`Binary Number Systems`

Digital Electronics — Lecture 2

8/23/2023

Intro

- Software (typically from a high level programming langauge), abstracts certain computational functions to make writing code more feasible
- the next step will be the assembly process, written in pure references of transistor processes
- transistor switching

Binary Numer Systems

there are three types of number systems → `Decimal, Binary, Hexadecimal`

Bits

A **bit** is simply a single binary value. These are the followin notations for increases in the number of bits.

- 8 bit → 1 byte
- 16 bits → 1 words, 2 bytes
- 32 bits → 2 words, 4 bytes (floats)
- 64 bits → 2 longs, 4 words, 8 bytes (double in programming)

64 bits are usually called a `long long` in hardware. Additionally, B → bytes 8B is 8 btes, and b → bits, 4b is 4 bits.

2^n, where n is the number os bits, is the number of possible values that could be represented by said number of bits. This entails

- Min and max number a binary
- possible input combination to a digital circuit

Number System

Numbers in everyday use are reffered to as decimal Numbers

$$7,392 = (7 * 10^3) + (3 * 10^2) + (9 * 10^1) + (2 * 10^0)$$

Value of $\# = V_0(B^d) + V_0(B^d) + V_0(B^d)$... where V is the value of a digit, B is the base, and d is the digit's place.

Binary systems are base 2, only compirsed of the value 1 & 0. Hexadecimal an doctal numvers are base 16 and 8 respectively. Hex digits have 16 values, and octal digits have 8 values.

Subscripts to the right most of a digit denotes its base.

- 1101_{10} → base 10
- 1101_2 → base 2

Converting Binary to decimal

Raise the value to the power of it's place. 10011011 $\downarrow [V_0(B^d) + V_0(B^d) + V_0(B^d)...]$

$$2^7 * 1 + 2^6 * 0 + 2^5 * 0 + 2^4 * 1 + 2^3 * 1 + 2^2 * 0 + 2^1 * 1 + 2^0 * 1$$

- `LSB` → least Siginficant Bit
- `MSB` → Most Siginficant Bit

*If the LSB is 1, the decimal number is odd, if 0 it's even.

Converting Decimal to Binary

- Divide decimal number by 2 and remainder is binary value
- continue to divide result by 2 using remainder as binary value
- stop when no more division can occur

10: 10/2 = 0 (LSB), 5/2 = 1, 2/2 = 0, 1/2 = 1

Hexidecimal

Hex is a convenient way of expressing very large Numbers. One digit is 4 bits and the digits range from 0 - F (16 values). A 0x prefix denotes a Hex value (0x90FBE).

Decimal	4-Bit Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Converting between Decimal, Binary, and Hex

- split the Binary number into groups of four, padding any remaining space with 0s
- replace them with their corresponding Hex value (table will be provided during assessments)

101111
0010 1111
2 F
0x2F

Converting from Decimal to Hex is just that process but in reverse, or you can use the `remainder divisor method` with **16** as the base (LSB→MSB).

Given 590

$$\frac{590}{16} = 36\ r\ 14(E),\ \frac{36}{16} = 2\ r\ 4,\ \frac{2}{16} = 0\ r\ 2$$

$$\therefore\ 590_{10} = \text{0x24E}$$

This process reversed yeilds `Hexadecimal` → `Decimal`.

Octal

Octal is a predecesor to Hex, representing 8 possible values (0-7) within each digit. It was useful in the era of data being stored in multiples of 3 (3 bits represent one octal digit).

$$1111 \rightarrow 001\ \ 111 \rightarrow 1\ \ 7 \rightarrow 17_8$$

Decimal	Octal	Binary
0	0	0
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111

→LSB

Given 370

$$\frac{370}{8} = 46\ r\ 2,\ \frac{46}{8} = 5\ r\ 6,\ \frac{5}{8} = 0\ r\ 5$$

$$\therefore\ 370_{10} = 562_8$$