OS Simulator

Table of Contents

Contents

[1 Design Approach 3](#_Toc507166503)

[2 Implementation Modules 5](#_Toc507166504)

[3 Central Processing Unit 5](#_Toc507166505)

[4 CPU Module 5](#_Toc507166506)

# 

# Design Approach

At first, I gathered some information from the given information that the professor provided, and did a little bit of research as to how to design the OS simulator. I read through all of the specifications for phase 1 and finally came up with a design for the system.

Disk

Driver

MMU

Long Term Scheduler

Loader

Short Term Scheduler

CPU

Execute

Decode

Fetch

RAM

The main components that were identified in the phase 1 were the CPU, the short term scheduler, the long term scheduler, the MMU, the RAM, the Loader, the Disk, and the Driver. The main component that controls every aspect of the system and work’s as the engine of the system is the driver. From there they were other parts of the system that we needed to do which includes taking instructions from the instructions set and loads them onto the disk through the loader component. From there the disk that contains all of the information about the instructions set will be use by the long term scheduler and jobs will be loaded to the RAM. After jobs have been loaded to RAM, also known as the MMU gathers all of the jobs needed to be scheduled by the short term scheduler which then in turn utilizes the dispatcher to send jobs to the CPU. The CPU cycles through three parts of the system to fetch, decode, and execute until it has run out of instructions at which point , the OS terminates.

For the multiprocessing portion of the OS, I look more closely on the support for multiple cores, memory management among the processors, and the cache of each processor. In our design approach we took note of this and decided to implement the CPU module to handle all of these items as single processor and the using threads to handle multiple cores. Phase 2 of the project I redesign the approach to the CPU. The general architecture idea hasn’t really changed from phase 1. The Major changes came in the form of separating the Dispatcher into its own individual unit and placing it between the ShortTermScheduler and CPU modules, paging systems, and the cache.

adsad

Long Term Scheduler

RAM

MMU

Dispatcher

Fetch

Decode

Execute

CPU

Short Term Scheduler

Driver

Loader

Disk

RAM

# Implementation Modules

The modules below have been structured and implemented in C# programming language running on various hardware components. Each module has its own individual functions and properties that contribute to the system as a whole just as in a non-simulated system. Below can be found the implemented modules and a description of their major functionalities as pertaining of the system.

# Central Processing Unit

# CPU Module

As in phase 1, the CPU contains several key methods integral to the OS. The Fetch method integrates with the disk and acquires the information for the next instruction and increases the program counter value by 1, so as to retrieve the next instruction to be decoded upon next fetch call. The Decoder converts the hexadecimal instruction set into 32 bit binary sequence, seperates the instructions into different property types, and stages each for proper system execution. The execute method uses a switch loop to perform action based on what is has received as being the decoded instructions set. Upon successful completion of the execution method will then increment the program counter to the next necessary location. The CPU module contains the effective address method under the title of read(), which handles all the address transalation. The CPU module contains the Effective Address method under the title of read(), which handles all the address translation. One of the main differences of the module of 1 and 2 is that one of them can be cloned and several CPU modules can be emulated at once to emulate multicore processing in our simulated OS. The OS simulator is capable of N-CPU and has been tested at up to 2048 emulated cores, however, due to the size limitation of our RAM, only 12 CPUs are used based on the size of the jobs given in the instructions set.