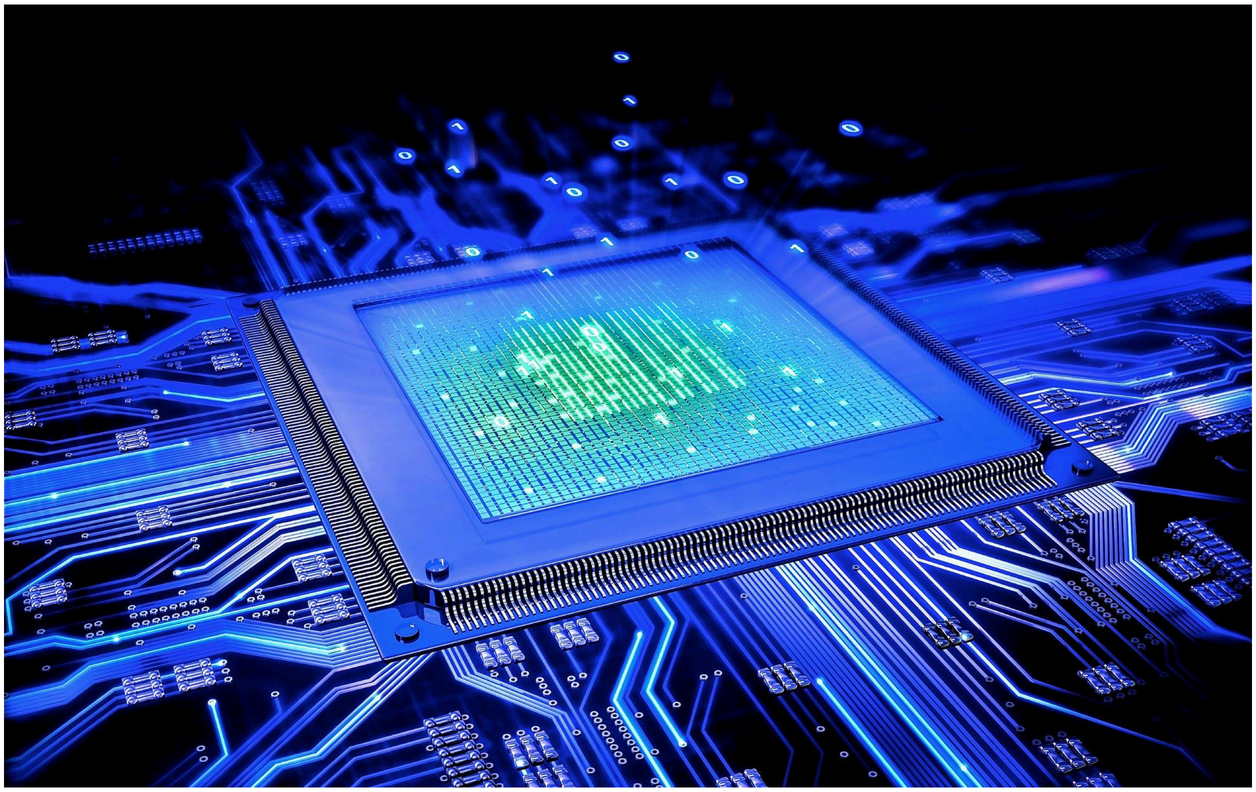
Object Oriented Programming



Assignment Title: Object-Oriented Computer Assembly Framework

Section: E

Fatima Ishtiaq, 23i-0696

Table of content

1. Introduction

The goal of this project is to create a C++ program that simulates the assembly of a computer system. The program offers the user the option to select either a **PC** or **MAC** system and subsequently assembles the respective system by creating and initializing various hardware components such as CPU, GPU, motherboard, ports, memory, and others. The assembly process is displayed through relevant messages that indicate successful creation of these components.

---

2. Project Overview

The user is prompted to choose between two types of systems:

- **PC**
- **MAC**

Based on the user's selection, the program creates corresponding components like:

- **CPU**: Either an Intel/AMD CPU for PCs or an Apple Silicon CPU for MACs.
- **GPU**: A generic GPU object.
- **Memory**: Physical and Main Memory components.
- **Ports**: Ports such as VGI, I/O, USB, and HDMI are created and configured with a baud rate.
- **Motherboard**: A motherboard object that houses the previously created components.
- **Computer Assembly**: The final step assembles all the components into a functional computer system.

The program makes use of multiple classes, such as `IntelAMDCPU`, `AppleSiliconCPU`, `MotherBoard`, `Port`, `NetworkCard`, `HDD`, `PowerSupply`, and `Battery`, to simulate the individual computer parts.

---

3. Key Components

The primary components involved in this project are:

3.1 CPU Classes

- **IntelAMDCPU**: Represents a generic CPU from Intel or AMD for PCs.
- **AppleSiliconCPU**: Represents Apple's proprietary Silicon CPU for MACs.

3.2 Memory

- **PhysicalMemory**: This class represents physical memory (RAM) used by the computer.
- **MainMemory**: Represents a specific configuration of the main memory.

3.3 Port Configuration

- Ports such as **VGI**, **I/O**, **USB**, and **HDMI** are used to connect various peripherals to the computer. Each port is assigned a baud rate and a type, representing the specific connection it supports.

3.4 Motherboard

- The motherboard serves as the backbone of the computer system, connecting all the components such as CPU, memory, and ports. It holds references to the various ports and components it supports.

3.5 Computer Assembly

- After all the components are initialized, a final computer assembly object is created, which combines all the hardware components into a final, functional computer system.

---

4. Program Flow

1. **User Input:** The program prompts the user to choose between two options:
    - Option 1: PC
    - Option 2: MAC
2. The user input is validated, and if an invalid option is entered, the program prompts the user again.
3. **PC Assembly:** If the user selects option 1 (PC), the program proceeds with the creation of a PC:
    - **CPU**: An `IntelAMDCPU` object is created.
    - **Ports**: Ports like VGI, I/O, USB, and HDMI are created and configured with appropriate baud rates.
    - **Motherboard**: A `MotherBoard` object is created with the configured ports and memory.
    - **Computer Assembly**: A final `ComputerAssembly_PC` object is created, which ties all the components together into a functional PC.
4. **MAC Assembly:** If the user selects option 2 (MAC), the program follows a similar flow:
    - **CPU**: An `AppleSiliconCPU` object is created.
    - **Ports**: Ports like VGI, I/O, USB, and HDMI are created and configured.
    - **Motherboard**: A `MotherBoard` object is created similarly to the PC.
    - **Computer Assembly**: A `ComputerAssembly_Mac` object is created, combining all the components into a functional MAC.

5. **Final Display:** After assembling the computer system, the program displays the success messages and details of the assembled computer system.

---

5. Code Structure

The program is divided into various classes and methods:

- **`ComputerComponents.h`**: The header file includes the declarations of the classes used in the program, such as `IntelAMDCPU`, `AppleSiliconCPU`, `MotherBoard`, `Port`, `PhysicalMemory`, `HDD`, and `NetworkCard`.
- **`Main.cpp`**: This file contains the main program logic, where the user is prompted to choose between PC or MAC, and the computer assembly process is executed.
- **Component Classes**: Each component, such as CPU, memory, ports, and motherboard, is implemented in separate classes, each handling the initialization and display of relevant attributes.

---

6. Enhancements and Suggestions

- **Code Refactoring**: The code that initializes ports and motherboards is repeated for both the PC and MAC. Refactoring this code into a separate function could reduce redundancy and make the program more maintainable.
- **Input Validation**: While the program validates user input for selecting PC or MAC, more robust validation could be added to ensure the user cannot enter any non-integer or invalid inputs.

---

7. Conclusion

The program successfully simulates the assembly of a computer system, allowing the user to choose between a PC or MAC configuration. By utilizing object-oriented principles, the program is structured in a modular and easy-to-understand manner. The creation and initialization of various computer components are displayed, providing an informative view of the internal process of computer assembly.

This project serves as a foundation for understanding object-oriented programming in C++, and it can be further extended to add more features, such as supporting additional components, custom configurations, or even integrating a graphical interface for easier interaction.