Operating System - Homework Assignment #3

Department of Political Economy

B096060041 YEN TE LEE

政經113 李彥德

1. Explain what memory-mapped I/O is and how it works.

Memory-mapped I/O is a method use the same address space to address both memory and I/O devices. What memory-mapped I/O using the ports of I/O map onto the memory directly. If error happens when input and output data, it could read memory for execution directly.

1. Explain what DMA is and how it works.

一張含有 圖表 的圖片

自動產生的描述

Direct Memory Access (DMA) is a process the relieve the burden of CPU when doing I/O operations. CPU will waste lots of time for read and write operations when access a device. How does the DMA work? Initially, CPU will send the data it reads to DMA and call the disk controller to put the data to the buffer zone, then CPU can execute other system calls and procedures. Secondly, DMA asks disk controller to send the data in buffer zone to the main memory and return a signal back. If mission is done, then DMA sends an interrupt to CPU for who can continuously execute the other procedures.

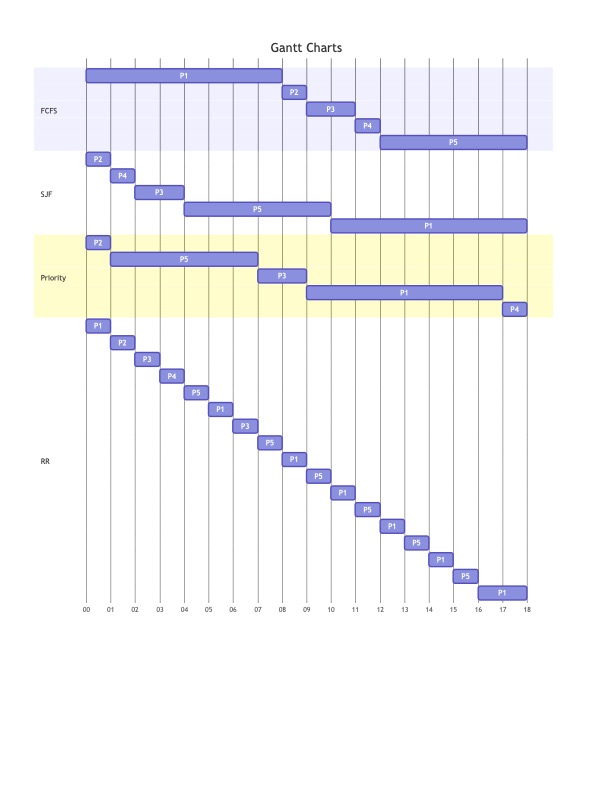
1. Consider the following set of process, with the length of the CPU-burst time given in milliseconds:

The processes are assumed to have arrived in the order *P1*, *P2*, *P3*, *P4*, *P5*, all at time 0.

|  |  |  |
| --- | --- | --- |
| *Process* | *Burst Time* | *Priority* |
| *P1* | *8* | *4* |
| *P2* | *1* | *1* |
| *P3* | *2* | *3* |
| *P4* | *1* | *5* |
| *P5* | *6* | *2* |

* 1. Draw four Gantt charts illustrating the execution of these process using FCFS, SJF, a non-preemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 1) scheduling.

Using mermaid in Markdown for drawing.



* 1. What is the turnaround time of each process for each of the scheduling algorithms in part 3a?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | FCFS | SJF | Priority | RR |
| *P1* | 8 | 18 | 17 | 18 |
| *P2* | 9 | 1 | 1 | 2 |
| *P3* | 11 | 4 | 9 | 7 |
| *P4* | 12 | 2 | 18 | 4 |
| *P5* | 18 | 10 | 7 | 16 |

* 1. What is the waiting time of each process for each of the scheduling algorithms in part 3a?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | FCFS | SJF | Priority | RR |
| *P1* | 0 | 10 | 9 | 10 |
| *P2* | 8 | 0 | 0 | 1 |
| *P3* | 9 | 2 | 7 | 5 |
| *P4* | 11 | 1 | 17 | 3 |
| *P5* | 12 | 17 | 34 | 29 |
| *Total* | 40 | 17 | 34 | 29 |

* 1. Which of the schedulers in part 3a results in the minimal waiting time (over all process)?

It’s SJF, whose waiting time is 17.

1. A UNIX process has two parts－the user part and the kernel part. Is the Kernel part like a subroutine and a coroutine? Why?

It’s more like a subroutine. The process in the user space invokes the kernel space who starts at the same place.