Binary Numbers

The number system we usually use (the one you probably learned in elementary school) is called **base 10**, because each digit has *ten* possible values (1, 2, 3, 4, 5, 6, 7, 8, 9, and 0).

But computers don't have digits with ten possible values. They have *bits* with *two* possible values (0 and 1). So they use **base 2** numbers.

Base 10 is also called **decimal**. Base 2 is also called **binary**.

Decimal, or "base-ten" numbers

To understand binary, let's take a closer look at how decimal numbers work. Take the number "101" in decimal:

$$101$$

Notice we have two "1"s here, but they don't *mean* the same thing. The leftmost "1" *means* 100, and the rightmost "1" *means* 1. That's because the leftmost "1" is in the hundreds place, while the rightmost "1" is in the ones place. And the "0" between them is in the tens place.

base 10

	100s	10s	1s
Digits	1	0	1

So this "101" in base 10 is telling us we have "1 hundred, 0 tens, and 1 one."

base 10

	100s	10s	1s
Digits	1	0	1
Digit column values in base 10	100 +	0 +	1
Value in base 10	= 101		

Notice how the *places* in base 10 (ones place, tens place, hundreds place, etc.) are *sequential powers of* 10:

- $10^0 = 1$
- $10^1 = 10$
- $10^2 = 100$
- $10^3 = 1000$
- etc.

Binary, or "base-two" numbers

Just as the *places* in base 10 are sequential powers of 10, **the places in *binary* (base 2) are sequential powers of *2*:**

- $2^0 = 1$
- $2^1 = 2$
- $2^2 = 4$
- $2^3 = 8$
- etc.

So let's take that same "101" but this time let's read it as a *binary* number:

base 2

	4s	2s	1s
Digits	1	0	1

Reading this from right to left: we have a 1 in the ones place, a 0 in the twos place, and a 1 in the fours place. So our total is 4 + 0 + 1 which is 5.

base 2

	4s	2s	1s
Digits	1	0	1
Digit column values in base 10	4 +	0 +	1
Value in base 10	= 5		

Counting to 10 in binary

Here's how we'd count from 0 to 10 in binary, along with how we'd interpret each bit:

Decimal	Binary	Interpretation
0	0000	0 + 0 + 0 + 0
1	0001	0 + 0 + 0 + 1
2	0010	0 + 0 + 2 + 0
3	0011	0 + 0 + 2 + 1
4	0100	0 + 4 + 0 + 0
5	0101	0 + 4 + 0 + 1
6	0110	0 + 4 + 2 + 0
7	0111	0 + 4 + 2 + 1
8	1000	8 + 0 + 0 + 0
9	1001	8 + 0 + 0 + 1
10	1010	8 + 0 + 2 + 0

> Some languages spoken in Nigeria and India use **duodecimal** numbers, or **base-12**. So "eleven" and "twelve" aren't built using 1s and 2s, they're entirely different digits. Some mathematicians argue that base-12 is a better system than our base-10, because 12 has more factors (1, 2, 3, 4, 6) than 10 does (1, 2, 5). We probably use decimal numbers because we have 10 fingers.

Negative Numbers and Two's Complement

Negative numbers are typically represented in binary using *two's complement* encoding. In two's complement, the leftmost digit is *negative*, and the rest of the digits are positive.

Let's look at what happens when we interpret that "101" as two's complement:

2's complement

	-1x4s	2s	1s
Digits	1	0	1
Digit column values in base 10	-4	+ 0	+ 1

Value in base 10 = -3

> Fun computer systems trivia fact: two's complement isn't the only way negative numbers could be encoded. Other encodings tossed around in the 1960s included "one's complement" and "sign and magnitude" encodings. Of the three encodings, two's complement is the one still used today for a few reasons:
>
> 1. There is only one way to represent zero.
> 2. Basic operations like addition, subtraction, and multiplication are the same regardless of whether the numbers involved are positive or negative.
>
> Since two's complement had both of these properties (and the others didn't), it stuck around and is still used today.

Counting from -5 to 5 in two's complement

Here are the base-10 numbers -5 through 5 in two's complement, along with how we'd interpret each bit:

Decimal	Binary	Interpretation
-5	1011	-8 + 0 + 2 + 1
-4	1100	-8 + 4 + 0 + 0
-3	1101	-8 + 4 + 0 + 1
-2	1110	-8 + 4 + 2 + 0
-1	1111	-8 + 4 + 2 + 1
0	0000	0 + 0 + 0 + 0
1	0001	0 + 0 + 0 + 1
2	0010	0 + 0 + 2 + 0
3	0011	0 + 0 + 2 + 1
4	0100	0 + 4 + 0 + 0
5	0101	0 + 4 + 0 + 1

> So, should 1011 be read as "eleven" (in binary) or "negative five" (in two's complement)?
>
> It could be either one! Many languages have two types of numbers: **signed** and **unsigned**. Signed numbers are represented in two's complement, and unsigned numbers use plain old base 2.
>
> So, if an interviewer asks you to convert base-2 into decimal, ask: "is that in two's complement or not?"

← course home

Next up: Bitwise AND →