- About
- Articles
- Work
- ds
- motor2

search in blog...

Bitwise gems – fast integer math

May 10, 2007 on 2:34 pm | In Actionscript | 83 Comments

Bitwise operators are very fast in AS3, so here is a small collection of code snippets which can speed up certain computations. I won't explain what bit operators are and how to use them, rather pointing to an excellent article hosted on gamedev.net: 'Bitwise Operations in C'.
If you know any good tricks that are not included here, feel free to leave a comment or send me an email. All benchmarks were done in AS3.

Left bit shifting to multiply by any power of two

Approximately 300% faster.

```
x = x * 2;
x = x * 64;

//equals:
x = x << 1;
x = x << 6;
```

Right bit shifting to divide by any power of two

Approximately 350% faster.

```
x = x / 2;
x = x / 64;

//equals:
x = x >> 1;
x = x >> 6;
```

Number to integer conversion

Using int(x) is 10% faster in AS3. Still the bitwise version works better in AS2.

```
x = int(1.232)
```

```
//equals:
x = 1.232 >> 0;
```

Extracting color components

Not really a trick, but the regular way of extracting values using bit masking and shifting.

```
//24bit
var color:uint = 0x336699;
var r:uint = color >> 16;
var g:uint = color >> 8 & 0xFF;
var b:uint = color & 0xFF;

//32bit
var color:uint = 0xff336699;
var a:uint = color >>> 24;
var r:uint = color >>> 16 & 0xFF;
var g:uint = color >>>  8 & 0xFF;
var b:uint = color & 0xFF;
```

Combining color components

'Shift up' the values into the correct position and combine them.

```
//24bit
var r:uint = 0x33;
var g:uint = 0x66;
var b:uint = 0x99;
var color:uint = r << 16 | g << 8 | b;

//32bit
var a:uint = 0xff;
var r:uint = 0x33;
var g:uint = 0x66;
var b:uint = 0x99;
var color:uint = a << 24 | r << 16 | g << 8 | b;
```

Swap integers without a temporary variable using XOR

Pretty neat trick, it is explained in detail in the link at the top of the page. This is 20% faster.

```
var t:int = a;
a = b;
b = t;

//equals:
a ^= b;
b ^= a;
a ^= b;
```

Increment/decrement

This is much slower than the pre/post decrement operator, but a nice way to obfuscate your code ;-)

```
i = -~i; // i++
i = ~-i; // i--
```

Sign flipping using NOT or XOR

Strangely enough, this is 300%(!) faster.

```
i = -i;

//equals
i = ~i + 1;

//or
i = (i ^ -1) + 1;
```

Fast modulo operation using bitwise AND

If the divisor is a power of 2, the modulo (%) operation can be done with:
modulus = numerator & (divisor - 1);
This is about 600% faster.

```
x = 131 % 4;

//equals:
x = 131 & (4 - 1);
```

Check if an integer is even/uneven using bitwise AND

This is 600% faster.

```
isEven = (i % 2) == 0;

//equals:
isEven = (i & 1) == 0;
```

Absolute value

Forget Math.abs() for time critical code. Version 1 is 2500% faster than Math.abs(), and the funky bitwise version 2 is again 20% faster than version 1.

```
//version 1
i = x < 0 ? -x : x;

//version 2
i = (x ^ (x >> 31)) - (x >> 31);
```

Comparing two integers for equal sign

This is 35% faster.

```
eqSign = a * b > 0;

//equals:
eqSign = a ^ b >= 0;
```

Fast color conversion from R5G5B5 to R8G8B8 pixel format using shifts

```
R8 = (R5 << 3) | (R5 >> 2)
G8 = (R5 << 3) | (R5 >> 2)
B8 = (R5 << 3) | (R5 >> 2)
```

- HAXE
- FLASHDEVELOP SPONSOR

- **Categories**
  - Actionscript (64)
  - data structures (16)
  - Games (6)
  - Links (4)
  - Miscellaneous (9)
  - Physics (22)
  - Tools (1)
- **Archives**
  - March 2011
  - February 2011
  - November 2010
  - March 2010
  - February 2010
  - December 2009
  - April 2009
  - March 2009
  - January 2009
  - July 2008

- June 2008
- March 2008
- December 2007
- November 2007
- October 2007
- September 2007
- August 2007
- July 2007
- June 2007
- May 2007
- April 2007
- March 2007
- February 2007
- January 2007
- December 2006
- November 2006
- October 2006
- September 2006
- August 2006
- July 2006
- June 2006

Proudly powered by WordPress Theme based upon Pool theme by Borja Fernandez.
Entries and comments feeds.