Logic Gates: Circuit Building Blocks

Input: Boolean value(s) (high and low voltages for 1 and 0)  
Output: Boolean value (0 or 1) result of boolean function

And
\[
\begin{array}{ccc}
A & B & \text{out} \\
0 & 0 & 0 \\
0 & 1 & 0 \\
1 & 0 & 0 \\
1 & 1 & 1 \\
\end{array}
\]

Or
\[
\begin{array}{ccc}
A & B & \text{out} \\
0 & 0 & 0 \\
0 & 1 & 1 \\
1 & 0 & 1 \\
1 & 1 & 1 \\
\end{array}
\]

Not
\[
\begin{array}{ccc}
A & \text{out} \\
0 & 1 \\
1 & 0 \\
\end{array}
\]

C’s bit-wise operators:

\&: AND  
\|: OR  
\~: NOT
Sign Extension

• When combining signed values of different sizes expanded smaller to equivalent larger size:

```c
char y = 2, x = -13;
short z = 10;

z = z + y;                z = z + x;
0000000000001010          0000000000000101
+                        +
 000000010          11110011
0000000000000010          1111111111110011
```

Fill in high-order bits with sign-bit value to get same numeric value in larger number of bytes.
Let’s verify that this works

4-bit signed value, sign extend to 8-bits, is it the same value?

0111 --> 0000 0111 obviously still 7
1010 ----> 1111 1010 is this still -6?

-128 + 64 + 32 + 16 + 8 + 0 + 2 + 0 = -6 yes!