作業系統 HW3 B075020033謝朋潔

1. memory-mapped I/O是I/O與memory共用記憶體空間，因為I/O controller暫存器的位址為memory位址的一部分，則不需要特別指令來處理I/O，存取指令本身與存取memory的指令相同(資料匯流排與資料記憶體共用)，故系統只須提供單一指令，即可存取Memory跟I/O Device。

將I/O port或memory mapping到memory address 當要寫入或讀取I/O por時，直接讀取即可。可以把I/O存取直接當成存取記憶體來用，但mapping到的區域原則上就不能放真正的記憶體。

1. DMA為Block-Transfer I/O device，且提供一個Device controller並且負責memory與I/O 之間的資料傳輸，傳輸過程中不需 CPU 的參與和監督。

DMA 的運作：

(1)CPU將讀取的資料傳給DMA，叫Disk controller把資料放進buffer，之後CPU就可以執行其他程序

(2)DMA要求Disk controller把buffer中的資料傳到memory

(3)Disk controller把buffer中的資料傳入memory

(4)Disk controller完成之後，回傳確認訊號給DMA

(5)DMA傳interrupt給CPU，讓CPU知道可以繼續執行剛剛的程序

(a)

➢ FCFS

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Time | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Process | P1 | | | | | | | | P2 | P3 | | P4 | P5 | | | | | |

➢ SJF

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Time | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Process | P2 | P4 | P3 | | P5 | | | | | | P1 | | | | | | | |

➢ Non-preemptive priority

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Time | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Process | P2 | P5 | | | | | | P3 | | P1 | | | | | | | | P4 |

➢ RR

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Time | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Process | P1 | P2 | P3 | P4 | P5 | P1 | P3 | P5 | P1 | P5 | P1 | P5 | P1 | P5 | P1 | P5 | P1 | |

(b)

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

(c)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | P1 | P2 | P3 | P4 | P5 |
| FCFS | 0 | 8 | 9 | 11 | 12 |
| SJF | 10 | 0 | 2 | 1 | 4 |
| Non-preemptive priority | 9 | 0 | 7 | 17 | 1 |
| RR | 10 | 1 | 5 | 3 | 10 |

(d)

FCFS：0 + 8 + 9 + 11 + 12 = 40 (單位時間)

SJF：10 + 0 + 2 + 1 + 4 = 17 (單位時間)

Non-preemptive priority：9 + 0 + 7 + 17 + 1 = 34 (單位時間)

RR：10 + 1 + 5 + 3 + 10 = 29 (單位時間)

最小的等待時間為 SJF

1. Subroutine：順序方式的執行次序

Coroutine：跳躍方式的執行次序

User part會呼叫fork來執行其他指令，像是呼叫函式，會從原有的呼叫點，跳到另一個地方。

Kernel part是被user part所呼叫，只能單獨從被呼叫的點依序執行，之後又傳回到原來所呼叫的點給user part，因此Kernel part屬於 Subroutine。