MAKE
SCHOOL

BIT MANIPULATION

Twiddling, Fiddling, Befuddling Bits

BITWISE OPERATIONS

AND	&	NOT	~
OR	|	LEFT SHIFT	<<
XOR	^	RIGHT SHIFT	>>

AND

Only true if both input bits are true

```
0 & 0 = 0
1 & 0 = 0
0 & 1 = 0
1 & 1 = 1
```

MAKE SCHOOL

```
  10010110
&
  00110010
----------
  00010010
```

OR

True if any input bit is true

```
0 | 0 = 0
1 | 0 = 1
0 | 1 = 1
1 | 1 = 1
```

```
  10010110
|
  00110010
----------
  10110110
```

XOR

True if one and only one input bit is true

$$0 \mid 0 = 0$$
$$1 \mid 0 = 1$$
$$0 \mid 1 = 1$$
$$1 \mid 1 = 0$$

```
  10010110
^
  00110010
----------
  10100100
```

NOT

Ones' complement operator

Flips the input bit

$$\sim 0 = 1$$

$$\sim 1 = 0$$

~

```
  00110010
--------
  11001101
```

LEFT SHIFT

Shift the binary digits by n, pad 0's on the right

Each shift is a multiply by 2 (unless there's overflow)

```
  00010110
<<
  00000010
----------
  01011000
```

RIGHT SHIFT

Shift the binary digits by n, pad 0's on the right

Each shift is a divide by 2 with round towards negative infinity

```
  00010110
>>
  00000010
----------
  00000101
```

BIT MANIPULATION BASICS

SET BIT

```
int setBit(int x, unsigned char position) {
  int mask = 1 << position;
  return x | mask;
}
```

x	00000110	00000110
position	00000101	| 00100000
mask	00100000	00100110

CLEAR BIT

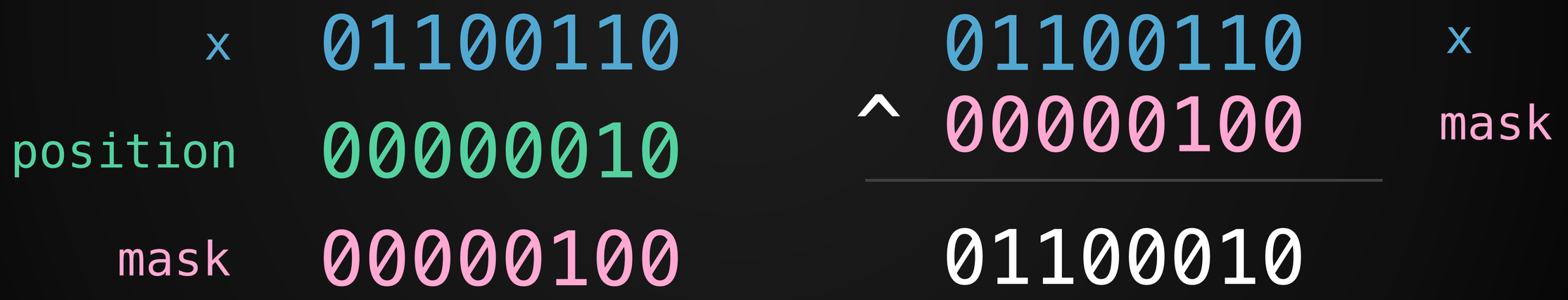
```
int clearBit(int x, unsigned char position) {
  int mask = 1 << position;
  return x & ~mask;
}
```

x	00000110
position	00000010
mask	00000100
~mask	11111011

```
  00000110   x
& 11111011   ~mask
------------
  00000010
```

FLIP BIT

```
int flipBit(int x, unsigned char position) {
  int mask = 1 << position;
  return x ^ mask;
}
```



IS BIT SET

```
bool isBitSet(int x, unsigned char position) {
  int shifted = x >> position
  return shifted & 1
}
```

MODIFY BIT

SET

```
int modifyBit(int x, unsigned char position, int state) {
  int mask = 1 << position;
  return (x & ~mask) | (-state & mask);
}
```

x	00000110
position	00000101
state	00000001
mask	00100000

~mask	11011111
-state	11111111
x & ~mask	00000110
-state & mask	00100000

```
  00000110   x & ~mask
| 00100000   -state & mask
----------
  00100010
```

MODIFY BIT

CLEAR

```
int modifyBit(int x, unsigned char position, int state) {
  int mask = 1 << position;
  return (x & ~mask) | (-state & mask);
}
```

x	00000110
position	00000010
state	00000000
mask	00000100

~mask	11111011
-state	00000000
x & ~mask	00000010
-state & mask	00000000

```
  00000010   x & ~mask
| 00000000   -state & mask
----------
  00000010
```

BIT TRICKS

CHECK IF EVEN

```
((x & 1) == 0)
```

```
  0110
&
  0001
------
  0000
```

MAKE SCHOOL

CHECK IF POWER OF TWO

```
((x & (x-1)) == 0)
```

```
  1000
&
  0111
------
  0000
```

Power of Two

MAKE SCHOOL

EXERCISE

Write a function to count the number of bits that are different between two numbers

RESOURCES

http://bits.stephan-brumme.com/

http://h14s.p5r.org/2012/09/0x5f3759df.html

http://en.wikipedia.org/wiki/Fast_inverse_square_root

MAKE
SCHOOL