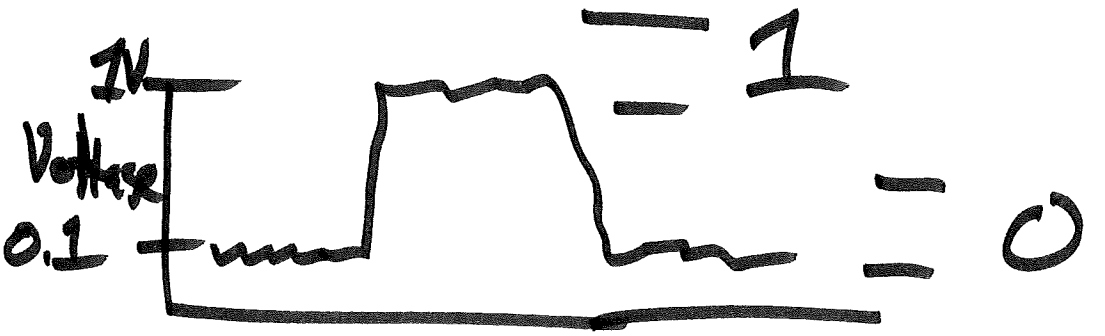
How Data is actually Represented



everything is 1's & 0's

Bytes 8-bits 0000 0000 0x0
to
0-255 (unsigned) 1111 1111 0xFF

Bytes are ~~important~~ important because memory is **BYTE** addressable

Words -> collection of Bytes
- 4-bytes (32 bits)
- 8-bytes (64
- 2-bytes (16

Representing positive ints -> easy

-> negative #'s

-> characters

-> non-integers (Reals)

Simple operations w/ bits

input 0	input 1	AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

```
   a  1  1
+  b  +1
-------
   c  0
```

a	b	c	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

8-bit → ??

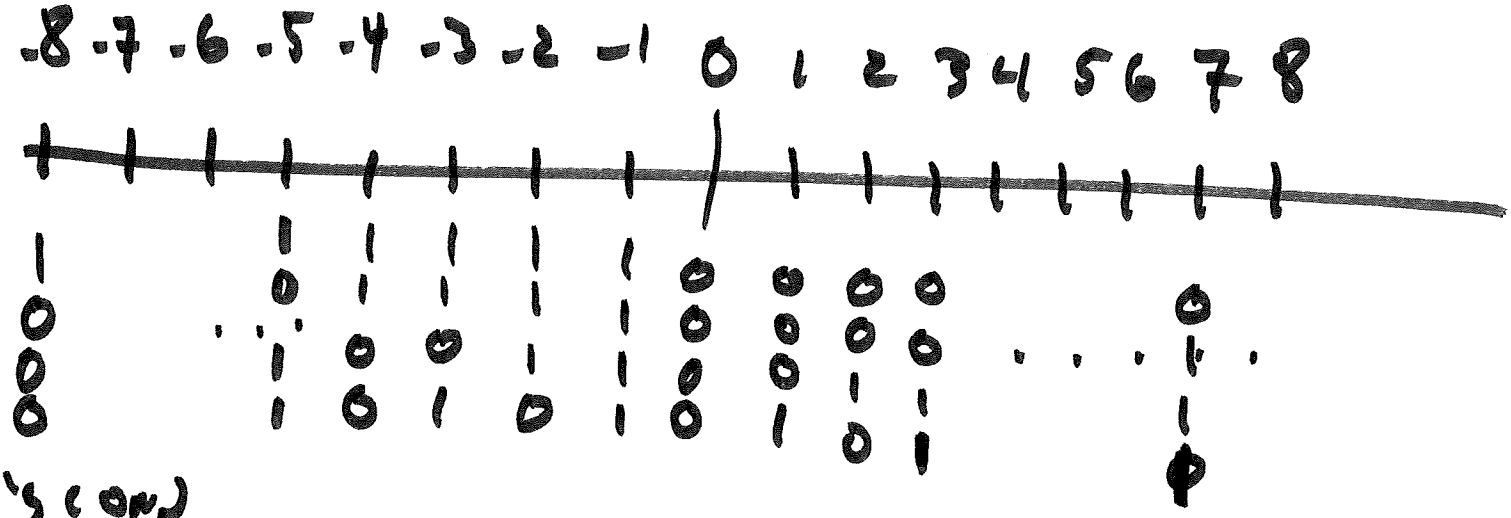
8-bit 1101 1001

\+ 0011 0010

Overflow

8-bits 0000 1011

Negative #'s → 2's complement



2's comp

4-bits [-8, 7]

unsigned

4-bits [0, 15]

-5 1011

\+ 2 + 0010

-3 ← 1101

$$\begin{array}{r} -8 \\ +\ -1 \\ \hline \end{array} \quad \begin{array}{r} 1000 \\ 1111 \\ \hline 0111 \end{array}$$

$$\begin{array}{r} 7 \\ +1 \\ \hline \end{array} \quad \begin{array}{r} 0111 \\ 0001 \\ \hline 1000 \end{array}$$

different = overflow

Subtraction?

$$\begin{array}{r} 7 \\ -\ 3 \\ \hline 4 \end{array} \quad \begin{array}{r} 0111 \\ -\ 0011 \end{array} \quad \rightarrow \quad \begin{array}{r} 0111 \\ +\ 1101 \\ \hline 0100 \end{array} \rightarrow 4$$

Converting from positive to negative:

$$b \rightarrow -b \Rightarrow (\sim b + 1)$$

$$0011 \rightarrow 1101 \Rightarrow \sim 0011 + 1$$

$$1100 + 1$$

$$1101$$