# Sprites

A sprite is a 16x16 bit graphic image that can be moved about on the screen independently of the background. When the sprite is displayed MMBasic will automatically save the background text and graphics under the sprite and when the sprite is turned off or moved MMBasic will restore the background.

The sprites are defined in a file which is loaded into memory using the SPRITE LOAD command, the number of sprites contained in the file is only limited by the amount of available memory. Each sprite in the file can contain pixels of any colour (on the Colour Maximite) and can also have transparent pixels which allow the background to show through. See below for a detailed description of creating a sprite file.

## Manipulating Sprites

To manipulate the sprites you can use the command SPRITE ON which will display a specific sprite at a specified location on the screen. SPRITE MOVE will move a sprite to a new location and restore the background at the old location. SPRITE OFF will remove a sprite from the screen and restore the background.

Sprites should not overlap but if they do you should turn them off in the reverse sequence that you turned them on before you turn them on again at their new location. This will enable the background image to be correctly maintained.

For example, the following two sprites overlap:

```
SPRITE ON 1, 100, 150      ' sprite 1 is drawn at x = 100, y = 150
SPRITE ON 2, 110, 160      ' sprite 2 overlaps
```

To move the sprites they need to be turned off in the reverse sequence:

```
SPRITE OFF 2
SPRITE OFF 1
```

Then they can be redrawn at their new location:

```
SPRITE ON 1, 104, 154      ' sprite 1 is drawn at x = 104, y = 154
SPRITE ON 2, 116, 166      ' sprite 2 still overlaps
```

Because sprites are drawn so fast the user is unaware that the sprite has been turned off then redrawn.

## Specifying the Background Colour

An alternative to turning sprites off in sequence is to specify the background colour when using the SPRITE ON or SPRITE MOVE commands. The background colour is optional and is specified at the end of the command. For example: SPRITE ON 1, 100, 100, BLUE

This results in a much faster operation when using a solid background colour because MMBasic does not have to copy the background to a buffer. It also means the MMBasic will always restore the correct colour, even if sprites overlap.

## Collision Detection

You can use the COLLISION() function to detect if a sprite has collided with another sprite or the edges of the screen. A collision is reported if the non transparent portion of the sprite is just touching (ie, the non transparent pixels are adjacent) or overlapping the non transparent portion of another sprite or the edge of the screen.

To detect if a sprite has collided with another sprite you use: R = COLLISION (n, SPRITE)

And to detect if it has collided with the screen edge you use: R = COLLISION (n, EDGE)

Where 'n' is the number of the sprite to test.

In both cases the value returned by the function indicates if the collision was on the left of the sprite, the right, the top, etc. Following a collision COLLISION () will return:

| | |
|---|---|
| &B0001 | Indicating a collision with something on the left of the sprite |
| &B0010 | Collision on the right |
| &B0100 | Collision on the top |
| &B1000 | Collision on the bottom |

Note that it is possible for these results to be combined. For example; a result of &B0101 indicates that the sprite has collided with something both at the top and left of the sprite (for example the top left corner of the screen). When testing for collisions with other sprites it is possible for the function to return &B1111

indicating that there are collisions on all sides. This can happen if the sprite is surrounded on all sides by other sprites.

If the sprite is overlapping another (ie, one or more non transparent pixels are on top of another sprite's non transparent pixels) bit &B10000 will be set in the value returned by COLLISION () in addition to the bits for left, right, etc as described above.

## Format of a Sprite File

A sprite file is similar to a font file except that it contains the definition of sprites which are 16x16 bit graphical objects. The sprite file is just a text file containing ordinary characters which are loaded line by line to build the bitmap of each sprite. Currently the dimensions of each sprite are fixed at 16x16 bits although alternative sizes may be allowed in the future.

The first non-comment line in the file must be the specifications for the sprite file as follows:

```
dimension, number
```

Where 'dimension' is the height and width of the sprites in pixels. At this time it must be the number 16. 'number' is the number of sprites in the file and is limited only by the amount of free memory available. The remainder of the lines specify the bitmap for each sprite.

Each line represents a horizontal row of pixels with each character in the line defining the colour of the pixel. The character can be a single numeric digit in the range of 0 to 7 representing the colours black to white or it can be a space which means that that particular pixel will be transparent (ie, the background will show through). On the monochrome Maximite 0 represents a black pixel and any other number represents a white pixel.

Each sprite must immediately follow the preceding sprite in the file and be defined by 16 lines each of 16 characters wide (although trailing spaces can be omitted and will be assumed to be transparent pixels).

A comment line has an apostrophe (') as the first character and can occur anywhere. A comment line is completely ignored; all other lines are significant.

The following example is of a file that contains a single sprite consisting of a red ball with a white border and a blue centre dot. On the monochrome Maximite this would display as a white ball.:

```
' example sprite
' TEST.SPR
16, 1
      7777
    74444447
  744444444447
 74444444444447
 74444444444447
7444444444444447
7444444114444447
7444441111444447
7444441111444447
7444444114444447
7444444444444447
 74444444444447
 74444444444447
  744444444447
    74444447
      7777
```