C programing basics

Lecture 6

Schedule

- Decimal, binary, hex - numeral systems
- Logical operators
- Shifting

Homework solving

Questions ?

Numeral systems

Numeral systems

- In order to express a number, you need a numeral system. The most used numeral system is the decimal. This is the most “human readable” numeral system.

Numeral systems

- Let’s look into detail the number 184 379 in DEC system.
- It consist of **1***10^5 + **8***10^4 + **4***10^3 + **3***10^2 + **6***10^1 + **9***10^0

Let’s look into detail

- We’ve got 3 main component
  - **Number** that is multiple by
  - numeral system base (10), that is in power of
  - the order

Numeral systems

- By logic when the decimal numeral system consist of numeral system base that is equal to 10, therefore the binary numeral system consist of base that is equal to 2.

Binary number

- 0*b*10011001 = **1***2^7 + **0***2^6 + **0***2^5 + **1***2^4 + **1***2^3 + **0***2^2 + **0***2^1 + **1***2^0 =
  1*2^7 + **1***2^4 + **1***2^3 + **1***2^0 =
  0d153

Calculate to DEC

- 0b10101010 = ?
- 0b00001111 = ?
- 0b10100001 = ?
- 0b11001101 = ?
- 0b10001101 = ?

Calculate to DEC

- 0b10101010 = 170
- 0b00001111 = 15
- 0b10100001 = 161
- 0b11001101 = 205
- 0b10001101 = 141

Numeral systems

- Ok, but how to calculate from DEC to BIN ?

Numeral systems

- let’s calculate the number 0d13 to binary.

From DEC to BIN

- Step one: divide the number by 2. Write the residue from back to front and subtract it from the number.
- Step two: do step one until you the number become 0.

- 13 / 2 = 6 + **1** - last number
- 6 / 2 = 3 + **0**
- 3 / 2 = 1 + **1**
- 1 / 2 = 0 + **1** - first number
- ==========> **0d13 = 0b1101**

Calculate to BIN

- 0d128 = ?
- 0d98 = ?
- 0d7 = ?
- 0d111 = ?

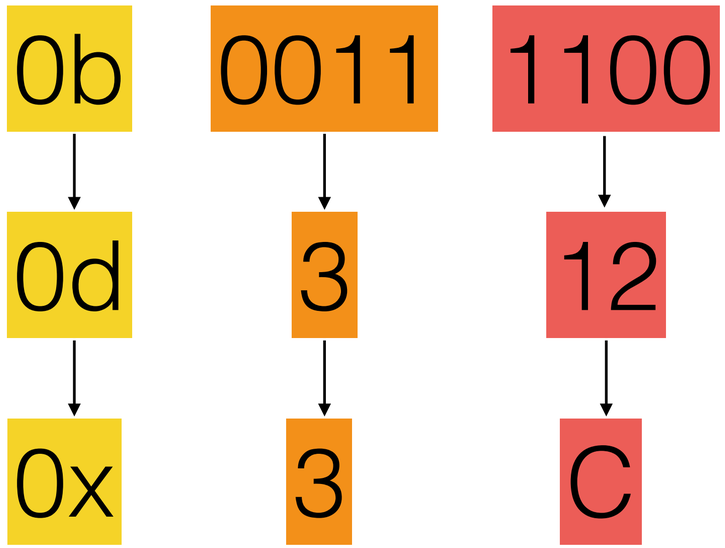
Calculate to BIN

- 0d128 = 0b1000 0000
- 0d98 = 0b0110 0010
- 0d7 = 0b0000 0111
- 0d111 = 0b0110 1111

Numeral systems

- HEX system (hexadecimal)
- The base is 16 (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F)
- Easy way to calculate it is to separate a binary number on equal parts of 4 digits.

Example



Numeral systems

- There are 3, 8, 12 base numeral systems, but they are not commonly used.

Code example

- Decimal to binary calculator

Questions ?

Logical operations

Logical operations

- The logical operations in binary are very similar to the logical operations in the “if” clauses.
- There are few basic operations:
  - AND
  - OR
  - NOT
  - XOR

Logical Operations

- The easiest way to calculate a logical operation is :
  - translate the numbers to binary
  - put the numbers on different rows
  - align to the right
  - calculate bit by bit

Logical operators

- AND - both of the statements must be true
- Symbol - &

a	b	a & b
0	0	0
0	1	0
1	0	0
1	1	1

Logical operations

- number 1 = 0b0011
- number 2 = 0b1001
- Apply AND

num1	0	0	1	1
num2	1	0	0	1
num1&num2	0	0	0	1

Logical operators

- OR - at least one of the statements must be true
- Symbol - |

a	b	a | b
0	0	0
0	1	1
1	0	1
1	1	1

Logical operations

- number 1 = 0b0011
- number 2 = 0b1001
- Apply OR

num1	0	0	1	1
num2	1	0	0	1
num1|num2	1	0	1	1

Logical operators

- NOT - inverts the bits of the number
- Symbol - ~

a	~a
0	1
1	0

Logical operations

- number 1 = 0b0011
- Apply NOT

num1	0	0	1	1
~num1	1	1	0	0

Logical operators

- XOR - the statements must be different
- Symbol - ^

a	b	a^b
0	0	0
0	1	1
1	0	1
1	1	0

Logical operations

- number 1 = 0b0011
- number 2 = 0b1001
- Apply XOR

num1	0	0	1	1
num2	1	0	0	1
num1^num2	1	0	1	0

Questions ?

Bit shifting

Bit shifting

- In low level programing it usually happens to “shift a bit”. This means that the whole binary number must be shifted with certain amount of bits to the left of to the right.

Bit shifting

- Shift right “>>”
- inputNumber = 0b0011
- result = inputNumber>>1 // shift input number by 1 bit right
- result is 0b001

Bit shifting

0b 0011

Shift with one position to the right (>>1)

0b 0011

Result : 0b 001

Bit shifting

- Shift left “<<”
- inputNumber = 0b0011
- result = inputNumber<<3 // shift 3 times to the left
- result is 0b0011000

Bit shifting

0b 0011

Shift with 3 positions to the left (<<3)

0b 0011000

Result : 0b001 1000

LSB, MSB

- LSB (least significant bit) and MSB (most significant bit) are systems for identification if the left bit is the one presenting the highest power of 2 or the lowest.
- Example
- LSB 0b1100 = 3, because the left bit is **least significant**
- MSB 0b1100 = 12, because the left bit is **most significant**

Calculate

- 16>>2 = ?
- 918<<1 = ?
- 48<<3 = ?
- 82>>2 = ?
- 255>>6 = ?

Homework 1

- Make a program that is getting user inputed decimal number and is translating it to hexadecimal and octal numeral systems.
- Example input : 11
- Example output:
  HEX: 0xB
  OCT: 0o13

Homework 2

- Implement four functions that are easily doing multiplication numbers by 2,4,8,16. Names must be : mult2(int number), mult4(int number), mult8(int number), mult16(int number). The multiplication operation should be done without using the multiply operator.