Real-time Operating system - 48450

Assignment 3 Report

Ashwin Rajesh - 14259321

Contents

Introduction

This assignment consists of two programs. Program 1 aims to simulate a round-robin CPU scheduling algorithm on 7 processes, with differing arrival times and burst times. The program then sends average wait and turnaround times to a different thread using a named pipe. Program 2 implements a FIFO page scheduling algorithm, and waits for a SIGINT interrupt from the user before outputting the total number of page faults from the algorithm. These results are summarised to show the strengths and weaknesses of algorithms for CPU and page scheduling in the realm of operating systems.

Program 1

Outline

Round Robin scheduling allocates each task an equal share of CPU time. At any given time, a process is selected and allowed to run on the CPU for a specified time quantum. After this time has passed, the task is 'pre-empted', and stopped mid-execution, after which the algorithm context-switches to a different process. This cycle is repeated until all processes in the ready-queue finish.

Once waiting times and turnaround times are calculated for each process, the averages are sent to thread 2 using a named pipe or FIFO. A named pipe is a method of inter-process communication in Unix systems, which is an extension to the traditional pipe concept. A named pipe can last as long as the system is up, beyond the life of the process.

Gantt Chart

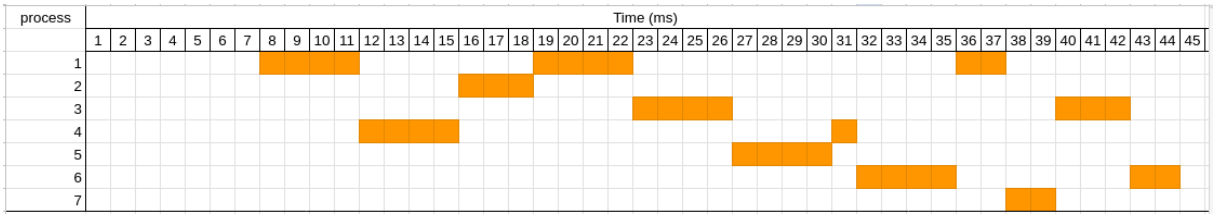

Figure 1: Round Robin Scheduling GANTT chart

Implementation

Round Robin Scheduling (Thread 1)

The round robin algorithm presented in this assignment firstly initiates a ready queue as well as a current time variable. The current time variable keeps track of the time that has passed since the beginning of the algorithm. The ready queue is the data structure that holds all the processes that have arrived and are ready to be run. When a process has the same arrival time as the current time, it is added into the ready queue.

Output to file (Thread 2)

Results

Program 2

Outline

Implementation

Memory management

Signal (Ctrl-C)

Results

Conclusion