HW11

408410094 葉X勛

硬體環境: intel i5-8265U 1.6GHz(8CPUs), ~1.8GHz

軟體環境: WSL

CPU速度: 1.6GHz~1.8GHz

Linux設定: sched_latency_ns設成100000,sched_min_granularity_ns設成100000

1.解釋： sched_latency_ns 和 sched_min_granularity_ns

(1) sched_latency_ns: 表示一個運行隊列所有process運行一次的週期。

(2) sched_min_granularity_ns: 表示process最少需要執行的時間，以防止一直context switch，對於交互系统而言（如桌面），該值可以設比較小，這樣可以保證得到更快的response。

2.設計實驗，說明 context switch 的次數與效能的關係

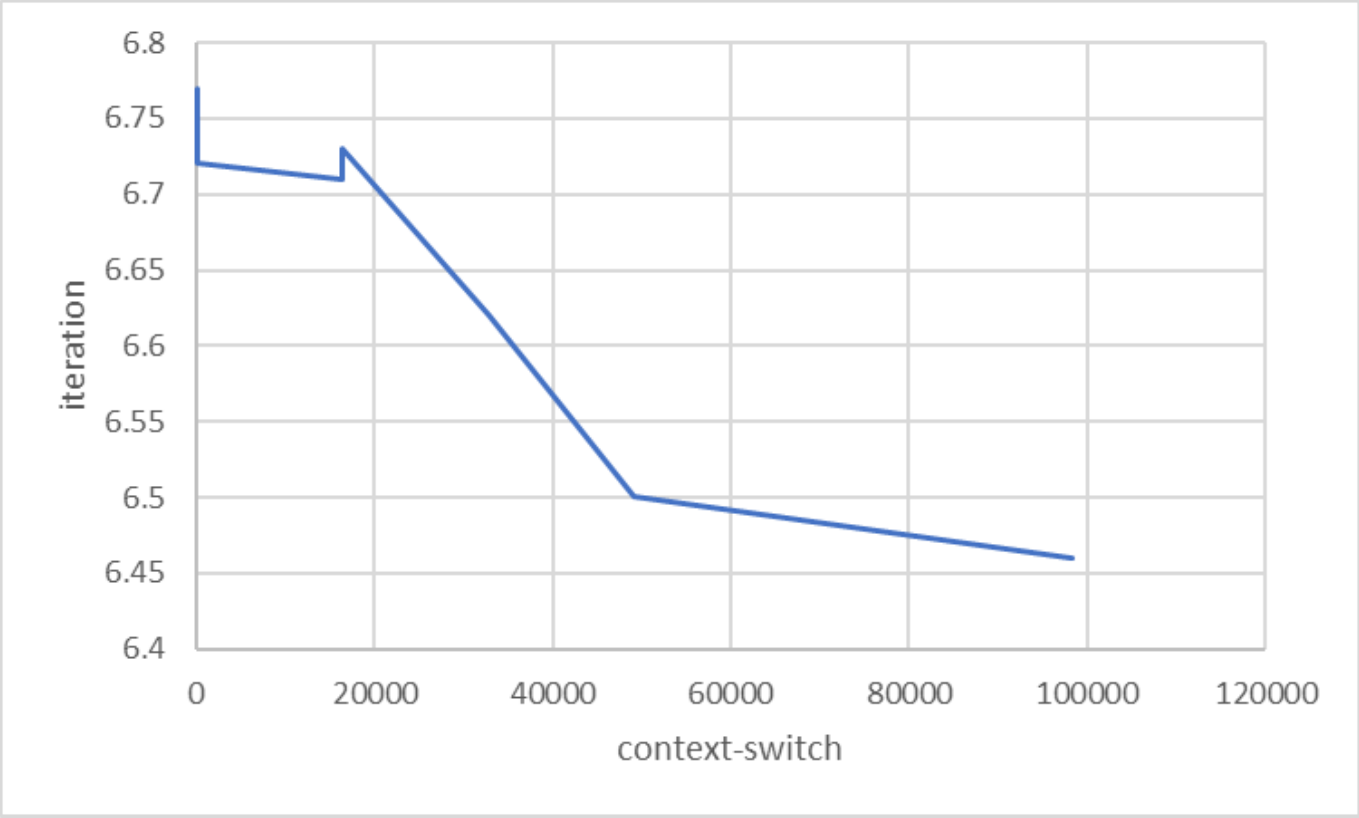
實驗: 改變cpu.c中randAcess()內的迴圈發生sleep()的條件: 當iteration%n = 0時就會執行sleep()，進而產生自願性的context-switch，而此實驗就是要去更改n的值來觀察context-switch跟iteration之間的關係。

執行方式: ./reportChildStat ./cpu

實驗數據:

n	10000	20000	30000	40000	50000	60000	70000	80000	90000	100000
context-switch	98318	49167	32784	16402	16401	16401	18	18	18	18
iteration	6.46	6.5	6.62	6.73	6.71	6.71	6.72	6.74	6.75	6.77

實驗結果:



結果解釋: 當n從10000往上加時，context-switch次數會從98318慢慢往下降，一開始n跟context-switch次數會成反比，因而造成iteration次數上升。當n為40000~70000時，context-switch次數幾乎一樣，故圖表中有一小部分為一小段垂直線。到後來由於n太大了導致sleep次數太少，因此幾乎沒有context-switch。根據圖表，可以發現當context-switch越多時，iteraton次數越少，顯示了效能的下降。

預估時間: 計算後發現每個context-switch需要大約7.75*10^-5秒，所以我預估在context-switch的次數達到46451612次時，執行時間會超過1小時。