ECE253: Computer and Digital Systems

QiLin Xue

Fall 2021

Contents

1 Information Representation

- Everything is encoded using binary, using 0 and 1. On a physical level, this could be represented as 0 V and 5 V.
- With two digits, we can represent 4 numbers: $00, 01, 10, 11$ which correspond with $0, 1, 2, 3$.
- We can represent 2^n values using n binary digits.
- **Notation:**
  - To represent that we are expressing a number in binary, we prepend $0b$ to the front.
  - To represent hexadecimal, we prepend $0x$.
  - Another notation is to write a number with a subscript. For example, $(123)_4$ is 27 written in base 4.

> **Idea**: An intuitive way of converting numbers between bases is to understand why we write numbers the way they are in base 10. For example, suppose we have the number 42069. This can be written as
>
> $$42069 = 4 \times 10^4 + 2 \times 10^3 + 0 \times 10^2 + 6 \times 10^1 + 9 \times 10^0 \tag{1}$$
>
> This is however, not unique. We could have instead just written $4206 \times 10^1 + 9 \times 10^0$. The *trick* to finding this *unique* representation is to "fit" the greatest amount of a power of 10, before moving on to the next (smaller) power of 10. If there were 42069 candies, you could represent it by trying to create as many groups of 10^4 as possible, then form as many groups of 10^3 as possible with the remainder, and so forth.
>
> The same is true for other bases, except replace 10 with the other base.

- An algorithm to convert from base 10 to base b is as follows:
  1. Divide the number by b. Let the remainder be b_0
  2. Take the whole part of the quotient, and divide it by b. Let the remainder be b_1.
  3. Repeat the previous step, except let the remainder be $b_2, b_3, \ldots$.
  4. Stop when the quotient is 0.
  5. The number in base b is thus $(b_k \ldots b_3 b_2 b_1 b_0)_b$.
- To convert a number from base b to decimal, we can write them in expanded form. For example:

$$(b_3 b_2 b_1 b_0)_b = b_3 \times b^3 + b_2 \times b^2 + b_1 \times b^1 + b_0 \times b^0 \tag{2}$$

- To convert a number from base b_1 to b_2, you can first convert the number to decimal, then convert it to b_2[1].
- We can create a logic gate that takes in two inputs a and b and gives two outputs, s (for the sum) and c (for the carry). This gives the following truth table

a	b	s	c
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

- we can say that $s = 1$ when $b = 0$ `AND` $a = 1$ `OR` $b = 1$ `AND` $a = 0$.
- We can also say that $c = 1$ if $b = 1$ `AND` $a = 1$.

> **Idea**: We can write this in Verilog (preview)
>
> ```
> module add(a,b,c,s);
>     input a, b;
>     output c, s;
>     assign c = a & b; // & = AND;
>     assign S = (~b & a) | (b & ~a);
> ```

[1] There is another trick explained by bprp here