# Homework

1. Convert the following decimal numbers to **6-bit** binary numbers (describe how you have done this).

   Unsigned: `0`, `13`, `24`, `63`.

   Signed: `16`, `-2`, `31`, `-32`.

2. Convert the following **6-bit** values to decimal numbers. Consider both unsigned and two's complement formats (provide a formula).

   Values: `000101`, `101011`, `111111`, `100000`

3. Convert the following decimal values to 8-bit hexadecimal numbers.

   Values: `7`, `240`, `171`, `126`

4. Convert the following hexadecimal numbers to 8-bit binary values:

   Values: `0x3C`, `0x7E`, `0xFF`, `0xA5`

5. Negate the binary values (integer negation) from the previous task.

6. Describe how bytes of the `0xDEADBEEF` value would be located in memory for Big- and Little-Endian convention.

7. Convert the following decimal values to 4-bit binary values. Then sign- and zero-extend them to 8-bit binary values.

   Values: `7`, `15`, `-16`, `-5`

8. Convert the following pairs decimal numbers to 4-bit binaries and add them.

   Values: unsigned ( `7`, '9'), signed ( `4`, `-5` )

# №1

unsigned:

a) 0, as it has unsigned notation

=> 000000

$2^5 \cdot 0 + 2^4 \cdot 0 + 2^3 \cdot 0 + 2^2 \cdot 0 + 2^1 \cdot 0 + 2^0 \cdot 0$

b) 13 = 001101

$2^3 \cdot 1 + 2^2 \cdot 1 + 2^1 \cdot 0 + 2^0 \cdot 1$

c) 24 = 011000

$2^4 \cdot 1 + 2^3$

d) 63 = 111111

$2^0 + 2^1 + 2^2 + 2^3 + 2^4 + 2^5$

Signed:

a) 16 = 010000

cause positive

b) 2 = 000010 => we have to invert it, then we add 1

111101

So, $-2 = 111110$

c) $31 = 011111$

$16+8+4+2+1$

d) $-32 = 100000$, cause

if we have $n$ bits $\Rightarrow$

$-2^{n-1} = 1\underbrace{00\ldots00}_{n-1}$

Formula:

for signed: $-a[5]\sum_{i=0}^{4} 2^i \cdot (1-a[i]) + 1) + (1-a[5]) \cdot \sum_{i=0}^{4} 2^i a[i]$

unsigned: $\sum_{i=0}^{5} 2^i \cdot a[i]$

a) $000101$ (a[5] a[4] a[3] a[2] a[1] a[0]; i: 5 4 3 2 1 0)

$\sqrt{2}$

unsigned: $1\cdot 2^0 + 0\cdot 2^1 + 1\cdot 2^2 + 0\cdot 2^3 + 0\cdot 2^4 + 0\cdot 2^5 = 5$

signed: also 5, cause MSB = 0

b) $101011$ unsigned: $1\cdot 2^0 + 1\cdot 2^1 + 0\cdot 2^2 + 1\cdot 2^3 + 0\cdot 2^4 + 1\cdot 2^5 = 43$

signed: $-\sum_{i=0}^{4} 2^i \cdot (1-a[i]) = -21$

c) 111111 unsigned: 1+2+4+8+16+32 = 63

signed: −1

d) 100000 unsigned: 32

signed: −(1+2+4+8+16) = −31

## ν°3

a) 7 → 00000 111 → 0x07

b) 240 = 128+64+32+16 →

→ 11110000 → 0xF0

c) 171 = 128+32+8+2+1 →

→ 10101011 → 0xAB

d) 126 → 64+32+16+8+4+2 →

→ 01111110 → 0x7E

## ν°4

a) 0x3C → 00111100

3 C

b) 0x7E → 0111 1110
7 E

c) 0xFF → 1111 1111
F F

d) 0xA5 → 1010 0101
A 5

## Nº 5

a) 00 1111 00 —inv→ 1100 0011 —+1→ 1100 0100

b) 0111 1110 —inv→ 1000 0001 —+1→ 1000 0010

c) 1111 1111 —inv→ 0000 0000 —+1→ 0000 0001

d) 1010 0101 —inv→ 0101 1010 —+1→ 0101 1011

## Nº 6

0x DEADBEEF

Big endian

0x00400000 0x00400001 0x00400002 0x00400003

| DE | AD | BE | EF |
|---|---|---|---|

Little endian

0x00400000 0x00400001 0x00400002 0x00400003

| EF | BE | AD | DE |
|---|---|---|---|

# №7

a) 7 : 4 bit : 0111
8 bit : 00000111 – zero and sign same

b) 15 : 4 bit : 1111 – if unsigned
8 bit : 00001111 – zero and sign same

c) -16 ; 4 bit: Actually, requires at least 5 bits, these case:

10000

sign ; 11110000

zero: 00010000

d) -5; 4 bit: 1011

8 bit : sign : 11111011

zero : 00001011

# №8

a) [7, 9)

7 = 0111
9 = 1001

```
  1 1
  0111
+ 1001
 10000
```

– overflow

b) 4 = 0100
-5 = 1011

```
  0100
+ 1011
 1111 = -1
```