ATARI COLOR GRAPHICS

Examples and discussions of the use of Color Graphics

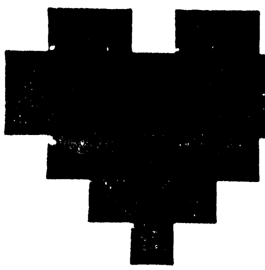
on the ATARI 400/800^{tm} Home Computer System

ATARI, INC.
CONSUMER PRODUCT SERVICE
PRODUCT SUPPORT GROUP
1312 Crossman Ave.
Sunnyvale, CA 94086

(800) 672-1404 inside CA
(800) 538-8543 outside CA

DEMOPAC #4
Rev.2 6-82/JB

THE CASE OF THE UNWANTED HEARTS



If you program in graphics mode 1 or 2 and use lower case characters, little hearts appear on the screen where spaces should be. The way the computer makes letters results in these hearts.

Your computer doesn't know what printed characters look like. It must interpret them using an internal table called a "character set". Graphics modes 1 and 2 have two different *character sets*, each with 64 different characters. One character set contains all the upper case characters (capital letters) and punctuation marks. The other set contains all the lower case letters and graphics characters.

Each key on the keyboard matches a position in the character set. The first position of all character sets represents the space bar. Thus, the computer looks in the first position of the character set whenever the space bar is pressed to see what to put on the screen.

The first position of the upper case character set contains a space, and the first position of the lower case character set contains a heart shape. Therefore, if you use upper case characters, you will see a space on the screen whenever you press the space bar, but if you use lower case characters, you will see a heart.

The computer clears the graphics screen by filling it with the internal character for the space bar. Thus, if you use the lower case character set, the screen will become filled with hearts instead of blank spaces.

Normally, you get the upper case characters, but you can use lower case if you use the command POKE 756,226. Two methods exist to avoid getting the unwanted hearts with this character set.

The first and easiest way is to set the hearts to the screen's background color by using the command SETCOLOR 0,0,0. This method, however, uses up one of your colors.

A better way, but it requires a little more work, is to change the character set so the first lower case character becomes a blank space rather than a heart. Before you try to change the character set, an understanding of its arrangement in the computer's memory would be helpful.

A character set is simply a picture of what the letters will look like when they are printed on the screen. If you look closely at the printing on your television screen, you will see that the letters actually consist of little dots. The computer represents each dot as a single *bit* of information in its memory. Each memory location can store eight bits (one *byte*) of information, or eight dots. A bit can either be 1 or 0. If a bit is set to 1, it represents a dot on the screen. If it is set to 0, it represents a place where no dot shows, or a blank space. The computer represents each character as an 8 x 8 arrangement of dots. Therefore, the computer uses eight locations in memory for each character. For example, the heart character looks like the following in your computer's memory:

```
00000000
00110110
01111111
01111111
00111110
00011100
00001000
00000000
```

See how the 1's and 0's form a heart?

Each row of 1's and 0's can also be read as a number. Page 55 of the BASIC Reference Manual displays the entire character set and corresponding numbers for your ATARI Personal Computer. Each box represents eight locations in the computer's memory and contains the image of one character. The heart is character 64 in the set. A space is character 0 in the set. The space would be represented by a block of 0's in memory. (Do you remember why?)

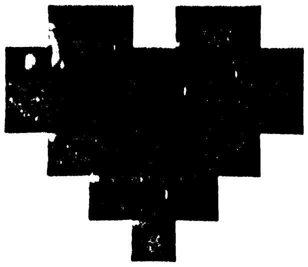
Your computer's character set is stored in read only memory *(ROM)*, but as a result of the versatility of the *ANTIC chip*, we can move the character set into the read/write memory *(RAM)* and the characters will still be displayed correctly. Once we have the set in RAM, we can modify the characters to our heart's content.

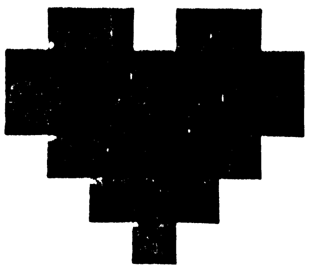
To avoid the unwanted hearts in lower case you need to move the character set into RAM and change the heart to a space, which the program listed below will do. From then on, a simple *POKE* statement will switch you from upper case to lower case characters.

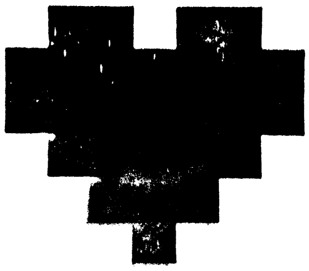
The first step required is to determine if adequate memory space exists for the program to change the heart to a space. This program requires about 4600 bytes of memory, so the first line in the program listing, *Line 10*, checks to see if 4600 bytes of memory are free.

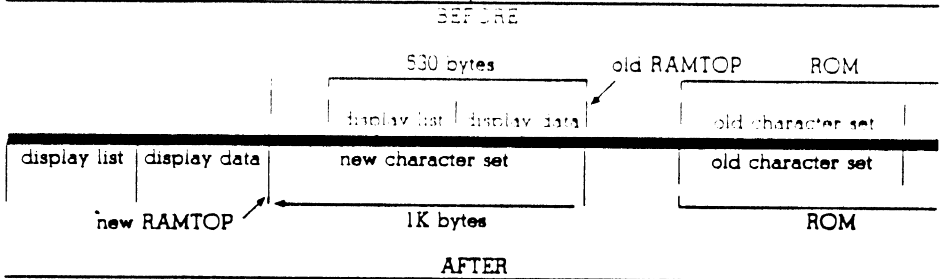
If adequate RAM exists, you need to make room above RAMTOP for the character set. The character set occupies 1K of memory. The computer's Display List occupies a little less than ½K. Since informaton is accessed from memory in 256 Byte or ¼K pages, you must move RAMTOP down 4 pages to allow 1K of space (¼K per page x 4 pages = 1K). The current value of RAMTOP is stored in the RAM location 106. *Line 20* looks into the memory location for RAMTOP (106) and sees what its present value is. *Line 30* substracts 4 pages, moving RAMTOP down the needed 1K.

Before you move your character set, you need to let the computer know you moved RAMTOP so it can save its Display List. You do that with a graphics command. *Line 40* sets up the computer to use graphics mode 1. If you move the character



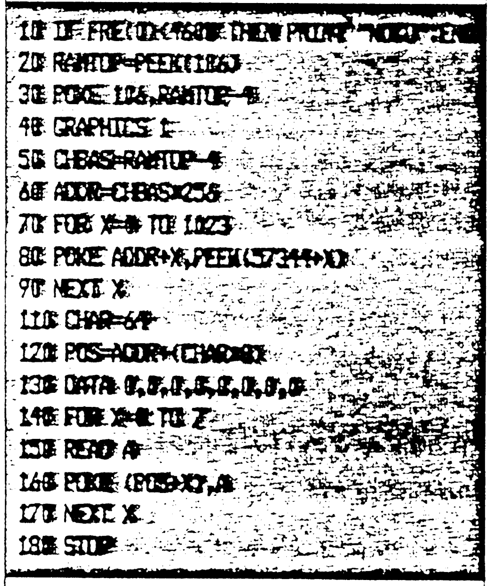






set before doing a graphics command, you might write over the Display List and ruin your display.

Now, you're ready to move the character set from ROM to the last 1K of RAM, just before the old RAMTOP. We'll call the beginning of the character set CHBAS. Since 1K equals 4 pages of memory (¼K per page × 4 pages = 1K), *Line 50* tells the computer where CHBAS, your new character set, should be located in RAM. *Line 60* tells the computer where the CHBAS should begin. (Remember, one page equals 256 bytes.)



```
10 IF FRE(0)<1600 THEN PRINT "NO[illegible]
20 RAMTOP=PEEK(106)
30 POKE 106,RAMTOP-4
40 GRAPHICS 1
50 CHBAS=RAMTOP-4
60 ADDR=CHBAS*256
70 FOR X=0 TO 1023
80 POKE ADDR+X,PEEK(57344+X)
90 NEXT X
110 CHAR=64
120 POS=ADDR+(CHAR*8)
130 DATA 0,0,0,0,0,0,0,0
140 FOR X=0 TO 7
150 READ A
160 POKE (POS+X),A
170 NEXT X
180 STOP
```

In ROM, the old character set began at location 57344. To move the character set so its RAM location begins at ADDR, the actual location of the character set, you need *Lines 70, 80 and 90. Line 80* looks into the ROM memory location and then moves the character set to its new location.

All you need to do now is change the heart to a space. Remember, the heart is character number 64, each character requires eight locations in memory, and the character set begins at ADDR. As a result, the beginning location of the heart character is ADDR + 64 × 8. *Lines 110 and 120* read character 64 from memory and put it into the correct location.

To replace the hearts with spaces, recall that eight values are required to represent a character in memory. For the space, these values are easy to determine . . . all the values are 0! *Line 130* defines character 64 to be represented as all 0's or to be a space.

Once the character has been defined with the DATA command, another FOR-NEXT loop *(Lines 140# 170)* will load it into the character set.

If you use this routine in a program, type in your other lines of code next. If not, end the program with a STOP command *(Line 180)*. Otherwise, the computer will automatically close the graphics display area and prevent you from being able to print in the graphics area.

After you have entered this routine, if you want the upper case character set, use the command POKE 756,CHBAS. For the lower case character set, use the command POKE 756,CHBAS + 2. The hearts will be gone!

COMPUTERESE DICTIONARY

Antic Chip—A microprocessor that enables the computer to display graphic images on the television screen.

Bit—Smallest amount of information your personal computer can work with; must be either a 1 or a 0.

Byte—Amount of information your computer requires to represent a character (letter, number or symbol); one byte consists of eight bits.

Character Set—Table in the computer's memory which contains a picture of what the letters, numbers and symbols will look like when they are printed on the screen.

Display List—Machine language program for the ANTIC chip.

FRE—BASIC command which determines amount of memory available.

PEEK—A BASIC command that tells the computer to look into a specific location in the computer's memory and see what is stored there.

POKE—A BASIC command that tells the computer to put a new number into a specific location in the computer's memory.

RAM—Acronym for Random Access Memory; computer memory for temporary storage of information; the computer can both read information stored here and write new information to be stored here.

RAMTOP—Top of memory; the number put in this location is the number of 256-byte pages of memory space available for RAMTOP.

ROM—Acronym for Read Only Memory; computer memory for permanently stored information; the computer can read information stored here but cannot write new information to be stored here.

ATARI COLOR GRAPHICS

Four-Color Modes, 3,5, and 7

JB 5/82

In four-color modes, four of the system's five color registers are available for use. There are three such modes, GRAPHICS 3,5, and 7. All three are high resolution, or map modes. This means that individual pixels on the screen are turned on and off, instead of being grouped together into characters. The three modes have difference resolutions, or pixel sizes, but the color characteristics are the same.

One of the four available registers controls the background color, so you get three foreground colors. The COLOR statement is used to select one of the four color registers. Once you select a register, all further PLOTs and DRAWTOs get their color from that register, until another COLOR statement is executed. Each register contains a default hue, as follows:

COLOR 1 selects register 0, which contains the color orange.
COLOR 2 selects register 1, which contains the color green.
COLOR 3 selects register 2, which contains the color blue.
COLOR 0 selects register 4, which contains the color black.

In all map modes, COLOR 0 selects the background register. In modes 3,5, and 7 this is register 4, which contains black.

You must use a COLOR statement to select a foreground register before doing any PLOTs, or your plotted points will not show up. If no COLOR statement is encountered, the register defaults to the background, register 4.

The SETCOLOR statement is optional. It is used to change the color information in a register. The SETCOLOR numbers correspond directly to the register numbers. If you select COLOR 1, and do not wish to use orange, you would change the color in that register with SETCOLOR 0. The statement is:

```
COLOR 1: SETCOLOR 0, hue, luminence
```

COLOR 1 selects register 0, and the contents of register 0 are controlled by SETCOLOR 0. When the SETCOLOR statement is executed, everything which you have drawn using COLOR 1 will change to the new hue and luminence.

To erase a point or line, simply draw over it using the background color, COLOR 0. To erase everything in one color, use SETCOLOR to change that register to the background color. Black is hue 0 and luminence 0. A table of hue numbers can be found on page 50 of the BASIC Reference Manual. For luminance, select an even number from 0 to 14. 0 is the darkest and 14 the brightest luminence.

NOTE: There is an error in the Hue Table on page 50. Hue 5, Purple, is missing. Please make the change in your manual.

SELECTING A REGISTER WITH COLOR/PLOT

To select a register for a COLOR/PLOT, you must use the charts on pages 55 and 56 of the BASIC Reference Manual. Find the character in the Internal Character Set chart on page 55. If it is in column 1 or 2 it is in the upper-case set. If it is in column 3 or 4, it is in the lower-case set, and you must POKE 756,226 to use it.

Once you have found your character number, look at the Character/Color Assignment chart on page 56. Registers are referred to by SETCOLOR number. To assign a register to the character, you must add the conversion number to the character number.

If you choose a character from column 1, and wish to assign register 1, the table tells you that no conversion is necessary. When you put the character number in the COLOR statement and PLOT it, the register defaults to 1. Since register 1 contains green, the character is green.

The statement COLOR 10:PLOT 5,5 puts a green star on the screen. The internal character number for a star is 10, which is in column one. With no conversion, register 1 is selected, so it comes out green.

To select another register for your character, add the offset given in the conversion table to the character number, and put this new number in your COLOR statement. To PLOT a star using register 3, use the statement:

```
COLOR 10+128:PLOT 5,5
```

+128 is the conversion for column 1 and register 3. Since register 3 defaults to red, the new star is red.

Any character in column 2 requires conversion. The letter A is character number 33, in column 2. If you want to PLOT an A, you must select COLOR 33+conversion. If you simply PLOT with COLOR 33, you get an orange exclamation point (character 1+32).

NOTE: There is an error in the conversion chart, which you should correct in your manual. Under Conversion 1, SETCOLOR 0, you should change -$32 to #+32. To convert a column 1 character to register 0, add an offset of 32 to the character number.

The registers contain the default colors mentioned, but these colors can be changed with the SETCOLOR statement, as in all other modes.

When you change a hue using SETCOLOR, everything written with that register changes to the new hue. If you want characters in separate colors, you must use separate color registers. A character can be erased by plotting or printing over it in the background color. All characters of one color can be erased by changing that register to the background color, using the corresponding SETCOLOR command.

ATARI COLOR GRAPHICS
Five-Color Text Modes
JB 5/82

Graphics modes 1 and 2 are the Text-Graphics modes. All five of the system's color registers are available for use, one as background, and four as character colors. Mode 2 characters are larger, but color is handled the same way in both modes.

There are two ways to put graphics-characters on the screen. The character can be placed at an x-y coordinate with the COLOR and PLOT statements, or it can be PRINTed through channel #6:

```
PRINT#6;"A"

or

COLOR 10:PLOT 5,5
```

The PLOT command allows you to specify where the character is to go, but not which one it is. The COLOR statement is used in these modes to select a character. The character number is taken from the Internal Character Set chart on page 55 of the BASIC Reference Manual.

SELECTING A REGISTER WITH PRINT#6

The color of a character is determined by which register is used to draw it, but the COLOR statement does not select a register in text modes, as it does in map modes. Selecting a register is a little complicated. Let's take the case of PRINT#6 first.

The statement PRINT#6;"A" prints an orange A. Orange is the default hue in register 0. Registers 1-3 are selected with the lower-case and logo (inverse video) keys. The lower-case key selects register 1, which contains green. The logo key selects register 2, which contains blue. Both keys at once select register 3, which contains red.

To print four letter A's in four different colors, use the following statement:

PRINT#6;"Aa $\boxed{A}$ $\boxed{a}$" (the final characters are in inverse video)

Remember, in these modes, you cannot get the upper-case and lower-case character sets at the same time. When you use the lower-case key, you are selecting a color register, not a lower-case letter. To get a lower-case 'a' you have to POKE 756,226 in order to use the lower-case character set. You cannot get lower-case and upper-case letters together on the same screen.

No matter what method you are using to put your characters on the screen, you can change the color contained in the register by using the SETCOLOR statement. SETCOLOR numbers are the same as register numbers. Any one of the 128 colors can be placed in any of the five color registers.

GRAPHICS 1 and 2

Use Lower Case Without a Screenful of Hearts

JB 4/82

In Text-Graphics modes 1 and 2, the upper-case and lower-case character sets can only be used separately. Because the lower-case character for a space contains a heart, calling up the lower-case set results in the screen filling up with hearts. The following article, reprinted from The ATARI Connection #1, explains in detail why that happens, and provides a program to correct the situation. The program moves the character set from ROM down into RAM, and then redefines the heart-character as a space-character.

When you use the character redefining program, you must include the program itself as part of your initialization. When you then call lower-case, use something like the following routine:

```
200 POKE 756,CHBAS+2:REM instead of the usual 226
210 POSITION 5,5:PRINT#6;"LOWER CASE"
220 POSITION 3,8:PRINT#6;"WITHOUT HEARTS"
```

Other characters can also be redefined using the same program. Simply change the character number (CHAR) in line 110, and the DATA in line 130.

Redefining the character set takes time, and reduces the amount of RAM available for your program. There is an alternative which is much simpler. You can simply erase the hearts by making them the same color as the background. The only problem with this is that you lose one of your color registers: you can use only three character colors, instead of the usual four. If this is OK in your program, use something like the following routine:

```
10 GRAPHICS 1
20 POKE 756,226 :REM call lower-case
30 SETCOLOR 0,0,0 :REM set register 0 to black
40 FOR I=1 TO 5
50 READ X
60 COLOR X+64 :REM add 64 to get another color
70 PLOT I,5
80 NEXT I
90 DATA 33,52,33,50,41
```

Try taking out line 30, and the hearts come back. The data and the offset for the color change are taken from the charts on pages 55 and 56 of the BASIC Reference Manual. The COLOR statement is used in modes 1 and 2 to determine which character is to be plotted.

```
1 REM ***** ETCH-A-SKETCH *****
2 REM *****  PY/JB 2/82   *****
3 REM draw lines on the screen, using the joystick
4 REM ****************************************
10 DIM XSTEP(20),YSTEP(20):REM arrays to hold x and y increments
20 COLOR 1:SETCOLOR 0,2,8:REM set up color info
30 GRAPHICS 7+16:REM set up whole screen in four-color high-res mode
40 REM initialize variables
50 X=80:Y=40
60 XSTEP(5)=1:YSTEP(5)=1:REM move southeast
61 XSTEP(6)=1:YSTEP(6)=-1:REM move northeast
62 XSTEP(7)=1:YSTEP(7)=0:REM move east
63 XSTEP(9)=-1:YSTEP(9)=1:REM move southwest
64 XSTEP(10)=-1:YSTEP(10)=-1:REM move northwest
65 XSTEP(11)=-1:YSTEP(11)=0:REM move west
66 XSTEP(13)=0:YSTEP(13)=1:REM move south
67 XSTEP(14)=0:YSTEP(14)=-1:REM move north
68 XSTEP(15)=0:YSTEP(15)=0:REM don't move
69 REM ****************************************
100 SOUND 0,0,0,0:REM turn off sound
110 IF STRIG(0)=0 THEN GRAPHICS 7+16:REM on trigger, clear screen
120 S=STICK(0):REM check joystick
130 X=X+XSTEP(S):Y=Y+YSTEP(S):REM increment position
140 TRAP 200:REM trap out-of-bounds error
150 PLOT X,Y:REM plot a point at the new position
160 GOTO 110:REM go check stick again
170 REM ****************************************
190 REM if out-of-bounds, do error routine
200 X=X-XSTEP(S):Y=Y-YSTEP(S):REM go back to last position
210 REM sound warning beep
220 FOR VOLUME=15 TO 0 STEP -1
230 SOUND 0,136,10,VOLUME
240 FOR DELAY=1 TO 10:NEXT DELAY
250 NEXT VOLUME
260 GOTO 100:REM try again
```

```
1 REM CIRCLEZ
2 REM EZ/JB 9/81
3 REM compute and draw a circle on the screen,
4 REM filling it in with the XIO fill command
5 REM ***********************************************************************
10 PRINT "GRAPHICS MODE...";:INPUT G:REM select mode 3-8
20 GRAPHICS G:COLOR 1:POKE 765,1
21 REM location 765, 'fildat', tells the fill command which color to use
30 PRINT "CENTER (X,Y)...";:INPUT X0,Y0
40 PRINT "RADIUS ...";:INPUT R
50 R2=R*R
60 PLOT X0,Y0+R:REM plot first point on circle
70 FOR Y=R TO -R STEP -1:REM right half
80 X=SQR(R2-(Y*Y)):REM compute drawto-point with circle formula
90 X=X*8/7:REM correct for uneven pixel size ('fudge factor')
100 DRAWTO X0+X,Y0+Y
110 NEXT Y
120 FOR Y=-R TO R:REM left half
130 X=SQR(R2-(Y*Y))*8/7:REM include fudge factor
140 PLOT X0-X,Y0+Y:REM plot each point
150 IF ABS(Y)=R THEN GOTO 170:REM check for top and bottom pixels
160 XIO 18,#6,0,0,"S:":REM fill as you go
170 NEXT Y
180 GOTO 10:REM try another one
```

```
1 REM FILL IN A SHAPE
2 REM LW/DBM 5/82
3 REM This program demonstrates the use of a BASIC algorithm
4 REM to do color filling of an arbitrary shape
5 REM *********************************************************************
8 REM draw shape
9 REM
10 GRAPHICS 7+16:REM .          full screen graphics mode
20 COLOR 1:DEG :REM .           select color register, compute in degrees
30 GOSUB 500:REM .              call math routine
40 PLOT X,Y:REM .               draw initial point in shape
50 FOR T=0 TO 360:REM .         number of points in shape
60 GOSUB 500:REM .              call math routine
70 PLOT X,Y:REM .               draw shape
80 NEXT T
90 REM ********************************************************************
98 REM shape drawn, now fill it in with color
99 REM
100 X=79:Y=47:REM .             starting point for fill routine
110 FILL=1000:TRAP 2000:REM .   fill routine is at line 1000
200 GOSUB FILL
300 GOTO 300:REM .              loop to keep image on screen
400 REM *******************************************************************
498 REM math routine computes next point along curve
499 REM
500 Y=30*COS(T)+47+10*COS(T*2)
510 X=30*SIN(T)+79+10*SIN(T*3)
520 RETURN
900 REM *******************************************************************
998 REM fill routine
999 REM
1000 PLOT X,Y:REM .             plot a point
1010 X=X-1:LOCATE X,Y,Z:REM .   check if next point is blank (color 0)
1020 IF Z=0 THEN GOSUB FILL:REM .if so, plot another point
1030 X=X+1
1040 Y=Y-1:LOCATE X,Y,Z:REM .   check in all directions
1050 IF Z=0 THEN GOSUB FILL
1060 Y=Y+1
1070 X=X+1:LOCATE X,Y,Z
1080 IF Z=0 THEN GOSUB FILL
1090 X=X-1
1100 Y=Y+1:LOCATE X,Y,Z
1110 IF Z=0 THEN GOSUB FILL
1120 Y=Y-1
1130 RETURN
1990 REM ******************************************************************
1997 REM error routine: locations 186 and 187 hold a pointer to
1998 REM the line number at which STOP or TRAP occurred.
1999 REM
2000 TRAP 2000:GOTO PEEK(186)+PEEK(189)*256+20
2001 REM .                      go back to error line+20
```

GTIA GRAPHIC MODES

Using Modes 9,10, and 11

JB 5/82

The new GTIA chip allows three extra graphic modes, 9,10, and 11. Modes 9 and 11 are complimentary; they work the same way, except that mode 9 has one hue and sixteen luminences, while mode 11 has one luminence and sixteen hues. Mode 10 combines the player and playfield color registers, so that nine registers are available at once.

In mode 9, the single hue is set in the background register, with the statement SETCOLOR 4,hue,0. In mode 11, the single luminence goes in register 4: SETCOLOR 4,0, luminence. In both modes, the COLOR statement selects one of the 16 variations of luminence (mode 9) or hue (mode 11). The STAR11 and STAR9 programs which follow demonstrate the technique.

Mode 10 combines all the player and playfield color registers, so that nine registers (1 background, 8 foreground) can be used at once. Since player registers cannot be set by SETCOLOR commands, it is best to set all nine registers with POKE commands. The locations are 704-712(decimal). 704 controls the background color. You can then select a register with the COLOR statement, 0-8.

The resolution in all three modes is the same, 80 by 192. Each pixel is one scan line high and four color clocks wide. In contrast, a mode 8 pixel is one scan line high and half a color clock wide. A picture drawn in a GTIA mode looks similar to one drawn in mode 7, although the individual pixels are a different shape.

Mode 9 is appropriate for the simulation of depth and 3-D effects, since the many luminances allow fine shading gradations. Mode 10 can be used to provide an illusion of motion, by cycling colors through the registers, as shown in the following demo program. Mode 11 allows more colors to be displayed at once than any other mode, without resorting to machine-language programming. For a thorough discussion of how these modes are selected by ANTIC, refer to APPENDIX E of De Re ATARI, available from the ATARI Program Exchange.

```
1 REM STAR9
2 REM JB 3/82
3 REM a starburst pattern in graphics mode 9
4 REM cycling through all colors and luminances
5 REM ****************************************
10 GRAPHICS 9:C=0
20 C=C+1:IF C>15 THEN C=0
30 SETCOLOR 4,C,0:REM change the color after each cycle
40 FOR X=0 TO 15:REM cycle through every luminance
50 RX=INT(RND(0)*80):RY=INT(RND(0)*190)
60 COLOR X:REM change the luminance
70 PLOT 40,95:REM center point
80 DRAWTO RX,RY
90 NEXT X
100 GOTO 20
```

```
1 REM STAR11
2 REM JB 3/82
3 REM a starburst pattern in graphics mode 11
4 REM cycling through all colors and luminances
5 REM ****************************************
10 GRAPHICS 11:C=0
20 C=C+1:IF C>15 THEN C=0
30 SETCOLOR 4,0,C:REM change the luminance after each cycle
40 FOR X=0 TO 15:REM cycle through every color
50 RX=INT(RND(0)*80):RY=INT(RND(0)*190)
60 COLOR X:REM change the color
70 PLOT 40,95:REM center point
80 DRAWTO RX,RY
90 NEXT X
100 GOTO 20
```

```
1 REM GTIA MODE 10
2 REM JB 5/82
3 REM Mode 10 has 9 color registers available. This program shows
4 REM how to simulate motion by cycling colors through the registers.
5 REM ******************************************************************
10 GRAPHICS 10
15 REM Hues are assigned by poking into the registers. 704 is background.
16 REM Here, each color is +16, to get the next hue with the same luminence
20 POKE 704,0:POKE 705,30:POKE 706,46:POKE 707,62:POKE 708,78
30 POKE 709,94:POKE 710,110:POKE 711,126:POKE 712,142
35 REM ******************************************************************
40 COLOR 1:REM select a register with the color statement, 0-8
50 FOR X=0 TO 9:PLOT X,0:DRAWTO X,190:NEXT X
60 COLOR 2:REM select the next register
70 FOR X=10 TO 19:PLOT X,0:DRAWTO X,190:NEXT X
80 COLOR 3
90 FOR X=20 TO 29:PLOT X,0:DRAWTO X,190:NEXT X
100 COLOR 4
110 FOR X=30 TO 39:PLOT X,0:DRAWTO X,190:NEXT X
120 COLOR 5
130 FOR X=40 TO 49:PLOT X,0:DRAWTO X,190:NEXT X
140 COLOR 6
150 FOR X=50 TO 59:PLOT X,0:DRAWTO X,190:NEXT X
160 COLOR 7
170 FOR X=60 TO 69:PLOT X,0:DRAWTO X,190:NEXT X
180 COLOR 8
190 FOR X=70 TO 79:PLOT X,0:DRAWTO X,190:NEXT X
195 REM ******************************************************************
199 REM Cycle colors through registers-poke each with peek of next one.
200 N=PEEK(705)
210 FOR I=705 TO 711
220 POKE I,PEEK(I+1)
230 NEXT I
240 POKE 712,N
250 GOTO 200:REM keep cycling
```

```
1 REM SWIRL
2 REM WEB/DBM  4/82
3 REM A demonstration of the graphics modes and their capabilities
10 GRAPHICS 2+16:REM No text window
20 POSITION 3,5:PRINT #6;"CHOOSE A MODE"
30 PRINT #6;"HOLD start TO RESET"
48 REM *********************************************************************
49 REM Read the keyboard
50 OPEN #1,4,0,"K:":REM Open the keyboard as a device
60 GET #1,X:REM Returns ATASCII code for the key pressed
70 CLOSE #1
80 MODE=X-48:IF MODE<3 OR MODE>8 THEN 50:REM Convert ATASCII code to mode numb
90 RESTORE 400+MODE:REM Read only data for chosen graphics mode
100 READ HORIZ,VERT:REM Read data as x and y coordinates
110 GRAPHICS MODE+16:REM Full-screen graphics
198 REM *********************************************************************
199 REM Create swirling effect
200 COLOR 1:REM Select color register
210 POKE 708,RND(0)*255:REM Put random color into register
215 GOSUB 300:REM Call drawing routine
220 IF PEEK(53279)<>7 THEN RUN :REM If START key pressed, start over
225 COLOR 2:REM Select new register
230 POKE 709,RND(0)*255:REM Put random color into register
235 GOSUB 300:REM Call drawing routine
240 IF PEEK(53279)<>7 THEN RUN :REM Check for START key
250 POKE 77,0:REM Disable the attract mode
260 GOTO 200:REM Start over
298 REM *********************************************************************
299 REM This subroutine draws the design
300 FOR I=VERT TO 0 STEP -1
310 J=VERT-I:REM J goes down as I goes up
320 PLOT 0,I
330 DRAWTO HORIZ,J:REM Draw the line
340 NEXT I
360 FOR I=0 TO HORIZ
370 J=HORIZ-I:REM J goes left as I goes right
380 PLOT I,0
390 DRAWTO J,VERT:REM Draw the line
395 NEXT I:RETURN
399 REM *********************************************************************
400 REM The data statements define the screen size for each graphics mode
403 DATA 39,23
404 DATA 79,47
405 DATA 79,47
406 DATA 159,95
407 DATA 159,95
408 DATA 319,191
```

```
1 REM RACE
2 REM WBB/DBM 5/82
3 REM This program demonstrates the use of ATARI graphics
4 REM and sound in a game format.
8 REM *******************************************************************
9 REM How many racers?
10 GRAPHICS 2+16:REM Full screen graphics
20 POSITION 0,5:PRINT #6;"SELECT # OF PLAYERS"
30 OPEN #1,4,0,"K:":REM Open the keyboard
40 GET #1,X:REM This returns ATASCII code of key pressed
50 X=X-48:REM Convert keycode to a number
60 IF X<1 OR X>9 THEN 40:REM Allow only 1 through 9
70 CLOSE #1
100 GRAPHICS 2+16:POSITION 0,4:PRINT #6;"PAY YOUR DUES"
105 POSITION 0,6:PRINT #6;"START TO CONTINUE"
110 IF PEEK(53279)<>6 THEN 110:REM Wait until START key pressed
120 POSITION 0,8:PRINT #6;"AND THEY'RE OFF..."
130 FOR WAIT=1 TO 250:NEXT WAIT:REM Delay loop
140 GRAPHICS 3+16:REM Full-screen graphics mode
150 DIM RACER(40):FOR I=0 TO 40:RACER(I)=0:NEXT I:REM Initialize
160 POKE 708,52:POKE 709,206:REM Set colors for racers
198 REM *******************************************************************
199 REM Advance a racer
200 TRACK=INT(RND(0)*X)+1:REM Pick a random track
205 POS=TRACK*2:REM Racer position
210 COL=TRACK:IF COL>3 THEN COL=COL-3:IF COL>3 THEN COL=COL-3:REM Set color
215 COLOR COL:REM Select color register
220 RACER(POS)=RACER(POS)+1:REM Increment the chosen racer's position
230 PLOT RACER(POS),POS:REM Plot the new position
240 SOUND COL,40-RACER(POS),6,10:REM Set sound according to racer number
250 IF RACER(POS)<39 THEN 200:REM If no winner, advance another racer
258 REM *******************************************************************
259 REM We have a winner!
260 FOR WAIT=1 TO 500:NEXT WAIT:REM Delay loop
270 GRAPHICS 2+16:REM Full screen graphics
280 POSITION 0,5:PRINT #6;"AND THE WINNER IS #";TRACK
290 FOR VOICE=0 TO 3:SOUND VOICE,0,0,0:NEXT VOICE:REM Turn off sound
300 FOR WAIT=1 TO 1000:NEXT WAIT
310 CLR :REM Clear all variables
320 GOTO 10
```

```
 1 REM BUMPER
2 REM WBB/DBM 4/82
3 REM A demonstration of image positioning and movement in a game format.
4 REM This game is played using joysticks in the #1 and #2 ports.
7 REM **********************************************************************
10 GRAPHICS 2+16:POKE 752,1:REM set full-screen text mode and eliminate cursor
20 DIM X(2),Y(2),D(2),BEGIN(1)
30 POSITION 4,3:PRINT #6;"PRESS start           TO PLAY"
40 POSITION 6,7:PRINT #6;"bumper!"
50 IF PEEK(53279)<>6 AND STRIG(0)<>0 AND STRIG(1)<>0 THEN 50:REM Check for STA
 key or triggers pressed
55 X(0)=2:Y(0)=2:X(1)=35:Y(1)=20:Z=0:REM Initialize
70 FOR PLRNO=0 TO 1:D(PLRNO)=0:BEGIN(PLRNO)=1:NEXT PLRNO
80 GRAPHICS 3+16:REM Set full-screen map mode
85 PLOT X(0),Y(0):PLOT X(1),Y(1):REM Starting positions
98 REM *********************************************************************
99 REM This section checks the sticks
100 FOR PLRNO=0 TO 1:REM Player number 0 or 1
110 IF STICK(PLRNO)=15 THEN NEXT PLRNO
120 IF STICK(PLRNO)=15 THEN 100
128 REM ***********************************************************************
129 REM This section changes player position if sticks have been pushed
130 ST=STICK(PLRNO)
140 IF ST=14 THEN Y(PLRNO)=Y(PLRNO)-1:GOTO 300
150 IF ST=6 THEN X(PLRNO)=X(PLRNO)+1:Y(PLRNO)=Y(PLRNO)-1:GOTO 300
160 IF ST=7 THEN X(PLRNO)=X(PLRNO)+1:GOTO 300
170 IF ST=5 THEN X(PLRNO)=X(PLRNO)+1:Y(PLRNO)=Y(PLRNO)+1:GOTO 300
180 IF ST=13 THEN Y(PLRNO)=Y(PLRNO)+1:GOTO 300
190 IF ST=9 THEN X(PLRNO)=X(PLRNO)-1:Y(PLRNO)=Y(PLRNO)+1:GOTO 300
200 IF ST=11 THEN X(PLRNO)=X(PLRNO)-1:GOTO 300
210 IF ST=10 THEN X(PLRNO)=X(PLRNO)-1:Y(PLRNO)=Y(PLRNO)-1:GOTO 300
220 NEXT PLRNO:GOTO 100
298 REM ***********************************************************************
299 REM This section keeps the player from going off the screen
300 IF X(PLRNO)<0 THEN X(PLRNO)=0:BEGIN(PLRNO)=1
310 IF X(PLRNO)>39 THEN X(PLRNO)=39:BEGIN(PLRNO)=1
320 IF Y(PLRNO)<0 THEN Y(PLRNO)=0:BEGIN(PLRNO)=1
330 IF Y(PLRNO)>23 THEN Y(PLRNO)=23:BEGIN(PLRNO)=1
340 IF BEGIN(PLRNO)=1 THEN BEGIN(PLRNO)=0:GOTO 430:REM This keeps player from
350 REM self-destructing if against the wall
398 REM ***********************************************************************
399 REM Check new position and move player
400 POSITION X(PLRNO),Y(PLRNO):GET #6,Z:REM Check to see if space is occupied
410 IF Z=1 OR Z=2 THEN 500:REM If space is occupied, call end routine
420 D(PLRNO)=D(PLRNO)+1:REM Add to score if player moves
430 COLOR PLRNO+1:PLOT X(PLRNO),Y(PLRNO):REM Put player in new position
448 REM ***********************************************************************
449 REM Sound routine
450 FOR WAIT=1 TO 10:SOUND 2,143,6,10:NEXT WAIT:SOUND 2,0,0,0
460 NEXT PLRNO:GOTO 100:REM Start over
498 REM ***********************************************************************
499 REM End routine (explosion on collision)
500 SOUND 2,50,8,10:FOR N=1 TO 20:FOR L=1 TO 10:SETCOLOR 2,0,L:SETCOLOR 4,0,L
505 NEXT L:NEXT N:SOUND 2,0,0,0
510 GRAPHICS 2+16:PRINT #6;"SCORE #1","  SCORE #2"
520 PRINT #6;"   ";D(0),"    ";D(1)
530 PRINT #6:PRINT #6
540 POSITION 3,8:PRINT #6;"PRESS start TO        PLAY AGAIN  "
550 GOTO 50
```