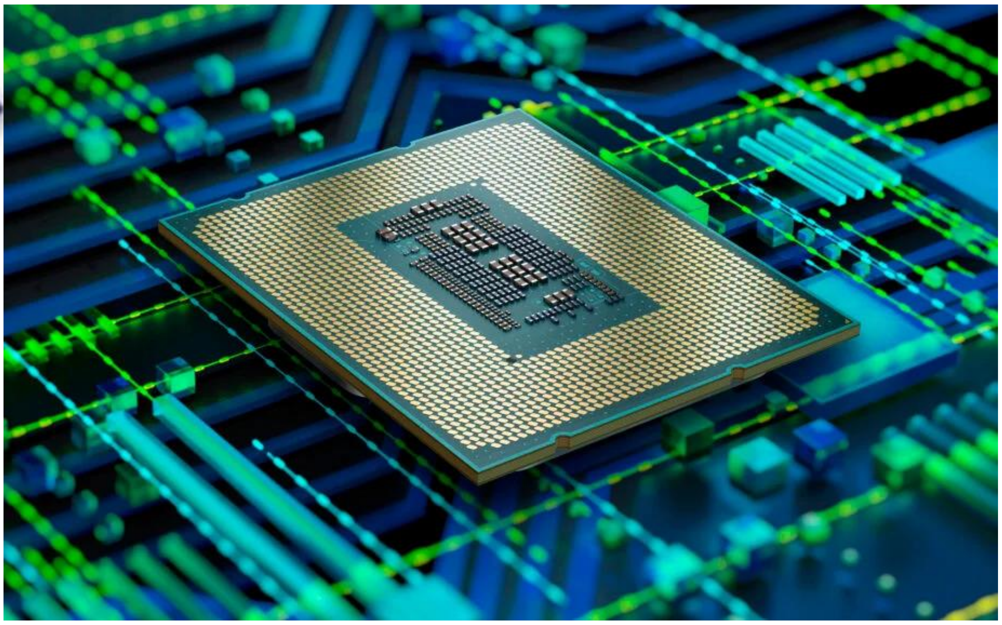

Processor Execution Simulator Assignment

Instructor:

Fahed Jubair

Done by:

Ghassan Yaseen

Introduction

This report aims to build a CPU simulator that manages all the tasks inside each processor.

System

UML diagram

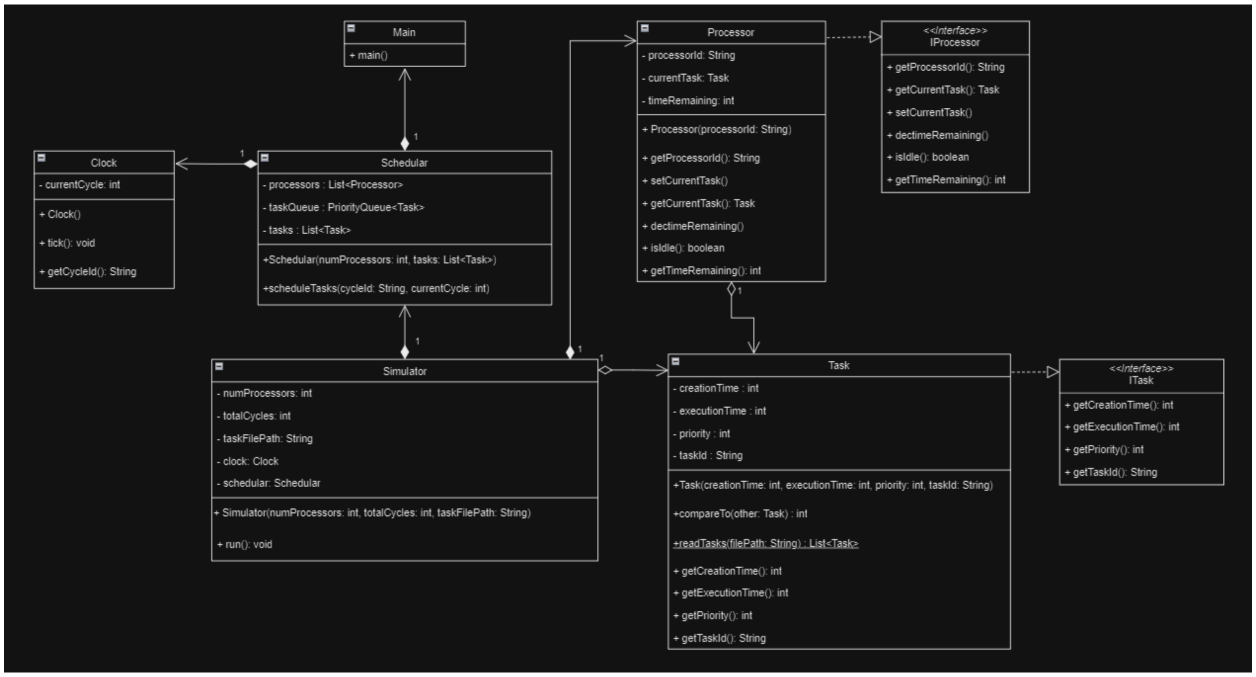

Figure 1 – UML diagram.

I tried to let the code have a high cohesion and the least possible coupling between the classes.

Logic

Modules

This section aims to explain the modules for the system.

Task

This module represents the tasks that the system will create and manage, it implements 'Comparable<Task>', implements ITask interface and each task has the following attributes:

- creationTime
- executionTime
- priority

- taskId

and the following methods:

- +Task (creationTime: int, executionTime: int, priority: int, taskId: String)
- +compareTo (other: Task): int
  - This method is used to activate the PriorityQueue.
- +readTasks (filePath: String): List<Task> {static}
  - This method is used for reading the tasks information from .txt file path in arg [3] and throw an error if the input of the task is wrong.
- + getCreationTime (): int
- + getExecutionTime (): int
- + getPriority (): int
- + getTaskId (): String

Scheduler

This module represents the part that will assign the tasks in the queue to the idle processors, and it contains the following attributes:

- processors: List<Processor>
  - This list contains all the processors in the system depending on the number of processors.
- taskQueue: PriorityQueue<Task>
  - This PriorityQueue arranges all the tasks depending on the priority.
- tasks: List<Task>
  - This list contains all the tasks in the system.

And the following methods:

- +Schedular (numProcessors: int, tasks: List<Task>)
  - Add the processors to the List.
  - Take copy from the tasks.
- +scheduleTasks (cycleId: String, currentCycle: int)
  - Execute the Cycles (cycle1, cycle2, cycle3, ...) every Cycle alone.
  - Manage all the tasks and the processors.
  - Print everything that we need.

Processor

This module represents the part that process the tasks implements IProcessor and it contains the following attributes:

- - processorId
- - currentTask
- - timeRemaining

And it has the following methods:

- + Processor ()
- + getCurrentTask (): Task
- + getProcessorId (): String
- + setCurrentTask ()
- + dectimeRemaining ()
- + isIdle (): boolean
- + getTimeRemaining(): int

Clock

This module represents the periodic clock in the system, it has only the one attribute, and it has the following methods:

- tick ()
  - This method sleeps for one second.
- getCycleId (): String
  - It is return the number of the Cycle.

Simulator

This module represents the box that contains the simulator, and it enables the user to use it as a black box, it has no attributes, and it has the following methods:

- run ()
  - It has a loop that manages and executes the work for every Cycle.

Main

This class contains the main method, it takes the information from the arg and checks if the user enters a wrong input and calls the run method.