Introduction to Computing

Michael Nowak

Texas A&M University

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

Basics of Computers

- When discussing computers, typical to break the topic into two parts:
  - Hardware
  - Software

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

Hardware

- To understand software, it is helpful to at least an elementary grasp of hardware
- The major components of a computer include:
  - Central processing unit (CPU)
  - Memory
  - Input/ouput devices

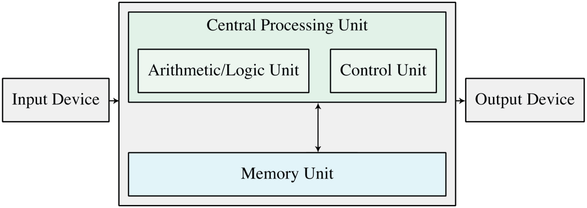
Overview

- Basics of Computers
    - Hardware
        - Central processing unit(CPU)
        - Instruction sets
        - Memory
        - I/O devices
    - Software
- Nature of programming
    - Machine language
    - Assembly language
    - Higher-level languages
- C++ Compilation
- References

Central processing unit (CPU)

- Modern computers work by regulating the flow of electricity through wires
  - Many of those wires are tiny elements that have been etched into silicon
  - The voltage on those wires is used to indicate the state of a bit
  - The wires connect up transistors that are laid out in a way that allows logical processing
- A modern computer processor can include billions of transistors; we will look at things from a much higher level, ignoring the individual wires and transistors.

von Neumann architecture



- In general, modern computers are built with minor modifications of the von Neumann architecture
- John von Neumann's idea was that programs for a computer are nothing more than data and can be stored in the same place as all other data
  - There is a single memory that stores both the programs and the data used by the program
  - This memory is connected to a central processing unit by a bus

Central processing unit

- Program steps (instructions) are to be stored in the computer memory alongside data
- During each computation cycle, the machine will retrieve the next step from memory
- And subsequently execute the computation associated with the retrieved instruction
- The fetch-execute cycle then continues until the machine is told to 'halt'

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

Instruction sets

- The actual things that the computer hardware can do is specified in the instruction set
- Provide limited and primitive facilities, such as
  - loading a register from memory
  - storing the contents of a register to memory
  - moving to a different part of the program
  - shifting the bits
  - arithmetic operations
  - logical operations
- Differences in instruction sets help explain why some programs run on one machine but not another

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

Memory

- The memory unit is used to store program instructions and data
- The byte is the measure of computer memory; most computers offer 'byte addressability'
  - In a 32-bit machine, each byte can be uniquely addressed
  - This allows us to read or update values store at each byte individually
- The basic operations on memory are 'fetching', 'loading', 'reading', and 'writing'

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

I/O devices

- For a computer to perform computation, it needs to get input in the form of instructions and data
  - Devices that provide such capability are called input devices
  - Examples include a keyboard, mouse, and secondary storage
  - The keyboard is the *standard input*
- Frequently we wish to output the results of computation
  - Devices that provide such capability are called output devices
  - Examples include the printer, terminal window, and secondary storage
  - The terminal screen is the *standard output*

Overview

Basics of Computers

- Hardware
  - Central processing unit(CPU)
  - Instruction sets
  - Memory
  - I/O devices
- Software

Nature of programming

- Machine language
- Assembly language
- Higher-level languages

C++ Compilation

References

Software

- Software are the programs that run on the hardware
- Like hardware, can be seen as having multiple components:
  - The *BIOS (basic input/oputput system)* is the base layer that provides computer initial instructions for what to do when powered on
  - *Operating system* is responsible for controlling the operations of the machine, how the user interacts with it, reading/writing files to disk, and loading and starting other programs
  - *Application and utility programs* are those that the user runs, such as your email client or web browser

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

Nature of programming

- Every piece of software is written by a programmer, but
  - what is programming, and
  - how do we do it?
- At the fundamental level, during each cycle, the computer loads an instruction and executes it

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

Machine language

- Each instruction is encoded as a binary sequence of numbers; the language of these instructions is known as *machine language*
- For instance, using the MIPs machine language, we could write the equation `wage = rate * hours` as:

```
100011 00000 00010 0000000000000000   # Load rate, register 2
100011 00001 00011 0000000000000000   # Load hours, register 3
000000 00010 00011 00100 00000 011000 # Multiply registers 2 and 3;
                                        store the result in register 4
101011 00100 00101 0000000000000000   # Store value of register 4
```

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

Assembly language

- Assembly language has an assembly instruction for each machine language instruction
- Unlike machine language, assembly language is entered as mnemonics (i.e., words) that describe what they do
- For instance, we could write the equation `wage = rate * hours` as:

```
lw    $s0 ,  $s2 ,  0
lw    $s1 ,  $s3 ,  0
mult  $s2 ,  $s3 ,  $s4
sw    $s4 ,  $s5 ,  0
```

- In order for the assembly language to be understood by the computer, we use an *assembler* to translate from assembly language to machine language

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

Higher-level languages

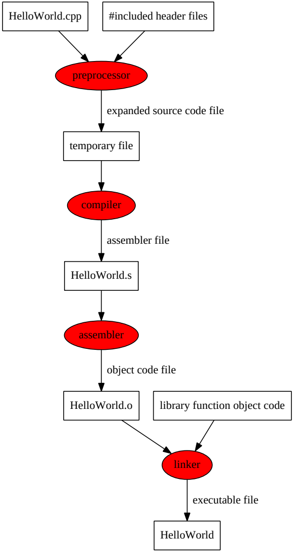
- It is hard for a programmer to express ideas in machine language and assembly language
- Higher-level languages use more complete mnemonics and allow more complex organization of ideas
- In C++, provided that `wage` had been *declared*, and `rate` and `hours` had been *defined*, we could simply write the following *statement* in our program:

```
wage = rate * hours;
```

Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

C++ Compilation Processes



Overview

- Basics of Computers
  - Hardware
    - Central processing unit(CPU)
    - Instruction sets
    - Memory
    - I/O devices
  - Software
- Nature of programming
  - Machine language
  - Assembly language
  - Higher-level languages
- C++ Compilation
- References

References

- Lewis, M. C. (2015). *Introduction to the art of programming using Scala*. CRC Press.
- Stroustrup, B. (2014). *Programming: principles and practice using C++* (2nd ed.). Pearson Education.