CS212 GBA

Memory

Spring 2008

1 Sprite Attributes In Depth

In order to see what all the possibilities for the usage of sprites are, we can look again to our header files for the definitions of the various sprite options. But before we can do that we must know what options each attribute is associated with. You should be aware we will focus on regular sprites, and not rotation and scaling sprites. When in rotation and scaling mode, the sprite attributes have slightly different meanings.

1.1 Attribute 0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Shape		Pal. Mode	Mosaic	Alpha		Double	Rotation	Y-coordinate							

- Bits 0-7: Sets Y-coordinate of the sprites. This is the Y-coordinate of the upper left corner of the sprite. These values wrap around the size of the screen in memory (256 pixels)
- Bit 8: Turns rotation and scaling on or off.
- Bit 9: Sets the sprite to be double its size.
- Bits 10-11: Controls transparency of the sprite against the background. You likely won't need this in the scope of this course.
- Bit 12: Enables mosaic for this sprite
- Bit 13: Sets the palette mode, either COLOR_16 or COLOR_256. If COLOR_16 we must set choose a palette in arribute 3.
- Bits 14-15: Sets the shape of the sprite. The possible shapes are SQUARE, TALL, WIDE. These bits are used with the form from attribute 1 to control the overall appearance of the sprite. Refer to the table in 1.4

1.2 Attribute 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Size		Vertical Flip	Horizontal Flip	Not Used			X-coordinate								

- Bits 0-8: Set the X-coordinate of the sprite. This it the X-coordinate at the upper left corner of the sprite. Note that is this is actually 9 bits, so that the values wrap around on twice the size of the screen (512 pixels)
- Bits 9-11: These bits would be used in rotation.
- Bit 12: Sets a horizontal flip on the sprite on screen.

- Bit 13: Sets a vertical flip on the sprite on screen.
- Bits 14-15: Sets the size of the sprite on screen. This combined with the value of shape in attribute 0 controls the overall appearance of the sprite on screen. Refer to the table in 1.4

1.3 Attribute 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Palette Number				Priority		Sprite Number									

- Bits 0-9: Location of the sprite graphic data in memory. Note that when using a 256 color palette each sprite requires 64 bytes. Since graphic memory is allocated in 32 byte blocks, a 256 color sprite effectively uses two slots.
- Bits 10-11: Sets the priority of the layering of this sprite with other sprites. 0 is the highest priority.
- Bits 12-15: Sets the palette of the sprite if you are using 16 color palettes.

1.4 Sprite Shape and Size

Determining what shape and size you should use is dependent on the game that you are making, but you should be aware of the effects it has on your memory usage. The obvious consequence, is when you are using a larger sprite, you are using more memory to store that sprite. Consider the possibility of using the SIZE_DOUBLE flag, as well as stretching provided by the WIDE and TALL shapes.

Attribute 0 (Bits 14-15)	Attribute 1 (Bits 14-15)	Size	Bytes per sprite (256 colors)	Bytes per sprite (16 colors)
SQUARE	SIZE_8	8X8	64	32
SQUARE	SIZE_16	16X16	256	128
SQUARE	SIZE_32	32X32	1024	512
SQUARE	SIZE_64	64X64	4096	2048
TALL	SIZE_8	8X16	128	64
TALL	SIZE_16	8X32	256	128
TALL	SIZE_32	16X32	512	256
TALL	SIZE_64	32X64	2048	1024
WIDE	SIZE_8	16X8	128	64
WIDE	SIZE_16	32X8	256	128
WIDE	SIZE_32	32X16	512	256
WIDE	SIZE_64	64X32	2048	1024

1.5 Attribute Flags

We have provided definitions of the useful flags for setting sprites attributes. They are just hex numbers to set values to specific bit positions. If you are having trouble understanding why these numbers are what they are, or how to use them, pleaes consult a TA.

From `gba_sprites.h`:

```
// Attribute 0
#define ROTATION_FLAG          0x0100
#define SIZE_DOUBLE            0x0200
#define MODE_NORMAL            0x0000
#define MODE_TRANSPARENT       0x0400
#define MODE_WINDOWED          0x0800
#define MOSAIC                 0x1000
#define COLOR_16               0x0000
#define COLOR_256              0x2000
#define SQUARE                 0x0000
#define WIDE                   0x4000
#define TALL                   0x8000

// Attribute 1
#define ROTDATA(n)             (n << 9)
#define HORIZONTAL_FLIP        0x1000
#define VERTICAL_FLIP          0x2000
#define SIZE_8                 0x0000
#define SIZE_16                0x4000
#define SIZE_32                0x8000
#define SIZE_64                0xC000

// Attribute 2
#define PRIORITY(n)            ((n)<<10)
#define PALETTE(n)             ((n)<<12)
```

From these, combined with what you've learned about bitfields, it should start becoming clear just how you would set a specific option. For example, if you had a 16x16 sprite (in the 2nd position in VRAM) with a 256 color palette, that you wanted to be flipped horizontally, and to appear at the coordinates (22,44) you would do something like the following:

```
sprites[0].attribute0 = COLOR_256 | SQUARE | 44;
sprites[0].attribute1 = SIZE_16 | 22;
sprites[0].attribute2 = 2;
```

2 Backgrounds

2.1 Palettes

Similar to the sprites, the backgrounds

2.2 Text Backgrounds

Each background (BG0-BG4) has a register that controls a variety of options that define the behavior and appearance of the background.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Size		n/a	Screen Base Block					Palette Mode	Mosaic	n/a		Char Base Block		Priority	

- Bits 0-1: Set the priority for this background. 0 is the highest priority.
- Bits 2-3: Select which character base block tile data for this background is stored in.
- Bits 4-5: Not used
- Bit 6: Enable mosaic mode for this background.
- Bit 7: Choose the palette mode for this background. Either one 256 color palette, or 16 16 color palettes.
- Bits 8-12: Select the screen base black for this background. This controls where we store the map data for the tiles on screen.
- Bit 13: Not used
- Bit 14-15: Choose the size of the background.

Once you set this register, and load tile graphics into memory (the character base block, specifically), you have to set up the screen base block entries, which controls the appearance of tiles on screen.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Palette				Vertical Flip	Horizontal Flip	Tile Number									

- Bits 0-9: Choose which tile in the character base block this position in the screen should display.
- Bit 10: Flip this screen position's tile horizontally.
- Bit 11: Flip this screen position's tile vertically.
- Bit 12-15: If we are using the 4 bit tile palette mode, then these bits select which palette this tile should use.

2.3 Background Modes

There are five background modes, which provide different combinations of the three different background formats available (bitmapped, text, and rotation/scaling) to the GBA on the 4 available backgrounds. We will be concerned mostly with text (tiled) backgrounds in this course. The modes for tiled backgrounds are as follows:

- Mode 0: All four backgrounds will be text backgrounds.
- Mode 1: Only three backgrounds are usable in this mode. Backgrounds 0 and 1 are text backgrounds, background 2 is a rotation/scaling background.
- Mode 2: Backgrounds 0 and 1 are text, 2 and 3 are rotation/scaling.