

## 設計:

利用函數 `int proc_exec( struct process proc )` 執行傳入參數的 process。

開始與暫停一個 **process**，我們透過設定該 **process** 的 **priority** 來實現這兩個功能。

主要是呼叫 `sched_setscheduler()` 函數來達成，其中，使用 `sched_setscheduler()` 時，需查閱 `int policy` 相關規定去設定 `struct sched_param` 的 `priority` 數值。例如 `SCHED_FIFO` 的 `priority` 介於 1 到 99 之間。

## 核心版本:

```
linux-4.14.25
```

## 比較實際結果與理論結果，並解釋造成差異的原因:

依照我們預期的，FIFO應該是要先入先出，先進來的process會做完才跑第二進來的process，但是在我們的實驗中，發現似乎FIFO會在執行第一個process還沒完全執行完的時候，會執行後面的其他process，如圖所示：

```
bill@mj-VirtualBox:~/Downloads/2020_05_proj$ sudo ./demo < data/FIFO_1.txt | tee
output/FIFO_1_stdout.txt
[sudo] password for bill:
P1 4583
P2 4584
P3 4585
P4 4586
P5 4587
bill@mj-VirtualBox:~/Downloads/2020_05_proj$ sudo dmesg | grep Project1
[ 530.067598] [Project1] 3092 1588173227.378105043 1588173228.538746043
[ 530.443724] [Project1] 3093 1588173228.538908404 1588173228.914870994
[ 531.886496] [Project1] 3094 1588173228.880161643 1588173230.357643938
[ 532.284300] [Project1] 3096 1588173230.363524974 1588173230.755448576
[ 532.987843] [Project1] 3095 1588173230.357762719 1588173231.458991576
[ 819.506900] [Project1] 4583 1588173517.787185363 1588173517.978048745
[ 819.709893] [Project1] 4584 1588173517.787657182 1588173518.181041686
[ 819.898767] [Project1] 4585 1588173517.787407614 1588173518.369915031
[ 820.092533] [Project1] 4586 1588173517.809231287 1588173518.563681336
[ 820.269230] [Project1] 4587 1588173517.809387637 1588173518.740378486
bill@mj-VirtualBox:~/Downloads/2020_05_proj$
```

我覺得原因可能是，`sched_setscheduler`是利用更改 `priority` 來選擇誰先執行，但是在設定`priority` 的時候可能沒有考慮到一些邊界效應，導致可能會有第二進入的process，在第一個還沒執行完畢的時候，又在執行。