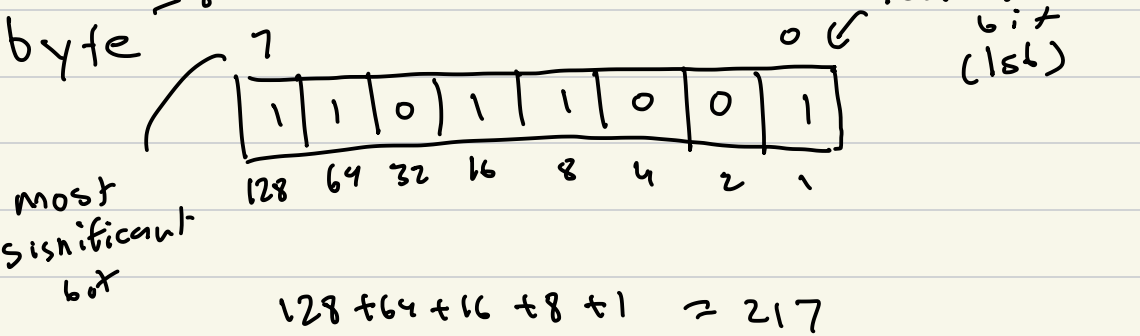
CS 315-01 Lab Bits Machine Code

	bitwise	relational
	>>	
Bit manipulation	&	&&
	|	||
C bitwise operators	~	!
RISC-V bitwise instructions	<<	

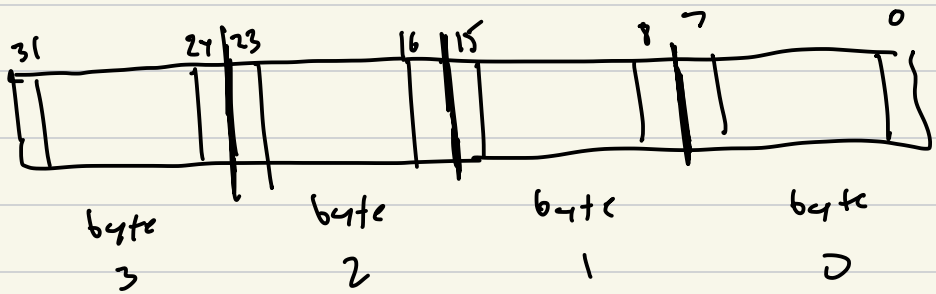
0	1
false	true
off	on
unset	set
low	high

byte – 8 bits



128 + 64 + 16 + 8 + 1 = 217

32 bit value (word)



Bitwise Operators

AND &

a	b	a&b
0	0	0
0	1	0
1	0	0
1	1	1

OR |

a	b	a|b
0	0	0
0	1	1
1	0	1
1	1	1

NOT ~

a	~a
0	1
1	0

XOR ^

a	b	a^b
0	0	0
0	1	1
1	0	1
1	1	0

uint8_t a, b;

a = 0b1100 1010

b = 0b1001 1001

a & b = 1000 1000

128 + 64 + 8 + 2

202

~ a = 0b0011 0101

a | b = 0b 1101 1011

a ^ b = 0b 0101 0011

Shifts

<< left shift

>> right shift

a << n

a >> n

n ↑ # bits to shift

LSL logical shift left

a << 2 = 0b0010 1000

LSR logical shift right

a >> 2 = 0b0011 0010

int8_t c; c = 0b1100 1010 -54

0011 0101

+ 1

0011 0110 = -54

32+16+4+2

-27

ASR c >> 2 = 0b 11 11 0010

arithmetic shift right

```
  0000 1101
+         1
-----------
  0000 1110  = 14
```

Assembly language

and / andi
or / ori
xor / xori
sll / slli
srl / srli
sra / srai

```
uint8_t a = 0b 0000 1100   12
  a >> 1      0b 0000 0110   6
  a >> 1      0b 0000 0011   3
```

int8_t b
-12

```
  1111 0011
+         1
-----------
  1111 0100   -12
```

b = 0b1111 0100 -12

b >> 1 = 0b1111 1010 -6

0000 0101
+ 1
0000 0110 6

---

uint8_t a = 0b1110 0101

shift mask

uint8_t a4 = a >> 2

0b 0011 1001

(a >> 2) & 0b0000 1111

= 0b0000 1001