VRAM Tile Data

Tile data is stored in VRAM in the memory area at $8000-$97FF; with each tile taking 16 bytes, this area defines data for 384 tiles. In CGB Mode, this is doubled (768 tiles) becaus of the two VRAM banks.

Each tile has 8x8 pixels and has a color depth of 4 colors/gray shades. Tiles can be displa as part of the Background/Window maps, and/or as OBJ tiles (foreground sprites). Note th OBJs don't use color 0 - it's transparent instead.

There are three "blocks" of 128 tiles each:

Block	VRAM Address	OBJs	Corresponding Tile IDs	
			BG/Win if LCDC.4=1	BG/Win if LCDC.4=0
0	$8000–$87FF	0–127	0–127	
1	$8800–$8FFF	128–255	128–255	128–255 (or -128–-1)
2	$9000–$97FF	(Can't use)		0–127

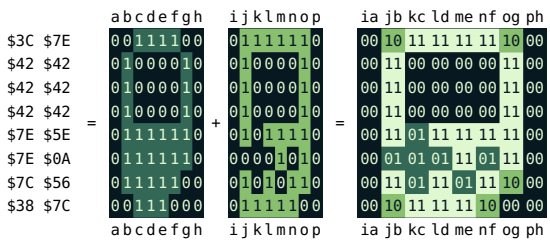
Tiles are always indexed using an 8-bit integer, but the addressing method may differ. The "$8000 method" uses $8000 as its base pointer and uses an unsigned addressing, meanin that tiles 0-127 are in block 0, and tiles 128-255 are in block 1. The "$8800 method" uses $9000 as its base pointer and uses a signed addressing, meaning that tiles 0-127 are in block 2, and tiles -128 to -1 are in block 1, or to put it differently, "$8800 addressing" take tiles 0-127 from block 2 and tiles 128-255 from block 1. (You can notice that block 1 is shared by both addressing methods)

Sprites always use "$8000 addressing", but the BG and Window can use either mode, controlled by LCDC bit 4.

Each tile occupies 16 bytes, where each line is represented by 2 bytes:

```
Byte 0-1  Topmost Line (Top 8 pixels)
Byte 2-3  Second Line
etc.
```

For each line, the first byte specifies the least significant bit of the color ID of each pixel, and the second byte specifies the most significant bit. In both bytes, bit 7 represents the leftmost pixel, and bit 0 the rightmost. For example, the tile data `$3C $7E $42 $42 $42 $42 $42 $42 $7E $5E $7E $0A $7C $56 $38 $7C` appears as follows:



Sample tile data

For the first row, the values `$3C $7E` are `00111100` and `01111110` in binary. The leftmost bits are 0 and 0, thus the color ID is binary `00`, or 0. The next bits are 0 and 1, thus the color II binary `10`, or 2 (remember to flip the order of the bits!). The full eight-pixel row evaluates 0 2 3 3 3 3 2 0.

A tool for viewing tiles can be found here.

So, each pixel has a color ID of 0 to 3. The color numbers are translated into real colors (c gray shades) depending on the current palettes, except that when the tile is used in an OB the color ID 0 means transparent. The palettes are defined through registers BGP, OBP0 a OBP1, and BCPS/BGPI, BCPD/BGPD, OCPS/OBPI and OCPD/OBPD (CGB Mode).