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 **5-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`)

Additionally:

1. Provide your explanation for (*) tasks from the class. Bonus point.

Commit the file with the solutions to your private GitHub account. Place it into the folder `ca/lab02`.

№1. $0_{10} = 000000_2$

$13_{10} = 001101_2$

$13 \mid 2 \to 6 \mid 2 \to 3 \mid 2 \to 1 \mid 2 \to 0 \mid 1$

We are dividing by 2 and remember remainder on each step.

then write resulting remainders in reverse order.

$24 = 16 + 8 + 4 + 2 + 1 = 011000_2$

* * * * *

1 1 0 0 0

Another approach is to split the number on 2^n summands and choose 1 or 0 multiplier with them

$63_{10} = 111111_2$ because $64_{10} = 1000000_2$ $63_{10} = 1000000 - 1 = 111111$

Signed: $16_{10} = 010000_2$

$-2_{10} = 111110_2$

$2_{10} = 000010$ ⇒ to get -2 we need to invert bits and add 1

$000010 \xrightarrow{\text{invert}} 111101 \xrightarrow{+1} 111110$

$31_{10} = 32 - 1 = 011111_2$

$-32_{10} = 100000$

$32_{10} = 100000 \to 011111 \to 100000$

W2. unsigned: $000101_2 = 0\cdot2^5 + 0\cdot2^4 + 0\cdot2^3 + 1\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 = 5_{10}$

signed: the same since leading bit is 0

UInt: $101011_2 = 32 + 8 + 2 + 1 = 43_{10}$

Int: $101011_2 = -32 + 8 + 2 + 1 = -21_{10}$

UInt: $111111_2 = 63_{10}$

Int: $111111_2 \rightarrow 000000_2 + 1_2 \rightarrow 000001_2 = 1_{10} \rightarrow -1_{10}$ (abs value)

UInt: $100000_2 = 32_{10}$

Int: $100000_2 = -32_{10}$

W3. $7_{10} = 0x07_{16}$

$240_{10} = 0xF0_{16}$

240 | 16 → 15 | 0

15 | 16 → 0 | F

10 → A, 11 → B, 12 → C, 13 → D, 14 → E, 15 → F

$171_{10} = 0xAB_{16}$

171 | 16 → 10 | B

10 | 16 → 0 | A

$126_{10} = 0x7E_{16}$

126 | 16 → 7 | E

W4. $0x3C = 00111100_2$

3 → 0011, 12 → 1100

$0x7E = 01111111_2$

7 → 0111, 14 → 1111

$0xFF = 11111111_2$

obv. ous

$0xA5 = 01100101_2$

A → 0110, 5 → 0101

W5. to make negative we invert bits and add 1.

$00111100 \rightarrow 11000011 + 1 \rightarrow 11000100$

$01111111 \rightarrow 10000000 + 1 \rightarrow 10000001$

~~$11111111 \rightarrow 00000000 + 1 \rightarrow 00000001$~~

$01100101 \rightarrow 10011010 + 1 \rightarrow 10011011$

W7. $7_{10} = 00111$

Sign-ext: 00000111

Zero-ext: 00000111

$15_{10} = 01111$

Sign-ext: 00001111

Zero-ext: 00001111

$-16_{10} = 10000$

Sign-ext: $11110000 = -16_{10}$

Zero-ext: $00010000 = 16_{10}$

$-5_{10} = 11011$

Sign-ext: $1111\,1011 = -5_{10}$

Zero-ext: $00011011 = 27_{10}$

W8. $7_{10} = 0111_2$

$9_{10} = 1001_2$

$$\begin{array}{r} {\scriptstyle 111\ \ 1} \\ 0111 \\ +\,1001 \\ \hline 10000 \end{array}$$

$4_{10} = 0100_2$

$-5_{10} = 1011$

$$\begin{array}{r} 0100 \\ +\,1011 \\ \hline 1111 = -1_{10} \end{array}$$

$4 - 5 = 4 + (-5)$

W6. 0xDEADBEEF Big Endian:

0x100	0x101	0x102	0x103
DE	AD	BE	EF

Least signif bit has highest addr

Little Endian:

0x100	0x101	0x102	0x103
EF	BE	AD	DE

least signif bit has lowest addr