Exercises 1

LLSMF2018\_ELEC

# Shadoks : base-4

The [video](http://www.college-chateaubriand-plancoet.fr/calcul-shadok/) (VF) shows a method that could be used to count with only 4 words using positional notation. (ga=0, bu=1, zo=2, meu=3). Can you translate “BU ZO GA MEU”?

# Computers : base-2

In a computer, all information is stored in systems that can only take two steady state (tension/no tension, open/closed, current/no-current). We represent these two state by symbols called bits: « 0 » and « 1 » ( or « False » and « True »). Using the positional convention, we can represent any number.

# Humans : base-10 and base-16

Human also use positional notation but with 10 symbols. You will notice soon that binary to decimal conversion is not immediate. To simplify transcription of binary words, it is easier to group them by 4 bits and represent them by 16 symbols that correspond to the 16 combination of 4 bits.

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

# Writing convention

To specify the base that was used to represent a number we need a convention. Otherwise, what does 10 represents?

* Decimal numbers are written without prefix: 12
* Binary numbers are written with prefix “0b” : 12 = 0b1100
* Hexadecimal number are written with prefix “0x” : 12 = 0xC

# General Formula

The represented number is equal to the sum of the symbols value times the base raised to the power corresponding to its position. We start numbering position on the right at position 0.

## Examples

**3459 in decimal (base-10)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Position (i) | 3 | 2 | 1 | 0 |
| Symbol (ai) | 3 | 4 | 5 | 9 |
| Weight (bi) | 10^3 = 1000 | 10^2 = 100 | 10^1 = 10 | 10^0 = 1 |
| Weight x symbol (aibi) | 3000 | 400 | 50 | 9 |

3459 = 3000 + 400 + 50 + 9

**0x3A4F (base-16)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Position (i) | 3 | 2 | 1 | 0 |
| Symbol (ai) | 3 | A = 10 | 4 | F = 15 |
| Weight (bi) | 16^3 = 4096 | 16^2 = 256 | 16^1 = 16 | 16^0 = 1 |
| Weight x symbol (aibi) | 4096 x 3 = 12288 | 256 x 10 = 2560 | 16 x 4 = 48 | 1 x 15 = 15 |

0x3A4F = 12288 + 2560 + 48 + 15 = 14911

**0b1101 (base-2)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Position (i) | 3 | 2 | 1 | 0 |
| Symbol (ai) | 1 | 1 | 0 | 1 |
| Weight (bi) | 2^3 = 8 | 2^2 = 4 | 2^1 = 2 | 2^0 = 1 |
| Weight x symbol (aibi) | 8 | 4 | 0 | 1 |

0b1101 = 8 + 4 + 0 + 1 = 13

# Conversions

## From any base to decimal

Use the general formula (see example above)

## Binary to Hexadecimal

We group digit by 4 starting in the right. Then we replace each group by its hexadecimal value (see table).

Example: 0b1010111001 = 0b0010 1011 1001 = 0x2B9

## Hexadecimal to binary

Each hexadecimal symbol can be replaced by its binary representation (see table)

Example: 0x2B9 = 0b0010 1011 1001 = 0b1010111001

## From decimal to any other representation

The process looks like a written division.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Process for any integer x | Example with x = 2131 to be converted in hexadecimal |  |
| 1 | Take the smallest index so that: > x | 161 = 16  162 = 256  163 = 4096   * We start with index i = 2 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **i** | 3 | 2 | 1 | 0 | |  | 0 | ? | ? | ? | |
| 2 | Take biggest so that | 8 x 256 = 2048  9 x 256 = 2304 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **i** | 3 | 2 | 1 | 0 | |  | 0 | 8 | ? | ? | |
| 3 | Replace x by and  repeat from step 1 | x = 2131 – 2048 = 83 |  |
| 1' | Take the smallest index so that: > x | 161<83<162   * i = 1 |  |
| 2’ | Take the biggest so that | 4 x 16 = 64  5 x 16 = 80  6 x 16 = 96 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **i** | 3 | 2 | 1 | 0 | |  | 0 | 8 | 5 | ? | |
| 3’ | Replace x by and  repeat from step 1 | x = 83 – 80 = 3 |  |
| 1’’ | Take the smallest index so that: > x | 3 < 161   * i = 0 |  |
| 2’’ | Take the biggest so that |  | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **i** | 3 | 2 | 1 | 0 | |  | 0 | 8 | 5 | 3 | |

Checking: 0x0853 = 8 x 16^2 + 5 x 16 + 3 = 2131

# Exercises

Fill-in the table. Start with any number in any columns for the empty rows.

|  |  |  |  |
| --- | --- | --- | --- |
| **Décimal** | **Binaire** | **Hexadécial** | **Shadock** |
| 27 |  |  |  |
| 356 |  |  |  |
|  | 0b10011001101 |  |  |
|  | 0b0110011000101 |  |  |
|  |  | 0x2CAFE |  |
|  |  | 0xAA55 |  |
| 52345 |  |  |  |
|  |  |  | BU ZO GA MEU |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Remarque

There are other conventions for negative number, floating point number and characters.

# Logic gates

# 

# Truth table

## Please fill-in the tables





## Bitwise operations

### and

0b01101101 & 0b00001111 =

0xAA & 0xF0 =

176 & 0xF0 =

176 & 240 =

### or

0b01101101 | 0b00001111 =

0xAA | 0xF0 =

### exclusive or

0b01101101 ^ 0b00001111 =

0xAA ^ 0xF0 =

### Interesting properties.

For any x bit:

x & 0 =

x | 1 =

x ^ 0 =

x ^ 1 =

## 