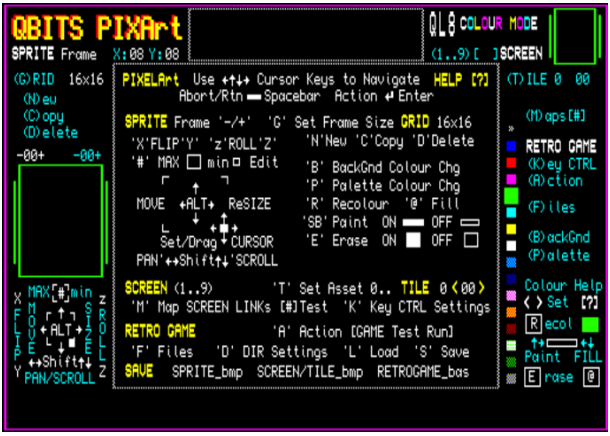
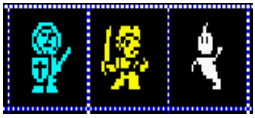


Introduction

The 1980's appeal and accessibility of Computer Games with home computers expanded rapidly as with their growing graphics capabilities and creation of 2D objects named Sprites. These Images represented by a matrix of tiny dots, pixels, displayed to screen were stored in memory as Character Bitmaps.

Creating Pixel Characters and Background screens soon became a recognised Art form. Game Screens in the nineteen eighties were typically 256 by 192 pixels, limiting detail and where the best of the Sprite Designers created characters with 8x8 Pixels Grids.



QBITS PIXEL Art

The 1980's aspiration was to write a Program in SuperBASIC that could be used to create simple RETRO Games for the Sinclair QL Platform. Unfortunately, the performance of running a Program through the QL Interpreter was unacceptably slow and Sprite Bitmaps required lots of memory. Today with the extended range of QL Platforms and enhanced Software, the Speed and available Memory may not be the issue, but code compatibility becomes a primary concern.

QBITS Prog Concept

A SuperBASIC program that builds in Stages the necessary elements for a Retro Game. For **Stage One** the Sprite Designer began with a rewrite of code taken from an earlier **QBITS** Progs **BITMap Designer** and **QLFont Editor.** Added to the Grid functions provided by earlier Progs are ReSize of the Edit area within a Frame, Plus Fill and Recolour.

Stage Two was to assemble Screen backgrounds with SPRITES designed as TILEs. These being copied across from the Spite Designer to form a Tile Library then deployed to build sections of the background. Each TILE to be set as Wall, Floor, Hazard, Reward etc. and linked to any required action. Then the ability to Link [MAP] multiple Background Screens.

Stage Three Control settings for a SPRITE moved by a Player or under Program control. A GAME Title that can be edited. Display of counters for Score & Lives and provide Settings for Hazard or Reward Gains and Losses. Plus, a limited Test to explore SPRITE actions.

Stage Four RETRO Game Test Run to check Play elements, then a SAVE as standalone code.

QBITS PIXELArt - Palette Choice

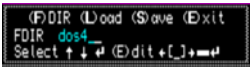
Developed using the **QPC2** environment this offers several Colour Modes. The **COLOUR_QL** displays 0-7 as **Black Red Magenta Green Cyan Yellow White** with **8…255** as a Colour, Contrast and Stipple combination. **COLOUR_PAL** uses **0…255** as a range of colours and shades from a Palette based on 24Bit RGB values (sixteen million variations).

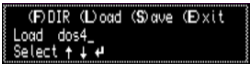
Select with Left [**QL8** or **PAL**] Right Cursor and Enter.

Note: Default is QL8 [Reverting to Mode 4 for BBQL]

QBITS PIXArt - File Management

Press (**F**) to change File **DIR**ectory Select Drive and Enter. Then Press (**E**) to Edit **Drive/SubDIRectory** name. Enter to Action.





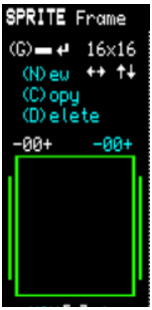
Press (**L**) for **Load**. Select Drive with Up/Down Cursors and Enter. Program carries out a search for QBITS **_bmp** files. Select a file with Up/Down Cursors. Abort with Spacebar or Load with Enter.

Press (**S**) to **Save** Current File. Press (**E**) to Edit filename. A file saved in SPRITE Mode will contain basic header followed by SPRITE BITMaps. A file saved in SCREEN Mode has an extended header with Screen BackGnd and MAPing information followed by TILE Library BITMaps. A RETRO Game file will Save as a standalone Program.

If file exists an Overwrite Y/N prompt is displayed.

QBITS PIXELArt - SPRITE Mode

Loading a SPRITE _bmp file will set the Grid Size. If Frame number is set to -00+ the SPRITES are displayed in Pixel Grid Size as a group across central screen.



QBITS PIXEL Art - (G)RID

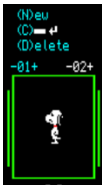
Press '**G**' for **SPRITE Mode** or to change **Frame** Size, use Left/Right Cursors for Columns, Up/Down for Rows. Abort with ▬ Spacebar or Action with ↵ Enter. Then you can use **-/+** to select a SPRITE Frame.

QBITS PIXEL Art - Frame

Press '**N**' to add a **New Frame**. (N)▬ ↵

Press '**C**' to **Copy** present Frame to another

Press '**D**' to **Delete** current Frame. (D)▬ ↵



QBITS PIXEL Art - Paint

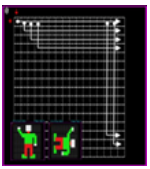
Select **Paint** Colour with Left **< >** Right Chevron keys. Colour is shown as Enlarged Square that moves Up/Down the Palette Bar.

(**B)ackGnd** sets the background colour for all frames. Press '**B**' Select a Colour from Palette Bar. Abort with ▬ or ↵ to Action.

(P)alette 0..7 set to **Black Red Magenta Green Cyan Yellow White** colours 8..15 hold startup default QL8 or PAL colours. Press 'P' to select a Colour and use ← ↑↓ →Cursor keys to change Palette number and display the Changed Palette colour [Abort ▬ or Action ↵].

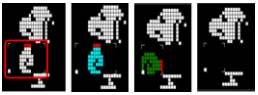
Select Grid position with ← ↑↓ → Press **Spacebar** to toggle **Paint ON/OFF** moving the cursor extends this into other columns and/or rows. Press '**E**' for **[E]rase**, moving over Painted Cells restores them to Background colour. Press '**@**' to Fill Frame Edit area with chosen colour or use with [**E**]rase to Reset cells to Background colour. To use **(R)ecol** - Set a New Colour and Press '**R**' then select the Frame colour you wish to change. Press ↵ or ▬ to Abort.

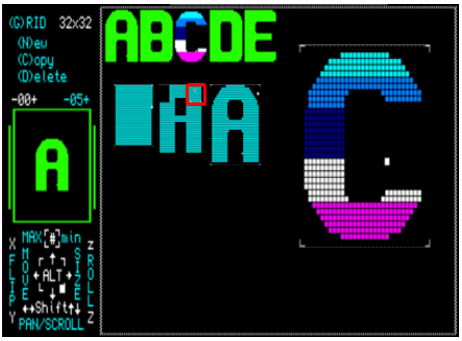
QBITS PIXEL Art - Edit



Press '**xX**' to **FLIP** on Horizontal or '**yY**' on Vertical axis, '**zZ**' to **ROLL** 90° clockwise or anticlockwise. To move cells within Frame Grid use **Shift⁺Cursors** to **PAN** or **SCROLL**.

Press '**#**' to MAX[**#**]min the Edit area of Frame Grid. Edit Area can be resized by placing Screen Cursor on a corner and dragged with **ALT⁺** ← ↑↓ → use **FLIP**, **ROLL**, **PAN / SCROLL, FILL [@]** and **[R]ecol**our within the ReSized Edit Area or Whole of Grid Frame.





QBITS PIXEL Art - Design

Use Edit Tools, Paint Options, ReSized Grid Areas and Copy to build colourful SPRITES.

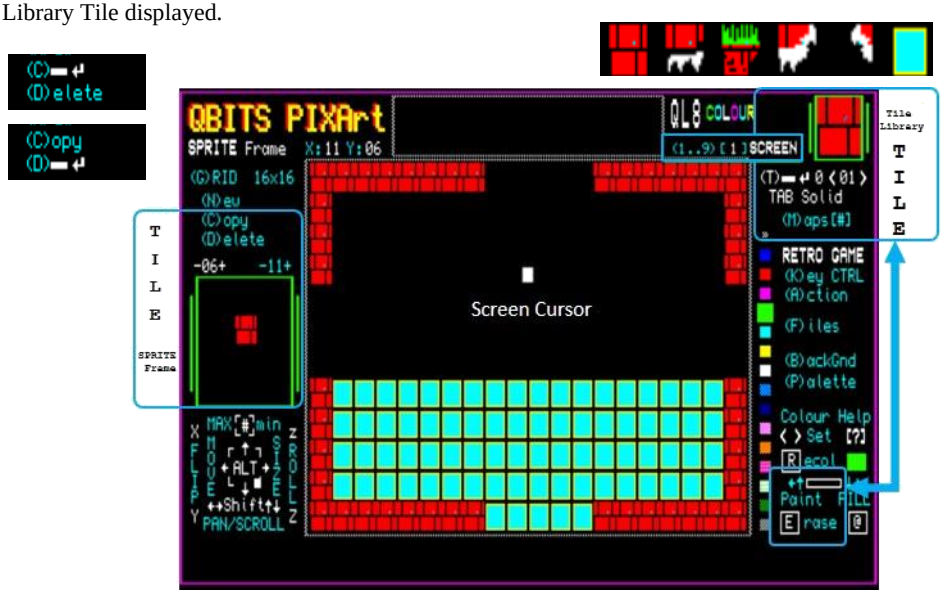
Note: FILL **[@]** Grid. ReSize **[#]** Edit Area Reposition with **ALT⁺Cursors. [E]**rase + **[@]** Cell Group. **[R]ecol**our Whole frame as in **A** or ReSized Areas of Cell groupings to achieve as shown in Fame **C**.

QBITS PIXEL Art - SCREEN Mode

Loading a SCREEN/TILE_bmp file includes the Tile Library used to build background screens. Press '**T**' for SCREEN Mode or to set TILE Attribute this will be used to determine action when a moving Sprite comes into contact.

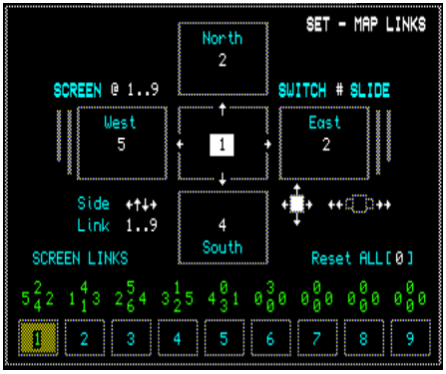
QBITS PIXELArt - (T)ILE

Adding a TILE. Select SPRITE Frame with **-/+** then Press (**C**) to copy to TILE Library. It is shown in TILE Frame top right with number below **<** 01 **>** incremented. Press (**D**) to Delete the Library Tile displayed.



Select a Library Tile with **< >** Chevron keys and Position screen Cursor with ⬅ ⬇ ⬆ ➡Toggle Spacebar ▬ **ON/OFF** to **Paint** individual Tile to SCREEN or extend as a row or column. Use [**E**]rase to Remove.

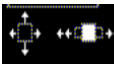
QBITS PIXEL Art - Maps

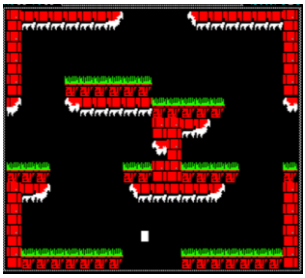


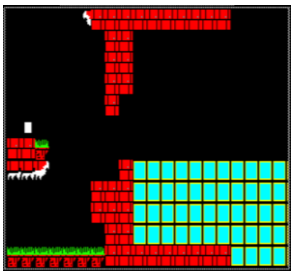
Press (**M**) to enter **MAP LINKS**
To Exit press Spacebar or Enter.

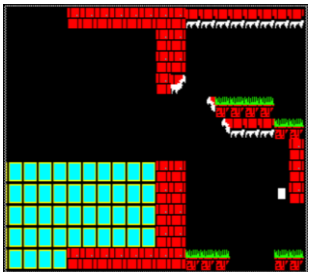
Press **@** to Select a Primary Screen **1..9**. Then use Cursors to Select a Compass direction and Set a Screen Link **1..9**. The bottom screen MAP displays the changed Link Settings.

Select **SWITCH** or **SLIDE** with **#** back in SCREEN Mode Press [**#**] and move Cursor to an EXIT point to Test SCREEN Links.

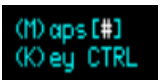








SWITCH - SLIDE



QBITS PIXELArt - Key CTRL

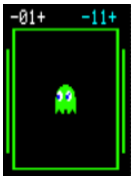
Press '**K**' to access setup for SPRITE 1&2 as Player Controlled and 3..6 Computer controlled. A SPRITE is inactive if the default Frame **@** is set to **[00]**. SPRITES 3..6 can be set as Hazard or Reward and need Score and/or Live Charges to be set.

Press 'N' to change Title, use ← [_] → cursors to position underscore. Add/Delete characters. [▬ Abort or ↵ Action]

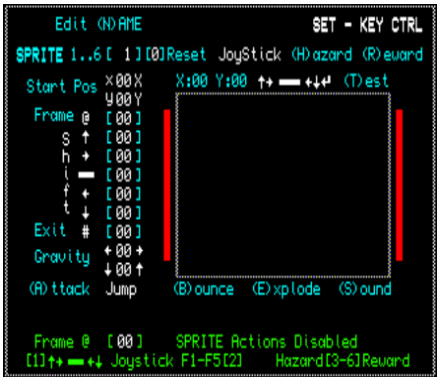
QBITS PIXEL Art - Action SPRITES

Note: First Load SPRITE_bmp File

-00+ View SPRITE Frames Select 1..6 **[ ]**



Screen Start Position **x00X y00Y**
Set Default Frame **@** **[ ]**
Direction SPRITES ↑→ **[ ]** ←↓
(A)ttack) Spacebar ▬ **[ ]**
ie. Explode Exit **#** **[ ]**
Prog CTRL Gravity **X00X Y00Y**
(**A**)ttack **JUMP : FIRE**



QBITS PIXEL Art - Test

Press '**T**' to Test movement and reaction. The screen is set with border Tiles and Exits on each side. Tiles act as solid objects, Test checks for Spacebar (**A**)ttack settings and Collision Responses dictated by (**B**)ounce and (**E**)xplode.

Program-run Sprites move according to Gravity settings. Player Controlled Sprites use cursors and can speed up or slow down dependant on key presses in any one direction.

Press '**S**' ON/OFF for accompanying (**S**)ound settings.

QBITS PIXEL Art – SPRITE Moves

These would start with a default Image displayed to screen with further images for directional changes made with Cursor Keys or Spacebar actions. Then Speed of movement is defined by Horizontal and Vertical Gravity values xx yy. Reaction to other screen objects static or moving are set with (**A**)ttack (**B**)ounce (**E**)xplode. The Program Controlled Sprites are sloe set for (**H**)azard or (**R**)eward. Accompanying any Action Sound can be set ON/OFF.

QBITS PIXEL Art – (A)ttack Concepts

Actions for Player Sprites are governed by the four Cursor keys and Spacebar that mimic a Joysticks direction and fire button. The fire button is used to make a Jump or fire a Laser or Shells dependant on choice. Platform Games that Slide between Screens tend to disable Player controlled vertical ⇅ movement and activate Jump [Spacebar] as the means of Ascending Levels. Vertical Gravity then applies only to computer controlled Decent [falls].

Jump action includes a stationary position and a moving one. If stationary Jump is calculated as Tile height **th** with Pixel coordinates set from top left of screen then **yy= -th** can be used to calculate the new Jump screen position. To calculate for a moving Jump **yy= -ABS(xx)** can be deployed. The Decent Speed or vertical Gravity is then set as **yy=+th/2.** Collision Detection checks are required for both Ascent and Decent to Landing Area.

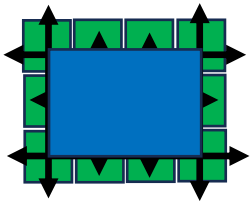
Fire action is determined by the last direction of movement. The Firing point is calculated from the Sprite Centre **xc=sx+cm/2+(2+cm/2)*cf & yr=sy+rm/2+(1+rm/2)*rf.** Where **cm** & **rm** are the Sprite width and height and **cf** & **rf** provide positive or negative direction of **sx sy** vales. The **2** & **1** Constants are added to avoid any XOR pixel debris. The Target position is then calculated as **xc=xc+4*cf yr=yr+4*rf** and incremented within a FOR loop to mimic a fired projectile with the results from any Collision Detection actioned such as exploding the Target Area.

QBITS PIXEL Art – Collision Detection

In two-dimensional Games Collision Detection occurs when a Sprite moves into areas occupied by other moving Sprites or a Background Tile. What then takes place can be one absorbed or destroyed by the other or with one or both continuing with altered direction and/or speed. Dependant on the encountered Object(s) setting a Players Status is updated.

As Sprites and Screen Tile might be of different sizes any Collision Detection must resolve all possible encounters The Screen Tile size is used as a default. When the xx yy Speed is added to a Sprites x,y position a FOR loop checks for any possible overlaps with other Object in range. The Y axis is checked first this helps the X axis moves appear smoother.

Check for YY Axis Collisions
Resolve & Set Response
Check for XX Axis Collision
Resolve & Set Response
Clear Old Sprite Position
Update Gravity xx yy
Draw Sprite Position with Response



WORK IN PROGRESS

QBITS PIXEL Art BITMaps

The 1980's Games utilised the BITMaps of character fonts and had one colour. **QBITS Font Edit** Special Edition has three Demo Games that explored this approach. **QBITS BITMap Design** explored multi colour Objects with various Grid sizes. The BITMaps saved contained Header to describe file Type and info such as Grid Size of the Bitmap Data that followed.

QBITS PIXEL Art - _bmp Files [type sz]

IF sz=0:bm=frt:mlth= 8+bm*cm*rm :addr=ALCHP(mlth):ptr=addr+8
IF sz=1:bm=frt:mlth=16+bm*cm*rm :addr=ALCHP(mlth):ptr=addr+16
IF sz=2:bm=tm:mlth=52+2160+bm*256 :addr=ALCHP(mlth):cm=16:rm=16
IF sz=3:bm=tm:mlth=52+2160+200+bm*256+bm:addr=ALCHP(mlth):cm=16:rm=16

SPRITE _bmp		Bytes
Bytes 0… 7	QL8 or PAL ; bm : cm: rm : bg : sz	8
Bytes 8…15	Changeable Palette Colours	16
	bm*cm*rm SPRITE Bitmaps ???	
TILE _bmp		
Bytes 0…….15	QL8 PAL bm,cm,rm,bg,sz + Palette	16
Bytes 16…….51	Screen Links 9x4 = 36	52
Bytes 52…….2212	Screen Tiles 9x20x12 = 2160	2112
Bytes 2212….2412	Key CTRL 9x20 = 180 +	
	bm*cm*rm TILES bitmaps ??? + bm TILE Action	

WORK IN PROGRESS

QBITS PIXEL Art – (A)ction [RETRO Play]

RGen Part One - The constructed RETRO Game is accessed in PLAY Mode to check actions are performing as expected. A return to PIXEL Art for any adjustments.

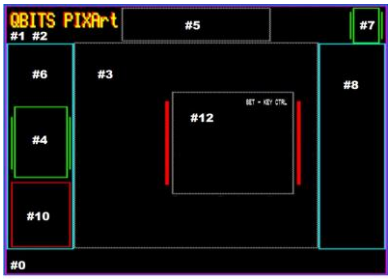
RGen Part Two - If all is Satisfactory then Press (F)iles and (S)ave to produce a standalone code that can be loaded and run independently of QBITS PIXEL Art…

,

UNDER CONSTRUCTION

QBITS PIXEL Art Prog Code [S/SuperBASIC]

```
1000 REMark QBITS_PIXELArt_RGen [QBITS PIXEL Art Game Design 2024 - QPC2] M1620

1002 dev$='dos1_':MODE 4:gx=0:gy=0  :REMark Basic Settings

1004 WHEN ERRor :eck=1:CONTINUE:END WHEN :COLOUR_QL

1006 REMark Import QBITSConfig Settings - QPC2
1007 OPEN_IN#9,dev$&'QBITSConfig':INPUT#9,gx\gy\dn$\dev$\dn%\dm%
1008 DIM drv$(dm%,16):FOR d=0 TO dm%:INPUT#9,drv$(d):END FOR d:CLOSE#9

1010 REMark PIXEL Array Settings
1011 DIM FG(32,64,64),TG(64,64),File$(20,20),CP(15),BEEP$(4,34),RGN$(13)
1012 DIM Tile(120,15,15),TAss(120),TScn(9,20,12),WScn(20,12),TMap(9,4),SK(9,20)
1013 DIM TAS$(9,6):PFile$='':TFile$='':SFile$='':k=0:lks=0:RGN$='RETROMASTER'
:
1015 COLOUR_QL:Init_Screens:Init_Palette:Init_Layout:PIXArt_Menu

1017 REMark PIXArt SetUP

1019 DEFine PROCedure Init_Screens
1020 WINDOW#0,512, 32,gx,gy+224  :PAPER#0,  0      :BORDER#0,1,3    :CLS#0
1021 WINDOW#1,512,256,gx,gy      :PAPER#1,  0:
1022 WINDOW#2,512,224,gx,gy      :PAPER#2,  0      :BORDER#2,1,3    :CLS#2
1023 OPEN#3,scr_ :WINDOW#3,328,196,gx+ 92,gy+ 35:BORDER#3,1,248:CLS#3
1024 OPEN#4,scr_ :WINDOW#4, 72, 68,gx+ 12,gy+ 96  :CLS#4
1025 OPEN#5,scr_ :WINDOW#5,200, 32,gx+156,gy+  2 :BORDER#5,1,248:CLS#5
1026 OPEN#6,scr_ :WINDOW#6, 90,196,gx+  2,gy+ 36  :CLS#6
1027 OPEN#7,scr_ :WINDOW#7, 36, 34,gx+466,gy+  2  :CLS#7
1028 OPEN#8,scr_ :WINDOW#8, 90,196,gx+420,gy+ 36:CLS#8
1029 OPEN#10,scr_:WINDOW#10,80, 72,gx+  8,gy+168
1030 QBT$='QBITS PIXArt':QTitle 1,2,4,4,QBT$:QGT$='TITLE'
1031 END DEFine
```



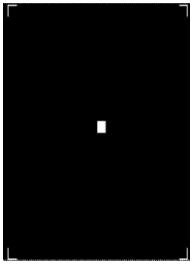

```
1033 DEFine PROCedure Init_Palette
1034 CSIZE#5,0,1:CSIZE 0,0:pmb$='COLOUR MODE':CP(7)=7
1035 CSIZE 1,1:pm$='QL8':QBold 1,7,360,2,pm$:RESTORE 1048
1036 IF VER$(1)>=2.98
1037    COLOUR_QL
1038    INK#5,5:CURSOR#5,12,6:PRINT#5,'Select Palette ';
1039    INK#5,7:PRINT#5,'QL8 ◄    ► PAL◄ ':BLOCK#5,2,6,181,12,7
1040    REPeat P_lp
1041      QBold 1,7,360,2,pm$:k=CODE(INKEY$(-1))
1042      IF k=192:pm$='QL8' :RESTORE 1048:tcol%=255
1043      IF k=200:pm$='PAL' :RESTORE 1049:tcol%=10
1044      IF k= 10:IF pm$='PAL':COLOUR_PAL:END IF :CLS#5:EXIT P_lp
1045    END REPeat P_lp
1046 END IF
1047 CSIZE#5,0,0:CSIZE 0,0:FOR i=0 TO  15:READ CP(i)
1048 DATA 0,1,2,3,4,5,6,7,225,200,227,230,210,31,224,254
1049 DATA 0,4,2,5,3,7,6,1,25,50,31,22,46,36,61,10
1050 FOR i=1 TO 11:QBold 1,RND(2 TO 7),380+i*6,6,pmb$(i)
1051 END DEFine:
```

Note: Palette Settings

8..15 Default QL8 or PAL

```
1053 DEFine PROCedure Init_Layout
1054 BLOCK 2,26,462,6,CP(4)      :BLOCK 2,26,504, 6,CP(4) :BORDER#7,1,CP(4)
1055 BLOCK 2,50,8,106,CP(4)      :BLOCK 2,50,86,106,CP(4) :BORDER#4,1,CP(4)
1056 BLOCK#8,14,12,21,182,CP(7)  :BLOCK#8,14,12,71,182,CP(7) :CLS#3:CLS#4
1057 QBold 1,7,  6, 24,'SPRITE'  :QBold 1,7,420,24,'SCREEN'  :CLS#5:CLS#7
1058 QBold 8,7,22,44,'RETRO GAME':QBold 8,7,19,138,'< >     [?]'
1059 FOR i=0 TO 15:BLOCK#8,8,6,6,36+10*i,CP(i):END FOR i
1060 BLOCK#8,14,12,22,150,CP(7):BLOCK#8,12,10,23,151,0
1061 RESTORE 1062:FOR i=1 TO 21:QPrnt:END FOR i
1062 DATA 1,7,48,25,'Frame',1,5,92,25,'X:  Y:',1,5,360,25,'(1..9)[  ]'
1063 DATA 6,5,4,4,'(G)RID',6,5,12,16,'(N)ew',6,5,12,26,'(C)opy',6,5,12,36,'(D)elete'
1064 DATA 8,5,4,4,'(T)ILE 0  00',8,5,20,26,'(M)aps[#]',8,5,22,55,'(K)ey CTRL'
1065 DATA 8,5,22,65,'(A)ction',8,5,22,81,'(F)iles',8,7,26,151,'R'
1066 DATA 8,5,22,99,'(B)ackGnd',8,5,22,110,'(P)alette',8,5,38,151,'ecol'
1067 DATA 8,5,40,138,'Set',8,5,21,128,'Colour Help',8,5,26,162,'←↑  ↓→'
1068 DATA 8,5,21,171,'Paint  FILL',8,5,38,183,'rase'
1069 BLOCK#8,26,5,40,165,CP(7)   :BLOCK#8,14,9,70,152,CP(4):INK#8,CP(7):QRubber
1070 RESTORE 1071:FOR i=0 TO 9:READ TAS$(i)
1071 DATA 'Solid ','Floor ','Hazard','Reward','???  '
1072 DATA '???  ','???  ','???  ','???  ','???  '
1073 END DEFine

1075 DEFine PROCedure Init_Guide
1076 ch=10:INK#ch,CP(7):RESTORE 1086
1077 CURSOR#ch,0, 4:PRINT#ch,'X          z':M1$='FLIPROLL'
1078 CURSOR#ch,0,48:PRINT#ch,'Y          Z':M2$='MOVESIZE':INK#ch,CP(5)
1079 FOR i=0 TO 3:CURSOR#ch, 0,14+i*8:PRINT#ch,M1$(i+1);'          ';M1$(i+5)
1080 FOR i=0 TO 3:CURSOR#ch,12,10+i*8:PRINT#ch,M2$(i+1);'       ';M2$(i+5)
1081 BLOCK#ch,24,20,28,17,CP(7):BLOCK#ch,22,18,29,18,0:BLOCK#ch,16,20,32,17,0
1082 CURSOR#ch,12,0:PRINT#ch,'MAX   min'
1083 BLOCK#ch,14,11,33,0,CP(7):BLOCK#ch,12, 9,34,1,0
1084 BLOCK#ch,14, 3,33,4,0:BLOCK#ch, 6,11,37,0,0:BLOCK#ch,5,5,47,33,CP(7)
1085 FOR i=1 TO 7:READ pc,px,py,p$:INK#ch,CP(pc):CURSOR#ch,px,py:PRINT#ch,p$
1086 DATA 5,9,52,'PAN/SCROLL',7,12,42,'← ↑ Shift ↓ →  ',7,37,1,'#'
1087 DATA 7,37,12,' ↑ ',7,22,22,'←   → ',7,31,23,'ALT',7,37,33,' ↓ '
1088 END DEFine

1090 DEFine PROCedure QTitle(ch,chz,tx,ty,str$)
1091 OVER#ch,1:CSIZE#ch,chz,1
1092 INK#ch,2:FOR i=0 TO 1:CURSOR#ch,tx+i,ty  :PRINT#ch,str$
1093 INK#ch,6:FOR i=2 TO 3:CURSOR#ch,tx+i,ty+1:PRINT#ch,str$
1094 OVER#ch,0:CURSOR#ch,0,0:CSIZE#ch,0,0
1095 END DEFine

1097 DEFine PROCedure QBold(ch,bc,bx,by,B$)
1098 INK#ch,CP(bc):CURSOR#ch,bx,by  :PRINT#ch,B$
1099 OVER#ch,1:CURSOR#ch,bx,by+1:PRINT#ch,B$:OVER#ch,0
1100 END DEFine

1102 DEFine PROCedure QPrnt
1103 LOCal ch,i,x,y,p$:READ ch,i,x,y,p$:INK#ch,CP(i):CURSOR#ch,x,y:PRINT#ch,p$
1104 END DEFine
```

```
1106 REMark PIXEL Art Menu

1108 DEFine PROCedure PIXArt_Menu
1109 INK#8,CP(7):rub=0:QRubber:k=0:ic=4:oic=4:sic=4:bg=0:pn=0:
1110 tmax=96:tn=0 :tm=0:sn=1:obj=0:frm=31:fr=0:frt=0:cm=16:rm=16
1111 Init_Guide:PCol:Info:PAUSE:GFrame
1112 REPeat Main_lp
1113   IF fed=1 AND fr>0                                   Gride Mode
1114     IF rub=7:c=x:r=y:FG(fr,c,r)=255:Gbit              Erase - Cell restored to Background
1115     IF cur=7:c=x:r=y:FG(fr,c,r)=ic :Gbit              Paint ON Set Cell colour
1116   END IF
1117   IF fed=2 AND obj=0                                  Tile Mode
1118     IF rub=7:tx=2+x*16:ty=1+y*16:BLOCK#3,16,16,tx,ty,CP(bg):TScn(sn,x,y)=0   Remove Screen Tile
1119     IF cur=7:tx=2+x*16:ty=1+y*16:TDraw 0:TScn(sn,x,y)=tn                     Paint ON Set Screen Tile
1120   END IF
1121   BLOCK#8,24,3,41,166,CP(cur):INK CP(7):INK#8,CP(7):QRubber
1122   CURSOR 106,25:PRINT FILL$('0',2-LEN(x+1));x+1       GRID Cell / Screen TILE XX Position
1123   CURSOR 136,25:PRINT FILL$('0',2-LEN(y+1));y+1       GRID Cell / Screen TILE YY Position
1124   GPos:k=CODE(INKEY$(-1)):ox=x:oy=y:GPos:GLoc
1125   SELect ON k
1126     =27:STOP                                          :REMark STOP
1127     =32:    IF cur=0:cur=7:rub=0:QRubber:ELSE cur=0   :REMark SB ON/OFF
1128     =69,101:IF rub=0:rub=7:QRubber:cur=0:ELSE rub=0:QRubber  :REMark (E>raser
1129     =45, 95:IF fr>0  :fr=fr-1:FrChg                   :REMark - Frame
1130     =43, 61:IF fr<frt:fr=fr+1:FrChg                   :REMark + Frame
1131     =70,102:DIRFile                                   :REMark (D)DIR
1132     =65 ,97:Retro_Play:GTile                          :REMark (A)ction
1133     =47, 63:col%=CP(6):Info:PAUSE:IF fed=1:GFrame :ELSE GTile  :REMark (?)Help
1134   END SELect
1135   IF fed=1:SGen          Edit Functions for SPRITE Frame
1136   IF fed=2:TGen          Edit Functions For SCREEN TILE MAPS & SPRITE Key-CTRL
1137 END REPeat Main_lp
1138 END DEFine

1140 DEFine PROCedure QRubber
1141 BLOCK#8,12,10,22,183,CP(rub):OVER#8,-1:CURSOR#8,24,183:PRINT#8,'E':OVER#8,0
1142 BLOCK#8,12,10,72,183,CP(cur):OVER#8,-1:CURSOR#8,74,183:PRINT#8,'@':OVER#8,0
1143 END DEFine

1145 DEFine PROCedure GPos          Grid Cursor Position
1146 OVER#3,-1:BLOCK#3,gpw,gph,gpx,gpy,CP(7):OVER#3,0
1147 END DEFine

1149 DEFine PROCedure GLoc          Grid Location XX/YY change
1150 SELect ON k
1151   =192:IF x> 0:x=x-1          :REMark ← x \
1152   =200:IF x<xm:x=x+1          :REMark → x  Grid Tile
1153   =208:IF y> 0:y=y-1          :REMark ↑ y  Position
1154   =216:IF y<ym:y=y+1          :REMark ↓ y /
1155 END SELect
1156 IF fed=1:gpw=cw+1:gph=rh+1:gpx=-1+zx+x*cw:gpy=-1+zy+y*rh   Settings for Gride Mode
1157 IF obj=1 AND lks=0:Scn_Switch                              Check for Screen Switch [Tile Mode]
1158 IF obj=1 AND lks=1:Scn_Slide                               Check for Screen Slide  [Tile Mode]
1159 IF fed=2:gpw=8:gph=8:gpx=6+x*16:gpy=5+y*16                 Settings Check for Tile Mode
1160 END DEFine
```



```
1162 DEFine PROCedure DIRFile
1163 CLS#5:INK#5,CP(7):CURSOR#5,16,1:PRINT#5,'(D)DIR (L)oad (S)ave (E)xit'
1164 OVER#5,1:CURSOR#5,23,1:PRINT#5,'D     L     S     E'    :OVER#5,0
1165 k=CODE(INKEY$(-1))
1166 SELect ON k
1167    =68,100:SelDrv 1,'Drive '     :REMark (D)DIR
1168    =76,108:BMLoad                :REMark (L)oad
1169    =83,115:BMSave                :REMark (S)ave
1170    =69,101:QExit                 :REMark (E)xit
1171 END SELect
1172 CLS#5
1173 END DEFine
```

Note: Press '**D**' then '**E**' to Edit Drive/SubDIR : to Load Press '**L**' Select Drive then Select from List presented: For '**S**'ave Press '**E**' to Edit Filename if required.

Note: Switch to GRID Mode – Set TILE Mode Inactive - Change Settings and Show SPRITE Frame

```
1175 DEFine PROCedure GFrame
1176 BLOCK#8,30,10,56,4,0:TNum 5:obj=0:CLSObj 5:CLS#7                :REMark Disable Tile Mode
1177 INK#6,CP(7):CURSOR#6,52,4:PRINT#6,cm;'x';rm :xm=cm-1:ym=rm-1
1178 px=34-cm/2:py=33-rm/2:zx=66:zy=1:cw=192 DIV rm:rh=cw:sz=1
1179 x=cm/2-1:y=rm/2-1:rub=0:cur=0:fed=1:GLoc:FrNum frt,5,56,50:FrChg
1180 END DEFine
```

Note: Switch to TILE Mode – Set GRID Mode Inactive - Change Settings and Show SCREEN/TILE Background

```
1182 DEFine PROCedure GTile
1183 INK#6,CP(5):CURSOR#6,52,4:PRINT#6,cm;'x';rm                     :REMark Disable Grid Mode
1184 QBold 8,7,55,4,'<   >':TNum 7:tmax=120:rub=0:cur=0:sz=3:lks=0:k=0
1185 x=9:xm=19:y=5:ym=11:obj=0:CLSObj 5:fed=2:GLoc:Tile_MAP
1186 END DEFine
```

Note: Switch to RETRO Mode – Set SCREEN/Tile Background and show Active SPRITES

```
1188 DEFine PROCedure Retro_Play
1189 INK#8,CP(7):CURSOR#8,20,65:PRINT#8,'(A)[Esc]':fed=0:Set_Score
1190 Tile_MAP:QTitle 3,2,132,50,'UNDER':QTitle 3,2,90,70,'CONSTRUCTION'
1191 REPeat Play_lp
1192    k=CODE(INKEY$(-1))
1193    IF k=70 OR k=102:DIRFile
1194    IF k=27:EXIT Play_lp
1195 END REPeat Play_lp
1196 INK#8,CP(5):CURSOR#8,20,65:PRINT#8,'(A)ction ':CLS#5
1197 END DEFine
```

UNDER
CONSTRUCTION

Note: Upon Exit close Open Channels and revert to QL_COLOUR Setting- LRUN QBITSProgs_bas If available

```
1199 DEFine PROCedure QExit
1200 CURSOR#5,160,13:PRINT#5,'Y/N':IF GAns=0:RETurn
1201 CLOSE#10:CLOSE#8:CLOSE#7:CLOSE#6:CLOSE#5:CLOSE#4:CLOSE#3:CLS#2:COLOUR_QL
1202 CURSOR#0,0,0:PRINT#0,'Bye...' :LRUN dn$
1203 END DEFine
```

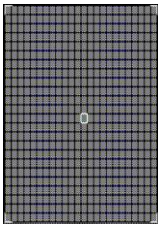
1205 REMark **SPRITE Generator**

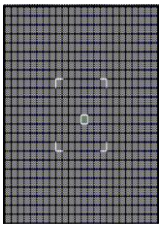
Note: SPRITE Frames 1…32 max - Saved as _bmp File
Bytes: [0 to 2] QL8 or PAL Header ID
[3] bm number of SPRITE Bitmaps GRID Size [4] cm columns [5] rm rows
[6] bg BackGnd colour [7] sz File Loading Reference
[8 to 15] BackGnd Colours range 8…255 of selected Palette QL or PAL
[16……..] SPRITE BITMaps [bm x cm x rm]


```
1207 DEFine PROCedure SGen
1208 SELect ON k
1209   =84,116:GTile                                          :REMark (T)iles
1210   =10,32,44,46,60,62,9,253::oic=ic:PCol                  :REMark < > Colour Chg
1211   =78,110:IF frt<frm:FrNEW :CURSOR#6,29,16:PRINT#6,'ew   ' :REMark (N)ew
1212   =67, 99:IF frt>1   :FrCOPY:CURSOR#6,29,26:PRINT#6,'opy ' :REMark (C)opy
1213   =68,100:IF frt>0   :FrDEL :CURSOR#6,29,36:PRINT#6,'elete' :REMark (D)elete
1214   =39, 64:IF frt>0   :FChg 1:GDraw                       :REMark (@) Grid Fill
1215   =82,114:IF frt>0   :FrSwap                             :REMark (R)eColour
1216   =70,103:GSize                                          :REMark (G)rid
1217   =66, 98:IF frt>0   :GBGnd                              :REMark (B)kGnd
1218   =80,112:GPalette                                       :REMark (P)alette
1219   =  35:IF fr>0:EMaxMin                                  :REMark # Edit Area
1220   =193:EMove:ELeft    :EMove                             :REMark ALT ¼
1221   =201:EMove:ERight  :EMove                              :REMark ALT ½ Move
1222   =209:EMove:EUp     :EMove                              :REMark ALT ¾ SIZE
1223   =217:EMove:EDown :EMove                                :REMark ALT ¿
1224   =196:dm=0:pa=cs+cn-1:pb=cs:pc=cs-1:pd=cs   :GSlid      :REMark Shift ¼ PAN
1225   =204:dm=0:pa=cs:pb=cs+cn-1:pc=cs:pd=cs-1   :GSlid      :REMark Shift ½ PAN
1226   =212:dm=1:sa=rs+rn-1:sb=rs:sc=rs-1:sd=rs   :GSlid      :REMark Shift ¾ SCROLL
1227   =220:dm=1:sa=rs:sb=rs+rn-1:sc=rs:sd=rs-1   :GSlid      :REMark Shift ¿ SCROLL
1228   =88,120:xf=cs+cn-1:yf=rs:xz=-1:yz=1        :GFlip      :REMark X FLIP
1229   =89,121:yf=rs+rn-1:xf=cs:yz=-1:xz=1        :GFlip      :REMark Y FLIP
1230   =122:rxm=rn-1:rym=0:xt=-1:yt=1             :GRoll      :REMark z ROLL Anti-CW
1231   = 90:rym=rn-1:rxm=0:yt=-1:xt=1             :GRoll      :REMark Z ROckWise
1232 END SELect
1233 END DEFine

1235 DEFine PROCedure PCol
1236 BLOCK#8,12,10,4,34+oic*10,0:BLOCK#8,8,6,6,36+oic*10,CP(oic)
1237 SELect ON k=44,60,253:ic=ic-1:IF ic<pn:ic=15:END IF :END SELect
1238 SELect ON k=46,62,  9:ic=ic+1:IF ic>15:ic=pn:END IF :END SELect
1239 BLOCK#8,12,10,4,34+ic*10,CP(ic) :BLOCK#8, 4,3,8,37,255
1240 IF ic=0:BLOCK#8,12,10,4,34,CP(7):BLOCK#8,10,8,5,35,0
1241 END DEFine

1243 DEFine FuNction GAns(ax,ay,a$)
1244 REPeat lp
1245   k=CODE(INKEY$(-1))
1246   SELect ON k=10,89,121:RETurn 1
1247   SELect ON k=32,78,110:RETurn 0
1248 END REPeat lp
1249 END DEFine
```

Highlighted Palette Colour

```
1251 REMark SPRITE Frame GRID Edit

1253 DEFine PROCedure EMaxMin
1254 IF cn<cm OR rn<rm
1255    EMove:cs=0:cn=cm:rs=0:rn=rm:EMove
1256 ELSE
1257    EMove:cs=cm/2-4:cn=8:rs=rm/2-4:rn=8:EMove
1258 END IF
1259 END DEFine
```





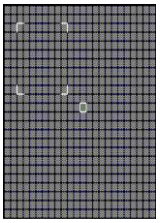
MAX min Edit Area

```
1261 DEFine FuNction Epos
1262 IF cs=x      AND rs=y          :RETurn 1      :REMark Top Left
1263 IF cs=x      AND rs+rn-1=y     :RETurn 2      :REMark Bottom Left
1264 IF cs+cn-1=x AND rs=y          :RETurn 3      :REMark Top Right
1265 IF cs+cn-1=x AND rs+rn-1=y     :RETurn 4      :REMark Bottom Right
1266 RETurn 0
1267 END DEFine
```

Position Edit Area

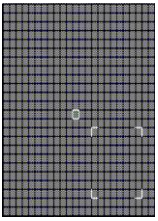
ReSIZE Edit Area

```
1269 DEFine PROCedure EUp
1270 IF rs>0 AND EPos=0:rs=rs-1
1271 IF rs>0 AND EPos=1 OR rs>0 AND EPos=3:rn=rn+1:y=y-1:rs=rs-1
1272 IF rn>8 AND EPos=2 OR rn>8 AND EPos=4:rn=rn-1:y=y-1
1273 END DEFine
```



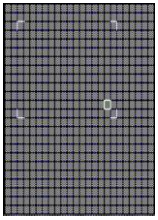
```
1275 DEFine PROCedure EDown
1276 IF rs+rn<rm AND EPos=0:rs=rs+1
1277 IF rn>8 AND EPos=1 OR rn>8 AND EPos=3:rn=rn-1:y=y+1:rs=rs+1
1278 IF rs+rn<rm AND EPos=2 OR rs+rn<rm AND EPos=4:rn=rn+1:y=y+1
1279 END DEFine
```

```
1281 DEFine PROCedure ELeft
1282 IF cs>0 AND EPos=0:cs=cs-1
1283 IF cs>0 AND EPos=1 OR cs>0 AND EPos=2:cn=cn+1:x=x-1:cs=cs-1
1284 IF cn>8 AND EPos=3 OR cn>8 AND EPos=4:cn=cn-1:x=x-1
1285 END DEFine
```



```
1287 DEFine PROCedure ERight
1288 IF cs+cn<cm AND EPos=0:cs=cs+1
1289 IF cn>8 AND EPos=1 OR cn>8 AND EPos=2:cn=cn-1:x=x+1:cs=cs+1
1290 IF cs+cn<cm AND EPos=3 OR cs+cn<cm AND EPos=4:cn=cn+1:x=x+1
1291 END DEFine
```

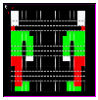
```
1293 DEFine PROCedure EMove
1294 x1=-1+zx+cs*cw:x2=-1+zx+(cs+cn)*cw:y1=-1+zy+rs*rh:y2=-1+zy+(rs+rn)*rh
1295 OVER#3,-1:col%=CP(7)
1296 BLOCK#3,cw+1,1,x1+1 ,y1,col%:BLOCK#3,1,rh+1,x1,y1   ,col%
1297 BLOCK#3,cw  ,1,x2-cw,y1,col%:BLOCK#3,1,rh+1,x2,y1   ,col%
1298 BLOCK#3,cw+1,1,x1+1 ,y2,col%:BLOCK#3,1,rh+1,x1,y2-rh,col%
1299 BLOCK#3,cw,1  ,x2-cw,y2,col%:BLOCK#3,1,rh+1,x2,y2-rh,col%
1300 OVER#3,0
1301 END DEFine
:
```

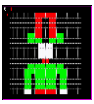


Repositioning Grid Cells

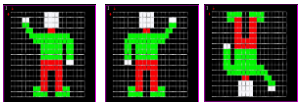
GSide	PAN Left	dm=0:pa=cs+cn-1:pb=cs:pc=cs-1:pd=cs		:
GSlide	PAN Right	dm=0:pa=cs:pb=cs+cn-1:pc=cs:pd=cs-1		
GSlide	SCROLL Up	dm=1:sa=rs+rn-1:sb=rs:sc=rs-1:sd=rs		
GSlide	SCROLL Down	dm=1:sa=rs:sb=rs+rn-1:sc=rs:sd=rs-1		
GFlip	FLIP X	xf=cs+cn-1:yf=rs:xz=-1:yz=1		
GFlip	FLIP Y	yf=rs+rn-1:xf=cs:yz=-1:xz=1		
GRoll	ROLL z	rxm=rn-1:rym=0:xt=-1:yt=1	Anti-CW	**Note**: Edit Area reduced to
GRoll	ROLL Z	rym=rn-1:rxm=0:yt=-1:xt=1	Clockwise	Square of shortest side

```
1303 DEFine PROCedure GSlid                                   Grid Slide
1304 IF dm=0
1305   FOR r=rs TO rs+rn-1
1306     FOR c=1 TO cn-1:TG(pa,r)=FG(fr,pb,r):TG(c+pc,r)=FG(fr,c+pd,r)      Left/Right
1307   END FOR r
1308 ELSE
1309   FOR r=1 TO rn-1
1310     FOR c=cs TO cs+cn-1:TG(c,sa)=FG(fr,c,sb):TG(c,r+sc)=FG(fr,c,r+sd)  Up/Down
1311   END FOR r
1312 END IF
1313 cur=0:GTSet:GDraw
1314 END DEFine
```

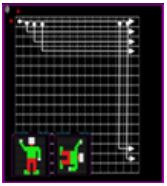




```
1316 DEFine PROCedure GFlip                                   Grid Flip XX  YY
1317 FOR r=0 TO rn-1
1318   FOR c=0 TO cn-1:TG(xf+c*xz,yf+r*yz)=FG(fr,cs+c,rs+r)
1319 END FOR r
1320 cur=0:GTSet:GDraw
1321 END DEFine
```



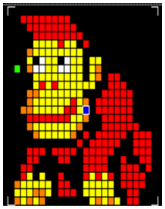
```
1323 DEFine PROCedure GRoll                                   Grid Rotate 90⁰
1324 IF cs+rn>cm:RETurn :ELSE EMove:cn=rn:EMove
1325 FOR r=0 TO rn-1
1326   rx=rxm+(r*xt):ry=rym
1327   FOR c=0 TO rn-1:TG(cs+c,rs+r)=FG(fr,cs+rx,rs+ry):ry=ry+yt
1328 END FOR r
1329 cur=0:GTSet:GDraw
1330 END DEFine
```



```
1332 DEFine PROCedure GTSet                                   Grid Temp
1333 FOR r=rs TO rs+rn-1:FOR c=cs TO cs+cn-1:FG(fr,c,r)=TG(c,r):END FOR c:END FOR r
1332 END DEFine
```

```
1336 DEFine PROCedure GDraw                                   Grid Draw
1337 FOR r=rs TO rs+rn-1:FOR c=cs TO cs+cn-1:GBit:END FOR c:END FOR r
1338 END DEFine
```

```
1340 DEFine PROCedure GPBit                                   Grid Pixel Bit
1341 IF FG(fr,c,r)=255:pcol%=CP(bg):ELSE pcol%=CP(FG(fr,c,r))
1342 IF fed=1:BLOCK#3,cw-1,rh-1,zx+c*cw,zy+r*rh,pcol%
1343 BLOCK#4,1,1,px+c,py+r,pcol%
1344 END DEFine
```



```
1346 REMark SPRITE Frame Actions

1348 DEFine PROCedure GSize
1349 ocm=cm:orm=rm:INK#6,CP(7):CURSOR#6,4,4:PRINT#6,'(G)':FrMes 6,22,4
1350 CURSOR#6,50,14:PRINT#6,' ← ↑  ↓ → ':zx=66:zy=1
1351 REPeat Size_lp
1352   CURSOR#6,52,4:PRINT#6,cm;'x';rm:k=CODE(INKEY$(-1))
1353   SELect ON k
1354     =192:IF cm>=24:cm=cm-8:cw=192 DIV rm:rh=cw
1355     =200:IF cm<=56:cm=cm+8 :IF cm>rm:cw=192 DIV cm:rh=cw
1356     =208:IF rm<=56:rm=rm+8  :IF rm>cm:cw=192 DIV rm:rh=cw
1357     =216:IF rm>=24:rm=rm-8:cw=192 DIV cm:rh=cw
1358     =  32:cm=ocm:rm=orm:EXIT Size_lp
1359     =  10:FrSet:EXIT Size_lp
1360   END SELect
1361 END REPeat Size_lp
1362 BLOCK#6,30,10,50,14,0:INK#6,CP(5):
1363 CURSOR#6,4,4:PRINT#6,'(G)RID  ';:INK#6,CP(7):PRINT#6,cm;'x';rm;' '
1364 END DEFine
```

Note: Grid Size multiples of 8
Range 16 to 64

```
1366 DEFine PROCedure FrSet
1367 cs=0:cn=cm:rs=0:rn=rm:xm=cm-1:ym=rm-1:zx=162-(cm*cw)/2:zy=98-(rm*rh)/2
1368 px=34-cm/2:py=33-rm/2:x=cm/2:y=rm/2:cs=0:cn=cm:rs=0:rn=rm:CLS#3:CLS#4:FrChg
1369 END DEFine
```

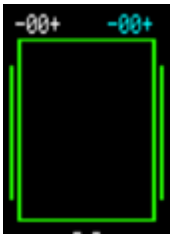
Note: Frame Set - Parameters

```
1371 DEFine PROCedure FrMes(ch,mx,my)
1372 INK#ch,CP(7):CURSOR#ch,mx,my:PRINT#ch,'  ← '
1373 BLOCK#ch,10,3,mx,my+4,CP(7):BLOCK#ch,2,4,mx+18,my+2,CP(7)
1374 END DEFine
```

Note: Frame Message Prompt
Abort Spacebar or Action Enter

```
1376 DEFine PROCedure FrChg
1377 IF fed=1 AND fr=0:FrShow:FrNum fr,7,8,50:RETurn
1378 IF fed=1:FrNum fr,7,8,50:cs=0:cn=cm:rs=0:rn=rm:CLS#3:CLS#4:GDraw:EMove
1379 IF fed=2:FrNum fr,7,8,50:CLS#4:GDraw
1380 END DEFine
```

Note: Frame Change Check




```
1382 DEFine PROCedure FrNum(fr%,fi%,fx%,fy%)
1383 INK#6,CP(fi%):CURSOR#6,fx%,fy%:PRINT#6,'-';FILL$('0',2-LEN(fr%));fr%;'+'
1384 END DEFine
```

Note: Frame Numbering displays Current and Total SPRITES

```
1386 DEFine PROCedure FrShow
1387 PAPER 0:CLS#3:nx=320 DIV cm:ny=192 DIV rm:max=nx*ny
1388 fx=2:fy=1:IF max>frt:max=frt
1389 FOR f=1 TO max
1390   FOR r=0 TO rm-1
1391     FOR c=0 TO cm-1
1392       IF FG(f,c,r)=255:pcol%=0:ELSE pcol%=CP(FG(f,c,r))
1393       BLOCK#3,1,1,fx+c,fy+r,pcol%
1394     END FOR c
1395   END FOR r
1396   fx=2+(f MOD nx)*cm:fy=1+(f DIV nx)*rm
1397 END FOR f
1398 END DEFine
```

Note: -00+ Frame Option for Display of **SPRITE** Grp

```
1400 DEFine PROCedure FrNEW
1401 INK#6,CP(7):FrMes 6,29,16:IF GAns=0:INK#6,CP(5):RETurn
1402 IF fr=frt:frt=frt+1:fr=fr+1:FCLS fr
1403 IF fr<frt:FOR f=frt TO fr STEP -1:FChg 4:END FOR f:frt=frt+1:FCLS fr
1404 FrNum fr,7,8,50:FrNum frt,5,56,50:INK#6,CP(5)
1405 CLS#3:cs=0:cn=cm:rs=0:rn=rm:xm=cm-1:ym=rm-1:fed=1:ch=3:GDraw:EMove
1406 END DEFine
```


```
1408 DEFine PROCedure FrCOPY
1409 INK#6,CP(7):FrMes 6,29,26:FrNum fr,5,8,50:f2=fr:fed=3
1410 REPeat Copy_LP
1411   FrNum f2,7,56,50:GDraw
1412   k=CODE(INKEY$(-1))
1413   SELect ON k
1414     =45, 95:IF f2>1  :f2=f2-1
1415     =43, 61:IF f2<frt:f2=f2+1
1416     =78,110,32:EXIT Copy_LP
1417     =89,121,10:FChg 6:fr=f2:GDraw:fed=1:EXIT Copy_LP
1418   END SELect
1419 END REPeat Copy_LP
1420 FrNum fr,7,8,50:FrNum frt,5,56,50
1421 INK#6,CP(5):EMove:cs=0:cn=cm:rs=0:rn=rm:EMove
1422 END DEFine
```


```
1424 DEFine PROCedure FrDEL
1425 BLOCK#6,42,10,29,36,0:INK#6,CP(7):FrMes 6,29,36:IF GAns=0:INK#6,CP(5):RETurn
1426 IF frt=1:fr=1:FCLS :frt=0:fr=0
1427 IF frt>1 AND fr=frt:FCLS frt:frt=frt-1:fr=frt
1428 IF frt>1 AND fr<frt:FOR f=fr TO frt:FChg 3:END FOR f:FCLS frt:frt=frt-1
1429 FrNum fr,7,8,50:FrNum frt,5,56,50:INK#6,CP(5)
1430 CLS#3:cs=0:cn=cm:rs=0:rn=rm:ch=3:GDraw:EMove
1431 END DEFine
```


```
1433 DEFine PROCedure FCLS(f%)
1434 FOR r=0 TO 63:FOR c=0 TO 63:FG(f%,r,c)=255:END FOR c:END FOR r
1435 END DEFine
```

```
1437 DEFine PROCedure FrSwap
1438 orub=rub:sic=ic:FrMes 8,38,151:BLOCK#8,14,9,70,152,CP(sic)
1439 REPeat P_lp
1440   k=CODE(INKEY$(-1))
1441   SELect ON k
1442    =44,46,60,62,9,253:oic=ic:PCol
1443    =10:IF oic=bg:EXIT P_lp:ELSE FChg 2:GDrawEXIT P_lp
1444    =32:oic=sic:EXIT P_l
1445   END SELect
1446 END REPeat P_lp
1447 INK#8,CP(5):CURSOR#8,38,151:PRINT#8,'eCol ':k=0
1448 rub=0:cur=0:sic=ic:PCol:BLOCK#8,14,9,70,152,CP(sic)
1449 END DEFine
```

Old Colour

New Colour

Note: First Select **New** Colour from **< >** Palette. Press **[R]** then Select **Old** [existing] Grid Colour
Press **Enter** and Colours are exchanged within Whole Grid or Resized Edit Area.

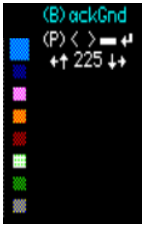
```
1451 DEFine PROCedure FChg(ck)
1452 FOR r=rs TO rs+rn-1
1453   FOR c=cs TO cs+cn-1
1454     IF ck=1 AND rub=0          :FG(fr,c,r)=ic          :REMark Set Edit Area  New Colour
1455     IF ck=1 AND rub=7          :FG(fr,c,r)=255         :REMark Set Edit Area  BackGnd
1456     IF ck=2 :IF FG(fr,c,r)=ic  :FG(fr,c,r)=sic         :REMark Swap Edit Area Colours
1457     IF ck=3 :FG(f,c,r)=FG(f+1,c,r)                     :REMark Delete Frame
1458     IF ck=4 :FG(f+1,c,r)=FG(f,c,r)                     :REMark Insert New Frame
1459     IF ck=5 AND FG(fr,c,r)=bg  :FG(fr,c,r)=255         :REMark Restore BackGnd
1460     IF ck=6 :FG(f2,c,r)=FG(fr,c,r)                     :REMark Copy Frame f2 - Frame fr
1461   END FOR c
1462 END FOR r
1463 END DEFine
```

Note: Frame Grp Array Change of Settings

Note: Press 'B' and Select Palette Colour with **< >** chevron keys shown as Enlarged Square on the Palette Bar.


```
1465 DEFine PROCedure GBGnd
1466 tic=ic:obg=bg:oic=ic:ic=bg:INK#8,CP(7)
1467 CURSOR#8,22,99:PRINT#8,'(B)< >   ':FrMes 8,60,99
1468 REPeat BGnd_lp
1469   PCol:k=CODE(INKEY$(-1))
1470   SELect ON k
1471     =44,46,60,62,9,253:oic=ic
1472     =10:IF ic=bg:EXIT BGnd_lp:ELSE bg=ic:FChg 5:GDraw:EXIT BGnd_lp
1473     =32:EXIT BGnd_lp
1474   END SELect
1475 END REPeat BGnd_lp
1476 INK#8,CP(5):oic=ic:ic=tic:PCol:CURSOR#8,22,99:PRINT#8,'(B)ackGnd ':FrChg
1477 END DEFine
```

Note: Only Palette Colours 8..5 are changeable. Select colour and change Number displayed with Cursor Keys. The New Colour is shown in Enlarged Square on the Palette Bar.




```
1479 DEFine PROCedure GPalette
1480 INK#8,CP(7):pn=8:IF ic<8:oic=ic:ic=8:PCol:END IF :tic=CP(ic):oic=ic
1481 CURSOR#8,22,110:PRINT#8,'(P)< >   ':FrMes 8,60,110
1482 CURSOR#8,22,119:PRINT#8,'◄▲  ▼►'
1483 REPeat Col_lp
1484   PCol:col%=CP(ic):CURSOR#8,43,119:PRINT#8,FILL$('0',3-LEN(col%));col%
1485   k=CODE(INKEY$(-1)):oic=ic
1486   SELect ON k
1487     =192:IF col%>    8:col%=col% -1:CP(ic)=col%
1488     =200:IF col%<255:col%=col% +1:CP(ic)=col%
1489     =208:IF col%<192:col%=col%+64:CP(ic)=col%
1490     =216:IF col%>  64:col%=col%-64:CP(ic)=col%
1491     = 32:CP(ic)=tic    :PCol:EXIT Col_lp
1492     = 10:CP(ic)=col%:PCol:FrChg:EXIT Col_lp
1493   END SELect
1494 END REPeat Col_lp
1495 BLOCK#8,60,20,20,109,0:INK#8,CP(5):CURSOR#8,22,110:PRINT#8,'(P)alette ':pn=0
1496 END DEFine
```

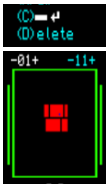
1498 REMark **SCREEN/TILE Generator**

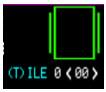
```
1500 DEFine PROCedure TGen
1501 SELect ON k
1502   =70,103:GFrame                                  :REMark (G)rid
1503   =75,107:IF frt>0:AGen                           :REMark (K)ey CTRL
1504   =49 TO 57:sn=k-48:IF tm>0   :Tile_MAP           :REMark (1...9)
1505   =67, 99:IF fr>0 AND tm<tmax :Tile_ADD:CURSOR#6,29,26:PRINT#6,'opy  '
1506   =68,100:IF tm>0             :Tile_DEL:CURSOR#6,29,36:PRINT#6,'elete'
1507   =66, 98:GBGnd:IF k=10       :Tile_MAP           :REMark (B)kGnd
1508   =60, 44:IF tn>1  :tn=tn-1   :TNum 7             :REMark <
1509   =62, 46:IF tn<tm :tn=tn+1   :TNum 7             :REMark >
1510   =84,116:IF tn>0             :Tile_Ass           :REMark (T)ile Asset
1511   =77,109:CLS#3:INK#3,CP(6) :Scn_MAP:Scn_Links    :REMark (M)AP Scn Links
1512   =35:IF obj=0:obj=1:CLSObj 7:ELSE obj=0:CLSObj 5 :REMark # Check Links
1513 END SELect
1514 END DEFine
```

```
1516 DEFine PROCedure Scn_Num(sn)
1517 INK CP(7):CURSOR 405,25:PRINT sn:cur=7
1518 END DEFine
```

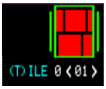
Note: SCREEN Num


```
1520 DEFine PROCedure Tile_ADD
1521 FrMes 6,29,26:IF frt=0 OR GAns=0:INK#6,CP(5):RETurn
1522 IF tn=0 OR tn=tm:tn=tn+1
1523 IF tn<tm:FOR td=tm TO tn STEP -1:Tile_Chg 2:END FOR td
1524 td=tn:Tile_Chg 1:tm=tm+1:TNum 7:INK#6,CP(5)
1525 END DEFine
```





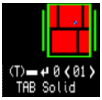
Copy SPRITE Frame to TILE Library




```
1527 DEFine PROCedure Tile_DEL
1528 BLOCK#6,30,10,32,36,0:FrMes 6,29,36:IF GAns=0:INK#6,CP(5):RETurn
1529 IF tn<tm:FOR td=tn TO tm:Tile_Chg 3:END FOR td:td=tm:Tile_Chg 4
1530 tm=tm-1:IF tn>tm:tn=tm:END IF :TNum 7:INK#6,CP(5)
1531 END DEFine
```

Delete TILE

```
1533 DEFine PROCedure TNum(ni)
1534 INK#8,CP(ni):CURSOR#8,46,4:PRINT#8,TAss(tn)
1535 TDraw 2      :CURSOR#8,64,4:PRINT#8,FILL$('0',2-LEN(tn))&tn
1536 END DEFine
```



Note: Tile Status & Library number

```
1538 DEFine PROCedure Tile_Chg(tck)
1539 FOR r=0 TO 15
1540   FOR c=0 TO 15
1541     IF tck=1:Tile(td,c,r)=FG(fr,cs+c,rs+r)    :REMark Add New Tile
1542     IF tck=2:Tile(td+1,c,r)=Tile(td,c,r)      :REMark Insert Tile Space
1543     IF tck=3:Tile(td,c,r)=Tile(td+1,c,r)      :REMark Cut Tile from List
1544     IF tck=4:Tile(td,c,r)=255                 :REMark CLS Tile
1545   END FOR c
1546 END FOR r
1547 END DEFine
```

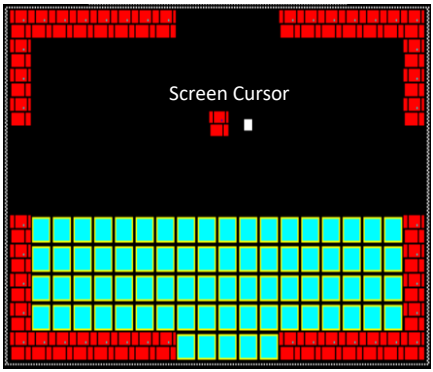

```
1549 DEFine PROCedure CLSObj(oi)
1550 INK#8,CP(oi):CURSOR#8,62,26:PRINT#8,'#'
1551 END DEFine
```

Note: ON/OFF **#** hash highlight for **SWITCH - SLIDE**

```
1553 DEFine PROCedure Tile_Ass
1554 CURSOR#8,4,4:PRINT#8,'(T)':FrMes 8,22,4:atn=TAss(tn):atn=TAss(tn)
1555 REPeat Ass_lp
1556   CURSOR#8,46,4:PRINT#8,atn:CURSOR#8,12,14:PRINT#8,'TAB ';TAS$(atn)
1557   k=CODE(INKEY$(-1))
1558   IF k=9 :atn=atn+1:IF atn>9:atn=0
1559   IF k=32:atn=TAss(tn):EXIT Ass_lp
1560   IF k=10:TAss(tn)=atn:EXIT Ass_lp
1561 END REPeat Ass_lp
1562 CURSOR#8,46,4:PRINT#8,atn:BLOCK#8,80,10,4,14,0
1563 INK#8,CP(5):CURSOR#8,4,4:PRINT#8,'(T)ILE '
1564 END DEFine
```

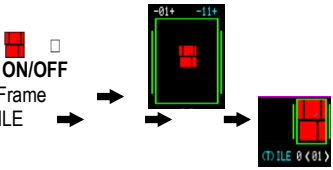
Note: 'TAB' through Assets :
Abort with Spacebar
Action with Enter


```
1566 DEFine PROCedure Tile_Chk
1567  tn=TScn(0,x,y)
1568 IF TAss(tn)=0:x=ox:y=oy                :REMark Solid Blocks Move
1569 IF TAss(tn)=1:x=x:y=y                  :REMark Floor No Action
1570 IF TAss(tn)=2:x=x:y=y  :Hazard         :REMark Hazard  WIP Score / Lives
1671 IF TAss(tn)=3:x=x:y=y  :Reward         :REMark Reward WIP Score / Lives
1572 IF TAss(tn)=4:x=x:y=y                  :REMark ???
1573 IF TAss(tn)=5:x=x:y=y                  :REMark ???
1574 IF TAss(tn)=6:x=x:y=y                  :REMark ???
1575 IF TAss(tn)=7:x=x:y=y                  :REMark ???
1576 IF TAss(tn)=8:x=x:y=y                  :REMark ???
1577 IF TAss(tn)=9:x=x:y=y                  :REMark ???
1578 END DEFine
```




```
1580 DEFine PROCedure Tile_MAP
1581 PAPER#3,CP(bg):CLS#3:Scn_Num sn
1582 FOR tr=0 TO 11
1583   FOR tc=0 TO 19
1584     TScn(0,tc,tr)=TScn(sn,tc,tr) MOD 64
1585     tn=TScn(0,tc,tr):IF tn>0:tx=2+tc*16:ty=1+tr*16:TDraw 0
1586   END FOR tc
1587 END FOR tr
1588 cur=0:rub=0:snp=sn:snw=TMap(snp,4):sne=TMap(snp,2):ec=0:wc=19
1589 END DEFine
```

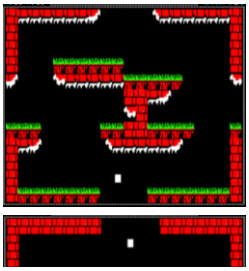
```
1591 DEFine PROCedure TDraw(ver)
1592 LOCal x,y
1593 IF ver=0:ch=3:w=1:h=1:x=tx:y=ty          :REMark Paint cur ON/OFF
1594 IF ver=1:ch=4:w=1:h=1:x=26:y=25:CLS#4    :REMark SPRITE Frame
1595 IF ver=2:ch=7:w=2:h=2:x=  0:y=  0:CLS#7  :REMark Library TILE
1596 FOR r=0 TO 15:FOR c=0 TO 15:PDraw:END FOR c:END FOR r
1597 END DEFine
```




```
1599 DEFine PROCedure PDraw
1600 pcol%=Tile(tn,c,r):IF pcol%=255:pcol%=bg
1601 BLOCK#ch,w,h,x+c*w,y+r*h,CP(pcol%)
1602 END DEFine
```

Note: Checks for Background. Fill Block at
TILE XY coordinates with TILE Colour

```
1604 DEFine PROCedure Scn_Switch
1605 ns=0:IF TScn(0,x,y)>0:Tile_Chk:END IF                :REMark Check IF Move Blocked
1606 IF y=0 :ns   =TMap(sn,1):IF ns>0:SEnt ns,1,11,18,11:x=c+1:y=10 :END IF
1607 IF x=xm:ns =TMap(sn,2):IF ns>0:SEnt ns,0, 1, 0,10  :x=1  :y=r+1  :END IF
1508 IF y=ym:ns =TMap(sn,3):IF ns>0:SEnt ns,1, 0,18, 0   :x=c+1:y=1   :END IF
1609 IF x=0 :ns   =TMap(sn,4):IF ns>0:SEnt ns,19,1,19,10  :x=18 :y=r+1:END IF
1610 IF ns>0:sn=ns:Tile_MAP
1611 END DEFine
```

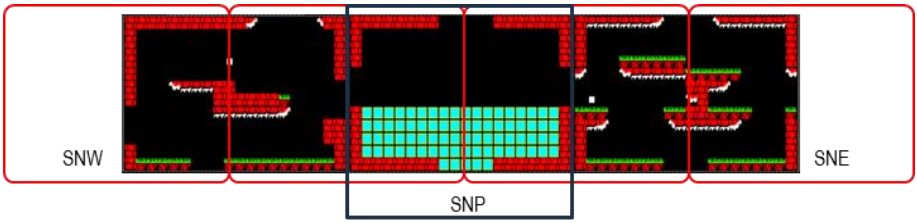


Note: See Maps for Links Set between Screens. Test by Pressing [#]

```
1613 DEFine PROCedure SEnt(ns,ax,by,cx,dy)
1614 FOR r=by TO dy:FOR c=ax TO cx:IF TScn(ns,c,r)=0:x=c:y=r:RETurn
1615 END DEFine
```

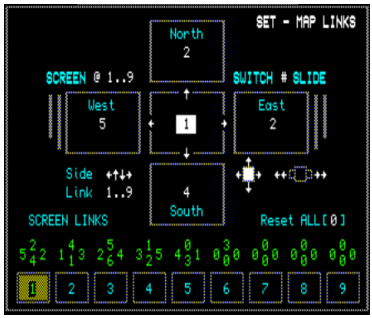
Note: This Option for Platform Games that use Sliding Screens and where Player's change in Levels are accomplished with a **Jump** action from pressing the **Spacebar**.

```
1617 DEFine PROCedure Scn_Slide
1618 IF TScn(0,x,y)>0:Tile_Chk:END IF                :REMark Check IF Move Blocked
1619 IF x+1>xm:PAN#3,-16:x=ox:PANEast :ec=ec+1:wc=wc+1:IF wc>19 :wc=19
1620 IF x-1<   0:PAN#3, 16:x=ox:PANWest:wc=wc-1:ec=ec-1:IF ec<0:ec=0
1621 END DEFine
```




```
1623 DEFine PROCedure PANEast
1624 IF ec>19:snp=sne:ec=0:wc=19:sne=TMap(snp,2):Scn_Num snp
1625 IF wc<19:sne=snp:ec=wc+1
1626 FOR tr=0 TO 11
1627   FOR tc=1 TO 19:TScn(0,tc-1,tr)=TScn(0,tc,tr):END FOR tc
1628   TScn(0,19,tr)=TScn(sne,ec,tr):tn=TScn(0,19,tr)
1629   IF tn>0:tx=2+19*16:ty=1+tr*16:TDraw 0
1630 END FOR tr
1631 END DEFine

1633 DEFine PROCedure PANWest
1634 IF wc<0:snp=snw:wc=19:ec=0:snw=TMap(snp,4):Scn_Num snp
1635 IF ec>0:snw=snp:wc=ec-1
1636 FOR tr=0 TO 11
1637    FOR tc=18 TO 0 STEP -1:TScn(0,tc+1,tr)=TScn(0,tc,tr):END FOR tc
1638    TScn(0,0,tr)=TScn(snw,wc,tr):tn=TScn(0,0,tr)
1639    IF tn>0:tx=2:ty=1+tr*16:TDraw 0
1640 END FOR tr
1641 END DEFine
```

```
1643 REMark MAPS SCREEN Links

1645 DEFine PROCedure Scn_MAP
1646 obg=bg:PAPER#3,0:CLS#3:QBold 3,7,224,4,'SET - MAP LINKS'
1647 QBold 3,5,34,40,'SCREEN':QBold 3,5,203,40,'SWITCH  SLIDE'
1648 FOR i=0 TO 1:BLOCK#3,68,40,52+i*152,54,CP(15):BLOCK#3,64,38,54+i*152,55,0
1649 FOR i=4 TO 5:BLOCK#3,3,i*7,5+i*8,56,CP(15):BLOCK#3,3,i*7,317-i*8,56,CP(15)
1650 FOR i=0 TO 2:BLOCK#3,68,40,128,8+i*46,CP(15) :BLOCK#3,64,38,130,9+i*46,0
1651 FOR i=1 TO 9
1652   mx=-27+i*35:BLOCK#3,30,20,mx,170,CP(15):BLOCK#3,28,18,mx+1,171,0
1653   CURSOR#3,-16+i*35,176:PRINT#3,i
1654 END FOR i
1655 RESTORE 1659:FOR i=1 TO 26:QPrnt
1656 BLOCK#3,11,10,212,102,CP(15):BLOCK#3, 9,8,213,103,0
1657 BLOCK#3,21, 6,254,104,CP(15):BLOCK#3,19,4,255,105,0
1658 BLOCK#3,11,10,259,102,CP(15):BLOCK#3, 9,8,260,103,0
1659 DATA 3,7,214,94,'↑',3,7,214,110,'↓',3,7,205,102,'←  →'
1660 DATA 3,7,240,102,'←  ←    →  →',3,7,246,40,'#'
1661 DATA 3,7,158,24,TMap(sn,1),3,7,158,114,TMap(sn,3)
1662 DATA 3,7, 82,70,TMap(sn,4),3,7,236, 70,TMap(sn,2)
1663 DATA 3,0,155,52,'   ',3,7,126,69,'←          →',3,0,155,90,'  '
1664 DATA 3,7,158,50,'↑',3,5,146, 12,'North',3,5,226,58,'East'
1665 DATA 3,7,158,88,'↓',3,5,146,126,'South',3,5,72,58,'West'
1666 DATA 3,7,77, 40,'@ 1..9',3,5,18,132,'SCREEN LINKS'
1667 DATA 3,5, 52,102,'Side' ,3,7,88,102,'←↑↓→'
1668 DATA 3,5, 52,114,'Link' ,3,7,88,114,'1..9'
1669 DATA 3,5,228,132,'Reset ALL[ ]',3,7,291,132,'0'
1670 BLOCK#8,40,10,38,26,0:FrMes 8,38,26:ScnTie:mx=158:my=70:sl=0
1671 END DEFine
```

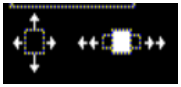


Map of SCREEN Connections

Note: Use @ and 1…9 to Select SCREEN. Use Cursor Keys to Select Side and 1..9 to set Link. A LINK will connect on the opposite side of SCREEN Chosen: ie WEST Side Links to the EAST Side of selected 1..9 SCREEN.

```
1671 DEFine PROCedure Scn_Links
1672 REPeat Scrn_lp
1673   PrntScn 7,7,0:PrntMap:k=CODE(INKEY$(-1)):PrntScn 0,0,7
1674   SELect ON k
1675     =   35 :IF lks=0:lks=1:ScnTie:ELSE lks=0:ScnTie
1676     =39,64 :mx=158:my=  70:sl=0                    :REMark @ Select Screen
1677     =192   :mx=  82:my=  70:sl=4:so=2              :REMark West  - East   Entry
1678     =200   :mx=236:my=  70:sl=2:so=4               :REMark East  - West  Entry
1679     =208   :mx=158:my=  24:sl=1:so=3               :REMark North  - South Entry
1680     =216   :mx=158:my=114:sl=3:so=1               :REMark South  - North Entry
1681     =  48   :DIM TMap(9,4):RESTORE 1661:FOR i=1 TO 4:QPrnt
1682     =49 TO 57 :IF sl=0
1683                 PrntCLS:sn=k-48:RESTORE 1661:FOR i=1 TO 4:QPrnt
1684               ELSE
1685                 TMap(sn,sl)=k-48:TMap(k-48,so)=sn
1686               END IF
1687   =32,10:bg=obg:PAPER#3,CP(bg):CLS#3:EXIT Scrn_lp          Note: Bg BackGnd obg Old BGnd Colour
1688   END SELect
1689 END REPeat Scrn_lp
1690 INK#8,CP(5):CURSOR#8,38,26:PRINT#8,'aps[#]':obj=0:Tile_MAP
1691 END DEFine
```

```
1695 DEFine PROCedure ScnTie
1696 IF lks=0:BLOCK#3,9,8,213,103,CP(7):BLOCK#3,9,8,260,103,0
1697 IF lks=1:BLOCK#3,9,8,213,103,0:BLOCK#3,9,8,260,103,CP(7)
1698 END DEFine
```



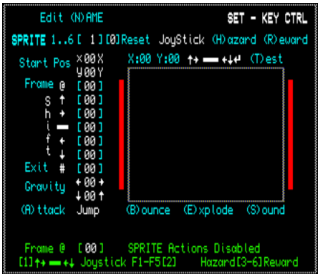
```
1700 DEFine PROCedure PrntScn(bink,sink,pink)
1701 BLOCK#3,18,12,mx-6,my-1,CP(bink):STRIP#3,CP(sink):INK#3,CP(pink)
1702 CURSOR#3,mx,my:IF sl=0:PRINT#3,sn:ELSE PRINT#3,TMap(sn,sl)
1703 END DEFine
```


```
1705 DEFine PROCedure PrntMap
1706 STRIP#3,0:INK#3,CP(4)
1707 BLOCK#3,26,16,-25+sn*35,172,240:CURSOR#3,-16+sn*35,176:PRINT#3,sn
1708 FOR i=1 TO 9:CURSOR#3,-25+i*35,154:PRINT#3,TMap(i,4);'  ';TMap(i,2)
1709 FOR i=1 TO 9:CURSOR#3,-16+i*35,148:PRINT#3,TMap(i,1)
1710 FOR i=1 TO 9:CURSOR#3,-16+i*35,158:PRINT#3,TMap(i,3)
1711 STRIP#3,2:INK#3,0
1712 END DEFine
```

```
1714 DEFine PROCedure PrntCLS
1715 BLOCK#3,26,16,-25+sn*35,172,0
1716 STRIP#3,0:INK#3,CP(5):CURSOR#3,-16+sn*35,176:PRINT#3,sn
1717 END DEFine
```


```
1719 REMark Key ConTRL Action Generator

1721 DEFine PROCedure Init_Agen
1722 obg=bg:PAPER#3,0:CLS#3:INK#3,CP(7): BLOCK#8,38,10,42,55,0:FrMes 8,38,55
1723 str$='[   ]':INK#3,CP(5):FOR i=1 TO 7:CURSOR#3,73,44+i*10:PRINT#3,str$
1724 str$='Shift':INK#3,CP(7):FOR i=1 TO 5:CURSOR#3,41,56+i*9:PRINT#3,str$(i)
1725 OPEN#12,scr_:WINDOW#12,164,98,gx+222,gy+81:BORDER#12,1,CP(15)
1726 BLOCK#3,5,80,119,54,CP(2):BLOCK#3,5,80,296,54,CP(2)
1727 QBold 3,5,6,20,'SPRITE':QBold 3,7,232,4,'SET - KEY CTRL'
1728 RESTORE 1741:FOR i=1 TO 28:QPrnt:END FOR i
1739 DATA 3,5,36,4,'Edit (N)AME',3,5,47,20,'1..6',3,5,73,20,'[   ][ ]Reset'
1730 DATA 3,7,109,20,'0',3,5,14,37,'Start Pos',3,5,20,52,'Frame',3,7,56,54,'@'
1731 DATA 3,7,109,20,'0',3,7,74,33,'x',3,7,96,34,'X',3,7,74,42,'y',3,7,96,44,'Y'
1732 DATA 3,7,56,63,'↑',3,7,56,73,'→',3,7,56,83,'←',3,7,56,93,'↓',3,7,80,124,'OFF'
1733 DATA 3,5,20,113,'Exit',3,7,56,114,'#',3,5,14,144,'(A)ttack'
1734 DATA 3,5,20,127,'Gravity',3,7,73,123,'←  →',3,7,73,133, '↑  ↓
1735 DATA 3,5,128,34,'X:00 Y:00          (T)est',3,7,190,34,' ↑ →   ← ↓ ← '
1736 DATA 3,4,20,172,'Frame @  [  ]   SPRITE Actions Disabled',3,7,83,172,'00'
1737 DATA 3,4,12,182,'[1] ↑→   ← ↓  Joystick F1-F5[2]   Hazard[3-6]Reward'
1738 BLOCK#3,12,3,54,107,CP(7):BLOCK#3,12,3,46,186,CP(4)
1739 BLOCK#3,16,3,207,38,CP(7):BLOCK#3, 2,4,244,36,CP(7)
1740 Actn$(0(='Jump':Actn(1)='Fire':Snum=10:LNum=4:osn=sn
1741 ka=1:kb=0:kx=0:kxm=176:ky=0:kym=96:k7=0:k8=0:k9=0:k10=0:k11=0
1742 END DEFine
```

Note: Screen Layout Action Generator



Note: Settings for Action SPRITE 1 & 2 Player Directly Controlled and Program Controlled 3 to 6 [Gravity Controlled].

```
1744 DEFine PROCedure AGen
1745 IF fr<1:BMLoad:END IF :Init_AGen:Set_Score:INK#3,CP(7):SP_Chg:SP_Actn
1746 REPeat CTRL_LP
1747   SP_Type:SP_Actn:IF kb>3 AND kb<16:SP_Prnt ka,kb                    Note: Action Generator Menu
1748   k$=(INKEY$(-1)):k=CODE(k$)
1749   SELect ON k
1750     =84,116    :IF SK(ka,6)>0:CTRL_Test:CURSOR#3,247,34:PRINT#3,'(T)est'
1751     =78,110    :Chg_Title                                          :REMark Edit (N)ame
1752     =48        :FOR i=1 TO 20:SK(ka,i)=0:END FOR i:SP_Chg          :REMark [0] Reset
1753     =49 TO 54 :ka=k-48:SP_Chg                                      :REMark Sprite 1..6
1754     =67,99     :IF ka<3:SK(ka,1)=1                                 :REMark (C)onTRL Sprite
1755     =72,104    :IF ka>2:SK(ka,1)=2:SP_Score                        :REMark (H)azard Sprite
1756     =82,114    :IF ka>2:SK(ka,1)=3:SP_Score                        :REMark (R)eward Sprite
1757     =45, 95    :IF fr>  0:fr=fr-1:FrChg                            :REMark -  Frame
1758     =43, 61    :IF fr<frt:fr=fr+1:FrChg                            :REMark + Frame
1759     =120       :IF SK(ka,4)>0    :SK(ka,4)=SK(ka,4)-1:kb=4         :REMark x  - Pos
1760     =88        :IF SK(ka,4)<kxm/4:SK(ka,4)=SK(ka,4)+1:kb=4         :REMark X + Pos
1761     =121       :IF SK(ka,5)>0    :SK(ka,5)=SK(ka,5)-1:kb=5         :REMark y  - Pos
1762     =89        :IF SK(ka,5)<kym/4:SK(ka,5)=SK(ka,5)+1:kb=5         :REMark Y + Pos
1763     =39, 64    :SK(ka, 6)=fr:kb= 6                                 :REMark @ Default Frame
1764     =212       :IF ka<6:SK(ka, 7)=fr:kb= 7                         :REMark Shift UP  Sprite
1765     =204       :IF ka<6:SK(ka, 8)=fr:kb= 8                         :REMark Shift Right Facing Sprite
1766     =196       :IF ka<6:SK(ka, 9)=fr:kb= 9                         :REMark Shift Left  Facing Sprite
1767     =220       :IF ka<6:SK(ka,10)=fr:kb=10                         :REMark Shift Down Sprite
1768     =252       :IF ka<6:SK(ka,11)=fr:kb=11                         :REMark Shift SB Jump/Fire Sprite
1769     =35        :IF ka<6:SK(ka,12)=fr:kb=12 :                       :REMark # Clear
1770     =192       :IF ka<6 AND SK(ka,13)>-8:SK(ka,13)=SK(ka,13)-1:kb=13  :REMark ←
1771     =200       :IF ka<6 AND SK(ka,13)< 8:SK(ka,13)=SK(ka,13)+1:kb=13  :REMark →
1772     =208       :IF ka<6 AND SK(ka,14)< 8:SK(ka,14)=SK(ka,14)+1:kb=14  :REMark ↑
1773     =216       :IF ka<6 AND SK(ka,14)>-8:SK(ka,14)=SK(ka,14)-1:kb=14  :REMark ↓
1774     =65, 97    :IF ka<3;kb=15:SP_Toggle                            :REMark (A)ction
1775     =66, 98    :IF ka<6:kb=16:SP_Toggle                            :REMark (B)ounce
1776     =69,101    :IF ka<6:SK(ka,16)=2                                :REMark (E)xplode
1777     =83,115    :kb=18:SP_Toggle                                    :REMark (S)ound
1778     =10,32:CLOSE#12:bg=obg:PAPER#3,CP(bg):CLS#5:EXIT CTRL_LP
1779   END SELect
1780 END REPeat CTRL_LP
1781 INK#8,CP(5):CURSOR#8,38,55:PRINT#8,'ey CTRL':CLS#3:sn=osn:Tile_MAP
1782 END DEFine

1784 DEFine PROCedure SP_Type
1785 INK#3,CP(5):CURSOR#3,160,20:PRINT#3,'JoyStick (H)azard (R)eward'
1786 CURSOR#3,124,144:PRINT#3,'(B)ounce  (E)xplode  (S)ound':INK#3,CP(7)
1787 IF ka=1 OR ka=2:CURSOR#3,160, 20:PRINT#3,'JoyStick':SK(1,1)=1:SK(2,1)=1
1788 IF SK(ka,  1)=2 :CURSOR#3,214, 20:PRINT#3,'(H)azard'
1789 IF SK(ka,  1)=3 :CURSOR#3,268, 20:PRINT#3,'(R)eward'
1790 IF SK(ka,16)=1 :CURSOR#3,124,144:PRINT#3,'(B)ounce'
1791 IF SK(ka,16)=2 :CURSOR#3,184,144:PRINT#3,'(E)xplode'
1792 IF SK(ka,18)=1 :CURSOR#3,250,144:PRINT#3,'(S)ound'
1793 END DEFine

1795 DEFine PROCedure SP_Actn
1796 CURSOR#3,74,144:PRINT#3,Actn$(SK(ka,15))
1797 END DEFine                          Note: Press 'A' to Toggle (A)ction
```

= 0 = 1

```
1799 DEFine PROCedure SP_Prnt(ka,kb)
1800 CURSOR#3,82,-6+kb*10:PRINT#3,FILL$('0',2-LEN(SK(ka,kb)))&SK(ka,kb)
1801 END DEFine

1803 DEFine PROCedure Chg_Title
1804 CLS#5:EditName 5,0,20,12,13,RGN$:Snum=0:LNum=4:Init_Title
1805 END DEFine

1807 DEFine PROCedure SP_Chg
1808 INK#3,CP(7):CURSOR#3,88,20:PRINT#3,ka:SP_Type:SP_Actn
1809 FOR i= 4 TO 14:SP_Prnt ka,i:END FOR i:kb=4
1810 END DEFine

1812 DEFine PROCedure SP_Score
1813 BLOCK#5,100,30,0,0,0:Snum=0:LNum=4
1814 CURSOR#5,46,6:PRINT#5,'Set ↓  ↑':CURSOR#5,58,20:PRINT#5,'Set ←  →'
1815 REPeat Score_lp
1816  CURSOR#5,80, 6:PRINT#5,FILL$(' ',4-LEN(Snum))&Snum
1817  CURSOR#5,92,20:PRINT#5,FILL$(' ',2-LEN(LNum))&LNum:k=CODE(INKEY$(1))
1818  SELect ON k
1819   =192:IF LNum>-4:LNum=LNum-1
1820   =200:IF LNum< 4:LNum=LNum+1
1821   =208:IF Snum< 100:Snum=Snum+10
1822   =216:IF Snum>-100:Snum=Snum-10
1823   = 32:Init_Title:EXIT Score_lp
1824   = 10:SK(ka,2)=Snum:SK(ka,3)=LNum:Set_Score:EXIT Score_lp
1825   END SELect
1826 END REPeat Score_lp
1827 END DEFine

1829 DEFine PROCedure Set_Score
1830 CLS#5:QTitle 5,1,1,1,RGN$:Snum=0:LNum=4
1831 QBold 5,7,118,5,'SCORE':QBold 5,7,118,19,'LIVES': Chg_Score
1832 END DEFine

1834 DEFine PROCedure Chg_Score
1835 CURSOR#5,160,1:CSIZE#5,1,1
1836 IF Snum>-1 AND Snum<9999:PRINT#5,FILL$('0',4-LEN(Snum))&Snum:ELSE Game_End
1837 CURSOR#5,160,1:CSIZE#5,0,0:IF LNum<1:Game_End:END IF: IF LNum>4:RETurn
1838 F BLOCK#5,32,8,160,20,0:OR i=1 TO LNum:CURSOR#5,152+i*8,20:PRINT#5,'#';
1839 END DEFine

1841 DEFine PROCedure SP_Toggle
1842 IF SK(ka,kb)=0:SK(ka,kb)=1:ELSE SK(ka,kb)=0:END IF :kb=4
1843 END DEFine

1845 DEFine PROCedure SP_BEEP(bs)
1846 SELect ON bs
1847   =1:BEEP 0,1,2,3,4,5,6,7      :REMark d,p,h,t,s,w,f,r
1848   =2:BEEP 4000,0,200,2,3,0,0,0
1849   =3:BEEP 15000,12,48,30,-8,15,15,15
1850   =4:BEEP 5000,32,12,8,0,0,0,0
1851 END SELect
1852 END DEFine
```

```
1854 DEFine PROCedure CTRL_Test
1855 INK#3,CP(7):CURSOR#3,242,34:PRINT#3,'(Esc)  ':Test_Scn
1856 chk=0:Snum=0::LNum=4:Score_chg:kb=6:tsz=4:tw=8:th=8:sw=cm:sh=rm
1867 sx=SK(ka,4)*4:sy=SK(ka,5)*4:osx=sx:osy=sy:kxm=160:kym=96
1858 xx=SK(ka,13):IF xx>4:xx=4:END IF :IF xx<-4:xx=-4:END IF
1859 yy=SK(ka,14):IF yy>4:yy=4:END IF :IF yy<-4:yy=-4:END IF
1860 IF SK(ka,1)=1 OR SK(ka,1)=2:ix=2:xx=0:iy=1:yy=0 :END IF
1861 IF SK(ka,15)=1:Sk(ka,16)=0:Sp_Type:END IF
1862 REPeat Test_lp
1863   SP_Draw sx,sy:PAUSE 5:SP_Draw osx,osy
1864   IF ka=1 OR ka=2:Ctrl_Moves:END IF :Chk_Move
1865   IF KEYROW(1)=8:CLS#12:INK#3,CP(5):EXIT Test_lp
1866 END REPeat T_lp
1867 END DEFine
```

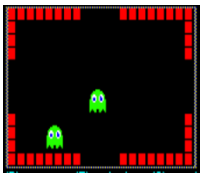
Note Speed <= Tile width
Speed <= Tile height

Note [Esc Key] KEYROW(1)=8

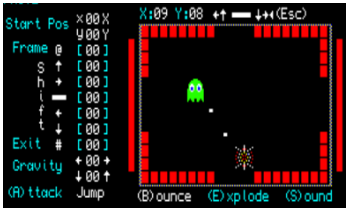
```
1869 DEFine PROCedure SP_Draw(px,py)
1870 OVER#12,-1:ch=12:fr=SK(ka,kb)
1871 FOR r=rs TO rs+rn-1:FOR c=cs TO cs+cn-1:PBit:END FOR c:END FOR r
1872 OVER#12,0
1873 END DEFine

1875 DEFine PROCedure PBit
1876 IF FG(fr,c,r)=255:pcol%=CP(bg):ELSE pcol%=CP(FG(fr,c,r))
1877 BLOCK#ch,1,1,px+c,py+r,pcol%
1878 END DEFine

1880 DEFine PROCedure Ctrl_Moves
1881 k=KEYROW(1):tmx=xx:tmy=yy
1882 SELect ON k
1883   =  32:IF SK(ka,15)=0:SP_Jump:ELSE SP_Fire
1884   =128:yy=yy+iy          :cr=  0:rf=+1   :REMark Down
1885   =144:yy=yy+iy:xx=xx+ix:cf=+1:rf=+1   :REMark Down/Right
1886   =  16:xx=xx+ix         :cf=+1:rf=  0   :REMark Right
1887   =  20:yy=yy-iy:xx=xx+ix :cf=+1:rf= -1   :REMark Up/Right
1888   =   4:yy=yy-iy         :cf=  0:rf= -1   :REMark Up
1889   =   6:yy=yy-iy:xx=xx-ix :cf= -1:rf= -1   :REMark UP/Left
1890   =   2:xx=xx-ix         :cf= -1:rf=  0   :REMark Left
1891   =130:yy=yy+iy:xx=xx-ix :cf= -1:rf=+1   :REMark Down/Left
1892 END SELect
1893 IF xx>4 OR xx<-4:xx=tmx:END IF :IF yy>4 OR yy<-4:yy=tmy:END IF
1894 END DEFine
```

Note: ▬ SpaceBar Actions

Note Max Set for CTRL Test




```
1896 DEFine PROCedure SP_Jump
1897 yy=-th:Chk_Move:SP_Draw sx,sy:PAUSE 5:SP_Draw osx,osy:yy=-ABS(xx)
1898         Chk_Move:SP_Draw sx,sy:PAUSE 5:SP_Draw osx,osy:yy=+th/2
1899 END DEF
```

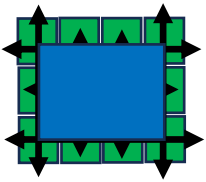



```
1901  DEFine PROCedure SP_Fire
1902 xc=osx+cm/2+(2+cm/2)*cf:yr=osy+rm/2+(1+rm/2)*rf
1903 tsx=osx:tsy=osy:SK(ka,16)=2:SP_Draw osx,osy
1904 FOR i=1 TO 8
1905   BLOCK#12,3,2,xc,yr,CP(7):PAUSE 2:BLOCK#12,3,2,xc,yr,0
1906   xc=xc+4*cf:yr=yr+4*rf:osx=xc:osy=yr:Chk_Move
1907 END FOR i
1908 osx=tsx:osy=tsy:SK(ka,16)=1:SP_Draw osx,osy
1909 END DEFine
```

```
1911 DEFine PROCedure Chk_Move
1912 IF osx+xx<   0      :osx=160-sw:sy=40:PAUSE 10
1913 IF osx+xx>160-sw :osx=   0      :sy=40:PAUSE 10
1914 IF osy+yy<   0      :osy=  96-sh :sx=70:PAUSE 10
1915 IF osy+yy>  96-sh  :osy=   0      :sx=70:PAUSE 10
1916 tc=osx DIV tw:tr=(osy+yy) DIV th:Chk_Hit:IF chk=1:YChg:chk=0
1917 tc=(osx+xx) DIV tw:tr=osy DIV th:Chk_Hit:IF chk=1:XChg:chk=0
1918 CURSOR#3,140,34:PRINT#3,FILL$('0',2-LEN(tc));tc
1919 CURSOR#3,170,34:PRINT#3,FILL$('0',2-LEN(tr));tr
1920 sx=osx+xx:sy=osy+yy:osx=sx:osy=sy
1921 END DEFine
```

```
1923 DEFine FunCedure Chk_Hit
1924 FOR row=0 TO sh DIV th
1925   FOR col=0 TO sw DIV tw
1926     IF TTile(tc+col,tr+row)>0:chk=1:RETurn
1927   END FOR col
1928 END FOR row
1929 END DEFine
```

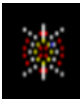
Note: sh screen th Tile height
sw Screen tw Tile width



```
1931 DEFine PROCedure XChg
1932 IF SK(ka,16)=0:xx=0:yy=0     :IF SK(ka,18)=1:SP_BEEP 2          :REMark Stop
1933 IF SK(ka,16)=1:xx=-xx        :IF SK(ka,18)=1:SP_BEEP 2          :REMark Bounce
1934 IF SK(ka,16)=2:xx=0  :SP_Ex:IF SK(ka,1)<5:xx=0:yy=0             :REMark Explode
1935 END DEFine
```

```
1937 DEFine PROCedure YChg
1938 IF SK(ka,16)=0:xx=0:yy=0     :IF SK(ka,18)=1:SP_BEEP 2
1939 IF SK(ka,16)=1:yy=-yy        :IF SK(ka,18)=1:SP_BEEP 2
1940 IF SK(ka,16)=2:yy=0  :SP_Ex:IF SK(ka,1)<5:xx=0:yy=0
1941 END DEFine
```

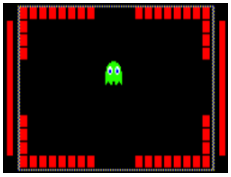
```
1943 DEFine PROCedure SP_Ex
1944 IF SK(ka,18)=1:SP_BEEP 3
1945 kb=12:FOR i=1 TO 6:SP_Draw osx,osy:PAUSE 5:SP_Draw osx,osy:PAUSE 5
1946 sx=SK(ka,4)*tw:sy=SK(ka,5)*th:xx=0:yy=0:kb=6:BEEP
1947 Snum=Snum+SK(ka,2)::LNum=LNum+SK(ka,3):Chg_Score
1948 END DEFine
```




```
1950 DEFine PROCedure Game_End
1951 CLS#12:INK#12,7:CSIZE#12,1,1
1952 CURSOR#12,48,20:PRINT#12,'GAME END':CSIZE#12,0,0:PAUSE:chk=1
1953 END DEFine
```

```
1955 DEFine PROCedure Test_Scn
1956 LOCal tc,tr :DIM TTile(20,12)
1957 FOR tr=0 TO 11
1958   FOR tc=0 TO 19
1959     IF tr= 0 OR tr =11:IF tc<8 OR tc>11:TTile(tc,tr)=1:BLOCK#12,6,7,tc*8,tr*8,2
1960     IF tc=0 OR tc =19:IF tr<4 OR tr>  7:TTile(tc,tr)=1:BLOCK#12,6,7,tc*8,tr*8,2
1961   END FOR tc
1962 END FOR tr
1963 END DEFine
```

Note: Test Screen Border Tile display



```
2000 REMark File Management

2002 DEFine PROCedure BMLoad
2003 chk=0:eck=0:SelDrv 2,'Load  ':INK CP(7):sf%=1:ft%=0:fm%=20
2004 CURSOR#5,6,20:PRINT#5,'Checking...':CLS#5,4:PAUSE 20:DirList:Dir_bmp
2005 IF ft%<1 :CURSOR#5,6,11:PRINT#5,'No Files Found...':CLS#5,4:PAUSE 30:RETurn
2006 FrMes 5,78,20:CURSOR#5,6,20:PRINT#5,'Select ¾ ¿ ':INK#5,CP(5)
2007 REPeat FSel_lp
2008  CURSOR#5,6,11:PRINT#5,'Load  ';File$(sf%):CLS#5,4
2009  k=CODE(INKEY$(-1))
2010  SELect ON k
2011    =208:sf%=sf%-1:IF sf%<1:sf%=ft%          :REMark Up
2012    =216:sf%=sf%+1:IF sf%>ft%:sf%=1          :REMark Dn
2013    =110,78,32:RETurn                        :REMark Y/N Spacebar
2014    =121,89,10:EXIT FSel_lp                  :REMark Y/N Enter
2015  END SELect
2016 END REPeat FSel_lp
2017 BLOCK#5,190,20,2,10,0:CURSOR#5,6,11:PRINT#5,'Loading.':CLS#5,4:pm$=''
2018 OPEN_IN#9,drv$(dn%)&File$(sf%):pm$=INKEY$(#9)&INKEY$(#9)&INKEY$(#9)
2019 IF pm$='QL8' OR pm$='PAL'
2020    bm=CODE(INKEY$(#9)):cm=CODE(INKEY$(#9)):rm=CODE(INKEY$(#9))
2021    bg=CODE(INKEY$(#9)) :sz=CODE(INKEY$(#9))
2022    IF sz=0:frt=bm:base=ALCHP(  8+frt*cm*rm):ptr=base+8
2023    IF sz=1:frt=bm:base=ALCHP(8+8+frt*cm*rm):ptr=base+16
2024    IF sz=2:tm=bm:base=ALCHP(8+8+36+2160+tm*256)
2025    IF sz=3:tm=bm:base=ALCHP(8+8+36+2160+200+tm*256+bm)
2026    LBYTES drv$(dn%)&File$(sf%),base
2027    IF sz=2:SnLoad base+16,base+52:ptr=base+2212
2028    IF sz=3:SnLoad base+16,base+52:AnLoad base+2212:ptr=base+2412
2029    FOR i=8 TO 15:BLOCK#8,8,6,6,36+10*i,CP(i)
2030    OVER#5,1:FrLoad bm,cm,rm,ptr:OVER#5,0
2031    IF sz<2:fr=1:GFrame:ELSE tn=1:sn=1:GTile
2032 ELSE
2033    CURSOR#5,6,11:PRINT#5,'File ERROR':PAUSE 30
2034 END IF
2035 CLOSE#9:RECHP base:IF sz<2:PFile$=File$(sf%):ELSE TFile$=File$(sf%)
2036 END DEFine

2038 DEFine PROCedure SnLoad(ptr1,ptr2)
2039 FOR sn=1 TO 9
2040   FOR sd=1 TO 4:TMap(sn,sd)=PEEK(ptr1):ptr1=ptr1+1:END FOR sd
2041   FOR tr=0 TO 11
2042     FOR tc=0 TO 19:TScn(sn,tc,tr)=PEEK(ptr2):ptr2=ptr2+1:END FOR tc
2043   END FOR tr
2044 END FOR sn
2045 END DEFine

2047 DEFine PROCedure AnLoad(ptr4)
2048 FOR i=1 TO 13:RGN$(i)=CHR$(PEEK(ptr4)):ptr4=ptr4+1:END FOR i:ptr4=ptr4+1
2049 FOR ka=1 TO 9
2050    FOR kb=1 TO 20:SK(ka,kb)=PEEK(ptr4):ptr4=ptr4+1:END FOR kb
2051 END FOR ka
2052 END DEFine
```

```
2054 DEFine PROCedure FrLoad(bm%,cm%,rm%,ptr)
2055 IF sz<2:DIM FG(32,64,64)
2056 IF sz>0:FOR i=8 TO 15:CP(i)=PEEK(base+i)
2057 FOR f=1 TO bm%
2058   IF f<48:CURSOR#5,48+f*3,11:PRINT#5,'.':PAUSE 1
2059   FOR r=0 TO rm%-1
2060     FOR c=0 TO cm%-1
2061       IF sz<2:FG(f,c,r)=PEEK(ptr) :END IF
2062       IF sz>1:Tile(f,c,r)=PEEK(ptr):END IF
2063       IF frt= 0:FG(f,c,r)=PEEK(ptr) :END IF
2064       ptr=ptr+1
2065     END FOR c
2066   END FOR r
2067 END FOR f
2068 IF sz>1:FOR f=1 TO bm%:TAss(f)=PEEK(ptr):ptr=ptr+1:END FOR f
2069 END DEFine
```

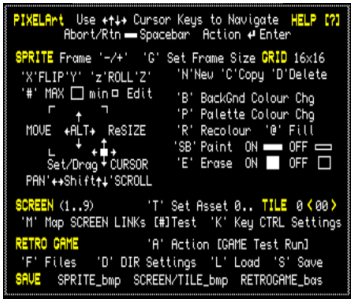

```
2071 DEFine PROCedure BMSave
2072 IF frt>0 OR tm>0:chk=0:eck=0:SelDrv 2,'Save  ':ELSE RETurn
2073 IF sz<2:SFile$=PFile$:ELSE SFile$=TFile$ :END IF
2074 IF SFile$='':SFile$='QBDefault_bmp':END IF
2075 IF LEN(SFile$)>20:SFile$=SFile$(1 TO 16)&'_bmp'
2076 INK CP(7):CURSOR#5,6,20:PRINT#5,'Action      (E)dit':FrMes 5,48,20
2077 REPeat chk_lp
2078   CURSOR#5,42,11:PRINT#5,SFile$;FILL$(' ',20-LEN(SFile$))
2079   k=CODE(INKEY$(-1))
2080   SELect ON k=101,69:EditName 5,2,42,11,20,SFile$
2081   SELect ON k=10:DirList:Dir_chk:EXIT chk_lp
2082   SELect ON k=32:BLOCK#5,190,20,2,10,0:RETurn
2083 END  REPeat chk_lp
2084 IF eck=1
2085   CURSOR#5,6,11:PRINT#5,'DEVICE ERROR...':CLS#5,4:PAUSE 30:eck=0:RETurn
2086 END IF
2087 IF chk=1
2088   CURSOR#5,6,20:PRINT#5,'Overwrite':CLS#5,4:CLS#5,2
2089   FrMes 5,72,20:IF GAns=0:RETurn
2090 END IF
```


```
2091 DELETE drv$(dn%)&SFile$:BLOCK#5,190,20,2,10,0
2092 INK#5,CP(5):CURSOR#5,6,11:PRINT#5,'Saving.':CLS#5,4
2093 IF sz=0:bm=frt:mlth= 8+bm*cm*rm                  :addr=ALCHP(mlth):ptr=addr+8
2094 IF sz=1:bm=frt:mlth=16+bm*cm*rm                  :addr=ALCHP(mlth):ptr=addr+16
2095 IF sz=2:bm=tm :mlth=52+2160+bm*256               :addr=ALCHP(mlth):cm=16:rm=16
2096 IF sz=3:bm=tm :mlth=52+2160+200+bm*256+bm:addr=ALCHP(mlth):cm=16:rm=16
2097 FOR i=0 TO 2:POKE addr+i,CODE(pm$(i+1)):END FOR i
2098 POKE addr+3,bm:POKE addr+4,cm:POKE addr+5,rm:POKE addr+6,bg:POKE addr+7,sz
2099 IF sz=2:SnSave addr+16,addr+52:ptr=addr+2212
2100 IF sz=3:SnSave addr+16,addr+52:AnSave addr+2212:ptr=addr+2412
2101 OVER#5,1:FrSave bm,cm,cm,ptr:OVER#5,0
2102 SBYTES drv$(dn%)&SFile$,addr,mlth:RECHP addr
2103 END DEFine
```

```
2105 DEFine PROCedure SnSave(ptr2,ptr3)
2106 FOR sn=1 TO 9
2107   FOR sd=1 TO 4:POKE ptr2,TMap(sn,sd):ptr2=ptr2+1:END FOR sd
2108   FOR tr=0 TO 11
2109       FOR tc=0 TO 19:POKE ptr3,TScn(sn,tc,tr):ptr3=ptr3+1:END FOR tc
2110   END FOR tr
2111 END FOR sn
2112 END DEFine

2114 DEFine PROCedure AnSave (ptr4)
2115 FOR i=1 TO 13:POKE ptr4,CODE(RGN$(i)):ptr4=ptr4+1:END FOR i:ptr4=ptr4+1
2116 FOR ka=1 TO 9
2117   FOR kb=1 TO 20:POKE ptr4,SK(ka,kb):ptr4=ptr4+1:END FOR kb
2118 END FOR ka
2119 END DEFine

2121 DEFine PROCedure FrSave(bm%,cm%,rm%,ptr)
2122 IF sz>0:FOR i=8 TO 15:POKE addr+i,CP(i)
2123 FOR f=1 TO bm%
2124   IF f<48:CURSOR#5,42+f*3,11:PRINT#5,'.':PAUSE 1
2125   FOR r=0 TO rm%-1
2126     FOR c=0 TO cm%-1
2127       IF sz<2:POKE ptr,FG(f,c,r)  :END IF
2128       IF sz>1:POKE ptr,Tile(f,c,r):END IF
2129       ptr=ptr+1
2130     END FOR c
2131   END FOR r
2132 END FOR f
2133 IF sz>1:FOR f=1 TO bm%:POKE ptr,TAss(f):ptr=ptr+1:END FOR f
2134 END DEFine

2136 DEFine PROCedure SelDrv(act%,Act$)
2137 INK#5,CP(7):CURSOR#5,6,20:PRINT#5,'Select ¾ ¿ ¼ ';:BLOCK#5,2,4,78,22,CP(7)
2138 IF act%=1:PRINT#5,'(E)dit':END IF :INK CP(5)
2139 REPeat drv_lp
2140   CURSOR#5,6,11:PRINT#5,Act$&drv$(dn%):CLS#5,4
2141   k=CODE(INKEY$(-1))
2142   SELect ON k
2143     =208:dn%=dn%-1:IF dn%<0:dn%=dm%
2144     =216:dn%=dn%+1:IF dn%>dm%:dn%=1
2145     =101,69:IF act%=1:EditName 5,act%,42,11,16,drv$(dn%)          :REMark Drive/SubDIR
2146     =  10:EXIT drv_lp
2147   END SELect
2148 END REPeat drv_lp
2149 END DEFine

2151 DEFine PROCedure DirList
2152 DELETE drv$(dn%)&'FList':OPEN_NEW#9,drv$(dn%)&'FList':DIR#9,drv$(dn%):CLOSE#9
2153 END DEFine
```

```
2155 DEFine PROCedure Dir_bmp
2156 OPEN_IN#9,drv$(dn%)&'FList':dl%=LEN(drv$(dn%))
2157 REPeat dir_lp
2158    IF EOF(#9) OR sf%>fm%:sf%=sf%-1:ft%=sf%:CLOSE#9:EXIT dir_lp
2159    INPUT#9,F$:F$=F$(dl%-4 TO):fl%=LEN(F$)
2160    IF fl%<=20 AND '_bmp' INSTR F$>0:File$(sf%)=F$:sf%=sf%+1
2161 END REPeat dir_lp
2162 END DEFine

2164 DEFine PROCedure Dir_chk
2165 OPEN_IN#9,drv$(dn%)&'FList':dl%=LEN(drv$(dn%))
2166 REPeat dir_lp
2167    IF EOF(#9):CLOSE#9:chk=0:EXIT dir_lp
2168    INPUT#9,F$:F$=F$(dl%-4 TO)
2169    IF F$==SFile$:CLOSE#9:chk=1:EXIT dir_lp
2170 END REPeat dir_lp
2171 END DEFine

2173 REMark Text Editor

2175 DEFine PROCedure EditName(ch%,act%,sx%,sy%,sm%,str$)
2176 IF act%=2:sl%=('_bmp' INSTR str$)-1:str$=str$(1 TO sl%)
2177 INK#ch%,CP(7):CURSOR#ch%,124,20:PRINT#ch%,'⬅ [_] ➡  ⬅'
2178 BLOCK#ch%,10,3,156,24,CP(7):BLOCK#ch%,2,4,172,22,CP(7)
2179 temp$=str$:sl%=LEN(str$):cp%=sl%+1
2180 REPeat Ed_lp
2181   Ln_Prn:Ln_Cur:k$=INKEY$(#0,-1):k=CODE(k$)
2182   SELect ON k
2183     = 10:IF sl%=0:str$=temp$:END IF :EXIT Ed_lp
2184     = 32:str$=temp$:EXIT Ed_lp
2185     = 48 TO 57,65 TO 90,95, 97 TO 122:Add_chr
2186     =194:IF cp%>1:cp%=cp%-1:Del_chr
2187     =202:Del_chr
2188     =192:IF cp%>1:cp%=cp%-1
2189     =200:IF cp%<sl%+1:cp%=cp%+1
2190   END SELect
2191 END REPeat Ed_lp
2192 IF act%=1
2193   IF sl%<5:str$=temp$:RETurn
2194   IF str$(sl%)<>'_':IF sl%<sm%:str$=str$&'_':ELSE str$(sl%)='_'
2195   IF str$(5)<>'_':str$(5)='_'
2196 END IF
2197 k=0:BLOCK#ch%,60,10,122,20,0:IF act%=2:str$=str$&'_bmp'
2198 END DEFine

2200 DEFine PROCedure Ln_Prn
2201 IF LEN(str$)>sm%:str$=str$(1 TO sm%):cp%=sm%
2202 INK#ch%,5:CURSOR#ch%,sx%,sy%:PRINT#ch%,str$:CLS#ch%,4
2203 END DEFine

2205 DEFine PROCedure Ln_Cur
2206 BLOCK#ch%,6,1,sx%+cp%*6-6,sy%+8,CP(7)
2207 END DEFine
```

```
2209 DEFine PROCedure Add_chr
2210 IF cp% =1 AND sl%=0  :str$=str$&k$
2211 IF cp%>=1 AND cp%<sl%:str$=str$(1 TO cp%-1)&k$&str$(cp% TO sl%)
2212 IF cp%>=1 AND cp%=sl%:str$=str$(1 TO cp%-1)&k$&str$(cp%)
2213 IF cp%> 1 AND cp%>sl%:str$=str$&k$
2214 IF cp%=sm%:str$(cp%)=k$
2215 IF sl%<sm%:sl%=sl%+1:ELSE sl%=sm%
2216 IF cp%<sm%:cp%=cp%+1:ELSE cp%=sm%
2217 END DEFine

2219 DEFine PROCedure Del_chr
2120 IF cp%=sl%:str$=str$(1 TO sl%-1):sl%=sl%-1
2221 IF cp%>=1 AND cp%<sl%:str$=str$(1 TO cp%-1)&str$(cp%+1 TO sl%):sl%=sl%-1
2222 IF cp%=sm%:str$=str$(1 TO sm%-1):cp%=cp%-1:sl%=sm%-1
2223 IF cp%=1 AND sl%=1:str$="":sl%=0
2224 END DEFine
```




```
2226 REMark PIXArt Help Screen

2228 DEFine PROCedure Info
2229 LOCal ik%,ix%,iy%,ip$              :CLS#3:CSIZE#3,0,0
2230 QBold 3,6,6,  4,'PIXELArt'         :QBold 3,6,270,  4,'HELP [?]'
2231 QBold 3,6,8, 29,'SPRITE'           :QBold 3,6,242, 29,'GRID'
2232 QBold 3,6,8,129,'SCREEN'           :QBold 3,6,242,129,'TILE  <  >'
2233 QBold 3,6,8,155,'RETRO GAME' :QBold 3,6,  8,179,'SAVE'
2234 OVER#3,1:INK#3,CP(7):RESTORE 2205
2235 FOR i=1 TO 27:READ ix%,iy%,ip$:CURSOR#3,ix%,iy%:PRINT#3,ip$
2236 DATA 66,  4,"Use ← ↑ ↓ → Cursor Keys to Navigate"
2237 DATA 54, 14,"Abort/Rtn  Spacebar  Action ← Enter"
2238 DATA 50,30,"Frame '-/+'  'G' Set Frame Size",272,30,'16x16'
2239 DATA 12,43,"'X'FLIP'Y' 'z'ROLL'Z'",160,41,"'N'New 'C'Copy 'D'Delete"
2240 DATA 12,54,"'#' MAX   min   Edit",18,113,"PAN'¼½Shift¾¿'SCROLL"
2241 DATA 160,68,"'P' Palette Colour Chg",160,57,"'B' BackGnd Colour Chg"
2242 DATA 18,79,"MOVE  ←ALT→  ReSIZE",38,102,'Set/Drag  CURSOR'
2243 DATA 65,69,'↑',65,87,'↓',89,87,'↑',80,93,'←  →',89,99,'↓'
2244 DATA 157,90,"'SB'",184,90,"Paint  ON    OFF"
2215 DATA 160,101,"'E' Erase  ON    OFF",160,79,"'R' Recolour  '@' Fill"
2246 DATA 48,130,"(1..9)       'T' Set Asset 0..",275,130,'0  00'
2247 DATA 12,141,"'M' Map SCREEN LINKs [#]Test  'K' Key CTRL Settings"
2248 DATA 132,156,"'A' Action [GAME Test Run]"
2249 DATA 12,168,"'F' Files  'D' DIR Settings  'L' Load  'S' Save"
2250 DATA 48,180,'SPRITE_bmp  SCREEN/TILE_bmp  RETROGAME_bas'
2251 OVER#3,1:ik%=CP(7)         :BLOCK#3,12,3,112,18,ik%  :BLOCK#3,2,4,234,16,ik%
2252 BLOCK#3,12,9,61, 54,ik%    :BLOCK#3,10,7,62,55,0     :REMark MAX[ ]min
2253 BLOCK#3,6,5,100, 56,ik%    :BLOCK#3,4,3,101,57,0
2254 BLOCK#3,18,4,244,93,ik%    :BLOCK#3,16,4,293, 93,ik% :BLOCK#3,14,2,294, 94,0
2255 BLOCK#3,12,9,246,101,ik% :BLOCK#3,12,9,295,101,ik% :BLOCK#3,10,7,296,102,0
2256 BLOCK#3,6,5,40,67,ik%      :BLOCK#3,5,4,41,68,0          :REMark [ALT] Edit Area
2257 BLOCK#3,6,5,90,67,ik%      :BLOCK#3,5,4,90,68,0
2258 BLOCK#3,6,5,40,96,ik%      :BLOCK#3,5,4,41,96,0          :BLOCK#3,7,5,89,96,ik%
2259 END DEFine
```