

Operations			
operation	example	before	after
NOT	<code>x = ~0111;</code>		1000
AND	<code>x = 0101 & 0011;</code>		0001
OR	<code>x = 0101 0011;</code>		0111
XOR	<code>x = 0101 ^ 0011;</code>		0110
Left Shift	<code>x = 0100 << 1;</code>		1000
Right Shift	<code>x = 0100 >> 1;</code>		0010
Set bit 5	<code>x = (1<<5);</code>	0b00000000	0b00100000
Clear bit 5	<code>x &= ~(1<<5);</code>	0b11111111	0b11011111
Wait until bit 5 is set	<code>while (!(x & (1<<5)));</code>		
Wait until bit 5 is cleared	<code>while (x & (1<<5));</code>		
Save value of bit 5 into variable	<code>int var = x & (1<<5);</code>		
Test if bit 5 is set	<code>if (x & (1<<5)) { ... }</code>		
Toggle bit 5	<code>x ^= (1<<5);</code>	0b00000000	0b00100000
Replace modulo of power of two with AND	<code>x % y == x & (y - 1)</code>	<code>x % 64</code>	<code>x & (63)</code>
Check if integer x is odd	<code>if (x & 1) { ... }</code>		
Turn off the rightmost 1-bit	<code>x = x & (x-1);</code>	0b01011000	0b01010000
Isolate the rightmost 1-bit	<code>x = x & (-x);</code>	0b01110000	0b00010000
Right propagate the rightmost 1-bit	<code>x = x (x-1);</code>	0b10111100	0b10111111
Isolate the rightmost 0-bit	<code>x = ~x & (x+1);</code>	0b01110111	0b00001000
Turn on the rightmost 0-bit.	<code>x = x (x+1);</code>	0b01110111	0b01111111
Right propagate the rightmost 0-bit	<code>x = x & (x+1);</code>	0b01110111	0b01110000
Multiply by 2	<code>x <<= 1;</code>	0b00000010	0b00000100
Divide by 2	<code>x >>= 1;</code>	0b00000010	0b00000001
XOR swap	<code>a ^= b; b ^= a; a ^= b;</code>		
Calculate 2^n	<code>1 << n;</code>		
Convert letter to lowercase	<code>x = (x ' ');</code>	A	a
Convert letter to uppercase	<code>x = (x & '_');</code>	a	A
Swap Nibbles	<code>x = (x << 4) (x >> 4);</code>	0b11110000	0b00001111

