CSD1100

DEC / BIN / OCT / HEX

Vadim Surov

DEC - Decimal numbering system

- Intuitively human beings count to ten because we have ten fingers.
- Therefore our numbering system is based on the number 10 or so called base 10.
- Suppose we have a number 1384.29, we can write its components like this:

$$1384.29 = 1 * 10^3 + 3 * 10^2 + 8 * 10^1 + 4 * 10^0 + 2 * 10^{-1} + 9 * 10^{-2}$$

- Any positive number can be represented as a sum of powers of 10.

DEC - Decimal numbering system

- Operations (+, -, / and *) done on the decimal numbering system somehow follow a procedure, for example the addition table:

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18

DEC - Decimal numbering system

- The addition table showed us how easy and simple is the addition for decimal numbers especially that it follows a pattern. As for the multiplication table:

*	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	10	12	14	16	18
3	0	3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5	0	5	10	15	20	25	30	35	40	45
6	0	6	12	18	24	30	36	42	48	54
7	0	7	14	21	28	35	42	49	56	63
8	0	8	16	24	32	40	48	56	64	72
9	0	9	18	27	36	45	54	63	72	81

OCT - Octal numbering system

- This numbering system is more natural for humans that any other numbering system but what about other beings that have less than ten fingers, birds for example have eight fingers.
- Since they have eight fingers they won’t need to count to ten so their numbering system is called octal numbering system or **base 8**.

OCT - Octal numbering system

- The octal numbering system consists of eight digits, so both digits 8 and 9 are not used in this numbering system.
- The numbers are: 0, 1, 2, 3, 4, 5, 6, 7.
- Number 10 in octal is the number that is after 7.
- If we continue counting we will have 11, 12, 13, 14, 15, 16, 17, 20, 21, …
- To differentiate between a decimal, an octal and other numbering systems we need to use the subscript 10 for the decimal numbers (1234_{10}), the subscript 8 for the octal numbers (1234_8) and so on …

OCT - Octal numbering system

- Suppose we have a number 3725, we can write its components like this:

$$3725 = 3 * 8^3 + 7 * 8^2 + 2 * 8^1 + 5 * 8^0$$

- We can add and multiply octal numbers the same way we add and multiply decimal numbers. For example:

$$\begin{array}{r} 5_8 \\ +\ 7_8 \\ \hline 14_8 \end{array} \qquad \begin{array}{r} 135_8 \\ +\ 643_8 \\ \hline 1000_8 \end{array}$$

OCT - Octal numbering system

$$\begin{array}{r} 5_8 \\ +\ 7_8 \\ \hline 14_8 \end{array} \qquad \begin{array}{r} 135_8 \\ +\ 643_8 \\ \hline 1000_8 \end{array}$$

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

OCT - Octal numbering system

*	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

How about a numbering system that is higher than the decimal, does it exist?

HEX - Hexadecimal numbering system

- The hexadecimal numbering system consists of 16 symbols:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

- Therefore it is also called **base 16**
- Any positive number can be represented as a sum of powers of 16, for example:

$$1A5_{16} = 1 * 16^2 + A * 16^1 + 5 * 16^0$$

- The first 20 Hexadecimal numbers starting from 0 are:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B,

C, D, E, F, 10, 11, 12, and 13.

HEX - Hexadecimal numbering system

- This table shows the meaning of the hexadecimal numbering system in the decimal one.

Hex	Dec
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15

HEX - Hexadecimal numbering system

- Operations on hexadecimal numbers are the same as the operations on decimal numbers including A, B, C, D, E and F digits, for example:

$$\begin{array}{r} 7_{16} \\ +\ 8_{16} \\ \hline F_{16} \end{array} \qquad \begin{array}{r} 1_{16} \\ +\ A_{16} \\ \hline B_{16} \end{array}$$

All possible positional numbering system

- Following the same concept as the previous numbering systems, we can go to higher bases 12, 16, 32, 64… or even go to lower basis like 4, 2.
- However the lowest we can go is the binary numbering system that is represented with only 2 digits.

BIN - Binary numbering system

- Binary digital systems form the basis of just about all hardware systems in existence today.
- The binary numbering system consists of 2 symbols: 0 and 1. Therefore, the **base is 2**.
- In binary we count like this: 0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010 …

BIN - Binary numbering system

- Any positive number can be represented as a sum of powers of 2, for example:

    Example 1:

    $1011_2 = 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0$

    Example 2:

    $100.11_2 = 1*2^2 + 0*2^1 + 0*2^0 + 1*2^{-1} + 1*2^{-2}$

BIN - Binary numbering system

- Operations on binary numbers are the same as the operations on decimal numbers. Here is a decimal and binary table.

- **MSB**: Most Significant Bit
- **LSB**: Least Significant Bit

Decimal	Binary			
0	*0*	*0*	*0*	*0*
1	*0*	*0*	*0*	*1*
2	*0*	*0*	*1*	*0*
3	*0*	*0*	*1*	*1*
4	*0*	*1*	*0*	*0*
5	*0*	*1*	*0*	*1*
6	*0*	*1*	*1*	*0*
7	*0*	*1*	*1*	*1*
8	*1*	*0*	*0*	*0*
9	*1*	*0*	*0*	*1*
10	*1*	*0*	*1*	*0*
11	*1*	*0*	*1*	*1*
12	*1*	*1*	*0*	*0*
13	*1*	*1*	*0*	*1*
14	*1*	*1*	*1*	*0*
15	*1*	*1*	*1*	*1*
	MSB			**LSB**

BIN - Binary numbering system

- In this table we can notice how the digits, in the binary column, alternate going down the column:
- The rightmost digit (LSB) alternates between 0 and 1.
- The next digit, from the right, alternates between two 0s and two 1s.
- And you can deduce the alternation for the rest of the digits.
- Now with this method we can write the next sixteen binary numbers by just repeating the first sixteen and putting a 1 in front.

Decimal	Binary			
0	*0*	*0*	*0*	*0*
1	*0*	*0*	*0*	*1*
2	*0*	*0*	*1*	*0*
3	*0*	*0*	*1*	*1*
4	*0*	*1*	*0*	*0*
5	*0*	*1*	*0*	*1*
6	*0*	*1*	*1*	*0*
7	*0*	*1*	*1*	*1*
8	*1*	*0*	*0*	*0*
9	*1*	*0*	*0*	*1*
10	*1*	*0*	*1*	*0*
11	*1*	*0*	*1*	*1*
12	*1*	*1*	*0*	*0*
13	*1*	*1*	*0*	*1*
14	*1*	*1*	*1*	*0*
15	*1*	*1*	*1*	*1*
	MSB			**LSB**

BIN - Binary numbering system

- Special (useful) values:

  $0000\ 0010_2 = 2^1 = 2$

  $0000\ 0100_2 = 2^1 * 2 = 2^2 = 4$

  $0000\ 1000_2 = 2^2 * 2 = 2^3 = 8$

  $0001\ 0000_2 = 2^3 * 2 = 2^4 = 16$

  $0010\ 0000_2 = 2^4 * 2 = 2^5 = 32$

  $0100\ 0000_2 = 2^5 * 2 = 2^6 = 64$

  $1000\ 0000_2 = 2^6 * 2 = 2^7 = 128$

  $10000\ 0000_2 = 2^7 * 2 = 2^8 = 256$

- A set of 8 bits or one byte can represent 256 numbers from 0 to 255.

BIN. Base 2 And Computers

- The computer system is based on electrical wires.
- The electrical wire can have two states: current is present or current is NOT present.
- Consequently, binary numbers system is the natural numbering system to use with a computer.
- In addition, other electrical devices can be used to represent binary numbers.
  - Such as, light bulbs or switches.
- In the next classes, the basic concepts of electricity are covered in order to demystify how it is used inside the computer systems.

BIN. Addition

- Arithmetic calculation in all positional numbering systems follows the same principles, so the addition table looks like this:

Operands	Sum
0 + 0	0
0 + 1	1
1 + 0	1
1 + 1	**0 (carry 1)**

BIN. Addition (Increment)

- Example 1:

$$01_2 + 01_2 = 10_2$$

Binary Addition		
Carry	1	
	0	1
+	0	1
Sum	1	0

BIN. Addition (Increment)

- Example 2:

$$01_2 + 01_2 + 01_2 = 11_2$$

Binary Addition		
Carry	1	
	0	1
	0	1
+	0	1
Sum	1	1

BIN. Addition

- Example 3:

$$1011101_2+1111001_2=11010110_2$$

Binary Addition								
Carry	1	1	1	1			1	
		1	0	1	1	1	0	1
	+	1	1	1	1	0	0	1
Sum	1	1	0	1	0	1	1	0

OCT in C

- In C an integer digit preceded by zero (ex: 012) is interpreted as an octal number.
- Example (What is the output?):

```
#include <stdio.h>

int main()
{
    int x = 012;
    printf("%d", x);
    return 0;
}
```

OCT in C

- Output is decimal 10 (%d).
- To output as octal ise %o.
- In C an integer digit preceded by 0 (zero) is interpreted as an octal number.
- An octal number for decimal value 10 is 12.
- But what if x equals 092 well then the compiler will show an error because 92 is not a valid octal number.

HEX in C

- In C programming language, a hexadecimal number is represented by preceding with "0x" or "0X", thus the value in hexadecimal can be written as "0x64" (which is equivalent to 100 in decimal).

```c
int main()
{
        unsigned char a=0x64;
        int b=0xFAFA;

        printf("value of a: %X [%x]\n",a,a);
        printf("value of b: %X [%x]\n",b,b);

        return 0;
}
```

```
value of a: 64 [64]
value of b: FAFA [fafa]
```

BIN in C

- Standard C doesn't define binary constants.
- There's a GNU C extension though (among popular compilers, clang adapts it as well): the 0b or 0B prefixes:

```
int foo = 0b1010;
```

What’s next?

- Lab this week:
  - Conversion table 0~255 construction
  - See how Oct and Hex numbers can be used.

- Next week:
  - Conversions