

Introduction to Computing

MCS1101B

Lecture 1

By
Soumadip Biswas

Outline of the Course

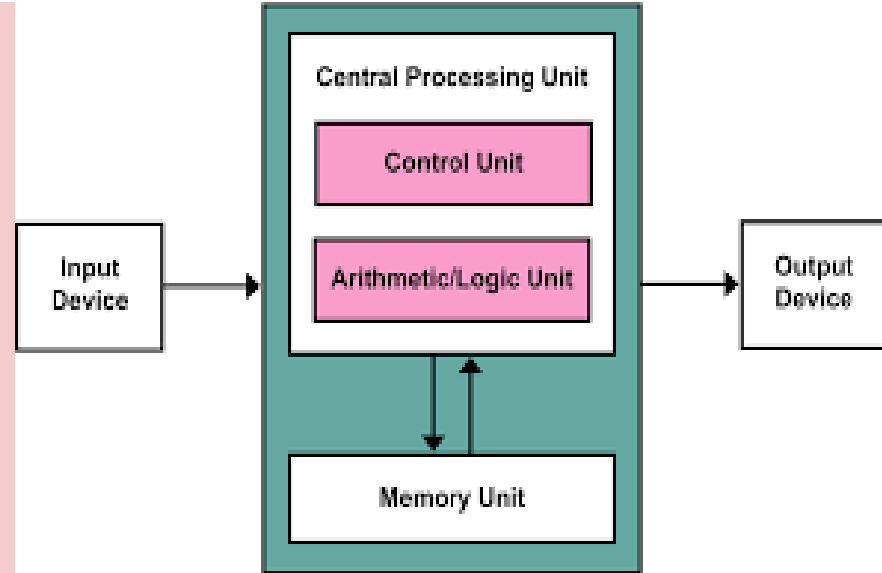
- Broad overview
 - Basic prerequisites of computation
 - Computation using the language **C**
 - Computation using the language **Python**
- Course webpage
 - <https://soumadip.github.io/courses/I2C>
- Reference books
 - Let Us C by Yashavant Kanetkar.
 - The C Programming Language by B. W. Kernighan and D. M. Ritchie.
 - A Book on C: Programming in C by AI Kelley and Ira Pohl

Computation

- What is computation?
 - The typical definition: The action of mathematical calculation
- Why computation is needed?
 - Almost everything requires computation in one form or the other
 - e.g. shopping, grades, rocket science, cellphone...
 - It can take many and any form - from printing a letter to predict who will win the World Cup to determine the number of atoms in the universe to ...
- Properties of computation
 - They are mostly boring monotonous tasks
 - No real thinking is required once you know exactly what formula to use and when to use
- The role of computer programs
 - It is the tool using which we can automate computation

How do Computers Compute?

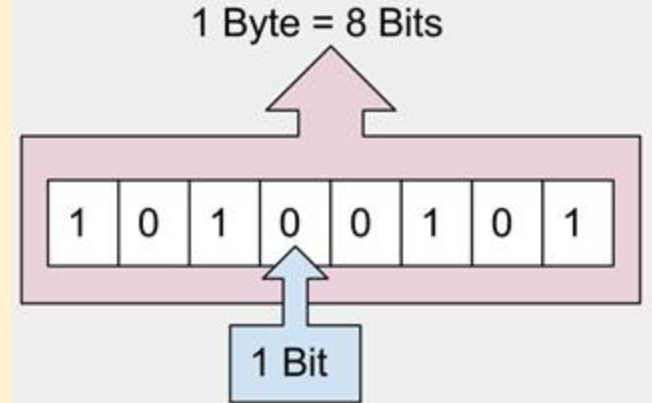
- It uses its brain - similar to humans
 - It's called a CPU (Central Processing Unit)
- Broadly a **typical** CPU consists of
 - ALU (Arithmetic and Logic Unit)
 - CU (control Unit)
 - Note: Modern CPUs are much more advanced and has more components, but let's ignore that for the moment ...
- It needs some input(s)
 - Keyboard, mouse, camera, etc.
- It produces some output(s)
 - Monitor, printer, speaker, etc.



The Von Neumann Architecture

How do Computers Store Data?

- It uses memory - similar to humans
- Different types of memory aka. storage
 - Primary memory - RAM (Random Access Memory)
 - Secondary memory - Hard disk, SSD
 - Other types of storages - ROM (Read Only Memory), Pen-drive, Floppy disk, etc.
- The unit of computer memory
 - It's called a bit
 - It can store either a 0 or a 1
 - 8 bits constitute a byte
 - ...refer to the image →
- Each memory location has an *unique address*



1 byte	= 8 bits
1 kilobyte	= 1024 bytes
1 megabyte	= 1024 kilobyte
1 gigabyte	= 1024 megabyte
1 terabyte	= 1024 gigabyte

What can a Computer store?

- Only 0s and 1s can be stored in computer memory (i.e. bits)
 - So, basically numbers
- But, it can store anything (image, text, videos), right?
 - How does it do it?
 - This is where **files/structures** come into the picture
- Computer interprets sequence of numbers in many different
 - But how do a computer know how to interpret something
 - This is where computer programs come into the picture
 - Subsequently computer programmers follow; that means you, in the near future, hopefully :-)

So, how do Computers Store Numbers?

- The binary number system
 - By using 0 and 1
 - The base (radix) is 2
- Why binary?
- How do you get a binary number?
 - Think how decimal numbers work : $103 = 1 * 10^2 + 0 * 10^1 + 3 * 10^0$
 - So, $103 = 1 * 2^6 + 1 * 2^5 + 0 * 2^4 + 0 * 2^3 + 1 * 2^2 + 1 * 2^1 + 1 * 2^0 = 1100111$
 - $150 = ?$
- So, what is the maximum number a byte can hold?

Binary Operations

- **Arithmetic operations:** Addition, Subtraction, Multiplication, Division
- **Logical operations:** And, Or, Xor, Not
- **Complements:** 1's complement, 2's complement

X	Y	X&Y	X Y	X^Y	~(X)
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

0 1 1 0 1 1 1 0 ← Original binary value

1 0 0 1 0 0 0 1 ← 1's complement

1 0 0 1 0 0 0 1
+ 1
1 0 0 1 0 0 1 0

← 2's complement

Data

- What is data?
 - It can be anything - a number, a character, a set, an image, etc.
 - A **meaningful data** is an information
- For a computer
 - At the smallest, it is a bit
 - Predefined interpretation of collection of bits constitutes different data types
- Basic types of data
 - Integer
 - Real numbers
 - Alphabets
 - Special symbols
 - ... anything else?

Programming and Software

Computer needs to be programmed to do tasks

- **Programming** is the process of writing instructions in a language that can be understood by the computer so that a desired task can be performed by it
- **Program**: sequence of instructions to do a task, computer processes the instructions sequentially one after the other
- **Software**: programs for doing tasks on computers

Computers understand machine language (set of instructions) which are different strings of 0s and 1s only

- They are hard to remember
- So, names are given to these instructions e.g. ADD, START, COPY, etc.

Machine Language and Programming

Instruction Set

- ♦ **Start**
- ♦ **Read M**
- ♦ **Write M**
- ♦ **Load Data, M**
- ♦ **Copy M1, M2**
- ♦ **Add M1, M2, M3**
- ♦ **Sub M1, M2, M3**
- ♦ **Compare M1, M2, M3**
- ♦ **Jump L**
- ♦ **J_Zero M, L**
- ♦ **Halt**

Program

0: Start
1: Load 33, 10
2: Load 24, 11
3: Add 10, 11, 12
4: Halt

Problems with Programming using Instruction Set

- Different CPU can have different instruction sets
 - Need to write same code multiple times
- They are, still, hard to remember
- Solution: *High level programming languages* e.g. C, C++, Java, etc.
 - Does not depend on CPU
 - There is a **compiler** that converts the high level language to low level machine language that computers can understand

Programming Levels

High-Level Program

```
Variables x, y;  
Begin  
Read (x);  
Read (y);  
If (x = y) then Write (x)  
           else Write (y);  
End.
```

Low-Level Program

```
0: Start  
1: Compare 20, 21, 22  
2: J_Zero 22, 5  
3: Write 20  
4: Jump 6  
5: Write 21  
6: Halt
```

Steps of Programming

- **Step 1:** Write the program in a high-level language
 - In your case, **C**
- **Step 2:** Compile the program using a compiler
 - C compiler - **gcc**
- **Step 3:** Run the program
 - i.e. ask the computer to execute it

So, Let Us C

- Make sure you have a laptop with you
 - Or else, you are paired with a friend who has one.
- You need to have a C compiler installed in your computer
 - I prefer **gcc** - The GNU Compiler Collection
 - You can find “how to install gcc” tutorials all over the internet for your operating system
- To write a code
 - You will need any text editor
 - You may choose one with syntax highlighting e.g. sublime text(mac), notepad++(windows), gedit(linux), vi, emacs, ... there are many
- To compile a code
 - You need to open a terminal and execute the following: **gcc** <filename>.c
 - <filename> is basically the name you have given to your text file
 - The .c extension is optional, but you **should always** use it while naming a c file

Anatomy of Programming

- You have a problem to solve
 - You take steps to solve the problem
- What are these steps, really?
 - Represent the problem formally
 - Take a decision
 - Some tasks based on the decision
 - Evaluate outcomes
 - Repeat until problem is solved.

The Customary First C Code

↑ The preprocessor

↑ A function definition

↑ *Start of the function*

↑ *A comment*

↑ *A function call*

↑ *A return value*

↑ *End of the function*

```
#include <stdio.h>
```

```
int main (void)
```

```
{
```

```
    /* my first program in C */
```

```
    printf ("Hello, World! \n");
```

```
    return 0;
```

```
}
```

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    /* my first program in C */
```

```
    printf("Hello, ");
```

```
    printf("World! \n");
```

```
    return 0;
```

```
}
```

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int a;
```

```
    a = 10;
```

```
    printf ("%d\n", a);
```

```
    return 0;
```

```
}
```

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int a, b;
```

```
    a = 10;
```

```
    b = 20;
```

```
    printf ("%d %d\n", a, b);
```

```
    return 0;
```

```
}
```

```
#include <stdio.h>

int main()
{
    int a = 10, b = 20, x;
    printf ("a = %d, b = %d\n", a, b);

    if (a > b)
    {
        x = a;
    }
    else
    {
        x = b;
    }
    printf ("The larger value is %d\n", x);
    return 0;
}
```

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    float c, f;
```

```
    f = 212;
```

```
    c = 5 * (f - 32) / 9;
```

```
    printf ("Fahrenheit value %f - Celsius value is %f \n", c,
```

```
f);
```

```
}
```

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    float c, f;
```

```
    scanf ("%f", &f); //take a keyboard input
```

```
    c = 5 * (f - 32) / 9;
```

```
    printf ("Fahrenheit value %f - Celsius value is %f \n", c,  
f);
```

```
}
```

Structure of a C Program

- They are a collection of functions
- Exactly one special function called “**main**” which must be present
- Each function has statements
 - e.g. declaration, assignment, condition check, looping
 - Statements are executed one by one

The C Character Set

- A-Z
- a-z
- 0-9
- Special Characters

! # % ^ & * () - _ + = ~ [] \ | ; : ' " { } , . < > / ? blank

A C program should not contain anything else.

Things One Might Use in C Programming

- Variables
- Constants
- Expressions
 - Arithmetic, Logical, Assignment
- Statements
 - Declaration, Assignment,
 - Control Structures - conditional branching, looping
- Arrays
- Pointers
- Functions
- Structures

Variables

- Very important concept for programming
- An entity that has a value and is known to the program by a name
- Can store any temporary result while executing a program
- Can have only one value assigned to it at any given time during the execution of the program
- The value of a variable can be changed during the execution of the program

Variables (contd.)

- Variables stored in memory
- Remember that memory is a list of storage locations, each having a unique address
- A variable is like a bin
 - The contents of the bin is the value of the variable
 - The variable name is used to refer to the value of the variable
 - A variable is mapped to a location of the memory, called its address

Constants (Read-only variables)

- Sometimes you need to have some values that remain the same throughout a program
 - e.g. universal constants, limits, ranges of data, etc.
- In that case you can use a constant type indicator to enforce that property
- Prevents accidental change of the value

Data Types of Variables or Constants

- Each variable has a type,
 - It indicates what type of values the variable can hold
- Four common data types in C
 - `int` - can store integers (usually 4 bytes)
 - `float` - can store floating point numbers (usually 4 bytes)
 - `double` - can store floating point numbers (usually 8 bytes)
 - `char` - can store a character (1 byte)
- A keyword called `const` is used to declare a read-only variable

Variable Declaration

- Must declare a variable (specify its type and name) before using it anywhere in your program
- All variable declarations should be at the beginning of the `main()` or other functions
- A value can also be assigned to a variable at the time the variable is declared.
 - `int speed = 30;`
 - `char flag = 'y';`

Variable Naming

- Sequence of letters and digits
- First character must be a letter or '_' No special characters other than '_'
- No blank in between
- Names are case-sensitive (max and Max are two different names)
- Examples of valid names:
 - i rank1 MAX max Min class_rank
- Examples of invalid names:
 - a's fact rec 2sqroot class,rank

C Keywords

- Used by the C language, cannot be used as variable names
- Examples:
 - int, float, char, double, main, if else, for, while. do, struct, union, typedef, enum, void, return, signed, unsigned, case, break, sizeof, ...
 - There are others, see textbook

In The Next Class...

- You will learn more about printf and scanf functions
- You will learn more about operators
 - Logical operators
 - arithmetic operators
 - Special operators
- You will learn about expressions
- You will learn more about control structures
 - Conditional branching
 - Looping
- You will be introduced to pointers and array