 IF X<10 THEN X=10:XA=-(XA-XA\2):GOSUB 1170:GOTO 1700
BC    1670 IF X>232THEN X=232:XA=-(XA-XA\2):GOSUB1170:GOTO 1700
BE    1680 IF Y<10 THEN Y=12:YA=0:GOTO 1700
EK    1690 ' ---- put sprites
AD    1700 GOSUB 830
EH    1710 IF Y>155 THEN 930 ' hit foor
AK    1720 GOTO 1510
DE    1730 ' -----*   Initialize      *-----
KM    1740 ' ---- get info from player(s)
CJ    1750 CLEAR 200:DEFINT A-Z:DIM XD(8),YD(8),A(8),S(8),D(2),S$(3)
DB    1760 SCREEN 0,0:WIDTH 39:COLOR 15,2,1:CLICK OFF
DH    1770 LOCATE4,2,0:PRINT"<**: COMPUTER VOLLEY BALL :**>":GOSUB 1300
AF    1780 LOCATE 0,8,1:PRINT "One player or two (1/2)? ";
DG    1790 OP=VAL(INPUT$(1)):IF OP<1OR OP>2THEN 1790
GE    1800 PRINT OP:K=1:GOSUB 2060:N1$=N$:J1=J
AP    1810 IF OP=1 THEN N2$="  Specky":J2=3:GOTO 1840
FD    1820 K=2:GOSUB 2060:N2$=N$:J2=J
EM    1830 ' ---- set up game
AM    1840 IF PLAY(0) THEN 1840
CN    1850 CLS: COLOR 15,13:LOCATE11,10,0:PRINT"GET READY TO PLAY"
IK    1860 LOCATE0,15:PRINT N1$ " will play on the << Left <<"
BA    1870 PRINT:PRINT N2$ " will play on the >> Right >>"
IF    1880 GOSUB 1400 ' more music
HP    1890 GOSUB 2430 ' def literal variables
CC    1900 GOSUB 2330 ' def sprites
CJ    1910 GOSUB 2160 ' setup court screen
FL    1920 GOSUB 880:GOSUB 830:SPRITE ON:GOTO 1510
CC    1930 ' ---- end of game
EL    1940 GOSUB 1070:SCREEN 0:COLOR 15,4
DH    1950 PRINT:PRINT N1$;USING ":##";S1:PRINT:PRINT N2$;USING":##";S2
BF    1960 LOCATE 0,10,0:PRINT " THE WINNER IS :";
AE    1970 GOSUB 1300:IF S1>S2 THEN A$=N1$ ELSEA$=N2$
KA    1980 A$="    "+A$+" ":FOR K=0TO10:LOCATE 16,K:PRINT A$:FOR J=0TO300:N
         EXT:LOCATE 20,K:PRINT"          ":NEXT
AM    1990 FORK=1TO12:IF MID$(A$,K,1)=" " THEN LOCATE16,10:PRINTRIGHT$(A$,1
         3-K):NEXTELSE GOSUB 1400
DO    2000 IF INKEY$<>"" OR PLAY(0) THEN 2000
IH    2010 LOCATE0,12,1:PRINT "To play another game press any key ";
BI    2020 A$=INPUT$(1)
FA    2030 RUN
FB    2040 ' -----* Init. Subroutines *----
EP    2050 ' ---- get name & stick(#)
HE    2060 IF INKEY$>"" THEN 2060
DD    2070 A$="Player"+STR$(K):PRINT:PRINT A$;" ";
BE    2080 IF K=2 THEN PRINT "(to the right)" ELSE IF OP=2 THEN PRINT "(to
         the left)"
HO    2090 INPUT"your name please";A$:A$=LEFT$(A$,8)
IB    2100 N$=SPACE$(8-LEN(A$))+A$
MM    2110 PRINT"Press your joystick's fire button ";
EA    2120 J=0
```

```
NC    2130 IF STRIG(J) THEN PRINT ELSEJ=(J+1)MOD3:GOTO 2130
BF    2140 RETURN
AH    2150 ' ---- setup court screen
BH    2160 COLOR 15,4,13:SCREEN 1,2
EB    2170 LINE(0,0)-(255,191),1,B
FK    2180 LINE(16,8)-(240,150),1,BF
DF    2190 LINE(16,167)-(238,167),0
CH    2200 LINE(16,8)-(0,0),1
BH    2210 LINE(240,8)-(255,0),1
BG    2220 LINE(16,150)-(0,191),1
CM    2230 LINE(240,150)-(255,191),1
BD    2240 LINE(126,184)-(129,121),15,BF
BC    2250 LINE(126,120)-(129,100),14,BF
DP    2260 LINE(24,16)-(232,40),11,B
FA    2270 LOCATE 32,24 :COLOR5:PRINT N1$": 0"
AN    2280 LOCATE 140,24 :COLOR8:PRINT N2$": 0"
FI    2290 COLOR 15,4
HG    2300 FORK=0TO3:SPRITE$(K)=S$(K):NEXT
BA    2310 RETURN
GG    2320 ' ---- sprites
LE    2330 DATA FF,FF,0,0,0,0,0,0,0,0,0,0,0,0,0,0,FF,FF,0,0,0,0,0,0,0,0,0,0
      ,0,0,0,0
DH    2340 DATA 60,78,3E,0F,03,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,80,E0,F8,3E,0E,0
      ,0,0,0,0,0,0,0
BN    2350 DATA 0,0,0,1,7,1F,7C,70,0,0,0,0,0,0,0,0,0,6,1E,7C,F0,C0,0,0,0,0,0,
      0,0,0,0,0,0
IJ    2360 DATA 3,F,19,37,37,6F,7F,6F,7F,7F,7F,3F,3F,1F,F,3,C0,F0,78,FC,FC,
      FE,FE,FE,FE,FE,FE,FC,FC,F8,F0,C0
FC    2370 RESTORE 2330
DB    2380 FOR J=0TO3:FOR K=1TO32:READ D$
BJ    2390 S$=S$+CHR$(VAL("&H"+D$)):NEXT K
CJ    2400 S$(J)=S$:S$="":NEXT J
AP    2410 RETURN
EK    2420 ' ----- def literel variables
AK    2430 DATA 0,-4,2,0, 4,-4,2,1, 4,0,0,1, 4,4,-5,1, 0,4,-5,0, -4,4,-5,2,
       -4,0,0,2,-4,-4,2,2
EP    2440 RESTORE 2430
AL    2450 FOR K=1TO8:READ XD(K),YD(K),A(K),B(K) :NEXT
AE    2460 D(1)=3:D(2)=-3
CF    2470 ON SPRITE GOSUB 510
BG    2480 RETURN
END
```

Merry Christmas

buy, trade & sell

NO TIME FOR ADS-
BACK NEXT ISSUE!

MERRY CHRISTMAS
& HAPPY NEW YEAR
FROM
THE S.A.U.G. STAFF